משפט PCP והשימושים להוכחת קושי של בעיות קירוב

יהודה קליין

שלום קורא יקר. נראה שאתה מתעניין בסמינר שלי – מעולה, מישהו יקרא אותו! כמו שאמרה הכותרת, הסמינר יעסוק ב- "משפט PCP, ואיך אפשר להשתמש בו כדי להראות על בעיות קירוב מסוימות שהן NP קשות". ניסיון כלשהו, גם אם מועט, בהתעסקות עם מכונות טיורינג והערכת קושי של הכרעת שפות, יעזור מאוד בהבנת הסמינר.

בתור התחלה, לפני שנעמיק בנבכי הקסמים של פרוטוקולי הוכחה אינטראקטיבית, העלאה של גרפים בחזקות ורדוקציות מבעיות קירוב קשות לבעיות קירוב קשות אחרות, כדאי שנסביר את מבנה הסמינר:

**חלק א**

בו ניזכר קצת במושגים שקשורים לחישוביות, קצת במושגים שקשורים לסיבוכיות, ונציג גם כמה מושגים שאולי אתה לא מכיר בכלל.

**חלק ב**

בו נגדיר מערכות הוכחה הסתברותית, נציג את שתי הגרסאות של משפט ה- PCP, ונוכיח שהן שקולות.

**חלק ג**

בו ניגע בשני הנושאים הבאים: הראשון הוא גרפים מרחיבים, השני הוא שערים לוגיים בוחני השמה (מבחני השמה, באנגלית: (assignment testers.

**חלק ד**

בו נציג את עיקרי הוכחת משפט PCP לפי המאמר של פרופ' אירית דינור מ- 2005.

**חלק ה**

בו נשתמש במשפט החדש שלנו כדי להראות על כמה בעיות נבחרות שהן קשות לקירוב.

את **חלק א**, החימום שלנו לפני שמתחילים לעבוד באמת, נתחיל ממש בעמוד הבא.

**חלק א – חזרה קצרה על מושגים נבחרים מתורת החישוביות והסיבוכיות**

המחלקה NP

בוודאי יצא לך להתעסק עם בעיות NP קשות בעבר, אבל המושג PCP קשור מאוד להגדרה הפורמלית של המחלקה NP ולכן אולי כדאי להזכיר מהי:

המחלקה NP, שהיא מחלקת כל השפות הכריעות בזמן פולינומיאלי על ידי מחשב (או: מכונת טיורינג) לא דטרמיניסטי, מוגדרת להיות כל השפות שעבורן קיימת מכונת טיורינג דטרמיניסטית כך שלכל מילה , קיימת מחרוזת כך שבהינתן המילה והמילה , המחשב יוכל להכריע בזמן פולינומיאלי ב- ש- היא אכן מילה ב- .

לדוגמה: השפה SAT היא NP שלמה. לכל נוסחת SAT ספיקה, קיימת איזושהי השמה למשתנים שבתוך הנוסחה. נייצג את ההשמה בתור הסטרינג , וקל יהיה לכתוב תכנית שמקבלת את ואת ומוודאת שהשמת המשתנים ב- לפי אכן מספקת את הנוסחה.

בהמשך נתבונן במושג **מערכת הוכחה אינטראקטיבית**, ובמקום הייצוג הקלאסי של המערכת בתור שני מודלים חישוביים עם פרוטוקול תקשורת ביניהם, ננסה להתייחס אליהם בתור מחשב דטרמיניסטי עם גישה להוכחה כלשהי – כמו המחרוזת שראינו למעלה.

פתרון בעיות הסתברותי

לפעמים הכרעה של בעיות מסוימות היא משימה אקספוננציאלית במשאבי זמן ומקום. לפעמים גם אם אפשר לחסום את המשאבים הנדרשים על ידי פולינום, הפתרון הדטרמיניסטי לבעיה פשוט יקר מדי – למשל בעיית הכרעת הראשוניות של מספר טבעי, שאלגוריתם AKS יודע לפתור פחות או יותר בזמן . זה נחמד, אבל לשימושים סטנדרטיים נעדיף את אלגוריתם מילר-רבין, שאומנם יכול לתת לנו תשובות שגויות, אבל בהסתברות נמוכה מספיק כדי שנעדיף את זמן הריצה שלו.

