

گراف

گراف یک ساختمان داده غیر خطی است که کاربردهای وسیعی دارد. در این بخش برخی از اصطلاحات نظریه گراف، روش های نمایش و عملیات روی گراف آورده شده است.

[تعاریف گراف](#)

[نمایش گراف](#)

[بیمایش گراف](#)

[درخت پوشای حداقل](#)

[کاربردهای گراف](#)

تعاریف گراف

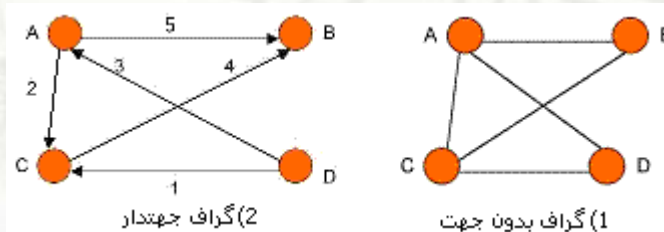
یک گراف شامل دو مجموعه است؛ مجموعه غیر تهی از گره ها یا رئوس (vertex) و مجموعه ای از یال ها (edge) که راس ها را به هم متصل می کنند.

مثال. می توان شهر های یک کشور را رئوس و جاده های بین آن ها را یال های یک گراف تصور کرد. به هر راس یا هر یال گراف نامی اختصاص داده می شود.

یک گراف تهی (null graph) گرافی است که تنها شامل راس است و مجموعه یال های آن تهی است یعنی یالی ندارد.

جهت

یک گراف می تواند به دو شکل جهتدار (directed) یا غیرجهتدار (undirected) باشد. یک گراف جهتدار گرافی است که جهت هر یال در آن تعیین شده است. در گراف جهتدار ترتیب رئوس در هر یال اهمیت دارد و یال ها با پیکان هایی از راس ابتدا به راس انتها رسم می شوند. در گراف غیرجهتدار می توان در هر دو جهت بین راس ها حرکت کرد و ترتیب راس های یال اهمیت ندارد.



(2) گراف جهتدار

(1) گراف بدون جهت

حداکثر تعداد یال ها در یک گراف جهتدار با n راس برابر است با $n \times (n-1)$.
حداکثر تعداد یال ها در یک گراف غیرجهتدار با n راس برابر است با $n \times (n-1) / 2$.

وزن

یال های گراف می توانند وزن دار (weighted) یا بدون وزن (unweighted) باشند. گرافی که یال های آن وزن باشد گراف وزن دار نامیده می شود. وزن می تواند نشان دهنده هزینه، مسافت، زمان یا هر مشخصه دیگری از یال باشد.

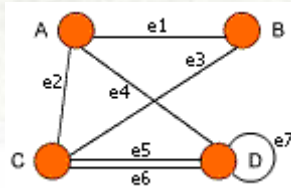
مجاورت

هر یال بوسیله یک جفت راس مشخص می شود. دو راسی که توسط یک یال به هم متصل می شوند را رئوس مجاور (adjacent) و یال را یک لبه تلاقی (incident) دو آن رو راس می نامند.

حلقه

یک حلقه (loop) یالی است که یک راس را به خودش متصل می کند. به عبارت دیگر راس ابتدا و انتهایش یکسان باشد.

مثال. در شکل زیر یال e_7 یک حلقه روی راس D است



یال های موازی

یال های موازی (parallel edges) یا چندگانه یالی هستند که رئوس یکسان را بهم مرتبط می کنند. گرافی که دارای یال های موازی باشد را گراف چندگانه (multigraph) می نامند.

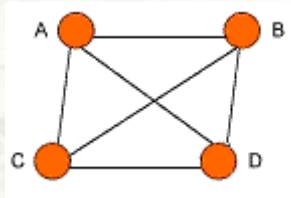
مثال. در گراف چندگانه شکل قبل یال های e_5 و e_6 که رئوس C و D را به هم متصل می کنند موازی هستند

گراف ساده

گراف بدون یال موازی و حلقه را گراف ساده (simple graph) می نامند. گراف جهندار را وقتی ساده می گویند که یال موازی نداشته باشد.

