چالش برنامه‌نویسی اِی آی سافت ۲۰۲۴:
مسئله‌ی مسیریابی موجودی (IRP)
-توصیف مساله و قوانین-

دانشگاه شیراز

۱۱ مهرماه ۱۴۰۳

۱ توصیف مسئله

با الهام از چالش برنامه‌نویسی DIMACS که از سال ۲۰۲۰ تا ۲۰۲۲ برگزار گردید، اولین مسابقه‌ی برنامه‌نویسی اِی آی سافت به مسئله‌ی مسیریابی موجودی (IRP) می‌پردازد. مساله مسیریابی موجودی در واقع ترکیبی از دو مسئله‌ی مسیریابی و کنترل موجودی انبار بوده که در حوزه مدیریت زنجیره تامین کاربرد فراوان دارد. به همین دلیل، در سال‌های اخیر این مساله مورد توجه گروه‌های تحقیقاتی مختلف در سراسر جهان قرار گرفته است (آرکتی و همکاران ۲۰۰۷، ۲۰۱۷؛ وادست و همکاران ۲۰۲۱؛ آرکتی و همکاران ۲۰۲۱؛ اسکالانز و همکاران ۲۰۲۳؛ لاگانا و همکاران ۲۰۲۴).

مسئله‌ی مسیریابی موجودی بر روی گراف کامل $G = (V, E)$ تعریف می‌شود که در آن $V = C \cup \{0\}$ مجموعه نودها و $E = \{(i,j) | i,j \in V\}$ مجموعه یال‌ها هستند. نود صفر، که به آن مبدا (انبار) هم گفته می‌شود، نشانگر یک تامین‌کننده مرکزی است که در مکان آن m حامل (خودرو) آماده به خدمت قرار دارند. خودروها از لحاظ پیکربندی همگن بوده و ظرفیت هر یک از آنها با Q نشان داده می‌شود. این حامل‌ها وظیفه ارائه خدمات به مشتریانی را برعهده داشته که با مجموعه $C = \{1,2, \dots, n\}$ مشخص شده‌اند. این سرویس‌دهی در مدت یک افق زمانی که شامل H پریود است انجام می‌شود (به عبارت دیگر داریم $T = \{1,2, \dots, H\}$). هر نود نظیر نود $i \in V$ (یعنی چه نود مبدا و چه مشتریان) از یک انبار برخوردار بوده که حداقل ظرفیت آن با L_i و حداکثر ظرفیت آن با U_i نمایش داده می‌شود. I_{i0} نشانگر موجودی آغازین انبار هر نود $i \in V$ است. نرخ تولید محصول توسط تامین‌کننده در هر پریود نظیر $t \in T$ با r_{0t} مشخص می‌گردد. به علاوه، r_{it} نشانگر میزان نیاز (نرخ مصرف) مشتری $i \in C$ در پریود $t \in T$ است. در هر پریود $t \in T$، مشتری $i \in C$ باید بتواند میزان نیاز خود را برطرف نماید بدون آنکه با کسری موجودی مواجه شود. برای این منظور، تامین‌کننده باید از قبل با استفاده از حامل‌های موجود و با در نظر گرفتن میزان موجودی انبار مشتریان و انبار خود محصولاتش را به دست مشتریان برساند. همه حامل‌ها باید مسیر خود را از نود مبدا شروع کرده و در پایان به این نود بازگردند. همچنین در هر پریود هر مشتری را حداکثر یک بار می‌توان ملاقات کرد. مطابق با سیاستی که آن را سیاست حداکثر موجودی می‌نامیم، یک حامل در هر بار ملاقات یک مشتری می‌تواند هر میزان محصول در اختیار او قرار دهد البته با این شرط که این میزان منجر به بیشتر شدن میزان موجودی انبار مشتری نسبت به ماکزیمم ظرفیت آن نشود.

