מבוא:

נושאים שנלמד במהלך הקורס:

* אלגוריתמים אקראיים
* אלגוריתמים אלגבריים
* אלגוריתמי קירוב - טוב ב99 אחוזים ולא באופן מושלם.

מטרות כאשר מתכננים אלגוריתם:

* נכונות
* יעילות
* פשטות
* קל למימוש

מדידות יעילות של אלגוריתמים:

אי אפשר למדוד זמנים כדי להגיד האם אלגוריתם יעיל או לא, כיוון וזה לא מדד שעמיד לשינויים טכנולוגים. נצטרך לנתח בצורה תיאורית כי נדרש מדד יציב שלא דורש גורמים שתלויים בסביבת המחשוב. כמו שאנחנו מכירים מקורסים קודמים, נמדוד בהערכה סיבוכיות אסימפטומטית כפונקציה של אורך הקלט שזה בעצם . מודל החישוב שלנו הוא מכונת טיוריניג.

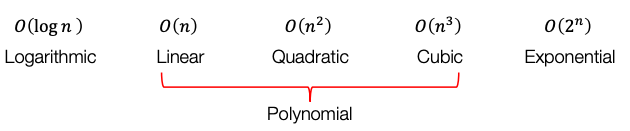
נמדוד את מורכבות המקרה הגרוע ביותר.

לסיכום-נאמר שלאלגוריתם יש זמן סיבוכיות עבור כל קלט באורך . אלגוריתם רץ לפחות צעדים.

נדבר בהמשך גם על סיבוכיות זיכרון.

*סיבוכיות אסימפטוטית-*

נאמר שפונקציה אם יש קבוע שמקיים:   
  
שיעורי צמיחה של סיבוכיות:



אלגוריתמים אקראיים|הסתברותיים|רנדומלים:

מיון מהיר Quicksort  
 הרעיון הכללי והידוע- בוחרים pivot שהוא איבר בx ונפצל ל2 קבוצות- כל האיברים שגדולים מהאיבר וכל האיברים שקטנים מהאיבר וממשיכים כך בצורה רקורסיבית.

קלט: קבוצת מספרים כלשהם.

פלט: כל האלמנטים בתוך ממוינים בסדר עולה.

שלבים:

1. לבחור אלמנט pivot
2. הגדר Y כל האלמנטים שקטנים מ  ו Z אחרת
3. בצורה רקורסיבית לחזור על כל חלק Y וגם עבור Z

**סיבוכיות ריצה:**

תלויה בpivot שבחרנו.

* אם  הוא החציון, ישנם אלמנטים בY וגם בZ ובצורה רקורסיבית נקבל (2 בעיות בגודל כל אחת). הפתרון הסופי יהיה
* אם  הוא או אפילו הפתרון יהיה זהה, גם עבור או באופן מפתיע.
* אם  הוא הקיצוני ימין או שמאל זה האפשרות הכי גרועה כיוון והסיבוכיות

**אז איך לבחור את הpivot?**

1. באופן שרירותי- לבחור את האיבר הראשון במערך. בדרך כלל זה עובד על רוב הקלטים.
2. לבחור את החציון לפי אלגוריתם כלשהו בזמן
3. לבחור בצורה אקראית!

\*\* אלגוריתמים אקראיים- אנחנו מניחים שלמחשב יש גישה למאגר של ביטים בתור קלט. הפלט יהיה ההתפלגות של פלטים. זמן הריצה של האלגוריתם הוא כעת ההתפלגות. אנחנו עדין שוקלים את ניתוח המקרה הגרוע.

אלגוריתם אקראי בquicksort-

אם נבחר pivot בצורה אקראית, תוחלת זמן ממוצע (על ההגרלות) היא

**עובדה זו נכונה עבור כל קלט להבדיל מהדוגמה הקודמת שיש פלט שהיא לא נכונה עבורו.**

ההסתברות לכישלון תהיה קטנה כל כך. יש אפשרות אבל עם זמן ארוך יותר.

מציאת Prime numbers:

מטרה: בהינתן למצוא מספר ראשוני עם ספרות.

שימושים אפשריים: למציאת מפתח RSA להצפנה- למצוא 2 מספרים כאלו ולהכפיל אותם.

אלגוריתם דטרמיניסטי: להתחיל מ00..100 עם ספרות ולבדוק אחד אחד. ככל הידוע זה לוקח **.** כי במקרה הגרוע עוברים על כולם.

אלגוריתם רנדומלי: לבחור מספר רנדומלי שלם עם ספרות ולבדוק ראשוניות עם אלגוריתם כלשהו.   
כדי לבדוק אם מספר ראשוני נרוץ עד השורש ( או אלגוריתמים אחרים).

הפלט: ההתפלגות. מה הסיכוי שנבחר במספר ראשוני.

מה הסיכוי לבחור בדיוק מספר ראשוני?   
 השבר ( ההסתברות) של בחירת מספר ראשוני עם ספרות הוא . לא נשמע טוב כל כך.

אם נחזור על האלגוריתם הזה פעמים באופן בלתי תלוי?

נניח ש הוא האופציה שלא מצאנו מספר ראשוני בריצה הI.

=> אז ההסתברות שלא מצאנו מספר ראשוני היא

=> הסיכוי לכישלון הוא

ההסתברות שנכשלנו זה מספר קטן וזניח.

באופן כללי לגבי אלגוריתמים:

* כאשר הקלט תמיד נכון כמו במיון מהיר: אלגוריתם זה נקרא לאס וגאס או במילים אחרות zero error. אין שגיאה, זמן ריצה רע.
* כאשר הקלט הוא רע עם סיכויים נמוכים כמו במציאת ראשוני: נקרא מונטה קרלו או במילים אחרות bounded error. יש שגיאה לטעות אבל קטנה, זמן ריצה טוב.

בתרגיל הראשון נראה שהאלגוריתם הראשון של zero error הוא טוב יותר.

מציאת חיתוכים בגרף:

ישנו גרף קשיר.   
הנחה: הגרף קשיר כלומר כולם מחוברים.

חיתוך בגרף מחלק 2 קודקודים לשתי קבוצות זרות .

