تئوری پیچیدگی

برنامه‌ریزی عدد صحیح

حسین کریمی

فهرست

مقدمه

- برای یک مسئله الگوریتم‌های مختلف با زمان‌های مختلف وجود دارد.
- برای محاسبه زمان اجرای الگوریتم می‌توان از روش گام‌شماری (تعداد اجرای همه خطوط) استفاده کرد.
- محاسبه تعداد اجرای همه خطوط به صورت دقیق مورد نظر نیست؛ بلکه فقط مرتبه آن خواسته می‌شود.
- در الگوریتم‌ها برای محاسبه مرتبه لازم نیست تعداد اجرای همه خطوط محاسبه شوند، بلکه تعداد اجرای جمله اصلی (مثلاً جملات داخل حلقه) مرتبه را مشخص می‌کنند
- در الگوریتم‌ها، تعداد اجرای جمله اصلی، متفاوت است و به شرایط بستگی دارد.
- پس بهترین حالت، حالت متوسط و بدترین حالت برای الگوریتم مطرح می‌شود.

نمادهای پیچیدگی الگوریتم

- بـرای تخمـین زمـان اجـرای الگـوریتم و میـزان پیچیـدگی آن، از نمادهایی به نام نمادهای مجانبی یا حدی استفاده می‌شود.
- این نمادها عبارتند از: O ، Ω، Θ و o، ω کـه هـر کـدام طبـق روش خاصی محاسبه می‌شوند و در تحلیل الگوریتم کاربرد فراوانی دارند.
- زمان اجرای یک الگوریتم Θ (g(n)) است اگر و فقط اگر بـدترین حالت زمان O (g(n)) و بهترین حالت Ω (g(n)) باشد.

۴

برنامه‌ریزی عددصحیح- تئوری پیچیدگی

نمادهای پیچیدگی الگوریتم–نماد Θ

- برای تابع g(n) ، Θ (g(n)) یک مجموعه توابع است که به شـکل زیـر تعریف می‌شود.

$$\Theta(g(n)) = \{f(n): \exists C_1, C_2, n_0 > 0, \forall n \geq n_0, 0 \leq C_1 g(n) \leq f(n) \leq C_2 g(n)\}$$

- بنابراین یک چند جمله ای درجه m که m ثابت است، از مرتبه Θ، n^m است و بطور کلی Θ دقیقا مرتبه رشد تابع را تعیین می‌کند.

۵

برنامه‌ریزی عددصحیح- تئوری پیچیدگی

نمادهای پیچیدگی الگوریتم–نماد O

- همانطور که بیـان شـد، نمـاد Θ یـک تـابع را از بـالا و پـایین بصـورت مجانبی محدود می‌کند. اما نماد O تنها یک کران بالای حدی مشـخص می‌کند.

$$O(g(n)) = \{f(n): \exists C, n_0 > 0, \forall n \geq n_0, 0 \leq f(n) \leq Cg(n)\}$$

- به عنوان مثال، $O(n^2)$ شامل همه توابعی است که رشدشـان کمتـر یـا مساوی n^2 است.

۶

برنامه‌ریزی عددصحیح- تئوری پیچیدگی

نمادهای پیچیدگی الگوریتم–نماد Ω

- نماد Ω یک کران پایین حدی برای تابع مشخص می‌کند.

$$\Omega(g(n)) = \{f(n) : \exists C, n_0 > 0, \forall n \geq n_0, 0 \leq Cg(n) \leq f(n)\}$$

- به عنوان مثال، $\Omega(n^2)$ شامل همه توابعی است که رشدشان بیشتر یا مساوی n^2 است.

رشد توابع و الگوریتم‌ها

- گوییم رشد تابع f(n) از تابع g(n) بیشتر است اگر n به سمت بی-نهایت میل کند، آنگاه f(n) زودتر به بی‌نهایت میل می‌کند. ترتیب رشد زیر برای توابع نام برده قابل اثبات است:
- $1 < \log n < n < n\log n < n^2 < n^3 < n^4 < \ldots < 2^n < 3^n < 4^n < \ldots < n! < n^n$
- ۱ یعنی بدون رشد، یعنی بـه n وابسـته نیسـت. مـثلا حلقـه ای کـه ۱۰۰۰ بار اجرا میشود رشدش ۱ است.