با فرض اینکه c_{ij} هزینه‌ی سفر از نود $i \in V$ به نود $j \in V$ باشد، هدف ساخت یک برنامه توزیع برای افق زمانی مد نظر است به گونه‌ای که به تمامی مشتریان با کمترین هزینه سرویس داده شود. این هزینه شامل هزینه سفر و هزینه انبارداری، چه برای مشتریان و چه برای تامین‌کننده، است. به طور خلاصه، به منظور حل مساله مسیریابی موجودی باید به سه سوال اساسی پاسخ داد:

- هر مشتری در چه زمانی بازدید شده و سرویس دریافت نماید؟
- در هر بازدید از یک مشتری چه میزان محصول به او تحویل داده شود؟
- هر حامل باید چه مسیرهایی را برای سرویس‌دهی به مشتریان طی کند؟

۱-۱ مفاهیم ابتدایی

مبدا: یک تامین‌کننده مرکزی که با استفاده از تعداد محدودی حامل مسئولیت سرویس‌دهی به مشتریان را بر عهده دارد.

مشتری: یک خرده‌فروش که در هر پریود نرخ مصرف مشخصی داشته و دارای انباری با حداقل و حداکثر ظرفیت قابل قبولِ از پیش تعریف شده است.

نود: تامین‌کننده مرکزی (نود مبدا) و مشتریان

افق زمانی: یک مدت زمانی از پیش تعیین شده که به H پریود مجزا تقسیم شده است.

سطح انبار: میزان محصولات موجود در انبار

نرخ مصرف: نیاز هر مشتری در هر پریود

نرخ تولید: میزان محصولاتی که تامین‌کننده در هر پریود تولید می‌نماید.

۱-۲ محدودیت‌ها

محدودیت‌هایی که باید در ساخت برنامه‌ی توزیع لحاظ شوند عبارتند از:

- حداقل و حداکثر ظرفیت انبار هیچ نودی نباید نقض شوند.
- وقوع کسری موجودی در انبارها مجاز نیست.
- هر حامل می‌تواند حداکثر یک مسیر را در هر پریود طی کند.
- هر مشتری می‌تواند در هر پریود حداکثر یکبار ملاقات شود.
- هر مسیر باید از نود مبدا شروع شود و در این نود به پایان برسد.
- ظرفیت هر حامل محدود بوده و نباید نقض شود.

۱-۳ تابع هدف

هدف در مساله مسیریابی موجودی، مینیمم کردن هزینه، یعنی مجموع هزینه حمل و نقل و هزینه نگه‌داری محصولات در انبارهای مشتریان و تامین‌کننده است. هزینه حمل و نقل همان هزینه سفر از هر نود به نود دیگر بوده که از قبل و با در نظر گرفتن اطلاعات ترافیک، هزینه سوخت و دیگر اطلاعات مرتبط مشخص می‌گردد. به علاوه، هزینه انبارداری به ازای هر واحد محصول برای تامین‌کننده و مشتریان نیز از پیش تعیین شده است. میزان موجودی انبار هر مشتری $i \in C$ در پریود $t \in T$، برابر با مجموع موجودی این انبار در پریود $t-1$ و میزان سرویس دریافت شده در پریود t بوده که از آن میزان سرویس مصرف شده در پریود t کسر شده باشد. به عبارت دیگر، داریم $I_{it} = I_{i(t-1)} + q_{it} - r_{it}$. توجه داشته باشید که هزینه‌های انبارداری برای موجودی آغازین انبارها نیز در نظر گرفته می‌شوند.

