پیچیدگی ارتباطی

امید یعقوبی

دانشکده علوم ریاضی، دانشگاه صنعتی شریف

بهمن ۱۴۰۱

چکیده

در سال ۱۹۷۹ مفهوم پیچیدگی ارتباطی اولین بار توسط اندرو یاو [1] معرفی شد. این مفهوم از بخش‌های فعال در نظریه‌ی پیچیدگی است و در آن دو محاسبه‌گر می‌خواهند با ارتباط باهم یک مساله مشترک را حل کنند و برای ما اینجا تنها بیت‌هایی مهم است که بین این دو ردوبدل می‌شود. در کنار جذابیت این مفهوم از نگاه نظری، کاربردهای آن در VLSI ، اتوماتون‌های متناهی، نظریه‌ی محاسبه و بسیاری از حوزه‌های دیگر مورد توجه است. این مفهوم به ما ابزارهای قدرتمندی برای نشان دادن کران‌پایین در مدل‌های محاسباتی مختلف می‌دهد. در این سمینار من مدل ساده‌ی دو محاسبه‌گره و قطعی را به عنوان مدل مبنا قرار داده‌ام و پس از آن به مفهوم و تکنیک‌های دادن کران‌بالا و کران‌پایین برای مساله‌های مختلف روی این مدل مبنا پرداخته‌ام. در اینجا کلاس پیچیدگی P^{cc} را معرفی می‌کنیم و یکی از مساله‌های درون این کلاس را برای آشنایی ابتدایی با آن بررسی می‌کنیم. سپس به یکی از کاربردهای این نظریه در ماشین‌های تورینگ می‌پردازیم.

کلمات کلیدی: Communication complexity,Complexity theory

۱ مقدمه

پیچیدگی ارتباطی [2] مطالعه‌ی سناریوهایی است که در آن چند نفر یا چند محاسبه‌گر می‌خواهند با ارتباط باهم تابعی را که وابسته به ورودی هر یک از آن‌هاست محاسبه کنند و آن چه برای ما در اینجا اهمیت دارد تعداد بیت‌هایی است که برای محاسبه تابع باهم رد و بدل می‌کنند.

اندرو یاو در سال ۱۹۷۹ [۱] مدل ساده‌ای را معرفی کرد که در آن دو نفر، آلیس و باب، می‌خواهند با ارتباط با یکدیگر در یک کانال دودویی تابع $f(x,y)$ را محاسبه کنند به شکلی که x ورودی آلیس بوده و y ورودی باب است. همین مدل مقدماتی برای پیچیدگی ارتباطی بیشتر مفاهیم و مساله‌های اصلی این نظریه را پوشش می‌دهد و در خیلی از موارد می‌توان نتایج آن را برای استفاده در مدل‌های پیچیده‌تر گسترش داد.

کاربردهای پیچیدگی ارتباطی فراتر از آن است که در نگاه اول به نظر می‌رسد. برای مثال در VLSI ها [۲] ، اتوماتون‌های متناهی [3] [۳] ، ماشین‌های تورینگ [4] [۴] [۳] ، در پیچیدگی اثبات [5] [۵] ، درخت‌های

[1] Andrew Chi-Chih Yao

[2] Communication complexity

[3] DFA

[4] Turing machines

[5] Proof complexity

تصمیم [6] [۳] و بسیاری از بخش‌های مهم دیگر در علوم‌کامپیوتر.

تحقیقات در پیچیدگی ارتباطی در چند مسیر متفاوت دنبال می‌شود، از مهم‌ترین آن‌ها می‌توان به موارد زیر اشاره کرد:

- مطالعه‌ی خصوصیت‌های مدل دوــ محاسبه‌گر
- گسترش مدل اصلی به شیوه‌های مختلف
- یافتن کاربردهای پیچیدگی ارتباطی در حوزه‌های پژوهشی دیگر

در این سمینار مدل *قطعی دو محاسبه‌گر* مدل مبنا قرار داده شده و مفاهیم و تکنیک‌های اصلی پیچیدگی ارتباطی به شکل مثال محور مورد بررسی قرار خواهند گرفت. همچنین یکی از کاربردهای پیچیدگی ارتباطی در ماشین‌های تورینگ را به عنوان نمونه‌ای از کاربردهای بی‌انتهای این نظریه خواهیم دید.

۲ مدل مبنا

بگذاریم X, Y, Z سه مجموعه‌ی متناهی دل‌خواه باشند. حال بگذاریم f تابعی دل‌خواه به شکل

$$f : X \times Y \to Z$$

باشد. دو محاسبه‌گر [7] به نام‌های آلیس و باب می‌خواهند تابع $f(x, y)$ را به دست بیاورند. [8] به این هدف دو محاسبه‌گر باهم مکالمه می‌کنند و مکالمه‌ی آن‌ها در کانالی دودویی انجام می‌شود. به بیان ساده‌تر هر پیام ارسالی بین دو محاسبه‌گر در قالب یک بیت می‌باشد. از آن‌جا که تنها تعداد بیت‌های مخابره شده بین این دو برای ما مهم است، قدرت محاسباتی هر دو محاسبه‌گر در دو طرف کانال ارتباطی را بی‌نهایت فرض می‌کنیم. آلیس و باب برای به دست آوردن جواب براساس یک قاعده مشترک عمل می‌کنند که به این قاعده *پروتکل* [9] می‌گوییم. به بیانی این قاعده براساس مکالمه‌ای که تا به حال بین آلیس و باب شکل گرفته در ابتدا تعیین می‌کند که نفری که بیت بعدی را می‌فرستد کیست و براساس همان مکالمه و ورودی فرستنده تعیین می‌کند که این بیت چه بیتی است. در ادامه می‌خواهیم تعریفی رسمی از پروتکل بدهیم.