پیچیدگی زمانی الگوریتم‌ها

- با توجه به دانستن درستی یک الگوریتم، چگونـه سـرعت حـل آن را محاسبه کنیم؟
- عوامل متعددی برای پاسخ به این سوال وجود دارند؛ عواملی هماننـد مشخصات رایانه، کامپایلر، زبان برنامه‌نویسی مورد استفاده و غیره.
- می‌بایست روش استانداردی وجود داشته باشد که بـه مشخصـه‌هـای کامپیوتر وابسته نباشد و سرعت حل تنها به ذات الگوریتم برگردد.
- تئـوری پیچیـدگی، شـاخه ای از نظریـه محاسـبات، ریاضـی و علـوم کـامپیوتری اسـت کـه بـه بررسـی دشـواری حـل مسـائل بصـورت الگوریتمی می‌پردازد.

پیچیدگی زمانی الگوریتم‌ها–ادامه

مسائل رام‌نشدنی (Intractable)

- مسئله‌ای رام‌نشدنی است کـه حـل آن توسـط الگـوریتمی بـا زمـان چندجمله‌ای غیر ممکن باشد.
- بحث ما بر سر خود مسئله است و نه الگوریتم حل مسئله.
- ممکن است برای حل مسئله، الگوریتم‌هـای متعـددی بـا زمـان‌هـای مختلف وجود داشته باشد.
  - مثال: حل مسئله سری فیبوناچی با روش تقسیم و حل از مرتبه نمایی و بـا روش پویا از مرتبه خطی اسـت، پـس مسـئله فیبونـاچی از نـوع رام‌نشـدنی نیست.

پیچیدگی زمانی الگوریتم‌ها–ادامه

- مسائل علوم کامپیوتر سه دسته هستند:

مسائلی که الگوریتم‌هایی با پیچیدگی زمانی چندجمله ای برای آنها پیدا شده است.

- جستجوی دودویی در یک آرایه مرتب، مسئله کوتاه‌ترین مسیر، درخت پوشای کمینه

مسائلی که اثبات شده است که رام‌نشدنی هستند، یعنی الگوریتم چندجمله-ای برای آن یافت نمی‌شود.

- این مسائل نسبتا کم هستند، مثال: مسئله حساب پرزینگر Presburger Arithmetic

مسائلی که رام‌نشدنی بودن آنها ثابت نشده اسـت ولـی از طـرف دیگـر هـیچ الگوریتم چندجمله‌ای هم برای آنها پیدا نشده است.

- مسئله کوله‌پشتی و فروشنده دوره گرد

تفاوت مسائل بهینه‌سازی و تصمیم‌گیری

- خروجی مسائل بهینه‌سازی یک حل بهینه است.
- خروجی مسائل تصمیم‌گیری جواب بله یا خیر است.

- هر مسئله بهینه‌سازی را می‌توان متناظر با یک مسئله تصـمیم‌گیـری در نظر گرفت.
- در ادامه روی مسائل تصمیم‌گیری به جای مسائل بهینه سازی تمرکز می‌شود.

کلاس P

- مسائلی که برای حل آن‌ها الگوریتم یا الگوریتم‌هایی با مرتبه زمانی چندجمله‌ای $O(n^m)$ یافت شده است، کلاس الگوریتم‌های معین یا الگوریتم‌های قطعی را تشکیل می‌دهند.
- این کلاس با علامت ویژه P نمایش داده می‌شود.
- الگوریتم‌هایی که برای کامپیوترهای رایج طراحی می‌گردند الگوریتم‌های معین نامیده می‌شوند.
- در این الگوریتم‌ها نتیجه هر عمل کاملاً مشخص، معین و قطعی است.
- P کلاس کلیه مسائل تصمیم گیری است که برای حل آنها الگوریتم معین با مرتبه زمانی چندجمله‌ای وجود دارد.

