فصل دهم

موضوع : خلاصه و نتیجه گیری Mining Social-Network Graphs

به نام خدا

در این بخش قصد داریم خلاصه و موضوعاتی که از این فصل از کتاب که تحت عنوان Mining Social Network Graphs می باشد را داشته باشیم.

در این کار گروهی ابتدا تمامی تصاویر کتاب جدا شده است و در نهایت فصل دهم به طور کامل توسط گروه ما ترجمه شده است. حالا در این فایل خلاصه ای به صورت گزارش ارائه شده است.

گروه ما :

- امیر شکری9811920009
- فرشاد اصغرزاده همپا 9811920004

تمامی کارهای مربوط به این کار گروهی که به عنوان تکلیف در داده کاوی در نظر گرفته شده در گیت هاب انجام شده است زیرا در دوران فعلی که بیماری کرونا باعث ایجاد طرح فاصله گذاری اجتماعی شده است کارهای گروهی باید از راه دور انجام شود.

لینک گیت هاب:

https://github.com/semnan-university-ai/Mining-Social-Network-Graphs

ایمیل اعضای گروه :

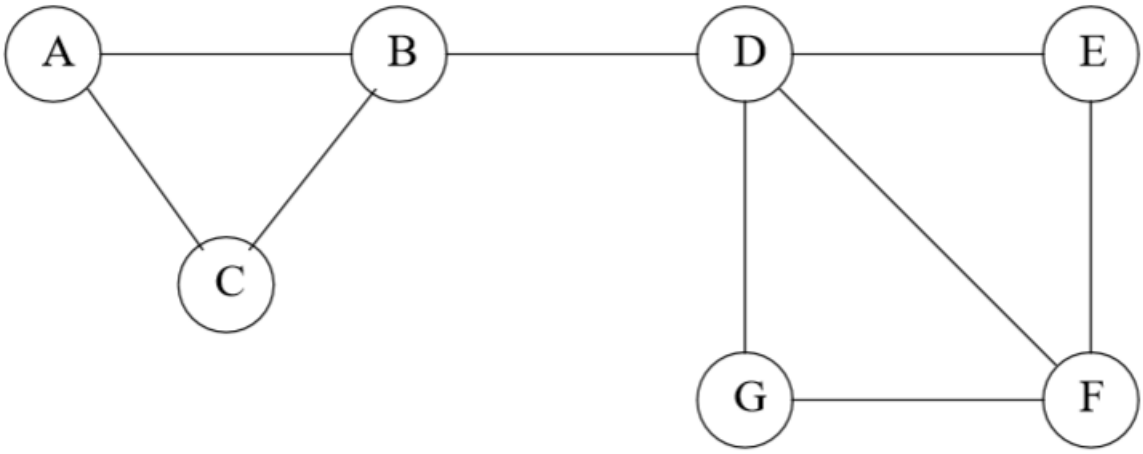
Amirsh.nll@gmail.com
Farshad_asgharzade@hotmail.com

در آخر از دکتر رحمانی منش بابت ارائه ی این تکلیف تشکر میکنیم زیرا با توجه به این فعالیت گروه ما این موضوع را به طور کامل بررسی کرد و تجربه ی انجام یک کار دانشگاهی به صورت تیمی را پیدا کرده است.
استفاده از گیت هاب خلاقیت فکری دونفره ی ما بوده است که امیدواریم مورد قبول شما واقع شود.
ترجمه ی این بخش توسط تیم ما انجام شده است و استفاده از آن در گیت هاب و موارد ... از لحاظ ما مانعی ندارد و به صورت open source در اختیاز عزیزان خواهد بود.

ارادتمند،
تابستان 1399
امیر شکری، فرشاد اصغر زاده همپا

گزارش و خلاصه ی بحث Social network Graphs :

با توجه به تعاریف موجود در کتاب در واقع گراف های شبکه های اجتماعی به این شکل به وضوح بررسی خواهد شد:



به عنوان مثال ما شبکه اجتماعی مثل فیس بوک را در نظر میگیریم.
در شکل بالا هر یک از دایره های که نودهای گراف هستند به عنوان یک فرد یا در واقع یک اکانت موجود در فیس بوک در نظر گرفته می شود.
خطوط یا یالهای گراف بالا ارتباط دوستی کاربران این شبکه را نمایان می کند.
مثلا : گره ی A کاربری به نام amir است که با گره B و C در تماس است. (کاربر B را همان Bahram و کاربر C را ciros در نظر میگیریم.) با توجه به این گراف امیر با بهرام و سیروس دوست است و همزمان سیروس و بهرام هم با یکدیگر دوست می باشند که این روابط در شبکه اجتماعی فیس بوک تحت عنوان Friend شناخته شده است.
البته با وجود این بحث ممکن است به این فکر باشید که تمام بحث Socail Network Graphs در شبکه های اجتماعی اتفاق می افتد که این تفکر غلط است. انواع Social Graphs ها که می توانند در شبکه های مختلف استفاده شوند شامل :