*באופן כללי, אלגוריתם הסתברותי מכיל רכיב אקראי לחלוטין (בדרך כלל הפיכת ביט בין 0 ל- 1 באופן רנדומלי, המקבילה החישובית להטלת מטבע) שבעזרתו ניתן להפיק פתרונות עם "ביטחון גבוה" בנכונות התשובה, במהירות רבה יותר מזמן הריצה של אלגוריתם שמבטיח פתרון מדויק. בדרך כלל אלגוריתם הסתברותי כזה מורכב ממבחן פשוט (לדוגמה, בדיקת השורשים של בחשבון מודולו לפי מספר אולי-ראשוני עבור בסיס אקראי כלשהו – אם נחזור לדוגמא של מילר-רבין), ועל המבחן נחזור כמה פעמים כדי להוריד את ההסתברות לטעות מתחת לסף רצוי כלשהו.*

*פעמים רבות, נעדיף להגדיר את הטלת המטבע של המכונה מחדש בתור גישה למחרוזת אקראית מעל . השקילות נובעת מהעובדה שאם לא היינו נותנים למכונה את המחרוזת האקראית, היא הייתה יכולה לייצר אותה בעצמה, ומצד שני מכונה דטרמיניסטית יכולה לקרוא את האות הבאה במחרוזת אקראית בתור תחליף להטלת מטבע.*

מערכת הוכחה אינטראקטיבית

נתחיל מתיאור שפת הגרפים הלא איזומורפיים (GRAPH-NONISOMRPHISM, NONISO). נתונים לנו תיאורים של שני גרפים, G ו- H. לכל צומת בכל אחד מהגרפים יש ID ייחודי (מספר, שם או כל guid כזה או אחר. יכול מאוד להיות שבסידור מחדש של זהויות הצמתים בגרף G, נקבל תיאור מדויק של H.

\\\\\\\\\\\\\\\\\\\ תמונות של H ו- G

הבעיה שלנו היא שאנחנו רוצים להיות בטוחים ש- G ו- H הם אכן גרפים שונים (נניח שיש לנו random graph generator ואנחנו רוצים לוודא שהוא אכן מייצר גרפים אקראיים ולא את אותו המבנה המסובך שחוזר על עצמו בשמות שונים כל פעם).

את הבעיה ההפוכה, GRAPH-ISOMORPHISM, שבה אנחנו מנסים להראות שהגרפים שווים, קל לשים במחלקה NP. כל מה שאנחנו צריכים היא הוכחה שתכיל את שמות זוגות הצמתים המקבילים ב- G ו- H, ומחשב שמוודא שלאחר החלפת השמות אכן מתקבל תיאור של אותו הגרף. בזמן כתיבת הסמינר, עוד לא ידוע אם GRAPH-NONISOMORPHISM פתירה על ידי מכונה לא דטרמיניסטית בזמן פולינומיאלי.

עכשיו נעבור לסיפור קצר:

למיכאל יש שני גרפים, G ו- H. מסיבות שחורגות מהיקף הסמינר[[1]](#footnote-1), למיכאל *חשוב מאוד* שהגרפים לא יהיו איזומורפיים. מסיבות אחרות שגם כן חורגות מהנושא שלנו, מסתובבים בינינו **מוכיחים** מרושעים, שתכליתם היחידה היא לשכנע את מיכאל שהגרפים שהוא מחזיק הם אכן לא איזומורפיים, למרות ש- G ו- H הם אכן אותו הגרף. המוכיחים האלו הם בעצם מחשבים מהעתיד, שאינם מוגבלים על ידי הטכנולוגיה הפרימיטיבית שאני ואתה מכירים, וכל בעיה חישובית שנוכל לחשוב עליה, אותם המחשבים יכולים לפתור ב- 0 זמן (כן, גם SAT).

מיכאל צריך מחשב חזק כדי לבדוק אם הגרפים שלו לא איזומורפיים, אבל הוא לא יכול לסמוך על המוכיחים שסביבו (נזכיר שהבעיה החישובית שהמוכיחים מתמודדים איתה כרגע היא "איך אפשר לשכנע את מיכאל שהגרפים האיזומורפיים שלו הם לא איזומורפיים).

מיכאל מחליט ליצור **פרוטוקול הוכחה אינטראקטיבית**. הוא פותח את הלפטופ שלו, וכותב את התכנית הבאה:

1. בחר באופן רנדומלי לגמרי (מחולל הרנדומליות היחיד שאף מוכיח לא יכול לנבא את פעולתו הוא מחולל רנדומלי אמיתי, לצורך הדיון מיכאל מחזיק כזה) אחד מהגרפים H או G. הגרף שנבחר יהיה K.
2. בחר באופן רנדומלי לגמרי סידור מחדש של הצמתים בגרף K, ושלח אותו ביחד עם H ו- G אל מוכיח כלשהו. שאל את המוכיח אם K הוא סידור מחדש של H או של G.

