مواجه با مسائل NP-Hard

در مواجه با مسائل سخت، مانند مسائل NP-Complete و نسخه بهینه‌سازی آنها، چهار راه پیش رو داریم:

- راه حل brute-force : همه حالات ممکن را بررسی کنیم. برای مثال، برای حل مسئله SAT همه مقداردهی های ممکن را امتحان کنیم. این راه حل زمان اجرایش حداقل $\Omega(2^n)$ است اگر n تعداد متغیرهای فرمول باشد. برای مثالی دیگر، در مسئله دور همیلتونی، همه دورهای ممکن را امتحان کنیم. این راه حل زمانش $\Omega(n!)$ است.
- سعی کنیم الگوریتمهایی هوشمندانه [1] اگرچه با زمان اجرای نمایی، اما سریعتر از brute-force پیدا کنیم. در زیر مثالهایی از این قبیل را ذکر می‌کنیم.
- سعی کنیم برای حالتهای خاص مسئله الگوریتم سریع پیدا کنیم. مثلا مسئله 2-SAT که نوع خاصی از مسئله SAT است، در زمان چندجمله‌ای قابل حل است. برای مثالی دیگر، مسئله مسیر همیلتونی در گرافهای DAG آسان است. مسئله پوشش راسی وقتی گراف دوبخشی است در زمان چندجمله‌ای قابل حل است، و مثالهای دیگر.
- در نهایت، در حل نسخه بهینه‌سازی مسائل NP-Complete با تقریب زدن جواب بهینه راضی باشیم. برای مثال در زمان چندجمله‌ای یک پوشش راسی برای یک گراف می‌توان پیدا کرد که اندازه‌اش حداکثر دو برابر اندازه پوشش بهینه است. چند مثال از این گونه الگوریتمها، که به آن الگوریتمهای تقریبی [2] گفته می‌شود هم ارائه می‌کنیم.

۱ الگوریتمهای دقیق برای مسائل NP-Hard

در این قسمت دو نمونه راه حل برای مسائل NP-Complete ارائه می‌کنیم بر اساس یک روند بازگشتی و تکنیک عقبگرد [3] عمل می‌کنند. در واقع الگوریتم سعی می‌کند با انجام چند چک کردن ساده، تعداد حالاتی که بررسی می‌شود را کاهش دهد. بدین وسیله زمان اجرا کمتر از حالت brute-force شود.

۱.۱ الگوریتم هفت برای مسئله 3-SAT

می‌دانیم که الگوریتم bruteforce برای حل مسئله SAT مجموعه‌ای از 2^n حالت مقداردهی را چک می‌کند. یک مشاهده بسیار ساده می‌تواند تعداد مقداردهی ها را برای حالت خاص 3-SAT بطرز قابل توجهی کاهش دهد. می‌دانیم یک فرمول ۳SAT از ترکیب عطفی مجموعه‌ای از جملات سه تایی تشکیل شده است. برای هر جمله $(x \vee y \vee z)$ از هشت حالت مقداردهی ممکن برای لیترالهای جمله، هفت حالت جمله را تصدیق می‌کند. مثلا اگر جمله بصورت $(x_1 \vee \overline{x_2} \vee x_3)$ باشد، هیچ کدام از مقداردهی ها که در آن $x_1 = False$ و $x_2 = True$ و $x_3 = False$ فرمول را تصدیق نمی‌کند. روشن است تعداد این مقداردهی ها 2^{n-3} است که لازم نیست برای بقیه عبارت چک شود. برای بقیه جملات نیز این ایده را می‌توان بکار برد. پس از میان هشت مقداردهی ممکن برای سه لیترال، هفت حالت آن را بررسی می‌کنیم. این مقداردهی در جملات دیگر تاثیر می‌گذارند. جمله‌ای ممکن است

