انواع الگوریتم‌های ژنتیکی

الگوریتم‌های ژنتیک که نمونه اولیه آن توسط «هولند» در سال ۱۹۷۵ ارائه شد، تکامل طبیعی را در سطح ژن و کروموزوم شبیه‌سازی می‌کنند. عملکرد غالب در تولید نسل جدید، پیوند کروموزوم هاست، گرچه جهش در ژن‌ها نیز به عنوان یک عملکرد ثانوی به کار می‌رود.[13]

انواع بسیاری برای GA شناخته شده است که در اینجا به تعدادی از آنها اشاره می‌کنیم:

۱- الگوریتم ژنتیک سری[۱]

۲- الگوریتم ژنتیک موازی[۲]

۳- الگوریتم ژنتیک آشفته[۳]

٤- الگوریتم ژنتیک هیبرید[٤]

۵- الگوریتم ژنتیک خودسازمان[۵]

٦- الگوریتم ژنتیک زایشی[٦]

۷- الگوریتم ژنتیک حالت دائمی[۷]

[3]

۲-۱۹-۱- الگوریتم ژنتیکی سری

الگوریتم ژنتیک سری همان الگوریتم ژنتیک معمولی است که در مقابل نوع موازی سری نام گرفته است.

Sequential GA

Parallel GA

Messy GA

Hybrid GA

Adaptive GA

Generational GA

Steady State GA

تکامل یک پروسه بهینه‌سازی مبتنی بر تغییرات تصادفی تدریجی نمونه‌های مختلف در یک جمعیت و انتخاب اَحسن آنهاست. با مدل سازی این پروسه می‌توان یک تکنیک بهینه‌سازی آماری را به دست آورد که امروزه در مسائل پیچیده مختلف و بخصوص مسائل طراحی، کارائی خود را نشان داده است. در الگوریتم ژنتیک به عنوان یکی از الگوریتم‌های تکاملی، اثر کدهای ژنتیکی در ترکیب و انتقال اطلاعات و همچنین فرآیند انتخاب طبیعی بر اساس سازگاری موجود با شرایط زیست‌محیطی مدل‌سازی است. در این الگوریتم نمونه‌هایی که در پروسه تکاملی قرار می‌گیرند، جواب‌های مختلف در فضاهای جواب هستند. متناظر هر جواب (نقطه در فضای جواب)، یک نمونه ژنتیکی (Genotype) به صورت یک رشته از کاراکترها (ژن‌ها)، نسبت داده می‌شود.

الگوریتم ژنتیک در هر تکرار محاسباتی (نسل) روی جمعیتی از رشته‌ها عمل می‌کند.

تغییرات تصادفی روی مجموعه نمونه‌ها، از طریق اعمال مدل‌های ایده‌آل فرآیندهای ژنتیکی روی رشته‌ها انجام می‌شود، اما انتخاب طبیعی براساس نمود رفتاری (Phenotype) هر رشته انجام می‌شود.

بدین مفهوم که رشته‌ها رمزگشایی می‌شوند و جواب‌های مختلف از نظر عملکرد بر اساس تابع هدف ارزیابی شده و انتخاب، بر مبنای این ارزیابی و تصادف انجام می‌شود.[13]

۲-۱۹-۲- الگوریتم ژنتیکی موازی

تا کنون دو مدل اصلی در الگوریتم ژنتیک موازی مطرح گشته است:

- مدل جزیره‌ای[1]
- مدل همسایگی[2]

در مدل جزیره‌ای چندین زیرجمعیت مجزّا مطابق با الگوریتم ژنتیک معمولی تکامل می‌یابد و هر از چند گاهی زیر جمعیت‌های همسایه، بهترین کروموزوم یکدیگر را معاوضه می‌کنند.

در مدل همسایگی یک مدل منفرد تکامل می‌یابد. هر کروزوزوم این جمعیت در یک سلول از یک شبکه مشبک قرار دارد و الگوریتم ژنتیک سری، به صورت مجزا به هر سلول و همسایگانش که بر حسب

---

[1] Island Model

[2] Find Grained Model

شعاع همسایگی مشخص می‌شوند، اعمال می‌گردد. شبکه به صورت تروید[1] در نظر گرفته می‌شود تا از اثرات مرزی اجتناب گرد.

