Sharif University of Technology

Department of Computer Engineering

M.Sc. Thesis

Approximation Algorithms for Clustering Points in the Distributed Model

By:

Emad Aghajani

Supervisor:

Dr. Hamid Zarrabi-Zadeh

July 2016

Abstract

Clustering is one of the most well-known fundamental problems in computer science. In this thesis we have focused on a particular version of such problem, called capacititated k-center, and we analyze and survey them in the distributed model, when massive data is given. Our contribution in this research includes a new approximation algorithms with constant approximate factors for such problems in the distributed model, as well as improving the best available algorithm for capacitated k-center problem. Composable coreset as one of the most important techniques in distributed and streaming model is our primary tools in designing these algorithms. This technique gives the opportunity of solving the problem in smaller chunks of data, and giving the result by combining them.

Keywords: Approximation algorithm, Composable coreset, Clustering, Capacitated k-center, Distributed model

دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

پایان‌نامه‌ی کارشناسی ارشد
گرایش مهندسی نرم‌افزار

عنوان:

الگوریتم‌های تقریبی برای خوشه‌بندی نقاط در مدل توزیع‌شده

نگارش:

عماد آقاجانی

استاد راهنما:

دکتر حمید ضرابی‌زاده

مرداد ۱۳۹۵


بسم الله الرحمن الرحیم

به نام خدا

دانشگاه صنعتی شریف

دانشکده‌ی مهندسی کامپیوتر

پایان‌نامه‌ی کارشناسی ارشد

عنوان: الگوریتم‌های تقریبی برای خوشه‌بندی نقاط در مدل توزیع‌شده

نگارش: عماد آقاجانی

کمیته‌ی ممتحنین

استاد راهنما: دکتر حمید ضرابی‌زاده امضاء:

استاد مشاور: دکتر محمد علی آبام امضاء:

استاد مدعو: دکتر علیرضا زارعی امضاء:

تاریخ:

سپاس

نخستین سپاس و ستایش از آن خداوندی است که بنده کوچکش را در دریای بیکران اندیشه، قطره‌ای ساخت تا وسعت آن را از دریچه اندیشه‌های ناب آموزگارانی بزرگ به تماشا نشیند. لذا اکنون‌که در سایه‌سار بنده‌نوازی‌هایش پایان‌نامه حاضر به انجام رسیده است، بر خود لازم میدانم تا مراتب سپاس را از بزرگوارانی به‌جا آورم که اگر دست یاریگرشان نبود، هرگز این پایان‌نامه به انجام نمی‌رسید.

ابتدا از استاد گران‌قدرم جناب آقای دکتر ضرابی‌زاده که زحمت راهنمایی این پایان‌نامه را بر عهده داشتند، کمال سپاس رادارم.

سپس از دوست عزیزم، مهندس کیان میرجلالی، که در به دست آوردن نتایج این مقاله همکاری داشتند، صمیمانه سپاس‌گزارم.

و سپاس ویژه به مهربان‌ترین همراهان زندگی‌ام، به پدر، مادر و برادر عزیزم تقدیم می‌کنم که حضورشان در فضای زندگی‌ام مصداق بی‌ریای سخاوت بوده است.

چکیده

مسائل خوشه‌بندی نقاط از مهم‌ترین و شناخته‌شده‌ترین مسائل پایه‌ای در علوم کامپیوتر است. در این پژوهش ما بر روی دسته‌ی خاصی از این مسائل که با نام «k-مرکز ظرفیت دار» شناخته می‌شوند تمرکز خواهیم کرد، و سعی می‌کنیم تا این دسته از مسائل را در مدل توزیع‌شده که با حجم زیادی از داده روبرو هستیم، بررسی نموده و الگوریتم‌های تقریبی جدیدی برای آن ارائه کنیم. در ارائه این الگوریتم‌ها از شیوه «مجموعه‌های هسته‌ی ترکیب‌پذیر» که در مدل‌های توزیع‌شده موردتوجه قرارگرفته است، کمک گرفته‌شده است. این مفهوم به ما کمک می‌کند تا به ازای داده‌های حجیم، مسئله را به بخش‌های کوچک شکسته و از اجتماع نتایج، برای ارائه یک جواب تقریبی استفاده کنیم. ارائه چند الگوریتم تقریبی با ضریب تقریب ثابت برای نسخه‌های مختلف مسئله k-مرکز ظرفیت دار در مدل توزیع‌شده و بهبود ضریب تقریب الگوریتم‌های فعلی، از نتایج این پژوهش به‌شمار می‌آید.

کلیدواژه‌ها: الگوریتم تقریبی، مجموعه‌های هسته‌ی ترکیب‌پذیر، خوشه‌بندی، k-مرکز ظرفیت دار، مدل توزیع‌شده

فهرست مطالب

فهرست شکل‌ها

فهرست جدول‌ها

فصل ۱

مقدمه

در این فصل مقدمه‌ای از مسائل موردنظر در این پژوهش و اهداف و انگیزه‌های آن‌ها را ارائه خواهیم داد و به معرفی مسائل خوشه‌بندی و به‌طور خاص مسئله‌ی k-مرکز و نسخه‌های ظرفیت‌دار آن که هدف اصلی این پژوهش است می‌پردازیم. سپس به نحوه‌ی برخورد با این دسته از مسائل به ازای ورودی‌های بزرگ و پژوهش‌های انجام‌شده در مدل توزیع‌شده و به‌طور خاص با استفاده از مفهوم مجموعه‌های هسته، تمرکز خواهیم کرد.

مسئله‌ی خوشه‌بندی نقاط، یکی از مهم‌ترین و شناخته‌شده‌ترین دسته مسائل پایه‌ای است که کاربردهای آن به‌طور گسترده در صنعت مشاهده می‌شود. به‌عنوان نمونه می‌توان مجموعه‌ای از نقاط که فاصله بین نقاط میزان شباهت را نشان می‌دهد در نظر گرفت. در این صورت مسائل خوشه‌بندی به دنبال ارائه یک دسته‌بندی از نقاط خواهند بود طوری‌که اجسام مشابه در یک دسته قرار بگیرند و این موضوع می‌تواند در موتورهای جستجو مورداستفاده قرار بگیرد. مسائلی از قبیل k-مرکز و k-میانه، ازجمله این مسائل خوشه‌بندی می‌باشند که در تعریف تابع هدف [۱] تفاوت‌های جزئی با یکدیگر دارند. همچنین در دسته‌ای از مسائل، عمل خوشه‌بندی با محدودیت‌های جدید همراه می‌شود. به‌عنوان نمونه، در نسخه‌های ظرفیت‌دار مسئله‌ی k-مرکز ، برای اندازه هر خوشه یک محدودیت در نظر گرفته می‌شود.

ازآنجاکه اغلب مسائل خوشه‌بندی ان‌پی-سخت [۲] هستند [۱] و نمی‌توان در زمان چندجمله‌ای پاسخ

[۱] Objective Function

[۲] NP-hard

دقیقی برای آن‌ها به دست آورد، (مگر $P = NP$)، عموماً پژوهشگران به دنبال ارائه الگوریتم‌هایی برای حل تقریبی مسئله بوده‌اند.

از طرف دیگر، در سال‌های اخیر حجم داده‌های خامی که نیازمند پردازش می‌باشند به شکل گسترده‌ای افزایش یافته است. به همین دلیل استفاده از مدل‌های توزیع‌شده و چارچوب‌هایی همچون نگاشت-کاهش مورد استقبال قرار گرفته است. در این مدل‌ها سعی می‌شود تا مسئله توسط تعدادی ماشین مجزا و به‌صورت موازی حل شود. بین ماشین‌ها می‌تواند به‌صورت محدود تبادلات داده‌ای وجود داشته باشد.

در این پژوهش قصد داریم تا چند روش تقریبی برای حل مسئله‌ی k-مرکز ظرفیت‌دار در فضای توزیع‌شده ارائه دهیم. این پژوهش بر دو نسخه از مسئله‌ی k-مرکز ظرفیت‌دار، نسخه با ظرفیت‌های نرم و نسخه‌ی با ظرفیت‌های سخت، تمرکز خواهد داشت. الگوریتم‌های ارائه‌شده در این پژوهش، یک ضریب تقریب بهبودیافته نسبت به الگوریتم‌های موجود ارائه خواهند داد.

۱-۱ تعریف مسئله

مسئله اصلی موردبحث در این پژوهش مسئله‌ی k-مرکز است. هدف از مسئله k-**مرکز** انتخاب مجموعه‌ی P از k نقطه از میان مجموعه نقاط ورودی مسئله S است به‌طوری‌که فاصله دورترین نقطه از مجموعه نقاط P کمینه باشد[۳]. هر یک از نقاط مجموعه P نماینده یک خوشه فرض می‌شوند و در اصطلاح «مرکز» یک خوشه نامیده می‌شوند. در بعضی فرمول‌بندی‌های خوشه‌بندی، خوشه‌بندی تنها با انتخاب مرکزها انجام می‌گیرد و نیازی به انتساب سایر نقاط به این مراکز نیست. چراکه اگر هر نقطه را به نزدیک‌ترین مرکز خود منتسب کنیم به یک خوشه‌بندی از نقاط دست پیدا خواهیم کرد؛ اما این حرف تنها تا زمانی درست است که محدودیتی بر اندازه هر خوشه وجود نداشته باشد و در نسخه‌های ظرفیت‌دار که موردعلاقه ما است، نیاز است تا نحوه انتساب نیز به‌طور دقیق مشخص گردد.

با این فرض که مسئله در فضای متری[۴] موردبحث است، می‌توان ورودی مسئله را یک گراف تصور نمود. در این صورت، مسئله‌ی k-مرکز به دنبال انتخاب k رأس از گراف و انتساب سایر رئوس به آن‌ها است به‌طوری‌که بیشترین فاصله از یک رأس تا مرکز منتسب شده به آن که آن را **شعاع خوشه** می‌نامیم،

[۳] فاصله یک نقطه مانند p از یک مجموعه نقطه S برابر فاصله نقطه p تا نزدیک‌ترین نقطه از مجموعه S است

[۴] Metric

کمینه باشد. این مسئله به‌صورت رسمی به‌صورت زیر تعریف می‌شود:

تعریف ۱-۱ (مسئله‌ی k-مرکز) *به ازای گراف وزن‌دار $G = (V, E)$، زیرمجموعه‌ی $S \subseteq V$ با اندازه حداکثر k پیدا کنید طوری‌که عبارت زیر کمینه شود.*

$$\max_{u \in V} \min_{v \in S} d(u, v)$$

مسئله‌ی k-مرکز به شکل‌های مختلف قابل‌تعمیم است. در دسته‌ای از مسائل به ازای هر نقطه از ورودی به‌عنوان یک مرکز، یک هزینه بازگشایی برای آن نقطه در نظر گرفته می‌شود که به مسئله «مکان‌یابی تسهیلات [۵]» مشهور است. در دسته‌ای دیگر از پژوهش‌ها، هدف کم کردن تعداد مراکز با یک شعاع ثابت است. دسته‌ای از پژوهش‌ها نیز به اعمال ظرفیت بر روی اندازه هر خوشه می‌پردازند و به مسئله «k-مرکز ظرفیت‌دار» مشهور است که ما در این پژوهش به‌طور خاص بر روی آن تمرکز خواهیم داشت. همچنین در نسخه‌ی خاصی از مسئله‌ی k-مرکز ظرفیت‌دار، به ازای هر نقطه یک وزن قائل می‌شویم. در این صورت پوشش یک نقطه، به‌اندازه وزن آن نقطه، از ظرفیت مرکز استفاده می‌کند. این نسخه از مسئله به «نسخه وزن‌دار مسئله‌ی k-مرکز ظرفیت‌دار» مشهور است. این نسخه از مسئله به‌عنوان ابزاری برای ارائه الگوریتم نهایی مورداستفاده قرار خواهد گرفت و در بخش ۴-۲ به تفضیل به آن پرداخته خواهد شد.

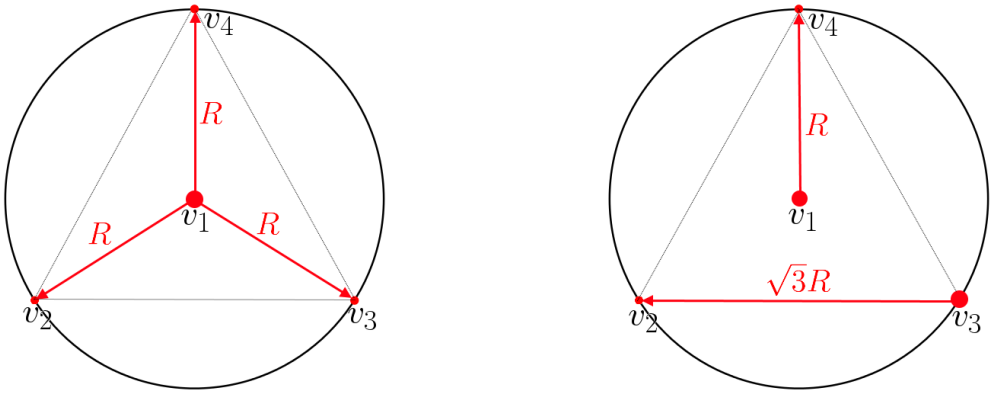
در مسئله k-**مرکز ظرفیت‌دار**[۶] محدودیت حداکثر تعداد نقاط مجاز در هر خوشه (حداکثر نقاط قابل انتساب به یک مرکز) به مسئله اصلی اضافه می‌شود. این نسخه باانگیزه‌ی هرچه کاربردی‌تر شدن مسائل موردتوجه قرارگرفته است. برای نمونه فرض کنید که ما به دنبال قرار دادن دو آنتن ($k = ۲$) بر روی مجموعه رئوس $V = \{v_۱, v_۲, v_۳, v_۴\}$ هستیم، به‌طوری‌که شعاع لازم برای پوشش تمام نقاط کمینه باشد (همانند مسئله‌ی k-مرکز بیان‌شده).

پاسخ بهینه برای این مسئله (شعاع برابر با R) در شکل ۱-۱-سمت چپ نمایش داده‌شده است؛ اما در دنیای واقعی یک آنتن ظرفیت محدودی در سرویس‌دهی دارد. اگر ظرفیت تمام آنتن‌ها را، به‌صورت یکسان، یک واحد فرض کنیم ($L = ۱$)، در این صورت جواب بهینه مسئله متفاوت خواهد بود (شعاع برابر با $\sqrt{۳}R$) که در شکل ۱-۱-سمت راست نمایش داده شده است.

---

[۵] Facility Location

[۶] Capacitated k-Center



شکل ۱-۱: مقایسه شعاع بهینه مسئله k-مرکز برای حالت بی‌ظرفیت و ظرفیت‌دار

مسئله k-مرکز ظرفیت‌دار به‌صورت رسمی به‌صورت زیر تعریف می‌شود:

تعریف ۱-۲ (مسئله‌ی k-مرکز ظرفیت‌دار) *به ازای گراف وزن‌دار $G = (V, E)$ ، زیرمجموعه‌ی $S \subseteq V$ با اندازه حداکثر k پیدا کنید طوری‌که حداکثر L رأس به هر مرکز از S منتسب شده باشد و عبارت زیر نیز کمینه شود.*

$$\max_{u \in V} \min_{v \in S} d(u, v)$$

تاکنون نسخه‌های ظرفیت‌دار با تعاریف متعددی در پژوهش‌های مختلف موردمطالعه قرارگرفته‌اند. خولر و سوسمان [۲] دو نسخه ظرفیت‌دار پایه‌ای از این مسئله با نام‌های «k-مرکز با ظرفیت‌های نرم[۷]» و «k-مرکز با ظرفیت‌های سخت[۸]» را موردبررسی قرار داده‌اند.

در تعریف مسئله k-**مرکز با ظرفیت‌های سخت**، ظرفیت هر خوشه حداکثر L در نظر گرفته می‌شود. در مسئله k-**مرکز با ظرفیت‌های نرم** در کنار محدودیت حداکثر Lتایی بر روی هر خوشه، این اجازه داده می‌شود تا یک نقطه بیش از یک‌بار به‌عنوان مرکز استفاده شود. در این صورت ما می‌توانیم در هر نوبت از الگوریتم، بدون در نظر گرفتن انتخاب‌های قبلی، هر یک از نقاط ورودی را به‌عنوان مرکز انتخاب کنیم.

[۷] k-Center with soft capacities / Capacitated multi-k-Center

[۸] k-Center with hard capacities / L-Balanced k-Center

۱-۲ اهمیت موضوع

حجم گسترده مطالعات انجام‌شده بر روی دسته مسائل خوشه‌بندی از یک‌سو، و میزان توجه پژوهشگران به حالت‌های خاص و نسخه‌های گوناگون مسئله از سوی دیگر، می‌تواند اهمیت این طیف از مسائل را در دنیای علوم کامپیوتر به‌خوبی نشان دهد. در رابطه با مسئله‌ی k-مرکز به‌طور خاص، کاربردهای عملی گوناگونی وجود دارد که ازجمله آن‌ها می‌توان به موضوع مکان‌یابی تسهیلات، دسته‌بندی داده‌ها و موتورهای جستجو اشاره نمود. همچنین گفتنی است که دسته‌ای از پژوهش‌ها به‌طور خاص به بررسی کاربردهای مسئله‌ی k-مرکز پرداخته‌اند [۳].

با توجه به آنکه در بعضی از کاربردها مانند مثالی که در بخش قبل ارائه شد، وجود تعادل در خوشه‌بندی ضروری است، نسخه‌های ظرفیت‌دار از مسئله‌ی k-مرکز مطرح‌شده‌اند. همچنین تعداد پژوهش‌های صورت گرفته در یک دهه اخیر بر روی این دسته از مسائل نشان‌دهنده توجه ویژه پژوهشگران و صنعتگران به این مسائل است. همچنین ازآنجاکه در عمل یکی از بزرگ‌ترین کاربردهای مسائل خوشه‌بندی برای تحلیل مجموعه داده‌های حجیم [۹] است و امکان پردازش چنین حجم از داده‌هایی به‌صورت ترتیبی و متمرکز وجود ندارد، پژوهشگران به دنبال روش‌های حل توزیع‌شده و یا تقریبی مسئله می‌باشند.