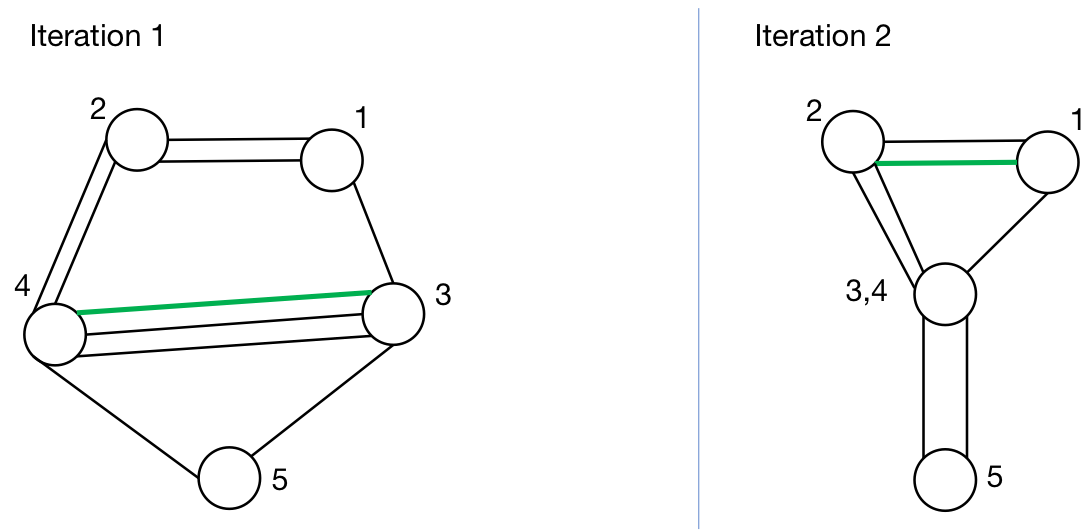
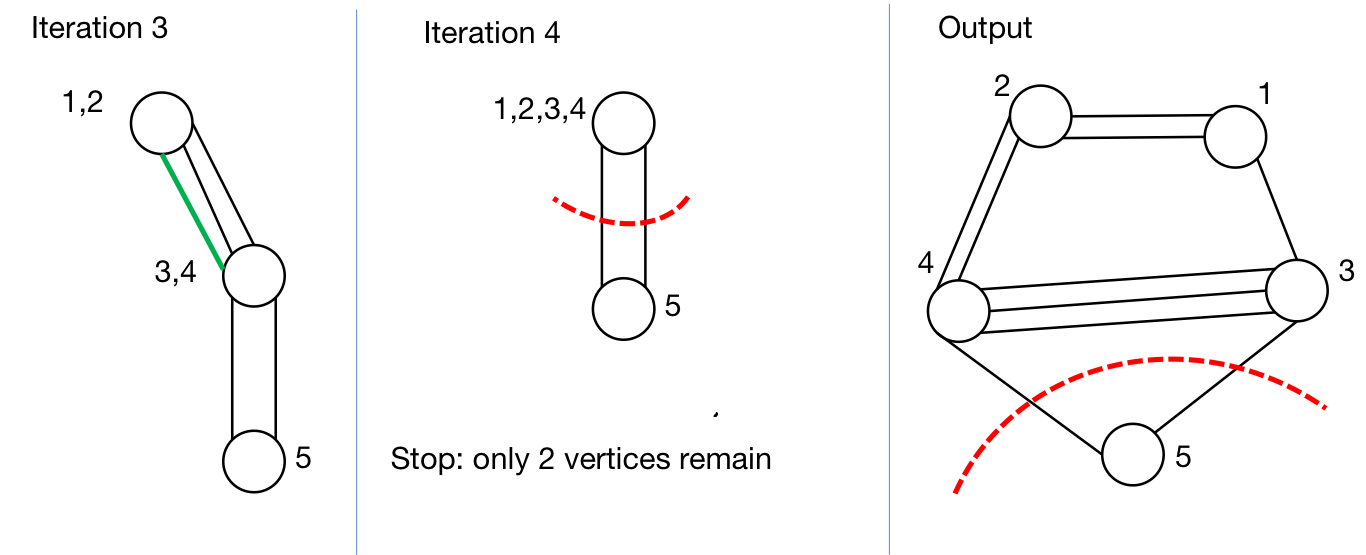
חיתוך מינימלי ומקסימלי:

* למצוא את החיתוך המקסימלי זה בעיה שהיא NP קשה. אי אפשר לעשות את זה בזמן פולינומי. תזכרו את זה כי זה יעשה קאמבק בסוף הקורס.
* מציאת החיתוך המינימלי ניתנת לפתרון ביעילות באמצעות זרימה מקסימלית. נראה אלגוריתם רנדומלי לפתרון.  
  חתך מינימלי לא חייב להיות יחיד.

אלגוריתם Karger’s min-cut:

1. אם יש שני קודקודים בגרף - סיימנו
2. אחרת כל עוד יש יותר מ2 קודקודים:  
   2.1 תבחר קשת רנדומלית   
   2.2 לאחד את לחוליה חדשה ולמחוק כל הקשתות אליו.

דוגמת ריצה:



נרצה להוכיח שהדבר הזה באמת טוב על ידי האלגוריתם האקראי.

K גודל החתך המינימלי ויש k קשתות שחוצות אותו. ההסתברות שהאלגוריתם לא בוחר את אחת הקשתות שחוצות למשל m קשתות -> בוחר ו לא בוחר.

ההבחנה שצריך לשים לב אליה: הדרגה של כל קודקוד בגרף היא לפחות k.

אם הדרגה קטנה מk - הדבר הזה נותן לנו הערכה למספר הצלעות בגרף.

ניתוח אלגוריתם Min-Cut

טענה: לכל קודקוד בגרף , הדרגה שלו היא .

הוכחה: אחרת הוא חיתוך קטן יותר. *(?? לבדוק מה זה אומר)*

תוצאה ישירה: (סכום הקשתות)

ההסתברות שבחרנו קשת שחוצה חתך AB בפעם הראשונה היא

ההסתברות שלא בחרנו קשת שחוצה את החתך AB בפעם הראשונה היא

עכשיו יש פחות קשתות וגם פחות קודקודים. לכן מספר הקודקודים הוא והחתך עדין לפחות

כל איטרציה נוריד את הi בעצם ונקבל

ההסתברות שהחתך שרד את כל האיטרציות הוא (נוסחת בייס)

קיבלנו חתך! נעשה את זה פעמים וניקח את המינימלי מבניהם.

