دانشکده علوم ریاضی

سمینار درس بهینه سازي ترکیبیاتي

مسأله‌ی پوشش مجموعه

استاد گرامی

آقای دکتر رضا قنبري

نگارش

زهرا قربانی

دي ماه ۱۳۹۰

فهرست مطالب

چکیده

مساله‌ی پوشش مجموعه یکی ازمسایل وسیع مطالعه شده دربهینه‌سازی ترکیبیاتی وعلوم کامپیوتر وتئوری پیچیدگی می‌باشد که کاربردهای فراوانی ازجمله در مکان‌یابی تسهیلات و زمان‌بندی کارکنان دارد. این مساله در دسته مسائلNp-hard قرار می‌گیرد یعنی هیچ الگوریتم دقیق چند جمله‌ای برای حل آن وجود ندارد. الگوریتم‌های فراوانی از جمله الگوریتم‌های دقیق برپایه‌ی روش‌های شاخه وکران والگوریتم‌های تقریبی برای حل این مساله ارائه شده‌اند. چون روش‌های دقیق نیازمند محاسبات فراوان برای مسائل دراندازه‌های بزرگ هستند، الگوریتم‌های فراابتکاری برای یافتن یک جواب نزدیک بهینه‌ی خوب در زمان معقول برای مساله به‌کار می‌روند. دراین جمع آوری در فصل اول به معرفی مساله ، کاربردها و زیرمساله‌های آن می‌پردازیم . در فصل دوم مروری بر روشهای حل داشته و چند نمونه ازالگوریتم‌های موجود را بررسی و مقایسه می‌کنیم.

فصل ۱

معرفی مساله، کاربردها و زیر مساله‌ها

۱.۱ مقدمه

مساله‌ی پوشش مجموعه یکی ازاولین مسایلی است که در۲۱ مساله‌ی کارپ[۱]، Np-compelet بودن‌آن ثابت شد. درابتدا برای این مساله الگوریتم‌های تقریبی موردبررسی قرارگرفت. *جانسون*[۲] ۱۹۷۴ و *لاواز*[۳] ۱۹۷۴ یک الگوریتم حریصانه که $H(d)$ -تقریب بود برای این مساله پیدا کردند. $(H(d) = \sum_{i=۱}^{d} \frac{۱}{i})$ سپس این الگوریتم حریصانه برای حالت وزن دار توسعه داده

---

[۱] Karp's ۲۱ Np-compelet problems [۲] Johnson [۳] Lavás

شد[۱].درسال ۱۹۷۹ توسط *گری وجانسون* [۲] ثابت شدمساله‌ی پوشش مجموعهNp-hard درحالت strong است.

نمونه‌ی (X, F) ازمساله‌ی پوشش مجموعه شامل یک مجموعه‌ی متناهی X وخانواده Fاززیرمجموعه‌هایX می‌باشدبطوریکه هرعنصرازX متعلق به حداقل یک زیرمجموعه درF باشد.[۱]

$$X = \cup_{S \in F} S$$

می‌گوییم زیرمجموعه‌ی$S \in F$ عناصرX رامی‌پوشاند. مساله یافتن زیرمجموعه‌ای ازF مانندC بااندازه‌ی می‌نیمم می‌باشد که اعضای آن تمام X را می‌پوشاند.

$$X = \cup_{S \in C} S$$

هرC که دررابطه‌ی بالاصدق کندX را می‌پوشاند.

درادامه به آشنایی بیشتر با مساله وروش‌های حل [۸]آن می‌پردازیم .

[۱]Chvatal، ۱۹۷۹ [۲]Garey and Johnson

۲.۱ مساله‌ی پوشش مجموعه

مدل برنامه‌ریزی خطی عدد صحیح مساله

مدل برنامه‌ریزی خطی عدد صحیح مساله‌ی پوشش مجموعه به صورت زیر توصیف میشود: ماتریس a_{ij} را $m \times n$ با درایه‌های صفرویک در نظر بگیرید. هر ستون j هزینه‌ای برابره $c_j >$ دارد. مساله عبارتست از پوشاندن سطرهای ماتریس توسط زیر مجموعه‌ای از ستونها بطوریکه کل هزینه می‌نیمم شود. متغیر x_j را تعریف می‌کنیم:

$$x_j = \begin{cases} ۱ & \text{اگر ستون } j \text{ درجواب باشد} \\ ۰ & \text{درغیر اینصورت} \end{cases}$$

همچنین متغیر a_{ij}

$$a_{ij} = \begin{cases} ۱ & \text{اگرستون } j \text{ سطر } i \text{ را بپوشاند} \\ ۰ & \text{درغیراینصورت} \end{cases}$$

مدل به شکل زیر است:

$$\min \quad \sum_{j=۱}^{n} c_j x_j$$

$$\text{subject to:} \quad \sum_{j=۱}^{n} a_{ij} x_j \geq ۱ \qquad \text{i=1,...,m}$$

$$x_j \in \{۰, ۱\} \qquad j = ۱, ۲, ..., n$$

مساله‌ی پوشش مجموعه در حالت وزن دار

درحالت وزن دارمساله به مساله‌ی پوشش مجموعه باکمترین وزن[۱] تبدیل میشود.دراین

[۱] minimum weighted set covering problem

حالت مجموعه‌ی پایه‌ی $U = \{u_۱, u_۲, ..., u_n\}$ و زیر مجموعه‌های آن $S_۱, S_۲, ..., S_k$ را در نظر بگیرید. برای هر S_i نیز وزنهای مثبت c_j مفروض است. هدف یافتن مجموعه‌ی $I \subseteq \{۱, ۲, ..., m\}$ می‌باشد بطوریکه $\cup_{i \in I} S_i = U$ می‌نیمم شود و $\sum_{i \in I} c_i$. در حالت بدون وزن برای تمام c_j ها داریم۱ $c_j = ۱$:.

۳.۱ کاربردها

مساله‌ی پوشش مجموعه کاربردهای گوناگونی درزمینه های مختلف دارد و بسیاری ازمسایل به شکل آن فرمول‌بندی میشوند.دراینجا به چندنمونه ازاین کاربردها اشاره می‌کنیم.

۱.زمان‌بندی کارکنان[۱]۱[۱۱]

تقاضاهای حمل ونقلی که توسط کمپانی‌های حمل ونقل برآورده میشوند به وسیله‌ی یک جدول زمانی نشان داده میشود.کارکنان و وسایل نقلیه،ایستگاههای آغازوپایان ومکان‌های استراحت کارکنان معین است. هدف مساله این است که هریک ازسفرهای درون جدول حداقل یک بارتوسط زیرمجموعه ای بهینه ازسفرها انتخاب شود به طوریکه کل ساعات کاری کارکنان می‌نیمم شود.این مساله به شکل یک مساله پوشش فرمول میشود.

