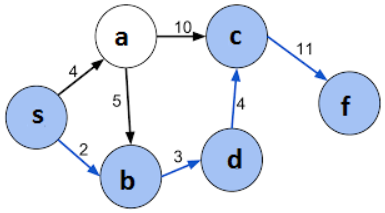
درس طراحی الگوریتم
دانشکده ریاضی. دانشگاه صنعتی خواجه نصیرالدین طوسی
کتاب مرجع: طراحی الگوریتم، جان کلاینبرگ، اوا تاردش

الگوریتم های حریصانه

۱ مسئله کوتاهترین مسیر در گراف

گراف جهت دار $G = (V, E)$ داده شده است. یک تابع طول $\ell : E \to \mathbb{R}^+$ هم داده شده که طول هر یال را مشخص میکند. دقت کنید طول هر یال یک عدد مثبت است. علاوه بر این راس $s \in V$ به عنوان راس مبدا مشخص شده است. میخواهیم از راس s طول کوتاهترین مسیر به بقیه رئوس گراف را پیدا کنیم. اگر راس x از s قابل دسترسی نیست آنگاه کوتاهترین مسیر از s به x وجود ندارد و لذا طول آن را بینهایت فرض میکنیم. دقت کنید طول مسیر برابر با مجموع طول یالهای مسیر است.

در مثال زیر کوتاهترین مسیر از s به f مشخص شده است.



۲ الگوریتم دایکسترا

یک الگوریتم حریصانه برای مسئله کوتاهترین مسیر وجود دارد که به نام مبدع آن ریاضیدان هلندی دایکسترا شناخته شده است. این الگوریتم با شروع از راس مبدا هر بار کوتاهترین مسیر از s به یک راس جدید x را پیدا می‌کند. سپس x به مجموعه S اضافه می‌شود. در واقع S در هر لحظه شامل رئوسی که کوتاهترین مسیر به آنها پیدا شده است. در شروع کار $S = \{s\}$. همینطور $d(s) = 0$. این به این معنی است که طول کوتاهترین مسیر از s به خودش صفر است. در ابتدای کار برای همه x های غیر از s داریم $d(x) = \infty$.

آرایه d در طول زمان بروز می‌شود و هر بار که یک مسیر بهتر به x پیدا شد، مقدار $d(x)$ کاهش پیدا میکند.

همانطور که گفته شد مجموعه S شامل همه رئوسی است که کوتاهترین مسیر از s به آنها محاسبه شده است. برای هر $x \in S$ مقدار $d(x)$ برابر با طول کوتاهترین مسیر از s به x است.

برای اضافه کردن راس جدید به S بدین صورت عمل میشود. مقدار $d(x)$ بصورت زیر بروزرسانی می‌شود.

$$\forall x \in V, \quad d(x) = \min_{u \in S}\{d(u) + \ell(u, x)\}$$

دقت کنید که همواره $\ell(x, x) = 0$.

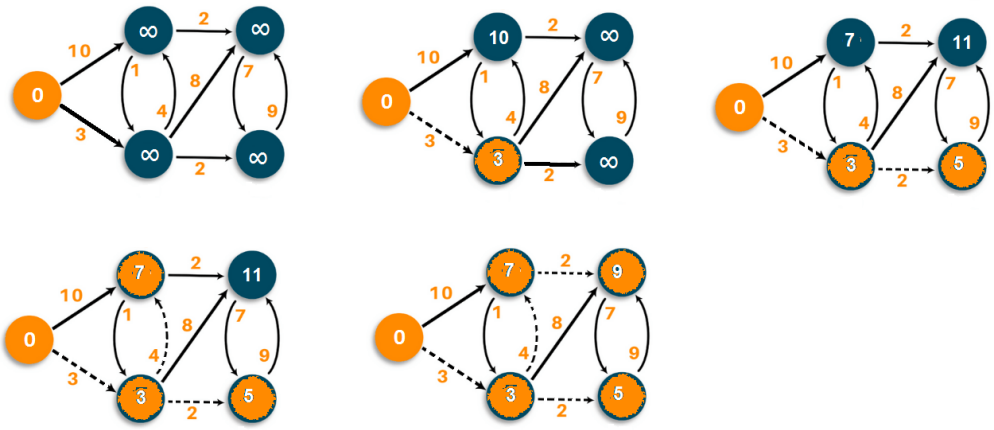
حال در بین همه رئوس خارج از S راسی که کمترین مقدار $d(x)$ را داشته باشد به S اضافه می‌شود.

فرض کنید که آخرین راسی که به S شده باشد u باشد. توجه کنید برای بروزرسانی آرایه d کافی است تنها یالهای خروجی از u را بررسی کنیم (لازم نیست در هر مرحله همه یالهای خروجی از S را بررسی کنیم!)

الگوریتم ادامه پیدا می‌کند تا زمانیکه همه رئوس قابل دسترسی از s به S اضافه شوند. به عبارت دیگر، الگوریتم ادامه پیدا می‌کند تا زمانی که آرایه d را بتوانیم بروز رسانی کنیم.

۱.۲ یک نمونه اجرای الگوریتم دایکسترا

در مثال زیر برچسب یالها طول یال مربوطه را نشان می‌دهد. رئوس مجموعه S با رنگ نارنجی مشخص شده اند. برچسب راس x مقدار $d(x)$ را نشان میدهد که در طول زمان کاهش پیدا می‌کند.