ההסתברות ש לא נבחר אף פעם קטנה באופן אקספוננציאלי בn. הסבר:

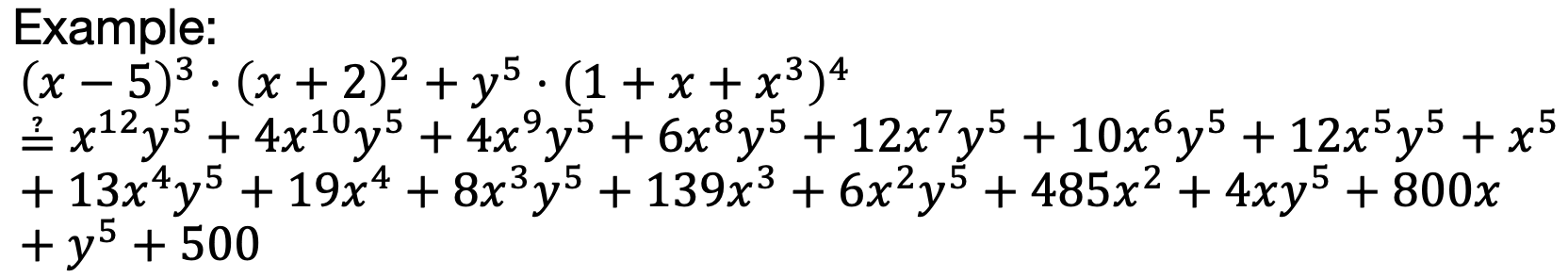
מה זה אומר מספר חתכים מינימלי? כמה חתכים מינימליים יכולים להיות בגרף?

האלגוריתם גורר את המשפט הזה - מספר החתכים המינימלי

מבחן זהות פולינומים:

הבעיה: בהינתן שני פולינומים בn משתנים , נבדוק האם הם זהים.

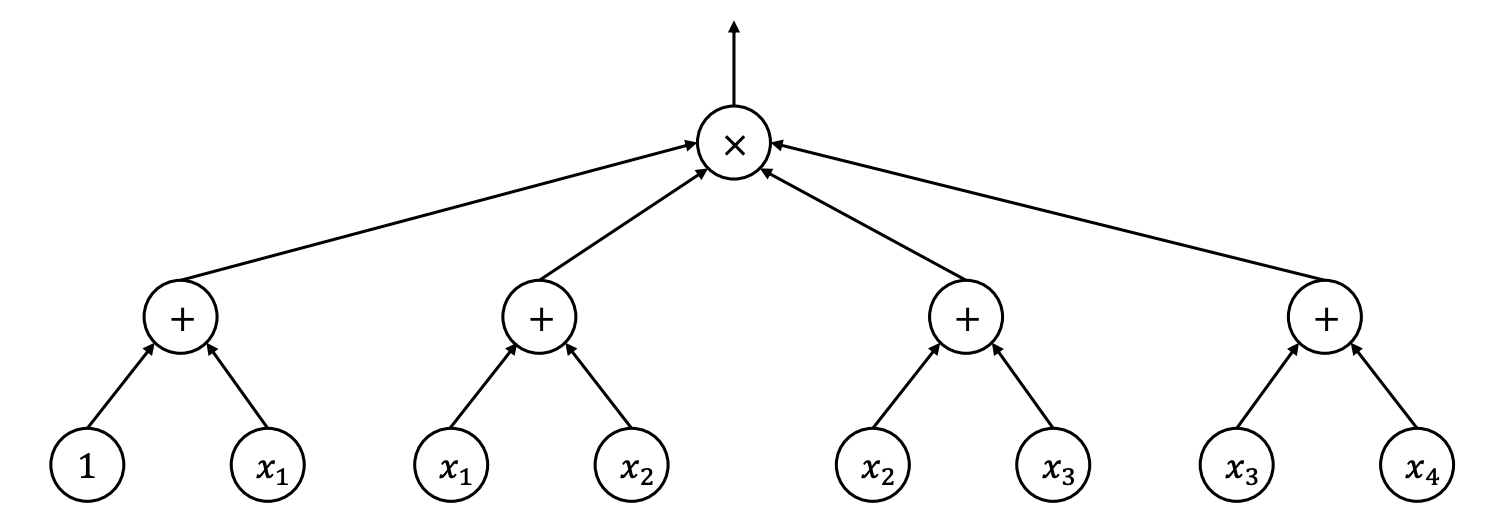
זהות היא זהות מונומית - האם המקדמים זהים (כאשר נפתח את כולם)  
יש דוגמאות פשוטות שנוכל להכריע מיד וזה קל. כמו כן יש דוגמאות מורכבות שצריך לפתוח סוגריים.



אלגוריתם דטרמיניסטי: לפתוח סוגריים ולהשוות, נבדוק שהמקדם אותו מקדם ואז נכריע.

חסרון: ייקח זמן אקספוננציאלי לפתרון.

מודל טבעי לייצוג- נציע את הפולינומים כקלט כפורמולה (נוסחה):

נוסחה זו איזשהו עץ שבעלים שלו יש משתנים/קבועים והקודקודים שהם לא עלים מסומנים על ידי חיבור או כפל.  


גודל הקלט הוא גודל הגרף = כמה קודקודים יש בנוסחה הזו.

לדוגמה- לפולינום יש 8 פתרונות אפשריים.

הפולינום מחשב את

דרך נוספת לפתרון: הצבת ערכים. אך יש בעיה- אפשר להציב ערכים כך שיתקבל שוויון למשל x+5=y+3 . ערכים אקראיים לx y יתנו לנו פתרונות שונים.

פתרון אקראי:

PIT: בהינתן אנחנו יכולים להגדיר נוסחה חדשה ולבדוק האם במקרה שלנו h=F

רעיון: נבחר **מספרים אקראיים** ונבדוק האם

כלומר מחשבים תמיד את פולינום האפס. הוא שווה זהותית ל0, זה אומר תמיד 0, לא יש הצבות שהן 0.

החישוב הזה יהיה בזמן פולינומי.

ועכשיו השאלה שלנו היא- מה הסיכוי ש?