۲ فایل‌های نمونه

هر نمونه مساله در یک فایل جداگانه و با ساختار مشابه ارائه شده است. این ساختار در مثال زیر توصیف می‌گردد.

```
nVehicles:  10
nDepots:     1
nCustomers: 10
nPeriods:    2
VEHICLE_TYPE    FLEET_SIZE      CAPACITY
           1            10             8
DEPOT           X       Y       INV_COST        INITIAL_INV     MIN_LEVEL_INV   MAX_LEVEL_INV   PRODUCTION_0    PRODUCTION_1
       0        428     298             0               99              0             622             70              70
CUSTOMER        X       Y       INV_COST        INITIAL_INV     MIN_LEVEL_INV   MAX_LEVEL_INV   DEMAND_0        DEMAND_1
       1        211     283          0.22               7              0              31          5               8
       2        232     365          0.22               1              0              54          3               6
       3        223     318          0.05               8              0              14          3               5
       4        234     325          0.16               6              0              69          6               9
       5        254     325          0.06               3              0              23          4               4
       6        295     428          0.02               3              0              47          0               0
       7        181     490          0.19               9              0              32          6               8
       8        110     177          0.08              11              0              56          6              10
       9         71     236          0.21               5              0              21          4               5
      10        459     143          0.21              10              0              27          4               6
COST_MATRIX
          0      1      2      3      4      5      6      7      8      9     10
  0
  1    109
  2    104     43
  3    103     19     24
  4     98     24     21      7
  5     89     31     23     16     11
  6     93     84     45     66     60     56
  7    157    105     68     89     87     91     65
  8    171     74    113     91     97    104    156    161
  9    182     74    104     87     93    102    148    139     36
 10     80    143    159    147    145    138    165    223    176    200
```

شکل ۱مثالی از یک فایل داده

چهار سطر اول نشان‌دهنده تعداد حامل‌ها، مبدا، مشتریان و پریودها است. در دو سطر بعد نوع هر حامل، تعداد و ظرفیت آن‌ها نمایش داده شده‌اند. لازم به ذکر است که در همه‌ی نمونه‌ها حامل‌ها همگن در نظر گرفته شده‌اند. بنابراین، برای هر نمونه، حداکثر ظرفیت

حامل‌ها یکسان است. همچنین همانطور که پیش از این نیز ذکر گردید، در این چالش مساله مسیریابی موجودی با یک تامین‌کننده در نظر گرفته شده است. بنابراین، در تمام نمونه‌ها مقدار nDepots برابر با ۱ است. هشتمین سطر در هر فایل نمونه ویژگی‌های نود مبدا را ارائه می‌دهد. توصیف اطلاعات نمایش داده شده در این سطر به شرح زیر است:

DEPOT = شماره نود تامین‌کننده (مبدا)

X= مختصات x مکان تامین‌کننده

Y = مختصات y مکان تامین‌کننده

INV_COST = هزینه نگه‌داری از هر واحد محصول در انبار برای تامین‌کننده

INITIAL_INV = موجودی آغازین انبار برای تامین‌کننده

MIN_LEVEL_INV = حداقل ظرفیت انبار تامین‌کننده

MAX_LEVEL_INV = حداکثر ظرفیت انبار تامین‌کننده

PRODUCTION_i = مقدار کالایی که در هر پریود از افق زمانی توسط تامین‌کننده تولید می‌شود

در n سطر بعدی ویژگی‌های هر یک از n مشتری در نظر گرفته در آن نمونه توصیف می‌شود. در این بخش برای هر مشتری $i \in C$ اطلاعات زیر ارائه شده است:

CUSTOMER = شماره مشتری

X = مختصات x مکان مشتری i

Y = مختصات y مکان مشتری i

INV_COST = هزینه نگه‌داری هر واحد محصول در انبار برای مشتری i

INITIAL_INV = موجودی آغازین انبار مشتری i

MIN_LEVEL_INV = حداقل ظرفیت انبار مشتری i

MAX_LEVEL_INV = حداکثر ظرفیت انبار مشتری i

DEMAND_i = مقدار کالایی که مشتری i در هر پریود از افق زمانی نیاز دارد

در پایان هر فایل نمونه، هزینه‌های سفر (یعنی c_{ij} برای هر دو مشتری i و j) به صورت یک ماتریس قطری ارائه شده است.

۳ فایل‌های جواب

برای هر نمونه باید یک فایل جواب ارائه شود که حاوی اطلاعات بهترین جواب شدنی به دست آمده برای آن نمونه است. نام هر فایل جواب باید Solu-InsName.txt باشد که در آن InsName نام فایل نمونه مرتبط بدون پسوند است. برای مثال، فایل خروجی مربوط به نمونه Sabs1n20H3.dat به صورت Solu-Sabs1n20H3.txt است. به علاوه، محتوای این فایل باید از فرمتی که در ادامه توصیف خواهد شد پیروی نماید، در غیر اینصورت الگوریتم شما در رتبه‌بندی برای آن نمونه در مکان آخر قرار خواهد گرفت.