- شبکه های تلفن
- شبکه های ایمیلی
- شبکه های همکاری
- شبکه های اطلاعاتی
- شبکه های زیرساختی
- شبکه های بیولوژیکی
- شبکه های خرید محصول
- و ...

پس منحصرا بحث Social Network Graphs مربوط به شبکه های اجتماعی نمی باشد.

شبکه های تلفن : در شبکه های تلفن ما گره ها را شماره تلفن ها در نظر می گیریم و یال ها را ارتباط تماس بین افراد با آن شماره تلفن؛ به طور مثال در شکل بالا گره A آقای امیر شکری است و گراف B آقای فرشاد اصغرزاده که با توجه به یال های گراف شماره تلفن 0912XXXXXXX مربوط به آقای شکری و شماره تلفن 0935XXXXXXX مربوط به آقای اصغرزاده است.
در گراف بالا وقتی این دو گره به هم متصل هستند یعنی حداقل یک تماس بین آقای شکری و آقای اصغرزاده اتفاق افتاده است.
البته اگر این گراف جهت دار شود این حداقل تماس به صورت تماس دریافتی و تماس ارسالی تغییر می یابد.

شبکه های ایمیلی : شبکه های ایمیلی همانند شبکه های تلفنی است با این تفاوت که در این شبکه گره ها به جای شماره تلفن آدرس ایمیل ها هستند.
مثلا :
A : amirsh.nll@gmail.com
B : Farshad_asgharzade@hotmail.com
با توجه به شکل صفحه ی قبل حداقل یک ایمیل ارسالی بین گره A و گره B موجود است.
البته بحث مهمی که در این بخش موجود بود این است که در این نوع شبکه ها تکلیف ایمیل های اسپم چه می شود؟! با توجه به متن این فصل ایمیل های اسپم تحت گره هایی با یال های ضعیف تر یا یال های با وزن کمتر قرار می گیرند و ایمیل های حقیقی و غیر اسپم با یال های قوی تر یا یال های وزن بالاتر شناخته می شوند.

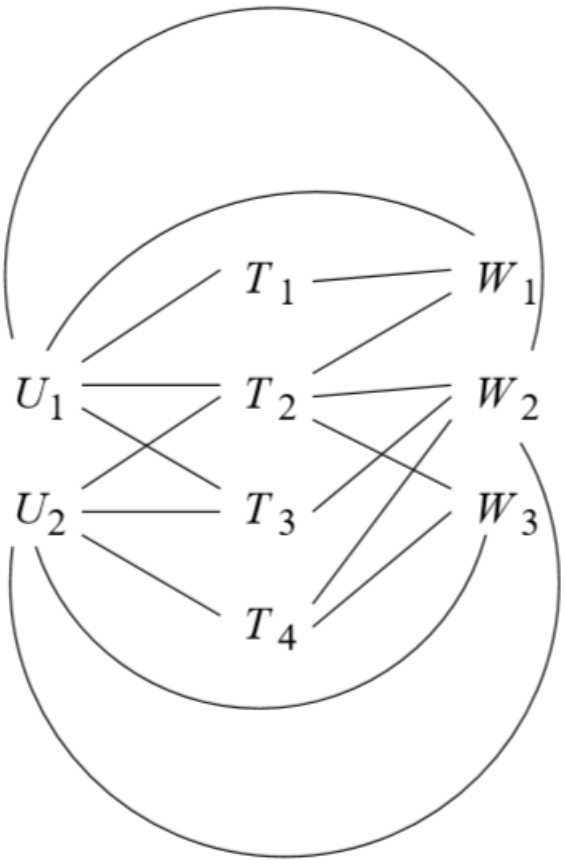
شبکه های همکاری : در این نوع شبکه گره ها تنها از یک نوع نیستند؛ مثال کتاب در این بخش شبکه های افراد محقق و مقاله های منتشر شده توسط آنها است که در این بخش به این شکل است که ارتباط بین دو ناشر و یک مقاله به مفهوم این است که به طور همزمان روی یک مقاله کار کرده اند و البته می توان به جز این دو نوع گره از نوع نشریه هم استفاده کرد که در کل به سه نوع گره بر می خوریم.