۱-۳ ادبیات موضوع

با توجه به آنکه مسئله اصلی موردبحث در این پژوهش ارائه یک الگوریتم تقریبی برای حل مسئله‌ی k-مرکز ظرفیت‌دار در مدل توزیع‌شده است، ما با دو طیف گسترده از پژوهش‌ها در ارتباط خواهیم بود. دسته اول پژوهش‌هایی هستند که به مسائل خوشه‌بندی و به‌طور خاص به مسئله‌ی k-مرکز و نسخه‌های ظرفیت‌دار آن توجه دارند و دسته دوم نیز مطالعات مربوط به مدل‌ها و شیوه‌های توزیع‌شدگی از قبیل مدل نگاشت-کاهش است. همچنین آن دسته از پژوهش‌ها که به ترکیب این دو پرداخته‌اند نیز موردعلاقه ما است.

پیش‌تر ثابت شده است که دسته مسائل خوشه‌بندی غالباً ان‌پی-سخت [۱۰] هستند[۱] و نمی‌توان در

[۹] Massive Data

[۱۰] NP-hard

زمان چندجمله‌ای پاسخ دقیقی برای آن‌ها به دست آورد، (مگر $P = NP$). درنتیجه، عموماً پژوهشگران این حوزه، به دنبال ارائه الگوریتم‌هایی برای حل تقریبی مسئله و یا نسخه‌های خاصی از آن بوده‌اند. یک الگوریتم تقریبی با تقریب ρ برای یک مسئله کمینه‌سازی، یک الگوریتم در زمان چندجمله‌ای است که یک جواب با هزینه حداکثر ρ برابر هزینه جواب بهینه را تضمین می‌کند. به‌طور خاص برای نسخه اصلی از مسئله‌ی k-مرکز بهترین الگوریتم ممکن که یک الگوریتم ۲-تقریب است در ۱۹۸۵ توسط گنزالس [۴] ارائه‌شده است.

همان‌طور که پیش‌تر بیان شد، مسئله‌ی k-مرکز به شکل‌های مختلفی تعمیم پیداکرده است که سه نسخه ظرفیت‌دار زیر، به‌طور ویژه موردتوجه ما خواهند بود:

۱. مسئله‌ی k-مرکز با ظرفیت‌های نرم

۲. مسئله‌ی k-مرکز با ظرفیت‌های سخت

۳. نسخه وزن‌دار مسئله‌ی k-مرکز با ظرفیت‌های نرم

تاکنون نسخه‌های ظرفیت‌دار با تعاریف متعددی در پژوهش‌های مختلف موردمطالعه قرارگرفته‌اند. خولر و سوسمان [۲] دو نسخه ظرفیت‌دار پایه‌ای از این مسئله، k-مرکز با ظرفیت‌های نرم و سخت را موردبررسی قرار داده‌اند و دو الگوریتم با ضرایب تقریب ۵ (برای نسخه نرم) و ۶ (برای نسخه سخت) ارائه‌شده است.

همچنین خولر و سایرین [۵] و هونگ چان آن و همکاران [۶] نیز اخیراً بر روی نسخه‌ای کلی‌تری از مسئله که در آن ظرفیت نقاط یکسان نباشد، مطالعه انجام داده‌اند که نتیجه آن، یک الگوریتم ۹-تقریب با استفاده از شیوه برنامه‌ریزی خطی است. این نسخه از مسئله که فرم کلی‌تر نسخه‌های پیشین است، با نام «k-مرکز با ظرفیت‌های غیر یکسان[۱۱]» شناخته می‌شود.

همان‌طور که در ابتدا بیان شد، ما به دسته دومی از پژوهش‌ها که به بحث توزیع‌شدگی پرداخته‌اند نیز توجه خواهیم داشت. مدل نگاشت-کاهش[۱۲]، به‌عنوان یکی از چارچوب‌های[۱۳] استاندارد محاسبات توزیع‌شده، به‌طور ویژه در بحث توزیع‌شدگی موردتوجه پژوهشگران قرار دارد و بسیاری از پژوهش‌های

[۱۱] k-Center with non-uniform capacities

[۱۲] MapReduce

[۱۳] Framework

این دسته به این مفهوم پرداخته‌اند. در این مدل داده بین تعدادی ماشین توزیع می‌گردد و در هر ماشین محاسبات بر روی بخشی از داده‌ها صورت می‌پذیرد، و این عمل در طی چندین دور[۱۴] تکرار می‌شود. یک مدل رسمی از مدل نگاشت-کاهش در پژوهش [۷] ارائه شده است.

به‌کارگیری این مدل برای حل مسئله‌ی k-مرکز اولین بار در پژوهش [۸] انجام‌شده است. همچنین در کار اخیر ایم و موسلی [۹]، مسئله‌ی k-مرکز با نقاط دورافتاده[۱۵] در مدل نگاشت-کاهش موردبازنگری قرارگرفته است. در این پژوهش یک الگوریتم ۲-تقریب برای نسخه اصلی مسئله و یک الگوریتم ۴-تقریب برای نسخه با نقاط دورافتاده ارائه گشته است.

عموماً در مدل توزیع‌شده ما به دنبال آن هستیم که الگوریتم به‌طور موازی (در هر ماشین) بر روی بخشی از ورودی‌های مسئله اجرا شود. این نیاز در مفهومی بنام «مجموعه‌های هسته‌ی ترکیب‌پذیر» موردتوجه قرارگرفته است. به زبان ساده، یک مجموعه‌ی هسته‌ی ترکیب‌پذیر برای مجموعه S، زیرمجموعه‌ای مانند $T \subseteq S$ است که حل مسئله بر روی T به ما تقریبی از جواب بهینه برای کل ورودی می‌دهد. همچنین ترکیب‌پذیری این مجموعه‌های هسته این امکان را به ما می‌دهد تا در مدلی مانند نگاشت-کاهش، با یک پردازش نهایی بر روی خروجی مرحله اول ماشین‌ها، به یک تقریب خوب از پاسخ بهینه به ازای کل داده‌های ورودی برسیم. این مفهوم در کارهای اخیر ضرابی‌زاده و سایرین [۱۰] و ایندیک و سایرین [۱۱] قابل‌مشاهده است.

استفاده از مفهوم مجموعه‌های هسته‌ی ترکیب‌پذیر برای حل مسائل خوشه‌بندی بسیار محدود بوده است و ما به دنبال تمرکز بر روی این حوزه هستیم. به‌طور خاص در کار اخیر باطنی و همکاران [۱۲] چند الگوریتم تقریبی با ضریب تقریب ثابت برای مسائل خوشه‌بندی و خوشه‌بندی ظرفیت‌دار در حالت توزیع‌شده ارائه‌شده است که از مفهوم مجموعه‌های هسته‌ی قابل‌ترکیب استفاده‌شده است.

۱-۴ اهداف تحقیق

در این پایان‌نامه سعی می‌شود که مسئله‌ی خوشه‌بندی نقاط، نسخه‌های مختلف آن و حل مسئله در فضای توزیع‌شده موردمطالعه قرار گیرد. به‌طور ویژه مسئله‌ی k-مرکز با ظرفیت‌های نرم، k-مرکز با ظرفیت‌های

[۱۴] Round

[۱۵] Outlier

سخت و نسخه وزن‌دار آن موردتوجه ما قرار خواهد گرفت. در این پژوهش، پس از مطالعه دقیق کارهای مرتبط سعی شده است تا الگوریتم‌های موجود بازبینی گردند و در صورت امکان الگوریتم‌های جدید و بهبودیافته‌ای ارائه گردد. همچنین در عمل، در این پژوهش چهار الگوریتم جدید برای حل نسخه‌های مختلف مسئله‌ی k-مرکز ظرفیت‌دار ارائه گشته است.

۱-۵ ساختار پایان‌نامه

این پایان‌نامه شامل شش فصل است. فصل دوم دربرگیرنده تعاریف اولیه مرتبط با پایان‌نامه خواهد بود. در فصل سوم پژوهش‌های مرتبط با مسائل خوشه‌بندی و الگوریتم‌های توزیع‌شده برای حل این مسائل به‌تفصیل بیان می‌شود. در فصل چهارم و پنجم نتایج جدیدی که در این پایان‌نامه به دست آمده ارائه می‌گردد. در فصل چهار، یک الگوریتم جدید برای حل نسخه وزن‌دار مسئله‌ی k-مرکز با ظرفیت‌های نرم ارائه می‌گردد. سپس در فصل پنجم سه الگوریتم دیگر برای حل نسخه ظرفیت‌دار مسئله‌ی k-مرکز ارائه می‌گردد. همچنین در پایان، بین نتایج حاصل از این پژوهش و نتایج موجود، مقایسه‌ای صورت می‌گیرد. در پایان، فصل پنجم به نتیجه‌گیری و پیشنهادهایی برای کارهای آتی خواهد پرداخت.

فصل ۲

مفاهیم اولیه

در این فصل از پایان‌نامه به تعریف و بررسی مفاهیم اولیه و پیش‌نیاز مطالبی خواهیم پرداخت که در فصل‌های آتی از آن‌ها استفاده خواهد شد.

۲-۱ الگوریتم‌های تقریبی

برای حل مسائلی که در کلاس ان‌پی-سخت قرار دارند انتظار نمی‌رود که یک الگوریتم چندجمله‌ای وجود داشته باشد. همچنین از سوی دیگر، مسئله‌ی برابری یا عدم برابری P و NP مدت‌ها است که یک مسئله‌ی حل‌نشده باقی‌مانده است. لذا برای حل این دسته از مسائل، نیازمند الگوریتم‌هایی با پیچیدگی زمانی چندجمله‌ای هستیم تا بتوانند دست‌کم جوابی نزدیک به جواب بهینه داشته باشند و درعین‌حال کران بالایی از حداکثر نسبت جواب تقریبی به جواب بهینه داشته باشند. لذا الگوریتم‌های تقریبی از الگوریتم‌های ابتکاری[۱] که تنها جواب نسبتاً خوبی ارائه می‌دهند تفاوت داشته و برخلاف آن‌ها می‌توانند تضمینی نسبت به کارایی جواب داشته باشند.

تعریف ۲-۱ (الگوریتم‌های تقریبی) *به الگوریتم‌هایی گفته می‌شود که کران بالایی از نسبت جواب تقریبی به جواب بهینه را دارا هستند. این کران بالا* ***ضریب تقریب***[۲] *خوانده می‌شود.*

[۱] Heuristic

[۲] Approximation Factor

تعریف ۲-۲ (الگوریتم‌های تقریبی ضریب ثابت) *الگوریتم‌های تقریبی که ضریب تقریب ثابت و مستقل از اندازه و شرایط مسئله‌ی ورودی دارند را الگوریتم‌های تقریبی ضریب ثابت[۳] می‌خوانیم.*

یکی از مسائل کلاسیک کلاس NP ، مسئله‌ی پوشش رأسی [۴] است که الگوریتم تقریبی با ضریب ثابت ۲ برای آن وجود دارد که به‌صورت زیر بیان می‌گردد.

تعریف ۲-۳ (مسئله‌ی پوشش رأسی) *با داشتن گراف ساده و بی‌جهت $G = (V, E)$ ، می‌خواهیم مجموعه $S \subset V$ را به‌گونه‌ای پیدا کنیم که $|S|$ کمینه بوده و رابطه‌ی $\forall (u, v) \in E : u \in S \vee v \in S$ برقرار باشد.*

مسئله پوشش رأسی توسط یک الگوریتم حریصانه ساده حل می‌شود و به‌راحتی می‌توان نشان داد که این الگوریتم حریصانه، الگوریتمی تقریبی با ضریب ثابت ۲ است. برای حل مسئله به روش حریصانه، یک یال دلخواه uv را انتخاب می‌کنیم و سپس هر دو رأس u و v را به مجموعه M اضافه و کلیه یال‌های متصل به آن‌ها را از مجموعه‌ی یال‌های E حذف می‌کنیم. این کار را تا زمانی ادامه می‌دهیم که مجموعه‌ی E تهی شود. مجموعه M یک پوشش رأسی است، زیرا با تهی شدن E، حداقل یکی از گره‌های ابتدا و انتهای یال‌ها در مجموعه‌ی M وجود خواهند داشت. همچنین مشخص است که از هر یال uv که در حلقه‌ی الگوریتم حریصانه از مجموعه E انتخاب کرده بوده‌ایم، رأس u یا v و یا هردو، باید در مجموعه جواب بهینه موجود باشند. ازآنجایی‌که در بدترین حالت در هر مرحله تنها یکی از رئوس مجموعه جواب بهینه را انتخاب می‌کنیم، کران بالای ضریب تقریب این الگوریتم، برابر با عدد ثابت ۲ خواهد بود.

گروه دیگر از الگوریتم‌های تقریبی، الگوریتم‌های $PTAS$ [۵] هستند. الگوریتم‌های تقریبی از این گروه، با داشتن هر $\epsilon > 0$ به‌عنوان ورودی، می‌تواند جوابی با تقریب $1 + \epsilon$ (در مسئله‌های کمینه‌سازی) و یا $1 - \epsilon$ (در مسئله‌های بیشینه‌سازی) از جواب بهینه ارائه دهند. الگوریتم‌های $PTAS$ باید به ازای هر ϵ ثابت، نسبت به‌اندازه‌ی ورودی چندجمله‌ای باشند. درنتیجه مرتبه‌ی زمانی این الگوریتم‌ها می‌تواند حتی $O(n^{(\frac{1}{\epsilon})!})$ باشد؛ بنابراین الگوریتم‌های $PTAS$ نسبت به ϵ ممکن است چندجمله‌ای نباشند.

درنتیجه گروه الگوریتم‌های $EPTAS$ [۶]، زیرمجموعه‌ای از گروه الگوریتم‌های $PTAS$، تعریف می‌شوند که زمان اجرای آن‌ها به‌صورت $O(n^c)$ بوده، که در این رابطه c مستقل از ϵ است. البته حتی در

[۳] Constant Factor Approximation Algorithms

[۴] Vertex Cover

[۵] Polynomial-time Approximation Scheme

[۶] Efficient Polynomial-time Approximation Scheme

الگوریتم‌های گروه $EPTAS$ نیز همچنان ϵ می‌تواند به‌صورت نمایی در ضریب ثابت O ظاهر شود. گروه الگوریتم‌های $FPTAS$[۷] امکان نمایی شدن مرتبه زمانی نسبت به‌اندازه‌ی مسئله و $\frac{۱}{\epsilon}$ را از میان برده و تمامی الگوریتم‌های طبقه‌بندی‌شده در این گروه، نسبت به هر دو پارامتر اندازه‌ی مسئله و $\frac{۱}{\epsilon}$ چندجمله‌ای هستند.

۲-۲ خوشه‌بندی

مسائل خوشه‌بندی یکی از قدیمی‌ترین و پایه‌ای‌ترین مسائل علوم کامپیوتر می‌باشند و در طول زمان به‌طور گسترده‌ای موردمطالعه قرارگرفته‌اند. به ازای یک مجموعه نقاط ورودی، این مسائل به دنبال ارائه یک دسته‌بندی از نقاط هستند که به هر دسته در اصطلاح یک «خوشه[۸]» گفته می‌شود. همچنین معمولاً یک نقطه از هر خوشه به‌عنوان نماینده آن خوشه در نظر و به‌عنوان «مرکز خوشه[۹]» شناخته می‌شود و درنتیجه خوشه‌بندی می‌تواند تنها با مشخص کردن این مراکز صورت بپذیرد.

مسائل خوشه‌بندی معمولاً به دنبال ارائه یک خوشه‌بندی همراه با کمینه‌سازی یک پارامتر خاص از آن (مانند تعداد خوشه‌ها) هستند که در اصطلاح «تابع هدف[۱۰]» مسئله خوانده می‌شود. به همین دلیل، مسائل خوشه‌بندی به‌عنوان مسائل بهینه‌سازی شناخته می‌شوند. مسائلی از قبیل k-مرکز[۱۱]، k-میانه[۱۲] و k-میانگین[۱۳] ازجمله این مسائل خوشه‌بندی می‌باشند که در تعریف تابع هدف تفاوت‌های جزئی با یکدیگر دارند.

مسئله‌ی k-مرکز (که با نام مکان‌یابی شبکه[۱۴] نیز شناخته می‌شود) به دنبال ارائه یک خوشه‌بندی است به‌طوری‌که بیشترین فاصله از هر نقطه تا مرکز منتسب شده به آن، که به شعاع معروف است، کمینه باشد.

---

[۷] Fully Polynomial-time Approximation Scheme

[۸] Cluster

[۹] Center

[۱۰] Objective Function

[۱۱] k-center

[۱۲] k-median

[۱۳] k-means

[۱۴] Network Location Problem

تعریف ۲-۴ (مسئله‌ی k-مرکز) مسئله‌ی k-مرکز: *به ازای گراف وزن‌دار $G = (V, E)$، زیرمجموعه‌ی $S \subseteq V$ با اندازه حداکثر k پیدا کنید طوری‌که عبارت زیر کمینه شود.*

$$\max_{u \in V} \min_{v \in S} d(u, v)$$

به‌طور مشابه، تابع هدف سایر مسائل خوشه‌بندی قابل‌تعریف خواهد بود که تعدادی از مهم‌تری آن‌ها در جدول ۲-۱ آورده شده است. همچنین در بعضی از تعاریف، توابع هدف، یا تابع هزینه[۱۵] خوشه‌بندی، به ازای تمام مسائل خوشه‌بندی به‌صورت کلی و یکپارچه مطرح می‌شود.