تعریف ۱. *یک پروتکل قطعی* [10] π *یک سه تایی به شکل* (A, B, N) *است به شکلی که*

$$\begin{aligned} A: &\quad X \times \{0,1\}^* \to \{0,1\} \\ B: &\quad Y \times \{0,1\}^* \to \{0,1\} \\ N: &\quad \{0,1\}^* \to \{A, B, STOP\} \end{aligned}$$

[6] Ordered binary decision diagrams

[7] Two party

[8] در برخی از منابع قدیمی‌تر مانند [۱] به دست آوردن جواب تابع تنها برای یک محاسبه‌گر کافی است. ولی در منابع مدرن و با توجه به گسترش این نظریه و تلاش برای سازگاری آن با مفاهیم جدید داده شده مانند درخت پروتکل و یا مدل‌های گسترش یافته که توابع غیربولی را محاسبه می‌کنند و بیش از یک محاسبه‌گر در آن‌ها وجود دارد، به دست آوردن جواب تابع برای هر دو محاسبه‌گر لازم است. این تعریف مدرن‌تر از پیچیدگی ارتباطی در بیشتر منابعی مانند [۳] [۶] [۷] تعریف مبنا قرار داده می‌شود.

[9] Protocol

[10] Deterministic protocol

به صورت شهودی تابع A تابع تولید مکالمه‌ی آلیس است که پارامتر اول آن ورودی آلیس و پارامتر دوم آن مکالمه‌ی انجام شده تا به اینجا است. همچنین خروجی آن بیتی است که آلیس برای باب می‌فرستد. به تقارن همین تابع B مکالمه‌ی باب را تولید می‌کند و در نهایت تابع N نفربعدی را مشخص می‌کند. به شکل دقیق‌تر تابع N با توجه به تنها پارامتر آن که مکالمه‌ی تولید شده تا به حال است، خروجی A یا B یا $STOP$ دارد. اگر خروجی آن A بود ارسال کننده‌ی بیت بعدی آلیس و اگر B بود ارسال کننده‌ی بیت بعدی باب و اگر $STOP$ بود مکالمه پایان می‌یابد.

تعریف ۲. *به بیت‌های جابه‌جا شده تا انتهای مکالمه تاریخچه‌ی مکالمه* [11] *و به بیت‌های جابه‌جا شده تا هر نقطه از مکالمه (ممکن است در میان مکالمه باشیم) نیم‌ـ تاریخچه‌ی مکالمه* [12] *می‌گوییم.*

مرسوم است که از عبارت تاریخچه‌ی مکالمه هم برای نیم‌ـتاریخچه‌ی مکالمه و هم برای تاریخچه‌ی مکالمه استفاده شود. از این رو می‌توان با توجه به محتوا به تمایز بین این دو پی برد. ممکن است در ادامه‌ی این نوشتار نیز گاهی از پیشوند «نیم‌ـ» در نیم‌ـتاریخچه‌ی مکالمه صرف نظر کنیم.

توجه ۱.۲. *پارامتر دوم در توابع A و B و پارامتر اول در تابع N در پروتوکل قطعی π همان نیم‌ـ تاریخچه‌ی مکالمه هست.*

توجه ۲.۲. *با این تعاریف نیازی به ارسال بیت مخصوصی برای اتمام مکالمه نیست. چراکه هر دو محاسبه‌گر به تاریخچه‌ی مکالمه و پروتکل دسترسی دارند. مکالمه پایان میابد اگر که مقدار N به ازای تاریخچه‌ی مکالمه $STOP$ باشد. در نتیجه هر دو محاسبه‌گر شرط پایان محاسبه را به صورت محلی* [13] *بررسی می‌کنند.*

پروتکل داده شده‌ی $\pi = (A, B, N)$ را در نظر بگیریم. اگر برای هر $(x, y) \in X \times Y$ تاریخچه‌ی مکالمه‌ی آن $s_\pi(x, y)$ باشد به شکلی که $N(s_\pi(x, y)) = STOP$ و توابع A و B سازگار باشند، به این معنا که $A(x, s_\pi(x, y)) = B(y, s_\pi(x, y))$ آن‌گاه $A(x, s_\pi(x, y)) = B(y, s_\pi(x, y))$ را خروجی پروتکل π به ازای (x, y) می‌نامیم و آن را با $\pi(x, y)$ نشان می‌دهیم. مشاهده شود که $A(x, s_\pi(x, y))$ و $B(y, s_\pi(x, y))$ به صورت محلی محاسبه می‌شود و برای محاسبه‌گر دیگر فرستاده نمی‌شود. چراکه $N(s_\pi(x, y)) = STOP$ است.

تعریف ۳. *فرض کنیم π یک پروتکل باشد. اگر برای هر $(x, y) \in X \times Y$ خروجی π برابر $f(x, y) \in Z$ باشد می‌گوییم π تابع f را محاسبه می‌کند.*

مثال ۳.۲. *تابع $\{0, 1\}$ بیتی $OR(x, y) \in \{0, 1\}$ به شکلی که $x, y \in \{0, 1\}^n$ باشد را همان تابع OR بیتی در نظر بگیرید. یک پروتکل بدیهی برای آن تمام بیت‌های آلیس را به باب فرستاده و باب نیز $OR(x, y)$ را در قالب یک بیت برای آلیس می‌فرستد. این پروتکل به ازای هر ورودی به طول n دقیقا $n + 1$ بیت مکالمه انجام می‌دهد. فرض کنیم $n = 2$ و $x = 00$ و $y = 01$ باشد. آن‌گاه اگر $\pi = (A, B, N)$ پروتکل*

[11] Communication history

[12] Partial communication history

[13] Local

توصیف شده باشد، داریم:

$$
\begin{array}{lll}
(1) & N(\epsilon) = A & \Rightarrow \quad A(00, \epsilon) = 0 \\
(2) & N(0) = A & \Rightarrow \quad A(00, 0) = 0 \\
(3) & N(00) = B & \Rightarrow \quad B(01, 00) = 1 \\
(4) & N(001) = STOP & \Rightarrow \quad A(00, 001) = B(01, 001) = 1 = \pi(00, 01)
\end{array}
$$