پ.ن : شبکه های دیگر را به دلیل زیاد نشدن متن خلاصه چشم پوشی می کنیم ولی در متن ترجمه شده ی گروه ما که در فایل
[pdf][persian] chapter-10-Mining Social Network Graphs.docx
است قرار دارد.

یک بحثی که در بخش شبکه های همکاری به آن اشاره شد گراف هایی بود که در آن گره های آن دارای چندین نوع مختلف بودند. روش پیشنهادی کتاب برای این بخش روشی تحت عنوان k-partite بود که همیشه $k>1$ است که این kدر واقع جدا کننده ی نوع های مختلف است.
در روش k-partite هیچ یک از گره های هم نوع به هم متصل نمی شوند و گروه آنها به گره های دیگر در نوع های دیگر متصل می شوند.
مثالی که در کتاب زده شده است با توجه به اطلاعات سایت del.icio.us است که البته برای اینکه از ماهیت این سایت پی ببریم آن را باز کردیم و متاسفانه تا تاریخ 24 ژولای این سایت از دسترس خارج است و مشکلات فنی دارد؛ اما به بررسی مثال این بخش می پردازیم.
در این بخش در نظر گرفته شده است که نود ها سه نوع هستند:

- برچسب ها یا tags
- صفحات وب یا web page
- کاربران یا users ها

که هر کدام را با کلمه ی اول آنها که T و W و U است در مثال استفاده کرده است.



در شکل بالا می بینید که در واقع ما در این مثال دو کاربر داریم که هیچ اتصالی بین آنها نیست که این قانون اصلی روش k-partite بوده است و چهار tag و سه صفحه ی وب داریم.

این مثال سناریویی به این شکل دارد که کاربران در هر صفحه ی وب از چه تگ هایی استفاده کرده اند و البته تعلق صفحات وب به طور مستقیم به صاحب آن صفحه هم موجود است و همچنین تعلق یک تگ به کاربران و آن تگ به صفحه ی وب هم در تصویر بالا قابل مشاهده است.

بحث های مقدماتی این فصل تا همین جا کافیست به بحث خوشه بندی social graphs ها می پردازیم.
در این فصل بررسی شده است که اگر قصد خوشه بندی گره های موجود در گراف شبکه اجتماعی را داشته باشیم اولین قدم یافتن یک معیار فاصله است که این معیار فاصله باید با توجه به وزن های موجود روی گراف ها در نظر گرفته شود.
اگر گرافی وزن دار نبود تمام یال ها را با وزن 1 در نظر می گیریم و مقدار ثابتی برای آنها در نظر می گیریم.
برای این کار ابتدا یک ماتریس فاصله را برای گراف می سازیم که اگر تعداد گره ها برابر n باشد ما یک ماتریس n*n برای این فاصله ها داریم.
ماتریس فاصله ها در کنار ماتریس اتصال استفاده می شود در ماتریس اتصال هر سطر و ستون نشان دهنده ی اتصال گره سطر به ستون است که در صورتی که آن دو گره به هم متصل بودند مقدار 1 در خانه و در غیراینصورت مقدار 0 را قرار می دهیم. (البته در متن کتاب مقدار 1 و ∞ نیز برای این کار ذکر شده است که تفاوتی ندارد.)
یک مشکلی که در بحث فاصله ها مطرح شده است مشکل نابرابری مثلثی است که در صورتی که گره ها با هم رابطه ی تعدی داشته باشند پیش می آید که برای این موضوع راه حل پیشنهادی این بوده است که گره هایی که اتصال مستقیم دارند را با فاصله ی 1 و گره

های که اتصال غیرمستقیم دارند با فاصله ی 1.5 اعمال می کنیم. به این شکل گره های غیرمستقیم فاصله ی بیشتری دارند و مشکل نابرابری مثلثی را به این شکل حل کرده است.