گراف کامل

یک گراف کامل (complete graph) گراف ساده ای است که هر جفت راس آن مجاور باشند یعنی هر از راس به کلیه راس های دیگر یالی وجود داشته باشد.

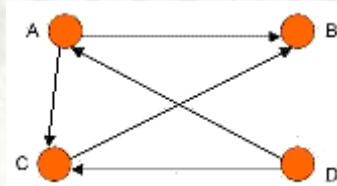


درجه

درجه (degree) هر راس توسط تعداد یال های متلاقی با راس مشخص می شود. در گراف جهندار درجه ورودی (indegree) یک راس تعداد یال هائی است که به آن راس وارد شده اند و درجه خروجی (outdegree) یک راس تعداد یال هائی است که از آن راس خارج شده اند.

راس منبع راسی است که درجه خروجی آن مثبت و درجه ورودی آن صفر باشد. راس چاه راسی که درجه ورودی آن مثبت و درجه خروجی آن صفر باشد.

مثال. در گراف زیر درجه خروجی راس A دو و درجه ورودی آن یک است. راس D منبع و راس B چاه است.



درجه گراف برابر درجه ماکزیمم رئوس گراف است.

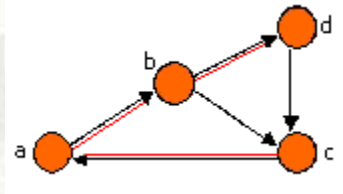
گرافی که کلیه راس های آن از یک درجه باشد گراف منتظم (regular graph) نامیده می شود. گراف مکعب گراف منتظم درجه ۳ است. در یک گراف مجموع درجات کلیه رئوس همواره عددی زوج است.

مسیر

یک مسیر (path) در گراف یک گذر از راس های متوالی در امتداد یک سری از یال ها است. راس انتهایی یک یال راس ابتدایی یال بعدی در توالی محسوب می شود. طول مسیر تعداد یال های مسیر است که در طول مسیر طی می شود. یک مسیر با طول n دارای $n+1$ راس و n یال است. در یک گراف وزن دار طول مسیر برابر مجموع وزن های یال های مسیر است.

دوراس را متصل (reachable) می گویند اگر مسیری بین آنها وجود داشته باشد. یک مسیر (simple path) ساده مسیری است که همه رئوس آن بجز احتمالاً راس شروع و پایان تکراری نباشد.

مثال. در شکل زیر یک مسیر نشان داده شده است که از راس C آغاز و به راس D ختم می شود.



دور

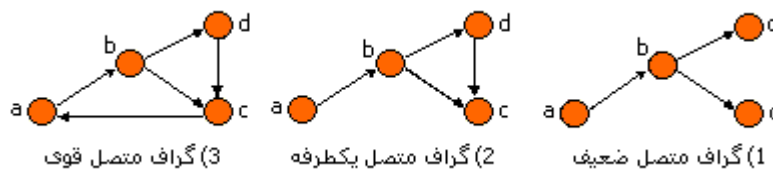
یک دور (cycle) مسیر ساده ای است که راس شروع و پایانی آن یکی باشد. گراف ساده جهنداری که دارای دور نیست را غیر مدور (acyclic) می نامند. یک دور در گراف ساده بدون جهت حداقل شامل سه یال متفاوت است که هیچ راسی در آن تکراری نیست بجز راس شروع و پایان.

اتصل

یک گراف غیرجهتدار متصل یا همبند (connected) گفته می شود اگر مسیری بین هر جفت راس آن وجود داشته باشد. یعنی هر دو راس آن متصل باشند. در گراف جهتدار چون جهت باید در نظر گرفته شود اتصال پیچیده تر است. ممکن است راس a به b متصل باشد ولی مسیری از راس b به a وجود نداشته باشد.

