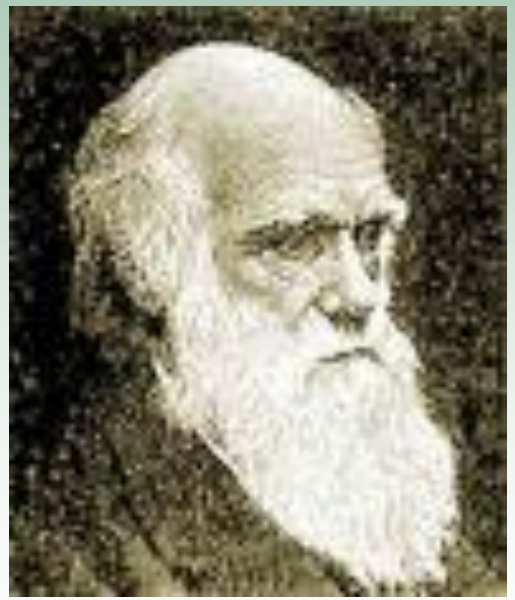
بسم الله الرحمن الرحيم

تاریخچه الگوریتم ژنتیک

الگوریتم های ژنتیک با توجه به نظریه داروین در مورد تکامل به وجود آمدند.



تاریخچه الگوریتم ژنتیک

در سال ۱۹۷۵ دانشمندی در دانشگاه میشیگان به نام John Holland ایده استفاده از الگوریتم ژنتیک را در بهینه‌سازی‌های مهندسی مطرح کرد.

ایده اساسی الگوریتم ژنتیک انتقال خصوصیات موروثی توسط ژنهاست.

نظریه تکامل

در جهان طبیعت موجودات دائماً در حال مبارزه برای زنده ماندن و فرار از دست دشمنان خود بوده و نیز برای بقا وکسب غذا نیاز به غلبه بر مکانیسمهای دفاعی دیگر گونه‌های حیاتی بوده تا آنها را به عنوان غذا طعمه خود قرار دهند. هر گونه‌ای از موجودات اعم از گیاهی و جانوری برای نیل به موارد بالا به ابزارهای خاصی مسلح هستند که در صورت ناکارآمد بودن این ابزارها در مقابل تهدیدات خارجی، حیات و بقای آن گونه خاص از موجودات با خطر جدی مواجه می‌شود. اما در موارد بسیاری ، ارگانیسمهای زنده به نحوی خود را با شرایط دشواری که بقای گونه آنها را تهدید می‌کرد، کنار آمده اند و در طول چندین نسل موفق شده اند تا با تغییر وجهش ژنتیکی در اندام خود، به زندگی ادامه بدهند.

مفاهیم الگوریتم ژنتیک

ژن (Gene)

ژن ها خصوصیات هر فرد را کد می کنند .هر ژن می‌تواند مقادیر متعددی داشته باشد که فرم می نامند مثلاً برای تعیین رنگ چشم در انسان یک ژن وجود دارد که این ژن می تواند دارای فرم‌های سیاه ، آبی ، سبز و .... باشد.

فرم (Allele)

حالت‌های مختلف هر ژن

کروموزم (Chromosome)

به گروهی از ژن‌ها اطلاق می‌شود که اطلاعات وراثتی را از نسلی به نسل دیگر انتقال می‌دهند. جواب هر مسئله به عنوان یک کروموزوم در نظر گرفته می شود و سپس حل مسئله با یک تعداد تصادفی از این کروموزوم‌ها شروع می شود .

مفاهیم الگوریتم ژنتیک

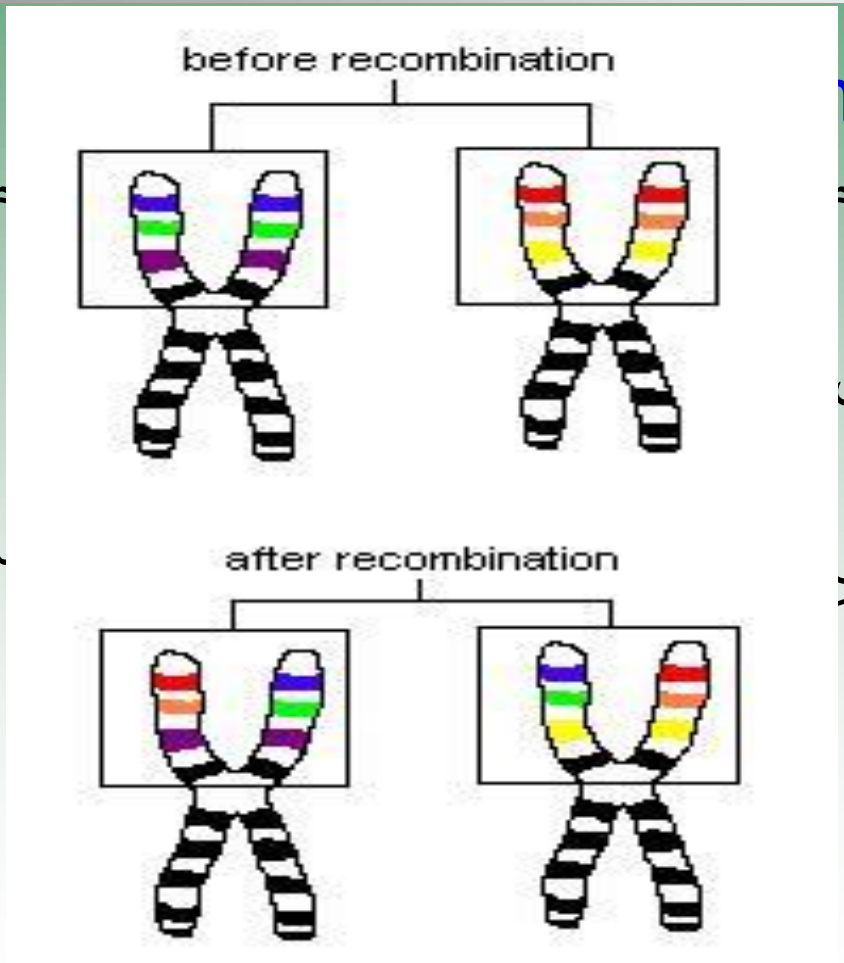
تابع برازش (Fitness Function)

شرایط محیطی را که جمعیت در آن قرار دارد و به صورت یک رابطهٔ ریاضی درآمده تابع برازش می‌نامند. برای ارزیابی کروموزوم‌ها از این تابع استفاده می‌شود.

انتخاب (Selection)

عمل حذف در شرایط طبیعی باعث نابودی افرادی که سازگاری کمتری با محیط دارند می‌شود؛ به عبارت ساده‌تر نتیجه مبارزه بین جانداران باقی ماندن افراد سازگارتر می‌باشد. جمعیت نسل جدید با انتخاب کروموزوم‌های برازنده تر از نسل فعلی به وجود می‌آیند.

مفاهیم الگوریتم ژنتیک



برش (Crossover) یا (
تبادل ویژگی های کروموزو
جهش (mutation)
جانشین شدن تصادفی ژنی
کد گذاری (Encoding)
الگوریتم ژنتیک به جای این
شکل کد شده آنها سرو

م‌های جدید

اله کار کند، با

مفاهیم الگوریتم ژنتیک

جمعیت اولیه (population)

پس از تعیین سیستم کدینگ و مشخص شدن روش تبدیل هر جواب به کروموزوم، باید یک جمعیت اولیه از کروموزوم‌ها تولید نمود. در اکثر موارد، جمعیت اولیه به صورت تصادفی تولید می‌شود. اما گاهی اوقات برای بالا بردن سرعت و کیفیت الگوریتم از روش‌های ابتکاری نیز برای تولید جمعیت اولیه استفاده می‌گردد.

موقعیت (locus)

هر ژن در یک موقعیت خاص از کروموزوم، قرار می گیرد. این مکان، locus نامیده می‌شود.

نگاشت بین طبیعت و کامپیوتر

طبیعت	کامپیوتر
جمعیت	مجموعه ای از جواب ها
فرد	جواب یک مساله
برازندگی	کیفیت یک جواب
کروموزوم	کدگذاری برای یک جواب
ژن	قسمتی از کدگذاری یک جواب.
توالید	برش

مفاهیم الگوریتم ژنتیک

در مورد ساختار کلی الگوریتم ژنتیک قبل از هر چیز باید مکانیسمی برای نمایش هر جواب مساله به صورت یک کروموزوم تعریف نمود، سپس مجموعه‌ای از کروموزوم‌ها را که در حقیقت بیانگر یک مجموعه از جواب‌های مساله هستند، به عنوان جمعیت اولیه در نظر گرفت. اندازه جمعیت دلخواه بوده و توسط کاربر تعیین می‌شود.

بعد از این مرحله باید با بکارگیری اعمال ژنتیک اقدام به تولید کروموزوم‌های جدید، موسوم به نوزاد(offspring) نمود. پس از تولید تعدادی کروموزوم جدید باید به انتخاب برازنده‌ترین کروموزوم‌ها پرداخت به طوری که تعداد کروموزوم‌های منتخب برابر با اندازه جمعیت اولیه باشد.

را

مفاهیم الگوریتم ژنتیک

فرایند انتخاب مبتنی بر مقدار برازندگی هر رشته بوده که اغلب تابع برازش را برابر با همان تابع هدف مساله بهینه‌سازی درنظر می‌گیرند الگوریتم بعد از طی چندین نسل به تدریج به سمت جواب بهینه همگرا می‌شود.

. شرط توقف مساله می‌تواند طی کردن تعداد معینی تکرار بوده که از قبل توسط کاربر تعیین شده است،یا عدم تغییر در چند تکرار مشخص از الگوریتم و یا شرط خاص دیگری باشد.

پارامترهای GA

یک الگوریتم GA دارای پارامترهای زیر است:

GA(Fitness, Fitness_threshold, p, r, m)

- Fitness :تابعی برای ارزیابی یک جواب که مقداری عددی به هر جواب نسبت می‌دهد.
- Fitness_threshold:مقدار آستانه که شرط پایان را معین می‌کند.
- p :تعداد جواب هائی که باید در جمعیت در نظر گرفته شوند.
- r: درصدی از جمعیت که در هر مرحله توسط الگوریتم crossover جایگزین می‌شوند.
- m:نرخ mutation

یک الگوریتم ژنتیک ساده

- Initialize: جمعیت را با تعداد p جواب بطور تصادفی مقداردهی اولیه کنید.
- Evaluate: برای هر جواب h در p مقدار تابع Fitness(h) را محاسبه نمائید.
- تا زمانی که $[\max_h \text{Fitness}(h)] < \text{Fitness\_threshold}$ یک جمعیت جدید ایجاد کنید.
- در نهایت، جوابی که دارای بیشترین مقدار Fitness است را برگردانید.

نحوه ایجاد جمعیت جدید

۱. select:تعداد (1-r)p جواب از میان P انتخاب و به P_s اضافه کنید. احتمال انتخاب یک جواب h_i از میان P عبارت است از:

$$P(h_i) = Fitness(h_i) / \Sigma_j Fitness(h_j)$$

هر چه برازش جوابی بیشتر باشد احتمال انتخاب آن بیشتر است.این احتمال همچنین با مقدار برازش جواب‌های دیگر نسبت عکس دارد.

۲. Crossover :با استفاده از احتمال بدست آمده توسط رابطه فوق، تعداد (rp)/2 زوج جواب از میان P انتخاب و با استفاده از اپراتور Crossover دو فرزند از آنان ایجاد کنید .فرزندان را به P_s اضافه کنید.

۳. Mutate :تعداد m درصد از اعضا P_s را با احتمال یکنواخت انتخاب و یک بیت از هر یک آنها را بصورت تصادفی معکوس کنید

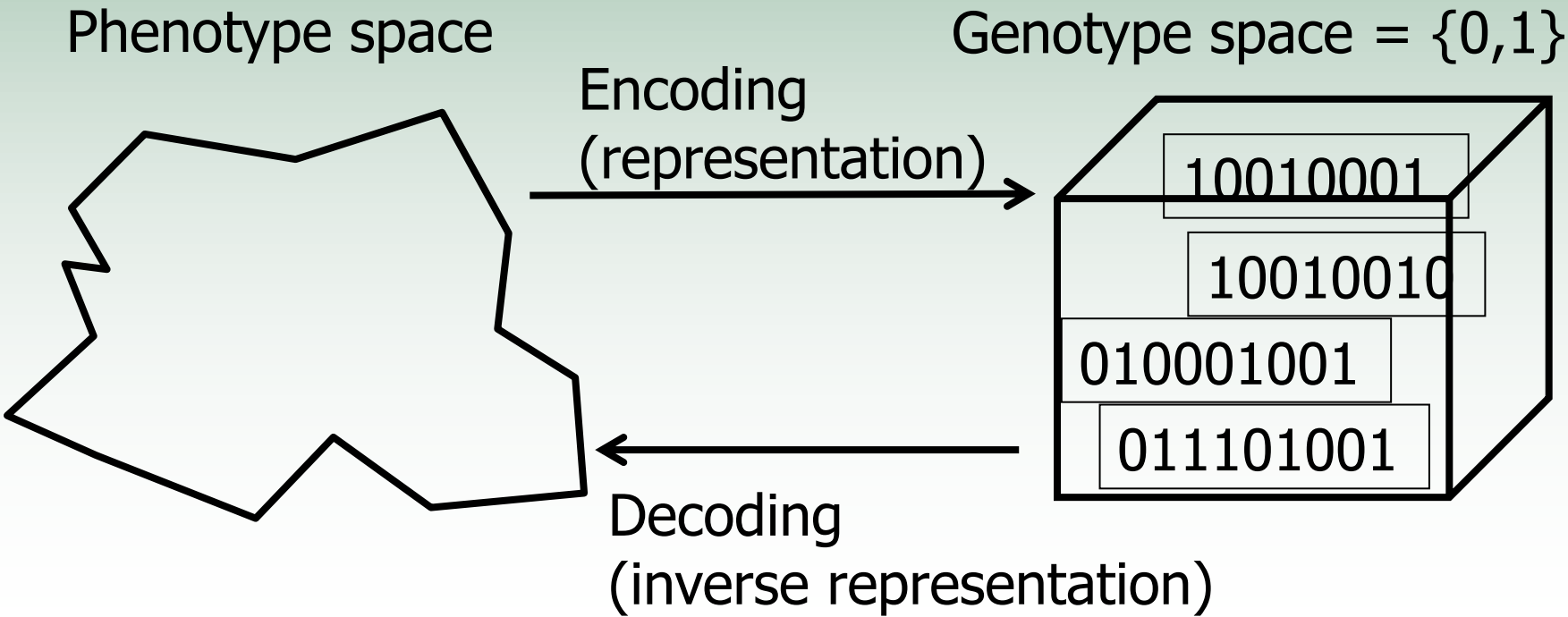
۴. Update: $P \leftarrow P_s$

۵. برای هر جواب h در P مقدار تابع Fitness را محاسبه کنید

نمایش جواب‌ها

در الگوریتم ژنتیک معمولا جواب‌ها بصورت رشته‌ای از بیت‌ها نشان داده می‌شوند تا اعمال اپراتورهای ژنتیکی برروی آنها ساده‌تر باشد.

