סיכום בסיבוכיות

Time Complexity:

3SAT: משפט coock-levine מראה לנו שהשפה הזו היא NPC על ידי רדוקציה מכול שפה אחרת ב NP על ידי בניית נוסחת 3CNF מטבלת הריצה של המכונה של השפה ב NP. (למעשה מראה לנו רדוקציה ל SAT כללי, ומשם לעבור ל CNF זה פשוט, כי הניפוח אינו אקספוננציאלי, ולעבור מ CNF ל 3CNF זה גם פשוט.

3NAE-SAT: שפת כול הנוסחאות 3CNF כך שיש להן השמה מספקת כך שבכול clause יש לפחות ליטרל אחד שקיבל ערך שקר. במילים אחרות בכול clause יש ליטרל אחד שקיבל אמת ואחד שקיבל שקר. קל לראות כי השפה הזו ב NP, כי השמה כזו היא עד שקל לוודא.

כמו כן השפה הזו NP קשה. כי $3SAT \leq_p 4NAE-SAT \leq_p 3NAE-SAT$. קל לראות כי $3SAT \leq 4NAE-SAT$, כי בהנתן נוסחת $3CNF$ נוסיף לכול הפסוקיות (clause) את אותו המשתנה, משתנה שלא היה שם קודם. בבירור אם הייתה השמה מספקת לנוסחת ה 3CNF אז על ידי נתינת ערך שקר למשתנה הנוסף נקבל השמה חוקית ל $4NAE-SAT$. כמו כן אם יש השמה מספקת לנוסחת ה $4NAE-SAT$ שלנו, אז אם המשתנה שהוספנו קיבל ערך שקר אז בבירור בכול פסוקית אחד מהמשתנים המקורים קיבל ערך אמת ולכן זו השמה מספקת לנוסחה המקורית. אחרת הוא קיבל ערך אמת, לכן בכול פסוקית יש לפחות משתנה אחד שקיבל ערך שקר, לכן אם נהפוך את ההשמה הזו נקבל השמה שלפחות משתנה אחד בכול פסוקית קיבל אמת, וזו השמה חוקית לנוסחה המקורית.

כמו כן $4NAE-SAT \leq 3NAE-SAT$ כי בהנתן נוסחת $4\text{NAE}-\text{SAT}$ נפרק אותה לנוסחת $3NAE-SAT$ ע"י שכול פסוקית של 4 משתנים נפרק לשתיים של שלושה משתנים ונוסיף משתנה חדש שיופיע בפסוקית אחת כמו שהוא ובשנייה יופיע בשלילה. בבירור אם הנוסחא החדשה היא $3NAE-SAT$ אז לא משנה מה הערך שקיבל המשתנה החדש שהוספנו, בפסוקיות השנייה הוא יופיע בשלילה וזה יכריח שלפחות אחד מה4 המשתנים יקבל אמת ולפחות אחד מהארבע יקבל שקר. אם הנוסחא המקורית היא $4NAE-SAT$ אז נתבונן בפסוקית x_1, x_2, x_3, x_4. אם בהשמה המספקת x_1, x_2 קיבלו את אותו הערך, אז w שהוא המשתנה שהוספנו לפסוקית הזו יקבל הערך ההפוך מהן, ומובטח לנו כי לפחות אחד מ x_3, x_4 קיבל ערך שונה, ולכן יחד עם השלילה של w גם בפסוקית השנייה יופיע גם אמת וגם שקר. כנ"ל אם x_3, x_4 קיבלו אותו ערך. אחרת זה לא משנה מה נשים בw, ההשמה הזו תעבור להשמה מספקת באופן ישיר.

Max-Cut: חתך בגרף מוגדר להיות חלוקה של הקודקודים לשתי קבוצות זרות שאיחודן הוא כול הקודקודים. גודל של חתך הוא מספר הקשתות שעוברות מקבוצה אחת לשנייה. נגדיר $MaxCut_k = \{(G,k) |\ G\ is\ undirected\ with\ a\ cut\ of\ size\ at\ least\ k\}$. ברור שהבעיה הזו ב NP כי החתך עצמו הוא עד.

הבעיה הזו NP קשה על ידי רדוקציה מ $3NAE-SAT$. בהנתן φ נוסחת $3NAE-SAT$ בעלת n משתנים ו m פסוקיות נבנה גרף באופן הבא: הגרף יהיה איחוד של n גרפים דו צדדים מלאים, שבכול צד יש 3m קודקודים, כך שכול גרף שייך למשתנה אחד, וצד אחד של הגרף זה המופעים שלו בחיוב והשני בשלילה. כעת לכול פסוקית בנוסחא נחבר את הליטרלים שלה בקשת, כאשר לא יהיו קשתות כפולות כי לכול פסוקית יש את השורה שלה בגרף הדו צדדי של המשתנה הזה (יש לו m שורות בכול צד שבכול שורה 3 קודקודים). כעת נשים לב שבגרף הזה יש חתך בגודל $n * (3m)^2 + 2m$ אמ"מ הנוסחא היא $3NAE-SAT$.

2SAT: שפת כול הנוסחאות מצורת $2CNF$ שיש להן השמה מספקת. נשים לב שהשאלה האם נוסחא כזו היא ספיקה היא שאלה קלה, ובפרט $2CNF \in NL$. השאלה הזו שקולה למציאת מסלול בגרף ההשלכות. כלומר נגדיר גרף מכוון, שלכול ליטרל בנוסחא שלנו יהיה קודקוד, והקשת בין הליטרלים (α, β) תהיה אם כאשר α מקבל ערך אמת זה חייב גם כן לקבל ערך אמת כדי שהנוסחא תהיה ספיקה. למשל אם יש את הפסוקית $(\alpha\ \beta)$ בנוסחא שלנו אז הקשתות $(\bar{\alpha}, \beta), (\bar{\beta}, \alpha)$ יופיעו בגרף (זה קשתות מכוונות). בהינתן נוסחא לבנות את הגרף הזה לוקח מקום לוגריתמי, וכעת נותר רק לוודא כי בגרף הזה אין מסלול מליטרל לשלילתו, ואת זה אפשר לעשות על ידי מעבר סידרתי על כול הקודקודים ושימוש פעמיים ב $ST - Conn$.

Shortest Path with Forbidden Pairs:

נתון גרף G ושני קודקודים s, t וקבוצה של זוגות (u_i, v_i) והשאלה היא האם יש מסלול מ s ל t שלא עובר גם u_i וגם v_i לכול i. נשים לב שהבעיה הזו היא NPC. תחילה ברור שהיא ב NP כי מסלול כזה הוא עד, וקל לוודא שהוא נכון.

מצד שני יש רדוקציה מ $3SAT$. בהינתן נוסחא φ לכול פסוקית $(x_1\ x_2\ x_3)$ נתנן שלושה קודקודים בגרף. קודקוד אחד ייצג את הפסוקית, וקודקוד נוסף לכול ליטרל, כך שמהקודקוד שמייצג את הפסוקית יש קשת לכול אחד מהקודקודים של הליטרלים, ומכול קודקוד של ליטרל יש קשת לקודקוד שמייצג את הפסוקית הבאה בתור. הזוגות האסורים יהיו קודקודים שמייצגים משתנה ושלילתו, וכעת הנוסחא ספיקה אם ורק אם יש מסלול בגרף החדש מקצה לקצה.

באופן דומה קל לראות שאם היינו מחפשים לא רק קיום של מסלול אלא למצוא את הקצר ביותר, אז לקרב את הבעיה הזו לכול פקטור קבוע זה $NP - Hard$. כי נניח והיינו יכולים לקרב הבעיה הזו, אז בהנתן נוסחת $3SAT$ היינו בונים את הגרף, מפעילים את אלגוריתם הקירוב, אם הנוסחא לא ספיקה אז אין מסלול, אז הקירוב יחזיר שהאורך הוא לכול היותר c הקבוע שלנו, בעוד שאם הנוסחא ספיקה הוא יחזיר שיש מסלול באורך לכול היותר $n * c$ ואז נדע שהנוסחא ספיקה.

Diagonalization:

Space Complexity:

מכונות טיורינג חסומות מקום: למכונות טיורינג חסומות מקום יש 3 סרטים, סרק קלט סרט עבודה וסרט פלט. כאשר סרט הקלט הוא לקריאה בלבד, סרט העבודה הינו סרט רגיל, וסרט הפלט הוא לכתיבה בלבד, ואי אפשר לחזור בו אחורה. המקום שבו המכונה משתמשת זה אך ורק המקום שהיא תופסת בסרט העבודה.

מכונת טיורינג לא דטרמינסטית חסומת מקום: יהיו לה שלושה סרטים. סרט קלט שהוא לקריאה בלבד, סרט עבודה שהוא סרט רגיל אבל חסום במקום, וסרט עד שהוא לקריאה בלבד ואסור לחזור בו אחורה. אם קיים עד שגורם למכונה לקבל נאמר שהמילה בשפה.

נשים לב שחשוב מאוד כי אסור לחזור אחורה בעד, כי אחרת המחלקה NL הייתה שווה ל NP כי אפשר לוודא את $3SAT$ באופן הזה.

הגדרות/סימונים:

- $Space[t(n)] = \{L | L\ is\ decided\ by\ O(t(n)) space\ determinstic\ turing\ machine\}$
- $NSpace[t(n)] = \{L | L\ \ decided\ by\ O(t(n)) Non\ Determinstic\ Turing\ machine\}$

- $L = Space[\log(n)]$
- $NL = Nspace[\log(n)]$
- $Pspace = \cup_k Space[n^k]$

<u>חסם על מקום מניב חסם על זמן:</u>

נשים לב כי עבור מכונת טיורינג עם אלפבית סרט Γ ואלפבית Σ אז מספר הקונפיגורציות השונות שיכולות להיות לה הוא $\Sigma^n * N * \Gamma^S * S * Q$ כאשר n זה אורך הקלט, ו S זה החסם על מקום הריצה שיש למכונה, ו Q זה מספר המצבים השונים שיש לה. ולכן מכונה חסומת מקום לא יכולה לרוץ זמן יותר ארוך מהמספר שמופיע לעיל כי אחרת היא תכנס ללולאה. לכן למשל מכונות חסומות מקום לוגריתמי רצות בזמן פולינומיאלי.

<u>רדוקציות חסומות מקום:</u>

נאמר כי $A \leq_L B$ אם קיימת פונקציה $f: \Sigma^* \to \Sigma^*$ חשיבה במקום לוגריתמי (כלומר קיימת מכונת טיורינג חשיבה במקום לוגריתמי שפולטת בסרט הפלט שלה את f) כך ש $w \in A \leftrightarrow f(w) \in B$.

נשים לב כי L סגורה לרדוקציות כאלה, כלומר $B \in L$ וגם $A \leq_L B$ אז $A \in L$ כי כדי להכריע את A לכאורה נריץ את המכונה שמכריעה את B על $f(w)$ כאשר f היא הרדוקציה. אבל יכול להיות שהפלט של f אינו לוגריתמי. לכן נוציא כול פעם אות בודדת מהפלט של f כאשר המכונה שמכריע את B תזדקק לאות הבאה. כך אנחנו לא משתמשים ביותר ממקום לוגריתמי. וכדי לזכור איזה אות צריך להוציא כעת יהיה לנו קאונטר שיתפוס מקום לוגריתמי.

<u>S-T Connectivity:</u>

$$STConn = \{(G, s, t) | G \ is \ \boldsymbol{directed} \ and \ there \ is \ a \ path \ from \ s \ to \ t\}$$

קל לראות כי $Conn \in NL$ כי פשוט נתחיל מקודקוד s וכול פעם באופן לא דטרמניסטי נבחר להתקדם לקודקוד הבא במסלול. כול מה שצריך לזכור זה את המיקום הנוכחי, וזה דורש מקום לוגריתמי. באופן שקול אפשר לחשוב על עד שהוא המסלול ולוודא אותו זה פשוט ללכת בעקבותיו במקום לוגריתמי.