*ידוע לנו* שלפולינום ממעלה D יש לפחות D שורשים.

המשפט הזה כמובן לא נכון עבור פולינום ממעלה x-y. עבור הפולינום ממעלה x-y יש ∞ שורשים. כלומר אפשר לחשב לכל .

מה שכן נכון הוא ההכללה הבאה שנקראת הלמה של שוורץ-זיפל:

הלמה של Schwartz Zippel

אם f הוא פולינום שונה מאפס ויש בו n משתנים. המקדמים של f רציונליים כלומר . d זו המעלה של הפולינום.

Sתת קבוצה של מספרים רציונליים ו זה גודל של תת קבוצה.

אז:

כאשר נבחר את בהתפלגות אחידה מ S. ובהסתברות בלתי תלויה אחד מהשני. כלומר לא תלוי ב

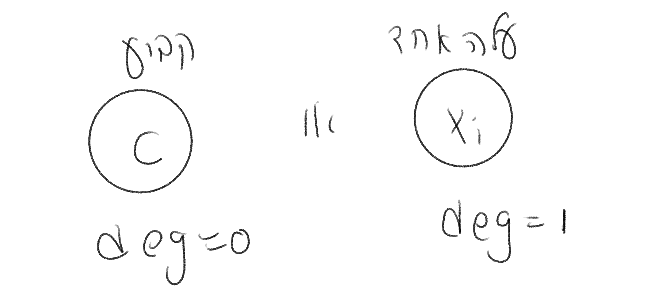
לדוגמה אם S = 10\*d אז ההסתברות שכל זה יקרה הוא לכל היותר עשירית.

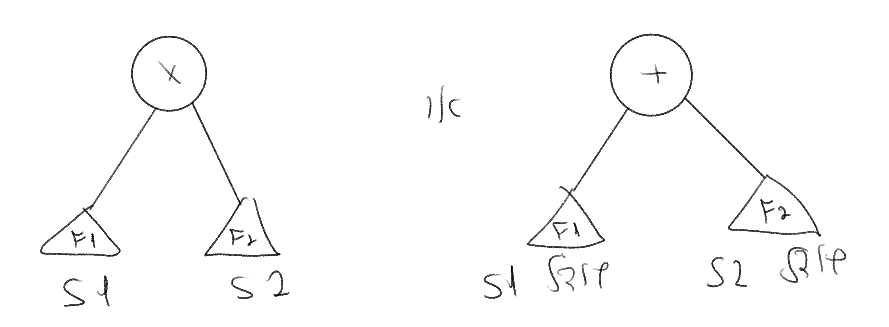
*הערות על הלמה:*

* ההוכחה תהיה בתרגיל בית באינדוקציה.
* בייצוג כנוסחה פחות ברור מה הדרגה, אבל לא נצטרך לדעת מה הדרגה בדיוק אלא חסם עליון של הדרגה. ההנחה היא שבנוסחה יש גודל S (גודל של נוסחה היא מספר הקודקודים בה) , מחשבת פולינום שמעלתו היא לכל היותר S. הוכחה באינדוקציה:

**הטענה: נוסחה בגודל S מחשבת פולינום ממעלה לכל היותר S (הערה: או 0).**

הוכחה באינדוקציה:

בסיס: S=1 . נוסחה עם קודקוד אחד זה נראה ככה:   
  
צעד האינדוקציה: ניקח נוסחה בגודל היא יכולה להיות אחת מהצורות הבאות:



יכול להיות או כפל או חיבור.

הגודל של F הוא (הפלוס אחד הוא בגלל הקודקוד הנוסף של הסוכם או המכפלה)

הערות : F זה הנוסחה וS זה הגודל

אם מהנחת האינדוקציה ) כלומר המעלה של היא לכל היותר (.

וגם ) כלומר המעלה של היא לכל היותר (.

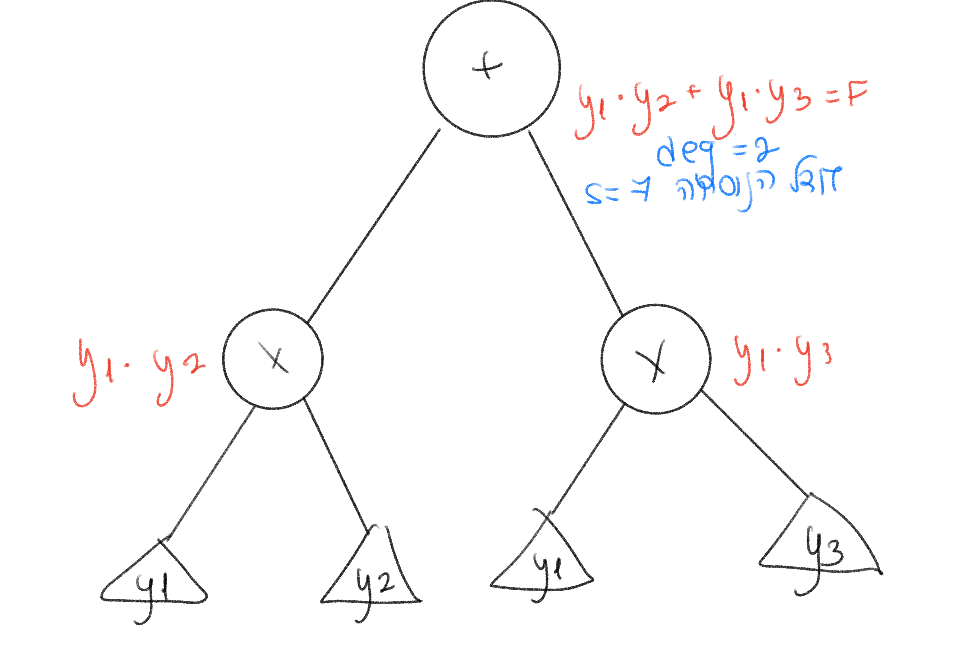
בנוסף

כאשר זה הגודל של F.

אם מהנחת האינדוקציה היא ) כלומר המעלה של היא בדיוק המעלה של ).