۱crew scheduling

۲. مساله‌ی بلیت‌های بخت آزمایی[۱] [۷]

درمساله‌ی بلیت‌های بخت آزماییn شماره از میان یک مجموعه‌ی m شماره‌ای به تصادف بیرون کشیده میشوند.روی بلیت‌ها،ما بایدn شماره را پرکنیم با این امیدکه این n شماره باn شماره‌ی بیرون کشیده شده یکسان باشد.حال به دنبال می‌نیمم تعداد بلیت‌هایی که بایدپرکنیم هستیم بطوریکه حداقل یک بلیت وجودداشته باشد که p شماره یا بیشتر ، تعداد شماره‌ی یکسان با n شماره‌ی بیرون کشیده شده داشته باشد. این مساله نیزبه صورت مساله پوشش فرمول‌بندی میشود. ازدیگر کاربردهای مساله پوشش مجموعه می‌توان به مکان‌یابی تسهیلات[۲][۳]،ممانعت ازترافیک[۳] و ردیابی ویروسهای کامپیوتری [۴]اشاره کرد.

۴.۱ زیرمساله ها

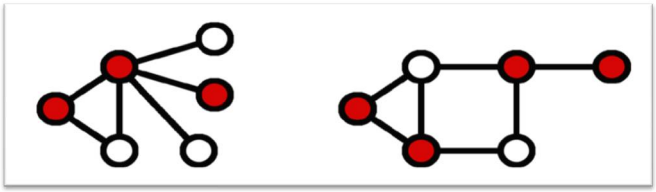
از زیر مساله‌های مساله‌ی پوشش مجموعه به دو مورد اشاره می‌کنیم:

۱. مساله‌ی پوشش راس[۵][۲] :گراف$G = (V, E)$ مفروض است. پوشش راسی این گراف عبارتست از زیر مجموعه‌یC از راسها بطوریکه هریال در گراف G مجاورحداقل یک راس درC باشد. درشکل زیرنمونه‌ای از این مساله را مشاهده می‌کنید:

---

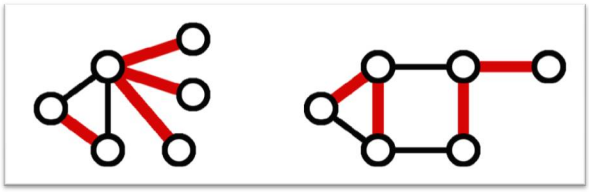
[۱] Lottery problem [۲] facility location [۳] Traffic checks [۴] computer viruses detection

[۵] vertex cover



۲. مساله‌ی پوشش یال[۱]:

گراف $G = (V, E)$ مفروض است. پوشش یال این گراف عبارتست از زیرمجموعه‌ی C از یالها بطوریکه هرراس درگراف G مجاور حداقل یک یال در C باشد.درشکل زیرنمونه‌ای را مشاهده می کنید:



[۱] edge cover

فصل ۲

روش‌های حل و چند نمونه‌از الگوریتم‌ها

۱.۲ روشهای حل

همان طورکه قبلا گفته شدمساله‌ی پوشش مجموعه یک Np-hardدرحالت strong می باشد وبسیاری ازالگوریتم‌هابرای حل آن ارائه شده‌اند.الگوریتم‌های دقیق[۱] اکثرابرپایه‌ی روشهای شاخه وکران[۲] وشاخه و برش[۳] برای این مساله توسعه داده شده‌اند.الگوریتم‌های دقیق

---

[۱] Fisher and Kedia، ۱۹۹۰؛ Beasley and Jornsten Balas؛ ۱۹۹۲ and Carrera ۱۹۹۶ [۲] Branch and bound [۳] Branch and cut

مختلفی برای پوشش مجموعه مقایسه شده‌اند.[۱] این مقایسه‌ها نشان می‌دهندبهترین الگوریتم دقیق برای مساله‌ی پوشش مجموعه$CPLEX$[۱] می‌باشد.چون روشهای دقیق نیازمند محاسبات فراوان برای مسائل دراندازه‌های بزرگ هستند،الگوریتم‌های ابتکاری معمولا برای یافتن یک جواب نزدیک بهینه‌ی خوب درزمان معقول به کارمی‌روند.شایدالگوریتم‌های حریصانه معمول‌ترین رویکرد ابتکاری برای حل سریع مسایل ترکیبیاتی بزرگ باشند که هرچندساده وسریع درکدکردن،کمترمی‌توانند جوابهایی باکیفیت خوب تولید کنند.برای مساله‌ی پوشش مجموعه ساده ترین رویکرد،الگوریتم حریصانه‌ای است که توسط$chvatal$ درسال۱۹۷۹ طراحی شد. برای بهبودبخشیدن به کیفیت جواب،الگوریتم‌های ابتکاری مدرن مانند سردکاری تدریجی[۲]،ژنتیک[۳]،مورچگان[۴] وشبکه عصبی[۵]بارفتارتصادفی معرفی شدند.این الگوریتم‌های ابتکاری معمولادرطبقه‌ی الگوریتم‌های فراابتکاری[۶] قرارمی‌گیرند. دو اشکال درموردالگوریتم‌های ابتکاری موجودبرای مساله‌ی پوشش مجموعه وجوددارد.اولا بیشترالگوریتم‌های ابتکاری برای حالت وزن دارطراحی شده‌اند.زیرااطلاعات درموردهزینه نقش مهمی دراین الگوریتم‌هابازی می‌کند.تعدادبسیارکمی ازالگوریتم‌های ابتکاری به روی هردونوع مسایل وزن دار و بدون

[۱] $CPLEX$ یک بسته‌ی نرم‌افزاری شامل مجموعه‌ای ازبرنامه‌های مدلسازی ریاضی است.

[۱] Capara et al.، ۲۰۰۰ [۲] simulated anealing؛Jacobs and Brusco، ۱۹۹۵ [۳] Genetic؛Beasley and chu، ۱۹۹۶؛Aickeli، ۲۰۰۲ [۴] Ant colony؛Zhigang Ren،Zuren Feng، ۲۰۰۸ [۵] Neureal Network؛Ohlsson et al، ۲۰۰۱ [۶] Meta-Heuristic

وزن کارمی‌کنند.اشکال دوم اینکه پیاده‌سازی بیشترالگوریتم‌های ابتکاری که جواب خوب تولید می‌کنند برای شاغلانی که پیش زمینه‌ی قوی درتحقیق درعملیات ندارند مشکل است،ازطرف دیگرالگوریتم‌های ابتکاری ساده مانندالگوریتم‌های ابتکاری حریصانه معمولا نتایج خیلی خوبی تولید نمی‌کنند.