در گراف جهتدار ساده سه حالت برای اتصال وجود دارد:

- متصل ضعیف (weakly connected). یک گراف متصل ضعیف گرافی است که اگر جهت گراف ندیده گرفته شود متصل است.
- متصل یکطرفه (unilaterally connected). یک گراف متصل یکطرفه گرافی است که حداقل یک راس آن به هر راس دیگری متصل باشد.
- متصل قوی (strongly connected). یک گراف متصل قوی گرافی است که هر جفت راس متصل باشد.



(3) گراف متصل قوی

(2) گراف متصل یکطرفه

(1) گراف متصل ضعیف

یک گراف ساده متصل بدون دور را درخت (tree) می نامند. درخت گرافی است که فقط یک مسیر بین هر دو راس آن وجود دارد. یک درخت با n راس است دارای $n-1$ یال باشد.

نمایش گراف

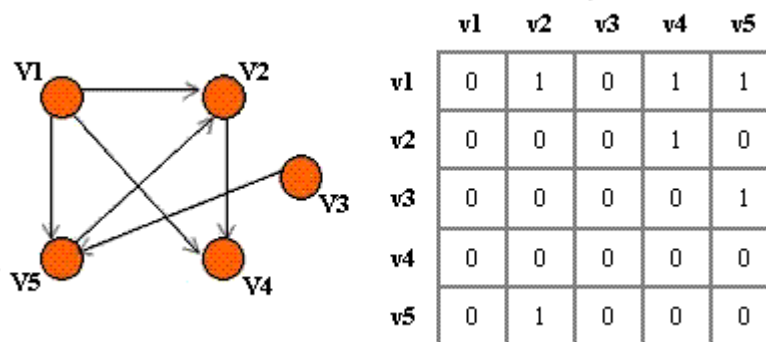
راه های متعددی برای نمایش گراف در کامپیوتر وجود دارد دو ساختمان داده پایه ای که برای نمایش گراف استفاده می شوند ماتریس مجاورت و لیست مجاورتی هستند.

ماتریس مجاورت

ماتریس مجاورت (adjacency matrix) گراف G با n راس (که رئوس آن به ترتیب از v_1 تا v_n نامگذاری شده است) یک ماتریس بیئی $n \times n$ با نم A است که در آن:

درایه a_{ij} برابر با 1 است اگر یالی از v_i به v_j وجود داشته باشد
درایه a_{ij} برابر با 0 است اگر یالی از v_i به v_j وجود نداشته باشد

مثال. گراف جهتدار زیر را با پنج راس در نظر بگیرید. ماتریس مجاورت آن در سمت راست نشان داده شده است.



همانطور که در شکل دیده می شود اگر یعنی دو راس مجاور باشند در موقعیت متناظر در ماتریس باید 1 باشد. برای مثال از رئوس مجاور با v_1 راس های v_2 ، v_4 و v_5 هستند بنابراین در سطر اول در ستون های دوم، چهارم و پنجم باید 1 قرار بگیرد.

ماتریس مجاورت برای یک گراف بدون جهت متقارن است.

در یک گراف غیر جهتدار درجه راس v_i برابر با مجموع عناصر سطر i ام در ماتریس مجاورت است. و در یک گراف جهتدار درجه خروجی راس v_i برابر مجموع عناصر سطر i ام و درجه ورودی آن برابر مجموع عناصر ستون i ام در ماتریس مجاورت است.

فضای مورد نیاز در روش ماتریس مجاورت برای نمایش یک گراف با مجموعه رئوس V برابر $O(|V|^2)$ است. اگر تعداد یال های گراف کم باشد می توان آنرا به صورت ماتریس اسپارس نمایش داد.

قضیه. اگر A ماتریس مجاورتی گراف G باشد، درایه a_{ij} در ماتریس A^k تعداد مسیرهای با طول k از راس v_i به v_j را نشان می دهد.