تعریف ۲-۵ (تابع هزینه مسئله خوشه‌بندی p) *اگر مجموعه نقاط ورودی مسئله خوشه‌بندی را V بنامیم و زیرمجموعه‌های C_1 تا C_r یک خوشه‌بندی از این نقاط باشد، به‌طوری‌که v_i به مرکز خوشه C_i اشاره کند، تابع هزینه خوشه‌بندی را می‌توان به‌صورت زیر تعریف نمود:*

$$Cost(C_p) = (\sum_{i} \sum_{c \in C_i} d(v, v_i)^p)^{\frac{1}{p}}$$

به ازای $p = 1$، $p = 2$ و $p = \infty$ در تعریف ۲-۵، به توابع هدف مسائل k-میانه، k-میانگین و k-مرکز (مطابق با جدول ۲-۱) خواهیم رسید.

مسئله	تابع هدف
k-مرکز	بیشترین فاصله از هر نقطه تا مرکز منتسب شده به آن
k-میانه	میانگین فاصله نقاط یک خوشه از مرکزش
k-میانگین	مربع میانگین فاصله نقاط یک خوشه از مرکزش

جدول ۲-۱: تابع هدف تعدادی از مسائل خوشه‌بندی

ازآنجاکه تمام مسائل خوشه‌بندی ان‌پی-سخت[۱۶] هستند [۱] و نمی‌توان در زمان چندجمله‌ای پاسخ دقیقی برای آن‌ها به دست آورد، (مگر $P = NP$)، عموماً پژوهشگران به دنبال ارائه الگوریتم‌هایی برای حل تقریبی مسئله بوده‌اند. به‌عنوان نمونه، بهترین الگوریتم تقریبی موجود برای مسئله‌ی k-مرکز یک الگوریتم ۲-تقریب است که در سال ۱۹۸۵ توسط گنزالس [۴] ارائه‌شده است.

[۱۵] Cost Function

[۱۶] NP-hard

مسائل خوشه‌بندی، و به‌ویژه مسئله‌ی k-مرکز ، به شکل‌های مختلف قابل‌تعمیم است. در دسته‌ای از مسائل به ازای هر نقطه از ورودی به‌عنوان یک مرکز، یک هزینه بازگشایی برای آن نقطه در نظر گرفته می‌شود. در این نسخه از مسئله خوشه‌بندی که به «مسئله مکان‌یابی تسهیلات [۱۷]» شناخته می‌شود، علاوه بر شعاع خوشه، ما به دنبال کاهش هزینه‌های بازگشایی مراکز نیز هستیم.

در دسته‌ای دیگر از پژوهش‌ها، هدف کم کردن تعداد مراکز با یک شعاع ثابت است. در این مسائل شعاع خوشه، به‌عنوان ورودی مسئله داده می‌شود و تابع هدف بر روی تعداد مراکز تعریف می‌شود. مصداق عملی این مسئله، قرار دادن تعدادی آنتن با بُرد مشخص برای پوشش تعدادی مشتری است. همچنین بخشی از مطالعات بر روی نسخه‌های «با نقاط دورافتاده[۱۸]» از مسئله خوشه‌بندی متمرکز شده است. در این نسخه‌ها این امکان وجود دارد که تعداد محدودی از نقاط ورودی در جواب نهایی در نظر گرفته نشود و یک جواب (مستقل از وجود چنین نقاطی) برای مسئله به دست آید.

همچنین دسته دیگری از پژوهش‌ها به دنبال اعمال محدودیت بر اندازه خوشه‌ها و ایجاد توازن بین آن‌ها هستند که به مسائل خوشه‌بندی «ظرفیت‌دار[۱۹]» مشهور هستند. در این مسائل، به هر نقطه یک ظرفیت[۲۰] تخصیص داده می‌شود (به‌عنوان بخشی از ورودی مسئله) که نشان می‌دهد یک نقطه درصورتی‌که به‌عنوان مرکز انتخاب گردد تا چه تعداد نقطه را می‌تواند به خود منتسب کند. از دیدگاه ظرفیت نقاط، مسائل ظرفیت‌دار، به دو دسته «با ظرفیت‌های یکسان[۲۱]» و «با ظرفیت‌های غیر یکسان[۲۲]» قابل‌طبقه‌بندی است.

علاوه بر تقسیم‌بندی فوق، مسائل ظرفیت‌دار با تعاریف متعدد دیگری مانند دو مسئله «k-مرکز با ظرفیت‌های نرم[۲۳]» و «k-مرکز با ظرفیت‌های سخت[۲۴]» نیز در پژوهش‌های مختلف موردمطالعه قرارگرفته‌اند. در تعریف مسئله k-**مرکز با ظرفیت‌های سخت**، ظرفیت تمام خوشه‌ها یکسان و برابر L در نظر گرفته می‌شود. در مسئله k-**مرکز با ظرفیت‌های نرم**، در کنار این محدودیت بر روی اندازه

---

[۱۷] Facility Location

[۱۸] Outlier

[۱۹] Capacitated

[۲۰] Capacty

[۲۱] Uniform

[۲۲] Non-uniform

[۲۳] k-center with soft capacities OR Capacitated multi-k-center

[۲۴] k-center with hard capacities / L-Balanced k-center

خوشه‌ها، این آزادی عمل داده می‌شود تا یک نقطه بیش از یک‌بار بتواند به‌عنوان مرکز استفاده شود. در این صورت ما می‌توانیم در هر نوبت از الگوریتم، بدون در نظر گرفتن انتخاب‌های قبلی، هر یک از نقاط ورودی را به‌عنوان یک مرکز جدید انتخاب نماییم.

همچنین در نسخه‌ی خاصی از مسئله‌ی k-مرکز ظرفیت‌دار، به ازای هر نقطه، علاوه بر ظرفیت، یک وزن نیز قائل می‌شویم. در این صورت انتساب یک نقطه به یک مرکز، به‌اندازه وزن آن نقطه از ظرفیت مرکز منتسب شده استفاده خواهد کرد. این مسئله به «k-مرکز وزن‌دار با ظرفیت‌های نرم/سخت[۲۵]» مشهور است.

۲-۳ مجموعه هسته

به ازای مجموعه نقاط P، مجموعه‌ی $C \subset P$ را یک «مجموعه‌ی هسته[۲۶]» می‌خوانیم به شرطی که علاوه بر $|C| < |P|$، محاسبه جواب بهینه بر روی مجموعه C تقریبی برای جواب بهینه بر روی مجموعه اصلی P باشد. یک مجموعه هسته «مجموعه هسته α-تقریب[۲۷]» نامیده می‌شود اگر جواب بهینه بر روی مجموعه هسته، یک جواب α تقریب از جواب بهینه بر روی مجموعه اصلی باشد.

مفهوم مجموعه‌های هسته، به‌عنوان یک شیوه کارا در مدل توزیع‌شده و جریانی[۲۸]، اولین بار توسط آگاروال و سایرین [۱۳] در سال ۲۰۰۴ استفاده شد. این مفهوم به ما کمک می‌کند تا به ازای داده‌های حجیم، مسئله بر روی بخش‌های کوچک (که بر روی یک ماشین قابل‌اجرا است) حل شود، و از اجتماع نتایج برای ارائه یک جواب تقریبی استفاده کنیم. این مفهوم در پژوهش‌های اخیر به‌عنوان یک ابزار ارزشمند برای طراحی الگوریتم‌های تقریبی (و به‌طور خاص به ازای داده‌های حجیم) مورد استقبال قرارگرفته است.

مجموعه‌های هسته به دو دسته «ترکیب‌پذیر[۲۹]» و «تجزیه‌پذیر[۳۰]» قابل تقسیم‌بندی می‌باشند. برای

---

[۲۵] k-center with soft/hard capacities and demands

[۲۶] Coreset

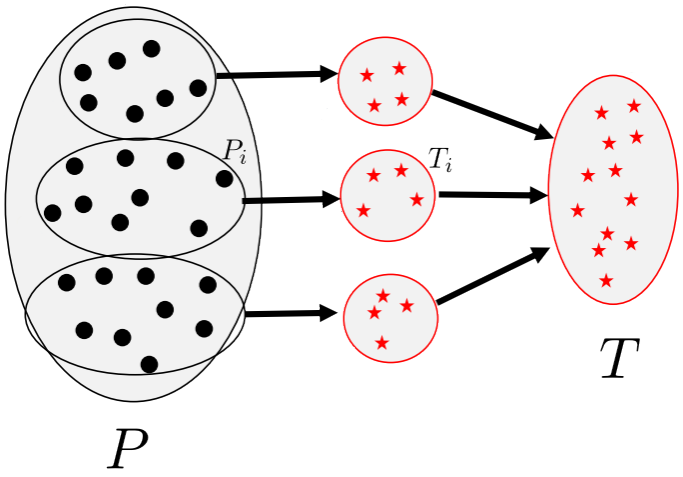
[۲۷] α-coreset

[۲۸] Streaming

[۲۹] Composable

[۳۰] Decomposable

بیان تفاوت این دو، از یک مثال استفاده می‌کنیم. در شکل ۲-۱، مجموعه ورودی P به زیرمجموعه‌های P_1 تا P_r افزار گشته است و T_i مجموعه هسته‌ی α-تقریب برای P_i است. با توجه به تعریف مجموعه‌ی هسته، این بدان معناست که جواب بهینه بر روی مجموعه T_i، یک جواب α تقریب از جواب بهینه بر روی P_i خواهد بود. حال مجموعه T را از اجتماع T_iها را در نظر می‌گیرم. اگر بتوان اثبات کرد که T نیز یک جواب α تقریب برای مجموعه P است، آنگاه مجموعه هسته را تجزیه‌پذیر می‌خوانیم؛ اما اگر T یک جواب با تقریب $\beta \geqslant \alpha$ از جواب بهینه بر روی P باشد، مجموعه هسته را ترکیب‌پذیر می‌خوانیم. طبیعی است که هر مجموعه هسته‌ی تجزیه‌پذیر، حتماً ترکیب‌پذیر نیز است.



شکل ۲-۱: مثالی از مجموعه هسته

همچنین در پژوهش اخیر باطنی و سایرین [۱۲] مفهوم جدیدی بنام «مجموعه‌ی هسته‌ی ترکیب‌پذیر با نگاشت[۳۱]» معرفی‌شده است. در این تعریف، علاوه بر نگهداری نقاط مجموعه‌ی هسته، یک نگاشت از نقاط مجموعه اصلی به نقاط مجموعه هسته نیز نگهداری می‌شود.

[۳۱] Composable Mapping Coreset

فصل ۳

کارهای پیشین

در این فصل به بررسی دقیق مجموعه پژوهش‌هایی که به نحوی با پژوهش جاری در ارتباط هستند پرداخته خواهد شد. با توجه به آنکه مسئله اصلی موردبحث در این پژوهش ارائه یک الگوریتم تقریبی برای حل مسئله‌ی k-مرکز ظرفیت‌دار در مدل توزیع‌شده است، بنابراین دودسته از پژوهش‌ها موردتوجه ما خواهند بود. یک دسته پژوهش‌هایی که به مسائل خوشه‌بندی و بطور خاص مسئله‌ی k-مرکز و نسخه‌های ظرفیت‌دار آن در مدل غیرتوزیع‌شده پرداخته‌اند . دسته دیگر، مطالعات مربوط به مدل‌ها و شیوه‌های توزیع‌شدگی از قبیل مدل نگاشت-کاهش و پژوهش‌هایی که به نحوه‌ی حل توزیع‌شده مسائل خوشه‌بندی توجه داشته‌اند.

۳-۱ مسائل خوشه‌بندی

در این بخش به کارهای انجام‌شده به‌منظور حل مسئله‌ی k-مرکز و نسخه‌های ظرفیت‌دار آن خواهیم پرداخت. در این بخش ابتدا به پژوهش‌های صورت گرفته حول شکل اصلی مسئله اشاره خواهیم داشت و به‌تدریج بر روی پژوهش‌های مرتبط به مسئله موردعلاقه در این پژوهش، تمرکز خواهیم داشت.

مسئله خوشه‌بندی به‌عنوان یکی از پایه‌ای‌ترین مسائل علوم کامپیوتر مورد مطالعات گسترده‌ای قرارگرفته است که در بررسی[۱] [۱۴] مروری بر این مطالعات شده است. ازآنجاکه تمام مسائل خوشه‌بندی ان‌پی-

[۱] Survey

سخت [۲] هستند [۱] و نمی‌توان در زمان چندجمله‌ای پاسخ دقیقی برای آن‌ها به دست آورد، (مگر $P = NP$)، عموماً پژوهشگران به دنبال ارائه الگوریتم‌هایی برای حل تقریبی مسئله بوده‌اند.

یکی از اصلی‌ترین مسائل خوشه‌بندی، مسئله‌ی k-مرکز است. با توجه به ان‌پی-سخت بودن این مسئله، به‌طور خاص برای نسخه اصلی مسئله‌ی k-مرکز به ازای هیچ ϵ ای یک پاسخ تقریبی با ضریب بهتر از $۲ - \epsilon$ وجود ندارد (مگر $P = NP$). تنها برای نسخه اصلی مسئله‌ی k-مرکز ، الگوریتم‌ها گوناگونی ارائه شده است که در آن‌ها از شیوه‌های متفاوتی (مانند برنامه‌ریزی خطی [۳]، شیوه اولیه-دوگان[۴] و جستجوی محلی[۵]) استفاده می‌شود.

گنزالس [۴] بهترین الگوریتم ممکن را برای نسخه اصلی از این مسئله را در سال ۱۹۸۵ ارائه داده است. این الگوریتم کار خود را با انتخاب یک نقطه دلخواه شروع می‌کند و سعی می‌کند تا $k - ۱$ مرحله بعد، نقطه‌ای از میان نقاط باقی‌مانده انتخاب کند که فاصله‌اش از مجموعه نقاط انتخاب‌شده بیشینه باشد. الگوریتم بیان‌شده یک جواب ۲-تقریب ارائه می‌دهد که با توجه به توضیحات پیشین، بهترین جواب ممکن است.

همان‌طور که پیش‌تر نسخه‌های ظرفیت‌دار مسئله‌ی k-مرکز معرفی شد، نسخه‌های مختلفی از مسئله‌ی k-مرکز ظرفیت‌داری وجود دارد. باوجود مطالعات گوناگون برای شناخت نسخه‌های مختلف مسئله‌ی k-مرکز ، الگوریتم‌های محدودی با ضریب تقریب ثابت برای نسخه‌های ظرفیت‌دار این مسئله وجود دارد و تا چند سال اخیر شیوه‌هایی مانند برنامه‌ریزی خطی برای حل این دسته از مسائل وجود نداشته است.

بارلان و سایرین [۱۵] در سال ۱۹۹۳ یک الگوریتم تقریبی برای حل مسئله‌ی k-مرکز ظرفیت‌دار ارائه دادند. این الگوریتم که برای نسخه ظرفیت سخت مسئله است، یک جواب ۱۰-تقریب ارائه می‌دهد. این ضریب تقریب توسط پژوهش خولر و سوسمان [۲] در سال ۲۰۰۰ بهبود یافت. در این پژوهش، ضریب تقریب ۱۰ ارائه‌شده برای مسئله‌ی k-مرکز با ظرفیت‌های سخت، توسط یک الگوریتم ۶-تقریب بهبودیافته است. همچنین در این پژوهش، یک الگوریتم با ضریب تقریب ۵ برای مسئله‌ی k-مرکز با ظرفیت‌های نرم ارائه شده است.

---

[۲] NP-hard

[۳] Linear Programming

[۴] Primal-Dual

[۵] Local Search

به‌صورت سطح بالا، در هردو الگوریتم ارائه‌شده توسط خولر و سوسمان برای نسخه‌های سخت و نرم، عمل انتخاب مرکز خوشه‌ها را در دو گام اصلی صورت می‌پذیرد و این دو الگوریتم تنها در گام دوم با یکدیگر اختلاف دارند. همچنین ایده اصلی استفاده‌شده در گام اول هر دو الگوریتم، به‌کارگیری الگوریتم «شار بیشینه با هزینه کمینه[۶]» است. در الگوریتم مربوط به نسخه ظرفیت سخت مسئله، به دلیل محدودیت در نحوه انتخاب مراکز، نیاز است تا عمل انتخاب مراکز با دقت بیشتری انجام شود که این موضوع نقطه اختلاف دو الگوریتم ارائه شده در این پژوهش است.

همچنین در سال‌های اخیر مطالعاتی بر روی مسئله‌ی k-مرکز با ظرفیت‌های غیر یکسان صورت گرفته است. خولر و سایرین [۵]، برای مسئله‌ی k-مرکز با ظرفیت‌های غیر یکسان، اولین الگوریتم با ضریب تقریب ثابت[۷] را در سال ۲۰۱۲ ارائه کردند. این الگوریتم از شیوه رندسازی برنامه‌ریزی خطی[۸] استفاده می‌کند. همچنین خولر و سایرین در این پژوهش اثبات کرده‌اند که به ازای هیچ $\epsilon > ۰$، الگوریتمی با ضریب تقریب $۳ - \epsilon$ برای مسئله‌ی k-مرکز با ظرفیت‌های غیر یکسان وجود ندارد (مگر $P = NP$).

همچنین در این پژوهش یک الگوریتم با ضریب تقریب ۱۱ برای مسئله‌ی k-مرکز با ظرفیت‌های غیر یکسان برای حالتی که ظرفیت‌ها نرم در نظر گرفته شود ارائه شده است. هونگ چان آن و سایرین [۶] با بهبود چشمگیر پژوهش فوق، موفق شدند تا یک الگوریتم با ضریب تقریب ۹ به کمک شیوه‌های برنامه‌ریزی خطی ارائه دهند. بگفته‌ی مقاله، هدف بعدی این پژوهشگران، ارائه یک الگوریتم به شیوه مشابه برای مسئله مکان‌یابی تسهیلات است که در آن هزینه بازگشایی مراکز نیز باید در نظر گرفته شود.