בנוסף

דוגמה :



כעת יש לנו את כל הכלים שאנחנו צריכים בכדי **להראות את האלגוריתם האקראי**:

הקלט: נוסחה F בגודל S

הפלט: נגיד כן אם ולא אם

אלגוריתם:

* + - 1. הגדר את קבוצת המספרים השלמים מ1 עד 2S.
      2. נבחר בצורה אחידה ובלתי תלויה ונחשב את F על המספרים האלה. . אם יצא 0 התוצאה תהיה כן , אחרת לא.

האנליזה: אם F=0 יוחזר לנו כן בתוצאה. אחרת על ידי הלמה של שוורץ-זיפל הסיכוי שבחרנו קלטים גרועים היא לכל היותר חצי.

*איך אפשר להוריד את המספר חצי?*

* *להגדיל את S . במקום לבחור 2s אלא לבחור 100.*
* *את שלב 2 לעשות מאה פעמים. ואם תמיד יוצא 0 = השתכנעו שהוא 0. ואם אחד אפילו יוצא 0 אז השתכנעו שזה לא 0.*

*שימושים: שידוך מושלם בגרף דו צדדי-*

גרף הוא גרף דו צדדי כאשר יש n קודקודים בכל צד כלומר

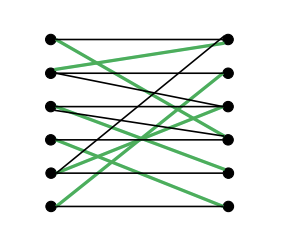
כל קשת מחברת בין צד לצד, אין בין הקודקודים באותו הצד.

שידוך מושלם היא דרך לשדך בין הקודקודים כך שצד אחד מותאם לצד אחר. בירוק אפשר לראות בדוגמה שידוך מושלם.

צריך למצוא אלגוריתם שמקבל גרף ומכריע האם יש בו שידוך מושלם.

יש אלגוריתם דטרמיסטי שמבוסס על זרימה (לא למדנו).

אנחנו רוצים להכריע אם יש שידוך מושלם. בעיה שהיא יותר טבעית לנו היא למצוא את השידוך המושלם. אלא ממש להחזיר את הקשתות הירוקות. זו בעיה שנדון בה בהמשך הקורס. נראה שהבעיה הזו היא מקרה פרטי של בעיית הפולינומים שראינו עכשיו. יש לה אלגוריתם לפתרון פשוט והוא אקראי.

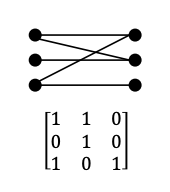
**

*יתרונות נוסף לאלגוריתם הזה:*

* *אלגוריתם יפה*
* *אפשר למקבל אותו – נדבר בהמשך על אלגוריתמים כאלו*

האלגוריתם מתבסס על מטריצת השכנויות של הגרף. היא מטריצה שהיא עוד דרך לקודד את הגרף במטריצה.

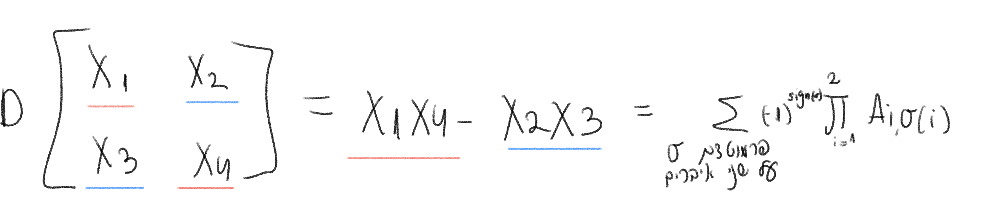
השורות הם קודקודים בצד שמאל והעמודות הן עמודות בצד ימין. ובתא הi,j יש מספר 1 אם יש קו ביניהם ו0 אחרת.



נראה שזו דרך לקודד את הגרף אבל בגלל שזה מטריצה אנחנו יודעים שיש להם ערכים עצמים ודטרמיננטה, קרנל וכו'. מתברר שיש קשרים בין תכונות של הגרף לבין תכונות אלגבריות של המטריצה. זה נס אלוהי. נראה כמה מהן:

* טענה: **אם ב אין שידוך מושלם, אז ה** *. G) זה הגרף שA היא המטריצה שלו.)*  
  הסבר: דטרמיננטה זה איזשהו פולינום בערכים של המטריצה.

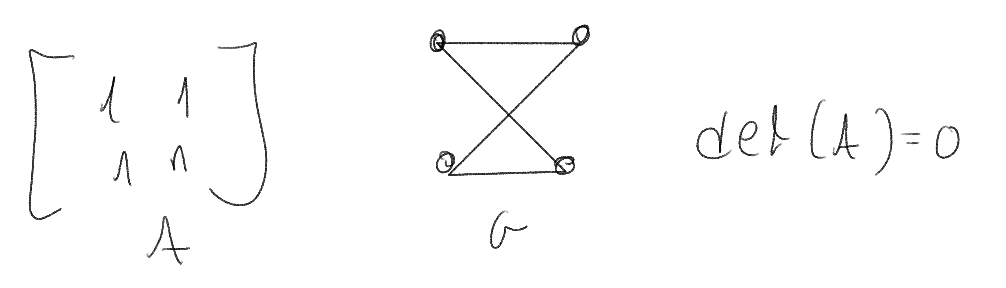
אם כותבים אותה זה יוצא סכום של כל הפרמוטציות של 1 עד n ונכפול את האיברים על האלכסון.

דוגמה שני איברים:

עכשיו באופן כללי- כל האיברים במטריצה יכולים להיות או 0 או 1. אם הם כולם 1 – יש קשת בין אחד 1 ל2 ובין 3 ל8 וכולי- כלומר יש שידוך מושלם בדיוק. אם אין שידוך מושלם 🡨 לפחות אחד מהם הוא 0 וזה מפיל את כל החישוב של הדטרמיננטה ל0 .