مثال. با ضرب ماتریس مجاورت مثال قبل در خودش ماتریس A^2 به صورت زیر بدست می آید. عدد 1 در سطر i و ستون j ماتریس نشان دهنده وجود مسیری با طول 2 از راس v_i به v_j در گراف است.

	v1	v2	v3	v4	v5
v1	0	1	0	1	0
v2	0	0	0	0	0
v3	0	1	0	0	0
v4	0	0	0	0	0
v5	0	0	0	1	0

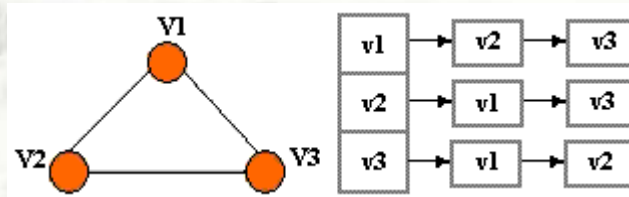
با توجه به ماتریس فوق می توان گفت که از راس v_1 با دو حرکت می توانیم به راس v_2 برویم. اگر گراف را بررسی کنیم می شویم که از v_1 به v_5 و سپس به v_2 می توانیم برویم. یعنی مسیری با طول 2 از راس v_1 به v_2 وجود دارد.

مزیت اصلی نمایش ماتریس در این است که محاسبه مسیر ها و دورها بسادگی توسط عملیات ماتریسی قابل انجام است. مجاورت بین دو راس با پیچیدگی زمان $O(1)$ تعیین می شود و اجازه رسم حلقه در گراف را می دهد. اشکال آن این است که از جنبه ظاهری گراف دور است و خواصی که بسادگی در شکل گراف نمایان است توسط ماتریس به سختی قابل رویت است. علاوه بر این ماتریس همجواری یالهای موازی را نشان نمی دهد

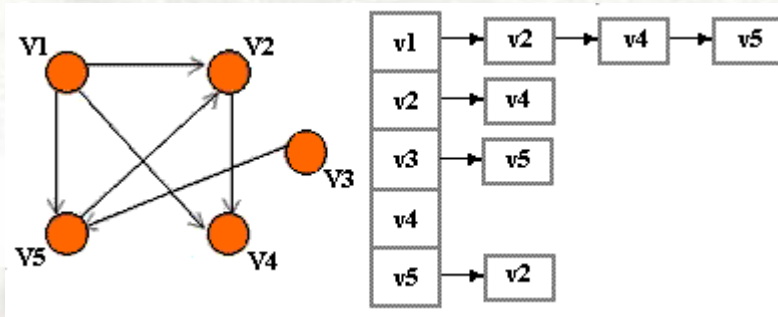
لیست مجاورت

لیست مجاورتی (adjacency list) فرم دیگر نمایش گراف در کامپیوتر است. این ساختمان داده شامل لیستی از کلیه رئوس گراف است. برای هر راس يك لیست پیوندی وجود دارد که گره های آن رئوس مجاور راس را دربر می گیرند. به عبارت دیگر لیست i حاوی رئوسی است که مجاور راس v_i است.

مثال. گراف غیرجهتدار زیر را در نظر بگیرید. لیست مجاورتی آن در سمت راست آمده است:



مثال. گراف جهتدار زیر را در نظر بگیرید. لیست مجاورتی آن در سمت راست آمده است:



درجه هر راس در يك گراف غیر جهتدار با شمارش تعداد گره های لیست پیوندی مربوط به راس در لیست مجاورتی تعیین می شود. در يك گراف جهتدار درجه خروجی هر راس با شمارش تعداد گره های لیست پیوندی مربوط به آن بدست می آید.

اگر تعداد یالها در گراف کم باشد این روش بهتر از ماتریس مجاورتی است. فضای مورد نیاز برای نمایش يك گراف با مجموعه رئوس V و مجموعه یال های E به روش لیست مجاورت برابر $O(|E|+|V|)$ می باشد.

لیست مجاورتی به روشنی طبیعت مجاورتی رئوس گراف را نشان می دهد و اغلب زمانی استفاده می شود که گراف دارای تعداد یال های نسبتاً متعادلی باشد.