۲.۲ الگوریتم جستجوی محلی برای حالت بدون وزن

دراین قسمت به بررسی الگوریتمی برپایه‌ی جستجوی محلی[۱] می‌پردازیم.[۹] برطبق این الگوریتم یک جواب اولیه بهبود داده میشود به این صورت که درهرتکراریک همسایگی حول جواب جاری تولیدمیشود و طبق معیاری مشخص جوابی دراین همسایگی برای تکراربعدانتخاب میشود.دراین الگوریتم یک ساختارهمسایگی ،یک تابع برازندگی،یک پارامتر کران بالا ویک لیست ممنوع معرفی شده است.درادامه به توصیف این مواردمی‌پردازیم.

الف) ساختار همسایگی:

همانطور که می‌دانیم جواب مساله‌ی پوشش مجموعه یک زیرمجموعه $C \subseteq F$ است که اعضای آن تمام X را می‌پوشانند.لازم به ذکراست یک جواب درابتدا و همچنین درفازبهبود می‌تواندغیرمجازباشد، یعنی جوابی که تمام عناصرX رانمی‌پوشاند.برای تولید همسایگی

[۱] Local search for unicost set covering problem

حول زیرمجموعه‌ی C دوحرکت اساسی به کارمی‌رود:

ADD-SET(S):یک مجموعه‌یS ازخانواده زیرمجموعه‌های F که درC نیست وبه آن اضافه می‌شود.$C : C \longleftarrow C \cup S$

REMOVE-SET(S):مجموعه‌یS که ازC حذف میشود.$C : C \longleftarrow C - S$

حرکت دیگری که دربرخی نوشته‌جات استفاده شده است تعویض یک مجموعه S_1 درC با مجموعه‌ی دیگرS_2 که درC نیست می‌باشد:

SWAP-SET(S):تعویض S_1, S_2

تمام همسایگی که با به کاربردن دو حرکت اول تولید میشودشامل$|F|$ جواب است. اگرحرکتswap به کاررود همسایگی تولیدشده بسیاربزرگ خواهدبود.دراین حالت تمام همسایگی شامل $|C| * (|F| - |C|)$ جواب است. برای اجتناب از تعدد جوابها در همسایگی ،روشی بکارمی‌بریم که فقط دوحرکتADDوREMOVE به کاررونـد. تجربه نشان داده این‌دو حرکت برای دستیابی به جوابهای راضی کننده درزمان معقول کافی هستند.

ب)محدود کردن همسایگی بوسیله‌ی معرفی یک پارامترکران بالا :

کران بالا به عنوان تعداد مجموعه‌های بهترین جواب مجاز تعریف میشود.

اگر فقط دوحرکتADDوREMOVE به کاربرده شوند این کران بالا می تواند برای محدود کردن همسایگی درطول جستجو به کاررود.بااین محدودسازی،حرکتADD می تواندرجواب جاری به کاررود فقط اگر تعداد مجموعه‌ها درجواب جاری کمتراز کران بالا منهای یک

باشد. $(upperbound - ۱)$ کران بالا ازاولین جواب مجازکه تمام X را می‌پوشاند محاسبه می‌شود.اگر جواب مجازدرطول جستجو بهبودپیدا کند کران بالا بطورخودکار به‌روز میشود. هدف ازمحدودکردن همسایگی قوت بخشیدن به جستجو درهمسایگی بهترین جواب جاری می‌باشد.

پ)تابع برازندگی : تابع برازندگی برای محاسبه‌ی کیفیت وچگونگی جواب بکار می‌رود. یک جریمه‌ی مقدار ۱ برای هر عنصر پوشیده نشده از مجموعه‌ی X وهرمجموعه درC درنظر گرفته میشود.برازندگی جواب برابر است با:

Fitness=Number of uncovered elements+|C|

هدف ، می‌نیمم کردن این تابع در طول جستجو می‌باشد.

ت)جواب اولیه: برای تولید جواب اولیه از یک الگوریتم حریصانه استفاده میشود. این الگوریتم بدین صورت عمل می کند که در هرمرحله مجموعه‌ای را انتخاب می‌کند که بیشترین عناصرپوشیده نشده را تاحد امکان بپوشاند.

ث)استفاده ازتاریخچه‌ی جستجو :

برای اجتناب از درون دور افتادن درطول جستجو این الگوریتم از تاریخچه‌ی جستجوی خود استفاده می کند.این اطلاعات دریک حافظه بنام لیست ممنوع[۱] ذخیره شده‌اند.این لیست را مجموعه‌هایی که در تکرارهای قبلی اضافه یاکم شده‌اند می‌سازند.برای مثال اگر

[۱] Tabu list

جوابی برای تکرار بعد قابل قبول بوده و با اضافه کردن مجموعه S_1 به C بدست آمده باشد، این S_1 وارد لیست ممنوع میشود.

ج) معیار انتخاب جواب: بهترین جواب از یک همسایگی، جواب جاری برای تکرار بعد است. جوابی که برای تکرار بعد انتخاب میشود نباید یک جواب ممنوع باشد. یک جواب ممنوع است اگر با بکار بردن یکی از حرکتها روی مجموعه ای از لیست ممنوع بدست آمده باشد.اگر بهترین جواب در یک همسایگی جوابی ممنوع باشد از معیار تنفس [۱] استفاده میشود.

چ)الگوریتم جستجوی محلی :

شمای کلی الگوریتم جستجوی محلی ذکرشده، بصورت زیر است:

(۱) تولید یک جواب اولیه.

(۲) مقداردهی پارامتر کران بالا و لیست ممنوع .

(۳)تولید همسایگی جواب جاری بااستفاده ازدوحرکت ADD-Set ,REMOVE-Set .

(۴) ارزیابی همسایگی جواب.

(۵) انتخاب یک جواب برای تکراربعد با درنظرگرفتن معیار انتخاب .

(۶) به روز کردن کران بالا ولیست ممنوع .

(۷) اگر شرایط توقف برقرارنبود به گام ۳ برمی‌گردیم وگرنه به گام ۸ می‌رویم.

(۸) بازگشت به بهترین جواب ممکن .

---

[۱] aspiration

۳.۲ الگوریتم تقریبی

یک راه برای حل مسائل Np-hard استفاده از الگوریتمهای تقریبی می‌باشد.یک الگوریتم برای یک مساله بهینه سازی α - تقریب نامیده میشود هرگاه اولا در زمان چندجمله‌ای قابل اجرا باشد ، ثانیا همواره جوابی تولید کند که فاصله آن تاجواب بهینه ضریبی ازα باشد.دراین بخش به معرفی یک الگوریتم تقریبی برای مساله‌ی پوشش به نام الگوریتم رند کردن [۱][۶]می‌پردازیم.