*הכיוון השני- האם זה נכון שאם יש שידוך מושלם🡨 det(A)=0 ? התשובה היא לא!*

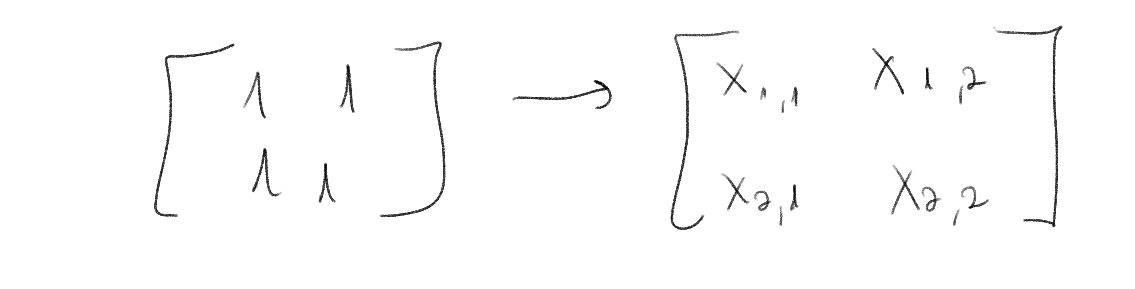
הפרכה:



איך נפתור זאת ונגרום לכך שלא יהיו ביטולים ?

במקום מטריצת שכנויות נמצא מטריצה טיפה אחרת . נשתמש במטריצת שכנויות סימבולית.   
 במקום לשים 1 או 0, נשתמש ב:  
 כאשר אין קשת נחזיר 0 🡨

אם יש קשת נחזיר משתנה ששונה מכל קשת שנקרא לו ככה 🡨

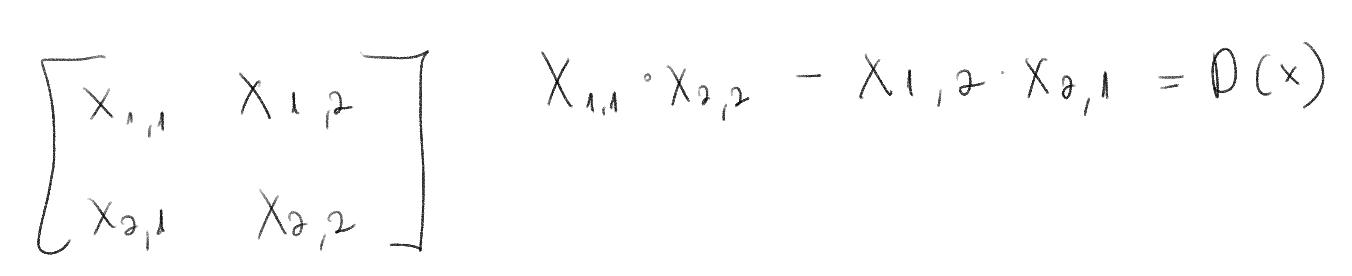


זו המטריצה שהאיברים שלה לא קבועים הם משתנים. נקבל איזשהו פולינום. ועכשיו יש טענה חדשה:

* טענה: **ה** (הפולינום שלו) **אם ורק אם לגרף G יש שידוך מושלם***.*  
  הוכחה: אנחנו יודעים שנוסחה לחישוב דטרמיננטה היא

אם אין שידוך מושלם אז דטרמיננטה הוא אפס ( אותו טיעון כמו קודם).

מה קורה אם יש שידוך מושלם?

נחשב את הדטרמיננטה של המטריצה שראינו:  


ברגע שיש לי את המונום הראשון, הוא לא יכול להתבטל אם אף מונום אחר כי אף אחד לא שווה לו.. כל פרמוטציה פה מופיעה רק פעם אחת. שלא כמו קודם היה 1 פחות 1.

כל מה שעשינו היה להתגבר על החישובים האלה.

* שאלה: **ה** לפי הפולינום?  
  פתרון: ידוע לנו לפולינום הזה יש לכל היותר משתנים. המעלה שלו היא לכל היותר .  
  ידוע לנו שלכל הצבה של משתנים אפשר לחשב על ידי פירוק של מטריצות. או בעזרת אלגוריתם לחישוב דטרמיננטה ( שנראה בהמשך) או בעזרת גאוסיאן.

*הקטנת השגיאה לטעות*

*כל אלגוריתם שראינו עד עכשיו, תמיד הייתה שגיאה כלשהי לתשובה לא נכונה, אבל אפשר להוריד את זה על ידי מספר רב של חזרות. עכשיו אנחנו רוצים לדבר על זה בצורה שיטתית. נדון בסוגיה איך מפחיתים שגיאה באלגוריתמים הסתברותיים. נדבר ספציפית בשאלות של הכרעה כרגע (כן או לא).*

שפה- בעיה חישובית שהתשובות אליה כן עולות. והשפה היא כל הקלטים שהתשובה עליהן היא כן.

לדוגמה: שפה- קבוצת כל הגרפים שיש ביניהם שידוך מושלם.

*נניח על כל הבעיות (השפות) A שיש להן אלגוריתם אקרא יעיל שיש להם את המשתנים הבאים:*

*הסבר:  
כלומר אלגוריתם אקראי יעיל שאם משהו בשפה- נקבל אותו בהסתברות גדולה או שווה מ . דוגמה  
ואם הוא לא בשפה- נקבל אותו בהסתברות קטנה או שווה ל דוגמה*

*ואם נרשום את זה לצורך ההגדרה: =*

*כדי להקטין את השגיאה נרצה ש יהיה כמה שיותר קרוב ל1 ו יהיה כמה שיותר קרוב ל0.*

*בדר״כ נקבל אלגוריתם עם מאותחלים עם מספרים רחוקים. המטרה לבנות אלגוריתם אחר שיצמצם את הפער ולא יותר מדיי רחוק בסיבוכיות מהאלגוריתם ההתחלתי.*

*צריך להיות קטן ממש מ .*

*סוגי שגיאות:*

*שגיאה חד צדדית:*

* טענה:

הסבר: *אם יש אלגוריתם שעונה על התנאי הבא :*

*אז עכשיו יש אלגוריתם אחר שמקיים את התנאי:*

*\*\*כמובן שזה מספר שרירותי קטן שנבחר בלי מחשבה מאחוריו.*\*\*איפה שיש שגיאה הורדנו הסתברות. איפה שאין שגיאה לא הורדנו והשארנו כמו שהיה.

