دانشکده علوم ریاضی
دانشگاه صنعتی شریف

تحقیق در عملیات ۱

محمدهادی فروغمنداعرابی
پاییز ۱۳۹۹

کاربرد برنامه‌ریزی خطی در کدها

جلسه هفدهم

نگارنده: امید ابوالاحراری

در این جلسه کاربردی از برنامه‌ریزی خطی از بخش ۴.۸ کتاب [JB07] را می‌گوییم.

۱ مروری بر مباحث گذشته

در جلسات قبل کاربردهای مختلف برنامه‌ریزی خطی را بررسی کردیم. کاربرد برنامه‌ریزی خطی در نظریهٔ بازیها را دیدیم. الگوریتم تقریبی را جلسه پیش بررسی کردیم. چند قضیه در گراف را اثبات کردیم. این جلسه قصد داریم کاربرد برنامه‌ریزی خطی را در کدگذاری را بررسی کنیم.

۲ مفاهیم و تعاریف اولیهٔ کدها

۱.۲ مثالی از کدگذاری

فرض کنید میخواهیم داده‌ای ۴ بیتی را روی DVD ذخیره یا مخابره کنیم. این داده که به شکل صفر یک است، ۱۶ حالت مختلف میتواند داشته باشد و ما در واقع میخواهیم یکی از این حالات رو روی دیسک بنویسیم یا ارسال کنیم. مشکل این است که ممکن است خطا داشته باشیم یعنی داده‌ای که ارسال می‌شود همانی نباشد که خوانده می‌شود و در فرایند ارسال خراب شده باشد. در نتیجه میخواهیم داده‌مان را نه لزوما به شکل ۴ بیتی بلکه طوری ارسال کنیم که اگر دچار خطا شود قابل تشخیص و حتی اصلاح باشد. یکی از راه‌های موجود برای حل این مشکل این است که هنگام ارسال، هر بیت را سه بار تکرار کنیم یعنی اگر میخواهیم ۱۰۰۱ را ارسال کنیم آن را به فرم ۱۱۱۰۰۰۰۰۰۱۱۱ ارسال کنیم. حال اگر داده دریافتی به شکل ۱۱۱۰۰۱۰۰۰۱۱۱ باشد میتوانیم حدس بزنیم که بیت ششم دچار خطا شده و احتمالا ۰ بوده است و بقیه بیت‌ها نیز درست دریافت شده‌اند.(لازمه‌اش این پیشفرض است که در هر ۳ بیت متوالی نهایتا یک خطا داریم.) اما این روش چون دقت بالایی ندارد، در شرایطی که روی دقت داده‌ها حساس باشیم کارامد نیست. برای افزایش دقت نیز میتوانستیم هر بیت را به جای سه بار تکرار، مثلا صد بار تکرار کنیم. حال مسئله را به صورت

کلی صورت‌بندی می‌کنیم:
در حالت کلی اگر داده ما N حالت داشته باشد و ما بخواهیم یکی از آن حالات را با n بیت ارسال کنیم، هدف ما این خواهد بود که اگر حداکثر rتا خطا رخ داده باشد بتوانیم دادهٔ اصلی را بازیابی کنیم. بعد از تعریف چند مفهوم به این سوال باز خواهیم گشت.

۲.۲ تعریف چند مفهوم در کدگذاری

برای اینکه تعریف خود را از مسئله دقیق‌تر کنیم، در ابتدا تابعی را به نام **فاصلهٔ همینگ**[1] تعریف میکنیم.

تعریف: برای دو کلمه: $\mathbf{w}, \mathbf{w}' \in \{0,1\}^n$

$$d_H(\mathbf{w}, \mathbf{w}') := |\{j \in \{1, \ldots, n\} : w_j \neq w'_j\}|$$

فاصله همینگ دو کلمه در واقع تعداد بیتهایی با اندیس برابر است که در آن دو کلمه مقدار متفاوتی دارند.

اندازه [2] یک کلمه را نیز به صورت زیر تعریف میکنیم:

$$|\mathbf{w}| := |\{j \in \{1, \ldots, n\} : w_j = 1\}|$$

در واقع اندازهٔ یک کلمه برابر با تعداد ۱های موجود در آن است.

همچنین XOR دو کلمه نیز به شکل مقابل تعریف میشود:

$$\mathbf{w} \oplus \mathbf{w}' = ((w_1 + w'_1) \bmod 2, \ldots, (w_n + w'_n) \bmod 2) \in \{0,1\}^n$$

XOR دوکلمه در بیتهایی که مقدارشان مساوی است برابر ۰ و در بیتهایی که مقدارشان متفاوت است برابر ۱ است.