تعریف: نمونه $E = \{e_۱, e_۲, ...e_n\}$ وزیرمجموعه‌های $S_۱, S_۲, ...S_m$ ازآن مفروض هستند.هزینه‌ی هر S_j نیز $w_j \geq ۰$ است.هدف یافتن مجموعه $I = \{۱, ۲, ..., m\}$ است بطوریکه $\cup_{j \in I} S_j = E$ و $\sum_{j \in I} w_j$ می‌نیمم شود.الگوریتم تقریبی که ارائه می‌کنیم برپایه‌ی برنامه‌ریزی عدد صحیح برای مساله پوشش می‌باشد. متغیرX_j را بصورت زیر تعریف می‌کنیم:

$$X_j = \begin{cases} ۱ & \text{اگر } i \in I \text{ باشد} \\ ۰ & \text{در غیر اینصورت} \end{cases}$$

مدل برنامه‌ریزی عددصحیح مساله پوشش را بصورت۲.۱ درنظربگیرید. همچنین f را بصورت زیر تعریف می‌کنیم:

---

[۱]Rounding algorithm

اگر داشته باشیم $f_i = |\{j|e_i \in S_j\}|$ آنگاه $f = max f_i$ به روی تمام i ها.بدین معنی که f ماکزیمم تعداد مجموعه هایی است که عناصر e_i را دربر دارند. باتوجه به آزادسازی قیدهای صحیح بودن درمدل عددصحیح یاد شده، الگوریتم تقریبی زیر را داریم.

الگوریتم تقریبی رند کردن

ورودی: مجموعه متناهی $E = \{e_۱, e_۲, ...e_n\}$ و زیرمجموعه‌های $S_۱, S_۲, ...S_m$ از آن وهزینه‌های $w_j \geq ۰$ برای هر S_j .

خروجی: مجموعه I

گام (۱) مساله را بدون در نظرگرفتن قید های صحیح بودن حل کنید تا جواب بهینه x^* بدست آید.

گام(۲) قرار دهید $I \Leftarrow \oslash$.

گام(۳) برای هر S_j انجام بده:

اگر $x_j^* \geq \frac{۱}{f}$ قرار بده $I \Leftarrow I \cup \{j\}$.

لم ۱.۳.۲. $I = \{j|x_j^* \geq \frac{۱}{f}\}$ یک *set cover* است.[۱۰]

قضیه ۱.۳.۲. *الگوریتم ذکر شده یک الگوریتم* f *- تقریب است.*[۱۰]

۴.۲ الگوریتم‌های فرا ابتکاری

۱.۴.۲ الگوریتم ژنتیک

دراین بخش به بررسی نمونه‌ای ازالگوریتم های فراابتکاری به نام الگوریتم ژنتیک[۴] برای مساله‌ی پوشش می‌پردازیم.

گامهای اصلی یک الگوریتم ژنتیک ساده بصورت زیراست:

۱.تولیدیک جمعیت اولیه

۲.ارزیابی برازندگی اعضادرجمعیت

حلقه‌ی تکرار

۱.انتخاب دو والداز میان جمعیت

۲.پیوستن دو والد وتولید فرزند

۳.ارزیابی برازندگی فرزندان

۴.جایگزین کردن برخی یاکل جمعیت اولیه بافرزندان

درمساله‌ی پوشش مجموعه یک ماتریس صفرویک با m سطر وn ستون درنظرمی‌گیریم. مساله عبارتست از پوشاندن سطرها توسط زیر مجموعه‌ای از ستونها با حداقل هزینه.(هر ستون هزینه‌ای برابرc_j دارد). تعریف می‌کنیم:

$$X_j = \begin{cases} ۱ & \text{اگر ستون } j \text{ درجواب باشد} \\ ۰ & \text{در غیر اینصورت} \end{cases}$$

$$\begin{aligned} min \quad & \sum_{j=۱}^{n} c_j x_j \\ s.t \quad & \sum_{j=۱}^{n} a_{ij} x_j \geq ۱ \qquad i = ۱, ...m \\ & x_j \in \{۰, ۱\} \qquad j = ۱, ...n \end{aligned} \tag{۱.۲}$$

همچنین بدون ازدست دادن کلیت مساله فرض کنید درمساله‌ی پوشش مجموعه،ستونها به ترتیب افزایش هزینه شان مرتب شده باشند وستونهایی با هزینه برابر به ترتیب کاهشی تعداد سطرهایی که می‌پوشانند مرتب شده باشند.(درقسمتهای بعد ازاین فرض استفاده میشود).

الف) نمایش جواب

درهرالگوریتم ژنتیک اولین گام تعبیه کردن یک نمایش جواب مناسب است.راه معمول نمایش درمساله پوشش مجموعه نمایش دودویی ۱-۰ می باشد.یک رشته‌ی n بیتی دودویی از ژنها به عنوان ساختار کروموزوم درنظرمی‌گیریم که هرژن نماینده یک ستون است و nتعداد ستونها درمساله می‌باشد.ارزش ۱ برای بیت i ام نشان دهنده این است که ستون iدرجواب هست وارزش ۰ یعنی ستون i درجواب نیست.درشکل زیر یک نمونه ازنمایش

دودویی را مشاهده می کنید:

column(gene)	1	2	3	4	5	...	$n-1$	n
bit string	1	0	1	1	0	...	1	0

نمایش دیگر بصورت غیردودویی است که در آن هرژن دریک کروموزوم نشان دهنده یک سطر می‌باشد وارزش هرژن بیانگر شماره ی ستونی است که آن سطررا پوشانده است.در شکل زیر نمونه‌ای ازاین نمایش را مشاهده می‌کنید:

row(gene)	1	2	3	4	5	...	$m-1$	m
string	10	7	10	213	5	...	49	7

یکی از مشکلاتی که نمایش دودویی با آن مواجه می‌باشد این است که تضمین طولانی مدتی برای شدنی باقی ماندن جوابها وجودندارد.یک راه برای مقابله با این مشکل استفاده ازنمایش غیردودویی است که هرچنددرطی عملیات تقاطع وجهش،شدنی بودن جواب حفظ میشود ولی ارزیابی برازندگی ممکن است مبهم باشد.به این علت که جوابهای یکسان میتوانند درفرمهای مختلف ظاهر شوند وهرفرم تابع برازندگی متفاوت به مابدهد.محاسبات تجربی نشان داده است که برای این مساله نمایش غیردودویی دررتبه دوم نسبت به نمایش دودویی قراردارد.درادامه الگوریتم ژنتیک ما ازنمایش دودویی استفاده می‌کنیم. یک راه دیگر برای مواجهه بامشکل ذکرشده استفاده از" عملگرابتکاری"[1] است که در قسمتهای بعدباآن آشنا می‌شویم. همچنین درحالت دودویی برازندگی f_i برای هر کروموزوم(جواب)

[1] Heuristic operator

i بصورت زیر محاسبه میشود:

$$f_i = \sum_{j=1}^{n} c_j s_{ij}$$

که $c_j > 0$ هزینه بیت j ام وs_{ij} ارزش بیت j ام متناظر با کروموزوم i ام می‌باشند.