אם תשובת המוכיח נכונה, בסבירות של לפחות חצי הגרפים לא איזומורפיים, כיוון שמוכיח הצליח להבדיל ביניהם בלי קשר לשמות הצמתים. אם המוכיח היה רק מנחש את התשובה היה לו סיכוי של 50% לטעות. אם התשובה לא נכונה, מיכאל מחזיק גרפים איזומורפיים, כיוון שהמוכיח ניחש את התשובה בלי לדעת בוודאות – וטעה.

1. חזור על התהליך עד שהמוכיח שכנע את הלפטופ בהסתברות גבוהה מספיק שהגרפים לא איזומורפיים.

*התיאור הקלאסי של* ***מערכת הוכחה איטראקטיבית****, או* ***פרוטוקול מוכיח-מאמת****, היא מכונת טיורינג פולינומיאלית הסתברותית (המאמת) שמתקשרת עם ישות חישובית בעלת כוח חישוב בלתי מוגבל (המוכיח) עם מניעים לא ידועים. התקשורת מתבצעת על ידי סדרה של מסרים (מחרוזות) – המאמת מקבל מילה שנמצאת או לא נמצאת בשפה שאותה הוא אמור להכריע. על סמך המילה הזו המאמת מחשב חישוב פולינומיאלי, שולח שאלה למוכיח, קורא את התשובה, ועל סמך התשובה מחשב שאלה נוספת, כך הלוך וחזור, כאשר למאמת נוספות כל פעם תשובות נוספות של המוכיח להתבסס עליהן בחישוב השאלה הבאה. בסופו של דבר המאמת יקבל את המילה אם ההסתברות לטעות נמוכה מספיק.*

*תיאור פורמלי יהיה: המאמת הוא פונקציה עם קלט שמכיל מילה שצריך להכריע, ואת ההודעות שנשלחו מהמאמת למוכיח ובחזרה עד עכשיו. הפלט יכול להיות קבלה של המילה, דחייה של המילה, או שאלה חדשה למוכיח. המוכיח יהיה פונקצייה עם קלט שמכיל את התקשורת בין המאמת למוכיח עד עכשיו, והפלט יהיה תשובה למאמת.*

*נשים לב שכדי שהמאמת שלנו יהיה מכונה פולינומיאלית, גודל כל הודעה שנשלחת או מתקבלת צריך להיות חסום על ידי פולינום של גודל המילה המוכרעת, וכך גם מספר ההודעות הכולל (כי המכונה צריכה לקרוא או לכתוב כל אחת מההודעות). אותנו מעניינת הגדרה שקולה לפרוטוקול: במקום להגדיר את המושג המסורבל של מוכיח, ניקח את כל שרשרת ההודעות ונגדיר אותה בתור הוכחה (כמו בחלק על* NP *למעלה). אם קיים מאמת שיכול לייצר שרשרת הודעות עם מוכיח ולהכריע את השפה בהסתברות גבוהה, קיים מאמת שיכול לחקות את פעולת המאמת הקודם בעזרת ההוכחה הפולינומיאלית בלי להצטרך לדבר עם אף מוכיח. כלומר נוכל לוותר על המוכיח ולהגדיר את מחלקת ההוכחה האינטראקטיבית* **IP** *– שפות שניתנות להכרעה על ידי מאמת הסתברותי פולינומיאלי שמקבל מילה והוכחה בגודל פולינומיאלי. בהמשך נגדיר עוד מגבלות על המאמת כדי ליצור מחלקות* PCP *עם מגבלות* *שונות על "כמות הרנדומליות" ו- "כמות הגישה להוכחה".*

*בעיות קירוב קשות*

*בתור התחלה, נגדיר את המושג בעיית אופטימיזציה. לשם כך נשתמש בשפה* 3SAT *המפורסמת. נזכיר ש-* 3SAT *היא גרסה של* SAT *שבה כל פסוקית מכילה 3 ליטרלים בדיוק. נוסחה לוגית נתונה היא מילה בשפה* 3SAT *אם ניתן לתת ערך 1 או 0 לכל אחד מהמשתנים, כך שהנוסחה תקבל ערך אמת. נתבונן בשני המופעים הבאים (לא צריך להתעמק בנוסחאות, אין להן משמעות מיוחדת חוץ מהעובדה שההשמות שבחרתי בהמשך מספקות חלק או את כל הפסוקיות בהן):*

*המופע הראשון ניתן לסיפוק על ידי ההשמה:*

*ולכן הנוסחה היא מילה ב-* 3SAT.