۱۳

برنامه‌ریزی عددصحیح- تئوری پیچیدگی

کلاس NP

- برای معرفی کلاس NP باید ماشینی (کامپیوتری) با مشخصات جدید تعریف شود. ماشینی را تصور کنید که افزون بر توانایی اجرای دستورهای معین و قطعی قادر است برخی دستورهای نامعین را نیز اجرا کند.
- یک دستور نامعین دستوری است که نتیجه اجرای آن از قبل قابل پیش بینی نیست.
- الگوریتم هایی که برای یک کامپیوتر نامعین طراحی می‌شوند الگوریتم های نامعین نامیده می‌شوند.

۱۴

برنامه‌ریزی عددصحیح- تئوری پیچیدگی

کلاس NP - ادامه

- برای بیان اینگونه الگوریتم ها افزون بر دستورهایی که برای بیان الگوریتم های معین وجود دارد از سه دستور زیر استفاده می‌شود:

دستور انتخاب

- این تابع یکی از عناصر یک مجموعه را به دلخواه انتخاب می‌کند. عمل انتخاب برای یک کامپیوتر نامعین عمل ساده‌ای است و مرتبه زمانی آن $O(1)$ است

دستور شکست

- این تابع پایان ناموفق عملیات الگوریتم را اعلام می‌کند. این اعلام به مفهوم جواب نادرست برای مسئله تصمیم‌گیری مورد نظر است. مرتبه زمانی این تابع هم $O(1)$ است.

دستور موفقیت

- این تابع پایان موفق عملیات الگوریتم را اعلام می‌کند. این اعلام به مفهوم جواب صحیح برای مسئله تصمیم‌گیری مورد نظر می‌باشد. مرتبه زمانی این تابع نیز برای کامپیوترهای نامعین $O(1)$ است.

۱۵

برنامه‌ریزی عددصحیح- تئوری پیچیدگی

کلاس NP - ادامه

- تشخیص اول یا اول نبودن یک عدد طبیعی، مرتب کردن اطلاعات و مسئله صدق-پذیری (SAT- Satisfiability) نمونه‌هایی از مسائلی هستند که می‌توان الگوریتم‌های نامعین ساده‌ای برایشان طراحی کرد.
- در بیان یک الگوریتم نامعین ممکن است از همه دستورها به ویژه از دستورهای انتخاب، شکست و موفقیت استفاده نشود.
- هر الگوریتم معین توسط یک کامپیوتر نامعین قابل اجرا است.پس P زیرمجموعه NP است.
- محققین زیادی سعی کرده اند ثابت کنند P = NP اگر این مسئله ثابت شود مفهوم آن این است که هر مسئله‌ای که برای آن الگوریتم نامعین با مرتبه زمانی چندجمله‌ای وجود دارد می‌توان برای آن یک الگوریتم معین با مرتبه زمانی چندجمله‌ای پیدا کرد.

کلاس NP - ادامه

- یکی از مسائل عضو کلاسNP مسئله صدق‌پذیری است.
- مسئله صدق پذیری بررسی این مطلب است که آیا یک عبارت منطقی داده شده به ازای مقادیری از لفظهای تشکیل‌دهنده آن درست خواهد بود.
- طراحی یک الگوریتم نامعین که تشخیص بدهد یک عبارت داده شده صدق پذیر است یا خیر مشکل نیست.
- در سال ۱۹۷۱، استفان کوک ثابت کرد که چنانچه ثابت شود که صدق‌پذیری به کلاس P تعلق دارد، آنگاه P = NP است و بالعکس.
- با بیان این مطلب مسئله صدق‌پذیری از اهمیت بالایی برخوردار می‌شود، به گونه‌ای که کلاس‌های NP-hard و NP-complete بر محور این قضیه شکل می‌گیرند.
- افزون بر آن تلاش دانشمندانی را که در مسیر اثبات P = NP فعالیت می‌کنند جهت‌دار می‌نماید.