پ) انتخاب والدین[1]

روشهای بسیاری برای انتخاب والدین وجود دارد.برای این مساله تکنیک مسابقه[2] بکار می رود.دراین روش بین دوگروه از کروموزوم ها که هرکدام شامل T کروموزومِ به تصادف انتخاب شده ازمیان جمعیت هستند،مسابقه‌ای شکل می‌گیرد.ازهر گروه یک کروموزوم که بهترین برازندگی را دارد انتخاب شده و عمل ترکیب میان آن دوانجام میشود.روش مسابقه‌ی دودویی به علت پیاده سازی کارآمدش برای مساله‌ی پوشش انتخاب شده است،دراین حالت داریم: $T = 2$.

ت) تقاطع[3]

تقاطع یک نقطه‌ای ودو نقطه‌ای ازروشهای اولیه برای تقاطع دو جواب هستند.درتقاطع یک نقطه‌ای فرض کنید P_1, P_2 دو والد باشند. نقطه K به تصادف انتخاب شده و ازآن نقطه برش انجام میشود.اگر C_1, C_2 دوفرزند جدید باشند و$1 \leq K < n$ ،تقاطع یک نقطه برش به شکل زیراست:

$$C_1 := P_1[1], ...P_1[K], P_2[K+1], ..., P_2[n]$$

[1]Parent selection [2]Tournament selection [3]Crossover

$$C_۲ := P_۲[۱], ..., P_۲[K], P_۱[K+۱], ..., P_۱[n]$$

تقاطع دونقطه‌ای نیز بطورمشابه انجام می‌گیرد.

چون دراین مرحله جوابهای مساله پوشش بیشتر شامل ستونهایی با هزینه کمترهستند ،بیشترژنها دربخش سمت راست کروموزومهای$P_۱, P_۲$ ارزش صفر دارند.(به یاد داریم که ستونها به ترتیب افزایش هزینه مرتب شده بودند).بنابراین اگر تقاطع چند نقطه‌ای درنقطه‌ای ازآن ناحیه رخ دهد جوابهای جدید با والدین خود یکسان خواهند شد.اما این مشکل باتعریف تقاطعی به شکل زیر، هموار خواهدشد.

دراینجا عملگر تقاطع تعمیم یافته‌ای را معرفی می‌کنیم که عملکرد آن بر اساس برازندگی می‌باشد.این عملگر هم ساختار تقاطع یک نقطه‌ای برش وهم تقاطع یکنواخت را در بر می‌گیرد و fusion operator نامیده میشود.اما برخلاف تقاطع چند نقطه‌ای برش،این تقاطع تنها یک فرزند(جواب) تولید می‌کند.فرض کنید$f_{P_۱}, f_{P_۲}$ برازندگی دو والد $P_۱, P_۲$ باشند وC فرزند باشد.مساله‌ی می‌نیمم سازی را درنظر بگیرید.برای تمام $i = ۱, ۲, ...n$ داریم:

$$(i)\ if\ P_۱[i] = P_۲[i], then\ C[i] := P_۱[i] = P_۲[i]$$

$$(ii)\ if\ P_۱[i] \neq P_۲[i], then$$

$$(a)\ C[i] := P_۱[i]\ with\ probability\ p = \frac{f_{P_۲}}{f_{P_۱} + f_{P_۲}}$$

$$(b)\ C[i] := P_۲[i]\ with\ probability\ ۱ - p$$

به عنوان مثال اگر$f_{P_1} = ۴$ و $f_{P_2} = ۶$ و$P_1[i] \neq P_2[i]$ آنگاه احتمال اینکه $C[i] := P_1[i]$ باشد برابر ۰.۶ و احتمال اینکه $C[i] := P_2[i]$ باشد برابر ۰.۴ است.پس احتمال انتخاب از والدی که برازندگی بهتری داشت[۱] بیشتر است واین نشان دهنده‌ی تاثیرمستقیم برازندگی دراین فرآیند تقاطع می‌باشد.

ث) جهش[۱]

عمل جهش پس از تقاطع ،روی جواب انجام میشود.بدین صورت که ارزش تعدادی از بیتها در یک جواب با احتمالی مشخص عوض میشود.احتمال وقوع جهش در یک جواب را نرخ جهش می‌نامند. Bäck نرخ جهش $\frac{۱}{n}$ راد برای کران پایین نرخ جهش بهینه پیشنهاد دا که n تعداد ستونهاست.مطالعات نشان داده است برای کاراتر شدن الگوریتم ژنتیک یک کران بالا نیز برای نرخ جهش لازم است.پس بکار بردن "نرخ جهش متغیر" نسبت به نرخ جهش ثابت سودمند تراست.در بخش پایانی الگوریتم ژنتیک درمورد اینکه کدام مجموعه از بیتها(ستونها) باید جهش پیدا کنند بحث خواهیم کرد.

ج) عملگر ابتکاری شدنی بودن[۲]

همانطور که اشاره شد جوابهای تولید شده توسط اعمال تقاطع وجهش ممکن است شدنی نباشند. یعنی برخی سطرها پوشیده نشوند.برای شدنی کردن جوابها یک عملگر مورد نیاز

[۱] توجه کنید که مساله می‌نیمم سازی بود پس برازندگی کمتر بهتراست.

[۱] Mutation [۲] Heuristic operator

است.عملگر شدنی بودن که دراین قسمت معرفی می‌کنیم هم جوابها را شدنی می‌کند وهم با استفاده از یک گام بهینه سازی محلی که ستونهای زائد[۱] را ازجواب حذف می کند سعی در کاراتر کردن الگوریتم دارد.نحوه‌ی کارکردن این عملگر در اگوریتم زیر آمده است.

ابتدا مجموعه های زیر رادر نظر بگیرید:

I = مجموعه‌ی تمام سطرها

J =مجموعه‌ی تمام ستونها

α_i =مجموعه‌ی ستونهایی که سطر i، $i \in I$ را می‌پوشانند

β_j =مجموعه‌ی سطرهایی که بوسیله‌ی ستون j، $j \in J$ پوشیده شده‌اند.

S =مجموعه‌ی ستونهایی که درجواب هستند.

U =مجموعه‌ی سطرهای پوشیده نشده

w_i =تعداد ستونهایی که سطر i، $i \in I$ را می‌پوشانند ودرجواب هستند.

الگوریتم:

۱. قرار بده $w_i := |S \cap \alpha_i|$ برای هر $i \in I$

۲. قراربده $U := \{i | w_i = ۰, \forall i \in I\}$

[۱] ستونهایی که باحذف آنها جواب همچنان شدنی باقی بماند

۳. برای هرسطرi درU (به ترتیب صعودی سطرها)

(الف) اولین ستون(به ترتیب صعودیj) j را در α_i پیدا کن که$\frac{c_j}{|U\cap\beta_j|}$ می‌نیمم شود.

(ب) j رابه S اضافه کن وقراربده $w_i := w_i + ۱, \forall\, i \in \beta_j$ همچنین قراربده $U := U - _j$.