در مثال پیش اگر کمی هوشمندانه عمل می‌کردیم می‌توانستیم به ازای هر n با 2 بیت مکالمه تابع OR را محاسبه کنیم. چراکه کافیست آلیس $(x_1 \vee x_2 \vee \cdots \vee x_n)$ را در قالب یک بیت برای باب بفرست و باب هم فصل منطقی نتیجه فرستاده شده توسط آلیس را با ورودی خودش برای آلیس بفرستد و به این ترتیب هر دو جواب OR را با دو بیت مکالمه خواهند داشت. ولی روش بدیهی گفته شده ما را به یک کران بالای بدیهی برای محاسبه هر تابع رهنمون می‌کند که در ادامه به آن خواهیم پرداخت. پیش از آن می‌خواهیم مفهوم هزینه‌ی پروتکل و پیچیدگی ارتباطی تابع f را به صورت رسمی تعریف کنیم.

تعریف ۴. *برای پروتکل π و ورودی $(x, y) \in X \times Y$ بگذاریم تاریخچه‌ی مکالمه $s_\pi(x, y)$ و اندازه‌ی آن $|s_\pi(x, y)|$ باشد. حال هزینه‌ی پروتکل π با $cost(\pi)$ نشان می‌دهیم و برابر است با*

$$
cost(\pi) := \max_{\{(x,y) \in X \times Y : |x| = |y| = n\}} |s_\pi(x, y)|
$$

به بیانی $cost(\pi)$ تابعی از n است و طول تاریخچه‌ی مکالمه روی بدترین ورودی به طول n برای پروتکل π می‌باشد. حال می‌توانیم پیچیدگی ارتباطی را برای تابع f تعریف کنیم.

تعریف ۵. *پیچیدگی ارتباطی قطعی تابع f را با $D(f)$ نشان می‌دهیم و برابر است با هزینه‌ی بهترین پروتکل π که تابع f را محاسبه می‌کند:*

$$
D(f) := \min_{\{\pi : \pi \text{ computes } f\}} cost(\pi)
$$

حال می‌توانیم با استفاده از ایده‌ی گفته شده برای محاسبه‌ی تابع OR یک کران بالا برای هر تابع f دل‌خواه بدهیم.

قضیه ۴.۲. *برای هر تابع دل‌خواه*

$$
f : X \times Y \to Z
$$

داریم

$$
\begin{array}{ll}
(1) & D(f) \leq \lceil \log X \rceil + \lceil \log Z \rceil \\
(2) & D(f) \leq \lceil \log Y \rceil + \lceil \log Z \rceil
\end{array}
$$

اثبات. (1) را نشان می‌دهیم و به تقارن آن (2) به دست می‌آید. کافیست پروتکلی بدهیم که f را محاسبه کند و هزینه‌ی آن همواره $\lceil \log X \rceil + \lceil \log Z \rceil$ باشد. پروتکل π روی ورودی $(x, y) \in X \times Y$ به این شکل عمل می‌کند که ابتدا آلیس تمام بیت‌های $x \in X$ را که در $\lceil \log X \rceil$ قابل نمایش است برای باب ارسال می‌کند. سپس باب $f(x, y) \in Z$ را محاسبه کرده و آن را که در $\lceil \log Z \rceil$ بیت قابل نمایش است

برای آلیس ارسال می‌کند. به این ترتیب هر دو محاسبه‌گر با $\lceil \log X \rceil + \lceil \log Z \rceil$ بیت مکالمه جواب $f(x,y)$ را به دست می‌آورند. به این ترتیب حکم اثبات می‌شود. ■

برای توابع $f : X \times Y \to Z$ که در ادامه خواهیم داشت، اگر به چیستی X و Y و Z اشاره‌ای نشده بود، قرارداد می‌کنیم که $X = Y = \{0,1\}^n$ و $Z = \{0,1\}$.

۳ کران بالا

برای آن‌که کران بالای $g(n)$ را برای تابع f نشان دهیم، باید نشان دهیم که $D(f) \leq g(n)$ و برای این کار کافیست پروتکلی بدهیم که همواره در بدترین حالت هزینه‌ی آن از $g(n)$ بیشتر نخواهد شد. نحوه‌ی دادن کران بالا به شدت وابسته به تکنیک‌های طراحی پروتکل در پیچیدگی ارتباطی است. این تکنیک‌ها را در ادامه در قالب چند مثال بررسی می‌کنیم.

۱.۳ PARITY

یک مثال کلاسیک در پیچیدگی ارتباطی محاسبه‌ی تابع $PARITY$ است. ابتدا تعریف می‌کنیم:

$$PARITY(x,y) = \left(\sum_{i=1}^{n} x_i + \sum_{i=1}^{n} y_i \right) \mod 2$$

می‌خواهیم نشان دهیم که $D(PARITY) \leq 2$ برای این کار کافیست یک پروتکل با هزینه‌ی 2 برای آن بدهیم. پروتکل ساده‌ای برای این مساله وجود دارد. کافیست آلیس $\left(\sum_{i=1}^{n} x_i \right) \mod 2$ را در قالب یک بیت برای باب بفرستد و باب هم مقدار $PARITY$ از ورودی خودش و بیت ارسالی آلیس محاسبه کرده و در قالب یک بیت برای آلیس بفرستد. به روشنی هزینه‌ی این پروتکل همواره برابر 2 است. در نتیجه $D(PARITY) \leq 2$.

تابع $PARITY$ مثال ساده‌ای است که می‌توان به سادگی برای آن کران پایین هم داد. به همین رو ادعا می‌کنیم که $D(f) \geq 2$.