- Phenotype :به مقادیر یا جواب‌های واقعی گفته می‌شود.
- Genotype:به مقادیر انکد شده یا کروموزم‌ها گفته می‌شود که در GA استفاده می‌شود.



Encoding

کدگذاری

انواع کدگذاری عبارتند از :

- ✓ Binary Encoding
- ✓ Value Encoding
- ✓ Tree Encoding

Binary Encoding

عمومی ترین روش وآشناترین نوع کدگذاری در GA همان روش کدگذاری باینری است در روش کد گذاری به روش باینری همه کورموزم ها با رشته هایی که شامل بیت هایی از ۰یا ۱ است کد می‌شوند.

Chromosome A 10110010110010101111100101

Chromosome B 11111111000001100000011111

Example of chromosomes with binary encoding

Value Encoding

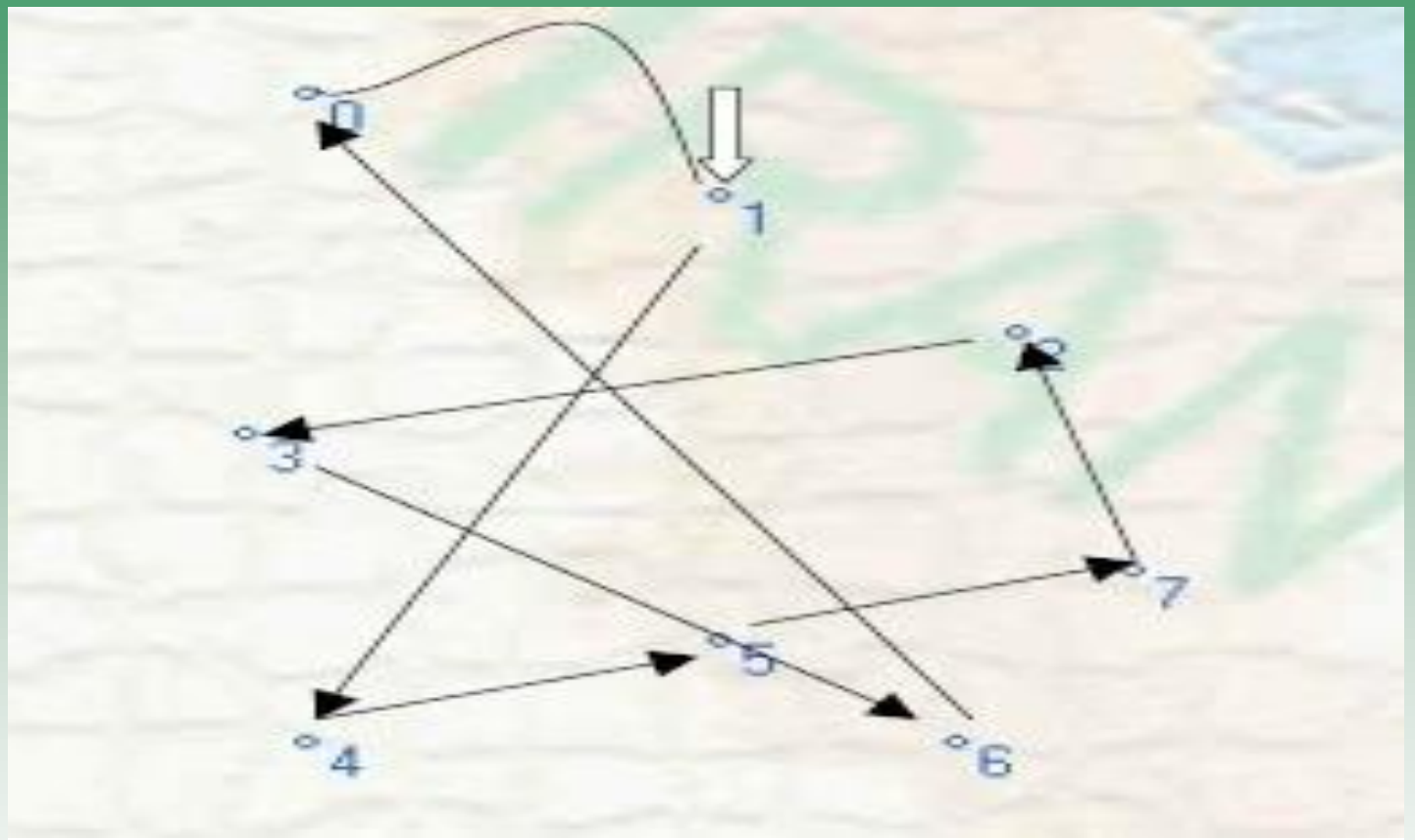
از شیوه ی Value Encoding می توانیم در حالت هایی که مسئله ارزش های پیچیده ای دارد استفاده کنیم ولی با این وجود برای اینگونه کد گذاری اغلب باید شیوه های توسعه یافته ی جدیدی در Crossover و Mutation به کار برد.
.

Chromosome A 1.2324 5.3243 0.4556 2.3293 2.4545

Chromosome B ABDJEIFJDHDIERJFDLDFLFEGT

Chromosome C (back), (back), (right), (forward), (left)

نمونه کروموزوم در مسئله فروشنده دوره گرد

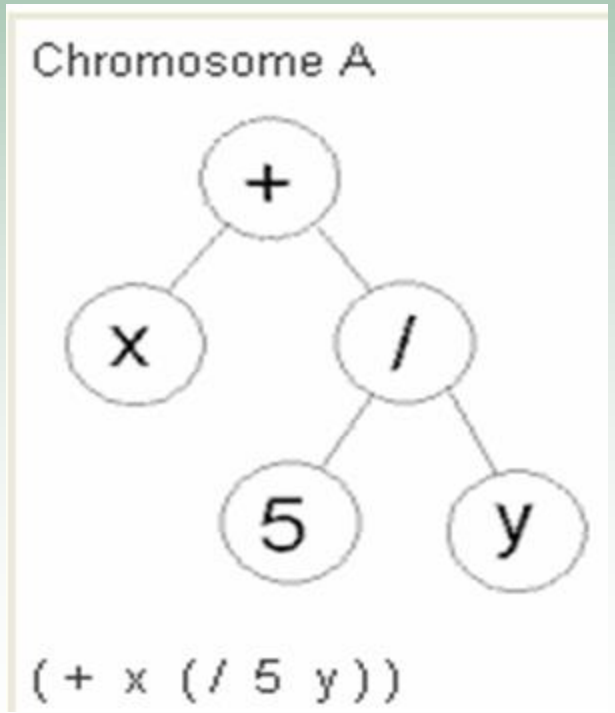


Sample chromosome

1	4	5	7	2	3	6	0

Tree Encoding

در کدگذاری درختی هر کروموزم یک درخت است که برای توابع یا دستورات در زبان برنامه‌نویسی استفاده می‌شود.



تابع برازش (Fitness Function)

تابعی است که ارزش هر کروموزوم در مسئله مورد نظر را تعیین می‌کند.

در مسئله کوله پشتی: بیشتر بودن مجموع قیمت اشیاء انتخاب شده به شرط رعایت محدودیت وزن

در مسئله فروشنده دوره گرد: کمتر بودن مجموع مسافت‌های طی شده

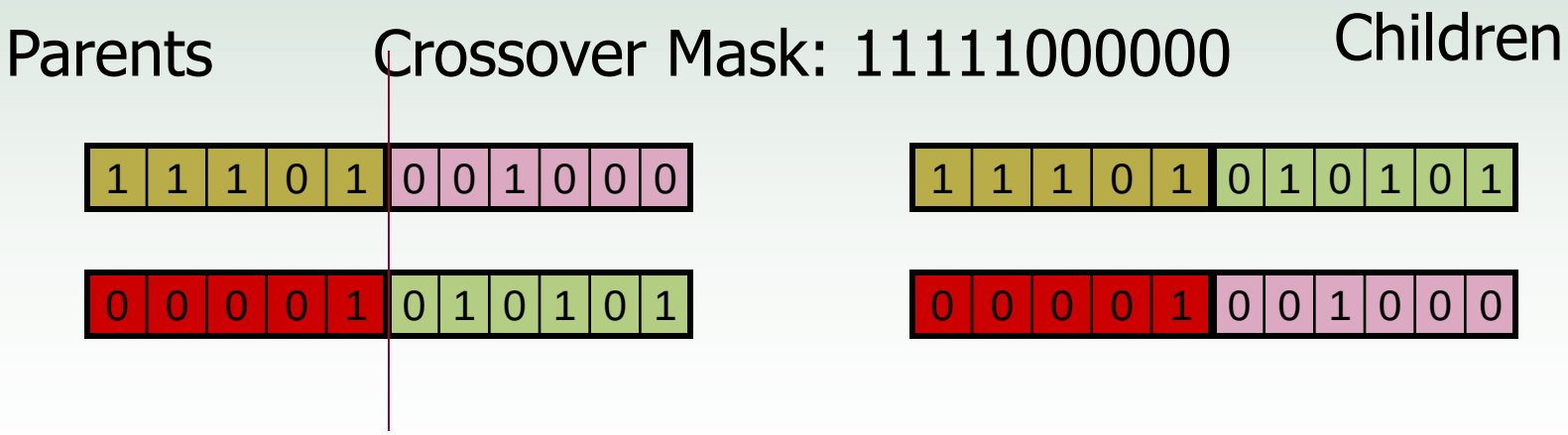
در مسئله پیدا کردن ماکزیمم: بیشتر بودن مقدار تابع

اپراتورهای ژنتیکی:برش (Crossover)

- اپراتور Crossover با استفاده از دو رشته والد دو رشته فرزند بوجود می‌آورد.
- برای اینکار قسمتی از بیتهای والدین در بیتهای فرزندان کپی می‌شود .
- انتخاب بیت‌هایی که باید از هر یک از والدین کپی شوند به روش‌های مختلف انجام می‌شود
  - Single-point crossover
  - Two-point crossover
  - Uniform crossover
- برای تعیین محل بیتهای کپی شونده از یک رشته به نام Crossover Mask استفاده می‌شود.

Single-point crossover

- یک نقطه تصادفی در طول رشته انتخاب می‌شود .
- والدین در این نقطه به دو قسمت می‌شوند.
- هر فرزند با انتخاب تکه اول از یکی از والدین و تکه دوم از والد دیگر بوجود می‌آید.

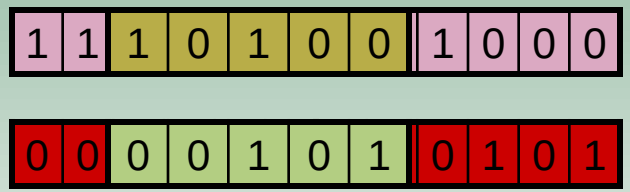


روشهای دیگر Crossover

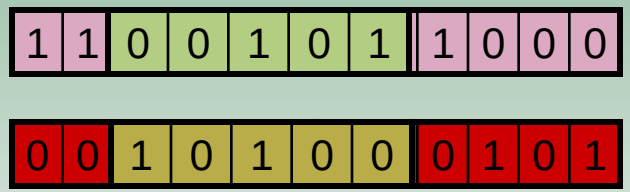
Two-point crossover

Crossover Mask: 00111110000

Parents



Children



Uniform crossover

Parents

Crossover Mask: 10011010011

Children

بیتها بصورت یکنواخت از والدین انتخاب می‌شوند

اپراتورهای ژنتیکی: جهش (Mutation)

- اپراتور Mutation برای بوجود آوردن فرزند فقط از یک والد استفاده می‌کند .اینکار با انجام تغییرات کوچکی در رشته اولیه بوقوع می‌پیوندد.
- با استفاده از یک توزیع یکنواخت یک بیت بصورت تصادفی انتخاب و مقدار آن تغییر پیدا می‌کند.
- معمولا mutation بعد از انجام crossover اعمال می‌شود.