הוכחה: ניקח קבוע קטן . וניקח אלגוריתם  *שעונה על התנאי הבא :*

*ניצור אלגוריתם חדש שבו ניקח ונריץ את אלגוריתם M כמות פעמים t:*

*ואז מתקיים התנאי הבא:*

*כלומר אם נחזור על זה t פעמים נקבל .*

*\*\* שימו לב שאפשר לבחור את להיות אקספוננציאלי קטן ב n. כלומר וזה עדין ירוץ בזמן פולינומי.*

*שגיאה דו צדדית:*

* טענה:

הסבר: *אם יש אלגוריתם שעונה על התנאי הבא :*

*אז עכשיו יש אלגוריתם אחר שרץ t פעמים ומקיים את התנאי:*

הוכחה: ההוכחה תהיה מאוד דומה אנחנו *רק צריכים לפתח קצת כלים מתמטיים כדי לסיים את ההוכחה*.

ידוע לנו שיש אלגוריתם עם הסתברות מסוימת, ניקח את האלגוריתם ונריץ אותו t פעמים.

נניח שיש לנו משתנים אקראיים בלתי תלוים ובאותם התפלגות .

ההסתברות שהוא וגם .

האלגוריתם שלנו טועה במאורע שבו (סכום המשתנים המקרים הוא לכל היותר t חלקי 2 – יותר מחצי פעמים האלגוריתם אמר כן)

התוחלת לכל i היא

התכונה הכי שימושית של תוחלת שהיא לינארית

סכום התוחלות הוא :  - כי יש פה סכום וכל החד הוא שליש.

האלגוריתם טועה מתי שהמשתנה המקרי הזה הוא לפחות ההסתברות שמשתנה מקרי רחוק מהתוחלת שלו.

נשתמש ב: **אי שוויון מרקוב**

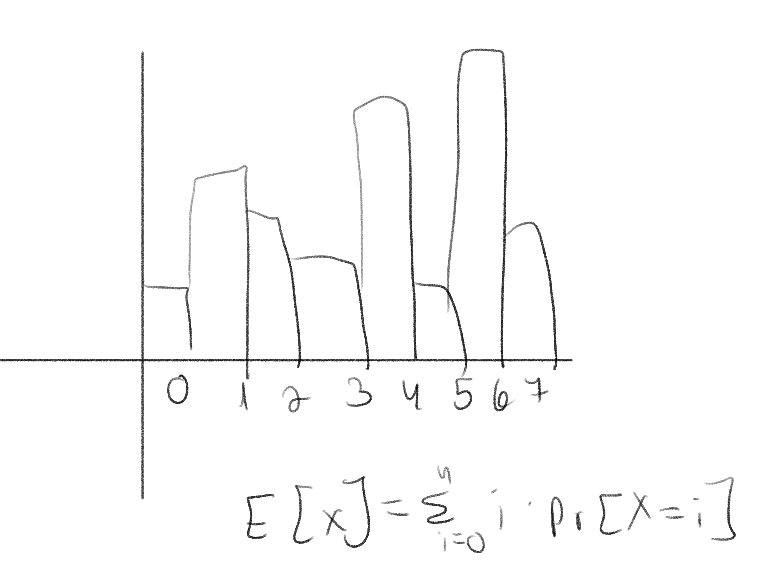
נניח שיש משתנים מקריים כאשר

אז ההסתברות שלכל

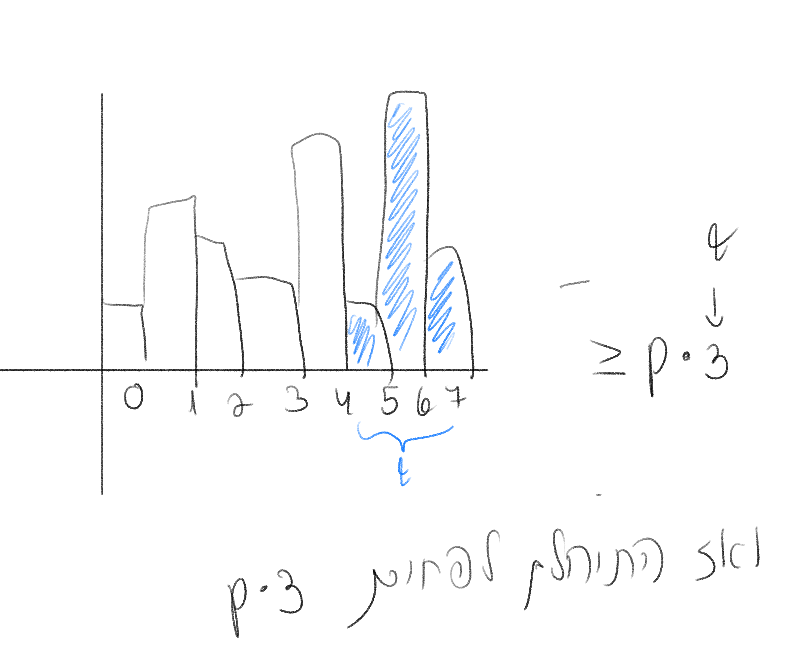
לדוגמה כאשר וגם אז ההסתברות

*זה בעצם אי שוויון שחוסם את ההסתברות שx גדול מאיזשהו פרמטר.*

*האינטואיציה של הדבר הזה היא שאם , רק אם אנחנו עושים ממוצע על המאורע יתרמו לנו לממוצע המשוקלל. אנחנו יודעים שהמשקל של כולם הוא לפחות p וסכום כל הערכים הוא לפחות t.*

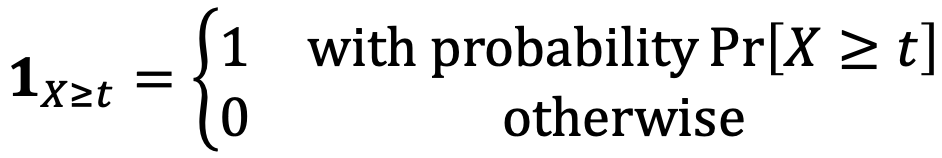
*באופן כללי סכום השטחים האלה צריך להיות 1 בתוחלת:  
*