اثبات. با برهان خلف فرض کنیم چنین نباشد. آن‌گاه پروتکل π وجود دارد که تابع $PARITY$ را با هزینه‌ی کمتر مساوی 1 محاسبه می‌کند. در نتیجه دست‌بالا یک بیت از یک محاسبه‌گر به دیگری ارسال می‌شود. بدون از دست دادن عمومیت فرض کنیم آلیس تنها بیت ارسالی را مخابره می‌کند. بگذاریم این بیت b باشد. از آن‌جا که π تابع $PARITY$ را محاسبه می‌کند، پس

$$A(x,b) = A(x, A(x,\epsilon)) = PARITY(x,y)$$

در نتیجه $PARITY(x,y)$ تابعی از y نیست. پس مقدار آن وابسته به y نیست. ولی با تغییر یک بیت از y مقدار $PARITY(x,y)$ تغییر می‌کند. این تناقض است. در نتیجه $D(PARITY) \geq 2$.

نتیجه ۱.۳. $D(PARITY) = 2$

اثبات. از آن‌جا که $D(PARITY) \geq 2$ و $D(PARITY) \leq 2$ پس $D(PARITY) = 2$.

۲.۳ MAJORITY

برای رشته‌ی $w \in \Sigma^*$ بگذاریم $\#_1 w$ تعداد رخدادهای 1 در رشته‌ی w باشد. همچنین برای رشته‌های $x, y \in \Sigma^*$ بگذاریم $x.y$ الحاق این دو رشته باشد. حال $MAJORITY(x, y) = 1$ است اگر که $\#_1(x.y) \geq \#_0(x.y)$ و در غیر این صورت $MAJORITY(x, y) = 0$ است.

ادعا می‌کنیم $D(f) \leq O(\log n)$. برای اثبات این ادعا کافیست یک پروتکل مناسب بدهیم. پروتکل π را به این شکل تعریف می‌کنیم که در ابتدا آلیس $\#_1 x$ را که در $\lceil \log n \rceil$ بیت قابل نمایش است برای باب بفرستد و باب هم $\#_1 y$ را که در $\lceil \log n \rceil$ بیت قابل نمایش است برای آلیس بفرستد. حال هر دو می‌توانند به صورت محلی

$$\#_1(x.y) = \#_1 x + \#_1 y$$

را محاسبه کنند. همچنین از آن‌جا که $X = Y = \{0, 1\}^n$ پس

$$\#_0(x.y) = n - \#_1(x.y)$$

درنتیجه هر دو می‌توانند مقدار $MAJORITY(x, y)$ را بیابند. تعداد بیت‌های ردوبدل شده بین این دو در این پروتکل $2.\lceil \log n \rceil$ است. در نتیجه $D(MAJORITY) \leq O(\log n)$.

دو مساله‌ای که تا به اینجا بررسی کردیم پروتکل‌های بسیار ساده‌ای داشتند. به عنوان آخرین مثالی که برای کران بالا می‌دهیم، می‌خواهیم پروتکلی را بررسی کنیم که ایده‌ی طراحی آن خیلی بدیهی نیست.

۳.۳ MEDIAN

برای هر زیرمجموعه $[s] \subseteq \{1, 2, \ldots, n\}$ بردار مشخصه‌ی [۱۴] آن را با s نشان می‌دهیم به شکلی که $s \in \{0, 1\}^n$ به شکلی که اگر s_i بیت i ام s باشد آن گاه

$$i \in [s] \Leftrightarrow s_i = 1$$

به بیان ساده s رشته‌ی بیتی است به طول n و متناظر مجموعه‌ی $[s]$ به شکلی که بیت i ام آن 1 است اگر و تنها اگر $i \in [s]$ باشد. برای مثال اگر $n = 3$ آن‌گاه مجموعه‌ی متناظر رشته‌ی بیتی $s = 101$ برابر $[s] = \{1, 3\}$ است.

حال مساله‌ی $MEDIAN$ به این شکل تعریف می‌شود که برای ورودی

$$(x, y) \subseteq \{0, 1\}^n \times \{0, 1\}^n$$

به شکلی که

$$\begin{aligned} [x] &\subseteq \{2, 4, \ldots, 2n\} \\ [y] &\subseteq \{1, 3, \ldots, 2n - 1\} \end{aligned}$$

می‌خواهیم میانه‌ی $[x] \cup [y]$ را محاسبه کنیم. به بیان ساده $[x]$ زیرمجموعه‌ای از اعداد زوج بین 2 تا $2n$ است و $[y]$ زیرمجموعه‌ی اعداد فرد بین 1 تا $2n - 1$ است و در نتیجه $[x] \cup [y]$ زیرمجموعه‌ای از اعداد بین 1 تا $2n$ است و در این مساله ما می‌خواهیم میانه‌ی این زیرمجموعه را محاسبه کنیم.

از قضیه‌ی ۲ می‌توان نتیجه گرفت که $D(MEDIAN) \leq n + \lceil \log 2n \rceil$. حال ادعا می‌کنیم که $D(MEDIAN) \leq O(\log^2(n))$. به بیانی یک کران‌بالای بسیار بهینه‌تر برای این مساله وجود دارد. برای اثبات این ادعا کافیست پروتکل مناسب برای آن را توصیف کنیم.