*לעומתה הנוסחה השנייה אינה מסופקת על ידי אף השמה.*

*אבל, בעיית* 3SAT *מגדירה באופן מידי את בעיית* Max-3SAT, *או: מה הוא המספר המקסימלי האפשרי של פסוקיות בנוסחה נתונה שניתן לספק בעזרת השמה של המשתנים ל- . במקרה והנוסחה הנתונה היא , קל לראות שניתן לספק ארבע פסוקיות בעזרת השמה מסוג:*

*מציאת המספר המקסימלי של פסוקיות הניתנות לסיפוק בנוסחה לוגית נתונה היא בעיה* NP *קשה. הרדוקציה מ-* Max-3SAT *ל-* 3SAT *מיידית: אם הראשונה פתירה, השנייה תחזיר את מספר הפסוקיות בנוסחה עבור כל מופע של* 3SAT*.*

Max-3SAT *היא בעצם* ***בעיית אופטימיזציה****. אי אפשר לספק את כל הנוסחה, אבל בכל זאת נרצה לספק כמה שיותר פסוקיות ממנה. דוגמה נוספת לבעיית אופטימיזציה מפורסמת היא:*

***בעיית כיסוי הקודקודים***(Vertex-Cover).

*בגרסתה המקורית, בעיית כיסוי הצמתים מנוסחת כך: בהינתן גרף ומספר , האם ישנה קבוצה של צמתים בגרף כך שלכל קשת בגרף, לפחות אחד הקצוות יהיה איבר בקבוצה. הבעיה ידועה להיות* NP *שלמה, ואנחנו לא מכירים לה פתרון פולינומיאלי דטרמיניסטי.*

*\\\\\\\\\\\\\\\\\\\ דוגמה לכיסוי צמתים בגודל 2 ודוגמה לגרף בלי כיסוי צמתים בגודל 2 (מעגל באורך 5)*

*גרסת האופטימיזציה* Min-Vertex-Cover *של הבעיה מבקשת ממנו, לכל גרף, להוציא בתור פלט את מספר הצמתים המינימלי כך שלכל קשת בגרף, לפחות אחד הקצוות יהיה בקבוצה. למשל עבור גרף מספר 2 למעלה, הפלט שיתבקש הוא 3, כיוון שרק תת קבוצה של 3 ומעלה צמתים בגרף יכולה לכסות את כל הקשתות.*

*הנה אלגוריתם שרץ בזמן אקספוננציאלי ופותר את* Min-Vertex-Cover:

*בהינתן גרף* G *עם צמתים ו- קשתות, :*

*עבור כל מ- עד :*

*עבור כל קבוצת צמתים בגודל :*

*עבור כל קשת ב- : בדוק האם אחד מקצוות הקשת* נמצא ב- .

אם כל הקשתות עברו את המבחן, החזר את בתור גודל הכיסוי המינימלי.

כפי שבוודאי שמת לב, הלולאה הפנימית "*עבור כל קבוצת צמתים בגודל :" גורמת לכך שהאלגוריתם מבצע חישוב פולינומיאלי עבור כל תת קבוצה של צמתים ב- , כלומר זמן הריצה הוא אקספוננציאלי. חשוב גם לציין שהאלגוריתם יכול באותה מידה להחזיר את קבוצת הקודקודים המינימלית שמהווה כיסוי צמתים במקום מספר, בלי תוספת רבה לחישוב.*

Min-Vertex-Cover *היא בעיה מצוינת להצגת הרעיון מאחורי חישובי אופטימיזציה, כיוון שיש לה אלגוריתם קירוב פשוט מאוד מסדר 2. האלגוריתם יעבוד כך:*

*כל עוד יש ב- G קשת לא מסומנת:*

*בחר קשת לא מסומנת ב- . סמן את והוסף אותה לקבוצה .*

*עבור כל אחד משני הקודקודים ש- מחברת:*

*עבור כל קשת שצמודה לקודקוד, סמן את הקשת (בלי להוסיף ל- ).*

*החזר בתור פלט כל קודקוד שצמוד לקשת שנמצאת ב- .*

*נקרא לקבוצת הקודקודים שהאלגוריתם מחזיר .*

*האלגוריתם שלנו סימן את כל הקשתות, וכל קשת מסומנת צמודה לקודקוד ב- , כלומר קיבלנו כיסוי קודקודים. כיוון שכל הקשתות שב- רחוקות לפחות מרחק של קשת אחת מהשנייה, כיסוי מינימלי מחייב לפחות קודקוד נפרד לכל אחת. אנחנו הוספנו שני קודקודים לכל אחת, לכן הכיסוי שהחזרנו לכל היותר כפול בגודלו מהכיסוי המינימלי.*