<u>$STConn \in NL - Complete$:</u>

כלומר לכול שפה $L \in L$ מתקיים $L \leq_L ST - Conn$ ואכן לכול שפה ב L קיימת מכונת טיורינג רצה במקום לוגריתמי שמכריעה אותה, נסמנה M. נראה פונקציה f חשיבה במקום לוגריתמי כך ש $f(w) \in Conn \leftrightarrow w \in L$. הפונקציה f תבנה גרף קונפיגרציות שהוא גרף המכיל את כול הקונפיגרוציות האפשריות למכונה M כשהיא מתחילה על הקלט w, ובין כול שתי קונפיגרציות שהן הקודקודים של הגרף תהיה קשת אם אפשר לעבור מאחת לשנייה. כעת קל לראות שהמכונה מקבלת את w אם ורק אם יש מסלול בגרף הזה מהקודקוד ההתחלתי לקודקוד שהוא המצב המקבל (בה"כ יש קונפיגורציה מקבלת יחידה...). כאשר f רצה במקום לוגריתמי כי כמו שראינו יש כמות פולינומיאלית של קונפיגורציות ולכן למנות עליהם דורש מקום לוגריתמי, כאשר f מונה עליהם פעמיים, ולכול זוג בודקת האם אפשר לעבור מאחת לשנייה, גם זה במקום לוגריתמי כי זה בדיקות מקומיות.

<u>$\textbf{Directed} - \textbf{Cycle} \in NL - Complete$</u>: הינה הבעיה בהנתן גרף מכוון וקודקוד, האם יש מעגל שעובר דרך הקודקוד הזה. תחילה ברור כי זה ב NL כי העד יהיה מעגל שעובר בקודקוד, וכדי לוודא אותו פשוט נעקוב אחריו עד שנחזור להתחלה.

זה גם $NL-HARD$ כי נראה רדוקציה ממסלול. בהנתן גרף G וקודקודים s,t נהפוך את הגרף לגרף שכבות, ונחבר חזרה ל s רק מופעים משוכפלים של t, אז בגרף החדש יש מעגל אמ"מ היה מסלול בגרף המקורי מ s ל t.

<u>NL ⊆ P:</u>

נובע מיידית מהטענה לעיל כי כמו שאנחנו יודעים $ST-Conn \in P$ (BFS/DFS/Dijkstra) ולכן כול שפה ב NL אפשר לעשות רדוקציה בזמן פולינומיאלי ל $ST-Conn$ ואז לפתור ב P.

<u>משפט Savitch: $NL \subseteq Space[log^2(n)]$</u>

כלומר אי דטרמיניזם לא מוסיף הרבה כוח לחישובים מוגבלי זיכרון.

ואכן קל לראות כי $ST-Conn \in Space[\log^2(n)]$ כי כדי לדעת האם יש מסלול בין u,v בלכול היותר V קודקודים נמנה על כול הקודקודים בגרף, ולכול $w \in V$ נשאל האם יש מסלול מ u לw ומw לv כול אחד בלא יותר מ $\frac{V}{2}$ קודקודים, וכך נמשיך ברקורסיה. עומק הרקורסיה בבירור לוגריתמי כי מחלקים המרחק בשתיים כול פעם, ובכול שלב צריך זכרון לוגריתמי, רק צריך מונה על הקודקודים. לכן סה"כ מקום לוגריתמי.

ומשום ש $ST-Conn \in NLC$ קיבלנו כי $NL \in Space[\log^2(n)]$.

<u>ארגומנט הניפוח:</u>

לכול שתי פונקציות $s_1, s_2: N \to N$ גדולות מלוג ולכול פונקציית ניפוח $e(n) \geq n$ אם מתקיים $NSpace[s_1(n)] \subseteq Space[s_2(n)]$ אז $NSpace[s_1(e(n))] \subseteq Space[s_2(e(n))]$

ואכן תהי $L \in NSpace[s_1(e(n))]$ נגדיר שפה חדשה $L^e = \{x\#^{e(|x|)-|x|} | x \in L\}$ אזי האורך של מילה בשפה הזו הוא $e(|x|)$ כאשר x זה הקלט המשמעותי. לכן השפה $L^e \in NSpace[s_1(n)]$ כי נקח את המכונה הלא דטרמינסטית שמכריעה את L במקום $e(n)$ כאשר n הקלט שלה, וניתן לה כקלט מילה ב L^e והיא תרוץ רק על החלק המשמעותי, אז היא תרוץ במקום $s_1(e(|x|))$, אבל $e(|x|)$ זה אורך המילה כולה, לכן היא למעשה רצה במקום $s_1(n)$ כאשר n זה $e(x)$, אורך המילה ב L^e . אנו יודעים הכלה אחת ולכן $L^e \in Space[s_2(n)]$ ולכן קיימת לה מכונה M שרצה במקום כזה. לכן כדי להכריע את L במקום $s_2(e(n))$ אז בהנתן קלט x נסמלץ את M ונטעה אותה לחשוב שיש כוכביות אחרי המילה x, M עצמה תרוץ במקום $s_2(x\#^{e(|x|)-|x|})$ ולכן המכונה עצמה רצה במקום $s_2(e(|x|))$ כנדרש.

<u>מסקנות:</u>

שילוב של ארגומנט הניפוח ומשפט סאביץ נותר לנו כי $NSpace[s(n)] \subseteq Space[s^2(n)]$ כי נבחר $e(n) = 2^{s(n)}$ ולכן משום ש $Nspcae[\log(n)] \subseteq Space[\log^2(n)]$ נקבל כי מתקיים $Nspace[\log(2^{s(n)})] \subseteq Space[\log^2(2^{s(n)})]$ ולכן $Nspace[s(n)] \subseteq Space[s^2(n)]$ כנדרש.

בפרט מתקיים $NPspace = PSpace$

<u>משפט אימרמן:</u> $NL = CO-NL$

נשים לב כי $NON-Conn$ שהיא כול הגרפים וזוג קודקודים כך שאין מסלול בין הקודקודים בגרף מקיימת $NON-Conn \in CO-NL\ Complete$ כי בבירור $NON-Conn \in Co-NL$ וכמו כן לכול $L \in Co-NL$ מתקיים $\bar{L} \leq_L Conn \to L \leq_L NON-Conn$.

לכן אם נראה כי $NON - Conn \in NL$ נקבל השוויון הרצוי. כי אז נקבל בבירור כי $Co - NL \subseteq NL$ ומלקיחת משלים נקבל גם הצד השני, כי נקבל $Conn \in Co - NL$.

ואכן נראה עד לכך שאין מסלול בגרף, שניתן לוודא אותו במקום לוגריתמי. ואכן עבור מילה (G, s, t) נגדיר $Reachable(G)$ להיות כול הקודקדוים בגרף שניתן להגיע אליהם מ s. ולכן אם נצליח להוכיח כי $|Reachable(G)| = |Reachable(G/t)|$ זה יספיק כדי להוכיח שאין מסלול ל t. ולכן מספיק להביא עד שמוכיח כי $Reachable(G) = r$ עבור r כלשהו, והמכונה תוכל להשוות בין הערכים עבור G ועבור G/t במקום לוגריתמי ולוודא שוויון.

לכן נגדיר $Reachable_l$ להיות כול הקודקודים שנגישים מ s במסלול באורך לכול היותר l. כעת נבנה עד באופן אינדוקטיבי שיוכל להוכיח את הערך $Rachable_i$ באופן שאפשר לוודא במקום לוגריתמי, והוא ישתמש בעובדה שניתן כבר לוודא את $Reachable_k$ לכול $k \leq i$.

בסיס: אכן עבור r_0 קל מאוד לבנות עד בניתן לוודא במקום לוגריתמי לערך של r_0 והוא פשוט השכנים של s.

צעד: בהנתן עד לכך ש $Reachable_l = r_l$ נרחיב אותו לעד עבור הערך של r_{l+1} באופן הבא. העד שלנו ישורשר לעד הקודם, והוא יהיה וקטור של אפסים ואחדות, באורך מספר הקודקודים כך שלכול קודקוד בגרף יהיה אפס או אחד בהתאם להאם הוא שייך ל $Reachable_{l+1}$. לאחר כול ביט כזה תופיע עדות לכך שהביט אכן נכון. אם הביט הוא אחד, כלומר יש מסלול בינהם העדות תהיה המסלול. אם הביט הוא אפס אז נכתוב שוב וקטור ביטים לכול הקודקודים שהפעם מציינים שייכות ל $Reachable_l$ ושוב כול ביט שהוא אחד נוכיח על ידי המסלול, אבל הפעם ביטים שהם אפס לא נוכיח, והנכונות שלהם תנבע מכך שהערך של R_l כבר ידוע, ולכן אם הצלחנו להוכיח כבר שייכות של R_l קודקודים ל $Reachable_l$ מה שנשאר חייב להיות לא שייך ל $Reachable_l$. לכן הצלחנו להוכיח שייכות ואי שייכות ל $Reachable_l$ ולכן להוכיח אי שייכות ל $Reachable_{l+1}$ פשוט בודקים שאין קשת מהקודקוד שלנו לאף קודקוד ב $Reachable_l$ ובכך נסיים.

<u>מסקנה:</u> מחלקות סיבוכיות מקום סגורות למשלים, נובע מארגומנט הניפוח. כלומר
$NSpace\big(S(n)\big) = coNspace\big(s(n)\big)$

<u>TQBF:</u> שפת כול הנוסחאות הבוליאניות שמכומתות לגמרי, כלומר אין משתנים חופשיים, כך שהן תמיד אמת.

<u>$TQBF \in PSpace - Complete$:</u>

תחילה נראה כי $TQBF \in PSpace$. בהינתן נוסחה האלגוריתם בודק האם יש בה משתנים, אם אין אז הוא מחזיר את ערכה המספרי (0,1) אם יש בה משתנים בהכרח יש כמתים. נתבונן בכמת הראשון, אם הוא לכול אז נחזיר ברקורסיה את הAnd של להציב בו אחד ולהציב בו אפס ולפתור בעזרת האלגוריתם. אם הוא קיים אז נחזיר את הערך של ה Or של שני ענפי הרקורסיה. בבירור עומק הרקורסיה הוא כמספר הכמתים ולכן לינארי, ובכול שלב צריך לזכור ביט בודד של האם לעשות or או And כשהרוקרסיה תחזור. לכן סה"כ בכול רגע נתון צריך רק מקום לינארי.

כעת נשים לב כי $TQBF \in Pspace - Hard$ וקלט נבנה נוסחה שתהיה PSPace בעלת ערך אמת אמ"מ הקלט בשפה. הנוסחא תקודד האם יש מסלול מקונפיגורציה התחלתית לקונפיגורציה מקבלת במכונה שמכריעה את השפה במקום פולינומיאלי. הבעיה היא שמסלולים יכולים להיות גדולים אקספוננציאלית ולכן הנוסחה שלנו $\varphi(u, v, d)$ תהיה אמת אם ורק אם יש מסלול מ u ל v בפחות מ 2^d צעדים והיא תהיה
$\varphi(u, v, d) = \exists w \forall u' \forall v' ((u' = u \; and \; v' = w) or (u' = w \; and \; v' = v) \rightarrow \varphi(u', v', d - 1)$

או במילים אחרות קיים קודקוד אמצע במסלול כך שיש מסלול מהקודקוד המקורי אליו, וממנו לקודקוד הסופי בחצי מהמרחק $\frac{2^d}{2} = 2^{d-1}$ וכך הבטחנו שהנוסחה תהיה לינארית בקלט, ובכך סיימנו. אזי לכול שפה $L \in Pspace$ הרדוקציה שלנו f מקיימת $f_L(x) = \varphi(start(x), accept, \log(config))$ כלומר האם יש מסלול ממצב התחלתי למצב מקבל בפחות ממספר הקונפיגורציות השונות שיש למכונה. בבירור הנוסחא ספיקה אמ"מ המילה בשפה.