یالهایی که با خط چین مشخص شده اند یالهایی هستند که باعث شده اند یک راس جدید به مجموعه S اضافه شود. این یالها درخت کوتاهترین مسیر برای راس مبدا را تشکیل میدهند.

۲.۲ تحلیل الگوریتم دایکسترا

زمان اجرای الگوریتم: الگوریتم دایکسترا را میتوان با استفاده از ساختار داده های مناسب در زمان $O(m \log n)$ پیاده سازی کرد. دانشجوی علاقه مند میتواند جزئیات این پیاده سازی را در کتاب مرجع پیدا کند.

در هر صورت، با استفاده از ابزارهای ساده، دانشجو باید بتواند الگوریتم دایکسترا را در زمان $O(mn)$ پیاده سازی کند. الگوریتم $n-1$ مرحله دارد. در هر مرحله یک راس جدید به S اضافه میشود. هر مرحله را هم میتوان در زمان $O(m)$ پیاده سازی کرد.

تحلیل درستی الگوریتم: برای اثبات بهینه بودن جواب الگوریتم از روش استقرا استفاده میکنیم. استقرا برای مجموعه S تعریف میشود. میخواهیم نشان دهیم در هر مرحله از الگوریتم طول کوتاهترین مسیر به رئوس داخل S بدرستی محاسبه شده است.

پایه استقرا حالتی است که $S = \{s\}$. در این حالت داریم $d(s) = 0$ که بوضوح درست است.

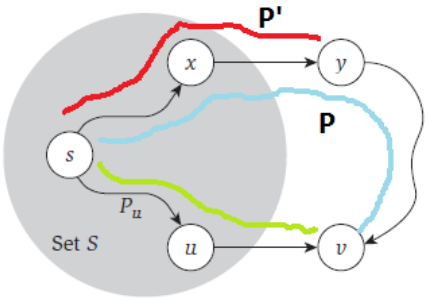
فرض استقرا: فرض کنید برای مجموعه S طول کوتاهترین مسیر بدرستی محاسبه شده است.

حکم استقرا: برای راس جدید v که به S اضافه میشود مقدار محاسبه شده $d(v)$ دقیقا برابر با طول کوتاهترین مسیر از s به v است.

از برهان خلف استفاده میکنیم. فرض کنید طول کوتاهترین مسیر از s به v کمتر از مقدار $d(v)$ باشد. دقت کنید طول کوتاهترین مسیر نمیتواند بیشتر از $d(v)$ باشد چون الگوریتم در واقع یک مسیر از s به v پیدا میکند و $d(v)$ برابر با طول این مسیر است.

فرض کنید کوتاهترین مسیر از s به v یک مسیر دیگر است و از طریق دیگری به v وصل میشود. شکل زیر را ببینید.

فرض کنید راس جدید v از طریق u به s وصل شده است. اینجا P_u با فرض استقرا کوتاهترین مسیر از s به راس u است. برهان خلف میگوید یک مسیر بهتر برای اتصال s به v وجود دارد. اسم این مسیر را P میگذاریم. این مسیر در قسمت بالای شکل به رنگ آبی مشخص شده است لاجرم باید جایی از S خارج شود. فرض کنید این مسیر برای اولین بار که از S خارج میشود از راس x (که داخل S است) وارد راس y میشود.



فرض برهان خلف میگوید

$$\ell(P) < \ell(P_u) + \ell(u, v)$$

روشن است طول آن قسمت از مسیر P که با رنگ قرمز مشخص شده، یعنی از راس s تا راس y، باید کمتر از طول P باشد. یعنی

$$\ell(P') \leq \ell(P)$$

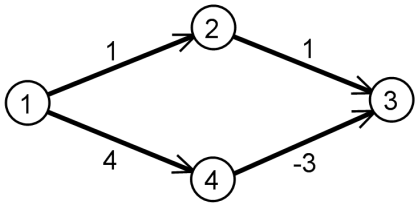
پس

$$\ell(P') < \ell(P_u) + \ell(u, v)$$

حال چگونه ممکن است که الگوریتم دایکسترا در حالیکه انتخاب بهتری داشت و میبایست در این مرحله راس y را به S اضافه میکرد، اما به جای آن راس v را اضافه کرده است که مسیری طولانی تر دارد. این با طرز کار الگوریتم متناقض است. از این درستی الگوریتم نتیجه میشود.

۳.۲ یال با طول منفی

مشاهده میشود که وجود یال با طول منفی الگوریتم دایکسترا را به اشتباه می اندازد. به نمونه زیر توجه کنید.



طول کوتاهترین مسیر از 1 به 3 برابر با 1 است اما الگوریتم آن را به اشتباه 2 محاسبه میکند. دانشجو باید تحقیق کند که چرا اثبات درستی الگوریتم دایکسترا در حضور طول منفی به مشکل بر میخورد.

۴.۲ پیاده‌سازی الگوریتم دایکسترا

برای پیاده‌سازی الگوریتم دایکسترا می‌توانیم از یک صف اولویت برای پیدا کردن راس بعدی که به مجموعه S اضافه می‌شود استفاده کرد. استفاده از هرم باینری منجر به زمان اجرای $O(m \log n)$ خواهد شد. اینجا m تعداد یالها و n تعداد رئوس گراف است. می‌توانیم از ساختار داده‌های بهتر برای صف اولویت استفاده کنیم و زمان اجرا را بیشتر کاهش دهیم. برای جزئیات بیشتر به اسلایدهای درس مراجعه کنید.