لیست مجاورتی معکوس

لیست مجاورتی معکوس مشابه لیست مجاورتی ساخته می شود با این تفاوت که در لیست پیوندی i رئوسی اضافه می شود که از آنها به راس v_i یالی وارد شده باشد. تعداد گره های لیست پیوندی i ام درجه ورودی راس v_i را نشان می دهد.

برای گراف غیرجهتدار لیست مجاورتی و لیست مجاورتی معکوس مشابه می شود.

پیمایش گراف

هدف از پیمایش ای «است که کلیه رئوسی که از طریق یک راس قابل دسترس هستند را بدست آوریم. دو روش معروف برای پیمایش وجود دارد؛ جستجوی اول عمق (Deep First Search) و جستجوی اول سطح (Breadth First Search)

جستجوی اول عمق

در جستجوی اول عمق پیمایش از یک راس آغاز می شود. هر راس که پردازش یا اصطلاحاً ملاقات می شود یکی از رئوس مجاور آن که قبلاً ملاقات نشده است انتخاب می شود و پیمایش با ملاقات راس مجاور ادامه پیدا می کند. اگر راس مجاور وجود نداشت که قبلاً ملاقات نشده باشد يك سطح به عقب برمی گردد.

الگوریتم بازگشتی جستجوی اول عمق به صورت زیر است. آرایه یک بعدی Visited تعیین می کند آیا راسی قبلاً ملاقات شده است یا خیر. اگر راس v_i ملاقات شود $Visited[i]$ برابر با یک می شود.

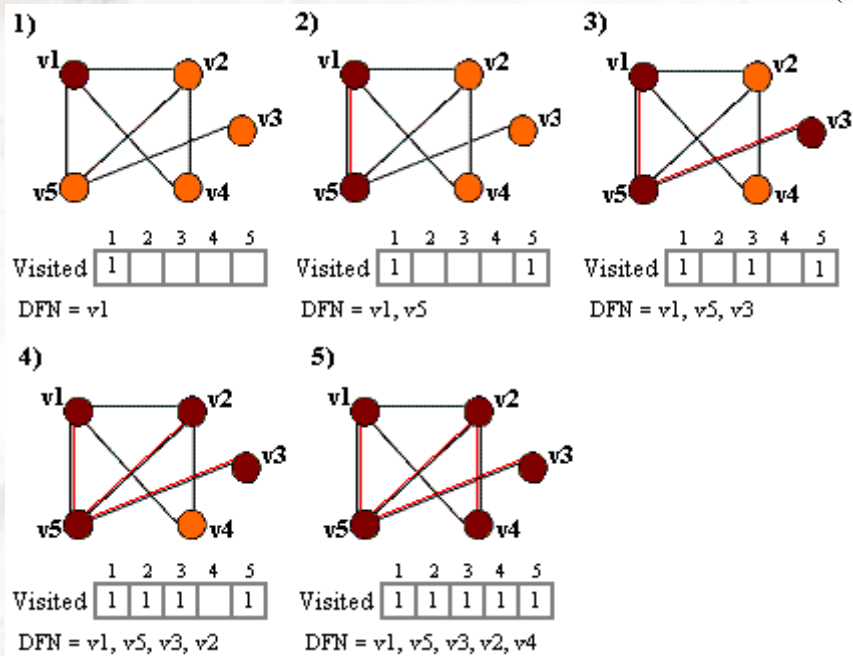
DFS (int v)

```
{
  int w
  Visited[v]:=1
  For (each vertex w adjacent to v)
    If (not visited[w]) then
      DFS(w)
    End if
  End For
}
```

ترتیب ملاقات رئوس را DFN می نامند.

برای گراف G با n راس و m یال، مرتبه اجرایی الگوریتم وقتی گراف توسط ماتریس مجاورتی نمایش داده شده باشد $O(n^2)$ و اگر از لیست مجاورتی استفاده شود $O(m)$ است.

مثال. مراحل اجرای DFS(v_1) برای یک گراف در شکل زیر نشان داده شده است.