Approximation Problems:

עד היום ראינו בעיקר בעיות הכרעה, מעתה נדבר על בעיות אופטימיזציה, כלומר מנסים למצוא את התוצאה המקסימלית/ מינימלית. למשל קליקה מקסימלית, השמה מספקת מקסימלית וכו'. בכול בעיה כזו יש פרמטר שאותו רוצים להביא למינימום/ מקסימום.

<u>קירוב:</u> עבור בעיית אופטימיזציה נגדיר אלגוריתם שהוא α קירוב אם הוא מקיים שהפלט שלו הוא טוב לפחות כמו $\alpha * M$ כאשר M הינו הפתרון האופטימלי לבעיה. כלומר אם זו בעיית מקסימום אז הפלט של האלגוריתם גדול לפחות כמו αM (בפרט α קטן מאחד) כמו כן אם זו בעיית מינימזציה אז הפלט של האלגוריתם מקיים שהוא קטן לפחות כמו $\alpha * M$ כאשר α גדול מ 1

<u>בעיית GAP:</u> בהנתן בעיית אופטימזציה *A* בעיית ה *GAP* המתאימה $GAP - A[a,b]$ היא בהנתן מופע של הבעיה *X*, אם הפתרון האופטימלי של *A* עבור *X* הוא לפחות טוב כמו הסף העליון(במקרה של מקסימום, והתחתון במקרה של מינימום) כלומר *A(X)>b* אז בעיית ה*GAP* היא לדעת להבחין בכך. וכמו כן אם *A(X)<a* (שוב עבור מינימום זה ההפך) אז בעיית ה*GAP* צריכה להיות מסוגלת להבחין בכך. אבל אם $a < A(x) < b$ אז אין חשיבות לתשובה שפותר בעיית ה *GAP* מחזיר על מופע זה. כלומר אינטואטיבית בעיית ה *GAP* היא להבדיל בין מקרים טובים מאוד, לבין מקרים רעים מאוד, כאשר ככול שהפער גדול יותר כך הבעיה קלה יותר. (עבור *GAP* בגודל 0 זה למעשה בעיית הכרעה)

<u>קשר בין קירוב ל GAP:</u>

נשים לב כי אם ידוע לנו α קירוב פולינומיאלי לבעיה *A* אז בעיית ה *GAP* עבור שפה *A* והספים $[\alpha\delta, \delta]$ היא בעיה קלה (מדובר במקסימום). כי בהנתן מופע *X* של הבעיה נריץ את הקירוב הפולינומיאלי שלנו. אם ערך הקירוב, שנסמנו *M* מקיים $M < \alpha\delta$ אז בבירור הפתרון האופטימלי למופע *X* קטן מ δ בגלל שידוע לנו טיב הקירוב, לכן אפשר לדחות את *X* בבטחה. בעוד שאם $M \geq \alpha\delta$ אז בפרט הפתרון האופטימלי גדול מכך ולכן ניתן לקבל בבטחה (ולא לדאוג עבור הערכים שהם בתוך הפער)

<u>דוגמא Vertex Cover:</u>

עבור גרף *G* נגדיר קבוצת קודקודים *C* להיות *Vertex Cover* אם לכול קשת בגרף מתקיים כי לפחות אחד מקודקודיה נמצא בקבוצה *C*. נחפש *Vertex Cover* מינמלי (בבירור כול *V* הינו *Vertex cover*, אבל ממש לא מינמלי).

נשים לב כי אם *C* היא *Vertex Cover* אז המשלים שלה הוא *IS* ולכן זו בעיה *NPC*. (כי *IS* קשה כי זה *Clique* בגרף המשלים, וראינו ש*Clique* היא *NPC* על ידי רדוקציה ל *3SAT*)

<u>אלגוריתם קירוב:</u> נבנה כיסוי באופן הבא, נתחיל מהכיסוי הריק וכול עוד יש קשת לא מכוסה, כלומר שני קודקודיה אינם בכיסוי שאנחנו בונים, אז נוסיף את שניהם לכיסוי ונוציא את כול הקשתות שנוגעות בהם מהגרף. ברור שבסוף התהליך קיבלנו כיסוי כי קשת הוסרה מהגרף רק אם היא נגעה בקודקוד שהוספנו לכיסוי. כמו כן בבירור האלגוריתם רץ בזמן פולינומיאלי. נשים לב כי הכיסוי האופטימלי חייב להכיל לפחות קודקוד אחד מכול קשת, משום שהאלגוריתם כול פעם מוסיף זוג קודקודים שלא חולקים קשת עם אף קודקוד שהוא הוסיף בעבר, ובין שניהם עוברת קשת, אז לפחות אחד מהם יהיה בכיסוי הסופי, ולכן משום שלקחנו את שניהם קיבלנו 2 קירוב לבעיה.

<u>דוגמא Set-Cover:</u>

נתונה קבוצה סופית U ומשפחה של תת קבוצות של F,U. כלומר $F \subseteq 2^U$. המטרה היא למצוא תת קבוצה C של F כך שאיחוד כול האיברים בקבוצת C יכסה את כל U. והמטרה היא למצוא את התת קבוצה המינימלית, כלומר זו שתכיל כמה שפחות קבוצות מתוך F.

נשים לב שבעיה זו היא NP קשה כי יש רדוקציה פשוטה מ VC. העולם שלנו U יהיה הקשתות שאותן נרצה לכסות. ומשפחת הקבוצות שלנו תהיה F כך שכול $A_v \in F$ הינה קבוצה כול הקשתות שנוגעות בקודקוד v. בבירור יש VC בגרף אמ"מ יש Set Cover.

<u>אלגוריתם קירוב:</u> הרעיון הוא אלגוריתם חמדן שבו כול פעם לוקחים את הקבוצה מתוך המשפחה שהחיתוך שלה עם מה שעוד לא מכוסה הוא הגדול ביותר. כלומר מתחילים מכיסוי ריק C וכול פעם מוסיפים אליו את הקבוצה $S \in F$ כך שS מביא למקסימום מבין כול שאר הקבוצות ב F שעוד לא לקחנו את $S \cap (U \setminus \cup C)$. בבירור האלגוריתם רץ בזמן פולינומיאלי כי כול קבוצה שמוסיפים מקטינה באחד לפחות את מה שנשאר.

ניתוח הקירוב:

- זה $\ln |U|$ קירוב. ואכן אם הכיסוי הטוב ביותר הוא בגודל K, אזי גם לכול תת קבוצה של U יש כיסוי בגול K לכול היותר, ולכן בכול שלב באלגוריתם אם נסמן ב U' את החלק שנותר עוד לא מכוסה אז יש לו כיסוי ב K קבוצות, ולכן לפחות אחד מהן מכילה $\frac{1}{k}$ מהאיברים שב U'. משום שהאלגוריתם לוקח את הגדולה ביותר, בשלב הבא נשאר עם לכול היותר $\left(1-\frac{1}{k}\right)U'$ ולכן בצעד ה ik מתקיים $|U_{ik}| \leq |U|\left(1-\frac{1}{k}\right)^{ik} \leq \frac{|U|}{e^i}$ ולכן עבור $i = \ln|U| + 1$ צעדים האלגוריתם מסיים, לכן הוא לקח קבוצה בגודל $k(\ln|U|)$ ולכן זה $\ln|U|$ קירוב לאופטימום.
- הקירוב אפילו יותר טוב, הוא כמו $Harm(S)$ עבור $S \in F$ הגדולה ביותר. נתבונן בפעולת האלגוריתם, בכול שלב הוא מוסיף קבוצה חדשה לכיסוי, וכאמור משלם מחיר של 1 עבור הוספתה. אם נחלק את המחיר הזה בין כול החברים החדשים שהצטרפו לכיסוי, כלומר כול מי שבקבוצה החדשה שאוספנו ועוד לא מכוסה כבר, אז כול אחד ישלם $\frac{1}{|s|}$ כאשר $|s|$ זה כמות האיברים החדשים שכוסו בשלב זה של האלגוריתם, ומשום שהאלגוריתם בוחר את הקבוצה עם החיתוך המקסימלי אז לא קיימת קבוצה שתניב $\frac{1}{|s|}$ קטן יותר. לכן אם נתבונן בכיסוי האופטימלי C, זה שאנחנו לא יודעים, ונדמיין איך לאט לאט האיברים נלקחים ומכוסים על ידי הכיסוי שהאלגוריתם נבנה, וכול פעם שמישהו נלקח הוא "משלם" את המחיר כפי שהוסבר לעיל, אז הכי הרבה שתשלם קבוצה כלשהי זה המספר ההרמוני שלה, ולכן סה"כ ששילמנו זה סכום המספרים ההרמונים של הקבוצות בכיסוי האופטימלי שזה פחות מגודל הכיסוי האופטימלי כפול המספר ההרמוני הגדול ביותר, ולכן זה הקירוב שמקבלים.

<u>דוגמא Traveling Salesman Problem:</u>

נתון גרף מלא וממושקל במשקולות אי שלילות, המטרה היא למצוא מעגל המליטון בעל המשקל הזול ביותר. בבירור הבעיה הזו $NP-Hard$ כי בהנתן גרף כלשהו, אפשר להוסיף לקשתות שלו משקל 1, ולהוסיף את הקשתות שאין לו ולתת להם משקל 5 ואז יש בו מעגל המליטוני אמ"מ הפתרון לבעיית הסוכן הנוסע היא מעגל ממשקל מספר הקודקודים.

- **<u>2-קירוב כאשר יש אי שוויון המשולש:</u>**
כלומר שלכול שלושה קודקודים $u,v,w \in V$ מתקיים $c(u,v)+c(v,w) \geq c(u,w)$, כאשר c זה משקל הקשת. האלגוריתם הוא למצוא עפ"מ לגרף בזמן פולינומיאלי בעזרת למשל אלגוריתם Kruskal. לאחר מכן נעבור על העץ שקיבלנו ב DFS ונוסיף את הקודקודים בדרך למעגל שלנו, כאשר אם עברנו בקודקוד שכבר בקרנו בו אז פשוט נדלג מעליו ונמשיך (הגרף

מלא). בברור המשקל של העפ"מ קטן ממשקל המעגל הזול ביותר (כי במקרה הגרוע ביותר העפ"מ יהיה חלקי למעגל, פחות קשת). כמו כן קל לראות שהמשקל של ללכת לאורך בעץ ב DFS הוא לכול היותר פי שתיים ממשקל העץ כי בכול קשת עוברים בדיוק פעמיים, ועבור קודקודים שקפצנו מעליהם רק הרווחנו כי מתקיים א"ש משולש. לכן משקל המעגל שמצאנו הוא לא יותר מפי שתיים ממשקל העץ.
$Tree\ weight \leq best\ Cycle \leq our\ Cycle \leq 2 * Tree\ weight$ ולכן זה שתיים קירוב.

- **<u>לא ניתן לקירוב במקרה הכללי לשום קבוע h:</u>**
נראה כי לכול $h \geq 1$ מתקיים $gap - TSP[|V|, h|V|]$ היא NP-HARD ולכן לא ניתן לקרב בזמן פולינומיאלי (אלא אם $P = NP$). וכמו כן קל לראות כי בהנתן גרף כלשהו, ניתן לקשתות שלו משקל 1, ולכול קשת שלא קיימת בו נוסיף אותה ונתן לה משקל $h|V|$. לכן אם בגרף המקורי היה מעגל המילטוני את הגרף החדש נמצא מתחת לסף התחתון של בעיית ה GAP, ואם לא היה בו מעגל המילטוני אז הפתרון מעל הסף העליון. בפרט להיות מסוגלים להבדיל בין הספים יאפשר לנו לפתור את HAM-CYCLE. לכן זה $NP - Hard$ להבדיל בין הספים.