کلاس NP - ادامه

کاهش‌پذیری

- جهت آشنایی با مسائل کلاس‌های NP-hard و NP-complete ابتدا مفهوم کاهش‌پذیری بیان می‌شود.
- فرض کنید M1 و M2 دو مسئله باشند.
- مسئله M1 به مسئله M2 کاهش پیدا می‌کند اگر و فقط اگر چنانچه یک الگوریتم معین با مرتبه زمانی چندجمله‌ای برای M2 وجود داشته باشد، آنگاه بتوان با به کارگیری الگوریتم معین با مرتبه زمانی چندجمله‌ای که M2را حل می‌کند، M1 را با الگوریتمی معین و زمان چندجمله‌ای حل کرد.

نکته: رابطه کاهش‌پذیری یک رابطه با خاصیت تعدی است.

کلاس NP - ادامه

- مسئله M یک مسئله NP-hard است اگر و فقط اگر صدق‌پذیری بـه آن کاهش پیدا کند.
- کلاس NP-hard تنها به مسائل تصمیم‌گیری محدود نمی‌شـود. یعنــی زیر مجموعه کلاس NP نیست.
- نمونـه ای از مســائل NP-hard کـه در عـین حـال NP-complete نیست "مسئله توقف پذیری" است.
- در این مسئله هدف ساختن الگوریتمی است که مشخص کند آیـا یـک برنامه کامپیوتری داده شده هیچ وقت پایان خواهد یافت یعنــی اجـزای آن تکمیل خواهد شد و توقف خواهـد کـرد و یـا وارد حلقـه نامتنـاهی خواهد گردید.

۱۹

کلاس NP - ادامه

- ورودی الگــوریتمی بــرای مســئله توقف‌پـذیری مــی‌توانـد هــر برنامــه کامپیوتری و طیف داده‌های آن باشد و خروجی چنین الگوریتمی بله یـا خیر است.
- بله یعنی این‌که برنامه ورودی توقف خواهـد کـرد و خیـر یعنــی برنامـه ورودی در حلقه‌ای نامتناهی گیر خواهد افتاد.
- الگوریتمی که برای هر برنامه ورودی و طیف داده‌های آن جواب درستی بدهد محاسبه ناپذیر است، یعنی هیچ راه حل الگوریتمی برای آن وجود ندارد. نتیجه اینکه "مسئله توقف پذیری" به کلاس NP تعلق ندارد.

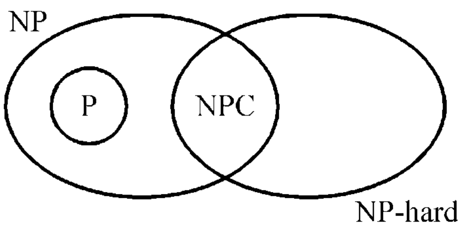
۲۰

کلاس NP - ادامه

- در واقع می‌توان این‌طور بیان کرد که کلاس دیگری به نـام NP-Hard برای مسائل وجود دارد که اگر این مسائل، مسائل تصمیم‌گیری باشـند، به کلاس NP-Complete تبدیل می‌شوند
- کــلاس NP-complete زیرمجموعــه، کلاســهای NP و NP-hard است.
- در واقع کلاس NP-complete فصل مشترک کلاسـهای NP و NP-hard را تشکیل می‌دهد.
- مسئله M یک مسئله NP-complete است اگر و فقـط اگـر M یـک مسئله NP-hard باشد و در عین حال M زیر مجموعه NP.

۲۱

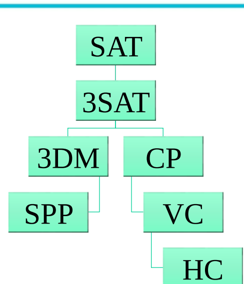
کلاس NP - ادامه



مسائل NP-Complete

- همانطور که بیان شد، ایـن مسـائل فصـل مشـترک NP-Hard و NP هستند.
- هسته اصلی این مسائل را مسئله SAT تشکیل می‌دهد.
- مسائلی که از مسئله SAT ناشی می‌شوند و بـرای اثبـات بـاقی مسـائل جهت NP-Complete بودن، به این مسائل کاهش پیدا می‌کنند

