الگوریتم از روش memetic استفاده می کند که یک روش بهینه سازی تکاملی مبتنی بر جمعیت است.

الگوریتم Memetic یک روش بهینه سازی ترکیبی از الگوریتم های فراابتکاری و الگوریتم های محلی است. این الگوریتم از مزایای هر دو نوع الگوریتم بهره می برد. الگوریتم های فراابتکاری مانند الگوریتم ژنتیک قابلیت جستجوی گسترده در فضای جواب دارند اما سرعت همگرایی کمی دارند. الگوریتم های محلی مانند الگوریتم شبیه سازی تبرید می توانند به سرعت به یک بهینه محلی همگرا شوند اما در بهینه های محلی اشتباه گیر می افتند. الگوریتم Memetic با ترکیب این دو می تواند از مزایای هر دو بهره ببرد.

الگوریتم Memetic شامل یک جمعیت اولیه از راه حل هاست که با استفاده از عملگرهای الگوریتم ژنتیک مانند انتخاب، ترکیب و جهش تکامل می یابد. سپس بر روی هر راه حل، یک الگوریتم بهینه سازی محلی مانند شبیه سازی تبرید اعمال می شود تا راه حل به بهینه محلی نزدیک شود. با ترکیب این دو مرحله، الگوریتم Memetic هم توانایی جستجوی گسترده دارد و هم می تواند به سرعت همگرا شود. این الگوریتم برای بهینه سازی ترکیباتی پیچیده مناسب است.

در این الگوریتم یک تابع هدف f(x) در نظر گرفته می شود که می خواهیم آن را بهینه کنیم. x برداری از متغیرهای تصمیم است.

الگوریتم شامل این مراحل است:

۱. یک جمعیت اولیه از راه حل ها ایجاد می شود.

۲. برای هر راه حل، مقدار تابع هدف محاسبه می شود.

۳. با استفاده از عملگرهای تکاملی مثل انتخاب، ترکیب و جهش، جمعیت جدیدی ایجاد می شود.

۴. مراحل ۲ و ۳ تکرار می شود تا جواب بهینه پیدا شود.

در این برنامه، تابع f به عنوان تابع هدف در نظر گرفته شده و الگوریتم سعی می کند مقدار آن را حداقل کند.

خروجی الگوریتم یک بردار x است که مقدار f(x) را حداقل می کند.

نماد ریاضی

با سلام، نمادهای ریاضی مورد استفاده در این برنامه عبارتند از:

f(x): تابع هدف

x: بردار متغیرهای تصمیم

n: تعداد اعضای جمعیت

np: شماره جمعیت

x, y: بردارهای راه حل

fx, fy: مقدار تابع هدف برای x و y

grad: بردار گرادیان تابع هدف

d: بردار جهت حرکت

ps: حاصلضرب نقطه‌ای بردار جهت و گرادیان

step: مقدار گام

iter: تعداد تکرارها

fgcnt: تعداد محاسبات تابع هدف

ginf: نرم بی‌نهایت گرادیان

eps: حد خطا

alpha: ضریب به روزرسانی

در مجموع، الگوریتم سعی می‌کند مقدار f(x) را با حرکت در جهت مخالف گرادیان و به‌کارگیری تکنیک‌هایی مثل backtracking بهینه کند.

Backtracking

backtracking در الگوریتم بهینه‌سازی memetic به این شکل استفاده می‌شود:

در هر تکرار الگوریتم، یک گام در جهت مخالف گرادیان برداشته می‌شود تا به نقطه‌ی بهینه‌ی محلی نزدیک شویم. اما اگر این گام منجر به افزایش تابع هزینه شود، یعنی دور شده‌ایم، آنگاه backtracking انجام می‌شود:

۱. گام قبلی کاملاً برگردانده می‌شود (مختصات به حالت قبل باز می‌گردند)

۲. اندازه گام کاهش می‌یابد (مثلاً به نصف)

۳. الگوریتم از نقطه جدید شروع به حرکت در جهت گرادیان می‌کند.

با این کار در صورتی که مسیر اشتباه بوده، برگشته و مسیر دیگری را امتحان می‌کنیم.

مزایای backtracking:

- اجتناب از گیر افتادن در بهینه‌های محلی اشتباه

- افزایش احتمال همگرایی به بهینه جهانی

- بهبود سرعت همگرایی در مسائل پیچیده

معایب آن:

- افزایش پیچیدگی محاسباتی

- ممکن است منجر به نوسانات زیاد شود

در مجموع، backtracking می‌تواند باعث بهبود عملکرد الگوریتم‌های بهینه‌سازی شود اما پیاده‌سازی آن نیازمند توجه ویژه است.

این برنامه یک الگوریتم بهینه‌سازی از نوع memetic اجرا می‌کند و خروجی زیر را تولید می‌کند:

- x: بردار متغیرهای تصمیم بهینه که مقدار تابع هدف f را به حداقل می‌رساند

- costdata: برداری حاوی اطلاعات زیر در مورد فرایند بهینه‌سازی:

- np: شماره جمعیت

- iter: تعداد تکرارهای انجام شده

- fgcnt: تعداد دفعات محاسبه تابع هدف

- zaman: زمان اجرای الگوریتم (ثانیه)

- fxmin: مقدار بهینه تابع هدف

در مجموع، این برنامه الگوریتم ممتیک بل را پیاده‌سازی می‌کند تا مقدار f را بهینه کند و در خروجی x و costdata را بازمی‌گرداند که به ترتیب جواب بهینه و اطلاعات فرایند بهینه‌سازی هستند.

تغییر alpha

با جایگزین کردن alpha با alpha3 در الگوریتم بهینه سازی، چند تغییر ممکن است رخ دهد:

- مقدار گام بهینه (step) ممکن است تغییر کند. alpha3 و alpha روش‌های متفاوتی برای محاسبه گام بهینه هستند، بنابراین جایگزین کردن آن‌ها منجر به مقادیر متفاوتی از گام بهینه می‌شود.

- سرعت همگرایی ممکن است تغییر کند. بسته به اینکه کدام روش بهتر گام بهینه را محاسبه کند، جایگزین کردن آن‌ها می‌تواند منجر به سرعت همگرایی بهتر یا بدتر شود.

- تعداد تکرارهای لازم برای همگرایی ممکن است تغییر کند. همانطور که گفته شد، تغییر سرعت همگرایی می‌تواند منجر به کاهش یا افزایش تعداد تکرارها شود.

- مقدار تابع هزینه در نقطه بهینه ممکن است تغییر نکند. هر دو روش باید به یک مینیمم محلی همگرا شوند، پس مقدار تابع هزینه در نقطه بهینه احتمالا ثابت می‌ماند.

- دقت جواب بهینه ممکن است تغییر نکند. هر دو روش باید به یک جواب بهینه محلی نزدیک شوند، پس دقت جواب نهایی احتمالا تغییر نمی‌کند.

در مجموع، جایگزین کردن alpha با alpha3 احتمالا منجر به تغییراتی در سرعت همگرایی و تعداد تکرارها می‌شود، اما مقدار نهایی تابع هزینه و دقت جواب به احتمال زیاد تغییر نمی‌کند.