<u>משפט ה PCP:</u>

נתבונן בבעיה $GAP - 3SAT[\frac{7}{8} + \varepsilon, 1]$ עבור ε מספיק קטן. (בחרנו דווקא $7/8$ כי עבור נוסחת SAT שבה בכול $cluase$ יש שלושה משתנים בלתי תלויים, אז בוודאות $7/8$ ממנה ניתנים לסיפוק, משום שאם נבחר השמה אקראית, היא מספקת בתוחלת $7/8$ מהליטרלים). אזי משפט ה PCP אומר שבעיית הGAP הזו היא NP-hard, כלומר לכול $L \in NP$ קיימת רדוקצית Karp ל$3CNF$ כך שאם הקלט בשפה אז הנוסחא ספיקה, ואם הקלט לא בשפה אז לא ניתן לספק יותר מ $\frac{7}{8} + \varepsilon$ מהליטרלים.
בפרט, בהנתן נוסחת 3SAT, למצוא לה קירוב $\frac{7}{8} + \varepsilon$ להשמה הטובה ביותר שלה היא בעיה קשה, כי אחרת היינו יכולים לפתור את בעיית הפער.

הרעיון הוא שניתן להוכיח ידיעת פתרון של שפה ב NP בעזרת כמות קבועה של ביטים אקראיים בהסתברות טובה מאוד. כלומר אם מישהו טוען שיש בידיו פתרון לבעיה שהיא NP אז הוא ימיר בעזרת הרדוקציה שהמשפט טוען שקיימת לנוסחת 3SAT. אם אכן יש בידיו פתרון אז הוא ידע את ההשמה המספקת לנוסחא, אבל אם אין בידיו פתרון הוא יוכל לדעת איך לספק לכול היותר $\frac{7}{8} + \varepsilon$ מהפסוקיות. לכן אם אני אדגום באקראי מספר קבוע של פסוקיות, הסיכוי שהוא הצליח לעבוד עלי הוא $1/8$ במספר הפסוקיות שדגמתי.

<u>רדוקציה משמרת GAP:</u> ניתן לעשות רדוקציה בין שתי שפות GAP, וזו תהיה פונקציה חשיבה בזמן פולינומיאלי כך שמופע טוב של השפה הראשונה עובר למופע טוב של השנייה, ומופע רע עובר למופע רע. לגבי המופעים שהם בתוך הפער אין חשיבות למה קורה. בבירור כי אם A בעיית GAP שהיא NP-HARD, וכמו כן מתקיים שיש רדוקציה משמרת GAP מ A ל B גם היא בעיית GAP אז B גם NP-HARD.

<u>דוגמא</u> $3SAT\left[\frac{7}{8} + \varepsilon, 1\right]$ מ$GAP - Clique\left[\frac{\frac{7}{8}+\varepsilon}{3}, \frac{1}{3}\right]$בעיה NP קשה. אכן נראה רדוקציה משמרת *GAP*

בהנתן נוסחת $3CNF$ נבנה גרף שיורכב משלשות, כול שלשה תייצג את *Clause* אחד, והקודקודים בשלשה יהיו הליטרילם של ה *Clause* הזה. בתוך כול שלשה לא יהיו קשתות, ובין שלשות יהיו קשתות בין שני קודקודים אמ"מ הם לא סתורים, כלומר זה לא משתנה ושלילתו. בבירור יש השמה מספקת בגודל L, כלומר שמספקת לפחות L מהפסוקיות אמ"מ יש קליקה בגודל לפחות $\frac{L}{3}$

מהקודקודים של הגרף, כי השמה מתרגמת לבחירה משתנה אחד מכול *Cluase* שיהיה אמת, ושאינו סותר את כול שאר ההשמות ב*Cluase* אחרים. ולכן מהקושי הידוע על $3SAT\left[\frac{7}{8}+\varepsilon,1\right]$ לפי משפט ה *PCP* נקבל הנדרש.

<u>דוגמא:</u> $gap-Max2SAT\left[\frac{55}{80}+\varepsilon,\frac{56}{80}\right]$ היא $NP-Hard$. זה נובע ישירות מרדוקציה מ $3SAT\left[\frac{7}{8}+10\varepsilon,1\right]$. רדוקציה מאוד טכנית.

<u>גרף אילוצים:</u>

בעיית אופטימיזציה שנתונה על ידי (V,E,Σ,φ) כאשר $<V,E>$ גרף, Σ רשימה של צבעים, ו $\varphi: E\to P(\Sigma^2)$ פונקציית אילוצים, כך שלכול קשת היא מתאימה קבוצה של זוגות צבעים חוקיים לצביעת קצוותיה של הקשת. (כאשר הזוגות מכוונים, כלומר לא בהכרח הצביעות החוקיות הן סימטריות. אבל זה לא אומר שהקשתות עצמן מכוונות בהכרח).

פתרון לבעיה הינה צביעה, $A: V\to\Sigma$, של הקודקודים, וניתן לבצע אופטימזציה על אחד משני הדברים הבאים:

- $Max-CGS_V$: הצבעיה A היא פונקציה לא מלאה אולי, כלומר חלק מהקודקודים לא קיבלו צבע, אבל בהכרח כול האילוצים מתקיימים, כלומר אם שני קודקודים שיש בינהם קשת קיבלו צבע, אז הצבעים תואמים את האילוצים על הקשת הזו. המטרה היא למקסם את מספר הקודקודים שקיבלו צבע.
- $Max-CGS_E$: הצביעה A היא צביעה מלאה של כול הקודקודים, אבל אולי חלק מהקשתות לא מקיימות את האילוצים לפי φ. המטרה היא למקסם את מספר הקשתות שכן מקיימות את האילוצים.

<u>קשיות של CSG:</u>

נשים לב כי $gap-3CSG_V\left[\frac{7}{8}+\varepsilon,1\right]$ שהיא הבעיה בהינתן גרף CSG שבו יש שלושה צבעים, להבחין האם הצביעה האופטימלית במובן של קודקודים צובעת את כול הקודקודים, או לא צובעת יותר מ $\frac{7}{8}+\varepsilon$, היא בעיה NP קשה. על ידי רדוקציה ל $3SAT\left[\frac{7}{8}+\varepsilon,1\right]$, כך שהקודקודים בגרף יהיו $clause$ בנוסחא. לכול $clause$ ניתן אחד משלושה צבעים לפי מי מהליטרלים בו קיבל ערך אמת. כאשר האילוצים יהיו בין ה $clause$ כך שיבטיחו שמשתנה ושלילתו לא קיבלו שניהם ערך אמת. בבירור כמות הclause שמסתפקים בהשמה כלשהי לנוסחה שווה לכמות הקודקודים שההשמה הזו צובעת באופן חוקי, לכן זו רדוקציה משמרת GAP.

<u>הרחבת הפער Amplification:</u>

נראה שניתן להרחיב את הפער בבעיית הצביעה תחת אילוצים, אולם נשלם במחיר הגדלת מספר הצבעים. ואכן הטענה היא $gap-kCSG_V[\delta,1]\leq_L gap-k^lCSG[\delta^l,1]$ אזי אנחנו מגדילים את הפער (כי $\delta<1$ אז $\delta^l<\delta$) אבל אנחנו צריכים הרבה יותר צבעים.

הרדוקציה היא כזו: בהינתן גרף CGS G שמיועד לצביעה ב K צבעים, נגדיר גרף חדש H. בגרף החדש יהיו $\left|V(G)\right|^l$ קודקודים, כך שכול קודקוד של H הינו וקטור באורך l של קודקודים של G. נשים לב כעת שצביעה של קודקוד בגרף H זה למעשה לתת לו וקטור של l צבעים מקוריים. קשתות H יהיו בין שני קודקודים $\bar{v},\bar{u}$ (שניהם וקטורים באורך l) אם אחד הרכיבים של u ושל v זהה, כלומר אם קיים j,i כך ש $v_j=u_i$, והאילוץ על הקשת הזו יהיה שכשצובעים את הקודקודים הללו, אז בוקטור הצבעים שקיבלו u,v יתקיים שהצבע של הרכיב שזהה ביניהם יהיה אותו הצבע. כמו כן בין

שני קודקודים בגרף H תהיה קשת, אם בין איזהשהם שניים מהרכיבים שלהם היה אילוץ בגרף G, ואז האילוץ על הקשת הזו הוא שהאילוץ בגרף G יתקיים עבור הרכיבים של הוקטורים.

בבירור אם אפשר לצבוע את כול הקודקודים ב G אזי אפשר לצבוע את כול הקודקודים ב H, כי כול וקטור $\bar{u}$ שהוא קודקוד ב H נצבע אותו בצבע $\left(A(v_1), A(v_2), \dots, A(v_l)\right)$ כאשר A זו הצביעה של הגרף G. ברור שהצביעה הזו מקיימת את כול האילוצים כנדרש.

נשים לב שכול צביעה B' של קודקודי הגרף H תקיים שלכול קודקוד $\bar{v} \in H$ שהיא צובעת, כול אחד מהרכיבים שלו חייב להיצבע בגרף המקורי G. לכן הרכיבים שמרכיבים את ה l יות שניתן לצבוע הם רכיבים שניתן לצבוע בגרף המקורי. לכן אם בגרף המקורי אי אפשר לצבוע יותר מ M קודקודים אז בגרף החדש אי אפשר לצבוע יותר מ M^l קודקודים, כי זה מספר ה l יות השונות מעל M קודקודים. ולכן אם בגף המקורי אפשר לצבוע לכול היותר $\delta|V|$ קודקודים אז בגרף החדש אפשר לצבוע לכול היותר $\delta^l|V|^l$ כאשר $|V|^l$ זה מספר הקודקודים בגרף החדש, ולכן אכן ה GAP נשמר.

לכן לכול $\delta > 0$ אפשר לבחור K מספיק גדול, כך שהבעיה $gap - kCSG_V[\delta, 1]$ היא NP קשה, כי אנחנו יודעים ש $gap - 3CSG\left[\frac{7}{8} + \varepsilon, 1\right]$ היא NP קשה, ובעזרת הרדוקציה לעיל ננפח את $\frac{7}{8} + \varepsilon$ כך שיהיה δ (או פחות).

<u>קשיות של $gap - IS$:</u>

נשים לב כי מתקיים $gap - KCSG_V[\delta, 1] \leq gap - IS\left[\frac{\delta}{k}, \frac{1}{k}\right]$. כאשר $\delta < 1$ ו $K \in N$. כלומר לכול $\delta < 1$ אפשר לבחור K גדול מספיק שזה יהיה קשה. ולכן לכול δ קיים K כך שמתקיים $gap -$ $IS\left[\frac{\delta}{k}, \frac{1}{k}\right]$ היא NP קשה, ובפרט לקרב את $Max - IS$ ליחס δ לכול δ קבוע היא בעיה NP קשה, כי אחרת היינו יכולים לפתור את בעיית הפער.

בהינתן גרף CSG הנועד לצביעה ב K צבעים נבנה גרף חדש. עבור כול קודקוד v בגרף CSG, בגרף החדש יהיו K קודקודים, אחד לכול צבע אפשרי לצביעת v. כלומר כול קודקוד בגרף החדש הוא (v, i) כאשר v קודקוד של גרף ה CSG וi הוא צבע כלשהו. בין כול שני קודקודים $(v, i), (v, j)$ כלומר שמייצגים אותו קודקוד מקורי אבל שני צבעים שונים נשים קשת, ככה שקבוצה בלתי תלויה לא תוכל להכיל את שניהם יחד. בנוסף עבור כול אילוץ בגרף ה CSG המקורי נשים את הקשת המתאימה בגרף החדש. כלומר בגרף המקורי אסור לצבוע את u בצבע i ואת v בצבע j ביחד, אז בגרף החדש תהיה קשת בין $(v, i), (u, j)$.