Parent

1	1	1	0	1	0	0	1	0	0	0

Child

1	1	1	0	1	1	0	1	0	0	0

Mutation

انواع دیگر جهش

Order changing mutation - two numbers are selected and exchanged

(1 **2** 3 4 5 6 **8** 9 7) => (1 **8** 3 4 5 6 **2** 9 7)

Adding Mutation: a small number (for real value encoding) is added to (subtracted from) selected values

(1.29 5.68 **2.86** **4.11** 5.55) =>
(1.29 5.68 **2.73** **4.22** 5.55)

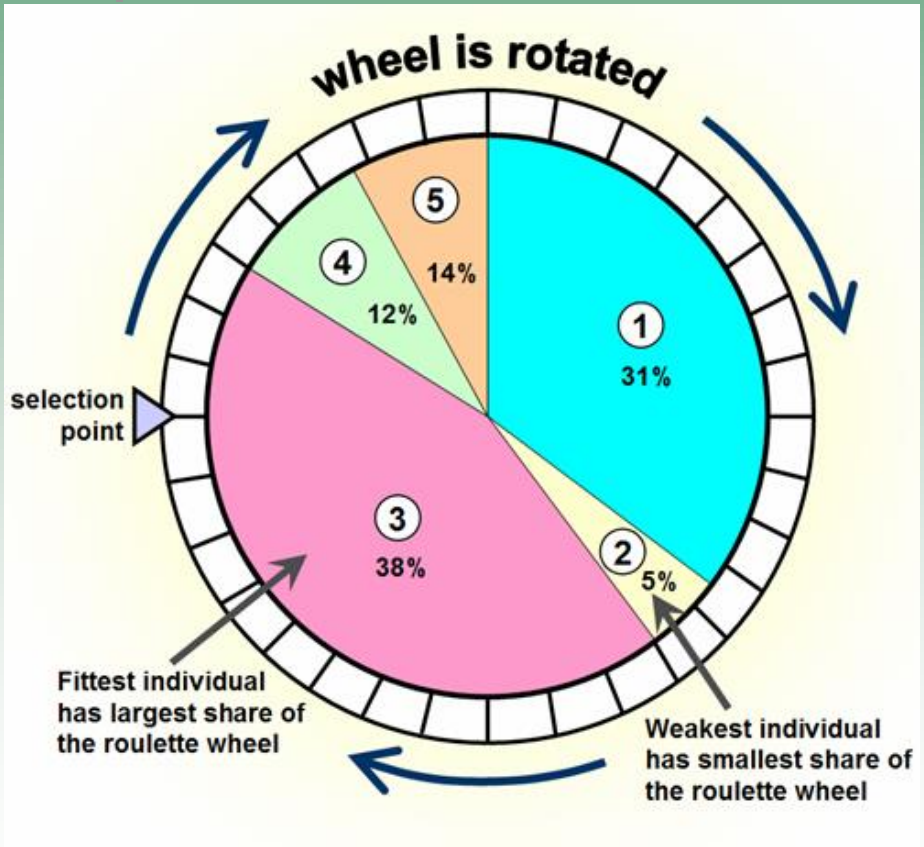
Selection

انتخاب

در روش معرفی شده در الگوریتم ساده GA احتمال انتخاب یک جواب برای استفاده در جمعیت بعدی بستگی به نسبت fitness آن به fitnessبقیه اعضا دارد .این روش Roulette Wheel selectionنامیده می شود.

Selection

انتخاب



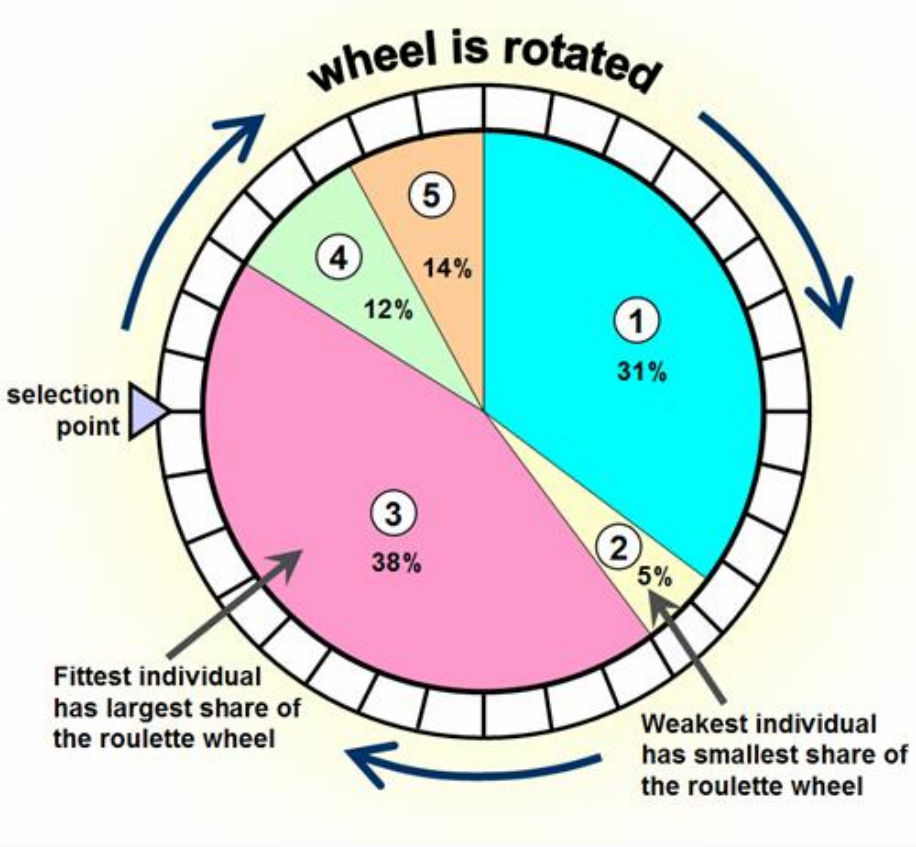
چرخ رولت

Roulette wheel selection

انتخاب در این نوع از یک نوع بازی الهام گرفته شده است و به این صورت است که گردونه را می چرخانیم و بعد از مدتی گردونه می ایستد و ما از قبل یک نشانه در بیرون گردونه گذاشته‌ایم. کروموزومی را که نشانه به آن اشاره می‌کند را انتخاب می‌کنیم.

Selection

انتخاب



آشکار است که کوروموزومی که ارزش Fitness بالایی دارد قسمت بزرگی از گردونه را به خود اختصاص داده است در نتیجه با توجه به علم آمار ، احتمال انتخاب شدن بیشتری دارد و دفعات بیشتری از انتخاب را به خود اختصاص می دهد. تصویر، همه ی کورموزوم های یک نسل را نشان داده است . همانطور که دیده می شود اندازه ی این قسمت ها با هم برابر نیست بلکه این اندازه ها به ارزشی که تابع Fitness به هر کورموزوم می دهد بستگی دارد.

Selection

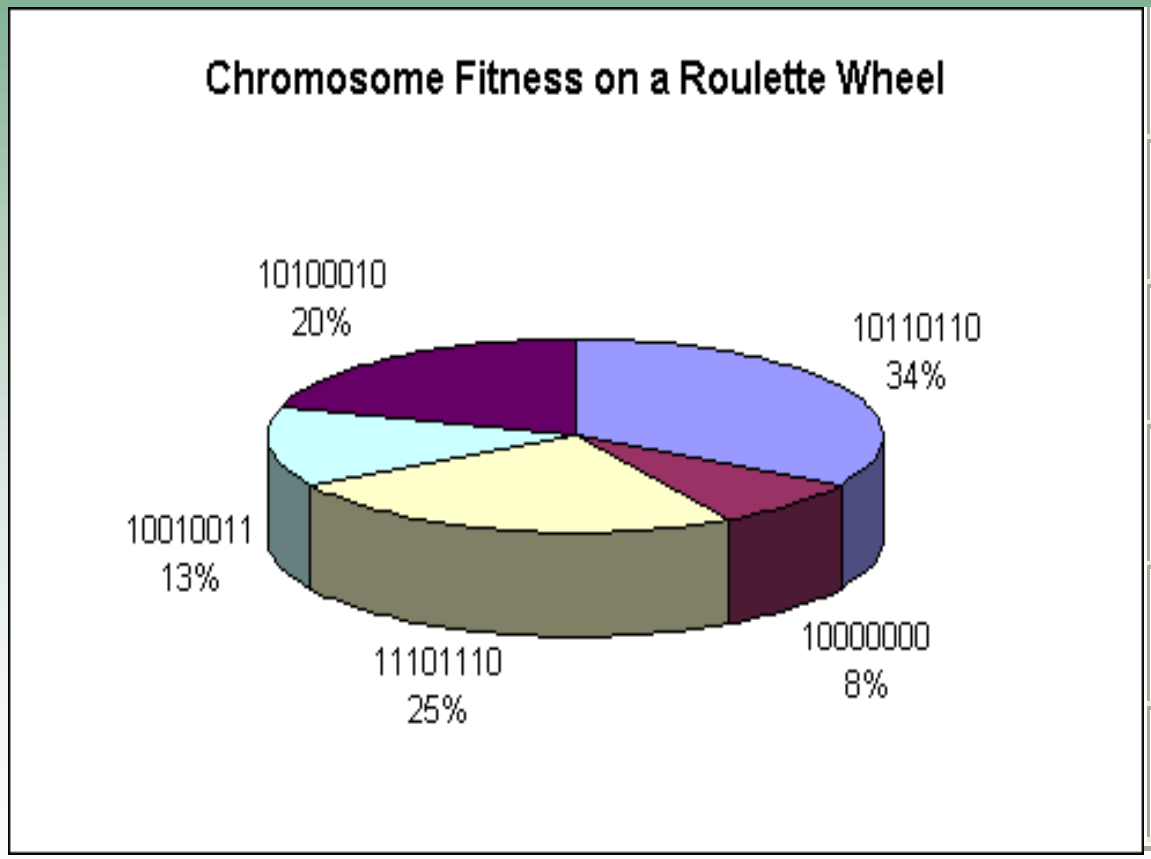
انتخاب

انتخاب چرخ رولت که اولین بار توسط هالند پیشنهاد شد یکی از مناسب‌ترین انتخاب‌های تصادفی بوده که ایده آن، احتمال انتخاب می‌باشد. احتمال انتخاب متناظر با هر کروموزوم، بر اساس برازندگی آن محاسبه شده که اگر f_k مقدار برازندگی کروموزوم k ام باشد، احتمال بقای متناظر با آن کروموزوم عبارت است از:

$$P_K = f_k / \sum_{i=1}^{n} f_i$$

Selection

انتخاب



Chromosome	*Fitness*
10110110	20
10000000	5
11101110	15
10010011	8
10100010	12

انواع مختلف GA

- انواع مختلف انتخاب (Selection)
  - Roulette - Wheel
  - Tournament
  - Elitism, etc.
- انواع مختلف برش (Cross-Over)
  - One-point crossover
  - Multi-point crossover
  - etc.
- انواع مختلف کدگذاری (Encoding)
  - Bit string
  - Integer values
  - Ordered set of symbols
- انواع مختلف جهش (Mutation)

مثال: پیدا کردن ماکزیمم یک تابع

پیدا کردن ماکزیمم یک تابع

- **Finding the maximum of a function: f(x) = x²**
  - Range [0, 31] → Goal: find max ($31^2 = 961$)
  - Binary representation: string length 5 = 32 numbers (0-31)
- **Genotype** – collectivity of all genes
- **Phenotype** – expression of genotype in environment

genotype	0 0 1 0 1	
mapping	$2^8\ 2^4\ 2^2\ 2^1\ 2^0$ 16 8 4 2 1	
phenotype	0*16+0*8+1*4+0*2+1*1 = 5	
fitness	25	= f(x)

جمعیت شروع: تابع $f(x) = x^2$

	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

انتخاب: $F(x) = x^2$

	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

- بدترین، حذف می شود.

انتخاب: $F(x) = x^2$

	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

- بهترین فرد تکرار می شود تا اندازه جمعیت ثابت بماند.

انتخاب: $F(x) = x^2$

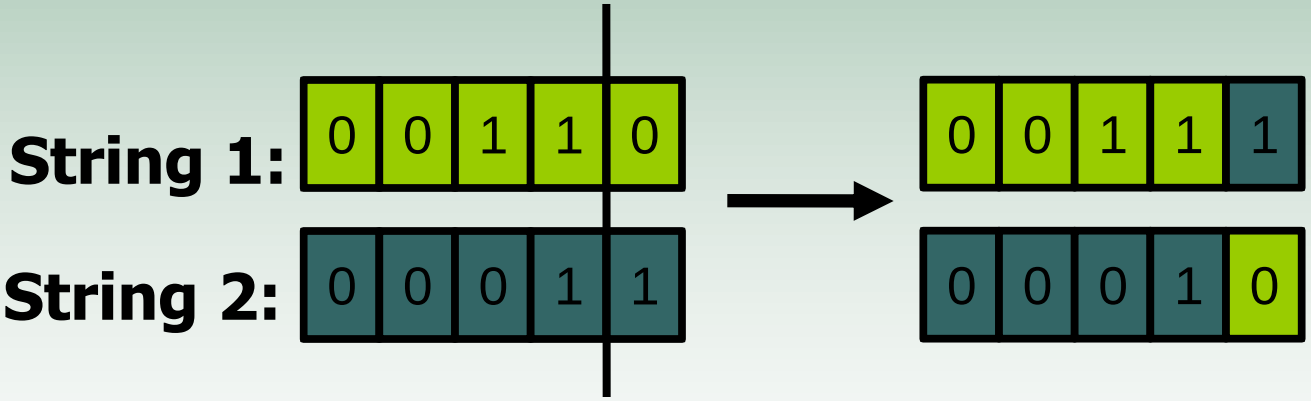
	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

- بقیه افراد همان یک بار تکرار می شوند.