[14] Characteristic vector

ایده‌ی طراحی این پروتکل به روش جست‌وجوی دودویی بسیار شبیه است. ادعا می‌کنیم که برای هر بازه‌ی $[i,j]$ دل‌خواه، اگر آلیس و باب هر دو بدانند که میانه در این بازه قرار دارد، می‌توانند با $O(\log n)$ بیت مکالمه بازه را به نصف کاهش دهند. برای این هدف آلیس و باب هر دو

$$mid = \lfloor \frac{i+j}{2} \rfloor$$

را به صورت محلی محاسبه می‌کنند. سپس آلیس $L_x = |[i, mid] \cap [x]|$ و $R_x = |[mid+1, j] \cap [x]|$ را محاسبه می‌کند و برای باب می‌فرستد و باب $L_y = |[i, mid] \cap [y]|$ و $R_y = |[mid+1, j] \cap [y]|$ را محاسبه می‌کند و برای آلیس می‌فرستد. به بیانی L_x تعداد المان‌هایی در $[x]$ است که سمت چپ mid قرار دارد و R_x تعداد المان‌هایی در $[x]$ است که سمت راست mid قرار دارد. به همین ترتیب برای L_y و R_y. میانه در سمت راست mid است اگر که بیشتر المان‌ها در سمت راست mid باشند و سمت چپ mid است اگر که بیشتر المان‌ها در سمت چپ mid باشند. در نتیجه هر دو می‌توانند بازه‌ی $[i,j]$ را پس از این مکالمه نصف کنند.

حال به صورت استقرایی، در ابتدا بازه‌ی $[1, 2n]$ را نصف می‌کنند و در مرحله‌ی k از الگوریتم با توجه به این که R_x, L_x, R_y, L_y های قبلی در تاریخچه‌ی مکالمه موجود است، با روش گفته شده می‌توان به سادگی دید که بازه‌ی فعلی میانه قابل نصف کردن است. توجه شود که به صورت بدیهی نباید تنها براساس R_x, L_x, R_y, L_y فعلی بازه را نصف کنیم و برای نصف کردن بازه همچنین به R_x, L_x, R_y, L_y مخابره شده از ابتدای اجرا تا به حال نیاز است. به بیانی هربار که بازه نصف می‌شود و به نیمه‌ی راست یا چپ می‌رویم، همواره به یاد داریم که چه تعداد از المان‌ها در بازه‌ی رها شده هستند.

R_x, L_x, R_y, L_y هر کدام عددی بین 1 تا $2n$ هستند. در نتیجه در مجموع $4.\lceil \log n \rceil$ بیت برای نمایش آن‌ها مورد نیاز است. همچنین از بازه‌ی $[1, 2n]$ شروع می‌کنیم و هربار بازه را نصف می‌کنیم. در نهایت تا $\log 2n$ می‌توانیم بازه را نصف کنیم تا به بازه‌ی به طول 1 برسیم. در نتیجه این پروتکل $O(\log n)$ مرحله دارد که در هر مرحله $O(\log n)$ بیت مکالمه وجود دارد. در نتیجه هزینه‌ی این الگوریتم از مرتبه‌ی $O(\log^2 n)$ است. به این ترتیب $D(MEDIAN) \leq O(\log^2 n)$.

۴.۳ P^{cc}

از آن‌جا که برای برای هر تابع f کران بالای اشاره شده در قضیه‌ی ۲ وجود دارد شاید به نظر برسد که مطالعه‌ی کلاس‌های پیچیدگی که به وسیله‌ی پیچیدگی ارتباطی توصیف می‌شوند از لحاظ نظری جالب نباشد. ولی با توجه به ساختار خاص و ابزارهای فوق‌العاده‌ی نظریه پیچیدگی ارتباطی برای نشان دادن کران پایین، مطالعه‌ی این نظریه از نگاه ساختاری در پیچیدگی محاسبه مورد توجه است. این کلاس‌ها در [۸] مورد بررسی قرار گرفته است و تا به امروز جز حوزه‌های فعال در نظریه‌ی پیچیدگی محسوب می‌شود.

تعریف کلاس حل‌پذیر به صورت قطعی در زمان چندجمله‌ای P به نسبت این نظریه با توجه به قضیه‌ی ۲ کلاس بدیهی محسوب شده و از لحاظ نظری جالب نیست. به همین دلیل به جای حل‌پذیری در زمان چندجمله‌ای، مفهوم حل‌پذیری در زمان یک چندجمله‌ای از $\log n$ که آن را به شکل $poly(\log n)$ نمایش می‌دهیم مورد بررسی قرار می‌گیرد. این کلاس معروف پیچیدگی محاسبه که به وسیله‌ی پیچیدگی ارتباطی تعریف می‌شود کلاس P^{cc} است. این کلاس شامل تمام مساله‌هایی است که برای آن‌ها کران بالای پیچیدگی ارتباطی $O(\log^k(n))$ برای یک $k \geq 0$ وجود دارد. این کلاس را به شکل دقیق‌تر توصیف می‌کنیم:

تعریف ۶.

$$P^{cc} := \{f : D(f) = O(poly(\log n))\}$$

پیش از این دیدیم که $D(MEDIAN) \leq O(\log^2 n)$ از این می‌توان نتیجه گرفت که

$$MEDIAN \in P^{cc}$$

۴ کران پایین

یکی از دلایل محبوبیت نظریه‌ی پیچیدگی ارتباطی ابزارهای قدرتمند آن برای نشان دادن کران پایین است. در این بخش یک کران پایین برای مساله‌ی تساوی می‌بینیم. روشی که با آن کران پایین این مساله را نشان می‌دهیم روش *مجموعه‌ی گول‌زننده*[۱۵] نام دارد که می‌توان آن را برای نشان دادن کران پایین دیگر مساله‌ها نیز استفاده کرد. به همین رو پس از دادن این کران پایین روش را به شکل کلی ارزیابی خواهیم کرد.

۱.۴ EQUALITY

تابع $EQUALITY$ به شکل زیر تعریف می‌شود:

$$\begin{array}{lll} x = y & \Rightarrow & EQUALITY(x,y) = 1 \\ x \neq y & \Rightarrow & EQUALITY(x,y) = 0 \end{array}$$

با توجه به قضیه‌ی ۲ می‌دانیم که $D(EQUALITY) \leq n+1$ حال می‌خواهیم نشان دهیم که $D(EQUALITY) \geq n$ به این منظور ابتدا باید یک لم اساسی را اثبات کنیم.