مثال ۱. *اگر دو کلمه $w_1 = 1001$ و $w_2 = 0101$ داشته باشیم در نتیجه $d_H = 2$ خواهد بود که به معنای این است که تعداد بیت‌های با اندیس مساوی که مقدار آنها متفاوت است برابر با ۲ است. و هم چنین $|w_1| = 2$ و $|w_2| = 2$ که این نماد تعداد ۱ های کلمه‌مان را نشان می‌دهد و همچنین $w_1 \oplus w_2 = 1100$ است.*

در نهایت میتوان ادعا کرد که رابطه زیر نیز برقرار است:

$$d_H(\mathbf{w}, \mathbf{w}') = |\mathbf{w} \oplus \mathbf{w}'|$$

حال به سراغ تعریف **کد**[3] میرویم:

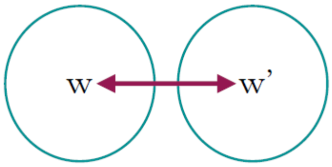
Definition. *A code $\mathcal{C} \subseteq \{0,1\}^n$ has* **distance** *d if $d_H(\mathbf{w}, \mathbf{w}') \geq d$ for any two distinct words $\mathbf{w}, \mathbf{w}'$ in $\mathcal{C}$. For $n, d \geq 0$, let $A(n,d)$ denote the maximum cardinality of a code $\mathcal{C} \subseteq \{0,1\}^n$ with distance d.*

به مجموعه ای از رشته‌های n بیتی کد میگوییم و میگوییم یک کد فاصله d را دارد اگر مینیمم فاصله همینگ رشته‌های موجود در کد ما به اندازهٔ d باشد. یعنی فاصله همینگ دوبه‌دوی تمام رشته‌ها را به دست می‌آوریم و همگی باید بزرگتر مساوی d باشند. $A(n,d)$ نیزماکزیمم تعداد کلماتی است که در کد nبیتی ما که فاصله آن d است میتوانیم داشته باشیم.
مثلا $n = 100$ را به ما میدهند و میگویند کدی ۱۰۰ بیتی تهیه کنید که فاصله دوبه‌دوی کلمات آن برابر ۷ باشد. هدف ما این است که کدی با بیشترین تعداد کلمات را بیابیم که این دو شرط در مورد آن برقرار باشد.

[1] distance Hamming
[2] weight
[3] Code

۳ تصحیح خطا در کدها

حال با تعاریف و توابعی که مشخص کردیم به سراغ حل پرسش اولیه‌مان می‌رویم. فرض کنید دو کلمه با فاصله d از یکدیگر داریم. اگر مطابق شکل دور هرکدام از این کلمات کره‌ای به شعاع $r < \frac{d}{۲}$ بزنیم هر کلمه‌ای در کره با کلمه موجود در مرکز فاصله ای کمتر از $\frac{d}{۲}$ و با کلمه کره دیگر فاصله‌ای بیشتر از $\frac{d}{۲}$خواهد داشت.



شکل ۱: برای $r < \frac{d}{۲}$، میتوان تا r خطا را اصلاح کرد.

حال فرض کنید کدی با فاصله d داریم یعنی فاصله تمام کلمات آن از یکدیگر بزرگتر مساوی d است. اگر کلمه x را داشته باشیم که با یکی از کلمات کد ما فاصله‌ای برابر $r < \frac{d}{۲}$ داشته باشد در نتیجه به طور یکتا کلمه‌ای در کد ما وجود دارد که به کلمه x شبیه‌تر است.(چونکه هیچکدام از کره‌ها به شعاع r با یکدیگر اشتراک ندارند).در این کد تا حداکثر r بیت خطا را میتوان مشخص واصلاح کرد و گفت که در اصل به چه صورت بوده است.

مثلا اگر در کد ما $d = ۱۱$ باشد ما تا خطای کمتر از ۵ را میتوانیم تصحیح کنیم و هرچه تعداد کلمات موجود در کد ما بیشتر باشد می‌توانیم تعداد بیشتر حالت را ارسال کنیم و برای ما مطلوب‌تر است.

۱.۳ بیشینه اندازهٔ کد

سوال اصلی اما این است که حداکثر چند کلمه به طول n با فاصله d از یکدیگر میتوانیم پیدا کنیم. برای روشن‌تر شدن پرسش ابتدا چند حالت خاص را بررسی می‌کنیم.

مثال ۲. *حداکثر تعداد رشته‌های nبیتی که حداقل با یکدیگر فاصله ۱ دارند برابر ۲^n است. چرا که ما حداکثر ۲^n رشته n بیتی داریم و می‌دانیم فاصله این رشته‌ها از یکدیگر حداقل ۱ است چونکه حداقل در ۱ بیت با یکدیکر اختلاف خواهد داشت. در نتیجه می‌توان نوشت:*