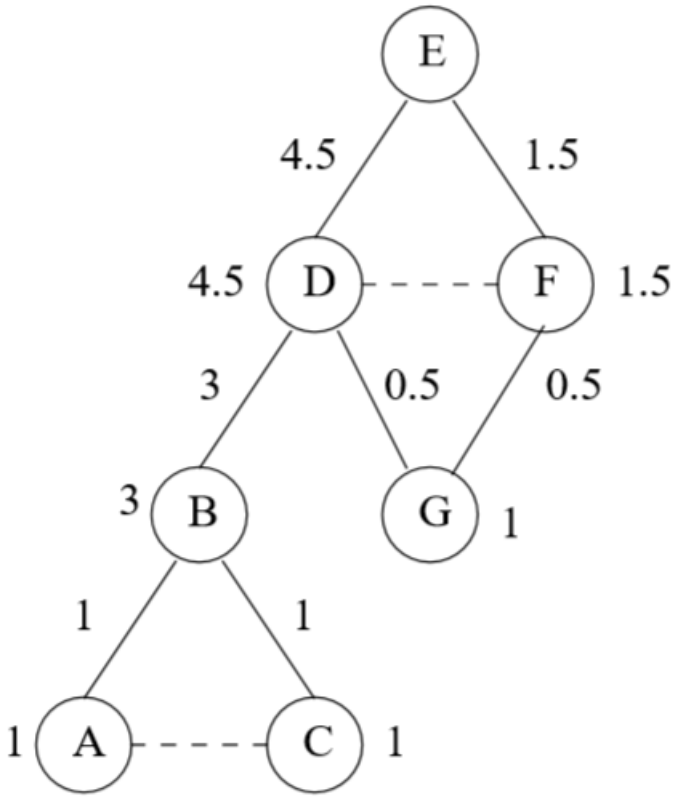
الگوریتم بعدی که در این بخش مطرح شده است الگوریتمی تحت عنوان Girvan-Newman (GN) است که در این الگوریتم از روش نمایش BFS در گراف ها استفاده می شود.
در این الگوریتم یک بار از هر گره X بازدید می کند و تعداد کوتاهترین مسیرها را از X به هر گره دیگری که از هر یک از لبه ها عبور می کند محاسبه می کند. این الگوریتم با روش (BFS) در گراف ، از گره X شروع می کند. توجه داشته باشید که سطح هر گره در نمایش BFS طول کوتاهترین مسیر از X تا آن گره است. بنابراین ، لبه هایی که بین گره ها در یک سطح قرار دارند هرگز نمی توانند بخشی از کوتاهترین مسیری از X باشند.

لبه های بین سطوح ، لبه های DAG نامیده می شوند ("DAG" مخفف directed, acyclic graph است). هر لبه DAG بخشی از حداقل یک مسیر کوتاه از ریشه X خواهد بود. اگر یک لبه ی Dag در (Y,Z) وجود داشته باشد جایی که Y در سطح بالاتر از Z قرار دارد (یعنی نزدیک به ریشه) ، آنگاه ما Y را والدین Z و Z فرزند Y می نامیم ، گرچه آنها لزوماً در DAG والدین یکتایی نیستند که به عنوان یک درخت قرار بگیرند.

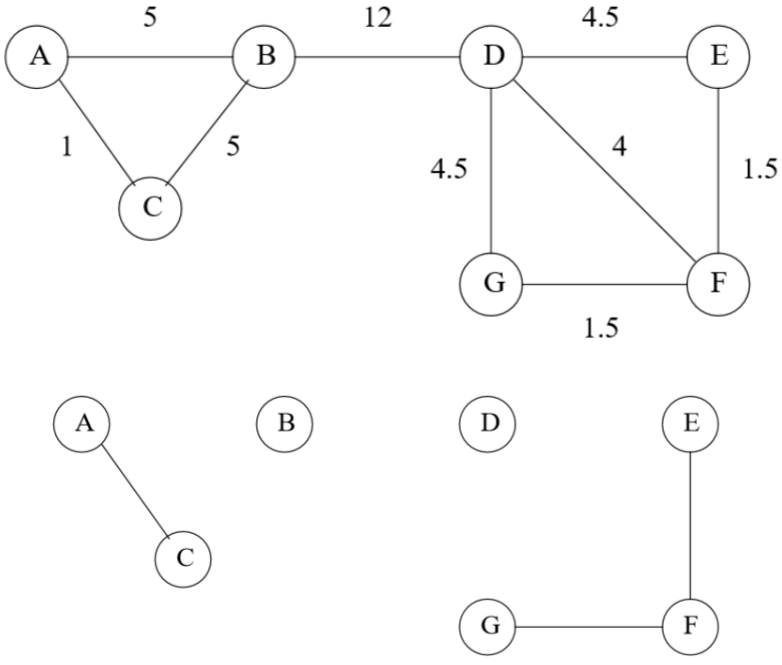
مرحله دوم الگوریتم Girvan-Newman ، برچسب زدن هر گره با تعداد کوتاهترین مسیری است که از ریشه به آن می رسد. با برچسب زدن به ریشه شروع کنید. سپس ، از بالا به پایین ، هر گره Y را بر اساس برچسب های والدین آن ، برچسب گذاری کنید.
مرحله سوم و پایانی محاسبه برای هر لبه ی e و مجموع بیش از همه گره ها Y از کسری از کوتاه ترین مسیرها از ریشه X تا Y است که از e عبور می کند. این محاسبه شامل محاسبه این هزینه برای گره ها و لبه ها ، از پایین است. به هر گره به غیر از ریشه هزینه ی 1 داده می شود ، که کوتاه ترین مسیر را به آن گره نشان می دهد. این هزینه ممکن است بین گره ها و لبه های بالا تقسیم شود ، زیرا ممکن است چندین مسیر کوتاه مختلف به گره وجود داشته باشد. قوانین محاسبه به شرح زیر است:

1. برگ ها در DAG دارای هزینه ی 1 هستند.
2. گره های غیر برگ دارای هزینه ی 1 به علاوه جمع هزینه ی DAG نسبت به نودهای سطح پایینتر است.
3. یک لبه DAG و بخش ورودی به گره Z از سطح فوق با هزینه ی Z متناسب با کسری از کوتاهترین مسیرها از ریشه تا Z که از E عبور می کند ، داده می شود. به طور کلی ، والدین Z را به شکل Y1، Y2، ...، Yk در نظر بگیرید و بگذارید pi تعداد کوتاهترین مسیرها از ریشه تا Yi باشد. این عدد در مرحله دو محاسبه شده است و توسط برچسب ها در شکل 10.4 نشان داده شده است. سپس هزینه لبه (Z ،Yi) برابر $\Sigma_{j=1}^{k} Pj$ است.

پس از انجام محاسبه هزینه ی هر گره به عنوان ریشه ، اعتبارات مربوط به هر لبه را جمع می کنیم. سپس ، از آنجا که هر کوتاهترین مسیر دو بار کشف شده است - یک بار وقتی که هر یک از نقاط انتهایی آن ریشه دارد - باید اعتبار هر لبه را به 2 تقسیم کنیم.
در نهایت ماتریس نهایی از این الگوریتم به شکل زیر می شود :

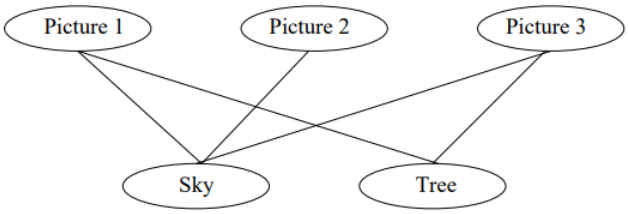


بعد از انجام این الگوریتم مفهومی تحت عنوان Betweenness مطرح می شود که در نهایت گراف بالا را به شکل زیر در می آورد:



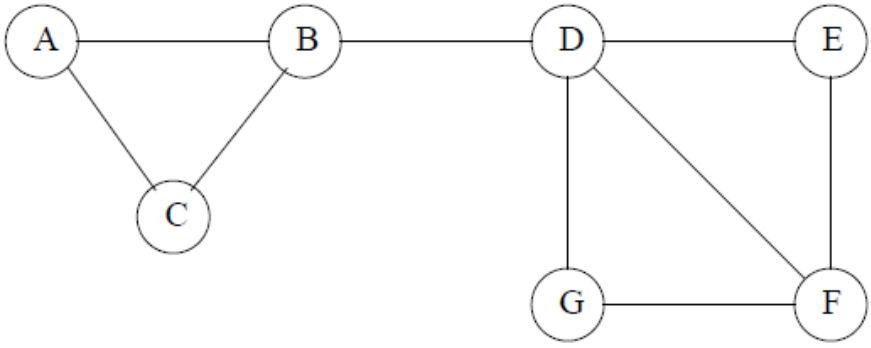
در ادامه ی بحث هم روش های پارتیشن بندی و برش گراف با توجه به ماتریس های لاپلاسین آمده است که برای کوتاه بودن این گزارش خلاصه از ذکر آنها چشم پوشی شده است و این از بخش رد خواهیم شد.