مقایسه‌ای بین رفتار این الگوریتم با الگوریتم‌های معمولی نشان می‌دهد که مدل همسایگی به خاطر مکانیزیم انتخاب محلی که از فشار انتخاب می‌کاهد، کاوش دقیقتری را در فضای جستجو فراهم می‌سازد.

از این جهت در مسائل ساده‌تر بدون بهبودی در عملکرد روش، تنها بار محاسباتی اضافی‌تر تحمل می‌گردد. ولی مسائل مشکلتر از این طریقه جستجو سود خواهد برد.

شعاع همسایگی مناسب نیز به مسأله مورد حل بستگی دارد و حتی همسایگی‌های کوچک به شعاع یک یا دو، انتخابی مقاوم و اطمینان از رفتاری خوب را فراهم می‌سازند.[13]

مطالعهٔ دیگر انواع GA به خوانندهٔ محترم واگذار می‌شود که با مراجعه به منبع [11] می‌تواند اطلاعات کافی و مناسبی دریافت کند.

۲-۲۰- مقایسه الگوریتم ژنتیک با سیستم‌های طبیعی

سیستم‌های طبیعی	الگوریتم ژنتیک
کروموزوم بسته‌های ژنی هستند که اطلاعات وراثتی را از نسلی به نسل دیگر عیناً انتقال می‌یابند.	پاسخ‌های ممکن مسأله به صورت رشته‌های عددی رمزگذاری شده است.
محیط شرایط محیطی را که جمعیت در آن قرار دارد.	تابع برازش مسأله به صورت یک رابطه ریاضی در آمده که تابع برازش می‌نامند.
اصل انتخاب طبیعی معیار بقای موجود زنده و تکثیر آن، سازش با محیط است.	تکثیر هر رشته جمعیت را به عنوان متغیر تابع برازش در نظر گرفته و مقدار تابع برازش هر رشته محاسبه می‌شود، متناسب با مقدار تابع برازش، رشته‌های جمعیت جدید انتخاب می‌شود.
تقاطع	تقاطع(crossover)

[1] Triode

در نتیجه تقاطع یا تبادل قسمتی از کروموزوم‌ها مبادله ژن‌های پیوسته صورت می‌گیرد.	رشته‌های جمعیت به صورت دو به دو مزدوج می شوند. زوج رشته‌ها از یک نقطه قطع می شوند. نیم بخشهای بین دو رشته تعویض می‌شوند.
جهش جانشین شدن ژنی به جای ژن دیگر در طول زنجیره DNA با تغییرات ایجاد شده در ژن	جهش(mutation) یک بیت از رشته عددی به صورت تصادفی انتخاب می‌شود و دچار تغییر می‌گردد.
ایجاد نسلهای جدید و تکامل موجودات	تکرار مراحل فوق بعد از هر مرحله تکثیر

[13]

۲-۲۱- نقاط قوّت الگوریتم‌های ژنتیک

اولین و مهمترین نقطه قوّت این الگوریتم‌ها این است که الگوریتم‌های ژنتیک ذاتاً موازی‌اند. اکثر الگوریتم‌های دیگر موازی نیستند و فقط می‌توانند فضای مسأله مورد نظر را در یک جهت در یک لحظه جستجو کنندو اگر راه‌حل پیدا شده یک جواب بهینه محلی باشد و یا زیر مجموعه‌ای از جواب اصلی باشد باید تمام کارهایی که تا به حال انجام شده را کنار گذاشت و دوباره از اول شروع کرد. از آنجایی که GAچندین نقطه شروع دارد، در یک لحظه می‌تواند فضای مسأله را از چند جهت مختلف جستجو کند. اگر یکی به نتیجه نرسید سایر راه‌ها ادامه می‌یابند و منابع بیشتری را در اختیارشان قرار می‌گیرد. در نظر بگیرید: همه ۸ عدد رشته باینری یک فضای جستجو را تشکیل می‌دهند، که می‌تواند به صورت ******** نشان داده شود.رشته ۰۱۱۰۱۰۱۰ یکی از اعضای این فضاست. همچنین عضوی از فضاهای *******۰ و *******۰۱ و ******۰** و *۱*۱*۱*۰ و **۰*۱*۰۱ و الی آخر باشد.