جستجوی اول سطح

در جستجوی اول سطح پیمایش از یک راس آغاز می شود. آن راس و کلیه رئوس مجاورش ملاقات می شود سپس پیمایش از راس مجاور ادامه پیدا می کند .

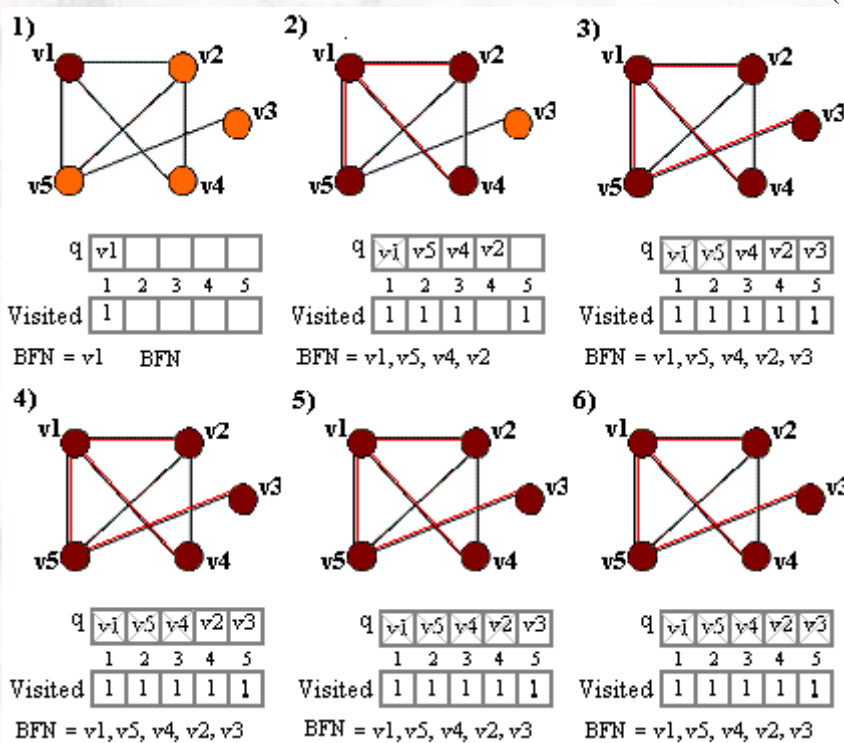
الگوریتم جستجوی اول سطح به صورت زیر است. آرایه Visited برای تعیین رئوس ملاقات شده بکار می رود. از یک صف برای نگهداشتن رئوس مجاور استفاده می شود. هر بار که راسی ملاقات می شود کلیه رئوس مجاور آن در صف اضافه می شود. پیمایش از راسی که از صف برداشته می شود ادامه پیدا می کند .

```

BFS(int v)
{
    int w
    Queue q

    Visited[v]:=1
    CreateQueue(q)
    AddQueue(q, v)
    While (not EmptyQueue(q))
        DeleteQueue(q,v)
        For (all vertex w adjacent to v)
            If (not visited[w]) then
                AddQueue(q,w)
                Visited[w]:=1
            End if
        End For
    End while
}
    
```

برای گراف G با n راس و m یال ، مرتبه اجرایی الگوریتم وقتی گراف توسط ماتریس مجاورتی نمایش داده شده باشد $O(n^2)$ و اگر از لیست مجاورتی استفاده شود $O(m)$ است .
 مثال. مراحل اجرای $BFS(v_1)$ برای یک گراف در شکل زیر نشان داده شده است .