مسائل NP-Complete— ادامه



مسائل NP-Complete – ادامه

SAT

- تعریفی جزئی از مسئله SAT در قسمت‌های قبلی آمده است. مسئله‌ی صدق‌پذیری یک مسئله تصمیم است که نمونه آن یک عبارت صفر و یکی می‌باشد که فقط با AND، OR، NOT، متغیرها و پرانتز نوشته شده است. سوال این است که: آیا می‌توان به متغیرها مقادیر "درست" و "نادرست" داد تا عبارت موردنظر همواره درست باشد؟ به عنوان مثال، عبارت زیر صدق پذیر است.

$$((x)OR(y))AND((NOT(x))OR(NOT(y)))$$

- زیرا اگر x برابر یک و y برابر صفر فرض شود، پاسخ عبارت یک (یعنی درست) خواهد بود.
- اما عبارت زیر صدق پذیر نیست، زیرا نه صفر و نه یک پاسخی درست (یعنی یک) برای این عبارت می‌دهند.

$$(x)AND((NOT(x))$$

۲۵

مسائل NP-Complete – ادامه

3SAT

- این مسئله همان مسئله صدق‌پذیری است، با این تفاوت که در اینجا حتما هر قسمت دارای سه عبارت است. به عنوان مثال

$$((x)OR(y)OR(z))AND((NOT(x))OR(NOT(y))OR(NOT(z)))$$

۲۶

مسائل NP-Complete – ادامه

3-DM

- مسئله 3-DM یا تطابق سه بعدی به صورت زیر تعریف می‌شود. فرض کنید که یک مجموعه M به صورت $M \subseteq X \times Y \times Z$ وجود دارد. که در آن X، Y و Z مجموعه‌های جدا از هم هستند که دارای تعداد مساوی از عناصر نظیر p هستند. حال مسئله 3-DM به این صورت بیان می‌شود که آیا زیر مجموعه‌ای نظیر $M' \subseteq M$ وجود دارد که دارای p عنصر باشد و هیچ دو عنصر متناظر M' از یک نوع نباشند؟

۲۷

مسائل NP-Complete– ادامه

- مثال $X=\{a,b,c,d\}$ ، $Y=\{a',b',c',d'\}$ و $Z=\{a'',b'',c'',d''\}$ وجود دارند.
- حال خواهیم داشت: $M=\{(a,a',a''),...,(d,d',d'')\}$
- به عنوان مثال $M'=\{(a,a',a''),(b,b',b''),(c,c',c''),(d,d',d'')\}$ یک 3-DM از M است.

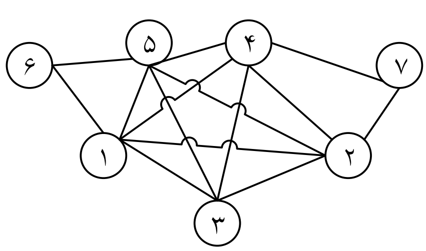
مسائل NP-Complete– ادامه

CP

- مسئله CP (Clique Problem)یا همان مسئله خوشه در یک گراف $G=(V,E)$ تعریف می‌شود. عدد صحیح n نیز داده شده به طوری که $n \le |V|$ حال سوال در مسئله خوشه این است که آیا G دارای یک زیر مجموعه $V' \subseteq V$ است که $|V'| \ge n$ و هر دو رأس موجود در V' به وسیله یک یال موجود در E به هم وصل شده باشند؟ به این زیر مجموعه یک خوشه می‌گویند.

مسائل NP-Complete– ادامه

- مثال: رئوس ۱،۲،۳،۴،۵ در گراف زیر یک خوشه را تشکیل می‌دهند.