به دلیل موازی بودن و این که چندین رشته در یک لحظه مورد ارزیابی قرار می‌گیرند الگوریتمهای ژنتیک برای مسائلی که فضای راه‌حل بزرگی دارند بسیار مفیدند. اکثر مسائلی که این گونه‌اند به عنوان غیرخطی شناخته شده‌اند. در یک مسأله خطی، Fitness هر عنصر مستقل است، پس هر تغییری در یک قسمت بر تغییر و پیشرفت کل سیستم تأثیر مستقیم دارد. می‌دانیم که تعداد کمی از مسائل دنیای واقعی به

صورت خطی‌اند. در مسائل غیرخطی تغییر در یک قسمت ممکن است تاثیری ناهماهنگ بر کل سیستم و یا تغییر در چند عنصر تاثیر فراوانی بر سیستم بگذارد. خوشبختانه موازی بودن GA باعث حل این مسأله می‌شود و در مدت کمی مشکل حل می‌شود. مثلاً برای حل یک مسأله خطی ۱۰۰۰ رقمی ۲۰۰۰ امکان حل وجود دارد ولی برای یک غیرخطی ۱۰۰۰ رقمی 2^{1000} امکان.

یکی از نقاط قوت الگوریتم‌های ژنتیک که در ابتدا یک کمبود به نظر می‌رسد این است که: GA ها هیچ چیزی در مورد مسائلی که حل می‌کنند نمی‌دانند و اصطلاحاً به آنها «ساعت ساز نابینا»[1] می‌گوییم. آنها تغییرات تصادفی را در راه‌حل‌های کاندید شان می‌دهند و سپس از تابع برازش برای سنجش این که آیا آن تغییرات پیشرفتی ایجاد کرده‌اند یا نه، استفاده می‌کنند. مزیت این تکنیک این است که به GA اجازه می‌دهد تا با ذهنی باز شرو به حل مسائل کند. از آنجایی که تصمیمات آن اساساً تصادفی است، بر اساس تئوری همه راه‌حل‌های ممکن به روی مسأله باز است، ولی مسائلی که محدود و به اطلاعات هستند باید از راه قیاس تصمیم بگیرند و در این صورت بسیاری از راه‌حل‌های نو و جدید را از دست می‌دهند.

یکی دیگر از مزایای الگوریتم این است که آنها می‌توانند چندین پارامتر را همزمان تغییر دهند. بسیاری از مسائل واقعی نمی‌توانند محدود به یک ویژگی شوند تا آن ویژگی ماکسیمم شود و باید چند جانبه در نظر گرفته شوند. GAها در حل این گونه مسائل بسیار مفیدند، و در حقیقت قابلیت موازی کار کردن آنها این خاصیت را به آنها می‌بخشد. و ممکن است برای یک مسأله ۲ یا چند راه‌حل پیدا شود، که هر کدام با در نظرگرفتن یک پارامتر خاص به جواب رسیده‌اند.

به طور خلاصه مزایای الگوریتم ژنتیک را می‌توان در موارد زیر برشمرد:

۱- با متغیرهای پیوسته و هم گسسته می‌تواند عمل بهینه‌سازی را انجام دهد.

۲- نیازی به محاسبه مشتق توابع ندارد.

۳- بطور همزمان می‌تواند تمامی ناحیه جستجو شونده و سیع تابع هزینه را جستجو کند.

٤- قادر به بهینه سازی مسائل با تعداد متغیرهای زیاد می‌باشد.

---

۱ Blind Watchmakers

۵- قابل اجرا از طریق کامپیوترهای موازی است.