כעת קל לראות כי אם ניתן לצבוע את כול הקודקודים בגרף ה CSG המקורי אז לכול קודקוד יהיה צבע, ולכן בגרף החדש נוכל לבחור אחד מבין כול K הקודקודים שמייצגים את הקודקוד המקורי, כך שביחד כול הקודקודים הללו לא מפרים הצביעה, ולכן אין בינהם קשתות, לכן יש IS בגודל $\frac{1}{K}$ מהקודקודים, כי מכול קבוצה בגודל K לקחנו אחד.

כמו כן בבירור כי אם יש IS בגודל לפחות $\frac{\delta}{k}$ אז מהיות IS הוא מכיל לכול היותר קודקוד אחד מכול קליקה בגודל K שמייצגת קודקוד. לכן הוא מכיל ב δ מהקליקות, אחד מכול קליקה, ולכן אם נצבע את הקודקודים בגרף המקורי בצבעים הללו, נקבל צביעה ל δ מהקודקודים.

<u>גרף אילוצי הפרשים:</u>

נתבונן במקרה פרטי של גרף אילוצים. נגדיר $qCSG_\Delta$ להיות (E, V, Σ, φ) כמו קודם, אולם כעת האילוצים על הקשתות תלויים אך ורק בהפרש בין הקצוות, כלומר $\Delta: E \to P([q])$, כלומר לכול קשת מתאימים קבוצה של הפרשים חוקיים, ואז הצביעות החוקיות לקשת הן אלו שמקיימות שהפרשי

הצבעים עומדים בהפרש, כלומר $\varphi(u,v) = \{(i,j) | i - j \ mod \ q \in \Delta(u,v)\}$. כלומר לקשת (u,v) נגדיר צביעות חוקיות להיות כאלה שהפרש הצבעים בקצוות שייך להפרשים החוקים לקשת הזו. המטרה היא למקסם את מספר הקודקודים שמקבלים צבע מבלי להפר אף אילוץ של קשת כלשהי.

בבירור זה מקרה פרטי של גרף אילוצים רגיל, אבל נראה שהוא קשה באותה המידה. ואכן הטענה היא כי $gap - kCSG_V[\delta,1] \leq gap - (|V|k)^5 CSG_\Delta[\delta,1]$, כלומר מקבלים קשיות במחיר ניפוח פולינומיאלי במספר הצבעים.

<u>למת יחידות בשלשות:</u> לכול קבוצה X ולכול $q \geq |X|^5$ ניתן לבנות בצורה יעיל פונקציה שמתאימה לכול איבר ב X צבע מתוך q כך שלכול $x_1, x_2, x_3, x_4, x_5, x_6$ איברים ב X, מתקיים כי $T(x_1) + T(x_2) + T(x_3) \equiv T(x_4) + T(x_5) + T(x_6) \ (mod \ q) \rightarrow \{x_1, x_2, x_3\} = \{x_4, x_5, x_6\}$ כלומר סכום הצבעים של כול שלשה מגדיר אותה באופן יחיד. נשים לב שזה גם מגדיר ביחידיות זוגות ובודדים, כי פשוט נוסיף את אותו האיבר בשני הצדדים...

נבנה את הפונקציה T לפי הסדר, כלומר נעבור על איברי X לפי הסדר ונתאים לכול אחד צבע. בבירור שלראשון אפשר להתאים צבע מבלי להפר את היחידות בשלשות. נניח ונתנו צבע לכול k הראשונים, אז לבא בתור נשים צבע שלא שווה ל $T(u) + T(v) + T(w) - T(y) - T(z)$ לאף חמישיה u, v, w, y, z של איברים שכבר נתנו להם צבע, כמו כן ברור שלבדוק את כול החימשיות זה זמן פולינומיאלי. כול מה שנשאר להראות הוא שקיים צבע כזה שלא יפר את התנאים, ואכן מספר הצבעים שאי אפשר לשים הוא לכול היותר k^5 כאשר כזכור k הוא כמה שצבענו עד עכשיו, וזה ברור כי לכול קומבינציה של חמישה צבעים אנחנו מקבלים איסור בודד, לכן משום שבחרנו את $q \geq |X|^5$ אז בוודאות יש עוד לפחות צבע אחד שאנחנו יכולים לשים מבלי להפר האילוץ.

<u>הוכחת הקשיות:</u> תחילה נשים לב כי ניתן להניח בה"כ שהגרף מלא, אם לא אז פשוט נוסיף קשתות ונשים עליהם אילוצים טריוויאלים שתמיד מסתפקים. לכן מעתה אין צורך לציין את הקשתות.

אזי בהינתן גרף $kCSG_V$ נסמנו $U = (V, \Sigma, \varphi)$ נבנה גרף חדש $U' = (V, [0, \dots, q-1], \Delta)$, כלומר גרף על אותם הקודקודים, אבל הגרף הזה יהיה גרף צביעת הפרשים, כאשר מספר הצבעים q יהיה $(k|V|)^5$, כך שנוכל למצוא $T: (V \ x \ \Sigma) \rightarrow [q]$ יחידה בשלשות כמו שראינו לעיל שאפשר. את האילוצים על הקשתות שגרף החדש נגדיר כמובן לפי הפרשים, ובאופן הבא

$\Delta(u,v) = \{T(u,i) - T(v,j) | (i,j) \in \varphi(u,v)\}$, כלומר ההפרשים החוקיים בין צבעים של קודקודים של קשת בגרף החדש הם אלה שמתקבלים מהפרש בין הפעלת הפונקציה היחודית שלנו T על פני הצבעים החוקיים באילוצים המקוריים על הקשת הזו.

אם יש לנו צביעה $A: |V| \rightarrow \Sigma$ שצובעת את כול הקודוקדים בגרף U המקורי, אזי בגרף החדש נצבע כול קודקוד $v \in V$ ב $T\big(v, A(v)\big)$, ובבירור הצביעה הזו תהיה חוקית בגרף ההפרשים ותבצע את כול הקודקודים.

כעת נראה כי אם יש צביעה של הגרף החדש U' של יותר מ δ מהקודקודים אז בהכרח יש צביעה של אותם הקודקדים בגרף המקורי, ובפרט אפשר לצבוע בו יותר מ δ מהקודקודים. אם $A': V' \rightarrow [q]$ היא צביעה של חלק מקודקודי $V' \subseteq U'$ באופן חוקי, אז אפשר להגדיר צביעה $A: V' \rightarrow \Sigma$ באופן שיתקיים $A'(v) \equiv T\big(v, A(v)\big) + d$ עבור d יחיד.

ואכן אם מספר הקודקודים שהצביעה A' צובעת הוא 2 כלומר $|V'| = 2$ אזי $V' = \{u, v\}$ ומהיות A' צביעה חוקית מתקיים כי $A'(u) - A'(v) \equiv T(u,i) - T(v,j)$ עבור i, j צבעים כלשהם המקיימים האילוצים בגרף המקורי. לכן $A'(u) = T(u,i) + d$ כלומר יש שוויון עד כדי הזזה בקבוע (זה ברור כול מספר הוא שווה לו עד כדי הזזה בקבוע) אבל אז יתקיים $A'(v) = T(v,j) + d$ עבור אותו ה d, כי ההפרש שלהם חייב להיות כמו שצוין לעיל. עכשיו נשים לב כי מתכונות T מתקיים כי i, j לעיל הם

יחידים כי נניח והיו i', j' שגם מקיימים את זה עם קבוע הזזה d' אז מתקיים כי $A'(u) = T(u,i) + d$ וגם $A'(v) = T(v,j) + d = T(v,j') + d'$ כמו כן $A'(u) = T(u,i') + d'$ ולכן מחיסור המשוואות נקבל כי $T(u,i) - T(v,j) = T(u,i') - T(v,j')$ ומהעברת אגפים ויחידות בזוגות נקבל כי $i = i'$ וגם $j = j'$ (לא יכול להיות הפוך כי זה לקודקודים שונים) ולכן גם ה $d = d'$ מחד ערכיות.

ולכן אם מספר הקודקודים הצבועים הוא 2 הצביעה של הגרף החדש U' אכן משרה צביעה יחידה של 2 קודקודים לגרף המקורי.

אם $|V'| = 3$ כלומר $|V| = \{u, v, w\}$, אזי לכול זוג מהם קיים ויחיד זוג צבעים בגרף המקורי, כך שלאחר הפעלת *T* עליהם עם הצבים הללו נקבל את הצביעה בגרף החדש (זה נובע ממה שהוכחנו לעיל לזוגות). צריך להראות שבמקומות שיש חפיפה אז הצבעים זהים. כלומר לזוג (u, v) קיים ויחיד זוג צבעים (i, j) ולזוג (v, w) קיים ויחיד זוג צבעים (j', k) ולזוג (w, u) קיים ויחיד זוג צבעים (k', i') וצריך להראות ש$i = i', j = j', k = k'$ כדי שהצביעה שמושרה בגרף המקורי קונסיסטנטית. ואכן מיחודיות בשלשות, אם נסכם את ההפרשים של קשתות המשולש הנ"ל אז הסכום הוא אפס כי הוא יוצא $A'(u) - A'(v) + A'(v) - A'(w) + A'(w) - A'(u) = 0$. ולכן נקבל כי

$T(u,i) - T(v,j) + T(v,j') - T(w,k) + T(w,k') - T(u,i') = 0$ ואז מיחידות בשלשות של *T* נקבל את השוויון הדרוש, ובפרט מחד ערכיות נקבל גם שוויון בהזזות *d* של הצבעים. לכן צביעה של *3* קודקודים בגרף החדש משרה צביעה חוקית קונסיסטנטית יחידה בגרף המקורי של 3 קודקודים.

לכול קבוצה גדולה יותר מ3 אז ראינו שהצביעה המושרית מוגדרת באופן יחיד וכי ההפרשים בכול שלשה זהים, ולכן ההפרשים בכול הקשתות זהים, לכן צבעיה של הגרף U' מגדירה באופן יחיד צביעה חוקית של אותם הקודקודים ב U ולכן ה *GAP* נשמר, לכן אם יכלנו לצבוע לפחות δ מהקודקודים בגרף U' אנחנו יכולים גם δ מהקודקודים בגרף *U*.

לסיכום ראינו שאם יש צביעה A' של U' אזי לכול קודקוד שנצבע קיים ויחיד צבע בגרף המקורי שהוביל לצביעה הזו, וההתאמה הזו היא קונסיסטנטית ויחידה.

<u>קושי לקרב המספר הכרומטי:</u>

בהנתן גרף, $\chi(G)$ הוא המספר הכרומטי שלו, והוא המספר הקטן ביותר של קבוצות בלתי תלויות זרות בקודקודים שאיחודן הוא כול הגרף. נשים לב כי $gap - \chi(G)\left[q, \frac{q}{\delta}\right]$ כלומר בהינתן גרף להגיד האם המספר הכרומטי שלו פחות מ q (טוב) או יותר מ $\frac{q}{\delta}$ (רע) זו בעיה קשה $(\delta < 1)$. ואכן הטענה היא כי מתקיים $gap - qCSQ_{\Delta}[\delta, 1] \leq gap - \chi\left[\mathrm{q}, \frac{\mathrm{q}}{\delta}\right]$. נשתמש באותה הרדוקציה בה השתמשנו להראות כי IS היא בעיה קשה, כלומר לכול קודקוד בגרף $(V, E, \mathrm{q}, \Delta)$ המקורי, ניצר q קודקדים שייצגו כול אחד צבע אחר, ואת הקשתות נעביר רק אם הצביעה בצבעים שהם מייצגים היא אסורה, כלומר הגרף יהיה $|V|$ קליקות בגודל q שחלק מהקשתות בינהן קיימות.