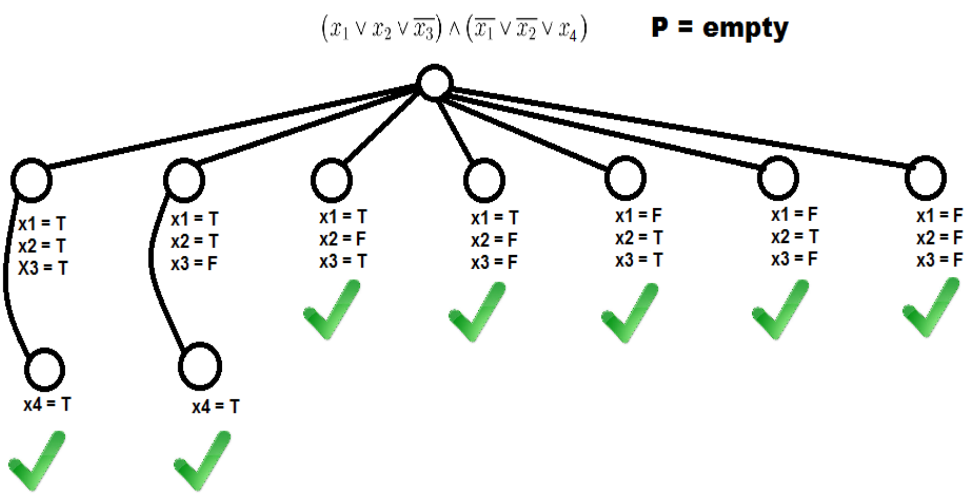
[1] اینجا لفظ هوشمندانه ربطی به هوش مصنوعی ندارد!
[2] Approximation Algorithms
[3] Backtracking

کاملا False شود (به معنای بن بست و عقبگرد است.) جملاتی ممکن است ۲ تایی یا ۱ تایی شوند یا اینکه کلا True شوند.

الگوریتم بصورت بازگشتی عمل می‌کند که در زیر توصیف شده است. اینجا P یک مقداردهی ناتمام را تعریف می‌کند. در ابتدا P تهی است. یعنی هیچ متغیری مقداردهی نشده است. یک ایده دیگر همه به الگوریتم اضافه می‌کنیم که باعث تسریع بیشتر می‌شود. اگر فرمول حاصل (بعد از یک سری مقداردهی) به یک فرمول ۲SAT تقلیل پیدا کرد، لازم نیست روند بازگشتی را ادامه دهیم چون فرمول ۲SAT مستقیما در زمان چندجمله‌ای قابل چک کردن است.

ALG(ϕ: 3CNF formula, P: Partial Assignment)

- If ϕ is in $2-CNF$ format use 2SAT algorithm to solve it. If ϕ is satisfiable return True otherwise return False.
- Otherwise find a clause $C = (x \vee y \vee z)$ that is not touched.
- Consider the 7 ways of initializing x, y, z that satisfy C. For each case, let P_i be the extension of P to the new assignment and let ϕ_i be the resulting formula. Return $\vee_{i=1}^{7} ALG(\phi_i, P_i)$



تحلیل زمان اجرای الگوریتم

فرض کنید $T(n)$ تعداد فراخوانی‌های بازگشتی در الگوریتم هفت باشد. اینجا n تعداد متغیرهاست. مقدار $T(n)$ در واقع تعداد رئوس در درخت بازگشت الگوریتم است. چون در هر انشعاب ۳ متغیر مقداردهی شده و از فرمول حذف می‌شوند، داریم

$$T(n) \le 7T(n-3) + 1$$

بعد از i مرحله بازگشت داریم:

$$T(n) \leq 7^i T(n-3i) + 7^i + 7^{i-1} + \cdots + 7 + 1$$

با قرار دادن $i = n/3$ بدست می‌آید،

$$T(n) \leq 7^{n/3} T(0) + 7^{n/3} + 7^{n/3-1} + \cdots + 7 + 1$$

با فرض $T(0) = 0$ بدست می آید،

$$T(n) \leq O(7^{n/3+1}) = O(1.913^n)$$

در هر فراخوانی بازگشتی مقداری کار انجام می‌شود. مثلا پیدا کردن جمله‌ای با سه متغیر دست نخورده و غیره. اگر تعداد جملات جمله m باشد، این کارها در زمان $O(m)$ قابل انجام است. در نتیجه زمان اجرای الگوریتم $O(m1.913^n)$ است. [۴]