۴. برای هرستون j درS (به ترتیب نزولیj) اگر۲ $w_i \geq$ ، $\forall i \in \beta_j$ ، قرار بده $S := S - j$ و ۱ $w_i := w_i -$ ، $\forall i \in \beta_j$.

۵. حال S یک جواب شدنی برای مساله‌ی پوشش است که شامل هیچ ستون زائدی نیست.

گام ۱ و۲ سطرهای پوشیده نشده را شناسایی می‌کنند. گام ۳ و ۴ گامهایی حریصانه هستند.

چ) جایگزینی جمعیت[۱]

به محض اینکه یک جواب شدنی تولید شد،این جواب با عضوی درجمعیت که بالاترین میانگین برازندگی را دارد جایگزین میشود. توجه کنید که میانگین برازندگی بالا تر یک جواب به معنی نا مناسب بودن آن جواب است.این نوع از جایگزینی جمعیت steady-state replacement نامیده میشود. مزیت این نوع جایگزینی دراین است که بهترین جوابها همیشه

[۱]Population replacement model

در جمعیت حفظ میشوند.

ح) مروری برگامهای الگوریتم

۱. یک جمعیت اولیه از N جواب تصادفی بساز وقراربده $t := ۰$

۲. دو والد $P_۱, P_۲$ را با تکنیک مسابقه ازمیان جمعیت انتخاب کن.

۳. دو والدرا با عملگر fusion operator تقاطع بده ویک جواب جدیدC را تولید کن.

۴. طبق نرخ جهش متغیر، k ستون تصادفی را جهش بده.(k درادامه توضیح داده میشود).

۵. C را بااستفاده از عملگرابتکاری شدنی بساز و همچنین ستونهای زائد را حذف کن.

۶. اگرC با هرکدام ازوالدین یکسان بود به گام ۲ برگرد. وگرنه قرار بده $t := t + ۱$ وبه گام (۷) برو.

۷. جواب C را با عضوی درجمعیت طبق تکنیک steady-state replacement جایگزین کن.

۸. گامهای ۲ تا ۷ را تکرارکن تازمانی که $t = M$ جواب غیرتکراری تولید شود.بهترین جواب پیدا شده جوابی است با کوچکترین برازندگی درجمعیت.

خ) تعیین پارامترها[۱]

جمعیت اولیه و اندازه جمعیت دو پارامترمهم هستند که باید تعیین شوند.این دو پارامتر باید طوری انتخاب شوند که دامنه جواب به قدر کافی پوشیده شود. برای درک بهتر انتخاب جمعیت اولیه، فرض کنید هر جواب اولیه S_p بوسیله‌ی الگوریتم زیر بطور تصادفی تولید میشود:(نمادهای قسمت(ج) را درنظربگیرید).

۱. قراربده $S_p := \phi$، $w_i := \phi$ برای هر $i \in I$.

۲. برای هر $i \in I$ انجام بده

(الف) یک ستونj را بطور تصادفی در α_i انتخاب کن.

(ب) ستون j را به S_p اضافه کن وقرار بده $w_i := w_i + ۱ \quad \forall i \in \beta_j$

۳. فرض کنید $T := S_p$

۴. ستون $j, j \in J$ را بطور تصادفی انتخاب کن و قرار بده $T := T - j$.

اگر $w_i \geq ۲, \forall\, i \in \beta_j$ قرار بده $S_p := S_p - j$ وقرار بده $w_i := w_i - ۱, \forall\, i \in \beta_j$

۵. گام ۴ را تازمانی که $T = \emptyset$ تکرار کن.

گام ۲ یک جواب شدنی تصادفی تولید می‌کند و گام ۴ ستونهای زائد را حذف می‌کند.

[۱] Parameter setting

حال تولید یک مساله پوشش با چگالی[۱] φ را درنظربگیرید.

اندازه جمعیت برابر $N = \mu\varphi n$ می باشد که μ میانگین تعداد ظاهر شدن یک ستون در جمعیت اولیه است.یعنی اندازه‌ی جمعیت متناظر با چگالی وتعداد ستونها می‌باشد.از این رو برای مسائل بااندازه بزرگ، جمعیت خیلی بزرگ خواهد شد واین مانع ازکارایی خوب الگوریتم ژنتیک می‌شود. برای کاهش وابستگی اندازه جمعیت به اندازه مساله(تعداد ستونها) یک مرحله از الگوریتم تولید جمعیت اولیه را اصلاح می‌کنیم.بدین صورت که جمعیت اولیه طوری تولید شود که فقط بخشی از دامنه‌ی جواب را بپوشاند.

در الگوریتم تولید جمعیت اولیه مرحله الف ازگام ۲ را بصورت زیر اصلاح می‌کنیم: ستون j را به تصادف در α_{ik} انتخاب می‌کنیم که $\alpha_{ik} \subset \alpha_i$. و α_{ik} مجموعه‌ی اولین k ستونها درα_i هستند که دارای کمترین هزینه اند. حال لازم است که جمعیت اولیه‌ی ما مجموعه‌ی$\alpha_{1k} \cup \alpha_{2k} \cup ... \cup \alpha_{mk}$ را بپوشاند که زیرمجموعه‌ای ازJ است. درحالت کلی مقدار k باید طوری انتخاب شود که احتمال اینکه ستونهای جواب بهینه (S_{opt}) زیر مجموعه‌ای از$\cup_{i=1}^{m}\alpha_{ik}$ باشند بیشترین باشد. برای مساله‌ی پوشش مجموعه وزن دار S_{opt} شامل ستونهایی با کمترین هزینه است. در مطالعات انجام شده در این زمینه مقدار۵ $k =$ برای تمام نمونه‌ها انتخاب شده است.

---

[۱] چگالی یاتراکم درمساله پوشش عبارتست از کسری ازماتریس a_{ij} که برابر یک باشد.

د) برنامه‌ریزی جهش[۱]

یکی دیگر از پارامترهای مهمی که عمل جهش به آن بستگی دارد مجموعه‌ی ستونهایی است که در یک جواب باید جهش پیدا کنند. انتخاب درست این مجموعه تاثیر به‌سزایی در کاراتر شدن الگوریتم ژنتیک دارد. واضح است مجموعه ستونهایی که بیشترین هزینه یا کمترین هزینه را دارند مناسب جهش نیستند. بنابراین برای کارآمدتر شدن الگوریتم بهتر است اجازه دهیم جهش فقط در مجموعه ستونهای "نخبه" رخ دهد. ستونهای نخبه ستونهایی هستند که شانس بیشتری برای حضور در جواب بهینه دارند. در واقع این ستونها براساس معیاری که درقسمت (خ) توضیح داده شد انتخاب میشوند. یعنی $S_{elite} = \cup_{i=۱}^{m} \alpha_{i۵}$ که$\alpha_{i۵}$ مجموعه‌ی اولین ۵ ستون با کمترین هزینه درα_i می‌باشد.