در بخش بعدی در مورد سیمرانک صحبت میشه که برای تحلیل گرافهای شبکههای اجتماعی به کار میرود. سپس در مورد واکر های تصادفی (Random Wallker) ها صحبت میشه که برای مثال می توان فردی رو در نظر گرفت که در یک گراف از یک جایی شروع به حرکت کرده و گره های دیگر را مشاهده می کند. در ادامه در مورد واکر های تصادفی با قابلیت بازنشانی صحبت می کنیم،
مثالی که در این بخش زده میشه به این صورته که تصور کنید سه تصویر با دو دسته بندی مختلف وجود داشته باشه، که میزان شبهات تصاویر رو با استفاده از برچسب های مورد نظر هر کدام با یکدیگر مقایسه می کند.



بخش بعدی مر بوط به شمارش مثلث در گراف ها است، به منظور شمارش انها و زیر گروه های مربوط به هر کدام، مثلا به ساز و کار دنبال کردن در شبکه های اجتماعی چون تویتتر و اینستاگرام فکر کنید، یعنی وقتی گروهی به تازگی تشکیل شده است ، افراد به سمت دوستان هم فکر خود می روند ، اما تعداد مثلث ها نسبتاً کم است. اگر A دوستان B و C را به عنوان دوست خود انتخاب کند، ممکن است که B و C همدیگر را نشناسند. با رشد جامعه ، B و C ممکن است به دلیل عضویت در جامعه تعامل داشته باشند. بنابراین ، یک فرصت خوب وجود دارد که در بعضی مواقع مثلث $\{A ، B ، C\}$ کامل شود.
در مورد مثلث های heavy-hitter و گره هایی با همین نام صحبت میکنیم، یک گره را heavy hitter نامگذاری می‌کنیم اگر درجه آن حداقل $\sqrt{m}$ باشد. یک مثلث heavy-hitter مثلثی است که تمام گره‌های آن heavy hitter باشند.
در بخش بعدی سعی می کنیم مثلثها را با استفاده از MapReduce پیدا کنیم. در مواقعی که گراف خیلی بزرگ باشه ، برای سرعت بخشیدن به عملیات و محاسبات از این روش استفاده می شود.
در ادامه با استفاده از Readuce Task ها کمتر سعی در بهبود سرعت و حافظه داریم.

در بخش بعدی در مورد ویژگی های همسایگی گراف ها صحبت میشه، به تعداد گره هایی است که می توانید از یک گره مشخص در طی یک مسیر کوتاه به آنها رسید. در بعضی موارد ، راه حلهای دقیق برای گرافهایی با میلیون ها گره موجود نیست و ما کار رو تقریبی جلو میبریم. در ادامه در مورد گراف های جهت دار و نحوه استفاده آنها در گراف شبکه های اجتماعی صحبت میشه مثلا از فرد 1 به عنوان گره اول تا فرد 10 به عنوان گره دهم چه مسیری باید طی بشه.



در ادامه در مورد قطر گراف صحبت می کنیم به این صورت که قطر یک گراف جهت دار کوچکترین عدد صحیح d است به گونه ای که برای هر دو گره u و v مسیری به طول d یا کمتر، از u تا v وجود داشته باشد. مفهوم قطر تا زمانی که آن گراف به هم متصل شود ، در یک گراف غیر جهت دار معنا پیدا می کند.
در قسمت بعدی در مورد بستار تعدی و دسترسپذیری یک گره توسط گره دیگر بحث میشود. بستار تعدی ، مجموعه ای از جفت گره ها $(u ، v)$ است به گونه ای که مسیری از u تا v به طول صفر یا بیشتر وجود داشته باشد.

در ادامه در مورد بستار تعدی هوشمند صحبت می کنه و تنها فرقش با نوع معمولی جلوگیری از کشف یا دیدن دوباره گره هایی هست که قبلا توسط الگوریتم مورد نظر دیده شده است، که خب نیاز به حافظه دارد که البته می توان این مشکل را نادیده گرفت.
در بخش بعدی سعی می کنیم با استفاده از بستار تعدی گره هایی که چندان تاثیری در هدف ما ندارند را از گراف مورد نظر حذف کنیم. که باعث بهینه تر شدن گراف اولیه می شود.
در نهایت هم سعی می کنیم تقریبی از اندازه همسایه ها رو به دست آوریم که البته در مورد گرافی با یک میلیارد گره اصلا کار راحتی نیست. ولی خب با استفاده از تکنیک Flajolet-Martin که در بخش 4.4.2 مورد بحث قرار گرفت ، یافتن تقریبی اندازه هر همسایگی چندان سخت نیست.