لم ۱.۴. *بگذاریم $\pi = (A, B, N)$ یک پروتکل دل‌خواه باشد و*

$$\{(x,y),(x',y')\} \subseteq \{0,1\}^n \times \{0,1\}^n$$

دو ورودی باشند. حال بگذاریم $s_\pi(x,y)$ ، $s_\pi(x',y')$ ، $s_\pi(x,y')$ و $s_\pi(x',y)$ تاریخچه‌های مکالمه‌ی ورودی‌های (x,y)، (x',y')، (x,y') و (x',y) باشد. حال اگر $s_\pi(x,y) = s_\pi(x',y')$ آن‌گاه

$$s_\pi(x,y) = s_\pi(x',y') = s_\pi(x,y') = s_\pi(x',y)$$

اثبات. با استقرا روی طول تاریخچه‌ی مکالمه. برای $i = 1$ از فرض داریم:

$$s_\pi(x,y) = s_\pi(x',y') = b_1, b_2, \ldots, b_k \tag{۱}$$

حال برای ورودی (x',y) بدون از دست دادن عمومیت فرض کنیم $N(\epsilon) = A$ آن‌گاه اولین بیت را آلیس ارسال می‌کند که همان اولین بیت $s_\pi(x',y)$ است که آن را با $s_\pi(x',y)[1]$ نشان می‌دهیم و برابر است با $A(x',\epsilon)$ حال از ۱ داریم

$$s_\pi(x,y)[1] = s_\pi(x',y')[1] = A(x,\epsilon) = A(x',\epsilon) = b_1$$

در نتیجه

$$s_\pi(x',y)[1] = s_\pi(x,y)[1] = s_\pi(x',y')[1]$$

[۱۵] Fooling set

به طور مشابه $s_\pi(x,y')[1] = A(x,\epsilon) = b_1$ است. در نتیجه بیت اول هر چهار تاریخچه‌ی مکالمه یکسان است.
حال فرض کنیم از بیت ۱ تا بیت i ام هر چهار تاریخچه برابر است. در نتیجه از ۱ داریم:

$$s_\pi(x,y)[1,i] = s_\pi(x',y')[1,i] = s_\pi(x,y')[1,i] = s_\pi(x',y)[1,i] = b_1, b_2, \dots, b_i \tag{۲}$$

حال ورودی‌های (x',y) و (x,y') را در نظر بگیریم. بیت $i+1$ این ورودی‌ها را $N(b_1, b_2, \dots, b_i)$ ارسال می‌کند. بدون از دست دادن عمومیت فرض کنیم $N(b_1, b_2, \dots, b_i) = A$ در نتیجه

$$s_\pi(x',y)[i+1] = A(x', b_1, b_2, \dots, b_i) \tag{۳}$$

و

$$s_\pi(x,y')[i+1] = A(x, b_1, b_2, \dots, b_i) \tag{۴}$$

حال از ۱ داریم می‌دانیم که بیت $i+1$ ام را روی ورودی (x,y) و (x',y') همان $N(b_1, \dots, b_i)$ ارسال می‌کند و $N(b_1, \dots, b_i) = A$ در نتیجه از ۱ و ۲ و ۳ و ۴ داریم

$$\begin{aligned} s_\pi(x,y)[i+1] &= s_\pi(x',y')[i+1] \\ &= A(x, b_1, b_2, \dots, b_i) \\ &= A(x', b_1, b_2, \dots, b_i) \\ &= s_\pi(x,y')[i+1] \\ &= s_\pi(x',y)[i+1] \end{aligned}$$

و به این ترتیب حکم اثبات می‌شود.
■

نتیجه ۲.۴. *اگر شرایط لم ۴.۱ برقرار باشد و π تابع f را محاسبه کند آن‌گاه*

$$f(x,y) = f(x',y') = f(x',y) = f(x,y')$$

اثبات. از آن‌جا که π تابع f را محاسبه می‌کند پس

$$\begin{aligned} f(x,y) &= \pi(x,y) \\ f(x',y') &= \pi(x',y') \\ f(x',y) &= \pi(x',y) \\ f(x,y') &= \pi(x,y') \end{aligned}$$

حال از لم قبل می‌دانیم که

$$s_\pi(x,y) = s_\pi(x',y') = s_\pi(x',y) = s_\pi(x,y')$$

از تعریف داریم

$$\begin{array}{lllll} (1) & \pi(x,y) & = A(x, s_\pi(x,y)) & = & B(y, s_\pi(x,y)) \\ (2) & \pi(x',y') & = A(x', s_\pi(x,y)) & = & B(y', s_\pi(x,y)) \\ (3) & \pi(x',y) & = A(x', s_\pi(x,y)) & = & B(y, s_\pi(x,y)) \\ (4) & \pi(x,y') & = A(x, s_\pi(x,y)) & = & B(y', s_\pi(x,y)) \end{array}$$

$(4) = (1)$ بخاطر عبارت راست اولین تساوی در آن‌ها و $(3) = (2)$ نیز بخاطر عبارت راست اولین تساوی در آن‌ها. سپس $(3) = (1)$ بخاطر عبارت راست دومین تساوی در آن‌ها و $(4) = (2)$ بخاطر عبارت راست دومین تساوی در آن‌ها. در نتیجه $(4) = (3) = (2) = (1)$ و به این ترتیب حکم اثبات می‌شود.
■

قضیه ۳.۴.