$$A(n, ۱) = ۲^n$$

مثال ۳. *حداکثر تعداد رشته‌ها n بیتی که حداقل با یکدیگر فاصله ۲ دارند برابر $۲^{n-۱}$ است. اگر تمام کلمات nبیتی را بنویسیم، نصف آنها تعداد ۱ هایشان فرد و نصفشان تعداد ۱ هایشان زوج خواهد بود. حال تمام کلماتی که تعداد ۱ هایشان فرد است با یکدیگر حداقل فاصله ۲ را دارند چرا که اگر دوتا از آنها فاصله‌شان ۱ باشد یعنی تنها در یک بیت، یکی از آنها ۰ و دیگری ۱ خواهد بود و چون یکیشان فردتا ۱ دارد پس دیگری زوج‌تا ۱ خواهد داشت که خلاف فرض ماست و از آنجا که فاصله هیچکدام ۰ نیز نمی‌تواند باشد پس حداقل فاصله‌شان ۲ است. این استدلال برای کلمات با تعداد ۱ های زوج نیز صادق است. پس ما مجموعه‌ای به طول نصف ۲^n یافتیم که فاصله‌شان d باشد اما هنوز نمیدانیم که آیا این ماکزیمم حالات نیز هست یا خیر. از طرف دیگر اگر هرکدام ازین کلمات را راسی از گراف در نظر بگیریم و تطابقی را بین هر دو راسی که فاصله‌شان ۱ است رسم کنیم مثلا هر کلمه را به کلمه‌ای که عین اوست و تنها در بیت اول اختلاف دارند وصل کنیم. ما در اینجا یک تطابق بین همه رئوس یعنی یک تطابق کامل داریم و از هر طرف ازین تطابق تنها یکی از آنها را میتوانیم در مجموعه‌مان بیاوریم چرا که اگر هردوی آنها باشند آنگاه دو کلمه با فاصله ۱ خواهیم داشت که خلاف خواستهٔ ماست. پس ازینجا میتوان گفت که مجموعهٔ ما حداکثر به اندازه نصف رئوس ما یعنی $۲^{n-۱}$ میتواند باشد. از طرفی کمی بالاتر نشان دادیم که یک جواب به همین اندازه یعنی$۲^{n-۱}$ داریم. پس میتوانیم بگوییم:*

$$A(n, ۲) = ۲^{n-۱}$$

مثال ۴. *به عنوان مثالی دیگر میخواهیم $A(۱۷, ۳)$ را به دست آوریم. با محاسباتی که انجام شده کران بالا و پایینی برای آن به دست آورده‌اند.*

$$۵۳۱۲ \leq A(۱۷, ۳) \leq ۶۵۵۲$$

برای به دست آوردن کران پایین آن تنها کافی است کدی با ۵۳۱۲ کلمه ارائه کنیم که فاصله دوبه‌دوی آنها از یکدیگر حداقل ۳ باشد. اما پرسش اصلی ما دربارهٔ چگونگی به دست آوردن کران بالاست.

۴ کران بالا برای $A(n,d)$

۱.۴ یافتن کران بالا برای $A(n,d)$ به کمک گوی زدن

روش اول استفاده از گوی است. مثال قبل را در نظر بگیرید. فرض کنید تمام کلمات به طول ۱۷ با حداقل فاصله ۳ را نوشته‌ایم. حال دور هر کدام گویی به شعاع یک رسم کنید یعنی کلماتی که حداکثر فاصله ۱ را از کلمه ما دارند درون یک گوی قرار می‌دهیم. میدانیم که گوی‌ها هیچ اشتراکی ندارند. حال تعداد کلمات درون هر گوی را به دست می آوریم و تعداد آنها را با هم جمع میکنیم و میدانیم این تعداد کوچکتر مساوی کل کلمات ۱۷ بیتی خواهد بود. میتوانیم این روش را تعمیم دهیم. اگر فرض کنیم کدی به طول $|c|$ داریم شامل رشته‌هایی طول n که فاصله هر کدام از یکدیگر حداقل $۲r+۱$ است. حال دور هر کلمه گویی به شعاع r رسم می‌کنیم. میخواهیم تعداد کلمات موجود در گوی را به دست آوریم. تعداد کلماتی که فاصله‌شان با کلمه مرکزی ما ۰ است برابر $\binom{n}{۰}$ است. تعداد کلماتی که فاصله‌شان با کلمه مرکزی ما ۱ باشد برابر $\binom{n}{۱}$ است چرا که انگار میخواهیم از n بیت، تنها ۱ بیت را انتخاب کنیم و آن را قرینه کنیم پس انگار انتخاب ۱ از n است. به همین صورت کلمات با فاصله ۲ از کلمه مرکزی ما برابر $\binom{n}{۲}$ خواهد بود و در نتیجه مجموع کلمات موجود در هر گوی برابر $\sum_{i=۰}^{n}\binom{n}{i}$ خواهد بود. حال واضح است که مجموع کلمات تمام گوی‌ها روی هم از مجموع کل کلمات موجود nبیتی کمتر خواهد بود چونکه در واقع مجموعه کلمات درون گوی‌ها زیرمجموعه کل کلمات است پس میتوانیم رابطه زیر را بنویسیم:

$$۲^n \geq |c| \sum_{i=۰}^{n} \binom{n}{i}$$

با جابجایی در جملهٔ بالا میتوانیم لم زیر را به دست آوریم:

Lemma (Sphere-packing bound). *For all n and r,*

$$A(n, 2r+1) \leq \left\lfloor \frac{2^n}{\sum_{i=0}^{r}\binom{n}{i}} \right\rfloor .$$

حال از رابطه بالا سعی میکنیم $A(۱۷,۳)$ را به دست آوریم. داریم:

$$A(۱۷,۳) \leq \lfloor \frac{۱۳۱۰۷۲}{۱۸} \rfloor = ۷۲۸۱$$

مشاهده میکنیم که اگرچه عددی را به عنوان کران بالا به دست آورده‌ایم اما از کران بالایی که می‌دانیم باید به دست می آوردیم فاصلهٔ زیادی دارد.

حال می‌خواهیم تلاش کنیم تا با استفاده از برنامه‌ریزی خطی کران بالای بهتری را برای $A(n,d)$ به دست آوریم.

۲.۴ یافتن کران بالا برای $A(n,d)$ به کمک برنامه‌ریزی خطی

برنامه‌ریزی خطی زیر را به صورت یک قضیه داریم. ابتدا قضیه را به صورت کامل نوشته و سپس سعی میکنیم بخش بخش آن را توضیح دهیم و اثبات کنیم.

8.4.3 Theorem (The Delsarte bound). *For integers n, i, t with $0 \leq i$, $t \leq n$, let us put*

$$A(n,d) \leq \boxed{\begin{array}{lll} \textit{Maximize} & x_0 + x_1 + \cdots + x_n & \\ \textit{subject to} & x_0 = 1 & \\ & x_i = 0, & i = 1,2,\ldots,d-1 \\ & \sum_{i=0}^{n} K_t(n,i)\cdot x_i \geq 0, & t = 1,2,\ldots,n \\ & x_0, x_1, \ldots, x_n \geq 0. & \end{array}}$$

$$K_t(n,i) = \sum_{j=0}^{\min(i,t)} (-1)^j \binom{i}{j}\binom{n-i}{t-j}$$

مطابق قضیه بالا یافتن کران بالا برای $A(n,d)$ از حل یک برنامه‌ریزی خطی حاصل میشود. تابع هدف آن برابر ماکزیمم مجموع $x_\circ$ تا x_n است. پیش از توضیح x_i ابتدا متغیری به نام y_i را به صورت زیر تعریف میکنیم:

$$y_i = |\{(w,w')|d_H(w,w') = i\}|$$

۱.۲.۴ محاسبهٔ قید اول

wوw' دو کلمه دلخواه از کد C هستند. y_i تعداد زوج مرتب‌های از w و w' است ک فاصله آنها از یکدیگر i هست. به عنوان مثال y_i تعداد زوج کلماتی از کد C است که فاصله آنها برابر ۱ است. حال از روی y_i به معرفی x_i میرسیم:

$$x_i = \frac{۱}{|c|} y_i$$

$y_\circ$ مطابق تعریف قبلی برابر تعداد زوج کلماتی است ک فاصله‌شان از یکدیگر صفر است. هر کلمه تنها فاصله‌اش از خودش صفراست در نتیجه در مجموع به اندازه تعداد کلماتمان زوج مرتب خواهیم داشت که این ویژگی را دارا باشند پس مینویسیم:

$$y_\circ = |c| \implies x_\circ = ۱$$

میبینیم که موفق شدیم قید اول برنامه ریزی خطی‌مان را به دست آوریم.

۲.۲.۴ محاسبهٔ قید دوم

از طرف دیگر چون فاصله کد ما d است یعنی هیچ دو کلمه متفاوتی را نمیتوان در آن یافت که فاصله‌شان از هم کوچکتر از d باشد؛ پس هیچ زوج مرتبی از کلمات را نمیتوان یافت که فاصله‌شان از یکدیگر کوچیکتر از d باشد پس y_i به ازای تمامی iهای کوچکتر از d برابر صفر است.

$$\begin{aligned} y_i &= \circ \qquad , \qquad i = ۱, ۲, ۳, ..., d-۱ \\ \implies x_i &= \circ \qquad , \qquad i = ۱, ۲, ۳, ..., d-۱ \end{aligned}$$

قید دوم برنامه‌ریزی خطی‌مان را نیز به دست آوردیم.