همچنین در پژوهش جدید سیگان و کویماکا [۱۶] در سال ۲۰۱۴، نسخه‌ی خاصی از مسئله‌ی k-مرکز ظرفیت‌دار (ترکیبی از مسئله‌ی k-مرکز با ظرفیت‌های غیر یکسان و نسخه همراه با نقاط دورافتاده) موردبررسی قرارگرفته است. نتیجه این پژوهش، ارائه اولین الگوریتم تقریبی با ضریب تقریب ۲۵ است. این مسئله با فرض ظرفیت‌های سخت صورت گرفته است و پیش‌بینی می‌شود که با در نظر گرفتن ظرفیت‌ها به‌صورت نرم، ضریب تقریب فعلی بهبود یابد.

---

[۶] Min-cost Max-flow

[۷] حدود چندصد-تقریب که مشخصاً محاسبه نشده است

[۸] LP Rounding

۳-۲ مسئله‌ی k-مرکز در مدل توزیع‌شده

همان‌طور که اشاره شد، به دلیل رشد سریع حجم داده‌های نیازمند پردازش، توجه پژوهشگران به استفاده از مدل‌های توزیع‌شده و چارچوب‌هایی همچون نگاشت-کاهش افزایش یافته است. هرچند در سال‌های اخیر، مسائل خوشه‌بندی به‌طور گسترده‌ای موردمطالعه قرارگرفته‌اند، ولی تعداد کمی از روش‌های ابداعی در این حوزه، قابلیت استفاده برای داده‌های حجیم و مدل‌های توزیع‌شده را دارند.

اکثر الگوریتم‌های ترتیبی که به‌طورمعمول در پژوهش‌های گذشته ارائه‌شده‌اند، متأسفانه با بزرگ شدن حجم داده‌ها نامطلوب ظاهر می‌شوند. یکی از دلایل این اتفاق نبود حافظه کافی است. درواقع طبیعی است که هر ماشین به اندازه محدودی حافظه دارد و درصورتی‌که حجم داده‌ها بیش از این اندازه باشد، ماشین عملاً قادر به اجرای الگوریتم نخواهد بود. همچنین مشکل دوم عدم امکان موازی‌سازی این الگوریتم‌ها است. درواقع الگوریتم‌های موجود، از ابتدا به‌صورت ترتیبی طراحی‌شده‌اند و امکان توزیع داده بین ماشین‌های مختلف را امکان‌پذیر نمی‌سازد.

مدل نگاشت-کاهش [۹] یکی از چارچوب‌های [۱۰] استاندارد محاسبات توزیع‌شده است و می‌تواند به‌عنوان راه‌حل برای مشکلات فوق در نظر گرفته شود. در این مدل، داده‌ها بین تعدادی ماشین توزیع می‌شوند و در هر ماشین محاسبات بر روی بخشی از داده‌ها صورت می‌پذیرد. این عمل در طی چندین دور [۱۱] متوالی صورت می‌پذیرد. در این مدل، امکان تبادل اطلاعات بین ماشین‌ها به شکل محدود در پایان هر دور در نظر گرفته می‌شود.

یک مدل رسمی از مدل نگاشت-کاهش در پژوهش [۷] ارائه گشته است. مهم‌ترین محدودیت در این مدل آن است که تعداد ماشین‌ها و حافظه هر ماشین باید به‌صورت زیرخطی [۱۲] از سایز ورودی باشد که یک فرض طبیعی و قابل‌پذیرش برای داده‌های بزرگ است. همچنین با هدف طراحی هرچه موازی‌تر الگوریتم‌ها در این مدل، هیچ ماشینی در طی محاسبات نباید کل داده‌ها را مشاهده کند. زمان لازم برای هر دور چندجمله‌ای در نظر گرفته می‌شود و درواقع، هدف اصلی در این مدل کاهش تعداد دورهای لازم برای انجام الگوریتم است.

---

[۹] MapReduce

[۱۰] Framework

[۱۱] Round

[۱۲] Sublinear

الگوریتم‌های ترتیبی فعلی برای مسئله‌ی k-مرکز در مدل نگاشت-کاهش به‌طور بهینه قابل‌استفاده نمی‌باشند. به‌عنوان‌مثال الگوریتم گنزالس را می‌توان در نظر گرفت. این الگوریتم در مرحله اول یک نقطه دلخواه انتخاب می‌کند و در مراحل بعد، دورترین نقطه از نقاط انتخاب‌شده تاکنون را انتخاب می‌کند. این الگوریتم در مدل نگاشت-کاهش به k دور نیاز خواهد داشت. چراکه انتخاب هر نقطه، وابسته به انتخاب‌های قبلی است و امکان موازی‌سازی در این الگوریتم مشاهده نمی‌شود.

به‌کارگیری مدل نگاشت-کاهش برای حل مسائل خوشه‌بندی اولین بار در پژوهش آینه و سایرین [۸] موردتوجه قرارگرفته است. در این پژوهش دو الگوریتم تقریبی تصادفی با ضریب تقریب ثابت برای مسئله‌ی k-مرکز و k-میانه ارائه شده است. همچنین در مقایسه انجام‌شده بین روش‌های مختلف خوشه‌بندی در مدل توزیع‌شده، الگوریتم‌های ارائه شده در این پژوهش کارایی مناسبی از خود نشان داده‌اند.

همچنین در کار اخیر ایم و موسلی [۹] در سال ۲۰۱۵، مسئله‌ی k-مرکز و نسخه‌ی با نقاط دورافتاده[۱۳] آن در مدل نگاشت-کاهش موردبازنگری قرارگرفته است. در این پژوهش برای نسخه اصلی مسئله، یک الگوریتم ۲-تقریب با تعداد ۴ دور و برای نسخه با نقاط دورافتاده نیز، با فرض دانستن جواب بهینه، یک الگوریتم ۴-تقریب با تعداد ۳ دور ارائه گشته است.

مفهوم مجموعه‌های هسته‌ی ترکیب‌پذیر به‌عنوان یک شیوه کارا در مدل توزیع‌شده و جریانی[۱۴]، به ما کمک می‌کند تا به ازای داده‌های حجیم، مسئله بر روی بخش‌های کوچک که بر روی یک ماشین قابل‌اجرا است حل شود و از اجتماع نتایج، برای ارائه یک جواب تقریبی استفاده کنیم. این مفهوم به‌طور کامل در پژوهش‌های ضرابی‌زاده و سایرین [۱۰] و ایندیک و سایرین [۱۱] قابل‌مشاهده است. هرچند مسئله‌ی موردبررسی در این دو پژوهش (بیشینه کردن تنوع[۱۵]) متفاوت از پژوهش جاری است، اما به‌عنوان دو نمونه ارزشمند از نحوه به‌کارگیری مفهوم مجموعه‌های هسته‌ی ترکیب‌پذیر قابل‌بررسی هستند.

ایده اصلی در این پژوهش‌ها توزیع داده‌ی ورودی به تعدادی ماشین است که به‌طور مجزا به حل مسئله می‌پردازند. سپس نتایج هر ماشین به‌عنوان مجموعه‌ی هسته بخشی از ورودی با یکدیگر ترکیب می‌شوند و پردازش نهایی بر روی مجموعه ترکیب‌شده صورت می‌پذیرد. ضرابی‌زاده و سایرین با به‌کارگیری از

[۱۳] Outlier

[۱۴] Streaming

[۱۵] Diversity Maximization

ویژگی‌های مجموعه‌های هسته‌ی ترکیب‌پذیر توانستند ضرایب تقریب ارائه‌شده در کار ایندیک و سایرین را به‌صورت چشمگیری بهبود دهند.

باوجود مزایای گسترده‌ی این مفهوم، استفاده از آن برای حل نسخه‌های ظرفیت‌دار مسئله‌ی k-مرکز بسیار محدود بوده است. همان‌طور که بیان شد، یک نمونه از به‌کارگیری از این مفهوم در این دسته از مسائل مربوط به پژوهش نسبتاً جدید باطنی و همکاران [۱۲] است. در این پژوهش چند الگوریتم تقریبی تصادفی با ضرایب تقریب ثابت برای مسائل خوشه‌بندی و خوشه‌بندی ظرفیت‌دار در حالت توزیع‌شده (به کمک تعداد ثابتی دور در مدل نگاشت-کاهش) و در حالت جریانی ارائه شده است که از مفهوم مجموعه‌های هسته‌ی قابل‌ترکیب استفاده‌شده است. در این پژوهش از نوع تعمیم‌یافته‌ای از مجموعه‌های هسته‌ی ترکیب‌پذیر استفاده‌شده است که نه‌تنها زیرمجموعه‌ای به‌عنوان معرف از نقاط اصلی را شامل می‌شود، بلکه نگاشتی از نقاط مبدأ به نقاط انتخابی را نیز به همراه دارد. در این پژوهش به‌طور خاص برای مسئله‌ی k-مرکز با ظرفیت‌های سخت، یک الگوریتم (۳۲α)-تقریب تصادفی در تعداد دور ثابتی ارائه شده است که α به ضریب تقریب حل نسخه اصلی مسئله‌ی k-مرکز اشاره دارد. درنتیجه، الگوریتم ارائه شده در این پژوهش یک جواب ۶۴-تقریب ارائه می‌دهد. الگوریتم ارائه شده در پژوهش جاری نه تنها این ضریب تقریب را بهبود می‌دهد، بلکه الگوریتمی غیرتصادفی و با تنها ۱ دور (در مدل نگاشت-کاهش) می‌باشد.

فصل ۴

الگوریتم تقریبی جدید برای k-مرکز وزن‌دار

در این فصل قصد داریم تا یک الگوریتم جدید برای حالت کلی‌تر مسئله‌ی k-مرکز با ظرفیت‌های نرم که به نام **k-مرکز وزن‌دار با ظرفیت‌های نرم** شناخته می‌شود ارائه دهیم. در این نسخه از مسئله برای هر نقطه یک وزن و یا یک میزان درخواست[۱] در نظر گرفته می‌شود. میزان درخواست یک نقطه می‌تواند توسط یک یا چند مرکز تأمین گردد. زمانی که یک نقطه به یک مرکز انتساب پیدا می‌کند، به‌اندازه وزن/درخواست آن نقطه از ظرفیت مرکز استفاده می‌شود. ارائه این الگوریتم به ما کمک می‌کند تا در فصل بعد چند الگوریتم تقریبی با ضریب تقریب ثابت برای مسئله k-مرکز با ظرفیت های نرم و سخت طراحی نماییم. شایان ذکر است که نتایج حاصل در این فصل و فصل بعد با همکاری جناب مهندس کیان میرجلالی بدست آمده است.

الگوریتم ارائه‌شده در این فصل بر مبنای الگوریتم خولر و سوسمان [۲] شکل گرفته است. الگوریتم مذکور، یک الگوریتم ۵-تقریب برای نسخه غیر وزن‌دار مسئله k-مرکز با ظرفیت‌های نرم می‌باشد، که ما در این بخش آن را به نسخه‌ی وزن‌دار ارتقا می‌دهیم. به‌منظور اجتناب از بیان بخش‌های تغییر نکرده‌ی الگوریتم ، بعضاً به قسمت‌هایی از آن ارجاع داده خواهد شد. الگوریتم نهایی یک جواب ۵-تقریب برای مسئله k-مرکز وزن‌دار با ظرفیت‌های نرم خواهد بود.

---

[۱] Request

۴-۱ تعاریف و مفاهیم اولیه

برای تعریف دقیق مسئله لازم است ابتدا چند تعریف پایه‌ای ارائه دهیم. همان‌طور که پیش‌تر بیان شد، در نسخهٔ وزن‌دار مسئله، هر رأس u یک میزان درخواست r_u دارد که به‌عنوان ورودی مسئله داده می‌شود. مجموع تمام درخواست رئوس را با R_{total} نمایش می‌دهیم و به‌صورت زیر تعریف می‌شود:

$$R_{total} = \sum_{v \in V} r_v$$

در نسخه وزن‌دار از مسئله، میزان درخواست هر رأس می‌تواند توسط چندین مرکز تأمین شود. مجموعه مراکزی که رأس u از آن‌ها سرویس دریافت می‌کند و به آن‌ها منتسب شده است را با نماد $\phi(u)$ نمایش می‌دهیم.

مجموعه رئوسی که از یک مرکز سرویس گیری می‌کنند و میزان این سرویس گیری، دامنه [۲] یک مرکز را تعریف می‌کند. دامنه یک مرکز مانند c درواقع از دو جز تشکیل می‌شود: (۱) مجموعه رئوسی که بخشی از درخواست خود را توسط این مرکز تأمین می‌کند. این مجموعه را با نماد $\mathcal{D}_c$ نمایش می‌دهیم. (۲) میزان درخواستی از هر عضو مجموعه فوق که توسط مرکز تأمین می‌گردد، که با نماد F_c و به‌صورت تابع زیر تعریف می‌شود:

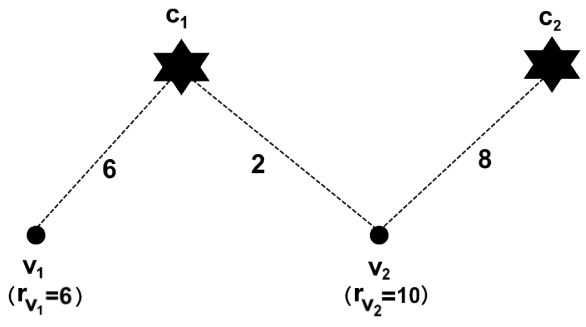
$$F_c : \mathcal{D}_c \longrightarrow \mathbb{Z}$$

همچنین به ازای هر مرکز m، «**اندازه دامنه** m» را با نماد R_m و برابر با میزان درخواست‌های پوشش داده‌شده توسط m تعریف می‌کنیم و داریم:

$$R_m = \sum_{v \in} F_m(v)$$

برای واضح شدن نمادهای معرفی‌شده، به مثال شکل ۴-۱ که در آن دو رأس $v_۱$ و $v_۲$ از دو مرکز $c_۱$ و $c_۲$ سرویس می‌گیرند توجه کنید.

[۲] Domain



شکل ۴-۱: یک مثال برای معرفی نمادهای معرفی‌شده در این بخش

$R_{total} = 16$ $R_{c_1} = 8$ $R_{c_2} = 8$

$\mathcal{D}_{c_1} = \{v_1, v_2\}$ $F_{c_1} = \{v_1 \to 6, v_2 \to 2\}$

$\mathcal{D}_{c_2} = \{v_2\}$ $F_{c_2} = \{v_2 \to 8\}$

$\phi(v_1) = \{c_1\}$ $\phi(v_2) = \{c_1, c_2\}$

همچنین از پیش می‌دانستیم که فاصله یک نقطه p از یک مجموعه نقاط S به‌صورت فاصله نقطه p از نزدیک‌ترین نقطه مجموعه S تعریف و با نماد $d(p, S)$ نمایش داده می‌شود. با توجه به تعاریف فوق، به بیان تعریف رسمی مسئله می‌پردازیم.

مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم: به ازای گراف ورودی $G = (V, E)$ که طول یال e_i با $w(e_i)$ نمایش داده‌شده است و میزان درخواست (یا وزن) هر رأس مانند u به شکل r_u مشخص‌شده است، یک زیرمجموعه $S \subseteq V$ با اندازه حداکثر k و نحوه انتساب سایر رئوس به این زیرمجموعه پیدا کنید که حداکثر فاصله هر رأس از دورترین مرکز منتسب به آن کمینه باشد. به‌عبارت‌دیگر:

$$\min_{S \subseteq V} \max_{\substack{u \in V \\ p \in \phi(u)}} d(u, p)$$

به‌طوری‌که:

$$\sum_{m \in \phi(u)} F_m(u) \leqslant r_u \qquad \forall u \in V$$

$$\sum_{u \in \mathcal{D}_v} F_v(u) \leqslant L \qquad \forall v \in S$$

۴-۲ الگوریتم پیشنهادی

در این بخش به بیان الگوریتم پیشنهادی خواهیم پرداخت. درستی الگوریتم در بخش بعد موردبررسی خواهد گرفت. همان‌طور که پیش‌تر بیان شد، برای اجتناب از تکرار تا جای ممکن در بخش‌هایی به الگوریتم اصلی ارجاع خواهیم نمود. همچنین در ادامه فرض می‌شود که گراف ورودی یک گراف کامل است؛ چراکه به ازای هر یالی که وجود نداشته باشد، می‌توان کوتاه‌ترین مسیر [۳] بین آن دو رأس را در نظر گرفت و الگوریتم همچنان به‌درستی عمل خواهد کرد.

در بالاترین سطح از الگوریتم، قطعه کد CAPACITED-CENTERS$(G = (V, E), k, L)$، ابتدا یال‌ها را بر اساس وزن به‌صورت غیر نزولی مرتب می‌کند. یال‌های مرتب‌شده را $e_۱, e_۲, ..., e_n$ در نظر بگیرید. به ازای هر i، گراف G_i را یک زیرگراف بی‌وزن از G شامل یال‌های با حداکثر طول $w(e_i)$ در نظر بگیرید. ازآنجاکه در تمام طول الگوریتم گراف G_i بی‌وزن در نظر گرفته می‌شود، میدانیم که فاصله هر دو رأس در آن برابر با «تعداد» یال‌های موجود در کوتاه‌ترین مسیر بین آن دو رأس خواهد بود. به ازای هر گراف G_i (با شروع i از ۱ تا n)، و تا زمانی که به جواب برسیم، الگوریتم ASSIGNCENTERS(G_i) را بر روی G_i اجرا می‌کنیم. در پایان هر مرحله تعدادی مرکز به‌عنوان جواب پیدا خواهد شد که دو حالت خواهد بود:

- **اگر تعداد این مرکزها از k بیشتر باشد**، ما اثبات خواهیم کرد که هیچ راه‌حلی با حداکثر هزینه $w(e_i)$ وجود ندارد. زیرا طبق قضیه ۴-۵ و ۴-۴ الگوریتم ما در بدترین حالت تنها به‌اندازه حداقل تعداد مراکز لازم در جواب بهینه، مرکز ایجاد می‌کند. پس اگر الگوریتم ما بیش از k مرکز را برای پوشش درخواست‌ها استفاده کند، این بدان معناست که هر جواب بهینه‌ای نیز به بیش از این تعداد مرکز نیاز خواهد داشت و درنتیجه جوابی وجود ندارد.
- **اگر تعداد مراکز به‌دست‌آمده حداکثر k باشد**، نشان خواهیم داد که در G_i، فاصله هر رأس تا دورترین مرکزی که به آن منتسب شده حداکثر ۵ خواهد بود که برابر با یک جواب ۵-تقریب برای مسئله خواهد بود. این حالت در قضیه ۴-۵ اثبات خواهد شد.