همچنین می‌توان تعداد ستونهایی که باید جهش پیدا کنند را نیز تعیین کرد. برای اطلاعات بیشتر در مورد این تعداد و نمادها وفرمول تعریف شده می‌توانید به [۴] مراجعه کنید.

۲.۴.۲ الگوریتم مورچگان

در دنیای واقعی مورچه ها ابتدا به طور تصادفی به این سو و آن سو می‌روند تا غذا بیابند. سپس به لانه برمی گردند و ردی از "فرومون[۱] " به جامی‌گذارند. مورچه‌های دیگر وقتی

[۱]Mutation scheduling [۱]pheromone

این مسیر را می‌یابند، آن را دنبال می‌کنند. سپس اگر به غذا برسند به خانه برمی‌گردند و رد دیگری از خود در کنار رد قبل می‌گذارند و مسیر قبل را تقویت می‌کنند. فرومون به مرور تبخیر میشود واین تبخیر باعث میشود مسیر جذابیت کمتری برای مورچه های بعدی داشته باشد. از آنجا که یک مورچه در زمان دراز راههای کوتاهتر را بیشتر می‌پیماید و تقویت می‌کند هر راهی بین خانه و غذا که کوتاهتر(بهتر) باشد بیشتر تقویت میشود و آنکه دورتر است کمتر. وقتی یک مورچه مسیر کوتاهی (خوبی) را از خانه تا غذا بیابد بقیه مورچه ها به احتمال زیاد همان مسیر را دنبال می‌کنند و با تقویت مداوم آن مسیر و تبخیر ردهای دیگر،به مرور همه مورچه ها هم مسیر میشوند. هدف الگوریتم مورچگان تقلید این رفتار توسط مورچه هایی مصنوعی است که روی نمودار مساله در حال حرکت‌اند.

در این بخش به بررسی یک نمونه از الگوریتم های بهینه سازی مورچگان [۲][۵] برای مساله پوشش مجموعه می‌پردازیم.

الف) طرح کلی الگوریتم مورچگان برای مساله پوشش:

مجددا مساله پوشش مجموعه را بصورت۱.۴.۲ فرض کنید.همچنین فرض کنید برای هرسطر $i \in I$ مجموعه J_j بصورت $J_j = \{j \in J | a_{ij} = ۱\}$ باشد.درواقعJ_j مجموعه ستونهایی است که سطرi را می پوشانند. برای هر ستون $j \in J$ نیز مجموعه $I_j = \{i \in$

[۲]Ant colony optimization

$\{۱ = I|a_{ij}$ رابعنوان مجموعه سطرهایی که توسط ستون j پوشیده می‌شوند درنظر بگیرید. دراین الگوریتم یک رویکرد جدید بهینه سازی گروه مورچه ها برای مساله پوشش بنام $Ant\text{-}Cover$ مطرح شده است.

تفاوت این روش با بقيه‌ي الگوريتمهاي موجود بر پايه مورچگان اين است كه برخلاف الگوريتم‌هاي ديگر كه هنگام انتخاب ستون جديد ازبين تمام ستونهاي انتخاب نشده ستون جديد را انتخاب مي‌كند اين روش ستون جديد را ازبين ستونهايي كه فقط يك سطر خاص را مي پوشانند انتخاب مي كند. بااين روش زمان كمتري براي انتخاب ستون جديد صرف ميشود. همچنين تضمين مي‌كند كه بعداز انتخاب ستون جديد حداقل يك سطر پوشيده شده است.

درالگوريتم مورچگان براي مساله پوشش مجموعه ستونها بعنوان مولفه هاي جواب در نظر گرفته ميشوند. هرستون j با يك مسير فرومون τ_j و يك ضريب η_j (مطلوبيت ستون j ام) وابسته است.براي ساختن يك جواب هرمورچه با يك جواب تهي آغاز مي كند و درهر تكرار ستونها به جواب اضافه مي شوند تا پوشش كامل شود. اين عمل بر اساس تقدم ستونها نسبت به يكديگر انجام ميشود.يعني احتمال انتخاب ستوني كه فرومون بيشتري دارد (ستون بهتر) بيشتر است. بعد از اينكه يك جواب توليد شد يك جستجوي محلي (بطور اختياري) براي بهبود كيفيت جواب روي آن انجام مي‌گيرد. به محض اينكه هر مورچه يك جواب ساخت، مسير فرومون به روز مي‌شود. طرح كلي الگوريتم مورچگان بصورت زير مي‌باشد:

۱. پارامترها و ارزش فرومون را مقداردهی کن.

۲. .تازمانی که شرایط توقف برقرار نیست انجام بده

۳. برای $antk = ۱ \ to \ nbAnts$ انجام بده

۴. S رابا$\emptyset$ مقداردهی کن. $S \Leftarrow \emptyset$

۵. تازمانی که جوابS شدنی نیست انجام بده

۶. یک ستون را برطبق احتمال آن انتخاب کن و سپس این ستون را به S اضافه کن.

۷. جستجوی محلی را برای S به کار ببر.(بطور اختیاری)

۸. مسیر فرومون را به روز کن.

۹. بهترین جواب یافت شده را برگردان.

ب) ساختن پوشش بوسیله مورچه ها در Ant-Cover: دربیشتر الگوریتم‌های موجود برپایه‌ی بهینه‌سازی مورچگان برای مساله‌ی پوشش مجموعه، درهرگام t ، ستون j ام ازمیان تمام ستونهای انتخاب نشده بااحتمال زیر انتخاب میشود:

$$P(s_t = j|S_{t-۱}) = \begin{cases} \frac{\tau_j \eta_j^\beta}{\sum_{q \in J/S_{t-۱}} \tau_q \eta_q^\beta} & if \ j \in J/S_{t-۱} \\ ۰ & o.w \end{cases}$$

که τ_j و η_j به ترتیب میزان فرومون ستون j ام و مطلوبیت ستون را نشان می‌دهند و $\eta_j = \frac{e_j}{c_j}$ کهe_j تعداد سطرهایی است که توسط ستون j پوشیده شده‌اند. نیز یک

پارامتراست.S_{t-1} جواب ساخته شده درگام قبل ازt می‌باشد. فرض کنید S یک جواب کامل باشد وn' تعداد ستونهای آن باشد. پیچیدگی زمانی ساختن جواب S بطور تقریبی $O(n'n)$ می‌باشد [۵]. پس محاسبات به‌خصوص برای مسائل بااندازه‌ی بزرگ بسیارپرهزینه خواهد بود.