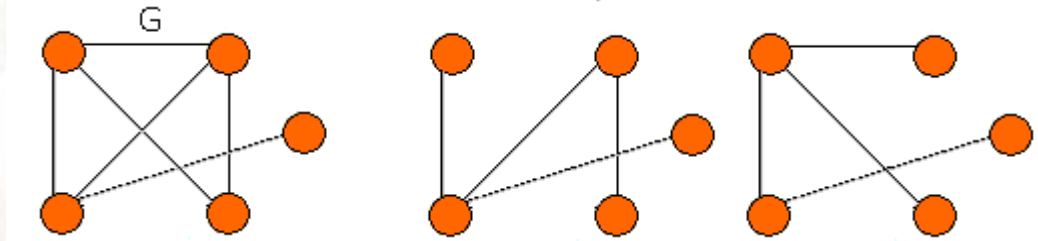


درخت پوشای حداقل

همانطور که قبلا تعریف شد درخت یک گراف متصل بدون است. یک درخت پوشا (spanning tree) زیرگرافی از گراف G است که شامل کلیه رئوس گراف G باشد و یک درخت باشد. بنابراین اگر گراف G دارای n گره باشد درخت پوشای آن دارای $n-1$ یال است.

پیمایش‌های DFS و BFS هر کدام یک درخت پوشا تولید می‌کنند. تعداد درخت‌های پوشای یک گراف کامل با n گره برابر $2n-1-1$ است.

مثال. دو درخت‌های پوشا که از پیمایش‌های DFS و BFS گراف G بدست آمده در شکل زیر نشان داده شده اند.



درخت پوشای حداقل (Minimum Spanning Tree) گراف وزن دار G ، درخت پوشائی است که مجموع وزن‌های آن حداقل باشد.

برای بدست آوردن درخت پوشای حداقل دو الگوریتم کروسکال (Kruskal) و الگوریتم پریم (Prim) را بررسی می‌کنیم.

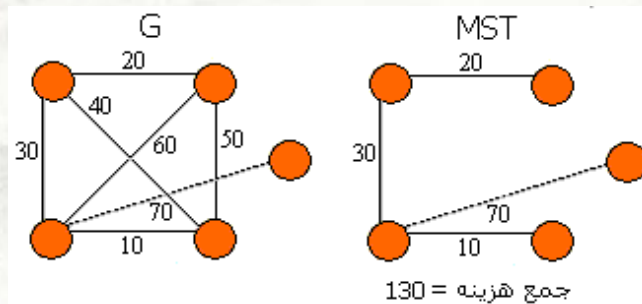
الگوریتم کروسکال

- گراف G با n راس را در نظر بگیرید. الگوریتم کروسکال به صورت زیر عمل می‌کند:
- تمام یال‌ها را به طور صعودی بر حسب وزن مرتب کنید.
 - درخت T را متشکل از گره‌های G بدون یال را ایجاد کنید.
 - عملیات زیر را $n-1$ بار تکرار کنید:
 - یک یال با حداقل وزن را به درخت T اضافه کنید به طوری که حلقه ایجاد نشود.

گاهی چند یال دارای یک وزن هستند، در این حالت ترتیب یال‌هایی که انتخاب می‌شوند مهم نیست. درخت‌های پوشای حداقل مختلفی ممکن است حاصل شود اما مجموع وزن آنها همیشه یکسان و حداقل می‌شود. پیچیدگی زمانی الگوریتم $O(mn)$ می‌شود. که m تعداد یال‌ها و n تعداد رئوس گراف G است.

الگوریتم پریم

- گراف G با n راس را در نظر بگیرید. الگوریتم پریم به صورت زیر عمل می‌کند:
- درخت تهی T را ایجاد کنید.
 - راس v از گراف را انتخاب کرده و به درخت اضافه کنید.
 - عملیات زیر را تکرار کنید تا کلیه رئوس گراف به درخت T اضافه شوند:
 - یالی که با حداقل وزن به رئوس T متصل است را پیدا کنید. یال و راس متصل به آن را به درخت T اضافه کنید به طوری که حلقه ایجاد نشود.
 - پیچیدگی زمانی الگوریتم $O(mn)$ می‌شود. که m تعداد یال‌ها و n تعداد رئوس گراف G است.
- مثل



کاربردهای گراف

گراف کاربردهای متعددی در مسائل مختلف دارد از جمله می توان کاربردهای زیر را نام برد:

- پیدا کردن کوتاهترین مسیر بین دو نقطه
- شبکه AOV برای کنترل پروژه و فعالیت ها و ارتباط بین آنها
- مسیر بحرانی