در ادامه با در نظر گرفتن ۱۰ مشتری، ۳ حامل و دو پریود مثالی از یک جواب برای مسئله مسیریابی موجودی ارائه شده است. مسیرهای حامل و کل هزینه توزیع برای هر پریود در یک خط نشان داده شده‌اند. این مسیرها با استفاده از براکت از یکدیگر جدا شده‌اند. هر براکت شامل تعدادی زوج-مقدار به صورت (CNum, Del) است، که در آن CNum نشانگر شماره مشتری و Del نشانگر مقدار سرویس ارائه شده به آن مشتری است. سه خط بعدی هزینه کل سفر و هزینه‌ی کل انبارداری برای مشتریان و تامین‌کننده را به صورت جداگانه مشخص می‌نمایند. در خط آخر هزینه کل جواب ارائه شده است.

```
Period 1: [(9,2)(8,3)(7,3)] [(6,1)(4,3)(5,1)(2,3)] cost: 406
Period 2: [(5,7)(9,1)] [(10,2)(7,3)(1,1)(3,1)] [(2,1)] cost: 1041
Total transportation cost: 1000
Total inventory cost at customer locations: 21
Total inventory cost at the depot: 20
Total cost: 1447
```

لطفا توجه داشته باشید که هر مسیر باید از نود مبدا شروع شود و به آن ختم گردد. اما به منظور سادگی نود صفر در فایل جواب به صورت مستقیم عنوان نشده است. بین هر دو براکت و بین آخرین براکت در هر خط و کلمه «cost» یک فاصله قرار دارد. به علاوه، در فرمت ذکر شده همواره یک فاصله بین علامت «:» و مقدار ذکر شده بعد از آن وجود دارد. برای مثال، در خط یک داریم Period space 1: space

space cost: space 406 [(6,1)(4,3)(5,1)(2,3)] space [(9,2)(8,3)(7,3)]، یا در خط ششم داریم Total cost: space 1447 . فایل‌های خروجی به صورت خودکار خوانده شده و پردازش می‌شوند. بنابراین هر فایلی که مطابق فرمت ذکر شده نباشد در نظر گرفته نخواهد شد.

۴ قوانین اصلی مسابقه

قانون ۱: شرکت‌کنندگان باید وابستگی دانشگاهی داشته باشند و مجازند در گروه‌های حداکثر سه نفره کار کنند.

قانون ۲: شرکت‌کنندگان باید رویکرد حل مسئله خود را به صورت تک نخی پیاده‌سازی کنند. آن‌ها می‌توانند از هر زبان برنامه نویسی که تحت ویندوز یا لینوکس اجرا می‌شود استفاده نمایند. همچنین، استفاده از هر کتابخانه منبع باز، مادامی که به صورت تک نخی اجرا شود، مجاز است.

قانون ۳: از روز آغاز مسابقه، یک مجموعه شامل ۱۰ نمونه با اندازه‌های مختلف در اختیار شرکت‌کنندگان قرار می‌گیرد. در پایان این ماه، ۲۰ نمونه دیگر در اختیار شما قرار داده خواهد شد که از این نمونه‌ها برای تعیین فینالیست‌ها استفاده می‌گردد. از یک مجموعه نمونه دیگر برای رتبه‌بندی نهایی فینالیست‌ها استفاده خواهد شد. این سه مجموعه نمونه به ترتیب Public1، Public2 و Hidden نامیده می‌شوند.

قانون ۴: الگوریتم شما باید یک فایل نمونه با فرمت مشخص شده را به عنوان ورودی دریافت کرده و یک فایل خروجی با فرمت ذکر شده در بالا تولید نماید. برای حل تمامی نمونه‌ها باید ورژن یکسانی از الگوریتم استفاده شود.

قانون ۵: الگوریتم می‌تواند به صورت قطعی یا تصادفی طراحی گردد، اما نتایج باید بتوانند مجددا تولید شوند. به طور خاص، شرکت‌کنندگانی که از یک الگوریتم تصادفی برای حل مسئله استفاده می‌کنند باید به گونه‌ای برنامه خود را بنویسند که جواب‌های ارسالی مجددا قابل تولید باشند (به طور مثال، با استفاده از یک سید رندم ثابت).