مسائل NP-Complete – ادامه

SPP

- مسئله افراز مجموعه یا SPP (Set Partitioning Problem) همانطور که بیان شد یک مسئله NP-Complete هست. در این مسئله فرض می‌شود که یک مجموعه متناهی A و یک اندازه $S(a) \in Z^{+}$ به ازای هر یک از عناصر $a \in A$ وجود دارد. حال سوال این است که آیا زیرمجموعه $A' \subseteq A$ موجود است که $\sum_{a \in A-A'} S(a) = \sum_{a \in A'} S(a)$ باشد؟

به عنوان مثال فرض کنید که $A = \{a_1, a_2, a_3, a_4, a_5\}$ و $S(a_1) = 2$ ، $S(a_2) = 1$ ، $S(a_3) = 4$، $S(a_4) = 3$ و $S(a_5) = 4$ در این صورت $A' = \{a_4, a_5\}$ یک افراز از A است.

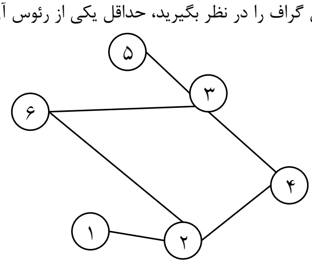
مسائل NP-Complete – ادامه

VC

- مسئله VC (Vertex Covering)، مسئله پوشش رئوس در یک گراف $G = (V, E)$ تعریف می‌شود. عدد صحیح n نیز داده شده به طوری که $n \le |V|$ حال سوال در مسئله پوششاین است که آیا G دارای یک زیر مجموعه $V' \subseteq V$ است که $|V'| \le n$ و و اگر هر $\{u, v\} \in E$ ، آنگاه $u \in V'$ یا $v \in V'$ باشد؟ به این زیر مجموعه یک پوشش رئوس می‌گویند.

مسائل NP-Complete – ادامه

- مثال: در گراف زیر رئوس ۲و۳ یک پوشش رئوس برای گراف است. زیرا هر یال در این گراف را در نظر بگیرید، حداقل یکی از رئوس آن ۲ یا ۳ است.



مسائل NP-Complete – ادامه

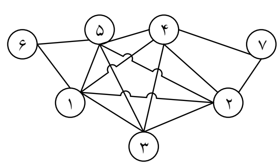
HC

- مسئله Hamiltonian Cycle)VC)، یا مسئله دور همیلتونی در یک گراف $G=(V,E)$ تعریف می‌شود. مسئله به این صورت تعریف مـی‌شـود که آیا گراف G دارای دور همیلتونی است؟
- دور همیلتونی دوری است که از یک رأس شروع شود و با گذشت فقـط یک بار از تمامی رئوس به همان رأس اول باز گردد.

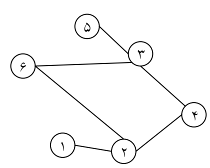
۳۴

مسائل NP-Complete – ادامه

- به عنوان مثال، گراف شکل الف دارای دور همیلتـونی ۱،۲،۳،۴،۵،۶،۷ است، اما گراف شکل ب هیچ دور همیلتونی ندارد.



الف



ب

۳۵

مطالعات بیشتر

- Cook, S. A. (1985). A taxonomy of problems with fast parallel algorithms. *Information and control*, *64*(1), 2-22.
- Cook, S. (2000). The P versus NP problem. In *Clay Mathematical Institute; The Millennium Prize Problem*.
- Kruskal, C. P., Rudolph, L., & Snir, M. (1990). A complexity theory of efficient parallel algorithms. *Theoretical Computer Science*, *71*(1), 95-132.
- Cook, S. A. (1983). An overview of computational complexity. *Communications of the ACM*, *26*(6), 400-408.
- Cook, S. A. (1973). A hierarchy for nondeterministic time complexity. *Journal of Computer and System Sciences*, *7*(4), 343-353.

۳۶

تمرین

- اثبات کنید که مسئله خوشه (CP) یک مسئله NP-Complete است. (راهنمایی: از روش کاهش مسئله به یک مسئله NP-Complete استفاده کنید.)
- برنامه‌ریزی عددصحیح NP-Hard است یا NP-Complete؟ چرا؟
- پیچیدگی زمانی الگوریتم سیمپلکس در حل برنامه‌ریزی خطی ریاضی چیست؟