$$D(EQUALITY) \geq n$$

اثبات. با برهان خلف فرض کنیم که $D(EQUALITY) < n$. پس پروتکل π وجود دارد که تابع $EQUALITY$ را محاسبه می‌کند به شکلی که هزینه‌ی آن برای هر $\{x,y\} \subseteq \{0,1\}^n$ از $n-1$ کمترمساوی است. حال تعداد تاریخچه‌های مکالمه‌ی با طول کمترمساوی $n-1$ برابر است با

$$\sum_{i=0}^{n-1} 2^i = 2^n - 1$$

سپس مجموعه‌ی

$$FS := \{(x,x) : x \in \{0,1\}^n\}$$

را در نظر بگیرید. مشاهده کنید که $|FS| = 2^n$ پس طبق اصل لانه کبوتری $\{(x,x),(y,y)\} \subseteq FS$ وجود دارند که $x \neq y$ به شکلی که تاریخچه‌ی مکالمه‌ی (x,x) با تاریخچه‌ی مکالمه‌ی (y,y) یکسان است. اگر تاریخچه‌ی مکالمه‌ی این دو به ترتیب $s_\pi(x,x)$ و $s_\pi(y,y)$ آن‌گاه از آن‌جا که $s_\pi(x,x) = s_\pi(y,y)$ پس طبق لم ۱.۴

$$s_\pi(x,x) = s_\pi(y,y) = s_\pi(x,y) = s_\pi(y,x)$$

پس براساس نتیجه‌ی ۲.۴ داریم

$$\begin{aligned} EQUALITY(x,x) &= EQUALITY(y,y) \\ &= EQUALITY(x,y) \\ &= EQUALITY(y,x) \end{aligned}$$

ولی از آن‌جا که $x \neq y$ پس

$$EQUALITY(x,x) = 1 \neq EQUALITY(x,y) = 0$$

که این تناقض است. در نتیجه $D(EQUALITY) \geq n$ و به این ترتیب حکم اثبات می‌شود.
■

به تکنیکی که توسط آن کران‌پایین n را برای $EQUALITY$ اثبات کردیم تکنیک مجموعه‌ی گول‌زننده [16] می‌گویند، در ادامه این روش را به صورت کلی معرفی می‌کنیم.

۲.۴ مجموعه‌ی گول‌زننده

مجموعه‌ی گول‌زننده مجموعه‌ای از ورودی‌ها مانند FS برای تابع f است که برای هر دو ورودی $(x,y) \in FS$ و $(x',y') \in FS$ داریم $f(x,y) = f(x',y')$ که برای آن‌ها همواره دستِ‌کم یکی از دو حالت $f(x,y) \neq f(x,y')$ یا $f(x,y) \neq f(x',y)$ برقرار است.

با توجه به ساختار این مجموعه، برای هر پروتکل π که تابع f را محاسبه می‌کند باید تاریخچه‌ی مکالمه برای هر دو ورودی $(x,y) \in FS$ و $(x',y') \in FS$ متفاوت باشد. چراکه در غیر این صورت با توجه به این که $f(x,y) \neq f(x,y')$ یا $f(x,y) \neq f(x',y)$ با استفاده از لم ۱.۴ و نتیجه‌ی ۲.۴ با استدلالی مشابه به قضیه‌ی ۳.۴ می‌توان به تناقض رسید.

[16] Fooling set method

نتیجه ۴.۴. *فرض کنیم FS یک مجموعه‌ی گول‌زننده برای تابع f است. حال برای هر پروتکل π که f را محاسبه می‌کند هزینه‌ی پروتکل همواره از $\lceil \log |FS| \rceil$ بیشترمساوی است.*

اثبات. نتیجه‌ی مستقیم استدلال پیش.

۵ کاربردها

می‌توان از نتایج پیچیدگی ارتباطی برای نشان دادن کران پایین در دیگر مدل‌های محاسباتی استفاده کرد. برای نمونه می‌خواهیم به کمک کران‌پایین $D(EQUALITY) \geq n$ نشان دهیم هر ماشین تورینگ تک‌نواره‌ی قطعی که زبان $PALINDROME$ را تصمیم می‌گیرد دست کم به زمان $\Omega(n^2)$ برای اجرا نیاز دارد.

۱.۵ کران پایین در ماشین‌های تورینگ

زبان $PALINDROME$ را با PAL نشان می‌دهیم و آن را به صورت زیر تعریف می‌کنیم:

$$PAL := \{w \in \{0,1\}^* : w = w^R\}$$

به شکلی که w^R رشته‌ی w است که برعکس شده.

قضیه ۱.۵. *هر ماشین تورینگ تک‌نواره که PAL را تصمیم می‌گیرد در زمان $\Omega(n^2)$ اجرا می‌شود.*

اثبات. می‌خواهیم نشان دهیم که اگر $D(EQUALITY) \geq n$ آن‌گاه هزینه‌ی تصمیم PAL در تورینگ تک‌نواره از مرتبه‌ی $\Omega(n^2)$ است. از آن‌جا که در قضیه‌ی ۳.۴ نشان دادیم که

$$D(EQUALITY) \geq n$$

پس از آن بلافاصله حکم نتیجه می‌شود.

بگذاریم M یک تورینگ تک‌نواره باشد که PAL را تصمیم می‌گیرد. فرض کنیم ورودی ماشین تورینگ مضربی از 3 باشد. بگذاریم طول ورودی n باشد. حال نوار ماشین تورینگ را به سه بخش با اندازه‌ی $\frac{n}{3}$ تقسیم می‌کنیم. فرض کنیم سلولی در موقعیت $\frac{n}{3} \leq i \leq \frac{2n}{3}$ سلولی باشد که تعداد گذرهای هد ماشین تورینگ از آن سلول k است. بگذاریم Q حالت‌های متناهی M باشد. حال ادعا می‌کنیم که می‌توان پروتکلی برای $EQUALITY$ داد که هزینه‌ی آن $\lceil \log |Q| \rceil .k + 1$ باشد.