برش: $F(x) = x^2$

- Parents and x-position randomly selected (equal recombination)

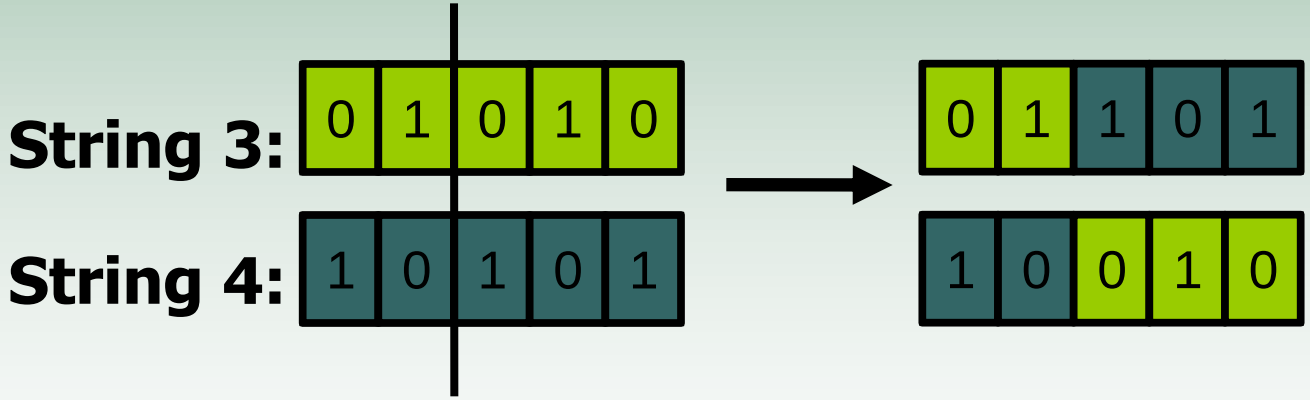
	partner	x-position
String 1	**String 2**	**4**
String 3	String 4	2



برش: $F(x) = x^2$

- Parents and x-position randomly selected (equal recombination)

	partner	x-position
String 1	String 2	4
String 3	**String 4**	**2**



جهش: $F(x) = x^2$

- bit-flip:
  - Offspring -String 1: **0**0111 (7) → **1**0111 (23)
  - String 4: 10**1**01 (21) → 10**0**01 (17)

$F(x) = x^2$

- تمام افراد در جمعیت والدین با افراد جدید تولید شده جایگزین می شوند.
- **جمعیت جدید(فرزندان):**

	binary	value	fitness
String 1	10111	23	529
String 2	00010	2	4
String 3	01101	13	169
String 4	10000	16	256
String 5	10001	17	289

- بعد از یک دور، بهترین فرد: 10111 (23) برازندگی: 529

Stop Condition شرط توقف

معیارهای مختلفی را می توان برای توقف الگوریتم در نظر گرفت و معمولا روتین های توقف مختلف است و بستگی به پیچیدگی و چگونگی مساله دارد.معمولا چند معیار برای توقف استفاده می شود تا احتمالهای مختلف وقوع پیشامدها در طی اجرای الگوریتم حساب شوند.

- ✓ توقف تکامل هر گاه پیشرفت یا بهبودی در جوابهای ایجاد شده مشاهده نشود .
- ✓ توقف تکامل هنگام همگرایی الگوریتم به عبارتی ، زمانیکه اکثریت یا همه افراد جمعیت همانند شده باشند .
- ✓ گذاشتن محدودیت زمانی
- ✓ محدود کردن تعداد نسلها مثلاً تا K نسل

سایر روش‌های برش (Crossover)

در بسیاری از مسایل مانند مساله فروشنده دوره گرد با جایگشت‌های مختلفی از مجموعه جوابها روبرو هستیم.

در این مساله تعدادی شهر داریم که فاصله میان آنها معلوم است و با شروع از یک شهر و ختم به همان شهر :

۱-از تمام شهرها فقط و فقط یکبار عبور نمائیم.

۲-کمترین مسافت ممکنه را طی نماییم.

اما نکته‌ای که در اینجا مهم است و باعث شده تا کدینگ باینری روش مناسبی برای این مساله نباشد، این نکته است که حتما باید برش میان دو والد بنحوی صورت بگیرد که هیچ عنصر تکراری وجود نداشته باشد.

Crossover

برش

روش تک نقطه به این شکل اصلاح میشود که تمام قسمت قبل از نقطه برش در والد اول عینا در فرزند کپی میگردد. بقیه ژنهای والد اول که مطمئناً هنوز در فرزند تکرار نشده‌اند، مطابق با ترتیب قرار گرفتنشان در والد دوم در فرزند کپی می‌شوند.

(1 2 3 4 5 6 7 8 9) + (4 5 3 6 8 9 7 2 1) = (1 2 3 4 5 6 8 9 7)

Single point crossover

Crossover

برش

برش نگاشته جزیی (Partial Mapped):

این روش به روش PMX معروف بوده و در حقیقت همان عملگر دو نقطه برش می‌باشد که برای حالت خاصی ارائه شده است.

در این روش دو عدد را به صورت تصادفی به عنوان نقاط برش به دست آورده، سپس قسمت مابین دو نقطه برش را در دو کروموزوم تعویض کرده و آنگاه قسمت‌های دو طرف طوری مقدارگذاری می‌شوند که در هیچ کدام از دو کروموزوم، تکراری صورت نگیرد. روش کار با یک مثال بیان شده است.

والد اول: **۷۹۱ /۵۶۲۸ /۴۳**

والد دوم: **۲۱۷ /۸۳۴۹ /۶۵**

Crossover

برش

حال برای تولید نوزاد به این صورت عمل می‌شود که قسمت مابین را عوض کرده سپس از والد خودش از ابتدای کروموزوم شروع نموده هر عددی را که در قسمت مابین کروموزوم جدید نباشد عیناً نوشته و برای تکراری‌ها جای خالی قرار داده می‌شود سپس از وا لد دیگر از ابتدای کروموزوم شروع کرده و هر عددی که در نوزاد جدید نباشد به جای محل خالی، گذاشته می‌شود تا کلیه جاهای خالی پر گردد. برای نوزاد دوم نیز به همین صورت عمل می‌شود. بنابراین ابتدا قسمت‌های میانی را جابجا کرده و دو کروموزوم زیر حاصل می‌شود.

Crossover

برش

نوزاد اول: **--/۸۳۴۹/---** **والد اول: ۷۹۱/ ۵۶۲۸/ ۴۳**

نوازد دوم: **--/۵۶۲۸/---** **والد دوم: ۲۱۷/ ۸۳۴۹/ ۶۵**

سپس جاهای خالی نوزاد اول در صورت تکرای نبودن ژن مورد نظر پر می‌شود:

نوزاد اول: **۱-۷/ ۸۳۴۹/ --**

حال جاهای خالی باقیمانده نوزاد اول پر می‌شود.

نوزاد اول: **۷۲۱/ ۸۳۴۹/ ۶۵**

با توجه به مطالب گفته شده، نوزاد دوم به صورت زیر به دست می‌آید:

نوازد اول: **--/۵۶۲۸/-۱۷**

نوزاد دوم: **۴۳/۵۶۲۸/۹۱۷**

Crossover

برش

Order برش ترتیب

این روش به روش OX معروف می‌باشد در این روش دو عدد را به صورت تصادفی به عنوان نقاط برش به دست آورده سپس قسمت مابین را در دو کروموزوم ثابت نگه داشته ولی قسمت‌های دو طرف به این صورت به دست می‌آید که برای نوزاد اول در والد دوم از ابتدای کروموزوم شروع کرده و آنهایی را که در قسمت مابین نوزاد وجود ندارد در جای خالی قرار می‌گیرند

Crossover

برش

والد اول: **۱۲/۳۵۹۸/۴۷۶**

والد دوم: **۲۹/۷۶۴۱/۸۳۵**

برای نوزاد اول قسمت مابین والد یک، بدون تغییر باقی می‌ماند .

نوزاد اول: **--/۳۵۹۸/---**

حال قسمت خالی از روی والد دوم پر می‌گردد. از ابتدای کروموزوم شروع کرده و آنهایی را که در قسمت مابین نوزاد وجود ندارد در جای خالی قرار می‌دهیم

نوزاد اول: **۲۷/۳۵۹۸/۶۴۱**

برای نوزاد دوم نیز به همان صورت عمل می‌شود:

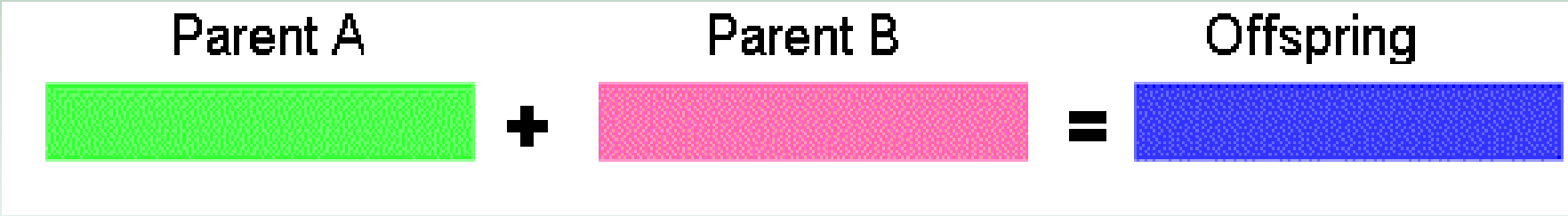
نوزاد دوم: **۲۳/۷۶۴۱/۵۹۸**

Crossover

برش

Arithmetic crossover

با توجه به قواعد و دستورات ریاضی خاصی فرزند ها تولید می شوند مثل . AND



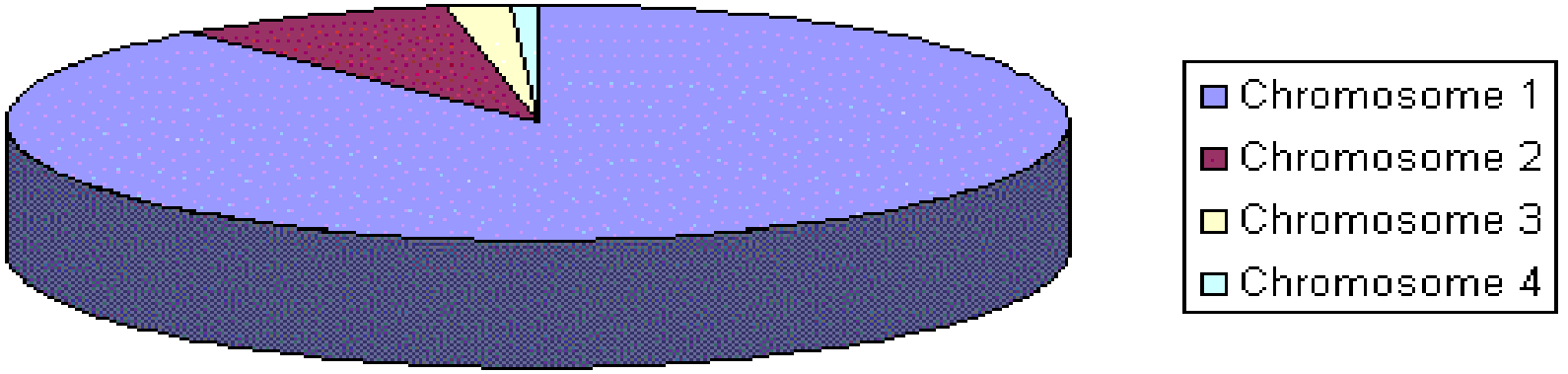
سایر روش های Selection

Rank Selection

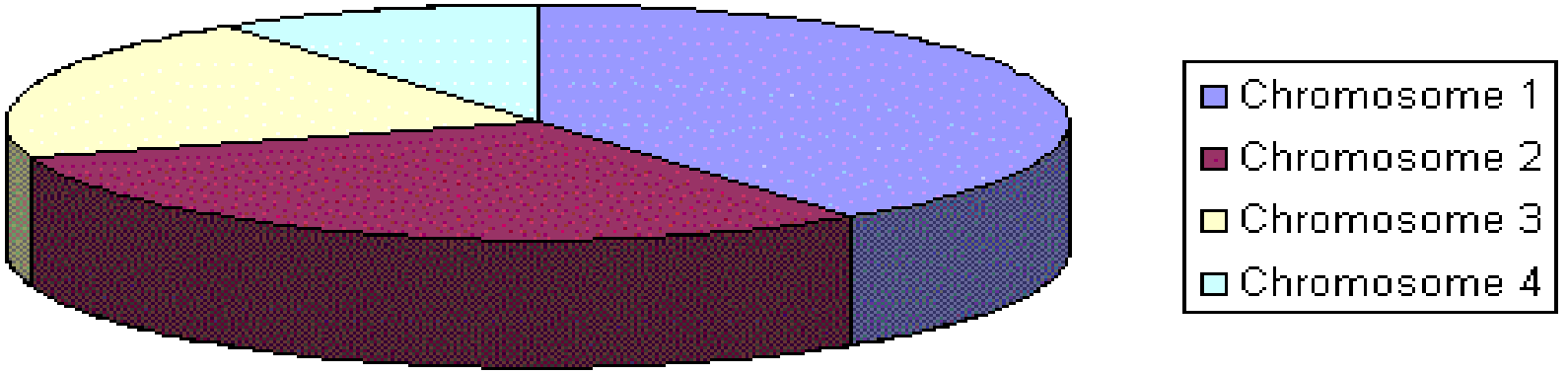
روش انتخاب چرخه رولت که توضیح داده شد روش خوبی است ولی در حالتی که اختلاف ارزش های Fitness در کروموزوم ها زیاد باشد دچار مشکل می شود مثلاً اگر ارزش بهترین کورموزوم ۹۰ ٪ باشد مجموع کورموزوم های دیگر بسیار شانس کمتری برای انتخاب شدن دارند .