۲.۱ الگوریتمی بهتر برای مسئله پوشش راسی

در مسئله پوشش راسی Vertex Cover می‌پرسیم آیا گراف ورودی $G = (V, E)$ زیرمجموعه $S \subseteq V$ با اندازه حداکثر k دارد که همه یالها را پوشش دهد؟ یعنی هر یال $e \in E$ حداقل یکی از رئوسش داخل S باشد. در بخشی قبلی اثبات کردیم پوشش راسی یک مسئله NP-Complete است. الگوریتم brute-force برای پوشش راسی، همه زیرمجموعه‌های k تایی از رئوس را امتحان می‌کند. بدین ترتیب اگر گراف ورودی m یال داشته باشد، زمان اجرای الگوریتم brute-force از مرتبه $O(\binom{n}{k}m) = O(n^k m)$ خواهد بود.

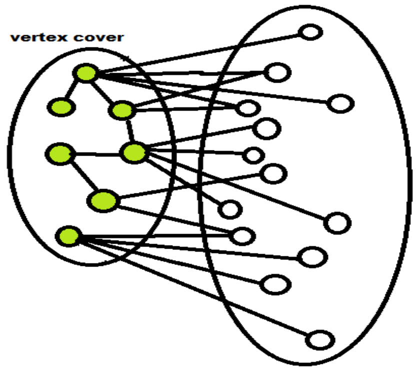
اینجا الگوریتمی برای مسئله پوشش راسی را معرفی می‌کنیم که زمان اجرایش $O(2^k kn)$ است. طبیعتا این زمان چند جمله‌ای نیست چون برای مثال وقتی $k = \Omega(n)$ زمان اجرا نمایی خواهد بود. اما در مقابل زمان اجرای brute-force وضعیت به مراتب بهتر است. مخصوصا حالتی را در نظر بگیرید که $k = O(\log n)$. در این حالت الگوریتم brute-force زمان اجرای فوق چند جمله‌ای دارد ولی الگوریتم جدید زمانش از مرتبه چندجمله‌ای خواهد بود. این خود نشان می‌دهد که مسئله پوشش راسی برای حالتی که $k = O(\log n)$ یک مسئله NP-Complete نیست! [۵]

الگوریتم یک توصیف بازگشتی دارد و از تکنیک عقبگرد استفاده می‌کند. الگوریتم از دو مشاهده ساده زیر استفاده می‌کند.

- اگر گراف یک پوشش راسی با k راس داشته باشد، آنگاه تعداد یالهای گراف حداکثر nk است. از این ناشی می‌شود که همه یالها توسط k راس پوشش داده شده‌اند.
- اگر G یک پوشش راسی با اندازه k داشته باشد که شامل راس u است، آنگاه گراف $G-u$ یک پوشش راسی با اندازه $k-1$ دارد.

[۴] در حال حاضر سریعترین الگوریتم برای حل مسئله 3-SAT پیچیدگی زمانی $O(1.439^n)$ دارد. مقاله Konstantin Kutzkov, Dominik Scheder, 2010 را ببینید. حدس قوی بر این است که هیچ الگوریتمی با زمان $2^{o(n)}$ برای مسئله 3-SAT وجود ندارد.

[۵] یک کلاس خاص از مسائل NP-Complete به نام FPT وجود دارد که شامل مسائلی مانند پوشش راسی می‌شود. اینها مسائلی هستند که الگوریتمی با زمان اجرای $O(f(k)n^{O(1)})$ دارند. مسئله Independent-Set الگوریتمی با چنین زمان اجرایی ندارد و حدس زده می‌شود این مسئله جزو FPT نیست.



ALG $(G = (V, E), k)$:

1. If G contains no edge, then return True.
2. If $|E| > nk$ return False.
3. Let (u, v) be an edge in E.
4. If ALG$(G - u, k - 1)$ or ALG$(G - v, k - 1)$ then return True, Otherwise return False.