۳.۲.۴ محاسبه تابع هدف

میخواهیم $\sum_{i=\circ}^{n} y_i$ را محاسبه کنیم. در اینکار در واقع داریم تعداد زوج‌مرتب‌هایی از کلمات با فاصله ۰، ۱، ۲ تا n را با یکدیگر جمع بزنیم. عملا میخواهیم تعداد تمام زوج‌مرتب های ممکن را به دست آوریم. برای نوشتن تمام زوج مرتب‌ها به این شکل است که برای w، $|c|$ حالت داریم و برای w' نیز $|c|$ حالت داریم پس در مجموع $c^۲$ زوج مرتب خواهیم داشت. پس مینویسیم:

$$\sum_{i=\circ}^{n} y_i = |c|^۲$$

حال با توجه به رابطه بین x_iو y_i میتوان نوشت:

$$\sum_{i=\circ}^{n} x_i = |c|$$

رابطهٔ بالا همان تابع هدف است. پس مشاهده میکنیم که عملا در تابع هدف با ماکزیمم کردن $\sum_{i=\circ}^{n} x_i$ در واقع داریم c یعنی اندازهٔ کدمان را ماکزیمم میکنیم و به عبارتی دیگر داریم $A(n,d)$ را بیشینه میکنیم و به این معنی داریم کرانی بالا برای آن محاسبه می‌کنیم که هدفمان از ابتدا از نوشتن برنامه‌ریزی خطی نیز همین بود.

۴.۲.۴ محاسبهٔ قید سوم

ابتدا اقدام به اثبات قضیه زیر میکنیم:
قضیه زیر مطرح میکند که اگر مجموعه‌ای از کلمات n بیتی با نام C داشته باشیم تعداد زوج‌مرتب‌هایی ازین مجموعه که فاصله‌شان از یکدیگر زوج است بزگتر مساوی تعداد زوج‌مرتب‌هایی است که فاصله فرد دارند.

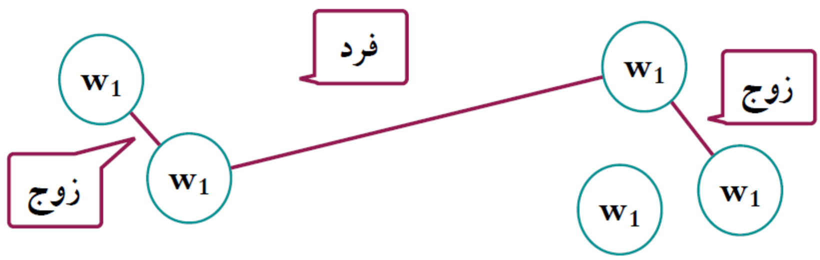
$$|\mathbf{w}|_I = |\{i \in I : w_i = 1\}|$$

$$d_H^I(\mathbf{w}, \mathbf{w}') = |\mathbf{w} \oplus \mathbf{w}'|_I$$

8.4.5 Lemma. *Let $I \subseteq \{1, 2, \ldots, n\}$ be a set of indices, and let $\mathcal{C} \subseteq \{0,1\}^n$. Then the number of pairs $(\mathbf{w}, \mathbf{w}') \in \mathcal{C}^2$ with $d_H^I(\mathbf{w}, \mathbf{w}')$ even is at least as large as the number of pairs $(\mathbf{w}, \mathbf{w}') \in \mathcal{C}^2$ with $d_H^I(\mathbf{w}, \mathbf{w}')$ odd. (In probabilistic*

اثبات. هر کلمه را به عنوان راسی از گراف در نظر میگیریم و هر زوج مرتب را به عنوان یک یال بین دو راس در نظر میگیریم و روی هر یال زوج یا فرد بودن فاصله دو راس را یادداشت میکنیم (بین هر راس و خودش هم باید یالی رسم کنیم). حال دو مجموعه ε و ϑ را در نظر می‌گیریم که اولی شامل کلماتی از مجوعه C با طول زوج و دومی شامل کلماتی با طول فرد است.

$$\mathcal{E} = \{\mathbf{w} \in \mathcal{C} : |\mathbf{w}|_I \text{ is even}\} \qquad \mathcal{O} = \{\mathbf{w} \in \mathcal{C} : |\mathbf{w}|_I \text{ is odd}\}$$



شکل ۲

سپس کل یال‌های گراف را بررسی میکنیم. سه نوع یال داریم: یال‌هایی که بین دو راس با طول زوج‌اند، یال‌هایی که بین دو راس با طول فردند و یال‌هایی که بین یک راس با طول زوج و یک راس با طول فردند. اگر هر دو راس یک یال فرد باشد آنگاه فاصله دو راس یعنی $d_H^I(w, w')$ که برابر $|w \oplus w'|$ است قطعا زوج است چرا که یا این دو کلمه هیچ اشتراکی ندارند در نتیجه $w \oplus w' = w + w'$ و جمع دو کلمه با فردتا عدد ۱، زوجتا عدد ۱ خواهد داشت؛ در نتیجه طول آن زوج است و یا اینکه این دوکلمه در اندیس‌هایی اشتراک دارند که در این صورت زوجتا از جمع این دو عدد فرد کم می‌شود(زوجتا به این علت که بیت‌های مشترک هر دوشان از مجموعشان کم خواهند شد پس در نتیجه مقدارش زوج است.) تفاضل زوجتا عدد ۱ از مجموع دو عدد که فردتا ۱ دارند نیز زوجتا ۱ خواهد داشت پس در هر حال اندزه فاصله دو راس با طول فرد زوج است. با همین استدلال نیز می‌توان گفت که فاصله دو راس با طول زوج نیز زوج خواهد بود و همینطور با استدلالی مشابه می‌توان گفت که فاصله دو راس که طول یکی زوج و دیگری فرد است، فرد خواهد بود.(با همان توضیح مشابه که اگر اشتراک نداشته باشند $w \oplus w' = w + w'$ که جمع دوعدد با زوجتا ۱ و فردتا ۱، فردتا ۱ خواهد داشت و همچنین اگر در k بیت اشتراک داشته باشند ازین مجموع فرد، $2k$ کم خواهد شد و همچنان فرد میماند.) حال تعداد یالهای بین رئوس با طول زوج برابر فلان است. تعداد یالهای بین رئوس با طول فرد نیز برابر فلان است و تعداد یالهای بین دو راس با طول زوج و فرد نیز برابر فلان است. پس قضیه بالا را به صورت زیر می‌توان نوشت