(נקודה חשובה היא שמספר צביעה זו בעיית מינימום, לכן הספים מתחלפים)

בהינתן צביעה A לגרף הCSG_{Δ} של כול הקודקודים, נקבל IS כפי שראינו בגודל $|V|$ בגרף החדש, שהוא יהיה קודקוד אחד מכול קליקה. אולם אם נקח את הצביעה ונוסיף לכול צבע 1 זו עדיין צביעה חוקית, וגם זו תניב IS בגודל $|V|$ שלוקח קודקוד אחד מכול קליקה, ולכן למעשה כול הזזה בקבוע של הצביעה תניב IS בגודל $|V|$ בגרף החדש, וזה משום שרק ההפרשים מעניינים אותנו. לכן יש חלוקה של הגרף החדש ל q קבוצות בלתי תלויות שמכסות את כולו, ולכן מספר הצביעה הוא פחות מ q.

בהינתן שידוע כי לא ניתן לצבוע יותר מ δ מהקדקודים של הגרף המקורי אזי הקבוצה הבלתי תלוי הגדולה ביותר בגרף החדש היא בגודל $\delta|V|$ ולכן מטענה פשוטה בגרפים נקבל כי מספר הצביעה בגרף החדש הוא $\chi(G) \geq \frac{q|V|}{\delta|V|} = \frac{q}{\delta}$ כי $q|V|$ זה מספר הקודקודים בגרף, ולכן קיבלנו הנדרש.

לכן קשה לקרב את מספר הצביעה של גרף לכול פקטור קבוע δ כי אחרת היינו יכולים לפתור את בעיית הפער.

<u>Unique-Games:</u>

הבעיה הינה גרף CSG מיוחד, כך שלכול קשת ולכול צבע מתאימים פרמוטציה על הצבעים, כלומר לכול קשת, אם צבענו קודקוד אחד שלה בצבע מסוים יש אך ורק צבע אחד לקודקוד השני שיספק את האילוצים של הצביעה.

נשים לב שלבדוק האם יש צביעה מספקת זה פשוט מאוד, נבחר קודקוד ונצבע אותו בצבע מסוים ונגרור את כול שאר הצבעים על הקודקודים האחרים, אם הצלחנו לצבוע את כולם יופי, אחרת נבחר את הצבע הבא. אם עברנו על כול הצבעים ולא הצלחנו אז אי אפשר לצבוע את כול הגרף.

הטענה אומרת אמנם כי בעיית הפער, כלומר כמה קודקודים אפשר לצבוע הכי הרבה מבלי להפר האילוצים, עבור פער $[\varepsilon, 1-\varepsilon]$ עבור ε מספיק קטן, אז קיים k מספר צבעים, כך שהבעיה הזו היא $NP-hard$. נשים לב שזו רק השערה! זה לא הוכח, ויכול להיות שזה ב P.

<u>Polynomial Hierarchy:</u>

נזכר ב TQBF שראינו שהיא $PSpace-Complete$. זו הייתה שפה של נוסחאות עם כמות לא חסומה של כמתים מתחלפים. כעת נחסום את כמו ההתחלפויות של הכמתים, ונגדיר לכול $i \in N$ את Σ_i להיות כול השפות L כך ששפת כול הנוסחאות עם i חילופי כמתים, שמתחיל בקיים, היא שלמה להן. כלומר כול השפות שיש רדוקציה פולינומיאלית מהן לשפת כול הנוסחאות שמתחילות בקיים ומכילות i חילופי כמתים והן ספיקות.

את Π_i נגדיר להיות השפות שמהשלים שלהן ב Σ_i או באופן שקול קל לראות שזה כול השפות ששפת כול הנוסחאות הספיקות עם i חילופי כמתים שמתחילות בלכול שלמה לה.

את ההיררכיה נגדיר להיות $PH = \cup_{i=1}^{\infty} \Sigma_i$. קל לראות כי $PH \subseteq PSPACE$ כי יש רדוקציה פשוטה ל $TQBF$. אבל נשים לב שלא ידוע שוויון, כי ב $TQBF$ מספר התחלפויות הכמתים יכול להיות תלוי באורך הקלט, ויכול להשתנות לפי המילים שבקלט.

קל לראות כי $\Sigma_i \subseteq \Sigma_{i+1}$ כי נוסיף משתנה פיקטיבי בסוף הנוסחא ולא נשתמש בו בכלל.

כמו כן $\Sigma_i \subseteq \Pi_{i+1}$ כי בלכול הראשון נשים משתנה פיקטיבי ולא נשתמש בו, ולאחר מכן תופיע הנוסחא, שתתחיל כעת בקיים.

<u>הגדרה שקולה:</u> $L \in \Sigma_i$ אם קיימת מכונת טיורינג M שרצה בזמן פולינומיאלי ולכול קלט x מתקיים $x \in L \leftrightarrow \exists u_1 \forall u_2, \dots, Q_i u_i : M(x, u_1, \dots, u_i) = 1$ כאשר Q_i הוא או קיים או לכול, תלוי בזוגויות של i. כלומר קלט הוא בשפה אמ"מ קיים לכול קיים לכול... כך שהמכונה מקבלת את x וכול העצות הללו. ובנוסף כול העצות הללו כולן באורך פולינומיאלי ב x.

נשים לב שאחרי הכמתים לא חייב להופיע משתנה יחיד, אלא יכולים להופיע כמה משתנים שלכולם אותו הכמת.

נוכיח את השקילות של ההגדרות. נסמן $TQBF_i$ להיות שפת כול הנוסחאות שהן אמת ובעלות i חילופי כמתים ומתחילות בקיים. נסמן ב C_i את השפות ש $TQBF_i$ שלמה להן. נראה כי C_i היא בדיוק כמו Σ_i שהגדרנו לעיל עם קיום המכונה.

תחילה נשים לב כי $C_i \subseteq \Sigma_i$ כי תהי $L \in C_i$ אז נסמן ב f את הרדוקציה הפולינומיאלית ממנה ל $TQBF_i$. אזי $x \in L \leftrightarrow f(x) \in TQBF_i \leftrightarrow \exists u_1 \forall u_2, \dots, \varphi_x(u_1, \dots, u_i) \equiv true$. כלומר f היא פונקציה שבהנתן x תחזיר לנו נוסחה שערכה תמיד אמת והיא בעלת i החלפות כמתים. לכן נגדיר מכונת טיורינג M שבהינתן קלט x תחשב את $f(x)$ בזמן פולינומיאלי, וכמו כן היא תקבל $u_1, \dots, u_i$ ותוודא כי מתקיים $f(x)(u_1, \dots, u_i) = \varphi_x(u_1, \dots, u_i) \equiv True$. ולכן מתקיים כי קיימת מכונת טיורינג שרצה בזמן פולי ו $x \in L \leftrightarrow \exists u_1 \forall u_2 \exists u_3, \dots., (M(x, u_1, \dots, u_i) accepts)$ כנדרש.

כעת נראה כי $C_i \subseteq \Sigma_i$. ואכן תהי $L \in \Sigma_i$ אז קיימת מכונת טיורינג פולינומיאלית M כך שמתקיים $x \in \leftrightarrow \exists u_1 \forall u_2, \dots, Q_i M(x, u_1, \dots, u_i)\ accepts$. נזכר כי ממשפט cook-levine אפשר לקודד ריצה של מכונה על קלט מסוים בנוסחא באורך פולינומיאלי בקלט, והמכונה מקבלת את הקלט אמ"מ קיימת השמה מספקת לנוסחא. לכן אם הכמת האחרון Q_i הוא קיים, אז אין שום בעיה פשוט נחליף את M בנוסחא המייצגת שלה φ ונחבר את שני הקיים האחרונים ביחד וזו תהיה הנוסחא שהרדוקציה תחזיר על הקלט x. אם Q_i הוא כמת לכול אז נגדיר מכונה חדשה M' שעושה את פעולת M ובסוף מחזירה ההפך. לכן מילה היא בשפה אמ"מ לא קיימת השמה לנוסחא של M', אמ"מ לכול השמה לנוסחא של M' הערך הוא שקר, ולכן נשים את הנוסחא של M' בשלילה ולפניה כמת לכול שיתחבר ללכול שהיה שם כבר קודם. נשים לב שאין צורך לפתוח את השלילה עם דה- מורגן, דבר שעשוי לקחת זמן אקפוננציאלי. ולכן יש רדוקציה ל $TQBF_i$ כנדרש.

<u>תכונות:</u>

- מההגדרה השקולה קל לראות כי $\Sigma_1 = NP$ כי זה פשוט לדרוש שקיים עד, ולכן $\Pi_1 = Co - NP$
- אם קיימת שפה שלמה ל PH אז ההיררכיה קורסת לרמה של השפה השלמה. כי לכול שפה אחרת L יתקיים כי $x \in L \leftrightarrow f(x) \in A$ כאשר A זו השפה השלמה. אזי הנוסחא לשפה A שמכיל כמות קבועה של חילופי כמתים תתאים, עבור מכונת טיורינג שקודם מחשבת את f ואז מריצה את המכונה של A.
- אם קיים i שעבורו $\Sigma_i = \Pi_i$ אז כול ההיררכיה קורסת ל Σ_i, כלומר $PH = \Sigma_i$. נוכיח באינדוקציה על j כי $\Sigma_j, \Pi_j \subseteq \Sigma_i$. עבור $j \leq i$ ברור שזה נכון בתוספת ההנחה שלנו. כעת נניח באינדוקציה עבור k ונוכיח ל $k+1$. אזי תהי $L \in \Sigma_{k+1}$ אזי קיימת מכונת טיורינג M ונוסחה מתחלפת כמתים באורך $k+1$ כך ש $x \in L$ אמ"מ הנוסחא ספיקה. נגדיר מכונה חדשה M' שתקבל קלט אחד פחות מ M כלומר M' מקבלת $(x, u_2, \dots, u_{k+1})$, תחילה מוודא כי x הוא מהצורה $x\#u_1$ ואז מריצה את $M(x, u_1, \dots, u_{k+1})$. אזי נגדיר שפה חדשה L' כך שאיבר $x \in L'$ אמ"מ נוסחא עם k החלפת כמתים שמתחילה בלכול, והמכונה היא M'. בבירור השפה החדשה L' נמצאת ב Π_k ומהנחת האינדוקציה $\Pi_k \subseteq \Sigma_i$ ולכן יש נוסחא עם i החלפת כמתים שמתחילה בקיים, ומכונה M^* כך שאיבר הוא ב L' אמ"מ הנוסחא ספיקה על המכונה M^*. ולכן לסיכום:
$x \in L$ אמ"מ הנוסחא עם $k+1$ ההחלפות והמכונה M ספיקה אמ"מ הנוסחא בעלת $k+1$ החילופים והמכונה M' כלומר $\exists u_1 \forall u_2, \dots, M(x\#u_1, u_2, \dots, u_{k+1})$ ואז אמ"מ קיים u_1 כך ש $x\#u_1 \in L'$, זה לפי הגדרת L', וזה אמ"מ קיים u_1 כך שהנוסחא בעלת i הכמתים מתקיימת, והיא מתחילה בכמת קיים. לכן אפשר לחבר את שני כמתי הקיים לכמת אחד ולקבל נוסחא בעלת i חילופי כמתים שמתחילה בלכול שהיא ספיקה אמ"מ $x \in L$ ולכן $L \in \Sigma_i$ כנדרש. וכנ"ל עבור Π_k.

Probabilistic Turing Machines:

מכונות טיורינג אקראיות הן מכונות שיש להן סרט נוסף מיוחד שמכיל ביטים אקראים, והן יכולות לקרוא ממנו. מה שמעניין אותנו הוא ההסבתרות על פני כול הרצפים האקראיים שיכולים להופיע בסרט הזה שהמכונה תקבל קלט מסוים.