[۳] Shortest Path

الگوریتم ۱ $Capacitated-Centers(G(V,E),k,L)$

۱: تمام یال‌ها را به‌صورت غیر نزولی مرتب کن $(e_۱, e_۲, ..., e_n)$
۲: به ازای $i = ۰$ تا n انجام بده:
۳: ساخت گراف $G_i = (V, E_i)$ به‌طوری‌که $E_i = \{e_۱, ..., e_i\}$
۴: پایان الگوریتم اگر AssignCenters$(G_i) = true$

الگوریتم ۲ $AssignCenters(G_i)$

۱: قرار بده $SUCCESSFUL = true$
۲: فرض کنیم $R_i^c = \sum_{v \in c} r_v$ مجموع میزان درخواست‌های مولفه‌ی همبندی G_i^c باشد
۳: اگر $\sum_{c \in G_i} \left\lceil \frac{R_i^c}{L} \right\rceil > k$:
۴: $SUCCESSFUL = false$
۵: در غیر این صورت:
۶: به ازای هر مولفه‌ی همبندی G_i^c در G_i:
۷: فراخوانی AssignMonarchs(G_i^c)
۸: فراخوانی AssignDomains(G_i^c)
۹: فراخوانی ReAssign(G_i^c)
۱۰: اگر مجموع تعداد مراکز ایجادشده بزرگ‌تر از k :
۱۱: $SUCCESSFUL = false$
۱۲: $SUCCESSFUL$ را برگردان

الگوریتم AssignCenters(G_i)[۴] وظیفه مشخص کردن مراکز در گراف ورودی G_i را بر عهده دارد. در این الگوریتم هر مؤلفه‌ی همبندی به‌صورت مجزا مورد پردازش قرار می‌گیرد. این کار ازآن‌جهت درست است که شعاع هر جواب بهینه (در صورت وجود چنین جوابی) حداکثر $w(e_i)$ خواهد بود و درنتیجه هیچ رأسی از یک مؤلفه‌ی همبندی به یک رأس از مؤلفه‌ی همبندی دیگر منتسب نخواهد شد و درنتیجه می‌توان مؤلفه‌ی همبندی را مجزا پردازش نمود.

[۴] با اندکی تغییر نسبت به الگوریتم همتای خود در پژوهش [۲]

اگر به ازای هر مؤلفه همبندی c از گراف G_i، داشته باشیم $R_i^c = \sum_{v \in c} r_v$، یک حد پایین برای تعداد مراکز لازم در این مؤلفه‌ی همبندی $\left\lceil \frac{R_i^c}{L} \right\rceil$ خواهد بود. درنتیجه اگر $\sum_{c \in G_i} \left\lceil \frac{R_i^c}{L} \right\rceil$ از حداکثر تعداد مرکز مجاز (k) بیشتر باشد، میدانیم که هیچ جوابی برای گراف G با حداکثر اندازه $w(e_i)$ وجود نخواهد داشت. در غیر این صورت، به ازای هر مؤلفه‌ی همبندی مرکزها به صورتی که در ادامه بیان می‌شود، به دست می‌آید.

در ادامه‌ی این الگوریتم و توسط فراخوانی الگوریتم AssignMonarchs(G_i) (رجوع به پژوهش [۲])، به ازای هر مولفه‌ی همبندی مثل G_i^c، یک مجموعه مستقل[۵] به نام M به دست می‌آید. اعضای این مجموعه مستقل که در اصطلاح پژوهش اصلی **پادشاه**[۶] نامیده می‌شوند، یک درخت (رابطه پدر-پسر) را تشکیل می‌دهند. هر یک از سایر رئوس مؤلفه همبندی نیز به یک پادشاه منتسب می‌گردند و در اصطلاح امپراتوری پادشاه را شکل می‌دهند. به ازای هر پادشاه مانند m، پدر پادشاه در درخت پادشاه‌ها با نماد $p(m)$ نمایش داده می‌شود. همچنین مجموعه رئوسی که طبق الگوریتم به پادشاه m منتسب شده‌اند و امپراتوری آن را تشکیل داده‌اند، با نماد $E(m)$ نمایش داده می‌شوند. همچنین برای سهولت اشاره به پادشاه صاحب یک امپراتوری، نماد $\mathcal{M}$ را به‌صورت زیر تعریف می‌کنیم.

$$\forall v \in E(m) : \mathcal{M}(v) = m$$

با توجه به الگوریتم مذکور، چهار ویژگی مهم زیر در رابطه با مجموعه پادشاه‌ها و امپراتوری‌ها وجود دارد:

۱. فاصله هر دو پادشاه از یکدیگر حداقل سه است.

۲. فاصله هر پادشاه m (به‌جز پادشاه ریشه درخت) تا «پادشاه پدرش» دقیقاً برابر سه است.

۳. فاصله هر پادشاه از رئوس موجود در امپراتوری خودش حداکثر دو است. رئوس در فاصله یک از پادشاه را «**امپراتوری سطح اول** m» یا $E۱(m)$ و رئوس در فاصله دو را «**امپراتوری سطح دوم** m» یا $E۲(m)$ می‌خوانیم و داریم: $E(m) = E۱(m) \cup E۲(m)$.

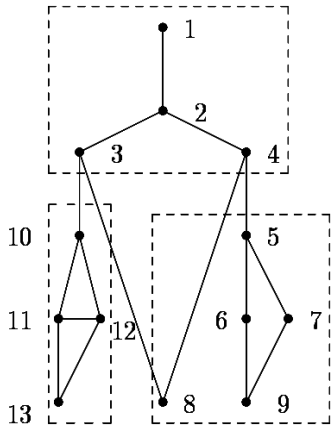
۴. هر پادشاه m (به‌جز پادشاه ریشه درخت) حداقل یک یال ارتباطی با یک رأس قرارگرفته در $E۲(p(m))$ دارد. همچنین با توجه به ویژگی ۱، هر چنین رأس سطح دویی، تنها یک پادشاه در مجاورت خود دارد.

---

[۵] Independent Set

[۶] Monarch

شکل ۴-۲ یک نمونه از اجرای الگوریتم AssignMonarchs(G_i) بر روی یک گراف با یک مولفه‌ی همبندی و ۱۳ رأس را نمایش می‌دهد. در اینجا رئوس ۱ و ۵ و ۱۰ به‌عنوان پادشاه‌ها انتخاب‌شده‌اند و ناحیه نقطه‌چین، امپراتوری هر یک از این پادشاه‌ها را مشخص کرده است و داریم:



شکل ۴-۲: مثالی از مجموعه پادشاه‌ها و امپراتوری

$$M = \{1, 5, 10\}$$

$$p(5) = p(10) = 1 \quad \& \quad p(1) = \emptyset$$

$$\mathcal{M}(2) = \mathcal{M}(3) = \mathcal{M}(4) = 1$$

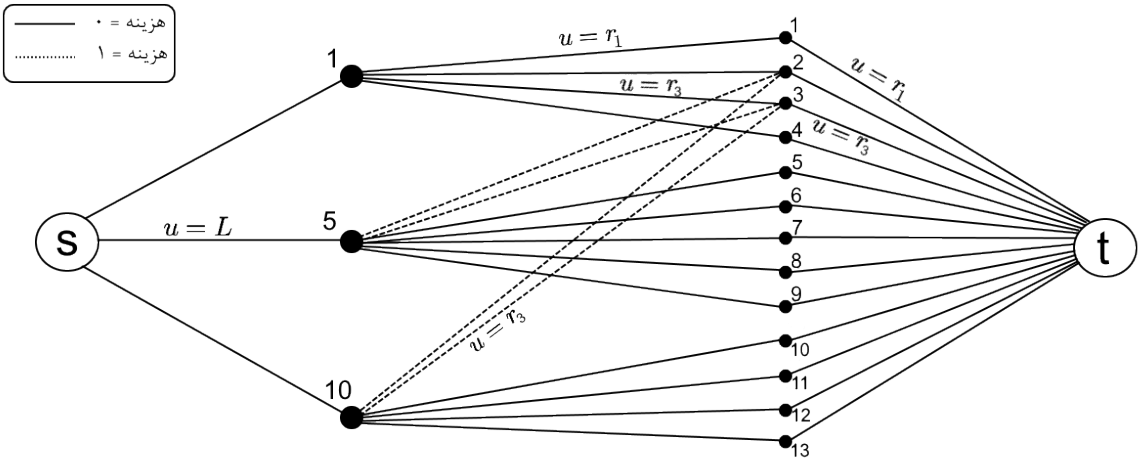
پس از به دست آمدن مجموعه پادشاه‌های M، الگوریتم AssignDomains(G_i^c)[۷] اجرا می‌گردد. این الگوریتم تلاش می‌کند تا یک دامنه به‌اندازه حداکثر L به هر پادشاه انتساب بدهد. ایده اصلی استفاده‌شده در این الگوریتم، استفاده از الگوریتم «بیشترین شار با هزینه کمینه» بر روی یک گراف دوبخشی $G'(M, V, E')$ است. به ازای گراف ورودی شکل ۴-۲ این گراف دوبخشی به‌صورت شکل ۴-۳ خواهد بود. هدف نهایی این الگوریتم، انتساب حداکثری درخواست رئوس به پادشاه‌ها است، به‌طوری‌که سه محدودیت زیر رعایت گردد:

۱. هر پادشاه در دامنه خود بیش از L درخواست را پوشش ندهد.

۲. یک رأس تنها به پادشاه‌هایی انتساب پیدا کند که حداکثر در فاصله دو از یکدیگر باشند.

[۷] با اندکی تغییر نسبت به الگوریتم همتای خود در پژوهش [۲]

۳. یک پادشاه تنها زمانی به درخواست یک رأس از خارج امپراتوری خود سرویس دهد که هیچ درخواست بی‌پاسخی در رئوس امپراتوری خود نداشته باشد.



شکل ۴-۳: مثالی از گراف دوبخشی G'

برای تطبیق این الگوریتم با نسخه وزن‌دار، لازم است تا تغییراتی در بدنه الگوریتم اصلی ایجاد گردد و تعدادی از تعاریف موجود گسترش یابند. ازآنجاکه در نسخه وزن‌دار، هر رأس یک میزان درخواست مشخص (r_v) دارد، نیاز است تا ظرفیت (u) یال‌های گراف دوبخشی G' به‌صورت زیر اصلاح شود:

$$\forall m \in M, v \in V: \qquad u(s,m) = L, \qquad u(m,v) = r_v, \qquad u(v,t) = r_v$$

واضح است که خروجی این الگوریتم، یک دامنه به ازای هر پادشاه خواهد بود. همان‌طور که پیش‌تر نیز بیان شد، تعریف دامنه یک مرکز در نسخه وزن‌دار اندکی متفاوت است و از دو جز (یک مجموعه رئوس و یک تابع) تشکیل می‌شود.

پس از اجرای الگوریتم $\textsc{AssignDomains}(G_i^c)$، ممکن است بخشی از درخواست‌های یک رأس توسط هیچ پادشاهی پوشش داده نشده باشد. به همین دلیل، الگوریتم $\textsc{ReAssign}(G_i^c)$[۸] سعی می‌کند تا با پیمایش درخت پادشاه‌ها، از سمت برگ‌ها به ریشه، تعدادی مرکز جدید برای پوشش این بخش از درخواست‌ها بر روی پادشاه‌های فعلی ایجاد نماید (این کار به دلیل نرم بودن ظرفیت‌ها امکان‌پذیر است). این الگوریتم درخواست‌های مربوط به یک رأس u را به $\mathcal{M}(u)$ یا $p(\mathcal{M}(u))$ منسوب می‌کند.

[۸] با اندکی تغییر نسبت به الگوریتم همتای خود در پژوهش [۲]

الگوریتم ۳ $AssignDomains(G_i^c)$

۱: مجموعه M را مجموعه پادشاه‌های G_i^c در نظر بگیر.

۲: قرار بده $E' = \{(m,v)|m \in M, v \in V, d(m,v) \leqslant ۲\}$.

۳: گراف دوبخشی $G' = M, V, E'$ را بساز.

۴: دو رأس s و t را اضافه کن.

۵: یال‌های $\{(s,m)|m \in M\}$ و $\{(v,t)|v \in V\}$ را اضافه کن.

۶: به ازای هر $m \in M, v \in V$ ظرفیت یال‌ها را قرار بده: $u(s,m) = L$، $u(m,v) = r_v$ و $u(v,t) = r_v$.

۷: هزینه هر یال $c(m,v)$ که v در امپراتوری m نیست برابر ۱ و هزینه سایر یال‌ها برابر ۰ باشد.

۸: الگوریتم «بیشینه شار با هزینه کمینه» را بر بروی G' اجرا کن.

۹: به ازای هر پادشاه $m \in M$:

۱۰: قرار بده $\mathcal{D}_m = \{v|$رأس v دست‌کم یک واحد شار از m دریافت کرده است$\}$

۱۱: قرار بده $F_m(v) = \{$میزان شاری که رأس v از m دریافت کرده است$\}$ $\quad \forall v \in \mathcal{D}_m$:

۱۲: به ازای هر رأس $v \in V$:

۱۳: قرار بده $\phi(v) = \{m|v \in \mathcal{D}_m\}$

الگوریتم ۴ $ReAssign(G_i^c)$

۱: مجموعه M را مجموعه پادشاه‌های G_i^c در نظر بگیر.

۲: به ازای هر پادشاه $m \in M$:

۳: قرار بده: $Unsatisfied(m) = \left\{ v \in \{m\} \cup E(m) | r_v > \sum_{m \in \phi(v)} F_m(v) \right\}$

۴: قرار بده: $U(m) = \sum_{v \in Unsatisfied(m)} (r_v - \sum_{m \in \phi(v)} F_m(v))$

۵: قراربده: $Passed(m) = \emptyset$

۶: قراربده: $P(m) = ۰$

۷: Tرا درخت پادشاه‌های G_i^c قرار بده.

۸: تا وقتی درخت T خالی نشده است:

۹: برگ m از T را در نظر بگیر.

۱۰: مرکز واقع‌شده بر روی پادشاه m را $c_۰$ بنام. (در الگوریتم $AssignDomains$)

۱۱: مقدار k' و ϵ را از رابطه زیر به دست بیاور:

$$U(m) + P(m) = k'.L + \epsilon$$

۱۲: تعداد k' مرکز **جدید** (بنام‌های $c_۱, ..., c_{k'}$) بر روی رأس m ایجاد کن و $k'.L$ واحد از درخواست‌ها را به آن‌ها منتسب کن. به ازای هر رأس v که بخشی از درخواستش توسط این مراکز جدید پوشش داده می‌شود، رأس m را به مجموعه $\phi(v)$ اضافه کن.

۱۳: مقدار ϵ واحد درخواست باقیمانده را $c_۰$ منتسب کن. (ϕ مجدداً به‌روزرسانی شود)

۱۴: میزان ϵ واحد از درخواست‌هایی که مرکز $c_۰$ توسط الگوریتم $AssignDomains$ پوشش داده بوده را به $Passed(p(m))$ اضافه کن.

۱۵: برگ m از T را پاک کن.

۴-۳ اثبات درستی الگوریتم

به‌سادگی می‌توان دید که پیچیدگی الگوریتم ما از مرتبه چندجمله‌ای است. حال در این بخش به اثبات درستی الگوریتم و محاسبه ضریب تقریب آن می‌پردازیم. بدین منظور ابتدا با توجه به الگوریتم $\text{AssignDomains}(G_i^c)$ تعدادی تعریف اولیه ارائه می‌دهیم. با توجه به این تعاریف به یک افراز بندی از مجموعه پادشاه‌ها خواهیم رسید. در ادامه تعدادی ویژگی ارائه خواهد شد و نهایتاً در قضیه ۴-۵ به اثبات نهایی درستی و ضریب تقریب الگوریتم پرداخته خواهد شد.