تحلیل زمان اجرای الگوریتم

زمان اجرای الگوریتم از رابطه بازگشتی زیر پیروی می‌کند.

$$T(n, k) \leq 2T(n - 1, k - 1) + O(nk)$$
$$T(n, 0) = O(1), T(0, k) = O(1)$$

از حل این رابطه بازگشتی نتیجه زیر حاصل می‌شود.

$$T(n, k) \leq O(2^k nk).$$

در حال حاضر سریعترین الگوریتم برای مسئله پوشش راسی پیچیدگی زمان $O(1.2832^k nk)$ دارد.

۲ الگوریتمهای چندجمله‌ای برای حالات خاص

همانطور که ذکر شد، یکی از راههای مواجه با سختی مسائل NP بررسی حالات خاص مسئله است که ممکن است چندان سخت نباشند. برای مثال اگرچه مسیر همیلتونی یک مسئله سخت است، اما حل مسئله در گرافهای DAG آسان است. مسئله Independent-Set برای درختها در زمان $O(n)$ قابل حل است و پوشش راسی در گرافهای دوبخشی همانطور که خواهیم دید معادل با مسئله تطابق بیشینه است. در این قسمت دو نمونه از حالات خاص مسائل سخت را بررسی می‌کنیم.

۱.۲ یک الگوریتم چندجمله‌ای برای 2-SAT

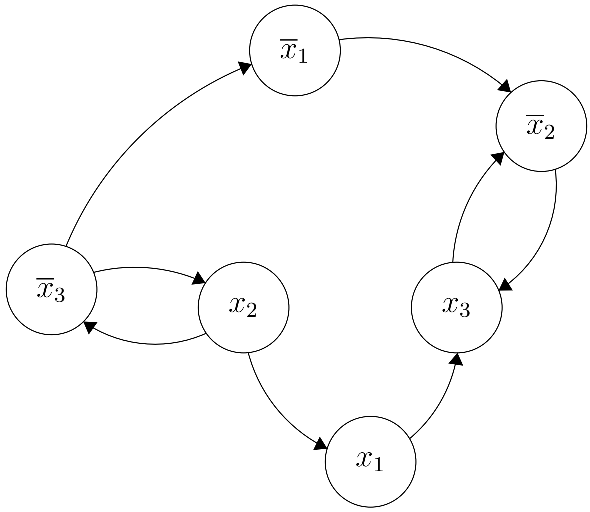
چگونه می‌توان در زمان چند جمله‌ای صدق پذیری یک فرمول منطقی که ترکیبی عطفی از جملات شامل دو لیترال است را مشخص کرد؟ در این قسمت به پرسش می‌پردازیم.

$$\phi = (x_1 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_3) \wedge (x_2 \vee x_3) \wedge (\overline{x}_2 \vee \overline{x}_3)$$

ایده اصلی: فرمول منطقی داده شده را تبدیل به یک گراف جهت دار می‌کنیم و مسئله را به چند پرسش در مورد وجود مسیر بین راسها تقلیل می‌دهیم.

گراف دلالت

با داشتن فرمول ϕ گراف جهت دار G_ϕ را می‌سازیم. برای هر متغیر و نقیض آن یک راس در گراف قرار می‌دهیم. اگر جمله $(\alpha \vee \beta)$ در فرمول آمده است، یالهای $(\overline{\alpha}, \beta)$ و $(\overline{\beta}, \alpha)$ را در گراف قرار می‌دهیم. به این گراف، گراف دلالت [۶] فرمول ϕ گفته می‌شود.



الگوریتم بر پایه قضیه زیر استوار است.