BPP: כול השפות שיש להן מכונת טיורינג אקראית M שרצה זמן פולינומיאלי ומקיימת כי הסיכוי לשגיאה שלה חסום בשני הכיוונים. $(Bounded\ probablistic\ polynomial)$

- $x \in L \rightarrow P(M(x,r)=1) \geq \frac{2}{3}$ כלומר אם המילה בשפה אז המכונה תקבל בסיכוי לפחות $2/3$ או במילים אחרות תטעה בסיכוי קטן מ$\frac{1}{3}$.
- $x \notin L \rightarrow P(M(x,r)=0) \geq \frac{2}{3}$ או במילים אחרות אם המילה לא בשפה, הסיכוי שנטעה ונגיד שהיא כן בשפה קטן משליש.

נשים לב כי אפשר להגדיר $BPP(\alpha,\beta)$ להיות כול השפות שקיימת מכונת טיורינג אקראית שרצה בזמן פולינומיאלי כך שאם x בשפה הסיכוי שהמכונה תקבל אותו הוא יותר(ממש) מβ ואם הוא לא בשפה אז הסיכוי שהיא תקבל אותו קטן שווה ל α. לכן $BPP = BPP\left(\frac{1}{3},\frac{2}{3}\right)$.

נשים לב ש BPP סגורה לחיתוך משיקולי אמפליפיקציה, וכנ"ל גם לאיחוד. כי אם $A,B \in BPP$ אז אם $x \in A \cap B$ אז אפשר בזמן פולי אקראי להכריע ש $x \in A$ בשגיאה קטנה, וש $x \in B$ בשגיאה קטנה, ולכן הסיכוי לטעות בחיתוך הוא מכפלת הסיכויים לטעות, אבל עדיין קטן. כנ"ל עבור איחוד, אבל לצד השני.

RP: כול השפות שיש להן מכונת טיורינג אקראית M שרצה בזמן פולינומיאלי כך שאם קלט הוא בשפה אז הסיכוי שנגיד שהוא בשפה הוא לפחות חצי, אבל אם הוא לא בשפה אז בוודאות לא נקבל אותו. כלומר יש סיכוי ל $false\ negative$. אפשר לראות כי $RP = BPP\left(0,\frac{1}{2}\right)$.

חסם צ'רנוף:

עבור x_i מ"מ בלתי תלויים ושווה התפלגות שמקבלים ערכים בקטע $[0,1]$ מתקיים

$P(\sum_i X_i > (1+\delta) * E(\sum_i X_i)) < \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^{E[\sum_i X_i]}$ כלומר הסיכוי שהסכום רחוק מהתוחלת בפקטור, קטן מביטוי אקספוננציאלי הולך לאפס.

$\mathbf{BPP \subseteq \Sigma_2}$**:**

תחילה נשים לב שאפשר להגדיל את הפער ב BPP ולקבל שגיאה בסיכוי $\frac{1}{3P(N)}$ כאשר $P(n)$ הוא פוילנומם זמן הריצה של המכונה, בעזרת חסם צ'רנוף. פשוט מריצים הרבה פעמים באופן בלתי תלוי את המכונה המקורית ומחזירים את הmajority.

עבור $L \in BPP$ ידוע שקיימת מכונה הסתברותית M עם שגיאה כמו לעיל. נראה כי מתקיים $x \in L \leftrightarrow \exists s_1,\dots,s_m \in \{0,1\}^m \forall r \in \{0,1\}^m (M(x, r \oplus s_i) accepts\ for\ some\ i)$ כאשר m זה מספר הביטים שהמכונה הרנדומית צריכה (פולינומיאלי). כלומר הטענה היא שאם המילה בשפה אז רוב המחרוזות הרנדומיות יגרמו למכונה למקבל, ולכן קיימת משפחה קטנה של מחרוזות כך שלכול מחרוזת אחרת, קסור עם אחת מהמחרוזות הללו יכניס אותה לתוך התחום שבו M מקבלת. בעוד

שאם המילה לא בשפה אז כמות המחרוזות שבהן המכונה תקבל אותה כול כך קטנה, שלא קיימת משפחה של מחרוזות שהזזה בהן תביא כול מחרוזות למחרוזות שתגרום ל M לקבל.

ואכן נשתמש בשיטה ההסתברותית להוכיח קיום של המחרוזות הללו. עבור $x \in L$ הסיכוי על פני כול ה $s_1, \dots, s_m$ שקיים r שעבור אף הזזה לא יתקבל במכונה, קטן מחסם האיחוד , מסכום הסיכויים לכול ה r האפשריים. הסיכוי עבור r מסוים שבכול ההזזות עדיין המכונה לא תקבל אותו זה $\left(\frac{1}{3m}\right)^m$ כי זה מכפלת הסיכויים כי הם ב"ת. ולכן גם אחרי הסכום הסיכוי הזה קטן מאחד, לכן קיימים $s_1, \dots, s_m$ שמקיימים הנדרש.

מצד שני אם $x \notin L$ אז לכול $s_1, \dots, s_m$ מתקיים כי הסיכוי על פני כול המחרוזות r שקיים i עבורו המכונה מקבלת עם r מוזז ב s_i קטן מאחד מחסם האיחוד. לכן לכול $s_1, \dots, s_m$ קיים r שהוא לא מתקבל. לכן הנוסחא נכונה.

נשים לב כי $BPP = Co - BPP$ ולכן $BPP \in \Pi_2$ ולכן $BPP \in \Sigma_2 \cap \Pi_2$

<u>Random Walks:</u>

נתבונן בבעית הקשירות בגרף לא מכוון. כלומר יש לנו גרף G לא מכוון וקודקודים s, t ואנחנו רוצים להכריע האם יש מסלול בין s ל t בגרף. הטענה היא שהשפה הזו ב RL כלומר במקום לוגריתמי אקראי.

האלגוריתם יהיה להוסיף לכול קודקוד קשתות לעצמו תחילה, אחר כך להתחיל ב s ולעבור לשכן שלו בהתפלגות אחידה על פני השכנים, ואם הגענו ליעד t נפסיק ונאמר שיש מסלול.

נשים לב שאם הגרף קשיר אז הסיכוי שבצד הn נהיה בקודקוד i כלשהו מתכנס להתפלגות הסטציונרית של הגרף, כלומר לערך עצמי של מטריצת ההתפלגות, והוא $\frac{d_i}{2|E|}$ כאשר d_i זו הדרגה של הקודקוד ה i. קל לראות שאכן אם הגענו כבר להתכנסות הסטציונרית אז זה אכן וקטור עצמי ולכן ההתפלגות תשאר כזו לנצח. (וזה נכון לכול קודקוד בנפרד)

ולכן אם ההתפלגות כבר התכנסה, אז בהנתן שהיינו בקודקוד i מספר הצעדים בתוחלת שייקח לנו עד שנחזור אליו שוב הוא $\frac{2|E|}{d_i}$

נשים לב שזה אלגוריתם עם שגיאה חד צדדית, כלומר אם אין מסלול בגרף בין הקודקודים הוא לעולם לא יגיד שיש, אולם אם יש מסלול, אז קיים סיכוי שהוא יגיד שאין כי הוא לא מצא אותו. מחלקת השפות שיש להן שגיאה חד צדדית כזו וצריכות מקום לוגריתמי נקראת RL.

נשים לב שאם יש מסלול בין s ל t אז בתוחלת מספר הצעדים שיקח לנו עד שנגיע ל t הוא לא יותר מ $2|V||E|$. כי אם נתחיל מ s, אז הסיכוי שנתקדם לאורך המסלול הוא $1/d_i$, ולכן בתוחלת כול d_i צעדים נתקדם לאורך המסלול, ובתוחלת אם לא התקדמנו לאורך המסלול אז כול $\frac{2|E|}{d_i}$ צעדים נחזור חזרה לקודקוד שעל המסלול. לכן בתוחלת כול $2|E|$ צעדים התקדמנו צעד אחד במסלול, ולכן משום שאורך המסלול לא יותר מ $|V|$ נקבל החסם הדרוש על התוחלת.

לכן ממרקוב, אם נלך $2 * c * |V| * |E|$ צעדים אז נקבל טעות שגיאה של $1/c$ לכול c קבוע.

סיכום קשיות של בעיות:

1. זה NP קשה לקרב את הסוכן הנוסע ללא א"ש משולש לכול פקטור קבוע
2. זה NP קשה לקרב את Max-Is לכול פקטור קבוע, ובפרט את max-clique.
3. זה NP קשה לקרב את מספר הצביעה של גרף לכול פקטור קבוע
4. (G,k) כך ש ב G יש Clique/IS בגודל K זו בעיה NPC
5. זה NP קשה לקרב את $shortest\ path\ with\ forbiden\ pairs$ לכול פקטור קבוע.
6. יודעים שתיים קירוב ל VC מינמלי.
7. יודעים שתיים קירוב ל $MaxCut$.

יחסי הכלות בין מחלקות:

- $BPP \subseteq \Sigma_2$ ובפרט אם ההיררכיה הפולינומיאלית קורסת(עד לבסיס) אז $BPP \subseteq NP$.
- $NP = \Sigma_1 \subseteq PH \subseteq PSPACE \subseteq EXPTIME$
- $NTIME(t) \subseteq NSPACE(t) \subseteq DSPACE(t(e))$
- $NTIME(t) \subseteq DTIME(2^t)$
- $LSPACE \neq PSPACE$ (מלכסון)
- $DSPACE(n) \neq DSPACE(n^2)$
- $PSPACE \neq DSPANE(n)$ כי אחרת נקבל $DSPACE(n^2) \subseteq PSPACE$ ואז בפרט $DSPACE(n^2) \subseteq DSPACE(n)$ וזה לא נכון...
- לא יכול להיות שמתקיים $NP \subseteq DTIME(n^k)$ עבור k ספציפי כי אז מתקיים $P \subseteq DTIME(n^k)$ ובפרט נקבל $DTIME(n^{2k}) \subseteq DTIME(n^k)$ כי בבירור $P \subseteq NP$ וזה לא אפשרי.
- אם $P = NP$ אז לרוב הבעיות ב NP לא רק שאפשר לוודא אותן בזמן פולינומיאלי אלא אפשר ממש לבנות פתרון בזמן פולינומיאלי. למשל נניח ו $P = NP$ אז בהנתן גרף ומספר k ונניח שיש בו קליקה בגודל k אז אפשר ממש לבנות אותה. כול פעם נסיר קודקוד, אם לאחר ההסרה עדיין יש קליקה בגודל k אז נמשיך, אחרת הקודקוד הזה בוודאות היה חלק מהקליקה, אז נוסיף אותו ונמשיך לבדוק עבור $k-1$.
- יכול להיות ש $NL = NP$!!

טענות בגרפים:

- אם בגרף הקבוצה C של קודקודים היא קבוצה בלתי תלויה, אז V/C היא כיסוי בקודקודים של הגרף. (כי לא קיימת קשת בין שני קודקודים של C ולכן כול קשת נוגעת לפחות בקודקוד אחד ב V/C.

שפות מעניינות:

- האם קיים מסלול פשוט באורך בדיוק $\alpha * |V|$ זה ב NP כאשר α פרמטר(ולא שווה 1)
- בהנתן נוסחת $3CNF$ האם יש לה השמה **לא** מספקת. נשים לב שזה ב L כי לכול נוסחא שבה לא בכול הפסוקיות מופיע משתנה ושלילתו יש השמה לא מספקת. לכן לבדוק האם בכול הפסוקיות מופיע משתנה ושלילתו זה ב L.
- בגרף מכוון ושני קודקודים s,t האם יש מסלול אחד בדיוק בין s ל t. נשים לב שהשפה הזו היא $NL-Complete$. כי המשלים שלה ב NL כי בהנתן גרף נצטרך לוודא או שאין מסלול, שזה ב NL או שיש לפחות שניים, ולכן העד יהיה שני מסלולים, ואינדקס שמוכיח שהם

שונים. כמו כן היא שלמה, כי המשלים שלה שלם. בהנתן גרף לבעיית $ST - conn$ נוסיף לו קודקוד התחלה וקודקוד סוף חדשים, נחבר אותם פעם אחת בקשת ופעם שנייה דרך הגרף G.