٦- توابع هزینه‌ای که بسیار پیچیده باشند نیز از این طریق قابل بهینه سازی می‌باشند و الگوریتم در اکسترمم محلی به دام نمی‌افتد.

۷- قادر است تا چند جواب بهینه را بطور همزمان به دست آورد نه فقط یک جواب.

۸- الگوریتم‌های ژنتیک بر روی مجموعه‌ای از راه‌حل‌ها اعمال می‌شوند و نه بر روی یک راه‌حل خاص.

۹- قادر است تا متغیرها را کد بندی نموده و بهینه سازی را با متغیرهای کدبندی شده انجام دهد. کد بندی سرعت همگرایی الگوریتم را افزایش می‌دهد.

۱۰- الگوریتم توانایی کار کردن با داده‌های عددی تولید شده و داده‌های تجربی را علاوه بر توابع تحلیلی دارد.

۱۱- فرآیند ارائه شده توسط الگوریتم‌های ژنتیک بر روی فضایی از مجموعه نمایندگان یا همان فضای کروموزوم‌ها اعمال می‌گردد و نه بر روی خود فضای راه‌حل‌ها.

۱۲- الگوریتم‌های ژنتیک از قوانین انتقالی احتمالی بجای قوانین انتقالی قطعی استفاده می‌کنند، بدین معنا که حرکت آن در هر نقطه از الگوریتم کاملا احتمالی بوده و بر اساس قطعیت صورت نمی‌پذیرد. این امر از مزایای مهم این روش بوده و از افتادن سیستم در کمینه محلی جلوگیری می‌نماید.

البته میزان احتمال به گونه‌ای است که احتمال حرکت به سمت مسأله بیشتر از احتمال حرکت آن به سمت مخالف جواب می‌باشد.

۱۳- تنها ملاک ارز شیابی و سنجش میزان شایستگی هر راه‌حل توسط الگوریتم‌های ژنتیک، مقدار تابع شایستگی آن در فضای کروموزوم‌ها می‌باشد و نه معیارهای مورد نظر در سطح فضای راه‌حل‌ها.

۱٤- برای حل برخی از مسائلی از رده NP-Hard نیز استفاده می‌شود.

۱۵- این الگوریتم بیشتر در مسائل بهینه سازی و امثالهم بکار می‌رود.

۲-۲۲- محدودیت‌های GAها

یک مشکل چگونگی نوشتن عملگر ارزیاب است که منجر به بهترین راه‌حل برای مسأله شود. اگر این کارکرد برازش به خوبی و قوی انتخاب نشود ممکن است باعث شود که راه‌حلی برای مسأله پیدا نکنیم یا مسأله‌ای دیگر را به اشتباه حل کنیم. به علاوه برای انتخاب تابع مناسب برای ارزیاب، پارامترهای دیگری مثل اندازه جمعیت، نرخ ترکیب، قدرت و نوع انتخاب هم باید مورد توجه قرار گیرند.

مشکل دیگر، که آن را «**نارس**» می‌نامیم این است که اگر یک ژنوم که فاصله‌اش با سایز ژنوم‌های نسل‌اش زیاد باشد. (خیلی بهتر از بقیه باشد) خیلی زود دیده می‌شود (ایجاد می‌شود) ممکن است محدودیت ایجاد کند و راه‌حل را به سوی جواب بهینه محلی سوق دهد. این اتفاق معمولاً در جمعیت‌های کم اتفاق می‌افتد. روش‌های Rank Scaling , Tournament Selection بر این مشکل غلبه می‌کنند.[13]

۲-۲۳- استراتژی برخورد با محدودیت‌ها

بحث دیگری که در اجرای الگوریتم ژنتیک وجود دارد چگونگی برخورد با محدودیت‌های مسأله می‌باشد، زیرا عملگرهای ژنتیک مورد استفاده در الگوریتم باعث تولید کروموزوم‌های غیرموجّه می‌شود. «میکالویچ» چند تکنیک معمول جهت مواجهه با محدودیت‌ها تقسیم‌بندی نموده است که در ادامه به چند تا از آنها اشاره می‌شود.[13]