قضیه ۱.۲. *فرمول ϕ صدق پذیر است اگر و فقط اگر متغیر x در فرمول نباشد بطوریکه در گراف دلالت G_ϕ مسیری از x به $\overline{x}$ و مسیری از $\overline{x}$ به x داشته باشیم.*

قبل از اینکه این قضیه را اثبات کنیم، توجه کنید که برای تعیین صدق پذیری فرمول ϕ کافی است که گراف دلالت آن را بسازیم و سپس برای هر متغیر در فرمول وجود مسیر بین آن متغیر و نقیضش (و بالعکس) را چک کنیم. اگر چنین متغیری پیدا کنیم، گزارش می‌کنیم که ϕ صدق پذیر نیست در غیر این صورت صدق پذیر است. اگر الگوریتمی مثل BFS یا DFS را بکار ببندیم، با توجه اینکه گراف دلالت $2m$ یال دارد، الگوریتم پیشنهادی در زمان $O(nm)$ قابل پیاده‌سازی است. یک ایده بهتر (که در اثبات قضیه هم به ما کمک می‌کند) استفاده از تجزیه به مولفه‌های قویا همبند است. روشن است اگر از x به $\overline{x}$ و از $\overline{x}$ به x مسیری داشته باشیم آنگاه x و نقیضش

[۶] Implication Graph

$\overline{x}$ در یک مولفه قویا همبند قرار خواهند گرفت. لذا در صورت داشتن مولفه‌های قویا همبند، اگر برای هر راس مشخص باشد که جزو کدام مولفه همبندی است، کافی است برای هر متغیر مولفه‌های همبندی x و $\overline{x}$ را بپرسیم. در صورتیکه یکسان بودند، گزارش می‌کنیم که فرمول صدق پذیر نیست. از آنجا که الگوریتمی با زمان اجرای خطی یعنی $O(n+m)$ برای محاسبه مولفه‌های همبندی وجود دارد [۷] پس مسئله 2-SAT هم در زمان $O(n+m)$ قابل حل است. اینجا n تعداد متغیرها و m تعداد جملات است.

حال سراغ اثبات قضیه می‌رویم. دو جهت گزاره مورد نظر را در دو لم جداگانه اثبات می‌کنیم. ابتدا جهت آسانتر را بیان می‌کنیم.

لم ۲.۱. *اگر متغیر x وجود داشته باشد بطوریکه مسیری از x به $\overline{x}$ و مسیری از $\overline{x}$ به x در گراف دلالت موجود باشد، آنگاه ϕ صدق پذیر نیست.*

اثبات: یک مقداردهی برای متغیرها در واقع معادل با یک برچسب گذاری برای رئوس گراف دلالت با برچسبهای T یا F است بطوریکه یک لیترال و نقیضش برچسبهای متفاوت بگیرند. به این یک برچسبگذاری معتبر می‌گوییم. حال یک برچسبگذاری معتبر را در نظر بگیرید. اگر یالی از T به F داشته باشیم، آنگاه ادعا می‌کنیم، جمله‌ای در فرمول داریم که تصدیق نشده است. این به سادگی قابل مشاهده است. فرض کنید (x,y) یک یال در گراف دلالت باشد که x برچسب T و راس y برچسب F گرفته است. طبق تعریف گراف وجود یال (x,y) یعنی وجود جمله $\overline{x} \vee y$ در فرمول داده شده. بدیهی است که این جمله ارزش True پیدا نکرده.

حال با این مشاهده، فرض کنید مسیری از x به $\overline{x}$ و مسیری از $\overline{x}$ به x داریم. ادعا می‌کنیم در هر برچسبگذاری معتبر یالی از T به F ایجاد می‌شود. دلیل این مطلب روش است. اگر x برچسب T بگیرد، چون لاجرم $\overline{x}$ برچسب F می‌گیرد پس به دلیل وجود مسیر از x به نقیضش، یک یال از T به F پیدا می‌شود. حالتی که x برچسب F می‌گیرد، نتیجه‌ای مشابه را بدست می‌دهد. $\square$

لم ۲.۲. *اگر هیچ متغیری نداشته باشیم که از خودش به نقیضش و از نقیضش به خودش مسیری داشته باشد، آنگاه ϕ صدق پذیر است.*