Error Correction Codes:

המטרה היא להעביר מידע בין שני צדדים, כאשר יש סיכוי שהמידע בדרך יתעוות ויקבל שגיאות בחלק מהביטים שלו. נניח כי כמות השגיאות שיכולה לקרות היא חסומה, ונרצה למצוא דרך להעביר את המידע שבידנו כך שבצד השני יוכלו להבין מה התכוונו גם לאחר השגיאות, ונרצה לעשות את זה בצורה כמה שיותר יעילה.

מה שנעשה הוא שהמילים שנשדר ילקחו מתוך עולם שגדול יותר מכמות המילים החוקיות השונות שנרצה להעביר, תהיה לנו התאמה בין מילים חוקיות בשפה שלנו, לבין חלק מהמילים בעולם הגדול. ונרצה לדאוג שהמרחק בין שתי מילים חוקיות יהיה כמה שיותר גדול, כך שאם היו שגיאות, עדיין יהיה אפשר לדעת מאיזה מילה מקורית זה הגיעה.

<u>הגדרה:</u> קוד תיקון שגיאות מוגדר להיות רביעיה $[n, r, d, q]$ כאשר n זה אורך המילה שאותה נשלח על הקו, r זה הקצב של הקוד, זה למעשה כמה אינפורמציה באמת העברנו במילה ששלחנו, וזה מוגדר להיות $r \coloneqq \log_q|C|$ כאשר q זה גודל האלפבית של המילים שלנו, ו C זה מאגר המילים החוקיות בשפה שלנו. כמו כן d מוגדר להיות מרחק האמינג המינימלי בין שתי מילים חוקיות בקוד. נשים לב שקוד יכול לתקן שגיאות אם כמות השגיאות קטנה מ $\frac{d-1}{2}$ ערך שלם תחתון.

<u>קוד לינארי:</u> אם נבחר את האלפבית שלנו להיות שדה, אז המילים שלנו יהיו Q^n, למעשה מרחב וקטורי, ונוכל לבחור את C המילים החוקיות להיות תת מרחב לינארי. במקרה כזה הקצב של C יהיה בדיוק המימד שלו, כמו כן סכום של כול שתי מילים בקוד גם הוא יהיה מילה חוקית בקוד, ובפרט המרחק המינמלי בקוד שווה למשקל של המילה בעלת הכי הרבה אפסים. ובנוסף יתרון גדול הוא שלתת מרחב לינארי יש בסיס, ולכן למילים החוקיות בקוד שלנו יש מטריצה מייצרית שהיא פשוט איברי הבסיס של C, וכול מילה חוקית בקוד היא המטריצה הזו כפול וקטור מקדמים מתאים.

<u>Reed-Solomon:</u> $[q, k, q-(k-1), q]$

עבור Q שדה סופי בגודל q, נתבונן במרחב הוקטורי $\{f|f: Q \to Q\}$ כלומר כול הפונקציות מהשדה לעצמו. גודל הקבוצה הזו זה q^q ולמעשה כול מילה f במרחב הוקטורי הזה אפשר להציג כוקטור בגודל q כך שבתא ה i שלו יושב $f(i)$. נבחר תת מרחב לינארי C להיות כול הפולינומים ממעלה קטנה שווה $k-1$ עבור k כלשהו. נשים לב שהמימד של המרחב הזה הוא k ולכן זה הקצב של הקוד. כמו כן המרחק של הקוד, משום שהוא לינארי, הוא המשקל של הוקטור שאינו אפס בעל הכי הרבה אפסים. משום שכול מילה חוקית בקוד שלנו היא פולינום ממעלה *k-1* לכול היותר אזי הוא יכול להתאפס לכול היותר ב *k*-1 מקומות שונים כי אחרת הוא פולינום האפס. לכן המרחק של הקוד הוא $q-(k-1)$ שזה כמה לא אפסים יש למילה.

נשים לב שזה קוד מצוין, יש לו קצב טוב ומרחק טוב, אבל זו טיפה רמאות כי הוא לא בינארי.

המטריצה המייצרת של קוד זה היא איברי הבסיס בעמודות שלה, כאשר הבסיס הוא $\{x^i|1 \le i \le k-1\}$ ולכול איבר כזה, הוקטור שלו הוא הערכים שלו על כול איברי השדה.

<u>Hadamard:</u> $[2^n, n, 2^{n-1}, 2]$

נתבונן במרחב הלינארי $\{f|f:\{0,1\}^n \to \{0,1\}\}$ כלומר כול הפונקציות ממילה באורך n של ביטים לביט. נשים לב כי יש 2^{2^n} כאלה. נתבונן בתת המרחב הלינארי C שיהיה כול הפונקציות הלינאריות מ $\{0,1\}^n \to \{0,1\}$. נשים לב שכול מילה בקוד הזה היא באורך 2^n והיא הערכים (הביטים) שהפונקציה מחזירה על כול אחת מהמילים ב $\{0,1\}^n$.

נשים לב שפונקציות לינאריות מהסוג הנ"ל שוות לפונקציות מהצורה $f(x_1,\dots,x_n) = \sum_{i\in s} x_i \ (mod\ 2)$ כלומר לכול מילה בעלת n ביטים זה יחזיר את הקסור של איזה שהיא תת קבוצה שלהם, $S \subseteq [n]$. לכול S שונה נקבל פונקציה אחרת. ולכן יש 2^n פונקציות לינאריות, מתוך 2^{2^n} הפונקציות כולן. לכן הקצב של הקוד הוא $\log_2(2^n) = n$. כעת משום שהקוד לינארי כדי למדוד את המרחק שלו רק צריך לבדוק את המשקל של המילה בעלת הכי הרבה האפסים בקוד. נשים לב שלכול מילה בקוד היא מתאפסת על בדיוק חצי מהקלטים, כלומר כול פונקציה לינארית $f:\{0,1\}^n \to \{0,1\}$ שאינה פונקציית האפס מקבלת את הערך אפס על בדיוק חצי מהמחרוזות מתוך $\{0,1\}^n$ ולכן הוקטור שייצג אותה בקוד יהיה חצי אפסים חצי אחדות, ובפרט נקבל שהמרחק של הקוד הוא $\frac{2^n}{2}$. ואכן אם $f \neq 0$ אז הקבוצה S שמייצגת את הפונקציה אינה ריקה. נקח $i \in S$ אז לכול מילה $x \in \{0,1\}^n$ מתקיים $x + e_i$ גם כן מילה, אחרת, ב $\{0,1\}^n$ והערך ש f מקבלת על שתי המילים הללו הוא הפוך. לכן על חצי היא מקבלת אפס ועל חצי היא מקבלת אחד.

<u>שרשור של קודים:</u>

הרעיון הוא לעבור מקוד טוב אבל בעל אלפבית גדול, לקוד טוב באותה המידה, או לפחות קרוב, אבל עם אלפבית קטן, ובתקווה בינארי.

הרעיון הוא שכול אחת מהאותיות בקוד הגדול נקודד כמילה בקוד הקטן. כלומר בקוד הקטן תהיה מילה חוקית לכול אות באלפבית של הקוד הגדול.

למה לא פשוט לקודד את האותיות של הקוד הגדול בבינארי? כי זה לא ייתן לנו מרחק טוב בכלל, כי המרחק בין האותיות יהיה פשוט ביט בודד, ואז הגדלנו מאוד את אורך המילה אבל לא שינינו את המרחק בין המילים, ולכן הרסנו את הקוד.

מקובל לשרשר את *RS* הקוד הטוב שראינו אבל הלא בינארי, עם קוד הדמרד שהיה בעל קצב רע מאוד אבל מרחק מעולה, וגם בינארי. ואכן אם ניקח קוד $[q, k, q-(k-1), q]$ שהוא *RS* ונשרשר אותו לקוד הדמרד שיכיל q מילים חוקיות מעל אלפבית בינארי, ולכן נרצה שהקצב שלו יהיה $\log_2 q$ ולכן הוא יהיה קוד $\left[q, \log_2 q, \frac{q}{2}, q\right]$. אז השרשור יהיה קוד שכול מילה בו תהיה מאורך $q * q$ באופן ברור, כי כול אות במילת ה RS הפכנו ל q אותיות בינאריות. הקצב של הקוד יהיה $\log_2 q^k$ כי כמות המילים החוקיות לא השתנתה, אבל הבסיס שלפיו מודדים כן השתנה. זה למעשה $klog_2 q$ שזה מכפלת הקצבים. כמו כן המרחק יהיה $\frac{q}{2}\big(q-(k-1)\big)$ כלומר מכפלת המרחקים. כי אם יש שתי מילים שונות, אז הן שונות בלפחות $q-(k-1)$ מאותיות ה RS, וכול אות כזו שהיא למעשה עכשיו מילה בקוד הדמרד נבדלות בלפחות חצי מהמקומות.

<u>מטריצה מייצרת אקראית:</u>

נשים לב שכול מטריצה מגודל $n * k$ מעל שדה סופי Q מייצרת איזשהו קוד לינארי עם גודל בלוק n וקצב k (בתנאי שהוקטורים בלתי תלויים...) נתעניין במאפיינים של קוד שנוצר באופן הזה ממטריצה שנבחרת באקראי. נרצה שהמרחק והקצב של הקוד יהיו לינארים בגודל הבלוק.

הטענה היא שאם בוחרים מטריצה אקראית מעל Z_2 בעלת n שורות, אז קיים k כך שבסיכוי אקספוננציאלי קרוב לאחד הקצב והמרחק של הקוד יהיו לינארים. נתחיל מלהראות שהמרחק טוב. ואכן נחסום עם חסם האיחוד ביחד עם צ'רנוף את הסיכוי שקיימת מילה בקוד עם מרחק פחות

מלינארי, וזה יוצא אקפוננציאלי קרוב לאפס. כמו כן הסיכוי שהעמודות של המטריצה לא בלתי תלויות מוכל בסיכוי שהמרחק קטן, כי בפרט יהיו שתי מילים ממרחק אפס, ולכן גם הסיכוי הזה אקספוננציאלי הולך לאפס. לכן קיבלנו קצב ומרחק לינארי.

Semi-Definite-Programing:

בהינתן בעיית תכנות בשלמים, למשל כמו Max-Cut שהיא למעשה למקסם את המשוואה $\sum_{(i,j)\in E}\frac{1}{2}(1-x_ix_j)$ כאשר $x_i \in \{\pm 1\}$. כלומר מנסים להביא למקסימום את מספר המשתנים עם סימנים הפוכים, כלומר שנמצאים בצדדים שונים של החתך. כידוע זו בעיה קשה.

אבל מה שאפשר לעשות זה לנסות לפתור את הבעיה לא בשלמים אלא בוקטורים n ממדים. ובמקום מכפלה תהיה לנו מכפלה פנימית. בהינתן שזה קל למקסם את הסכום לעיל במקרה של וקטורים, אז נקבל פתרון שהוא קירוב טוב לקמסימום הרציף שהוא בהכרח יותר מהקמסימום הבדיד. בהנתן פתרון מקורב כזה, נגריל וקטור אקראי על הספרה כדי להפריד בין הוקטורים שקיבלנו כפתרון ולהחליט מי ילך לאיזה צד של החתך בגרף שלנו.

המטרה היא שהוקטור שהגרלנו יגרום לכמה שיותר וקטורים להיות בצדדים שונים של העל מישור שהוא יוצר, כי זה יניב החתך הגדול ביותר. כאשר הסיכוי שהוקטור יפריד בין שניים הוא הזווית בינהם חלקי כול המעגל (כפול שתיים כי המישור יכול להגיע משני וקטורים שונים)