فرض کنیم که $\{x, y\} \subseteq \{0,1\}^m$ حال مشاهده شود که

$$EQUALITY(x, y) = 1 \Leftrightarrow x0^m y^R \in PAL$$

حال پروتکل π را از روی M می‌سازیم. برای ورودی $\{x, y\} \subseteq \{0,1\}^m$ می‌دانیم که $|x| = |y| = m$ حال از سلول 1 تا i را به آلیس اختصاص می‌دهیم و از سلول $i+1$ به بعد را به باب. آلیس در بخش مربوط به خود ابتدا x را که اندازه‌ی m دارد نوشته و بقیه‌ی آن را (در صورت وجود) با 0 پر می‌کند. مشاهده شود که از بخش $2m$ به بعد در هر صورت برای باب است، چرا که $m \leq i \leq 2m$ حال باب از سلول $2m$ تا $3m$ را با y^R پر می‌کند و سلول‌های $i+1$ تا $2m$ را (در صورت وجود) با 0 پر می‌کند. آلیس به بخش

باب دسترسی ندارد و باب هم به بخش آلیس دسترسی ندارد. حال ماشین تورینگ آلیس را اجرا می‌کنیم و تا زمانی که هد ماشین تورینگ تلاشی برای رد شدن از سلول i انجام نمی‌دهد آن را به صورت محلی توسط آلیس اجرا می‌کنیم. به محض آن‌که ماشین تورینگ بخواهد از سلول i رد شود، آلیس نمی‌تواند محاسبه را ادامه دهد و برای ادامه‌ی محاسبه باید کنترل را به باب بدهیم. باب برای ادامه‌ی محاسبه نیاز به دانشی در مورد حالتی دارد که آلیس در آخرین قدم محاسبه‌اش در آن بوده است. به همین جهت آلیس باید حالت فعلی را برای باب بفرستد. از آن‌جا که حالت‌های ماشین تورینگ Q است پس می‌توان هر حالت آن را در $\lceil \log |Q| \rceil$ بیت کد کرد. به این ترتیب آلیس حالت فعلی خود را با $\lceil \log |Q| \rceil$ بیت مکالمه برای باب ارسال می‌کند. حال باب به ادامه‌ی محاسبه می‌پردازد و به همین ترتیب اگر در بین محاسبه بخواهد از $i+1$ به i برود با ارسال حالت فعلی خود برای آلیس ادامه‌ی محاسبه را به آلیس پاس می‌دهد. حال محاسبه یا در بخش آلیس تمام می‌شود یا در بخش باب. به هر صورت پس از اتمام محاسبه، نتیجه‌ی محاسبه، که یا $Accept$ است یا $Reject$ باید برای طرف مقابل ارسال شود. این را می‌توان در قالب یک بیت برای طرف مقابل ارسال کرد. اگر حالت نهایی $Accept$ بود آن‌گاه جواب $EQUALITY(x,y) = 1$ و اگر این حالت $Reject$ بود آن‌گاه جواب $EQUALITY(x,y) = 0$ است.

می‌خواهیم هزینه‌ی این پروتکل را بررسی کنیم. تعداد گذرهایی که از سلول i داشتیم برابر k بود و هر بار $\lceil \log |Q| \rceil$ بیت مکالمه داشتیم و در نهایت یک بیت برای نتیجه باید ارسال شود. در نتیجه هزینه‌ی پروتکل برابر

$$k.\lceil \log |Q| \rceil + 1$$

است. از آن‌جا که $D(EQUALITY) \geq m$ در نتیجه

$$k.\lceil \log |Q| \rceil + 1 \geq m$$

و $m = \frac{n}{3}$ همچنین $\lceil \log |Q| \rceil + 1$ از اندازه‌ی ورودی مستقل است. پس

$$c.k \geq \frac{n}{3}$$

در نتیجه

$$k \in \Omega(n)$$

پس برای هر سلول i در بازه‌ی $\frac{n}{3}$ تا $\frac{2n}{3}$ تعداد گذرها از مرتبه‌ی $\Omega(n)$ است. در این بازه $\frac{n}{3}$ سلول وجود دارد. تعداد گذرها در هر کدام از مرتبه‌ی $\Omega(n)$ است. در نتیجه مجموع گذرهای هد در این سلول‌ها از مرتبه‌ی $\Omega(n^2)$ است. هر گذر هد متناظر یک قدم است. در نتیجه مجموع قدم‌های ماشین تورینگ تنها در سلول‌های بین $\frac{n}{3}$ تا $\frac{2n}{3}$ از مرتبه‌ی $\Omega(n^2)$ است. پس هر ماشین تورینگ تک‌نواره‌ای که PAL را تصمیم می‌گیرد در زمان $\Omega(n^2)$ اجرا می‌شود. به این ترتیب حکم اثبات می‌شود.
■

References

[1] A. C.-C. Yao, "Some complexity questions related to distributive computing (preliminary report)," in *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pp. 209–213, 1979. 1, 2

[2] C. D. Thompson, "Area-time complexity for vlsi," in *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pp. 81–88, 1979. 1

[3] E. Kushilevitz, "Communication complexity," in *Advances in Computers*, vol. 44, pp. 331–360, Elsevier, 1997. 1, 2

[4] B. Kalyanasundaram and G. Schnitger, *Communication complexity and lower bounds for sequential computation*. Springer, 1992. 1

[5] D. Itsykson and A. Riazanov, "Proof complexity of natural formulas via communication arguments," in *36th Computational Complexity Conference (CCC 2021)*, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021. 1

[6] A. Rao and A. Yehudayoff, *Communication Complexity: and Applications*. Cambridge University Press, 2020. 2

[7] T. Roughgarden *et al.*, "Communication complexity (for algorithm designers)," *Foundations and Trends® in Theoretical Computer Science*, vol. 11, no. 3–4, pp. 217–404, 2016. 2

[8] L. Babai, P. Frankl, and J. Simon, "Complexity classes in communication complexity theory," in *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pp. 337–347, IEEE, 1986. 7