۲-۲۳-۱- استراتژی اصلاح عملگرهای ژنتیک

یک روش برای جلوگیری از تولید کروموزوم غیرموجّه این است که عملگر ژنتیکی طوری تعریف گردد که پس از عمل بر روی کروموزم‌ها، کروموزوم تولید شده نیز موجّه باشد. در این حالت یکسری

مشکلات وجود دارد. مثلاً پیدا کردن عملگری که دارای شرط فوق باشد بسیار دشوار بوده و از مسأله دیگر متفاوت می‌باشد.[13]

۲-۲۳-۲- استراتژی رَدّی

در این روش پس از تولید هر کروموزوم آنرا از نظر موجّه بودن تست کرده و در صورت غیرموجّه بودن حذف می‌گردد. این روش بسیار ساده و کارا می‌باشد.[13]

۲-۲۳-۳- استراتژی اصلاحی

در این روش به جای اینکه کروموزوم غیرموجّه حذف گردد تبدیل به یک کروموزوم موجّه می‌شود. این روش نیز مانند روش اول به مسأله وابسته بوده و یافتن فرآیند اصلاح گاهی بسیار پیچیده می‌باشد.[13]

۲-۲۳-٤- استراتژی جریمه‌ای

در این روش بر خلاف سه روش قبل که از ورود جواب‌های غیرموجّه جلوگیری می‌کردند، جواب غیرموجّه با احتمال کم امکان حضور می‌یابند. سه روش فوق دارای این عیب بودند که به هیچ نقطه‌ای بیرون از فضای موجّه توجّه نمی‌کردند، اما در بعضی مسائل بهینه‌سازی، جواب‌های غیرموجّه درصد زیادی از جمعیت را اشغال می‌کنند. در چنین شرایطی اگر جستجو فقط در ناحیه موجّه انجام گیرد شاید یافتن جواب موجّه خیلی وقت‌گیر و مشکل باشد.

استراتژی جریمه‌ای از متداولترین تکنیک‌های مورد استفاده برای سر و کار داشتن با جواب‌های غیرموجّه می‌باشد که در آن ابتدا محدودیت‌های مسأله در نظر گرفته نمی‌شوند پس برای هر تخلّف از محدودیت‌ها یک جریمه اختصاص داده می‌شود که این جریمه در تابع هدف قرار می‌گیرد.

مسأله اصلی چگونگی انتخاب یک مقدار مناسب برای مقدار جریمه می‌باشد تا در حل مسائل به ما کمک نماید.

نکته‌ای که در روش جریمه وجود دارد این است که یک جواب غیرموجّه به سادگی حذف نمی شود زیرا ممکن است در ژنهای آن اطلاعات مفیدی وجود داشته باشد که با اندکی تغییر به جواب بهینه تبدیل شود.[13]

۲-۲٤- بهبود الگوریتم ژنتیک

برای بهبود دادن GA می‌توانیم تغییرات زیر را اعمال کنیم:

۱- استفاده از بهینه‌گر محلی.

۲- تغییر پارامترهائی از قبیل تغییر جمعیت اولیه، آهنگ جهش و کسر ادغام.

۳- تغییر GA باینری به پیوسته و بالعکس.

[13]

۲-۲۵- چند نمونه از کاربردهای الگوریتم‌های ژنتیک

نرم افزار شناسایی چهره (شناسایی چهره با استفاده از تصویر ثبت شده. در این روش، شناسایی چهره بر اساس فاصله اجزای چهره و ویژگی‌های محلی و هندسی صورت می‌گیرد که تغیرات ناشی از گریم، تغییرات نور و افزایش سن کمترین تأثیر را خواهد داشت. همچنین گراف‌ها برای چهره‌های جدید با استفاده از الگوریتم‌های ژنتیک ساخته شده و با استفاده از یک تابع تشابه، قابل مقایسه با یکدیگر هستند که این امر تأثیر به سزایی در افزایش سرعت شناسایی خواهد داشت).