$$|\varepsilon|^{۲} + |\vartheta|^{۲} \geq ۲.|\varepsilon|.|\vartheta|$$

سمت چپ رابطهٔ بالا نشان دهندهٔ یالهای زوج و سمت راست نشان دهندهٔ یالهای فرد است. رابطه بالا نیز با یک جابجایی ساده به صورت مربع کامل در می‌آید و صحتش بدیهی خواهد بود. پس قضیه ثابت میشود.

□

از طرفی از قضیه بالا میتوان نتیجه گرفت اگر ۱− را به توان فاصله دو راس برسانیم و به ازای تمامی فواصل این کار را انجام دهیم و با یکدیگر جمع کنیم حاصل عددی نامنفی خواهد بود چرا که ۱− به زای فواصل زوج ۱+ میشود و به ازای فواصل فرد، ۱−. از آنجا که تعداد فواصل زوج

بزرگتر مساوی فواصل فرد است پس تعداد ۱ ها بزرگتر مساوی ۱− ها خواهد بود در نتیجه مجموعشان نامنفی خواهد شد. حال با استفاده ازین توضیحات به سراغ قضیه زیر می‌رویم:

For every $\mathcal{C} \subseteq \{0,1\}^n$ *and every* $\mathbf{v} \in \{0,1\}^n$ *we have*

$$\sum_{(\mathbf{w},\mathbf{w}')\in\mathcal{C}^2} (-1)^{(\mathbf{w}\oplus\mathbf{w}')^T\mathbf{v}} \geq 0.$$

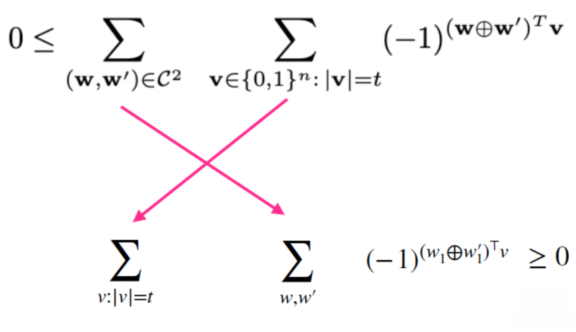
اثباتی جبری برای قضیهٔ بالا ارائه میدهیم:

$$\begin{aligned}
\sum_{(\mathbf{w},\mathbf{w}')\in\mathcal{C}^2} (-1)^{(\mathbf{w}\oplus\mathbf{w}')^T\mathbf{v}} &= \sum_{(\mathbf{w},\mathbf{w}')\in\mathcal{C}^2} (-1)^{(\mathbf{w}+\mathbf{w}')^T\mathbf{v}} \\
&= \sum_{(\mathbf{w},\mathbf{w}')\in\mathcal{C}^2} (-1)^{\mathbf{w}^T\mathbf{v}} \cdot (-1)^{\mathbf{w}'^T\mathbf{v}} \\
&= \left(\sum_{\mathbf{w}\in\mathcal{C}} (-1)^{\mathbf{w}^T\mathbf{v}}\right)^2 \geq 0.
\end{aligned}$$

در اثبات بالا به این دلیل توانستیم XOR را به جمع تبدیل کنیم چرا که در به توان رساندن ۱− تنها زوجیت و فردیتِ توان برای ما مهم است و مقدار طعددی آن اهمیتی ندارد و به لحاظ زوجیت و فردیت خرو جی XOR و جمع یکسان است چرا که در بیت‌های مشابه جمع آنها یا ۰ یا ۲ میشود که زوج است و XOR آنها نیز ۰ میشود و از طرفی دیگر در بیت‌های ناهمسانشان جمع آنها ۱ می‌شود و XOR آنها نیز ۱ میشود.
سپس به سراغ اثبات قضیهٔ زیر می‌رویم:

$$۰ \leq \sum_{(w,w')\in C^۲} \quad \sum_{v\in\{۰,۱\}^n:|v|=t} (-۱)^{(w\oplus w')^T v}$$

اثبات. ابتدا به صورت زیر میتوانیم جای دو سیگما را عوض کنیم.