در شیوه Rank Selection به این صورت عمل می کنیم که ابتدا جمعیت یا نسل را مرتب می کنیم سپس به هر کورموزوم با توجه به ارزش Fitness آن عددی اختصاص می دهیم مثلا بدترین کروموزوم۱ کورموزوم ما قبل بدتری ۲ الی آخرتا اینکه به بهترین کورموزوم N را می دهیم (N تعداد کورموزوم های نسل) . البته در این روش همگرایی بسیار آهسته اتفاق می افتد به خاطر اینکه اختلاف کورموزوم ها کاهش پیدا کرده است . در تصاویر صفحه بعد به وضوح چگونگی تغییرات نمودار از شیوه ی انتخاب چرخ رولت به Rank را مشاهده می کنید .

سایر روش های Selection(ادامه ...)



Situation before ranking (graph of fitnesses)

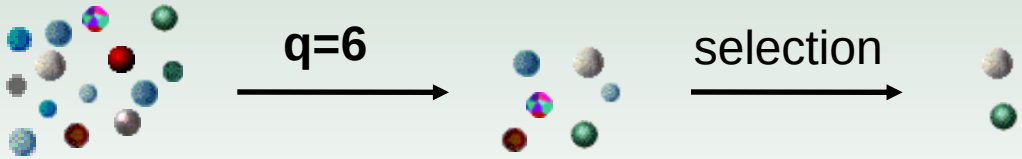


Situation after ranking (graph of order numbers)

سایر روش های Selection(ادامه ...)

Tournament Selection

- q نفر را از جمعیت فعلی انتخاب کنید.
- برنده: فرد (یا افرادی) که بهترین برازندگی را در بین این qنفر دارند.
- مثال: دو فردی که بهترین هستند را به عنوان والدین برای برش انتخاب کنید.



سایر روش های Selection(ادامه ...)

انتخاب نخبه‌گرا Elitism

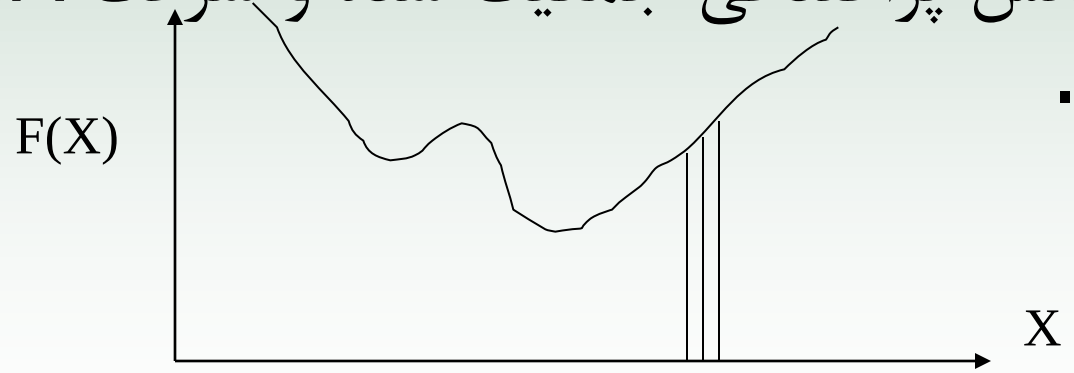
در این روش تعداد ثابتی از بهترین کروموزومها بدون تغییر به نسل بعد منتقل می‌شوند. این عمل باعث می شود بهترین برازندگی جمعیت کاهش نیابد. نخبه گرایی می تواند باعث افزایش سرعت الگوریتم ژنتیک شود.

Steady-State

ایده اصلی این روش برای انتخاب جمعیت یا نسل جدید این است ، که قسمت بزرگی (بر خلاف نخبه گرا) از کروموزوم ها بتوانند برای نسل جدید حفظ شوند .در این روش تعدادی از کورموزوم های بد (با Fitness پایین) حذف می شوند و به جای آنها فرزندان جدیدی که از برش و جهش ایجاد شده اند جایگذاری می شوند .

Crowding ازدحام

- crowding پدیده‌ای است که در آن عضوی که سازگاری بسیار بیشتری از بقیه افراد جمعیت دارد بطور مرتب تولید نسل کرده و با تولید اعضای مشابه درصد عمده‌ای از جمعیت را اشغال می‌کند.
- اینکار باعث کاهش پراکندگی جمعیت شده و سرعت GA را کم می‌کند.



راه حل رفع مشکل ازدحام

- استفاده از ranking برای انتخاب نمونه‌ها: با اختصاص رتبه به جوابی که بسیار بهتر از بقیه عمل می‌کند مقدار این برتری نشان داده نخواهد شد.
- استفاده از روش Tournament برای انتخاب نمونه ها.
- Fitness sharing : مقدار Fitness یک عضو در صورتی که اعضا مشابهی در جمعیت وجود داشته باشند کاهش می یابد.

چرا GA کار می‌کند؟

- سئوالی که ممکن است برای تازه واردین به روش‌های ژنتیکی ایجاد شود این است که آیا این روش واقعا می‌تواند کار مفیدی انجام دهد؟

ارزیابی جمعیت و قضیه Schema

- آیا می‌توان تکامل در جمعیت در طی زمان را بصورت ریاضی مدل نمود؟
- قضیه Schema می‌تواند مشخصه پدیده تکامل در GA را بیان نماید.
- یک Schema مجموعه ای از رشته بیت ها را توصیف میکند.یک Schemaهر رشته ای از 0 و1و *است .مثل 0*10 که * حالت don’t care است.
- یک رشته بیت را می‌توان نماینده هر یک از Schema های متفاوتی دانست که با آن تطابق دارند .مثلا 0010 را می‌توان نماینده 2^4 Schema مختلف دانست : 00**,0*10,**** و غیره

قضیه Schema

- قضیه Schema بیان می‌کند که چگونه یک Schema در طول زمان در جمعیت تکامل پیدا خواهد کرد.
- فرض کنید که در لحظه t تعداد نمونه هائی که نماینده یک Schema مثل s هستند برابر با m(s, t) باشد . این قضیه مقدار مورد انتظار m(s, t+1) را مشخص می‌کند.

قضیه Schema

- احتمال انتخاب یک جواب برابر بود با:

$$P(h_i) = Fitness\ (h_i) /\ \Sigma_j\ Fitness\ (h_j)$$

- این مقدار احتمال را می‌توان بصورت زیر نیز نشان داد:

$$P(h_i) = f\ (h_i) /\ n\ f'\ (t_i)$$

مقدار متوسط fitness برای تمامی جواب ها

قضیه Schema

- اگر عضوی از این جمعیت انتخاب شود احتمال اینکه این عضو نماینده s باشد برابر است با:

$$p(h \in s) = \sum_{h \in s \cap p_s} \frac{f(h)}{n\,\bar{f}(t)} = \frac{u(s,t)}{n\,\bar{f}(t)} m(s,t)$$

- که در آن مقدار u(s,t) برابر است با مقدار میانگین برازندگی اعضای s در لحظه t

$$u(s,t) = \frac{\sum_{h \in s \cap p_s} f(h)}{m(s,t)}$$

قضیه Schema

- از این رو مقدار مورد انتظار برای نمونه هائی از s که از n مرحله انتخاب مستقل حاصل خواهند شد برابر است با:

$$E[m(s,t+1)] = \frac{u(s,t)}{\bar{f}(t)} m(s,t)$$

- رابطه فوق به این معناست که تعداد Schema های مورد انتظار در لحظه t+1 متناسب با مقدار میانگین u(s, t) بوده و با مقدار برازندگی سایر اعضا نسبت عکس دارد.

قضیه Schema

- برای بدست آوردن رابطه فوق فقط اثر مرحله انتخاب نمونه ها در نظر گرفته شده است .با در نظر گرفتن اثر crossover و Mutation به رابطه زیر خواهیم رسید:

$$E[m(s,t+1)] \geq \frac{u(s,t)}{\bar{f}(t)} m(s,t)\left(1-p_c\frac{d(s)}{l-1}\right)\left(1-p_m\right)^{o(s)}$$

قضیه Schema

$$E[m(s,t+1)] \geq \frac{u(s,t)}{\bar{f}(t)} m(s,t)\left(1-p_c \frac{d(s)}{l-1}\right)\left(1-p_m\right)^{o(s)}$$

احتمال زنده ماندن شمای S تحت عمل برش تک نقطه ای

- p_c احتمال عمل برش تک نقطه ای
- $d(s)$ فاصله بین سمت چپ ترین و سمت راست ترین بیت تعریف شده (غیر *) در s
- l طول رشته بیتی هر فرد

قضیه Schema

$$E[m(s,t+1)] \geq \frac{u(s,t)}{\bar{f}(t)} m(s,t)\left(1-p_c \frac{d(s)}{l-1}\right)\left(1-p_m\right)^{o(s)}$$

احتمال زنده ماندن شمای s تحت عمل جهش

- p_m احتمال عمل جهش(و در نتیجه، $1-p_m$ احتمال عدم تغییر یک بیت)
- o(s) تعداد بیتهای غیر * در شمای s

قضیه Schema

$$E[m(s,t+1)] \geq \frac{u(s,t)}{\bar{f}(t)} m(s,t)\left(1-p_c \frac{d(s)}{l-1}\right)\left(1-p_m\right)^{o(s)}$$

نتیجه:

- تعداد مورد انتظار نمونه های یک schema در جمعیت، به برازندگی نسبی آن میل می کند.
- (The expected number of instances of a schema in the population tends toward its relative fitness)

خلاصه

- یک Schema اطلاعات مفید و امید بخش موجود در جمعیت را کد می‌کند.
- از آنجائیکه همواره رشته هائی که سازگارترند شانس بیشتری برای انتخاب شدن دارند، بتدریج مثالهای بیشتری به بهترین Schema ها اختصاص می یابند.
- عمل crossover باعث قطع رشته ها در نقاط تصادفی میشود. با این وجود در صورتی که اینکار باعث قطع Schema نشده باشد آنرا تغییر نخواهد داد. در حالت کلی Schema های با طول کوتاه کمتر تغییر می‌کنند.
- عمل mutation در حالت کلی باعث تغییرات موثر در Schema نمی‌گردد.

موازی سازی الگوریتم ژنتیک

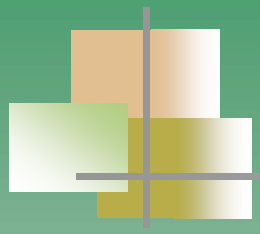
- الگوریتم های ژنتیک به طور ذاتی برای پیاده سازی موازی مناسب هستند و روش های مختلفی برای موازی سازی آنها ارائه شده است.
- روش Coarse grain جمعیت را به زیرگروه های مجزایی از افراد به نام دِم (deme) تقسیم می کند.
- هر دِم به یک نود محاسباتی داده می شود و یک جستجوی الگوریتم ژنتیک استاندارد روی هر نود اجرا می شود.

Coarse Grain

- مراوده(Communication) بین دِم ها با فرکانس کمتری نسبت به مراوده های درون دِم انجام می‌شود.
- انتقال بین دِم ها توسط فرایند مهاجرت(Migration) انجام می شود که در آن، افراد از یک دِم به دِم دیگر کپی یا منتقل می شوند.
- این فرایند، در گونه های زیستی بین زیرجمعیت های مجزا نیز ممکن است اتفاق بیفتد.
- یکی از مزایای این روش کاهش مشکل ازدحام است که اغلب در نسخه های غیر موازی اتفاق می افتد که در آن، سیستم در یک جواب بهینه محلی گیر می افتد.

Fine Grain

- بر خلاف روش coarse-grained، در روش Fine Grain معمولا یک پردازنده به هر فرد در جمعیت نسبت داده می شود.
- برش (یا ترکیب) بین دو فرد همسایه انجام می شود. انواع مختلفی از همسایگی وجود دارد (مثل grid ، torus و ...)



مساله فروشنده دوره گرد

Travelling Salesman Problem(TSP)

مساله فروشنده دوره گرد

در این مساله تعدادی شهر داریم که فاصله میان آنها معلوم است و با شروع از یک شهر و ختم به همان شهر :

۱-از تمام شهرها فقط و فقط یکبار عبور نمائیم.

۲-کمترین مسافت ممکنه را طی نماییم.