توپولوژی‌های شبکه‌هایی کامپیوتری توزیع شده.

بهینه سازی ساختار مولکولی شیمیایی(شیمی).

مهندسی برق برای ساخت آنتن‌های خمیده[۲].

مهندسی نرم‌افزار.

بازی‌های کامپیوتری.

مهندسی مواد.

مهندسی سیستم.

رباتیک[۳] .

تشخیص الگو و استخراج داده[٤] .

حل مسأله فروشنده دوره گرد.

آموزش شبکه‌های عصبی مصنوعی.

یاددهی رفتار به رباتها با GA.

یادگیری قوانین فازی با استفاده از الگوریتم‌های ژنتیک.

یک مثال ساده.

ǒCrooked-Wire Genetic Antenna

ǑRobotics

ǐData Mining

مثال: فرض کنید تابع $f(x) = -x^2 + 6 - 3$ را داریم می‌خواهیم برای $0 \le x \le 15$ و $x \in N$ ماکزیمم تابع را بیابیم.

مراحل حل مسأله به روش الگوریتم ژنتیک به شرح زیر است:

۱- کد کردن (Encoding)

برای اینکه ما جواب‌های ممکن یعنی $0 \le x \le 15$ را کدگذاری کنیم یکی از روش‌هایی که در پیش داریم تبدیل هر عدد به یک عدد باینری متناظر است. چون بیشترین جواب ممکن ما ۱۵ است و متناظر باینری آن ۱۱۱۱ چهار بیتی است لذا تمام جواب‌های ما (کروموزوم ها) دارای طولی برابر $l = 4$ می‌باشد.

۲- تابع ارزش(Evaluation)

تابعی که مقداری برای Fitness حساب می‌کند مستقیماً از خود تابع $f(x)$ به دست می‌آید.

یعنی اگر به ازای یک $b,a : f(b) > f(a)$ باشد آنگاه b دارای Fitness بیشتری است. یا اینکه b از a بهتر است.

۳- انتخاب (Selection)

فرض کنیم که در آن جمعیت اولیه m فرد که m<n دو عدد ۳ و ۸ دارای Fitness بیشتری نسبت به جمعیت خود بودند و احتمال انتخاب بیشتری داشتند. بعد از انتخاب این دو عدد به عنوان Parent الگوریتم وارد مراحل تولید نسل جدید می‌شود.

$$3 \to 0011 \qquad f(3) = 6$$

$$8 \to 1000 \qquad f(8) = -19$$

٤- ترکیب (Crossover)

فرض کنیم که ترکیب عمل زیر را انجام دهد.

۰۰۱۰ Point=3 ۰۰۱۱

۱۰۰۱ → ۱۰۰۰

۵- جهش (**Mutation**)

و نیز عملگر جهش:

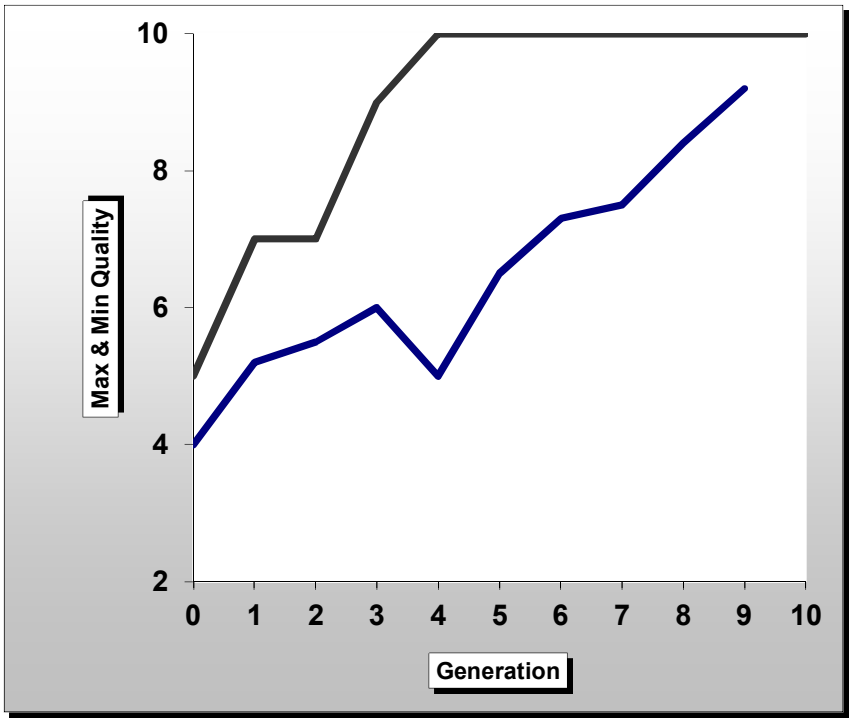
0110=6 Point=2 ۰۰۱۰

1101=13 → ۱۰۰۱

حال کروموزوم‌های متولد شده جزو نسل جدید به حساب می‌آیند و به جمعیت اولیه افزوده می‌شوند و جواب‌های با Fitness پایین حذف می‌شـــوند و الگوریتم دوباره با n فرد به کار خود ادامه می‌دهد. یک نکته لازم به تذکر اســـت که ممکن اســـت بهترین جواب (برای مثال x=6) طی مراحل بعد به نحوی از بین برود. برای جلوگیری از این امر بعد از هر بار انجام الگوریتم بهترین جواب را در جایی کنار می گذاریم (far) تا همیشه بهترین جواب حاصل از هر بار اجرای الگوریتم را داشته باشیم.

معیارهای مختلفی را می‌توان برای توقف الگوریتم در نظر گرفت و معمولاً چند معیار برای توقف اسـتفاده می‌شـود تا احتمال‌های مختلف وقوع پیشـامدها در طی اجرای الگوریتم حسـاب شـوند. یک معیار می‌تواند این باشـد که بهترین جواب را بعد از اجرای تعداد مشـخصـی بار از الگوریتم تغییر ندهد. یا معیار دیگر اینکه میانگین برازندگی جواب‌های موجود در جمعیت جاری همان برازندگی بهترین جواب یا بســیار نزدیک به آن باشد. یا اینکه می‌توانیم از پیش قرارداد کنیم که الگوریتم به تعداد مشخصی اجرا شود. معمولاً روتین‌های توقّف مختلف است و بستگی به پیچیدگی و چگونگی مسأله دارد. در اینجا ما برای تعریف تابع ارزشیابی از خود تابع استفاده کردیم و لزومی برای نسبت دهی یک مقدار به عنوان کیفیت[۵] نبود.

۵Quality



شکل ۲-۲۱- نمودار بررسی رابطه های جمعیت، کیفیت جواب و معیار توقف با یکدیگر.

اگر ما این مراحل را به گونه‌ای دنبال کنیم که بعد از هر بار اجرا جواب‌های بد حذف و جواب‌های نسل جدید که احتمالا Fitness بالایی دارند جایگزین آنها شوند و به مرور فاصله Fitness بین Min و Max کم می‌شود و میانگین جواب‌ها به بهترین جواب میل می‌کند و این نشان می‌دهد که به طور کلی به بهترین جواب نزدیک می‌شویم.

سؤالی که در اینجا مطرح می شود اینست که اگر در مثال قبل به جای فضای گسسته فضای پیوسته $[0,15]$ مطرح می‌شد الگوریتم به چه شکلی انجام می‌گرفت؟

در این حالت ما یک ε تعریف می‌کنیم تا هر بیت جواب در مراحلی مثل Mutation به اندازه ε تغییر کند. بدیهی است هر چقدر ما این ε را کوچکتر فرض کنیم میزان محسابات و تکرار الگوریتم بالا می‌رود و به همین دلیل زمان اتمام الگوریتم بالا می‌رود. عامل دیگری که در زمان مؤثر است n اولیه در جمعیت اولیه است که با زیاد فرض کردن n زمان اتمام بالا می‌رود.[13]