شکل ۳: جابجایی در سیگماها

اثبات این فرم جدید از قضیه دیگر راحت است. سیگمای درونی همان جمله موجود در قضیه قبل است که نشان دادیم بزرگتر مساوی صفر است. پس سیگنای بیرونی جمع یک سری عدد نامنفی خواهد بود در نتیجه جواب آن نیز نامنفی خواهد بود.

□

گام آخر اما تبدیل قضیهٔ بالا به قید سوم بهینه‌سازیمان و نشان دادن برابری این دو است. تلاش می‌کنیم سیگمای داخلی را به صورتی دیگر بنویسیم. جمله $(w\oplus w')^T v$ را j نامگذاری و سعی کرده جملات درون سیگما را بر اساس jهایشان مرتب کنیم. یعنی در تعدادی از آنها $j = ۰$ ، در تعدادی دیگر $j = ۱$ و به همین ترتیب میتوان پیش رفت. حال باید محاسبه کنیم در چه تعداد از آنها $j = ۱$ ، در چه تعداد $j = ۲$ و ... است و سپس با جمع زدن تمام حالات بتوانیم سیگمای ابتداییمان را محاسبه کنیم. در واقع در اینجا ما تنها جملات موجود در سیگمای جدیدمان را به شکلی جدید دسته‌بندی کردیم و تغییر دیگری ایجاد نکردیم. میدانیم که v به اندازهٔ tتا ۱ دارد و همچنین فاصله همینگ w و w' برابر I است در نتیجه $w \oplus w'$،

i تا ۱ دارد. حال در صورتی این ضرب داخلی برابر j میشود که در بیت‌هایی که $w \oplus w'$ برابر با ۱ است، v در آن نقاط j تا ۱ داشته باشد و بقیه ۱های v که تعداشان برابر $t - j$ است باید در نقاطِ صفرِ $w \oplus w'$ باشند تا در این صورت حاصل ضرب داخلی ما برابر با j شود. برای به دست آوردن تعداد حالات این اتفاق در واقع انتخاب jتا ۱ از iتا ۱ موجود در $w \oplus w'$ ضربدر انتخاب $t - j$ تا صفر از $n - i$ تا صفر موجود در $w \oplus w'$ است پس میتوانیم سیگمای درونمیان را به صورت زیر بازنویسی کنیم:

$$\sum_{j} \binom{i}{j}\binom{n-i}{t-j}(-1)^{j} \qquad , \qquad i := d_H(w, w')$$

حال متغیری به اسم k_t تعریف میکنیم و سیگمای بالا را برابر با آن قرار میدهیم در نتیجه قضیه بالا را به صورت جدید بازنویسی میکنیم:

$$k_t = \sum_{j} \binom{i}{j}\binom{n-i}{t-j}(-1)^{j}$$

$$\Longrightarrow 0 \leq \sum_{(w,w') \in C^{2}} K_t(n, d_H(w, w'))$$

حال تنها به جای اینکه سیگمایمان روی w و w' باشد باید آن را روی i تعریف کنیم. برای این تبدیل مشابه حالت قبل سعی میکنیم جمله درون سیگما را بر اساس iدسته‌بندی کنیم یعنی k_t های مختلف را بر حسب iهایشان با یکدیگر جمع کنیم. حال باید به دست آوریم که به ازای هر i چه تعداد $k_t(n, i)$ داریم تا بتوانیم سیگما را بازنویسی کنیم. i فاصله بین دو راس است. در صورت بهینه‌سازی ابتداییمان مشخص کردیم که y_i برابر تعداد زوج‌مرتب هایی است که فاصله‌شان برابر i است پس میتوانیم سیگما را به صورت زیر بازنویسی کنیم:

$$0 \leq \sum_{i} K_t(n, i) y_i$$

حال از آنجا که $x_i = \frac{1}{|c|} y_i$ میتوانیم رابطه بالا را به صورت زیر بازنویسی کنیم:

$$0 \leq \sum_{i} K_t(n, i) x_i$$

مشاهده می‌کنیم که رابطه بالا همان قید سوم بهینه‌سازی ما است. پس موفق شدیم این قید را نیز ثابت کنیم و در واقع توانستیم تمام بخش‌های بهینه‌سازیمان را به دست آوریم.
با مروری کلی بر برنامه‌ریزی خطی میبینیم که تابع هدف ما $|c|$ است و قصد داریم آن را بیشینه کنیم و با حل این برنامه‌ریزی میتوانیم کرانی بالای برای $A(n, d)$ محاسبه کنیم.
در پایان و پس از اثبات برنامه‌ریزی خطی بالا، اقدام به حل این برنامه‌ریزی برای $A(17, 3)$ میکنیم و جواب زیر به دست می‌آید.

$$A(17, 3) \leq 6553\frac{3}{5} \implies A(17, 3) \leq 6553$$

و کرانی که میدانیم صحیح است به صورت زیر است:

$$5312 \leq A(17, 3) \leq 6552$$

مشاهده میکنیم که با بهینه‌سازی جواب بسیار خوبی به دست آوردیم اما همچنان ۱ واحد با بهترین جواب فاصله دارد.جالب این است که این ۱ واحد را نیز میتوانیم اصلاح کنیم. باید به مراحلی از اثبات برگردیم و با استفاده از مسئله زوجیت و فردیت، بهینه‌سازی و کرانهای خود را دقیقتر کنیم.

