درس‌نامه الگوریتم‌های گراف

محمد جواد عبدالهی

با نظارت: دکتر بهناز عمومی

۸ تیر ۱۴۰۴

فهرست مطالب

فصل ۰

مفاهیم پایه و تعاریف

در این فصل، مفاهیم و تعاریف اولیه‌ای که در سراسر این جزوه به آن‌ها نیاز خواهیم داشت را مرور می‌کنیم.

۱.۰) تعریف گراف

به طور رسمی، یک **گراف** G یک سه‌تایی مرتب $G = (V, E, \psi)$ است که در آن:

- V یک مجموعه متناهی و ناتهی از عناصر به نام **رئوس** (Vertices) است.
- E یک مجموعه متناهی (و مجزا از V) از عناصر به نام **یال‌ها** (Edges) است.
- ψ یک **تابع وقوع** (incidence function) است که هر یال را به یک زوج (نامرتب) از رئوس نگاشت می‌دهد.

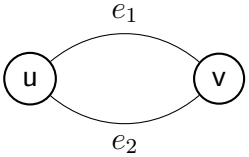
۱.۱.۰) طوقه (Loop)

اگر برای یک یال e، دو سر آن یکسان باشند، یعنی $\psi(e) = \{u, u\}$، آنگاه e را یک **طوقه** می‌نامیم.

شکل ۱: نمایش یک طوقه e روی رأس u.

۲.۱.۰) یال‌های موازی (Parallel Edges)

اگر دو یال مختلف e_1 و e_2 دو سر یکسانی داشته باشند، آن دو را **یال‌های موازی** می‌گوییم.



شکل ۲: نمایش دو یال موازی بین رئوس u و v.

۳.۱.۰) گراف ساده (Simple Graph)

گرافی که هیچ طوقه و یال موازی نداشته باشد، **گراف ساده** نامیده می‌شود.

۴.۱.۰) گراف جهت‌دار (Directed Graph)

در **گراف جهت‌دار**، تابع وقوع هر یال را به یک **زوج مرتب** از رئوس نگاشت می‌دهد: $\psi : E \to (V \times V)$.

۲.۰) نمایش گراف در کامپیوتر

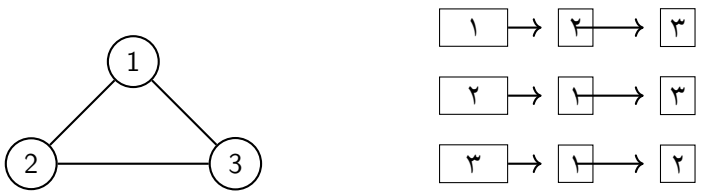
برای پیاده‌سازی الگوریتم‌های گراف، باید گراف را به شیوه‌ای کارآمد در حافظه کامپیوتر ذخیره کنیم. دو روش متداول برای این کار ماتریس مجاورت و لیست مجاورت هستند.

۱.۲.۰) ماتریس مجاورت (Adjacency Matrix)

یک ماتریس $n \times n$ (که $n = |V|$) به نام A است که در آن درایه A_{ij} برابر ۱ است اگر یالی بین رأس i و رأس j وجود داشته باشد و در غیر این صورت برابر ۰ است. این روش برای گراف‌های متراکم کارآمد است.

۲.۲.۰) لیست مجاورت (Adjacency List)

یک آرایه از لیست‌ها به اندازه n است که در آن، خانه iـ‌ام آرایه، لیستی از همسایگان رأس i را نگهداری می‌کند. این روش برای گراف‌های خلوت (که تعداد یال‌ها بسیار کمتر از n^2 است) بسیار بهینه‌تر است.



(a) نمونه گراف

(b) مجاورت لیست با نمایش

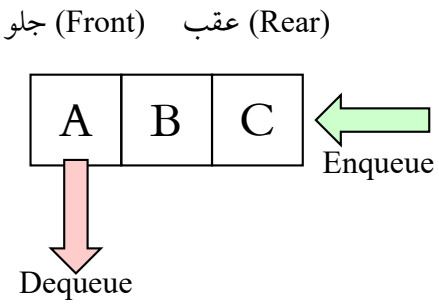
شکل ۳: یک گراف و نمایش آن با لیست مجاورت.

۳.۰) صف (Queue)

یک **صف** یک ساختمان داده خطی است که از اصل **ورودی اول، خروجی اول** (First-In, First-Out - FIFO) پیروی می‌کند. دو عمل اصلی روی صف عبارتند از:

- **Enqueue:** افزودن یک عنصر به **انتهای** صف (Rear).
- **Dequeue:** حذف یک عنصر از **ابتدای** صف (Front).