جمعیت اولیه برای TSP

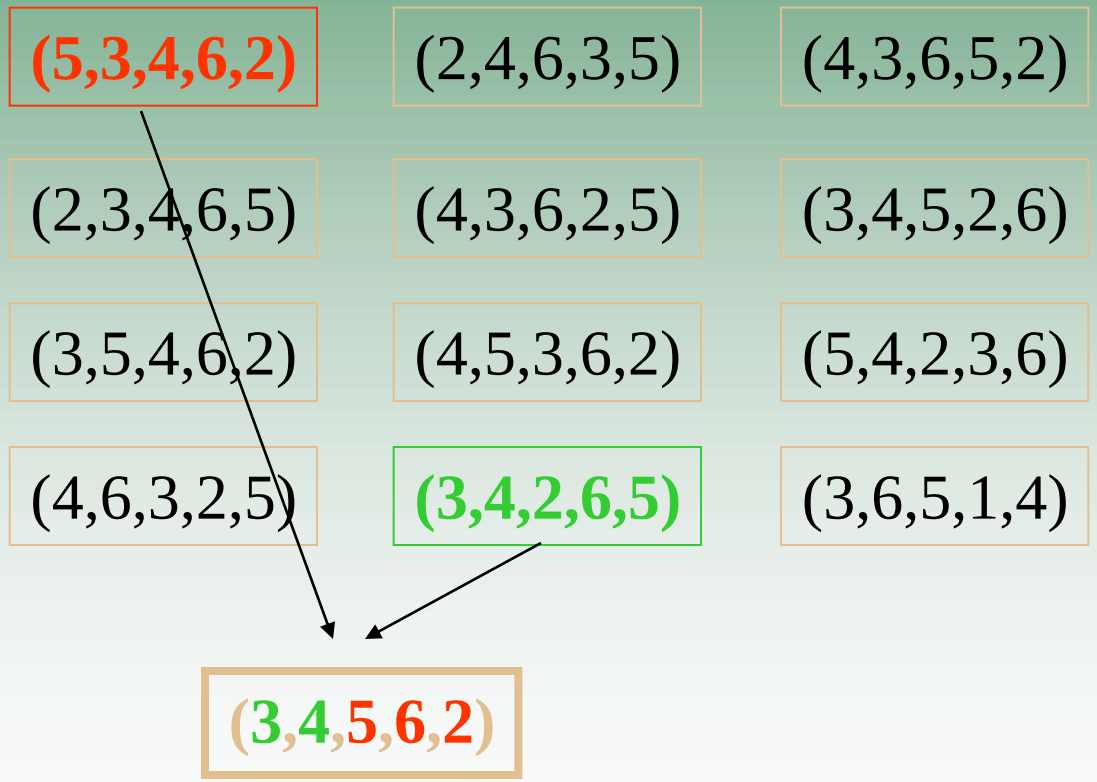
(5,3,4,6,2)	(2,4,6,3,5)	(4,3,6,5,2)
(2,3,4,6,5)	(4,3,6,2,5)	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	(5,4,2,3,6)
(4,6,3,2,5)	(3,4,2,6,5)	(3,6,5,1,4)

انتخاب والدین

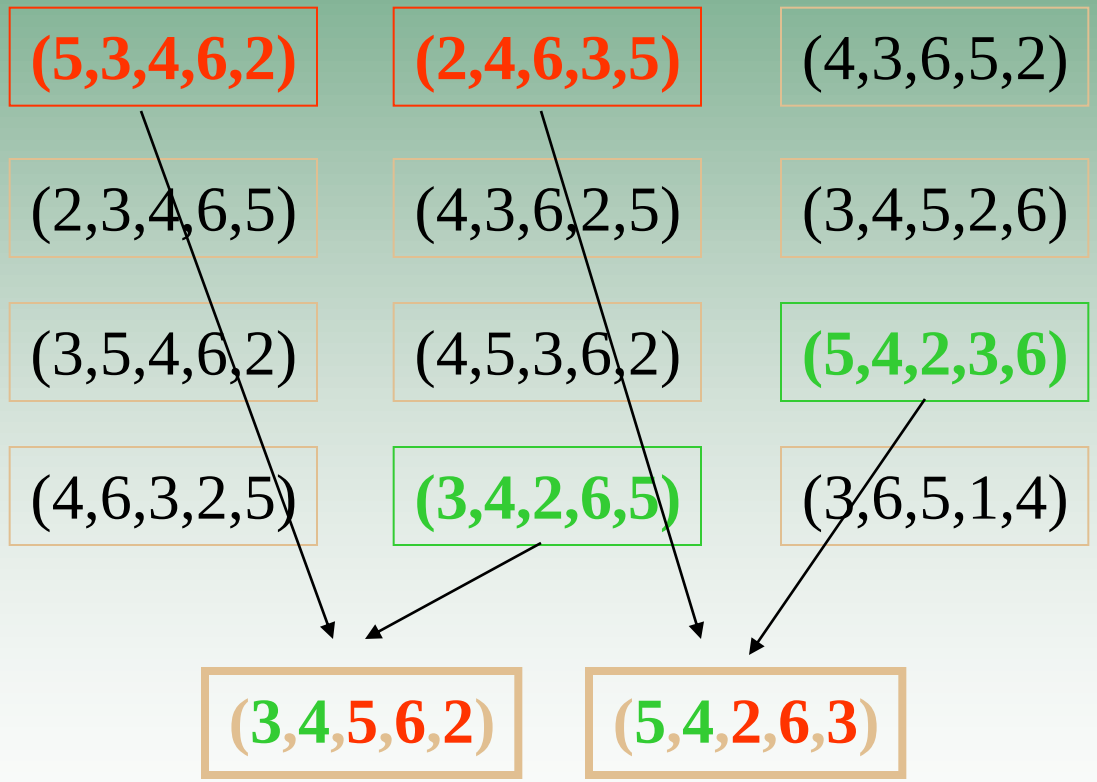
(5,3,4,6,2)	(2,4,6,3,5)	(4,3,6,5,2)
(2,3,4,6,5)	(4,3,6,2,5)	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	(5,4,2,3,6)
(4,6,3,2,5)	**(3,4,2,6,5)**	(3,6,5,1,4)

سعی کنید بهترین ها را انتخاب کنید

تولید نوزاد (برش تک نقطه ای)



تولید نوزاد دیگر



جهش

(5,3,4,6,2)	**(2,4,6,3,5)**	(4,3,6,5,2)
(2,3,4,6,5)	**(4,3,6,2,5)**	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	**(5,4,2,3,6)**
(4,6,3,2,5)	**(3,4,2,6,5)**	(3,6,5,1,4)

(3,4,5,6,2) **(5,4,2,6,3)**

جهش

(5,3,4,6,2)	**(2,4,6,3,5)**	(4,3,6,5,2)
(2,3,4,6,5)	**(2,3,6,4,5)**	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	**(5,4,2,3,6)**
(4,6,3,2,5)	**(3,4,2,6,5)**	(3,6,5,1,4)

(3,4,5,6,2) **(5,4,2,6,3)**

حذف

(5,3,4,6,2)	(2,4,6,3,5)	~~(4,3,6,5,2)~~
~~(2,3,4,6,5)~~	(2,3,6,4,5)	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	(5,4,2,3,6)
(4,6,3,2,5)	(3,4,2,6,5)	(3,6,5,1,4)

(3,4,5,6,2) (5,4,2,6,3)

Tend to kill off the worst ones.

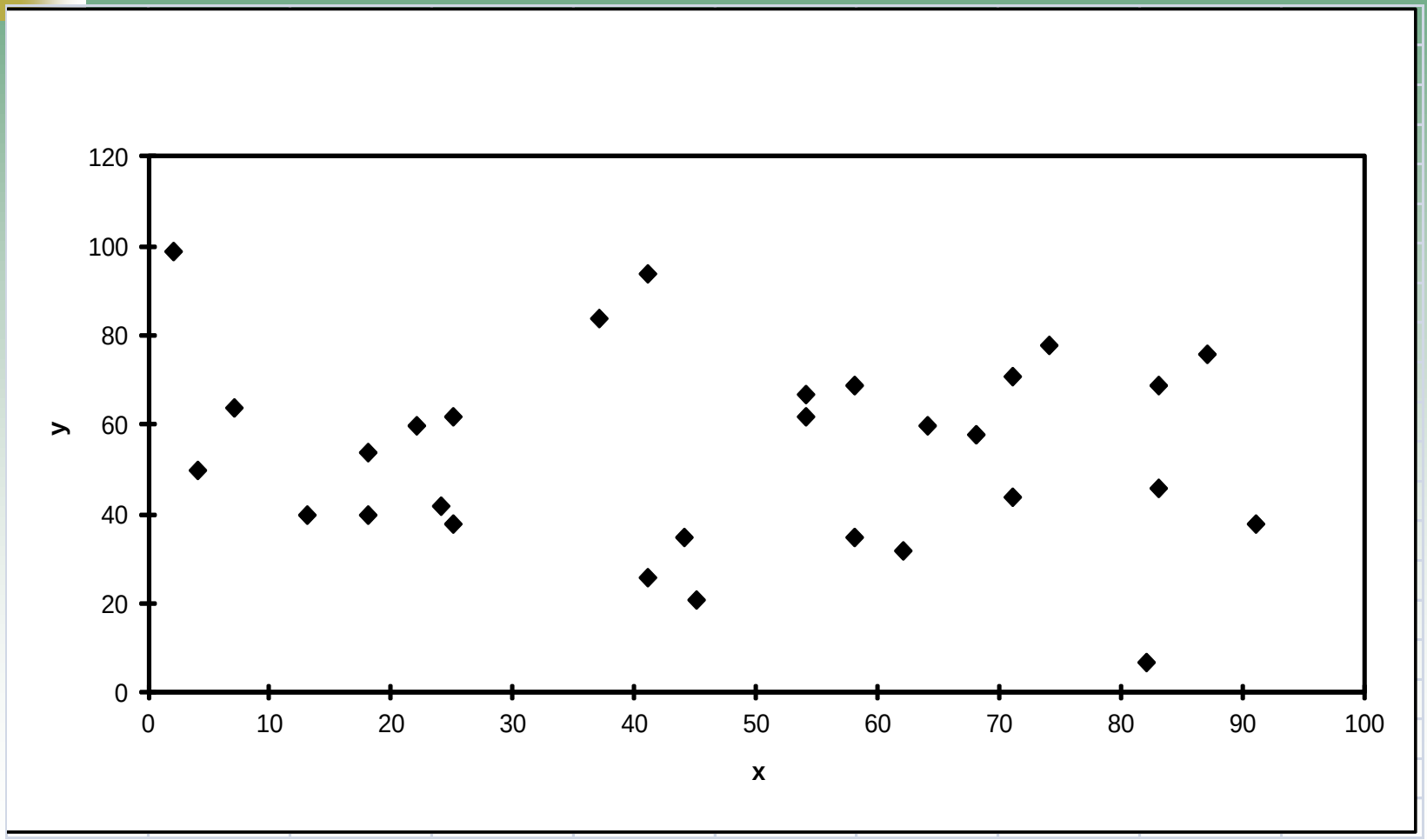
تجميع

(5,3,4,6,2)	(2,4,6,3,5)	(5,4,2,6,3)
(3,4,5,6,2)	(2,3,6,4,5)	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	(5,4,2,3,6)
(4,6,3,2,5)	(3,4,2,6,5)	(3,6,5,1,4)

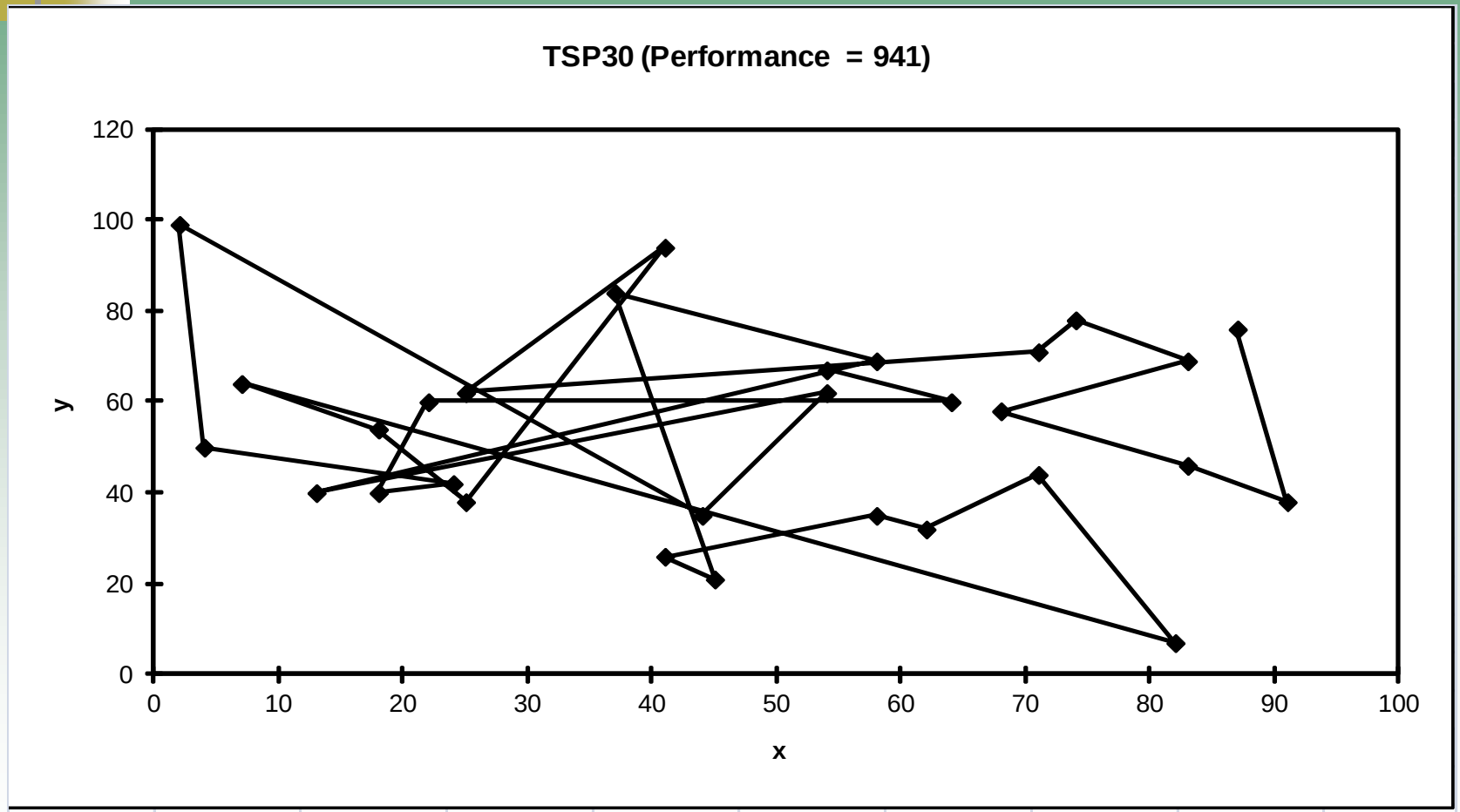
شروع دوباره

(5,3,4,6,2)	(2,4,6,3,5)	(5,4,2,6,3)
(3,4,5,6,2)	(2,3,6,4,5)	(3,4,5,2,6)
(3,5,4,6,2)	(4,5,3,6,2)	(5,4,2,3,6)
(4,6,3,2,5)	(3,4,2,6,5)	(3,6,5,1,4)