تعریف ۴-۱ (شار ورودی و خروجی به یک رأس) *با توجه به اجرای الگوریتم شار بیشینه با هزینه کمینه بر روی گراف دوبخشی G' داریم:*

$$\forall v \in M \cup V : f^-(v) = \{\text{مجموع شار واردشده به رأس } v \text{ پس از اجرای الگوریتم}\}$$

$$\forall v \in M \cup V : f^+(v) = \{\text{مجموع شار خارج‌شده به رأس } v \text{ پس از اجرای الگوریتم}\}$$

تعریف ۴-۲ (رأس پوشش داده نشده) *رأس v را پوشش داده نشده گوییم اگر $f^+(v) < r_v$.*

تعریف ۴-۳ *با توجه شار ورودی (f^-) و خروجی (f^+) رئوس گراف دوبخشی G'، پادشاه‌ها را به* ***سه دسته*** *زیر می‌توان افراز نمود.*

۱. *پادشاه m را* ***سبک (Light)*** *گوییم اگر اندازه دامنه (R_m) آن کمتر از L باشد، یا به‌عبارت‌دیگر:*

$$f^-(m) < L$$

۲. *پادشاه m را* ***سنگین (Heavy)*** *گوییم اگر یک رأس پوشش داده نشده در امپراتوری m وجود داشته باشد، یا به‌عبارت‌دیگر:*

$$f^-(m) = L \qquad \text{و} \qquad f^+(E(m)) < \sum_{v \in E(m)} r_v$$

۳. *پادشاه m را* ***تکمیل (Full)*** *گوییم اگر نه سبک و نه سنگین باشد، یا به‌عبارت‌دیگر:*

$$f^-(m) = L \qquad \text{و} \qquad f^+(E(m)) = \sum_{v \in E(m)} r_v$$

*همچنین مجموعه تمام پادشاه‌های سبک، سنگین و تکمیل را به ترتیب با نمادهای **LM** و **HM** و **FM** معرفی می‌کنیم.*

تعریف ۴-۴ *تعداد کل پادشاه‌های سبک را K_{LM} می‌نامیم. همچنین مجموع درخواست‌های موجود در دامنه پادشاه‌های سبک را با نماد R_{LM} معرفی و به‌صورت زیر تعریف می‌کنیم:*

$$R_{LM} = \sum_{l \in LM} R_l$$

لم ۴-۱ *اگر m یک پادشاه سنگین باشد، تمام درخواست‌های دامنه m مربوط به امپراتوری m است، یا به‌عبارت‌دیگر:*

$$\forall u \in \mathcal{D}_m \implies u \in E(m)$$

اثبات. فرض کنید رأس u در دامنه m قرار داشته باشد ($F_m(v) > ۱$)، درحالی‌که در امپراتوری m قرار نداشته باشد. حال فرض کنید یک رأس دیگر مانند x در امپراتوری m وجود داشته باشد که پوشش داده نشده باشد.

در این صورت ما می‌توانیم نحوه توزیع شار را به این صورت تغییر دهیم که یک واحد شار عبوری از m که به u واردشده را به x بدهیم. میزان کل شار جدید برابر با کل شار قبلی است ولی با هزینه کمتر خواهد بود که این در تناقض با الگوریتم «شار بیشینه با هزینه کمینه» است. □

مجموعه اولیه $\mathcal{E}_۰$ را برابر LM (مجموعه پادشاه‌های سبک) در نظر بگیرید. در مرحله jام، پادشاهی را که یک رأس مانند v در دامنه خود دارد، و با توجه به گراف دوبخشی G' امکان پوشش داده شدن v توسط یکی از پادشاه‌های موجود در مجموعه $\mathcal{E}_{j-۱}$ نیز وجود داشته است، به مجموعه اضافه کنید. به‌عبارت‌دیگر:

$$\mathcal{E}_j = \mathcal{E}_j \bigcup \{m \in M | \exists v \in V, \exists m' \in \mathcal{E}_{i-1}, m \in \phi(v) \quad and \quad d(v, m') \leqslant 2 \quad \text{in G'}\}$$

$\mathcal{E}$ را بزرگ‌ترین مجموعه $\mathcal{E}_j$ به‌دست‌آمده در فرآیند فوق بنامیم. همچنین $\mathcal{F}$ را سایر پادشاه‌های قرار نگرفته در این مجموعه می‌نامیم ($\mathcal{F} = M - \mathcal{E}$).

لم ۴-۲ *مجموعه $\mathcal{E}$ شامل هیچ پادشاه سنگینی نیست.*

اثبات. اثبات مشابه با اثبات ارائه‌شده در پژوهش [۲] است. فرض کنیم که پادشاه سنگین θ در مرحله jام به‌واسطه رأس مشترک v به $\mathcal{E}_{j-1}$ اضافه‌شده باشد. ما می‌توانیم یک واحد درخواست رأس v را از θ آزادکرده و به یک پادشاه θ' منتقل کنیم. با تکرار یک دنباله از این عمل به $\mathcal{E}$، خواهیم رسید که میدانیم ظرفیت خالی برای یک واحد منتقل‌شده در این سلسله جابجایی‌ها را حتماً دارد. با این کار ما پادشاه سنگین θ را برای سرویس دادن به یک واحد درخواست جدید به رئوس امپراتوری θ که پوشش کامل داده نشده است آماده می‌کنیم و شار کل را افزایش می‌دهیم که در تناقض با الگوریتم «شار بیشینه با هزینه کمینه» است. □

لم ۴-۳ *یک مرکز مانند θ در جواب بهینه را در نظر بگیرید که بخشی از درخواست‌های یک پادشاه مانند $e \in \mathcal{E}$ را پوشش می‌دهد. هیچ رأسی که در دامنه پادشاه‌های موجود در $\mathcal{E}$ قرار نداشته باشد، نمی‌تواند در جواب بهینه نیز توسط θ مورد پوشش قرار بگیرد و در دامنه θ باشد.*

اثبات. فرض کنیم که مرکز بهینه θ دو رأس $e \in \mathcal{E}$ و u را در دامنه خود پوشش می‌دهد. به‌عنوان فرض خلف، فرض می‌کنیم که u در دامنه هیچ پادشاهی از $\mathcal{E}$ نباشد، یا به‌عبارت‌دیگر u یکی از شرایط زیر را داشته باشد:

۱. رأس u کلاً در دامنه هیچ پادشاهی نباشد:
ازآنجاکه میدانیم فاصله θ تا هردو رأس مذکور حداکثر $w(e_i)$ است، پس متوجه می‌شویم که در گراف بی‌وزن G_i داریم: $d(e,u) \leqslant ۲$. مشابه به اثبات لم ۴-۲، می‌توان با دنباله‌ای از جابجایی‌ها، یک واحد درخواست از e آزاد نمود و به یک پادشاه در $\mathcal{E}$منتقل نمود. بنا بر لم ۴-۲، پادشاه نهایی ظرفیت خالی دارد و این سلسله جابجایی‌ها ممکن است. درنتیجه می‌توان شار را به‌گونه‌ای تغییر داد که u توسط e پوشش داده شود که در تناقض با فرض اولیه است.

۲. رأس u در دامنه پادشاه $f \in \mathcal{F}$ باشد:
به‌طور مشابه میدانیم که $d(e,u) \leqslant ۲$. با توجه به تعریف $\mathcal{E}$ میدانیم که پادشاه f می‌تواند عضو $\mathcal{E}$ شود که در تناقض با فرض اولیه است.

□

قضیه‌ی ۴-۴ *هر جواب برای مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم به ازای گراف G_i به حداقل $K_{LM} + \left\lceil \frac{R-R_{LM}}{L} \right\rceil$ مرکز نیاز خواهد داشت.*

اثبات. فرض کنیم که مجموعه درخواست‌های دامنه پادشاه‌های $\mathcal{E}$ را $R_{\mathcal{E}} = \sum_{m \in \mathcal{E}} R_m$ بنامیم. در جواب بهینه، هر پادشاه از $\mathcal{E}$ باید توسط یک (یا چند) مرکز متمایز پوشش داده شوند[۹] و بنا بر لم ۴-۳ می‌دانیم که این مراکز جواب بهینه هیچ پادشاه دیگری از $\mathcal{F}$ را نیز پوشش نمی‌دهند. درنتیجه، هر جوابی حداقل به $|\mathcal{E}| + \left\lceil \frac{R-R_{\mathcal{E}}}{L} \right\rceil$ مرکز نیاز دارد.
همچنین داریم:

$$|\mathcal{E}| = K_{LM} + \frac{|\mathcal{E}| - K_{LM}}{L}.L \qquad (۴-۱)$$

$$R_{\mathcal{E}} = R_{LM} + (|\mathcal{E}| - K_{LM}).L \qquad (۴-۲)$$

پس داریم:

$$|\mathcal{E}| + \left\lceil \frac{R - R_{\mathcal{E}}}{L} \right\rceil = K_{LM} + \left\lceil \frac{R + (|\mathcal{E}| - K_{LM}).L - R_{\mathcal{E}}}{L} \right\rceil \qquad \text{(طبق ۴-۱)}$$

$$= K_{LM} + \left\lceil \frac{R - R_{LM}}{L} \right\rceil \qquad \text{(طبق ۴-۲)}$$

درنتیجه هر جواب به حداقل $K_{LM} + \left\lceil \frac{R-R_{LM}}{L} \right\rceil$ مرکز نیاز خواهد داشت. □

قضیه‌ی ۴-۵ *در خروجی الگوریتم پیشنهادی ما حداکثر فاصله هر رأس از دورترین مرکزی که از آن سرویس می‌گیرد ۵ است. همچنین حداکثر $K_{LM} + \left\lceil \frac{R-R_{LM}}{L} \right\rceil$ مرکز استفاده می‌شود.*

اثبات. **بخش اول:**
با توجه به الگوریتم AssignDomains(G_i^c) میدانیم که بخشی از درخواست رئوس توسط پادشاهی که از آن شار دریافت می‌کند پوشش داده‌شده است. همچنین به ازای آن دسته از رئوسی که به‌صورت ناقص

[۹] درواقع مرکزی در جواب بهینه وجود ندارد که هم‌زمان چند پادشاه موجود در $\mathcal{E}$ را پوشش بدهد. چراکه در این صورت درواقع فاصله آن دو (یا چند) مرکز از هم دو است و طبق الگوریتم AssignMonarchs(G_i) عملاً یکی از آن‌ها نمی‌تواند اصلا پادشاه باشد.

پوشش داده‌شده‌اند (طبیعتاً متعلق به پادشاه‌های سنگین)، در الگوریتم $\text{ReAssign}(G_i^c)$ این بخش از درخواست‌ها توسط پادشاه امپراتوری خودشان، و یا حداکثر پادشاه یک سطح بالاتر در درخت، پوشش داده می‌شود. در حالت اول و طبق تعریف گراف دوبخشی G'، حداکثر فاصله دو است. در حالت دوم نیز این فاصله حداکثر پنج خواهد بود (دو واحد تا پادشاه خودشان و سه واحد تا پادشاه پدر).

آر آنجا که گراف G_i کوچک‌ترین گرافی است که دارای یک جواب بهینه است، پس $OPT \geqslant w(e_i)$ است. همچنین همان‌طور که بیان شد، شعاع جواب الگوریتم پیشنهادی حداکثر $۵.w(e_i)$ هست. درنتیجه جواب الگوریتم، یک جواب ۵-تقریب از جواب بهینه خواهد بود:

$$OPT \geqslant w(e_i) \qquad \& \qquad r \leqslant ۵.w(e_i)$$

$$\Longrightarrow \; r \leqslant ۵.OPT$$

بخش دوم:

پس از اجرای الگوریتم $\text{AssignDomains}(G_i^c)$، هر پادشاه (به‌جز پادشاه‌های سبک) L واحد از درخواست‌ها را پوشش داده‌اند. همچنین بدیهی است که در طی الگوریتم $\text{AssignDomains}(G_i^c)$ اندازه دامنه پادشاه‌های سبک کاهش پیدا نخواهد کرد[۱۰]. همچنین می‌دانیم که تمام مراکز جدید دقیقاً L درخواست را تحت پوشش قرار داده‌اند. درنتیجه مجموعاً $K_{LM} + \left\lceil \frac{R-R_{LM}}{L} \right\rceil$ مرکز ایجاد می‌گردد. □

نتیجه‌ی ۴-۶ *با قبول اثبات قضیه ۴-۵، الگوریتم پیشنهادی این فصل یک جواب ۵- تقریب برای مسئله k- مرکز وزن‌دار با ظرفیت‌های نرم می‌باشد.*

[۱۰] این موضوع ازآن‌جهت اهمیت دارد که اگر اندازه دامنه یک پادشاه کاهش پیدا کند، ممکن است یک مرکز به دو مرکز تبدیل شود که چنین چیزی در الگوریتم ما وجود ندارد

فصل ۵

الگوریتم‌های تقریبی جدید برای مسئله‌ی k-مرکز ظرفیت‌دار در مدل توزیع‌شده

در این فصل سایر نتایج جدید پژوهش را، که به‌طور ویژه بر حل مسئله‌ی k-مرکز با ظرفیت‌های نرم و سخت (تعریف ۱-۲) و در مدل توزیع‌شده تمرکز دارند ارائه می‌دهیم. در تمام الگوریتم‌ها از شیوه مجموعه‌های هسته‌ی ترکیب‌پذیر استفاده شده است. بدین منظور در بخش نخست به معرفی مفاهیم اولیه می‌پردازیم. سپس یک الگوریتم تقریبی توزیع‌شده برای حل مسئله‌ی k-مرکز با ظرفیت‌های نرم در مدل توزیع‌شده ارائه می‌دهیم. سپس با فرض وجود یک الگوریتم برای حل مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم، یک نسخه‌ی بهبودیافته از الگوریتم پیشین خود ارائه می‌دهیم. در ادامه نحوه استفاده از الگوریتم ارائه شده را برای حل مسئله‌ی k-مرکز با ظرفیت‌های سخت بررسی می‌کنیم. در انتها، میزان بهبود به‌دست‌آمده را به کمک مقایسه نتایج به‌دست‌آمده باکارهای پیشین نمایش می‌دهیم.

۵-۱ تعاریف و مفاهیم اولیه

برای راحتی در بخش‌های بعدی، چند تعریف و ساختار نوشتاری را در اینجا معرفی کنیم.

- اگر مجموعه‌ی مرجع U، فضای کل نقاط ممکن تصور شود، برای هر دو نقطه‌ی $a, b \in U$،

فاصله‌ی a و b را با $d(a,b)$ نمایش می‌دهیم:

$$d : U \times U \longrightarrow \mathbb{R}^+$$

- در این گزارش فرض بر این است که فاصله‌ی نقاط دارای خاصیت متریک است.

یعنی: $\forall a,b,c \in U : d(a,c) \leqslant d(a,b) + d(b,c) \qquad (۵-۱)$

و $\forall a,b \in U : d(a,b) = d(b,a)$

- برای نقطه‌ی $a \in U$ و مجموعه نقاط $B \subseteq U$:

منظور از $d(a,B)$، فاصله‌ی a تا نزدیک‌ترین عضو B است؛ یعنی: $d(a,B) = \min\limits_{b \in B} \{d(a,b)\}$

پس: $\forall b \in B : d(a,B) \leqslant d(a,b)$

تعریف ۵-۱ *منظور از $\delta(a,B)$، نزدیک‌ترین عضو B به a است.*

یعنی: $\delta(a,B) = \arg\min\limits_{b \in B} \{d(a,b)\}$

پس: $d(a,B) = d(a,\delta(a,B))$

و $\forall b \in B : d(a,\delta(a,B)) \leqslant d(a,b)$

تعریف ۵-۲ *مجموعه نقاط منتسب شده به یک مرکز مانند c را با $\mathcal{D}_c$ نمایش می‌دهیم و فرض می‌کنیم که $c \in \mathcal{D}_c$ است.*

تعریف ۵-۳ *برای نقطه دلخواه p، منظور از $\phi(p)$ مرکزی است که p به آن منتسب شده است. درنتیجه به ازای هر مرکز مانند c، داریم $c \in \phi(c)$. همچنین داریم:* $p \in \mathcal{D}_c \iff \phi(p) = c$

تعریف ۵-۴ *با توجه به تعریف مسئله «k-مرکز با ظرفیت‌های نرم» (تعریف ۱-۲)، یک مجموعه مانند $D = \{d_۱, d_۲, \ldots, d_m\}$ در صورتی می‌تواند یک «جواب قابل‌قبول» [۱] به ازای مجموعه ورودی S باشد که شرایط زیر وجود داشته باشد:*

۱. $m \leqslant k$

۲. $\forall p \in S : \exists d_i \in D \quad | \quad p \in \mathcal{D}_{d_i}$

[۱] Feasible Solution

۳. $\forall d_i \in D : |\mathcal{D}_{d_i}| \leqslant L$

تعریف ۵-۵ *منظور از r^*_S شعاع بهینه مسئله «k-مرکز با ظرفیت‌های نرم» به ازای ورودی S است.*

۵-۲ الگوریتم تقریبی برای مسئله‌ی k-مرکز با ظرفیت‌های نرم

در این بخش یک الگوریتم تقریبی برای مسئله‌ی k-مرکز با ظرفیت‌های نرم در مدل توزیع‌شده ارائه خواهیم داد. ایده به‌کاررفته در این الگوریتم، استفاده از مجموعه‌های هسته‌ی ترکیب‌پذیر برای ایجاد قابلیت توزیع‌شدگی در یک الگوریتم موجود و غیر توزیع‌شده از مسئله‌ی k-مرکز با ظرفیت‌های نرم است.

خروجی الگوریتم ارائه شده در این بخش یک مجموعه‌ی هسته شامل k نقطه خواهد بود و مراکز نهایی و نحوه انتساب سایر رئوس به این مراکز مشخص نمی‌گردد. اثبات خواهیم کرد که k مرکز در این مجموعه‌ی هسته وجود دارد که یک جواب ۹-تقریب از جواب بهینه است.

ما در بخش ۵-۳ یک الگوریتم کامل‌تر ارائه خواهیم که نه‌تنها ضریب تقریب بهتری دارد، بلکه علاوه بر k مرکز نهایی، نحوه انتساب سایر نقاط به این مراکز را نیز مشخص می‌کند؛ اما اگر فرض کنیم که هدف نهایی، تنها پیدا کردن یک مجموعه‌ی هسته k نقطه‌ای از مراکز باشد، این الگوریتم یک الگوریتم ساده و کارا برای این هدف خواهد بود.

پیش از ارائه الگوریتم و معرفی قضیه مربوط به آن، دو لم بسیار مهم و کلیدی که در ادامه مورداستفاده خواهد بود را معرفی می‌کنیم.