این رفتار در شکل ۴ به صورت گرافیکی نمایش داده شده است.



شکل ۴: نمایش گرافیکی یک صف و عملیات اصلی آن.

فصل ۱

تکنیک‌های اولیه پیمایش گراف

پس از آشنایی با تعاریف رسمی گراف در فصل قبل، در این فصل به سراغ دو روش بنیادین برای پیمایش رئوس و یال‌های گراف می‌رویم: جستجوی اول سطح (BFS) و جستجوی اول عمق (DFS). این الگوریتم‌ها سنگ بنای بسیاری از الگوریتم‌های پیشرفته‌تر گراف هستند. ما در این بخش، این الگوریتم‌ها را عمدتاً روی **گراف‌های ساده** توضیح می‌دهیم، اما منطق آن‌ها با استفاده از نمایش **لیست مجاورت** (Adjacency List) (بخش ۲.۲.۰)، برای گراف‌های دارای یال موازی نیز قابل تعمیم است.

۱.۱) جستجوی اول سطح (Breadth-First Search - BFS)

الگوریتم جستجوی اول سطح (BFS) یکی از ساده‌ترین و در عین حال پرکاربردترین الگوریتم‌های پیمایش گراف است. ایده اصلی این الگوریتم، شروع از یک رأس مبدأ و کاوش گراف به صورت لایه به لایه است.

۱.۱.۱) توضیح الگوریتم

الگوریتم BFS از یک رأس مشخص (مبدأ یا s) شروع به پیمایش می‌کند و فرض می‌کند گراف با یک نمایش مناسب مانند **لیست مجاورت** (Adjacency List) داده شده است. الگوریتم ابتدا تمام همسایه‌های مستقیم رأس s را ملاقات می‌کند (لایه اول) و این روند را تا زمانی که تمام رئوس قابل دسترس از s ملاقات شوند، ادامه می‌دهد. برای پیاده‌سازی این رفتار "لایه به لایه"، الگوریتم از یک **صف** (Queue) (بخش ۳.۰) استفاده می‌کند.

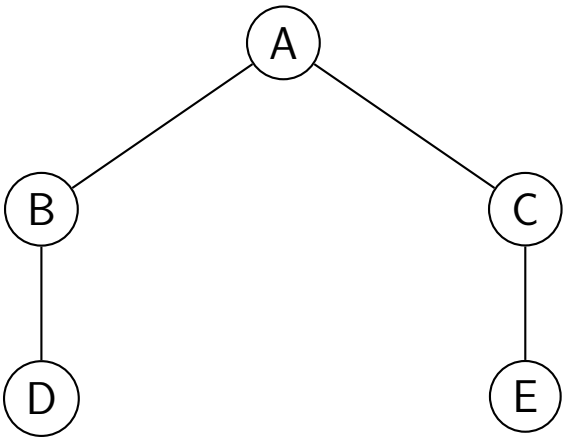
۲.۱.۱) شبه‌کد الگوریتم

الگوریتم ۱ Breadth-First Search (BFS)

```
1:  procedure BFS(G, s)
2:      Let Q be a queue
3:      Q.enqueue(s)
4:      Mark s as visited
5:      while Q is not empty do
6:          u ← Q.dequeue()
7:          Process u (e.g., print it)
8:          for each neighbor v of u do
9:              if v has not been visited then
10:                 Mark v as visited
11:                 Q.enqueue(v)
```

۳.۱.۱) مثال

یک گراف ساده و بدون جهت مانند شکل ۱.۱ را در نظر بگیرید. می‌خواهیم پیمایش BFS را از رأس A شروع کنیم.



شکل ۱.۱: یک گراف ساده و بدون جهت برای پیمایش BFS از رأس A.

ترتیب ملاقات رئوس: **A, B, C, D, E**.

۴.۱.۱) انیمیشن و ابزارهای پایتون

- **انیمیشن آنلاین:** برای درک شهودی، انیمیشن‌های تعاملی روند اجرای BFS را در وب‌سایت Visualgo مشاهده کنید.
- **اسکریپت پایتون تعاملی:** در کنار این جزوه، فایلی به نام bfs_animation.py قرار دارد. با اجرای این اسکریپت، می‌توانید روند اجرای الگوریتم BFS را روی یک گراف نمونه به صورت یک انیمیشن مشاهده کنید. این اسکریپت از کتابخانه‌های `networkx` برای منطق گراف و `matplotlib` برای تولید انیمیشن استفاده می‌کند.