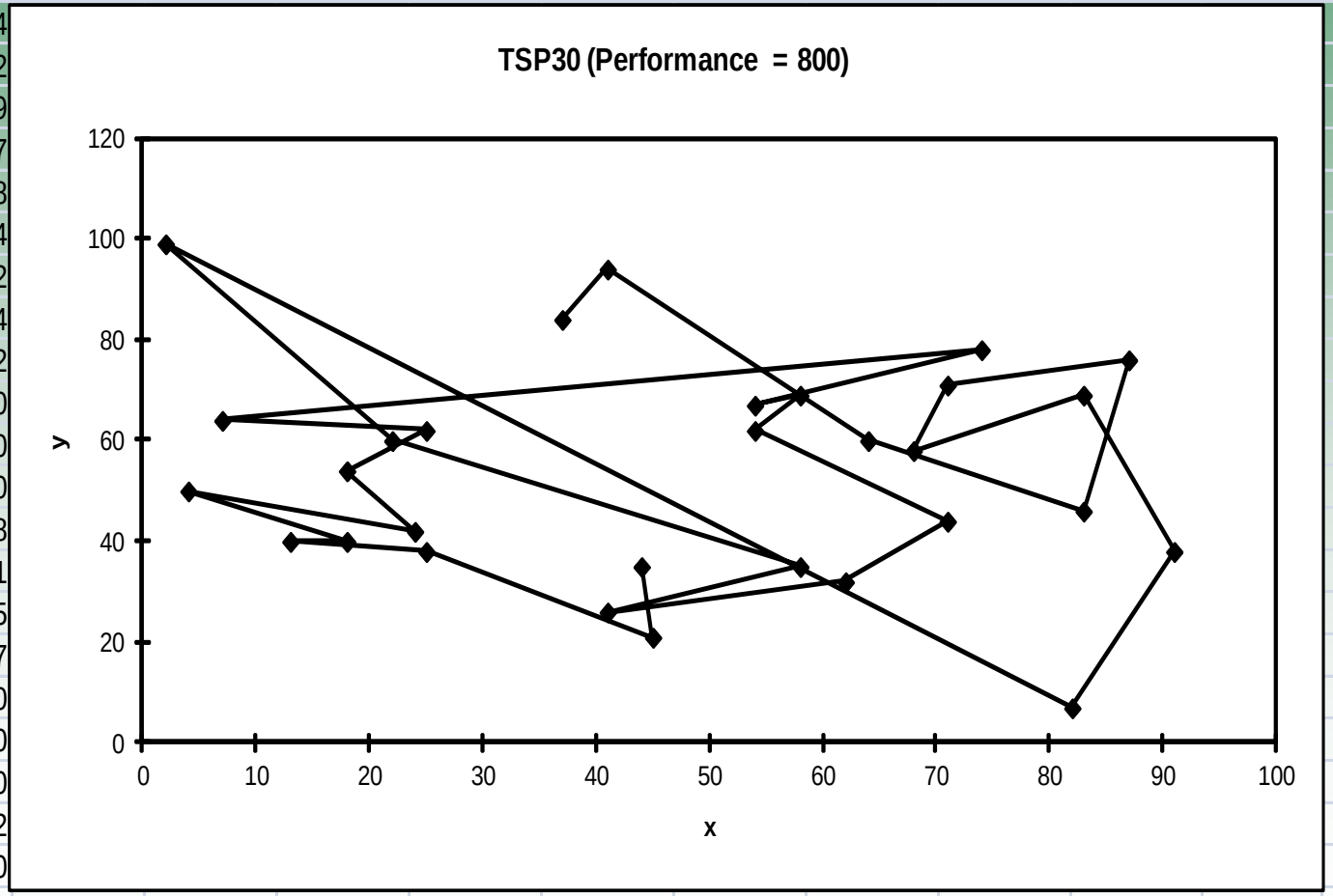
مثال TSP با ۳۰ شهر



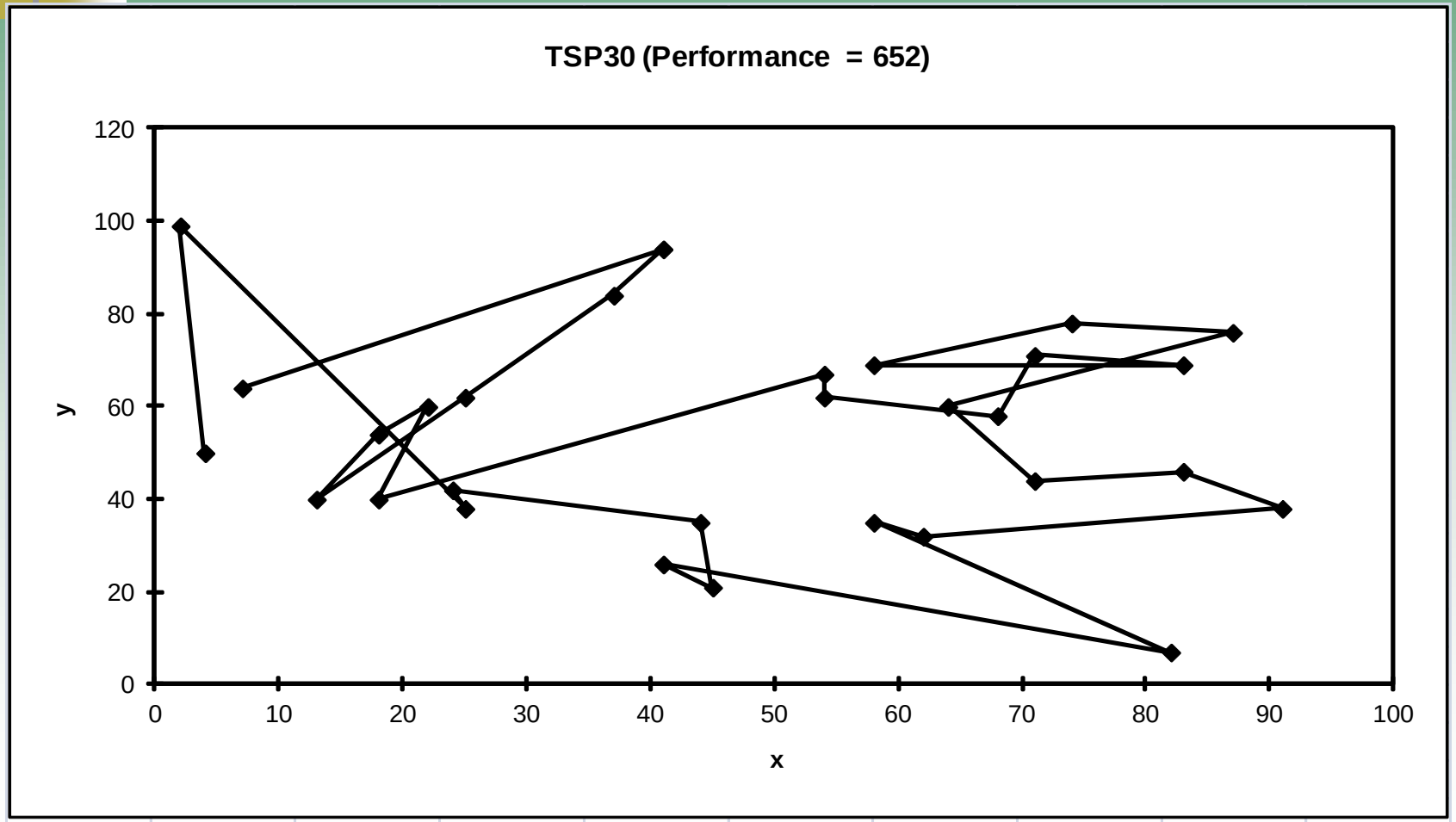
جواب i: (فاصله ۹۴۱)



جواب j: (فاصله ۸۰۰)

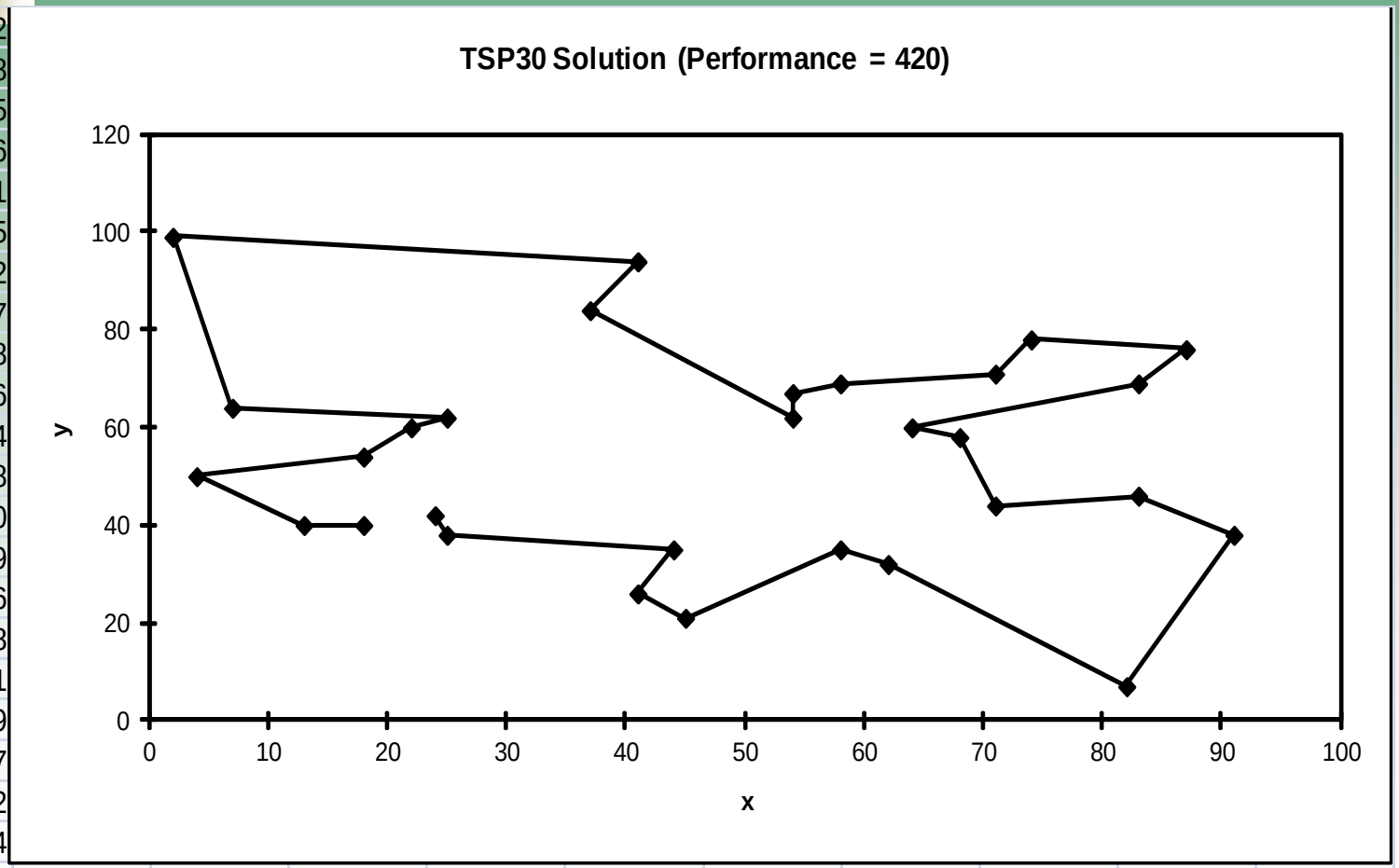


جواب k: (فاصله ۶۵۲)