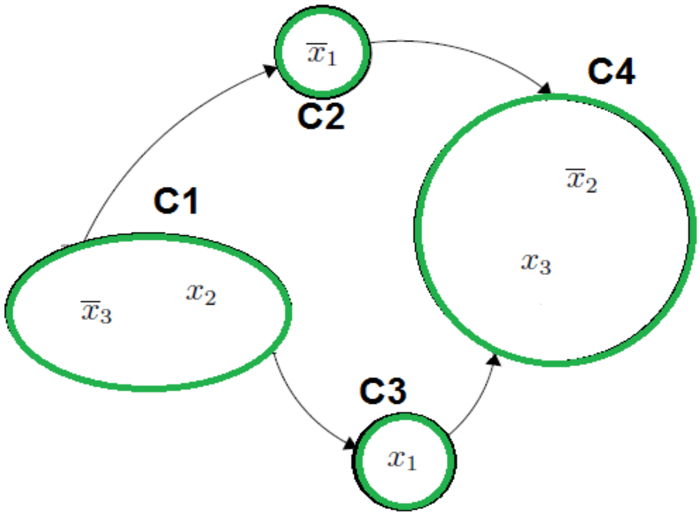
اثبات: ادعا می‌کنیم در این حالت یک برچسب گذاری معتبر برای رئوس گراف دلالت وجود دارد بطوریکه هیچ مسیری از T به F در آن وجود ندارد. این معادل با یک مقداردهی است که فرمول را تصدیق می‌کند.

برای اثبات ادعا از مولفه‌های قویا همبند گراف G_ϕ کمک می‌گیریم. فرض کنید C مجموعه مولفه‌های قویا همبند گراف باشد. گراف متناظر با مولفه‌ها C را تشکیل می‌دهیم [۸]. در این گراف هر مولفه $C_i \in C$ یک راس است. یالی از راس C_i به C_j قرار می‌دهیم اگر و فقط اگر $x \in C_i$ و $y \in C_j$ وجود داشته باشد بطوریکه (x,y) یک یال در گراف G_ϕ باشد. برای مثال شکل زیر گراف مولفه‌های مثالی که در ابتدای این بحث آورده شده را نشان می‌دهد.

---

[۷] به لطف زحمات Robert Tarjan این را می‌دانیم.

[۸] به این Component Graph گفته می‌شود.



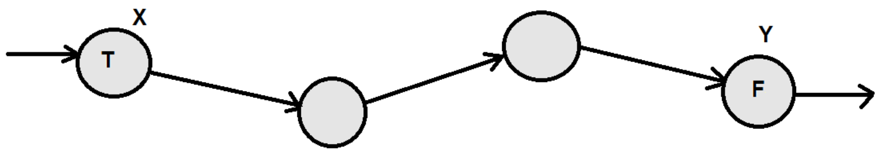
گراف مولفه‌ها یک DAG یعنی گراف جهت دار بدون دور تشکیل می‌دهد [۹]. در نتیجه یک ترتیب توپولوژیکی دارد. فرض کنید

$$C_1 \to \cdots \to C_k$$

یک ترتیب توپولوژیکی برای گراف مولفه‌ها باشد. حال آماده هستیم تا یک برچسبگذاری معتبر برای رئوس گراف G_ϕ پیشنهاد دهیم. صبر کنید! قبل از آن، یک مشاهده دیگر هم انجام دهیم. وقتی لیترالهای x و y در یک مولفه باشند، آنگاه $\overline{x}$ و $\overline{y}$ هم در یک مولفه خواهند بود. چون وجود مسیر از x به y در گراف G_ϕ به معنی وجود مسیر از $\overline{y}$ به $\overline{x}$ است (چرا؟). این مشاهده یعنی اینکه هر مولفه C_i یک مولفه دوگان $\overline{C}_i$ دارد که دقیقا شامل نقیض لیترالهایی است که داخل C_i هستند. به مثال شکل بالا توجه کنید. C_1 و C_4 دوگان هم هستند. C_3 و C_2 نیز همینطور.