می‌توانیم مساله رابافرض دیگری حل کنیم. برای اینکه یک جواب شدنی باشد باید هرسطرi توسط حداقل یک ستون j پوشیده شود. بنابراین درساخت گامt انتخاب یک سطرپوشیده نشده‌ی i وانتخاب یک ستون ازمجموعه‌ی J_i با احتمال زیر شدنی خواهد بود:

$$P(s_t = j|S_{t-1}) = \begin{cases} \frac{\tau_j \eta_j^\beta}{\sum_{q \in J_i} \tau_q \eta_q^\beta} & if\ j \in J_i \\ 0 & o.w \end{cases}$$

اگرفرض کنیم L نشان دهنده‌ی ماکزیمم تعداد ستونهایی است کهi را می‌پوشانند پس پیچیدگی زمانی محاسبه‌ی یک جواب کاملS برابر$O(n'L)$ است.[۵]

علاوه‌بر برتری در پیچیدگی ، این روش انتخاب جوابهایی باکیفیت بالاترتولید می‌کند. ترتیب انتخاب سطرهای پوشیده نشده نیز بطور تصادفی می‌باشد.

پ) به روز شدن فرومون:

در تمام تکرارها همین که مورچه یک جواب را ساخت دنباله‌های فرومون به روز میشوند. ابتدا تمام دنباله‌های فرومون بطور یکنواخت با نشاندن $\tau_j = \rho\tau_j$ برای هر $j \in J$ و$0 < \rho < 1$

تبخیرمی‌شوند.این عمل به مورچه‌هااجازه می‌دهدتابخشی از تجربه‌ی قبلی خود را فراموش کنند. پس از آن مورچه‌ها مقداری فرومون $\triangle\tau = \frac{1}{z}$ روی ستونهایی که از ابتدای الگوریتم در بهترین جواب قرار دارند به جا می‌گذارند. این جوابها با S_{gb} نشان داده میشوند. همچنین $Z = \sum_{j \in S_{gb}} c_j$ هزینه‌ی آن جواب است. این عمل مورچه‌ها رابه سمت جستجوی فضاهای جدید و وفضای امید بخش درتکراربعدی هدایت می‌کند.

ت) روند جستجوی محلی:

مي‌دانيم كه هرجواب توليد شده بوسيله Ant-Cover شدني است. پس ممكن است جواب شامل ستونهاي زائد باشد. چون سطرها بوسيله ستونهاي زودتر انتخاب شده پوشيده شده‌اند وممكن است توسط ستونهاي بعدي نيزپوشيده شوند پس آن ستونهاي اوليه زائد هستند. اين الگوريتم جستجوي محلي سعي دارد كيفيت جواب را با حذف ستونهاي زائد و همچنين جايگزيني ستونهاي كم هزينه با ستونهاي پرهزينه بهبود ببخشد. همچنين اين الگوريتم بصورت حريصانه عمل می کند. برای اطلاع از جزئیات بیشتر گامهای الگوریتم به[۵] مراجعه کنید.

ث) تعیین پارامترها:

دراین بخش نگاهی کوتاه به تاثیر هریک از پارامترهای β و ρ و $nbAnts$ درروند الگوریتم مورچگان داریم. β وزن ضریب مطلوبیت ستون یعنی η_i را نشان می‌دهد که بامقدارخیلی بزرگ الگوریتم بسرعت به سمت ایستایی هدایت میشود وبامقدارخیلی کوچک به علت

کمبودراهنمایی قوی زمان زیادی طول می‌کشد تامورچه‌هایک جواب رضایت‌بخش رابیابند. ρ نشان دهنده‌ی نرخ ماندگاری فرومون است و$nbAnts$ تعدادجوابهای تولید شده درهر تکراررا مشخص می‌کند که این نیز نباید خیلی کوچک یا خیلی بزرگ انتخاب شود.

درعین حال هیچ روش تحلیلی برای معین کردن این پارامترها بطورمستقیم وجود ندارد و معمولا این پارامترها ازطریق تجربی بدست می‌آیند.

۳.۴.۲ مقایسه‌ی نتایج محاسباتی الگوریتم ژنتیک ومورچگان

دراین بخش به مقایسه الگوریتم مورچگان بدون جستجوی محلی والگوریتم مورچگان با جستجوی محلی و الگوریتم ژنتیک ذکر شده در بخش قبل می‌پردازیم.

دقت کنید نتایج درون جدول، میانگین جوابهایی هستند که در۱۰ اجرای مستقل بدست آمده اند.

name	opt	GA	Ant-cover	Ant-cover+LS
۴.۱	۴۲۹	۴۲۹.۶	۴۳۱.۳	۴۲۹.۰
۴.۲	۵۱۲	۵۱۲.۰	۵۱۷.۷	۵۱۲.۰
۴.۸	۴۹۲	۴۹۲.۱	۴۹۳.۴	۴۲۹.۱
۵.۱	۲۵۳	۲۵۳.۰	۲۵۵.۸	۲۵۳.۰
۶.۱	۱۳۸	۱۳۸.۰	۱۴۰.۸	۱۳۸.۰
۶.۲	۱۴۶	۱۴۶.۲	۱۴۷.۰	۱۴۶.۰
۶.۳	۱۴۵	۱۴۵.۰	۱۴۵.۱	۱۴۵.۰

همانطور که مشاهده می‌کنید زمانی که جستجوی محلی به الگوریتم مورچگان اضافه می‌شود نتیجه‌ی بدست آمده مطلوب تر است و به غیراز یک نمونه در تمام نمونه‌ها جواب بهینه را یافته است. همچنین الگوریتم ژنتیک ارائه شده نیز در۳ نمونه جواب بهینه را یافته است.

مراجع

[1] Thomas H. Cormen,Charles E. Leiserson, Ronald L. Rivest, Clifford Stein:Introduction to algorithms,MIT press (2001),second edition.

[2] B. Korte and J. Vygen: Combinatorial optimization theory and algorithms, Springer(2008) ,fourth edition.

[3] W. L. Winston, Operations research: Applications and algorithms, Duxbury Press, 2003.

[4] J.E. Beasley , RC. Chu,"A genetic algorithm for the set covering problem", European Journal of Operational Research (1996).

[5] Zhigang Ren,Zuren Feng,Liangjun Ke,Hong chang,"A fast and efficient Ant colony optimization Approach for the set covering problem",IEEE (2008).

[6] Fernando C. Gomesa,, Panos M. Pardalosb, Gerardo Valdisio R. Vianaa," Experimental Analysis of Approximation Algorithms for the Vertex Cover and Set Covering Problems", Computers Operations Research (2006).

[7] Raf Jans ,Zeger Degraeve,"A note on a symmetrical set covering problem:The lottery problem", European Journal of Operational Research (2008).

[8] Guanghui Lan ,Gail W. DePuy ,Gary E. Whitehouse," An effective and simple heuristic for the set covering problem", European Journal of Operational Research (2007).

[9] Nysret Musliu," Local search algorithm for unicost set covering problem", Vienna University of Technology.

[10] David P.Williamson,"An introduction to approximation algorithm",Lecture (2000).

[11] Pezzella, F., Faggioli, E., 1997. Solving large set covering problems for crew scheduling.

[12] Tamara Stern,"set cover problem ",seminar in Theorical computer science(2006).