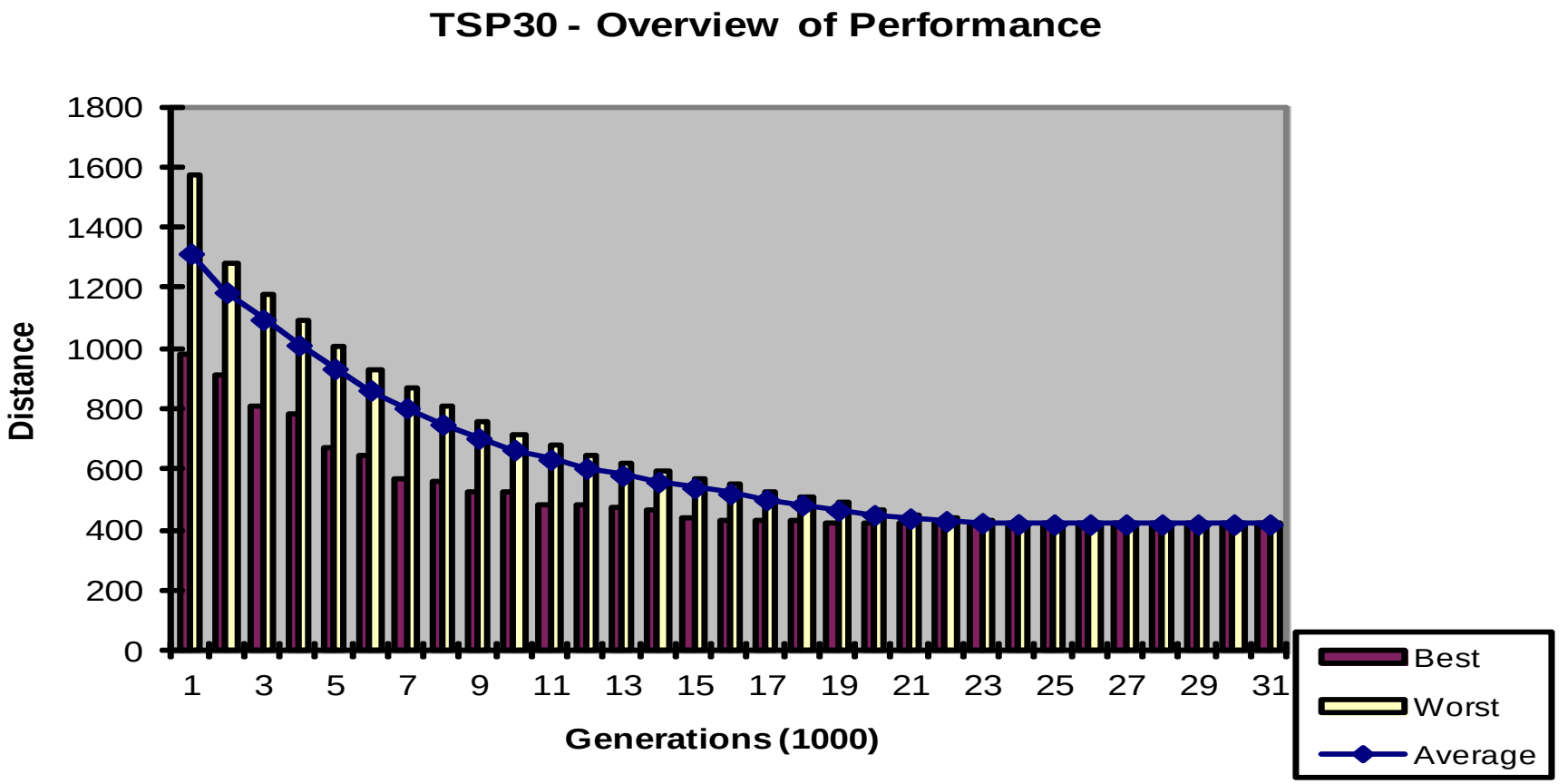


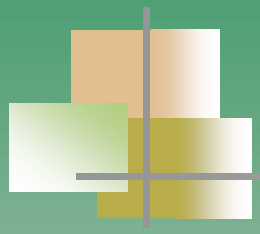
بهترین جواب : (فاصله ۴۲۰)

42
38
35
26
21
35
32
7
38
46
44
58
60
69
76
78
71
69
67
62
84
94



کارآیی


TSP30 - Overview of Performance
Distance
1800
1600
1400
1200
1000
800
600
400
200
0
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
Generations (1000)
Best
Worst
Average



مساله کوله پشتی

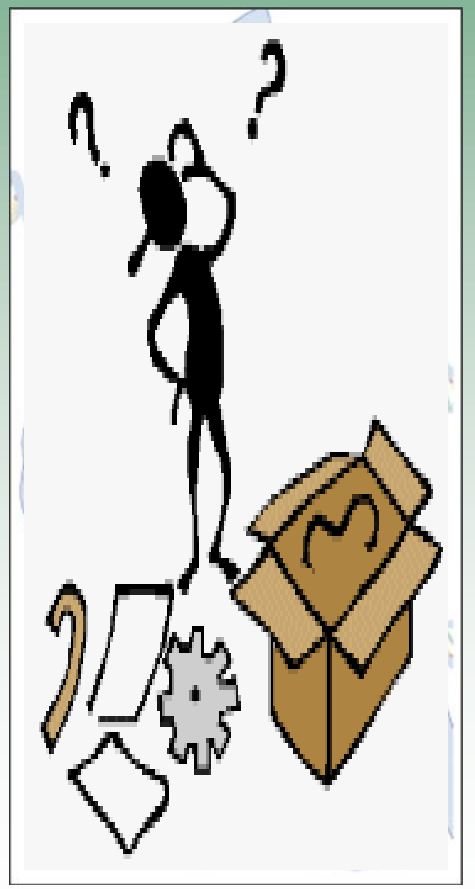
Knapsack

مساله کوله پشتی Knapsack Problem

- یک کوله پشتی با ظرفیت W و تعدادی شیء موجود است.
- هر شیء دارای یک وزن و یک ارزش است.
- می خواهیم اشیایی را انتخاب نماییم که در کوله پشتی جا شوند و بیشترین ارزش را داشته باشند. (از هر شیء حداکثر یک نسخه می توان انتخاب نمود.)

Knapsack Problem مساله کوله پشتی

- مثال:
- اشیاء : 1 2 3 4 5 6 7
- ارزش: 5 8 3 2 7 9 4
- وزن : 7 8 4 10 4 6 4
- حداکثر ظرفیت قابل تحمل کوله پشتی ۲۲



مساله کوله پشتی(کدگذاری)

- کدگذاری: ۰ = عدم وجود ۱ = وجود شیء در کوله پشتی

Chromosome: 1010110

Item.	1	2	3	4	5	6	7
Chro	1	0	1	0	1	1	0
Exist?	y	n	y	n	y	y	n

اشیا انتخاب شده => 1, 3 , 5, 6.

- تولید تصادفی n کروموزوم:

a) 0101010 1100100 0100011

مساله کوله پشتی(انتخاب و برازندگی)

0101010: Benefit= 19, Weight= 24 ✗ (a

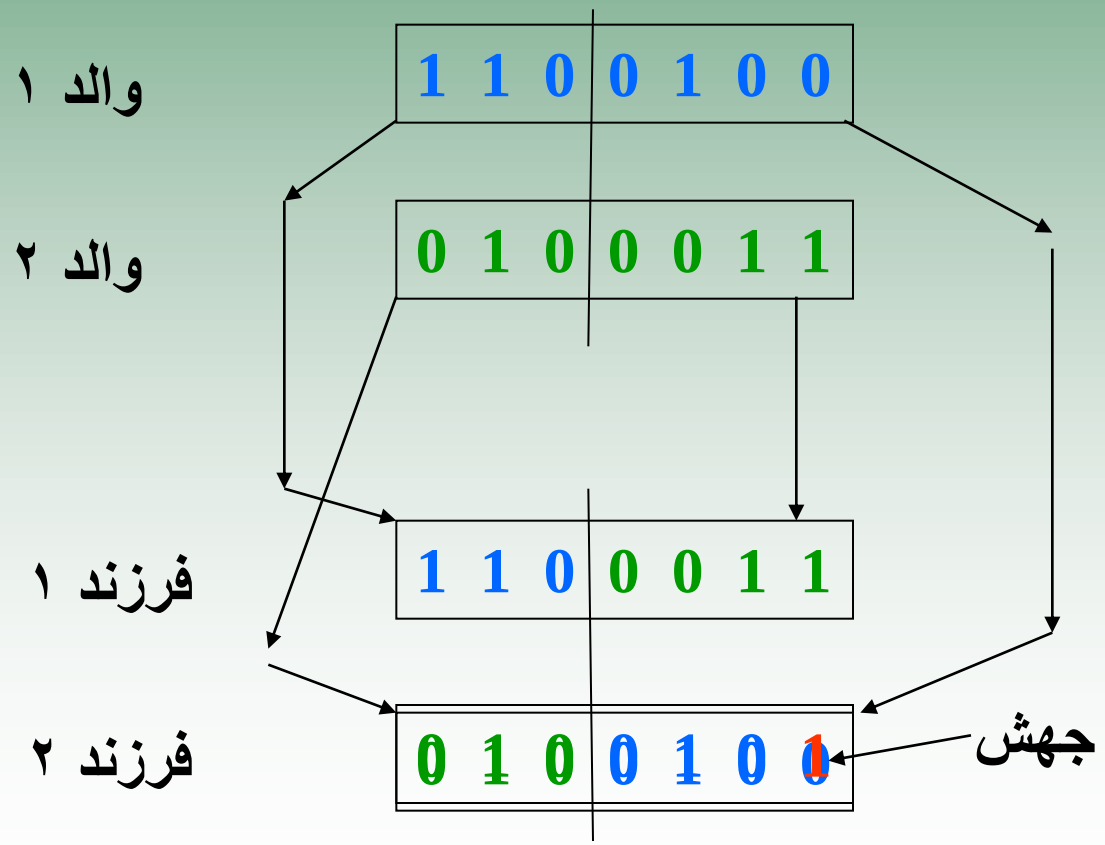
Item	1	2	3	4	5	6	7
Chro	0	1	0	1	0	1	0
Benefit	5	8	3	2	7	9	4
Weight	7	8	4	10	4	6	4

1100100: Benefit= 20, Weight= 19. ✓ (b

0100011: Benefit= 21, Weight= 18. ✓ (c

=> کروموزوم های b و c را انتخاب می کنیم.

مساله کوله پشتی(برش و جهش)



مساله کوله پشتی(جایگذاری و تست)

- فرزندان جدید را در یک جمعیت جدید قرار دهید.
- جمعیت جدید را برای اجرای بیشتر الگوریتم نگهداری نمایید.
- اگر شرط توقف برآورده شده است الگوریتم را خاتمه دهید. در غیر این صورت، عمل فوق را تکرار نمایید.

خلاصه ویژگی‌های الگوریتم ژنتیک

۱- الگوریتمهای ژنتیک به‌طور ذاتی دارای ماهیت تصادفی می‌باشند، یعنی از راه‌های مختلف به جوابهای مختلف خواهیم رسید. این کارکرد بدلیل نحوه عملکرد الگوریتم است که حدس اولیه که شامل جمعیتی از افراد است تصادفی انتخاب می‌شوند.

۲- الگوریتمهای ژنتیک برای مسائل دشوار استفاده می‌شود. مسائلی که فضای جستجو در ان بسیار وسیعتر از آن است که بخواهیم جستجوی کلاسیک و منطقی انجام بدهیم.

۳- متفاوت از بسیاری از الگوریتمهای حل مسائل عددی که فقط یک یا چند جواب خاص را به‌دست می‌دهند، الگوریتمهای ژنتیک یک جمعیت از راه‌حلها را به‌دست خواهد آورد که دارای بهترین برازش هستند و تابع هدف را بیشینه می‌کنند.

خلاصه ویژگی‌های الگوریتم ژنتیک(ادامه...)

۴-راه‌حل‌ها به شکل کروموزوم (فرد) دسته‌بندی و رمزبندی می‌شوند، که هر کروموزوم دارای مجموعه‌ای از ژنها است که خواص مختلفی را بیان می‌کنند.

۵- اصل بقای اصلح (تنازع بقا) درست مانند جهان واقعیت و عالم طبیعت، پیاده‌سازی می‌شود. یعنی افرادی از استخر ژنی یا جمعیت اولیه که شرایط تابع برازش را ارضاء نمی‌کنند، به‌طورقطع درنسلهای بعدی حضور نخواهند داشت و محکوم به فنا هستند!!

۶- GA محدودیت هایی هم دارد، مانند سایر روشهای تصادفی با GA لزوماً به جواب بهینه در تمام فضای جستجو نمی رسد بلکه جوابی که به اندازه قابل قبولی خوب باشد را می یابد . روش GA برای مسائلی مناسب است که کلیه ی روش های دیگر برای آنها بد عمل کرده اند و یا آنکه اطلاعات کافی در مورد فضای جستجوی مساله در دست نیست .

تفاوت GA با سایر روشهای جستجو

- GA بجای کد کردن پارامترها مجموعه آنها را کد می‌کند
- GA به جای جستجو برای یک نقطه بدنبال جمعیتی از نقاط می‌گردد.
- GA به جای استفاده از مشتق و یا سایر اطلاعات کمکی مستقیما از اطلاعات موجود در نتیجه بهره می‌گیرد.
- GA به جای قوانین قطعی از قوانین احتمال برای تغییر استفاده می‌کند.