۳.۴ بهبود کران $A(n, d)$

فرض میکنیم $|C| = 6553$ و در نتیجه فرد است. حال به سراغ بخشی از اثبات میرویم و با تغییر کرانها و تغییر برنامه‌ریزی با توجه به جواب به دست آمده باید نشان دهیم که در برنامه‌ریزی جدیدی که به دست آورده‌ایم دیگر این جواب ما بیشینه نیست و ازین طریق به خلاف فرض خود برسیم.
در واقع میخواهیم به کمک برهان خلف نشان دهیم که کران ما کوچکتر از ۶۵۵۳ است.
به شکل ۳ بازمیگردیم. در آن نشان داده بودیم که $0 \leq \sum_{w,w'} (-1)^{(w \oplus w')^T v}$ است. همچنین نشان دادیم که $\sum_{w,w'} (-1)^{(w \oplus w')^T v} = (\sum_{w \in C} (-1)^{w^T v})^2$ است. حال از آنجا که میدانیم $|c|$ فرد است پس عملا در سیگمایمان داریم فردتا ۱ و ۱− را با یکدیگر جمع میزنیم پس

جوابش هرگز صفر نخواهد شد و از آنجا که به توان ۲ رسیده میدانیم که این سیگما حتما بزرگتر مساوی صفر است پس میتوانیم کران جدیدی برای آن بنویسیم:

$$(\sum_{w\in C}(-۱)^{w^Tv})^۲ \geq ۱$$

حال میتوانیم سیگمای شکل ۳ را با کران جدبد بازنویسی کنیم.

$$\sum_{v:|v|=t}\quad\sum_{w,w'}(-۱)^{(w\oplus w')^Tv} \geq \binom{n}{t}$$

چرا که سیگمای بیرونی در واقع $\binom{n}{t}$ حالت مختلف میتوانست بگیرد چرا که v ما t تا ۱ داشت که از کلمات n بیتی ما میتوانست آنها را ۱ کند. در نتیجه طبق آنچه قبلا نشان داده شده:

$$\sum_{(w,w')\in C^۲}\quad\sum_{v\in\{۰,۱\}^n:|v|=t}(-۱)^{(w\oplus w')^Tv} = \sum_i K_t(n,i)y_i \geq \binom{n}{t}$$

از آنجا که $x_i = \frac{۱}{|c|}y_i$ میتوانیم رابطه بالا را به صورت زیر بازنویسی کنیم:

$$\sum_{(w,w')\in C^۲}\quad\sum_{v\in\{۰,۱\}^n:|v|=t}(-۱)^{(w\oplus w')^Tv} = \sum_i K_t(n,i)x_i \geq \frac{\binom{n}{t}}{|C|}$$

پس برنامه‌ریزی خطی جدید ما به صورت زیر در می‌آید:

$$A(17,3) \leq \boxed{\begin{array}{lll} \textit{Maximize} & x_0+x_1+\cdots+x_n & \\ \textit{subject to} & x_0 = 1 & \\ & x_i = 0, & i=1,2,\ldots,d-1 \\ & \sum_{i=0}^{n} K_t(n,i)\cdot x_i \geq \boxed{\frac{\binom{n}{t}}{|C|}} & t=1,2,\ldots,n \\ & x_0,x_1,\ldots,x_n \geq 0. & \end{array}}$$

با حل برنامه‌ریزی بالا به عدد $۶۵۵۲\frac{۳}{۵}$ میرسیم و در واقع کران بالای جدیدی برای $A(۱۷,۳)$ پیدا کرده‌ایم.

اما در واقع چه اتفاقی افتاد؟ با فرض اینکه $|C| = ۶۵۵۳$ کران بالایی یافتیم که کمتر از این عدد است پس به تناقض رسیدیم و فرض ما غلط است پس کران بالا ۶۵۵۳ نیست و ثابت میشود که کران بالایمان ۶۵۵۲ است و به همان کران داده شده کهمیدانستیم درست است، رسیدیم.(البته وارد جزئیات اثبات آن نشدیم.)

$$\implies ۵۳۱۲ \leq A(۱۷,۳) \leq ۶۵۵۲$$

مراجع

[JB07] Matoušek Jiri and Gärtner Bernd. *Understanding and using linear programming.* Springer, 2007.

[۱] ویدیو جلسه هفدهم در آپارات