الگوریتم پیشرفته

۱۵ آذر

دکتر حامد فهیمی

دانشگاه فردوسی مشهد

دانشکده علوم ریاضی

تعاریف پایه و الگوریتم‌های تاییدکننده

الگوریتم را یک **تاییدکننده (Verifier)** برای مسئله می‌نامیم، هرگاه شرایط زیر برقرار باشد:

۱. زمان چندجمله‌ای: الگوریتم B در زمان چندجمله‌ای نسبت به ورودی‌های خود اجرا شود. خروجی آن به صورت $B(s,t)$ نمایش داده می‌شود که در آن s نمونه مسئله و t رشته‌ای موسوم به گواهی یا سند (Certificate) است.

۲. رابطه چندجمله‌ای: تابع چندجمله‌ای P وجود داشته باشد به طوری که برای هر نمونه s متعلق به مسئله X، یک رشته t وجود داشته باشد که طول آن متناسب با ورودی باشد ($|t| \leq P(|s|)$) و الگوریتم تایید کند که پاسخ "بله" است ($B(s,t) = \text{yes}$).

نکته مهم در رابطه P و NP: همواره $P \subseteq NP$ است. زیرا اگر مسئله‌ای در زمان چندجمله‌ای قابل حل باشد ($X \in P$)، قطعاً در زمان چندجمله‌ای نیز قابل تایید است. در این حالت، الگوریتم تاییدکننده می‌تواند به سادگی گواهی (t) را نادیده بگیرد و خودش مستقیماً مسئله را حل کند.

تعریف مسائل NP-Complete

مسئله X را NP-Complete می‌نامیم اگر دو شرط زیر را دارا باشد:

۱. مسئله X عضو کلاس NP باشد.

۲. برای هر مسئله Y که عضو NP است، یک کاهش چندجمله‌ای به X وجود داشته باشد ($Y \leq_P X$).

قضیه: فرض کنید X یک مسئله NP-Complete باشد. در این صورت $P = NP$ است اگر و تنها اگر X در زمان چندجمله‌ای قابل حل باشد.

استراتژی اثبات NP-Complete بودن یک مسئله جدید

برای اثبات اینکه یک مسئله جدید مانند X در دسته‌ی **NP-Complete** قرار دارد، معمولاً مراحل زیر را دنبال می‌کنیم:

1. **عضویت در NP**
   اثبات اینکه $X \in NP$ است (یعنی یک تأییدکننده با زمان چندجمله‌ای برای آن وجود دارد).
2. **انتخاب یک مسئله‌ی معلوم**
   انتخاب مسئله‌ای که قبلاً NP-Complete بودن آن ثابت شده است (مانند Y).
3. **کاهش (Reduction)**
   نشان دادن اینکه:

$$Y \leq_P X$$

یعنی هر نمونه از مسئله‌ی معلوم را بتوان در زمان چندجمله‌ای به نمونه‌ای از مسئله‌ی جدید تبدیل کرد.

انواع کاهش

- **کاهش کارپ (Karp Reduction)**
  استفاده از تنها یک پرسش از «جعبه‌ی سیاه» مسئله‌ی سخت و استفاده‌ی مستقیم از پاسخ آن.
- **کاهش کوک (Cook Reduction)**
  حالت کلی‌تر که در آن اجازه داریم چندین بار از حل‌کننده‌ی مسئله‌ی سخت برای حل مسئله‌ی خود استفاده کنیم.

بررسی مسائل کلاسیک و کاهش‌ها

۱. مسئله صدق‌پذیری (SAT) و Circuit-SAT

اولین مسئله‌ای که NP-Complete بودن آن ثابت شد، **Circuit-SAT** (صدق‌پذیری مدار) است.
طبق **قضیه کوک–لوین**، تمام مسائل کلاس NP به این مسئله کاهش می‌یابند و این مسئله ریشه‌ی اصلی بسیاری از اثبات‌های NP-Complete بودن است.

۲. مسئله خوشه (Clique)

یک **خوشه** در گراف، زیرگرافی کامل است. برای اثبات NP-Complete بودن این مسئله می‌توان از دو روش استفاده کرد:

- **کاهش از Independent Set**
با ساختن مکمل گراف (G^c).
هر مجموعه‌ی مستقل در G، یک خوشه در G^c خواهد بود.
- **کاهش از 3-SAT**
برای هر بند (Clause) در فرمول، سه رأس ایجاد می‌کنیم.
بین دو رأس یال قرار می‌دهیم اگر:
  - متعلق به دو بند متفاوت باشند، و
  - با یکدیگر متناقض نباشند (مثلاً یکی x و دیگری $\neg x$ نباشد).

۳. مسئله پوشش رأسی (Vertex Cover)

پوشش رأسی زیرمجموعه‌ای از رأس‌ها مانند V' است به‌طوری که برای هر یال (u, v) در گراف، حداقل یکی از دو رأس u یا v در V' قرار داشته باشد.
به عبارت دیگر، این مجموعه باید تمام یال‌های گراف را **پوشش** دهد.

این مسئله رابطه‌ی مستقیمی با **Independent Set** دارد:

$$\text{پوشش رأسی است } S \Leftrightarrow V - S \text{ مجموعه مستقل است } S$$

۴. مسئله دور هامیلتونی (Hamiltonian Cycle)

دوری در گراف که از هر رأس **دقیقاً یک بار** عبور کرده و در نهایت به رأس شروع بازگردد.

۵. مسئله فروشنده دوره‌گرد (TSP)

هدف یافتن دوری است که از تمام شهرها عبور کند و مجموع هزینه‌ی آن حداکثر K باشد.

کاهش از دور هامیلتونی به TSP:

- برای هر یال موجود در گراف اصلی، وزن ۱ در نظر می‌گیریم.
- برای یال‌های غیرموجود، وزن ۲ قرار می‌دهیم.

اگر گراف دارای دور هامیلتونی باشد، مسئله‌ی TSP پاسخی با طول دقیقاً n (تعداد رأس‌ها) خواهد داشت.
بنابراین وجود دوری با طول حداکثر n معادل وجود دور هامیلتونی است.

ترتیب کاهش‌های معروف:

$$Circuit\text{-}SAT \le_P SAT \le_P 3\text{-}SAT \le_P Independent\ Set \le_P Clique \le_P Vertex\ Cover$$