نام و نام خانوادگی: فاطمه حدادی

نام استاد: باقر باباعلی

شماره دانشجویی: ۶۱۰۳۹۶۰۸۷

آذر ۹۹

گزارش تمرین حل TSP با تبرید شبیه سازی شده

- ## چکیده

در این تمرین سعی شده است که با الهام از الگوریتم زیستی تکامل مسئله TSP یا فروشنده دوره گرد را حل کنیم. پیاده سازی الگوریتم به زبان پایتون و با داده ی Data1 انجام شده است. همچنین برای بررسی بهتر بهترین جواب در هر نسل نیز در خروجی برنامه چاپ میشود. در این گرازش به تاثیرگذارترین پارامترها برای اجرای بهتر الگوریتم اشاره میشود.

- ## فهرست مطالب

۱. معرفی مسئله و داده ورودی

مسئله TSP یکی از مسائل معروف دسته NP-hard است. این مسئله به صورت داستان یک فروشنده دوره‌گرد مطرح میشود که قصد دارد یک تعداد شهر را پیماید و به نقطه شروع بازگردد بدون اینکه هر شهر را بیش از یکبار ببیند و کمترین هزینه یا به تعبیری مسافت را داشته باشد. داده ورودی انتخاب شده bayg29 است که به صورت ماتریس هزینه اطلاعات سفر از هر شهر به شهر دیگر ذخیره شده است.

۲. شرح بخش های الگوریتم

همانند مراحل الگوریتم در تئوری، ابتدا داده ورودی در ماتریس فاصله ها که همان هزینه است ذخیره میشود تا برای محاسبه تابع انرژی استفاده میشود. انرژی هر مسیر همان هزینه پیمایش آن مسیر است. هر مسیر یا تور یک جایگشت از شهر ها است. این مسیر و انرژی آن حالت ما را تشکیل میدهد که با هر تکرار وارد حالت جدید میشود تا بعد از پیمودن تعداد کافی مرحله به بهترین حالت برسد. در این الگوریتم میخواهیم به حالتی با انرژی مینیمم برسیم. پس با الهام از فرایند ترمودینامیک ابتدا دما را بالا برده و اجازه رفتن به حالت هایی با انرژی بالاتر میدهیم تا به خوبی فضا جستجو شود. سپس به تدریج دما را کاهش میدهیم تا به تعادل ترمودینامیکی که همان رسیدن به انرژی مینیمم است برسیم.

تابع اصلی، run_SA است که در آن ابتدا مسیر اولیه به صورت تصادفی ساخته میشود. سپس الگوریتم وارد حلقه میشود. در این حلقه ی تلکرار هر بار دما کمی کاسته میشود، T= 1-(curr/total) و سپس یک همسایه رندوم درتابع random_neighbor انتخاب میشود. این تابع از روش 2-opt استفاده میکند که بازدهی بالایی دارد.

البته در ابتدا از روش جابه جایی یا swap دو خانه برای پیدا پیدا کردن همسایه استفاده کردم اما جواب های به دست آمده بسیار ضعیف بودند و فضای جستجو به خوبی پیمایش نمیشد. با پیدا کردن روش 2-opt مسئله به خوبی حل شد و هزینه تور که همان انرژی یک حالت است، بسیار کاهش پیدا کرد.

این روش 2-opt دو نقطه به صورت رندوم در جایگشتی که مسیر ما را میشازد انتخاب میکند و قسمت بین دو نقطه را به صورت برعکس در مسیر قرار میدهد. این روش طبق آزمایش های انجام شده از روش جابه جایی دونقطه ی رندوم بازدهی بسیار بالاتری داشت.

اگر انرژی آن همسایه از حالت کنونی کمتر باشد به ان حالت میرویم. اگر بیشتر باشد در این صورت به احتمالی میتوانیم باز به آن خانه برویم که از عبارت $e^{-\Delta c/T}$ به دست می آید. دلتا تفاوت دو انرژی و T دما در آن حلقه است.

همچنین اگر به عنوان حافظه ی الگوریتم یک بهترین حالت را نیز نگه داری کنیم که بهترین حالت مشاهده شده از تمام حلقه ها باشد در این صورت جواب الگوریتم همین بهترین حالت گزارش میشود. بنابراین به تعداد sample_nums این حلقه ی تکرار انجام شده و پس از توقف مسیر متناظر با آن حالت و انرژی آن که همان هزینه تور است در خروجی چاپ میشود. البته برای مقایسه سرعت الگوریتم در اجرا با ورودی های مختلف، زمان اجرا نیز به ثانیه چاپ میشود.

۳. نمونه خروجی

به ازای پارامترهای زیر خروجی و جواب مسئله برای داده bayg29 بدین صورت است.

۱. پارامتر الگوریتم:

num_samples = 10000

خروجی الگوریتم:

23->27->16->13->24->8->1->28->6->12->9->5->26->3->29->21->2->20->10->4->18->14>17->22->11->15->19->25->7->23
cost: 1638 | fitness: 61.05
duration: 1.719 s

۲. پارامتر الگوریتم:

num_samples = 100000

خروجی الگوریتم:

14->17->22->11->19->13->16->25->7->23->27->24->8->28->1->21->6->12->9->5->26>3->29->2->20->10->4->15->18->14
cost: 1668 | fitness: 59.952
duration: 13.406 s

۳. پارامتر الگوریتم:

num_samples = 1000

خروجی الگوریتم:

28->12->9->5->6->21->26->3->29->2->20->10->19->25->7->11->14->22->17->18->15>4->16->13->24->27->23->8->1->28
cost: 1843 | fitness: 54.259
duration: 0.156 s

۴. پارامتر الگوریتم:

num_samples = 1000000

خروجی الگوریتم:

5->26->3->29->21->2->20->10->13->19->4->15->18->14->17->22->11->25->7->23->8>27->16->24->1->28->6->12->9->5
cost: 1654 | fitness: 60.459
duration: 134.953 s

- **تحلیل**

همانطورکه در بالا مشاهده شد این الگوریتم های پارامتر کمی دارد که این خود خوبی ها و بدی هایی دارد. یکی از خوبی های آن سادگی در پیاده سازی است که بدون داشتن مقادیر پارامتریک خاص میتوان جواب ها را بررسی کرد. از طرف دیگر به دلیل کم بودن پارامترها نمیتوان از طریق تعییر پارامتر ها آن را خیلی تغییر داد و باید جزئیات الگوریتم تغییر کند. مانند نحوه کاستن زمان و پیدا کردن همسایه رندوم.

اما طبق اجرا میتوان نتیجه گرفت تعداد تکرار ها در حلقه اهمیت بالایی دارد. البته از یه تعدادی بیشتری تاثیر خود را از دست میدهد و فقط زمان اجرا را زیاد میکند. پس تنها توصیه میشود از یه حدی بیشتر باشد تا بتوان به جواب مهم رسید.

البته مانند تحلیل های قبلی لازم به ذکر است این الگوریتم با مقدار پارامتر یکسان در هر اجرا مقدار متفاوت را در خروجی چاپ میکند پس این نتیجه گیری به صورت نسبی و احتمالاتی است.

در کل این الگوریتم توانسته در عین سادگی و مختصر بودن پارامترها به جواب بهینه قابل توجهی در زمان بسیار کمی نسبت به الگوریتم ژنتیک، مانند یک ثانیه در مقابل بیست ثانیه، برسد که تا حدی مرهون نحوه پیاده سازی و استفاده از روش 2-opt بوده است.

سپاس گزاری

در آخر از استاد گرامی دکتر باقر باباعلی تشکر میکنم.