حال می‌توانیم به برچسبگذاری بپردازیم. از مولفه آخر، یعنی C_k شروع می‌کنیم. به این مولفه برچسب T می‌دهیم (این یعنی اینکه همه لیترالهای داخل مولفه برچسب T می‌گیرند). این به نوبه خود باعث می شود، مولفه دوگان C_k برچسب F بگیرد. ادامه می‌دهیم، در ترتیب توپولوژیکی به سمت عقب می‌رویم و اولین مولفه‌ای که برچسب گذاری نشده را برچسب T می‌زنیم و دوگانش را F برچسب می‌زنیم. به این روند ادامه می‌دهیم تا اینکه همه مولفه‌ها برچسب گذاری شوند.

ادعا می‌کنیم در طی برچسب گذاری موقعیتی ایجاد نمی‌شود که در آن مولفه X به Y مسیر داشته باشد و درحالیکه X برچسب T و Y برچسب F گرفته است. مانند شکل زیر.



فرض کنید این اتفاق بیافتد (برهان خلف). اولا امکان ندارد که ما X را T برچسبگذاری کرده باشیم در حالیکه Y برچسب نخورده. چون با توصیفی که کردیم، وقتی مولفه‌ای را T برچسب می‌زنیم تکلیف آنهایی که در سمت راستش در ترتیب توپولوژیکی هستند (که شامل Y هم می‌شود) روشن شده. پس فقط وقتی ممکن است این اتفاق بد بیافتد که ما به X برچسب T می‌زنیم در حالیکه Y قبلا برچسب F خورده. طبق فرض، وجود دارد $x \in X$ و

[۹] این از تعریف تجزیه به مولفه‌های قویا همبند ناشی می‌شود.

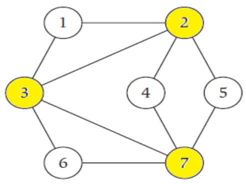
$y \in Y$ بطوریکه مسیری از لیترال x به y در گراف G_ϕ داریم. در نتیجه مسیری از $\overline{y}$ به $\overline{x}$ در گراف G_ϕ وجود دارد. دقت کنید که $\overline{y} \in \overline{Y}$ و $\overline{x} \in \overline{X}$. چون Y برچسب F گرفته پس قبلش $\overline{Y}$ برچسب T گرفته. از طرف دیگر، چون مسیری از $\overline{y}$ به $\overline{x}$ داریم، پس $\overline{X}$ قبل از برچسب خوردن $\overline{Y}$ تکلیفش مشخص شده. این یعنی اینکه دوگانش که X باشد زودتر از همه اینها برچسب خورده که متناقض با فرض ماست. این ادعای ما را ثابت می‌کند. □

۲.۲ پوشش راسی در گرافهای دوبخشی

در انتهای این بخش، به مسئله پوشش راسی برمی‌گردیم و نشان می‌دهیم این مسئله برای گرافهای دوبخشی در زمان چندجمله‌ای قابل حل است. ابتدا چند نمادی که می‌خواهیم استفاده کنیم را تعریف می‌کنیم.

تعریف: برای گراف G عدد پوشش راسی آن به معنی اندازه کوچکترین زیرمجموعه رئوس است که همه یالهای گراف را پوشش دهد. عدد پوشش راسی یک گراف را با $\beta(G)$ نمایش می‌دهیم.

مثال: عدد پوشش راسی گراف زیر ۳ است.



تعریف: اندازه یک تطابق بیشینه در گراف G را با $m(G)$ نمایش می‌دهیم.

لم زیر یک رابطه کلی میان عدد پوشش راسی و اندازه تطابق بیشینه یک گراف را بیان می‌کند.

لم ۳.۲. *برای هر گراف G، عدد پوشش راسی آن بیشتر یا مساوی با اندازه تطابق بیشینه است. به عبارت دیگر،* $\beta(G) \geq m(G)$.