۵-۲-۱ رابطه‌ی جواب مسئله‌ی k-مرکز و مسئله‌ی k-مرکز با ظرفیت‌های نرم

لم ۵-۱ *هر جواب α-تقریب برای مسئله‌ی k-مرکز ، یک مجموعه‌ی هسته‌ی $(\alpha+1)$-تقریب برای جواب مسئله‌ی k-مرکز با ظرفیت‌های نرم است.*

اثبات. در اثبات این بخش از شیوه‌ای مشابه آنچه در کار ضرابی‌زاده و سایرین [۱۰] دیده می‌شود استفاده می‌کنیم و از مقایسه نقاط موجود در مجموعه جواب دو مسئله بهره می‌بریم. مجموعه $C = \{q_1, q_2, \ldots, q_k\}$ که $C \subseteq S$ را بعنوان یک جواب α-تقریب برای مسئله‌ی k-مرکز به ازای ورودی S

درنظر بگیرید و شعاع جواب بهینه را r^* بنامیم. همچنین نماد $\phi(p)$ را مرکزی که نقطه‌ی $p \in S$ در جواب C به آن متصل شده است در نظر می‌گیریم.

بطور مشابه، مجموعه $C^*_{cap} = \{q_1, q_2, \ldots, q_k\}$ که $C^*_{cap} \subseteq S$ را بعنوان جواب بهینه‌ی مسئله‌ی k-مرکز با ظرفیت‌های نرم به شعاع r^*_{cap} درنظر بگیرید و نماد $\phi_{cap}(p)$ را مرکزی که نقطه‌ی $p \in S$ در جواب C^*_{cap} به آن متصل شده است در نظر می‌گیریم.

به ازای هر مرکز $o \in C^*_{cap}$ یک مرکز $o' = \phi(o)$ در نظر بگیرید و مجموعه مراکز جدید را O بنامید. واضح است که O یک زیرمجموعه برای C است. اگر به ازای هر مرکز $o \in C^*_{cap}$ تمام نقاط متصل شده به o را به $\phi(o)$ (مرکز جدید) منتقل کنیم، O یک جواب جدید برای مسئله‌ی k-مرکز با ظرفیت‌های نرم به ازای ورودی S خواهد بود. همچنین داریم:

$$\begin{aligned} \forall p \in S : d(p, \phi(\phi_{cap}(p))) &\leqslant r^*_{cap} + \alpha \cdot r^* && \text{(نامساوی مثلثاتی)} \\ &\leqslant r^*_{cap} + \alpha \cdot r^*_{cap} && (r^* \leqslant r^*_{cap}) \\ &\leqslant (1+\alpha) \cdot r^*_{cap} \end{aligned}$$

در این نگاشت ممکن است چند نقطه از C^*_{cap} به یک نقطه یکسان از C نگاشت شوند. ولی ازآنجاکه در این نسخه از مسئله ظرفیت‌ها نرم است، این نحوه نگاشت مشکلی ایجاد نمی‌کند و طبق تعریف ۵-۴، مجموعه O یک جواب قابل‌قبول برای مسئله‌ی k-مرکز با ظرفیت‌های نرم است. □

۵-۲-۲ یک لم کلیدی

لم ۵-۲ *اگر r^*_S شعاع جواب بهینه مسئله‌ی k-مرکز با ظرفیت‌های نرم به ازای مجموعه نقاط ورودی S باشد، به ازای هر مجموعه $\overline{S} \subseteq S$ داریم:*

$$r^*_{\overline{S}} \leqslant ۲ \cdot r^*_S$$

اثبات. ازآنجاکه در این قسمت دو جواب بهینه (یکی به ازای مجموعه S و دیگری به ازای مجموعه $\overline{S}$) موردبحث است، تعریف ۵-۲ و ۵-۳ را اندکی گسترش می‌دهیم.

فرض می‌کنیم به ازای هر نقطه $p \in \overline{S}$: نماد $\phi(p)$ مرکزی که p در جواب بهینه‌ی S به آن منتسب شده است و نماد $\overline{\phi}(p)$ مرکزی که p در جواب بهینه‌ی $\overline{S}$ به آن منتسب شده است را نشان می‌دهد.
همچنین فرض می‌کنیم که به ازای هر نقطه (مرکز) مانند c: نماد $\mathcal{D}_c$ مجموعه نقاطی که در جواب بهینه‌ی S به c منتسب شده‌اند و نماد $\overline{\mathcal{D}}_c$ مجموعه نقاطی که در جواب بهینه‌ی $\overline{S}$ به c منتسب شده‌اند را نشان می‌دهد.
پس اگر مجموعه $C^* = \{q_۱, q_۲, \ldots, q_k\}$ که $C^* \subseteq S$ جواب بهینه برای ورودی S باشد، می‌دانیم:

$$S = \bigcup_{q_i \in C^*} \mathcal{D}_{q_i}$$

قصد داریم یک مجموعه مانند $\overline{C}$ معرفی نماییم که یک جواب قابل‌قبول (و نه بهینه) برای ورودی $\overline{S}$ باشد. بدین منظور تابع $f : C^* \longrightarrow \overline{S}$ را به‌صورت زیر تعریف می‌کنیم:

$$\forall i \in \{۱, ۲, \ldots, k\} : f(q_i) = \delta\big(q_i, \overline{S}\big) \qquad (۵-۲)$$

حال بر اساس تابع f، یک جواب قابل‌قبول را با تعریف سه عنصر مجموعه $\overline{C} = \{\bar{q}_۱, \bar{q}_۲, \ldots, \bar{q}_k\}$ (که $\overline{C} \subseteq \overline{S}$) و $\overline{\mathcal{D}}$ و $\overline{\phi}$ به‌صورت زیر تعریف می‌کنیم:

$$\forall \bar{q}_i \in \overline{C} : \bar{q}_i = f(q_i)$$
$$\forall p \in \overline{S} : \overline{\phi}(p) = f(\phi(p))$$
$$\forall \bar{q}_i \in \overline{C} : \overline{\mathcal{D}}_{\bar{q}_i} = \big\{p \in \overline{S} | p \in \mathcal{D}_{f^{-1}(\bar{q}_i)}\big\}$$

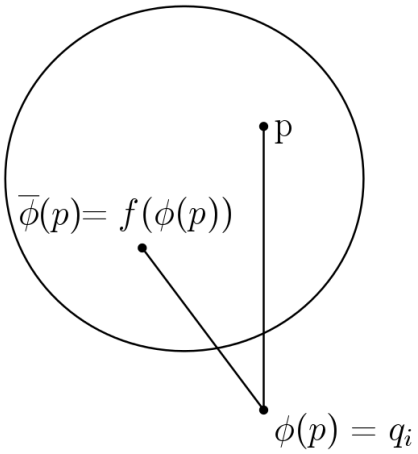
در این نگاشت ممکن است چند نقطه از C^* به یک نقطه یکسان از $\overline{S}$ نگاشت شوند. ولی ازآنجاکه در این نسخه از مسئله ظرفیت‌ها نرم است، این نحوه نگاشت مشکلی ایجاد نمی‌کند و طبق تعریف ۵-۴، مجموعه $\overline{C}$ یک جواب قابل‌قبول برای $\overline{S}$ با شعاع $r_{\overline{S}} = \max_{p \in \overline{S}} d(p, \overline{\phi}(p))$ است. همچنین طبق تعریف ۵-۱ و رابطه ۵-۲ می‌دانیم:

$$\forall p \in \overline{S} : d(\phi(p), \overline{\phi}(p)) = d(\phi(p), \delta\big(\phi(p), \overline{S}\big)) \leqslant d(\phi(p), p) \leqslant r^*_S \qquad (۵-۳)$$

مطابق شکل ۵-۱، به ازای تمام نقاط $p \in \overline{S}$ داریم:

$$\forall p \in \overline{S} : d(p, \overline{\phi}(p)) \leqslant d(p, \phi(p)) + d(\phi(p), \overline{\phi}(p)) \qquad (\text{طبق } ۵-۱)$$

$$\forall p \in \overline{S} : d(p, \overline{\phi}(p)) \leqslant ۲ \cdot r^*_S \qquad (\text{طبق } ۵-۳)$$



شکل ۵-۱: رابطه هندسی بین سه نقطه $p \in \overline{S}$ و $\phi(p)$ و $\overline{\phi}(p)$

که به معنی آن است که درواقع فاصله هر نقطه از مرکزش در جواب $\overline{C}$ می‌تواند حداکثر $۲ \cdot r^*_S$ است.

به‌عبارت‌دیگر: $r_{\overline{S}} \leqslant ۲ \cdot r^*_S \qquad (۵-۴)$

از طرفی می‌دانیم که به ازای هر جوابی مانند $r_{\overline{S}}$ داریم: $r^*_{\overline{S}} \leqslant r_{\overline{S}} \qquad (۵-۵)$

از دو رابطه ۵-۴ و ۵-۵ می‌توان نتیجه گرفت: $r^*_{\overline{S}} \leqslant ۲ \cdot r^*_S$ □

۵-۲-۳ الگوریتم پیشنهادی

روش پیشنهادی ما برای حل مسئله‌ی k-مرکز با ظرفیت‌های نرم به شرح زیر است:

۱. ابتدا مجموعه نقاط ورودی مسئله (S) را به شکل دلخواه به زیرمجموعه‌های $S_۱$ تا S_m افراز کن و هر افراز را به یک ماشین بده. مجموعه نقاط اختصاص داده‌شده به ماشین iام را S_i می‌نامیم.

۲. سپس الگوریتم گنزالس را برای حل مسئله‌ی k-مرکز را بر روی هر ماشین اجرا کن و مجموعه‌ی هسته حاصل را T_i بنام. می‌دانیم که طبق تعریف، مجموعه T_i یک جواب ۲-تقریب برای مجموعه S_i بشمار می‌آید. همچنین به ازای هر $v \in S_i$، $\phi_{S_i}(v)$ را برابر با رأسی از T_i در نظر می‌گیریم که v به آن متصل شده است.

۳. بار دیگر الگوریتم گنزالس را بر روی مجموعه T (اجتماع مجموعه T_iها) اجرا کن و مجموعه حاصل را Q بنام. همچنین به ازای هر $v \in T$، $\phi_T(v)$ را برابر با رأسی از Q در نظر می‌گیریم که v به آن متصل شده است.

۴. مجموعه Q یک مجموعه‌ی هسته‌ی شامل جواب ۹-تقریب است.

برای اثبات درستی الگوریتم فوق، نیاز است تا اثبات کنیم که k مرکز بر روی نقاط موجود در مجموعه Q وجود دارد که یک جواب ۹-تقریب برای مجموعه نقاط ورودی S است. برای اثبات قضیه ابتدا اثبات می‌کنیم که مجموعه Q یک جواب ۸-تقریب برای مسئله‌ی k-مرکز است، و در انتها از لم ۵-۱ برای اثبات وجود یک جواب ۹-تقریب استفاده می‌کنیم.

قضیه‌ی ۵-۳ *خروجی الگوریتم فوق شامل یک جواب ۹-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم است.*

اثبات. ابتدا اثبات می‌کنیم که مجموعه‌ی Q یک جواب ۸-تقریب برای مسئله‌ی k-مرکز به ازای ورودی S است. با توجه به گام دوم الگوریتم، می‌دانیم که مجموعه T_i مجموعه‌ی حاصل از اجرای الگوریتم ۲-تقریب گنزالس بر روی بخش S_i از ورودی است که توسط ماشین iام صورت می‌پذیرد. همان‌طور که در بدنه الگوریتم نیز بیان شد، نماد $\phi_{S_i}(p)$ را مشابه با تعریف اصلی (۵-۳)، مرکزی که نقطه‌ی $p \in S_i$ در جواب T_i به آن متصل شده است در نظر می‌گیریم. همچنین شعاع بهینه به ازای بخش S_i را $r^*_{S_i}$ می‌نامیم. با توجه به تعاریف فوق و الگوریتم استفاده‌شده داریم:

$$\forall p \in S_i : d(p, \phi_{S_i}(p)) \leqslant 2 \cdot r^*_{S_i}$$

همچنین از طرفی به کمک لم ۵-۲ می‌دانیم: $r^*_{S_i} \leqslant ۲ \cdot r^*_S$

پس داریم:

$$\forall p \in S_i : d(p, \phi_{S_i}(p)) \leqslant ۴.r^*_S \qquad (۵-۶)$$

به‌طور مشابه و با توجه به گام سوم الگوریتم نیز خواهیم داشت:

$$\forall t \in T : d(t, \phi_T(t)) \leqslant ۴.r^*_S \qquad (۵-۷)$$

پس:

$$\forall p \in S_i : d(p, \phi_T(\phi_{S_i}(p))) \leqslant ۴.r^*_S + ۴.r^*_S = ۸.r^*_S$$

و درنتیجه می‌توان گفت که به با توجه به توابع ϕ_{S_i} و ϕ_T، به ازای هر نقطه از ورودی، یک مرکز در مجموعه Q وجود دارد که فاصله آن حداکثر $۸ \cdot r^*_S$ است. حال با توجه به اینکه مجموعه Q یک جواب ۸-تقریب برای مسئله‌ی k-مرکز به ازای ورودی S است و همچنین با توجه به لم ۵-۱ می‌توان نتیجه

گرفت که مجموعه‌ی Q یک مجموعه‌ی هسته ۹-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم است و k مرکز را می‌توان طوری بر روی نقاط Q قرار داد که یک جواب ۹-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم باشد. □

۵-۳ الگوریتم بهبودیافته برای مسئله‌ی k-مرکز با ظرفیت‌های نرم

الگوریتم ارائه شده در بخش ۵-۲ یک الگوریتم ۹-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم است. در بعضی کاربردها لازم است تا علاوه بر k مرکز نهایی، نحوه انتساب مجموعه نقاط ورودی به این مراکز نیز مشخص شود. به همین منظور در این بخش یک الگوریتم جدید معرفی خواهد شد که علاوه ارائه k مرکز نهایی، نحوه انتساب رئوس را نیز مشخص می‌کند.

در پیاده‌سازی الگوریتم پیشنهادی این بخش از یک الگوریتم β-تقریب برای حل مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم استفاده می‌شود و یک جواب $۲\beta + ۴$ ارائه خواهد داد که نسبت به الگوریتم ارائه شده در بخش ۵-۲ بهتر است. مراحل الگوریتم پیشنهادی به‌صورت زیر است:

۱. ابتدا مجموعه نقاط ورودی مسئله (S) را به شکل دلخواه به زیرمجموعه‌های $S_۱$ تا S_m افراز کن و هر افراز را به یک ماشین بده. مجموعه نقاط اختصاص داده‌شده به ماشین iام را S_i می‌نامیم.

۲. سپس الگوریتم ۲-تقریب گنزالس را بر روی هر ماشین اجرا کن و مجموعه‌ی هسته حاصل را T_i بنام و به هر رأس $t \in T_i$ وزنی معادل تعداد نقاط منتسب شده به آن مرکز در نظر بگیر. می‌دانیم که طبق تعریف، مجموعه T_i یک جواب ۲-تقریب برای مجموعه S_i بشمار می‌آید. همچنین به ازای هر $v \in S_i$، $\phi_{S_i}(v)$ را برابر با رأسی از T_i در نظر می‌گیریم که v به آن متصل شده است.

۳. یک الگوریتم آفلاین β-تقریب برای حل مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم (مانند الگوریتم ارائه شده در بخش ۴-۲) بر روی مجموعه وزن‌دار T اجرا کن و مجموعه حاصل را Q بنام. همچنین به ازای هر $v \in T$، $\phi_T(v)$ را برابر با رأسی از Q در نظر می‌گیریم که v به آن متصل شده است.

۴. مجموعه Q یک جواب $(۲\beta + ۴)$-تقریب از جواب بهینه خواهند بود. همچنین به ازای هر رأس $v \in S_i$، رأس v را به مرکز $\phi_T(\phi_{S_i}(v)) \in Q$ متصل کن.

برای اثبات درستی الگوریتم فوق، ما نیاز داریم تا اثبات نماییم که k نقطه‌ی موجود در مجموعه Q یک جواب $(۲\beta + ۴)$-تقریب برای مجموعه نقاط ورودی S است. این ادعا در قضیه ۵-۴ آورده شده است که در ادامه بیان می‌کنیم.

قضیه‌ی ۵-۴ *به ازای ورودی S برای مسئله‌ی k-مرکز با ظرفیت‌های نرم، خروجی الگوریتم ۵-۳ یک جواب $(۲\beta + ۴)$-تقریب از جواب بهینه است.*

اثبات. برخلاف اثبات ارائه شده برای قضیه ۵-۳، در اثبات این قضیه از شیوه مقایسه خروجی الگوریتم با نقاط موجود در جواب بهینه استفاده نخواهیم کرد و اثباتی مستقیم ارائه خواهیم داد. با توجه به یکسان بودن گام دوم الگوریتم با الگوریتم قبلی، از تکرار بخشی از اثبات، تعاریف و روابط مربوط به این مرحله از الگوریتم خودداری می‌کنیم. به همین منظور تعریف تابع ϕ_{S_i} و g و شعاع $r^*_{S_i}$ و رابطه ۵-۶ را از اثبات قضیه ۵-۴ صحیح فرض می‌کنیم.

فرض کنیم که الگوریتم بکار گرفته‌شده برای حل مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم در گام سوم از الگوریتم پیشنهادی را $\mathcal{ALG}$ بنامیم. اگر فرض کنیم که الگوریتم $\mathcal{ALG}$ به‌درستی عمل می‌کند، آنگاه جواب الگوریتم پیشنهادی نیز مشخصاً یک جواب صحیح برای مسئله خواهد بود. این استدلال ازآن‌جهت درست است که ما می‌توانیم هر نقطه از مجموعه وزن‌دار T را نماینده تعدادی نقطه دیگر فرض کنیم و اگر الگوریتم $\mathcal{ALG}$ بتواند یک نقطه مانند $p \in T$ (با وزن r_p) را به کمک تعدادی مرکز پوشش دهد، می‌توان نقاطی که در دامنه نقطه p ($\mathcal{D}_p$ به‌دست‌آمده از گام اول الگوریتم) قرار دارند (و حداکثر تعدادشان r_p است) را به این مراکز $\phi(p)$ منتسب کرد. همچنین ازآنجاکه فرض می‌شود که الگوریتم $\mathcal{ALG}$ به هر عضو Q حداکثر L واحد از رئوس وزن‌دار مجموعه T منتسب کرده است، امکان نقض ظرفیت‌ها وجود ندارد.