قانون ۶: توصیه می‌شود که الگوریتم خود را به صورت any time طراحی نمایید به گونه‌ای که هر زمان که الگوریتم یک جواب بهتر پیدا می‌کند، فایل خروجی قبلی را با فایل خروجی جدیدی جایگزین نماید. به این ترتیب، می‌توانید اطمینان حاصل کنید که با اتمام الگوریتم در هر زمان دلخواه یک فایل خروجی دارید.

قانون ۷: بعد از پایان مهلت مسابقه، مجموعه‌ای از حداکثر ۱۰ فینالیست انتخاب می‌شوند (که با توجه به نتایج بدست آمده برای نمونه‌های عمومی دوم رتبه‌بندی می‌شوند). رتبه‌بندی نهایی فینالیست‌ها بر اساس امتیازهای به دست آمده بر روی نمونه‌های Hidden صورت می‌گیرد. برای این منظور، رتبه‌بندی بر اساس متوسط رتبه‌ی هر گروه شرکت‌کننده برای هر نمونه انجام می‌شود.

۵ برنامه زمانی مسابقه

تاریخ‌های مهم مربوط به این مسابقه به شرح زیر هستند:

- **۲۸ سپتامبر ۲۰۲۴- انتشار توصیف مساله و قوانین ابتدایی مسابقه:** در این تاریخ، مسابقه‌ی مسئله مسیریابی موجودی شروع شده و همه شرکت‌کنندگان می‌توانند با استفاده از اولین بخش نمونه‌های عمومی بر روی الگوریتم پیشنهادی خود کار کنند.
- **۱۹ اکتبر ۲۰۲۴- انتشار ورژن نهایی قوانین مسابقه و جوایز:** پس از این تاریخ، ممکن است تنها تغییرات اندکی در قوانین مسابقه ایجاد گردد. همچنین، در این تاریخ دومین بخش از نمونه‌های عمومی در دسترس قرار خواهد گرفت. از این نمونه‌ها برای تعیین فینالیست‌ها استفاده می‌شود.
- **۷ دسامبر ۲۰۲۴- ددلاین نهایی شرکت‌کنندگان برای ارسال کلیه فایل‌های خروجی**
- **۱۰ دسامبر ۲۰۲۴- نتایج مسابقه منتشر شده و گروه‌های فینالیست‌ها اعلام می‌گردند**
- **۱۴ دسامبر ۲۰۲۴- اعلام رتبه‌بندی نهایی:** دعوت نامه‌ها برای شرکت در ورکشاپ ارسال می‌شوند.
- **۲۶-۲۴ دسامبر ۲۰۲۴- برگزاری اولین ورکشاپ مسابقه‌ی برنامه‌نویسی اِی آی سافت ۲۰۲۴**

۶ مراجع

Archetti C, Bertazzi L, Laporte G, Speranza MG. A branch-and-cut algorithm for a vendormanaged inventory-routing problem. Transportation science 2007;41(3):382–391.

Archetti C, Boland N, Grazia Speranza M. A matheuristic for the multivehicle inventory routing problem. INFORMS Journal on Computing 2017;29(3):377–387.

Archetti C, Guastaroba G, Huerta-Mu~noz DL, Speranza MG. A kernel search heuristic for the multivehicle inventory routing problem. International Transactions in Operational Research 2021;28(6):2984–3013.

Lagan_a D, Malaguti E, Monaci M, Musmanno R, Paronuzzi P. An iterated local search matheuristic approach for the multi-vehicle inventory routing problem. Computers & Operations Research 2024; p. 106717.

Sk_alnes J, Vadseth ST, Andersson H, St_alhane M. A branch-and-cut embedded matheuristic for the inventory routing problem. Computers & Operations Research 2023; 159:106353.

Vadseth ST, Andersson H, St_alhane M. An iterative matheuristic for the inventory routing problem. Computers & Operations Research 2021; 131:105262.