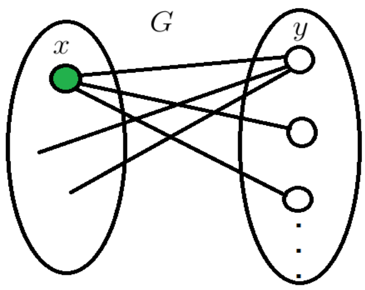
اثبات: اگر $S \subseteq V$ یک پوشش راسی برای گراف G باشد، مخصوصا باید یالهای یک تطابق بیشینه را پوشش دهد. پس از هر یال تطابق بیشینه یکی از دو سرش باید داخل S باشد. این لم را ثاتب می‌کند. □

در مورد گرافهای دوبخشی عدد پوشش راسی همواره برابر با اندازه تطابق بیشینه است.

قضیه ۲.۲. *اگر G یک گراف دوبخشی باشد، آنگاه* $\beta(G) = m(G)$.

اثبات: حالتی که G یک مسیر یا دور به طول زوج است، روشن است که عدد پوشش راسی برابر با اندازه تطابق بیشینه است. این را می‌توانید چک کنید. طبیعتا این مطلب در مورد گرافی که مجموعه‌ای از مسیرها یا دورهای به طول زوج است نیز صادق است. پس در ادامه فرض می‌کنیم که G مجموعه‌ای از مسیرها یا دورهای مجزا نیست.

از برهان خلف استفاده می‌کنیم. فرض کنید که G یک گراف دوبخشی است اما $\beta(G) > m(G)$. می‌توانیم فرض کنیم که G یک مثال نقض کمینه است. یعنی اگر یالی را از آن حذف کنیم، یا راسی را از آن حذف کنیم، عدد پوشش راسی و اندازه تطابق بیشینه برابر می‌شوند. از طرف دیگر چون $G = (V, E)$ مجموعه‌ای از مسیرها یا دورهای مجزا نیست، پس حتما یک راس با درجه حدقل ۳ دارد. فرض کنید $x \in V$ راسی با درجه حداقل ۳ باشد و y یکی از همسایه‌هایش باشد.



دو حالت داریم:

- در هر تطابق بیشینه برای G راس y استفاده می‌شود. این یعنی $m(G-y) < m(G)$. از طرف دیگر، چون G یک مثال نقض کمینه است، پس $\beta(G-y) = m(G-y)$. این دو تا را کنار هم بگذاریم، بدست می‌آید، $\beta(G-y) < m(G)$. اما اگر W یک پوشش راسی برای $G-y$ باشد و ما y را به آن اضافه کنیم، یک پوشش راسی برای G بدست می‌آید. یعنی

$$\beta(G) \leq |W| + 1 = \beta(G-y) + 1 \leq m(G).$$

این با فرض خلف ما یعنی $\beta(G) > m(G)$ در تناقض است.

- تطابق بیشینه M برای G وجود دارد که از راس y استفاده نمی‌کند. چون M از راس y استفاده نمی‌کند حتما راس x در تطابق استفاده شده. از طرف دیگر، چون x درجه ۳ دارد پس یالی روی آن، غیر از (x,y)، وجود دارد که عضوی از M نیست. این یال را f می‌نامیم.

چون $G-f$ یک مثال نقض کمینه است، پس $\beta(G-f) = m(G-f)$. فرض کنید W یک پوشش راسی کمینه برای $G-f$ باشد. ادعا می‌کنیم y نمی‌تواند عضوی از W باشد. فرض کنید اینطور نباشد و $y \in W$. چون W همه یالهای M را پوشش می‌دهد پس از هر یال M یک سرش باید در W باشد. چون M از راس y استفاده نمی‌کند، این یعنی اینکه $|W| = |M| + |\{v\}| = |M| + 1 > m(G-f)$ که با کمینه بودن مثال نقض G در تناقض است. پس y عضوی از W نیست.

از طرف دیگر، چون W یک پوشش راسی برای $G-f$ است، باید یال (x,y) را پوشش دهد و این یعنی اینکه لزوما $x \in W$. اما اگر x عضوی از پوشش راسی باشد، همه یالهای گراف که شامل f هم می‌شود، پوشش داده می‌شود. در نتیجه W یک پوشش راسی برای G است. یعنی

$$\beta(G) = |W| = \beta(G-f) = m(G-f) = m(G).$$

این با فرض $\beta(G) > m(G)$ متناقض است. □