***חלק ב – מערכות הוכחה הסתברותית ומשפט* PCP.**

*נתחיל בהגדרת אנוטציה: משמעות הסימן היא כזו:*

1. *PCP משמעו Probabilistically-Checkable-Proof או במילים שלנו: הוכחה עבור שייכות מילה לשפה , כאשר ההוכחה צריכה לעמוד במבחן תלוי-אקראיות כלשהו שמכונת טיורינג פולינומיאלית תייצר.*
2. *מבחינתנו מסמל את אורכה של מילת הקלט , כלומר: .*
3. *משמעו completeness (שלמות), או: כאשר ההוכחה נכשלה במבחן, מה מידת הביטחון שלנו שהמילה באמת לא נמצאת ב- . הדיון שלנו מתמקד במקרה שבו , כלומר כל מילה ב- , בוודאות תעבור את המבחן שלנו.*
4. *משמעו soundness (נאותות), ההגדרה המקבילה ל- completeness, או: כאשר ההוכחה* ***עברה*** *את המבחן, מה מידת הביטחון שלנו שהמילה באמת ב- . אנחנו נתמקד במקרה שבו , כלומר מילה שאינה ב- תעבור את המבחן בהסתברות של לא יותר מחצי.*
5. *משמעו randomness, או: כאשר נתונה מילה באורך , מה מספר הטלות המטבע שהמכונה האקראית שלנו יכולה להטיל, כפונקציה תלויה ב- . אפשר להתייחס למספר הזה בתור 'מידת האקראיות שאנחנו מאפשרים למכונה כאשר היא מקבלת קלט באורך '. במילים אחרות, אפשר לומר שזהו האורך של המחרוזת הרנדומלית שהמכונה הדטרמיניסטית שלנו מקבלת, אם אנחנו לא רוצים להשתמש במושג 'הטלת מטבע'. אני אגלה כבר עכשיו שאני לא אוהב הטלות מטבע, ולכן נשתמש במחרוזות אקראיות במקומן.*
6. *משמעו query, או: מספר התווים מההוכחה שאנחנו מאפשרים למכונה שלנו לקרוא, כפונקציה של אורך . כלומר, המכונה שלנו לא תקרא את כל ההוכחה, אלא תקבל גישה רק לחלקים מצומצמים ממנה, כפונקציה תלויה ב- .*

*אנחנו נצמצם את הסימון שלנו כך שלא יכלול את ואת , כלומר: .*

*הסימון שהגדרנו כרגע הוא סימון למשפחות של שפות, בדומה לסימונים .*

*כאשר אנחנו אומרים אנחנו מתייחסים למשפחת כל השפות שעומדות בתנאים שהגדרנו. כלומר: קיימת מכונה דטרמיניסטית ופולינומיאלית, כך שעבור כל מילה ב- , ניתן לכתוב הוכחה , והמכונה – עם קלט , ומחרוזת רנדומלית באורך – קוראת תווים מ- , ומקבלת את . בנוסף, עבור כל מילה שאינה ב- ,* **לא ניתן** *לכתוב את , כלומר: עבור כל מחרוזת הוכחה המכונה הדטרמיניסטית קוראת תווים מ- (זהות התווים תלויה במחרוזת הרנדומלית), ודוחה את בהסתברות של חצי לפחות.*

*עכשיו אנחנו מוכנים להכריז על המשפט המפוצץ הבא:*

***או: לכל שפה ב-* NP*, קיימת מכונה פולינומיאלית דטרמיניסטית, שמקבלת מילה , הוכחה*** ***ומחרוזת רנדומלית באורך לוגריתמי ב-*** *,* ***ומכריעה את השפה עם שלמות*** ***ונאותות***  ***תוך קריאה של*** *מספר תווים קבוע מההוכחה, בלי תלות באורך* ***.***

***בואו נבחן מה בעצם המשפט אומר. ברמה הבסיסית ביותר הוא א***

1. כאן הכוונה ב'חורגות מהיקף הסמינר', היא - לא היה לי כוח לחשוב על סיבות שכאלה. זה גם המקום לומר שמחשבים מעולם לא היו אמורים להתקיים מעבר להגדרתם המתמטית האבסטרקטית, ואולי גם להוסיף ש'שימושים מהעולם האמיתי' כפי שהם נקראים, הם באופן גורף וללא יוצאים מן הכלל – מעוררי דחייה. [↑](#footnote-ref-1)