اگر به ازای هر رأس $p \in S$ فرض کنیم که $t = \phi_{g(p)}(p)$ (که $t \in T$) آنگاه در رابطه با شعاع جواب خروجی الگوریتم فوق داریم:

$$
\begin{aligned}
d(p,\phi_T(t)) &\leqslant d(p,t) + d(t,\phi_T(t)) \\
&\leqslant 2.2.r^*_S + d(t,\phi_T(t)) && \text{(طبق ۵-۶)} \\
&\leqslant 4.r^*_S + \beta.r^*_T && \text{(طبق تعریف } \mathcal{ALG}\text{)} \\
&\leqslant 4.r^*_S + 2.\beta.r^*_S && \text{(طبق قضیه ۵-۲)} \\
&= (4+2\beta).r^*_S
\end{aligned}
$$

پس مجموعه Q یک جواب $(۴ + ۲\beta)$-تقریب از جواب بهینه است.

□

نتیجه‌ی ۵-۵ *اگر از الگوریتم ۵-تقریب ارائه شده در بخش ۴-۲ در پیاده‌سازی الگوریتم پیشنهادی فوق استفاده شود، یک الگوریتم ۱۴-تقریب توزیع‌شده برای مسئله‌ی k-مرکز با ظرفیت‌های نرم به دست می‌آید.*

۵-۴ الگوریتم تقریبی برای مسئله‌ی k-مرکز با ظرفیت‌های سخت

ازآنجاکه در بسیاری از کاربردها امکان «نرم» در نظر گرفتن ظرفیت‌ها وجود ندارد، در این بخش قصد داریم تا یک الگوریتم جدید بر مبنای الگوریتم بخش ۵-۳ برای حل مسئله k-مرکز با ظرفیت‌های سخت ارائه دهیم. درواقع در این بخش ما یک شیوه کلی ارائه خواهیم نمود که به کمک آن می‌توان هر جواب مسئله‌ی k-مرکز با ظرفیت‌های نرم را به یک جواب برای مسئله‌ی k-مرکز با ظرفیت‌های سخت تبدیل نمود. این شیوه را قالب قضیه‌ی زیر بیان می‌کنیم.

قضیه‌ی ۵-۶ *هر جواب α-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم را می‌توان به یک جواب (۲α)-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های سخت تبدیل نمود.*

اثبات. فرض کنید مجموعه $C = \{q_۱, q_۲, \ldots, q_k\}$ که $C \subseteq S$ و تابع انتساب $f : S \longrightarrow C$ یک جواب بهینه α-تقریب برای نسخه ظرفیت نرم مسئله باشد. لازم به ذکر نیست که ممکن است دو مرکز مانند

$q_i, q_j \in C (i \neq j)$ وجود داشته باشند که $q_i = q_j$. بسته به آنکه یک نقطه یک‌بار یا بیش از یک‌بار به‌عنوان مرکز انتخاب‌شده است، می‌توان مجموعه نقاط خروجی این جواب را به دودسته تقسیم‌بندی نمود:

۱. دسته اول آن نقاطی هستند که تنها یک‌بار به‌عنوان مرکز انتخاب‌شده‌اند. در این صورت حداکثر L نقطه وجود دارند که به چنین مرکزی متصل شده‌اند.

۲. دسته دوم آن نقاطی هستند که بیش از یک‌بار به‌عنوان مرکز انتخاب‌شده‌اند و بیش از L نقطه به این نقطه متصل شده است.

اگر تمام اعضای مجموعه C در دسته اول قرار بگیرند، جواب فعلی یک جواب درست برای نسخه سخت از مسئله نیز هست و هیچ عملی لازم نیست.

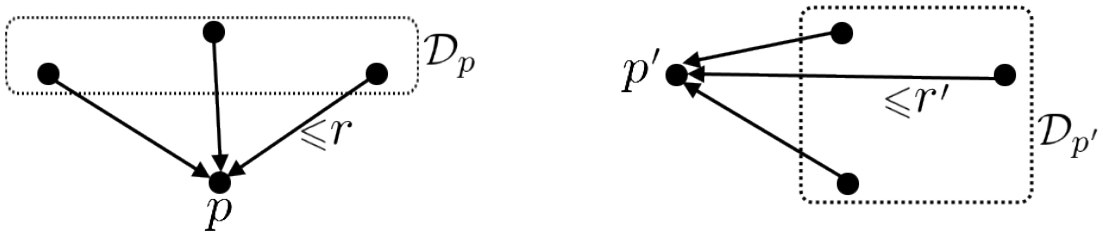
حال فرض کنیم یک نقطه مانند $p \in S$ وجود داشته باشد که t بار در مجموعه C ظاهرشده باشد. برای تمایز بین مراکز واقع‌شده بر روی p، آن‌ها را مجموعه $C_p = \{p_۱, p_۲, \ldots, p_t\}$ می‌نامیم. در این صورت میدانیم که حداکثر $t.L$ نقطه از مجموعه ورودی در دامنه مراکز C_p قرار دارند. ازآنجاکه هدف ما قرار نداشتن هیچ دو مرکزی از جواب بر روی یک نقطه یکسان است، با تکرار گام‌های زیر به ازای هر مرکز $p_i (۱ \leqslant i \leqslant t)$، آن مرکز را با یک نقطه جدید جایگزین می‌کنیم:

۱. دامنه مرکز موردبحث را $\mathcal{D}_{p_i}$ می‌نامیم. یک نقطه مانند $p'_i \in \mathcal{D}_{p_i}$ که $p'_i \neq p_i$ را در نظر می‌گیریم. میدانیم که چنین نقطه‌ای در مجموعه C قرار نداشته است.

۲. نقطه p'_i را به‌عنوان جایگزین مرکز p_i معرفی می‌کنیم.

۳. مجموعه نقاط $\mathcal{D}_{p_i}$ و نقطه p_i را به مرکز جدید متصل می‌کنیم.

ازآنجاکه هر نقطه مانند $p \in S$ فقط و فقط عضو دامنه یک مرکز بوده است، هیچ دو مرکز جدید نیز نمی‌توانند یکسان باشد. درنتیجه، با تکرار عمل بالا به ازای تمام مراکز، به مجموعه‌ای از مراکز جدید خواهیم رسید که از یکدیگر کاملاً متمایز هستند و می‌تواند به‌عنوان یک جواب صحیح برای مسئله‌ی k-مرکز با ظرفیت‌های سخت تلقی گردد. شکل ۵-۲ نحوه اجرای گام‌های فوق را به ازای یک مرکز فرضی p با سه رأس در دامنه خود نمایش می‌دهد. در این مثال، مرکز p با یک نقطه از دامنه $\mathcal{D}_p$ جایگزین شده است. شکل سمت چپ حالت قبل از جابجایی و شکل سمت راست حالت پس از جابجایی را



شکل ۵-۲: نحوه جایگزینی مرکز برای تبدیل جواب نسخه ظرفیت نرم به سخت

نشان می‌دهد.

همچنین به‌راحتی می‌توان اثبات کرد که شعاع جواب جدید، حداکثر دو برابر شعاع اولیه خواهد بود. برای اثبات، فرض کنیم که شعاع جواب به ازای مراکز C را r بنامیم. همچنین مطابق با شکل ۵-۲ فرض کنید $p \in C$ یک مرکز قدیمی باشد و ما قصد داریم $p' \in \mathcal{D}_p$ را به‌عنوان مرکز جایگزین انتخاب کنیم. در این صورت بر اساس اصل نامساوی مثلثاتی داریم:

$$\forall v \in \mathcal{D}_p : d(v,p') \leqslant d(v,p) + d(p,p') \leqslant r + r = ۲.r$$

پس واضح است که فاصله هر نقطه تا مرکز جدیدش حداکثر $۲.r$ خواهد بود. □

نتیجه‌ی ۵-۷ *با توجه به قضیه ۵-۶ و نتیجه ۵-۵ می‌توانیم یک جواب ۲۸-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های سخت به دست بیاوریم.*

فصل ۶

نتیجه‌گیری

در این بخش به جمع‌بندی نتایج نظری به‌دست‌آمده از پژوهش می‌پردازیم. همچنین در انتها به مقایسه این نتایج با باکارهای موجود می‌پردازیم. سپس در انتها مجموعه پیشنهادهایی برای کارهای آتی ارائه می‌شود.

۶-۱ نتایج بدست آمده

با فرض وجود یک الگوریتم β-تقریب برای مسئله k-مرکز وزن‌دار با ظرفیت‌های نرم، موارد زیر نتایج به‌دست‌آمده در این پژوهش را خلاصه می‌کند:

۱. یک الگوریتم ۵-تقریب برای مسئله‌ی k-مرکز وزن‌دار با ظرفیت‌های نرم (قضیه ۴-۵)

۲. یک الگوریتم ۹-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم در مدل توزیع‌شده و به کمک مجموعه‌های هسته‌ی ترکیب‌پذیر (قضیه ۵-۳).

۳. یک الگوریتم $(۴ + ۲\beta)$-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های نرم در مدل توزیع‌شده و به کمک مجموعه‌های هسته‌ی ترکیب‌پذیر (قضیه ۵-۴)

۴. یک الگوریتم $(۸ + ۴\beta)$-تقریب برای مسئله‌ی k-مرکز با ظرفیت‌های سخت در مدل توزیع‌شده و به کمک مجموعه‌های هسته‌ی ترکیب‌پذیر (نتیجه ۵-۷)

با توجه به قضیه ۶-۴ داریم $\beta = ۵$. درنتیجه به ضریب تقریب‌های جدول ۶-۱ برای نسخه‌های مختلف مسئله k-مرکز خواهیم رسید. لازم به ذکر است که تمام این الگوریتم‌ها برای مدل توزیع‌شده طراحی‌شده‌اند.

مسئله	ضریب تقریب	مشخص شدن نحوه انتساب نقاط به مراکز
k-مرکز با ظرفیت‌های نرم	۹	خیر
k-مرکز با ظرفیت‌های نرم	۱۴	بله
k-مرکز با ظرفیت‌های سخت	۲۸	بله

جدول ۶-۱: ضریب تقریب‌های بدست آمده برای نسخه‌های مختلف مسئله‌ی k-مرکز ظرفیت‌دار

همچنین جدول ۶-۲ مقایسه‌ای بین الگوریتم ارائه‌شده در این پژوهش برای حل مسئله‌ی k-مرکز با ظرفیت‌های سخت در مدل توزیع‌شده (بخش ۵-۴) و بهترین کار موجود (پژوهش [۱۲]) ارائه می‌دهد. مشخص است که الگوریتم پیشنهادی در این پژوهش به‌طور مناسبی ضریب تقریب بهترین الگوریتم ماقبل خود را بهبود داده است.

همچنین ازنقطه‌نظر تعداد دور لازم (در مدل نگاشت-کاهش)، الگوریتم پیشنهادی ما با تنها یک دور، بهترین نتیجه را ارائه می‌دهد. نکته حائز اهمیت دیگر آن است که الگوریتم پیشنهادی ما در این پژوهش یک الگوریتم کاملاً قطعی است، درحالی‌که الگوریتم [۱۲] تصادفی بوده و ضریب تقریب یا تعداد دورهای آن در بدترین حالت بسیار بیشتر خواهد بود.

الگوریتم	ضریب تقریب	دور در نگاشت-کاهش	قطعی/تصادفی
پژوهش جاری	۲۸	۱	قطعی
الگوریتم پژوهش [۱۲]	۶۴	$O(۱)$	تصادفی

جدول ۶-۲: مقایسه الگوریتم پیشنهادی با نمونه موجود برای مسئله‌ی k-مرکز ظرفیت‌دار

۶-۲ کارهای آینده

در این پژوهش سعی شد که مسئله‌ی k-مرکز با ظرفیت‌های نرم، k-مرکز با ظرفیت‌های سخت و نسخه وزن‌دار آن موردمطالعه قرار بگیرد. با توجه به اهمیت پردازش داده‌های حجیم و آنکه هدف اصلی ما در این پژوهش طراحی الگوریتم‌هایی برای مدل توزیع‌شده بود، در طراحی الگوریتم‌های ارائه‌شده از روش مجموعه‌های هستهٔ ترکیب‌پذیر استفاده شد. باوجود مزایای مختلف این روش، استفاده از این روش برای حل مسائل خوشه‌بندی بسیار محدود بوده است و به‌طور خاص در کار اخیر باطنی و همکاران [۱۲] قابل‌مشاهده است. درنتیجه به‌کارگیری این شیوه برای ارائه الگوریتم‌های توزیع‌شده جدید با ضرایب تقریب ثابت برای سایر مسائل خوشه‌بندی از قبیل k-میانه، k-میانگین می‌تواند موردتوجه پژوهشگران قرار بگیرد.

هرچند الگوریتم ۵-۴ برای حل مسئله k-مرکز با ظرفیت‌های سخت در مدل توزیع‌شده از بهترین الگوریتم ماقبل خود بهتر عمل می‌کند، در طراحی این الگوریتم از یک رویکرد دومرحله‌ای استفاده‌شده است که امکان بهبود آن وجود دارد. در مرحله اول، مسئله با فرض نرم بودن ظرفیت‌ها حل می‌شود و سپس در مرحله دوم حاصل به جوابی برای نسخه‌ی با ظرفیت‌های سخت تبدیل می‌گردد. طراحی یک الگوریتم مستقیم برای حل مسئله‌ی k-مرکز با ظرفیت‌های سخت در مدل توزیع‌شده می‌تواند به‌عنوان یکی دیگر از پژوهش‌های آتی در نظر گرفته شود. استفاده از الگوریتم ۶-تقریب خولر و سوسمان [۲] برای مسئله‌ی k-مرکز با ظرفیت‌های سخت می‌تواند به انجام این پژوهش کمک کند.

همچنین در این پژوهش فرض یکسان بودن ظرفیت تمام نقاط از ابتدا در تمام مسائل k-مرکز ظرفیت‌دار نظر گرفته شد. حال‌آنکه طراحی الگوریتم‌هایی برای نسخه‌های با ظرفیت‌های غیر یکسان مسئلهٔ k-مرکز می‌تواند به‌عنوان بخش دیگری از کارهای آتی موردتوجه قرار بگیرد.

کتاب‌نامه

[1] M. R. Garey and D. S. Johnson. A guide to the theory of NP-completeness. *WH Freemann, New York*, 1979.

[2] S. Khuller and Y. J. Sussmann. The capacitated k-center problem. *SIAM Journal on Discrete Mathematics*, 13(3):403–418, 2000.

[3] B. Chen, R. Dutta, and G. N. Rouskas. On the application of k-center algorithms to hierarchical traffic grooming. In *Proceedings of 2nd International Conference on Broadband Networks, 2005.*, pages 1218–1224. IEEE, 2005.

[4] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[5] M. Cygan, M. Hajiaghayi, and S. Khuller. LP rounding for k-centers with non-uniform hard capacities. In *Proceedings of the 53th IEEE Symposium on Foundations of Computer Science*, pages 273–282. IEEE, 2012.

[6] H.-C. An, A. Bhaskara, C. Chekuri, S. Gupta, V. Madan, and O. Svensson. Centrality of trees for capacitated k-center. *Mathematical Programming*, 154(1-2):29–53, 2015.

[7] H. Karloff, S. Suri, and S. Vassilvitskii. A model of computation for MapReduce. In *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms*, pages 938–948. Society for Industrial and Applied Mathematics, 2010.

[8] A. Ene, S. Im, and B. Moseley. Fast clustering using mapreduce. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 681–689. ACM, 2011.

[9] S. Im and B. Moseley. Brief announcement: Fast and better distributed mapreduce algorithms for k-center clustering. In *Proceedings of the 27th ACM Symposium on Parallel Algorithms and Architectures*, pages 65–67. ACM, 2015.

[10] S. Aghamolaei, M. Farhadi, and H. Zarrabi-Zadeh. Diversity maximization via composable coresets. In *Proceedings of the 27th Canadian Conference on Computational Geometry*, 2015.

[11] P. Indyk, S. Mahabadi, M. Mahdian, and V. S. Mirrokni. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 100–108. ACM, 2014.

[12] M. Bateni, A. Bhaskara, S. Lattanzi, and V. Mirrokni. Distributed balanced clustering via mapping coresets. In *Advances in Neural Information Processing Systems*, pages 2591–2599, 2014.

[13] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 51(4):606–635, 2004.

[14] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on neural networks*, 16(3):645–678, 2005.

[15] J. Barilan, G. Kortsarz, and D. Peleg. How to allocate network centers. *Journal of Algorithms*, 15(3):385–415, 1993.

[16] M. Cygan and T. Kociumaka. Constant factor approximation for capacitated k-center with outliers. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 25. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

Abstract

Clustering is one of the most well-known fundamental problems in computer science. In this thesis we have focused on a particular version of such problem, called capacititated k-center, and we analyze and survey them in the distributed model, when massive data is given. Our contribution in this research includes a new approximation algorithms with constant approximate factors for such problems in the distributed model, as well as improving the best available algorithm for capacitated k-center problem. Composable coreset as one of the most important techniques in distributed and streaming model is our primary tools in designing these algorithms. This technique gives the opportunity of solving the problem in smaller chunks of data, and giving the result by combining them.

Keywords: Approximation algorithm, Composable coreset, Clustering, Capacitated k-center, Distributed model

Sharif University of Technology

Department of Computer Engineering

M.Sc. Thesis

Approximation Algorithms for Clustering Points in the Distributed Model

By:

Emad Aghajani

Supervisor:

Dr. Hamid Zarrabi-Zadeh

July 2016