*ועכשיו נשאל מה סכום ההסתברויות של 5-7?*

**

*ועכשיו אם נבחר בכוונה את p נקבל סתירה.*

*ההוכחה המלאה : נגדיר משתנה מקרי אחד שהוא 1 אם x גדול מt או 0 אחרת.*

**

*ואז ההבנה היא ש נותן לנו ש*

*זה ההוכחה וזה בדיוק האי שוויון.*

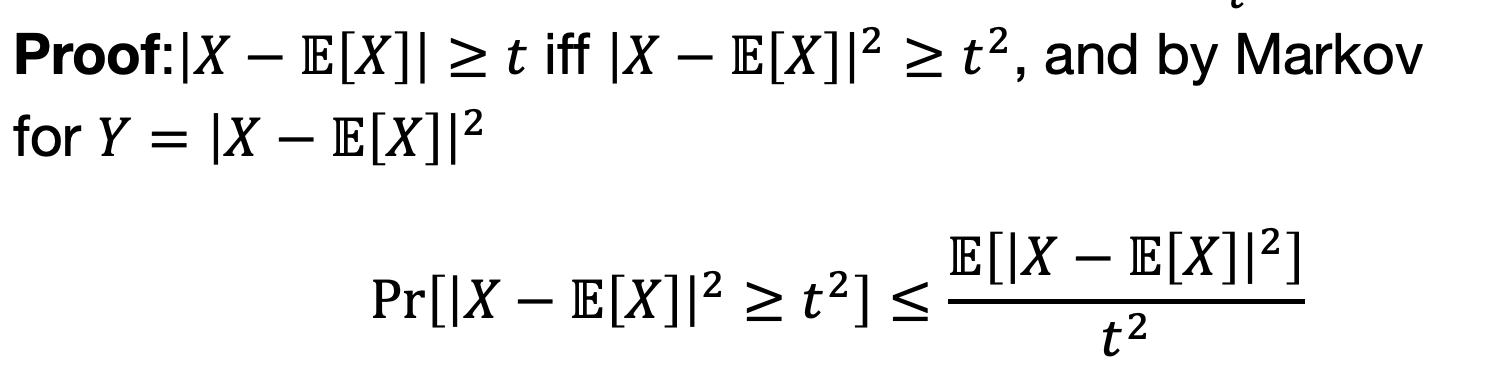
* *לא השתמשנו בזה שזה התפלגות בינומית.*
* *השתמשנו רק בזה שאיקס אי שלילי.*
* *קיבלנו חסם שהוא לא כזה טוב.*
* *נתחיל להשתמש בתכונות של איקס כדי לשפר את החסם.*

נשתמש ב: **אי שוויון צ׳בישב**

משפט: כאשר

הסבר: ההסתברות שאיקס פחות התוחלת שלו, זה לפחות (בערך מוחלט) t. כלומר מה ההסתברות שהוא רחוק מהתוחלת שלו. כל זה לכל היותר זה השונות של x חלקי t בריבוע.

הוכחה: שורה אחת, לקחת את אי שוויון מרקוב ולעלות בריבוע.



הערות:

ככל שמגדילים את m מקבלים שאיפה מהירה יותר לאפס.

אפשר לקחת את t להיות שורש m כלומר

לא לקחנו את העובדה שx. הוא סכום של m משתנים בלתי תלוים, אלא השתמשנו בעובדה שהם אי תלוים בזוגות.

נשתמש ב: **חסם צרנוף-** משתמש בזה שכולם בלתי תלוים.

ההסתברות שx סוטה מהתוחלת שלו היא לכל היותר הביטוי עם הקבועים

היא תהיה אקספוננציאלי קטנה מm.

כאשר כאשר מתפלג יוניפורמי בין {-1,1} והמשתנים בלתי תלוים.

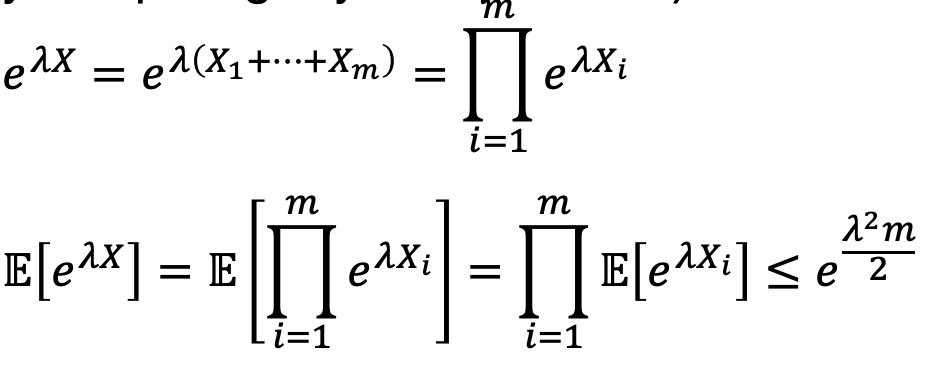
התוחלת היא כי זה סכום של אפסים

ואנחנו רוצים להראות שההסתברות שבה כאשר .

הרעיון הוא לקחת את אי שוויון מרקוב ולהעלות דברים באקספוננט (בחזקת).

הוכחה:

נבחר פרמטר ונסתכל על המשתנה המקרי ונחשב את התוחלת

הדרך: 

בזה שלקחנו אקספוננט הרווחנו שעכשיו הסכום הופך למכפלה.

מסקנה מפה- התוחלת של חסומה ב

בזה סיימנו את העיסוק בזה. ונעבור לאלגוריתם אחר אקראי.

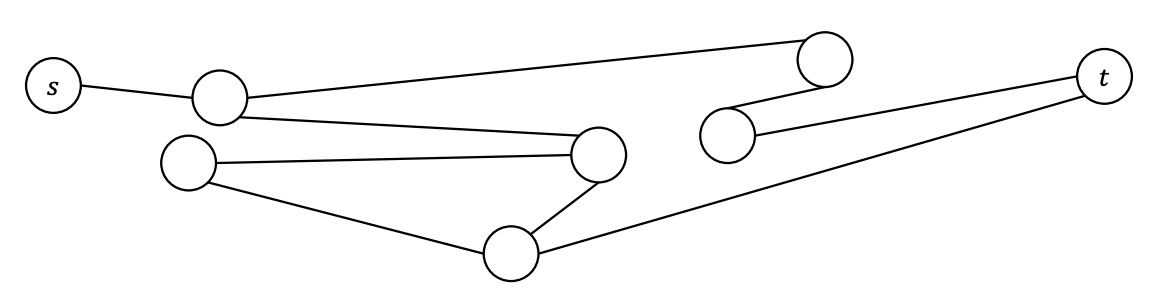
קשר בין גרפים לא מכוונים:

הבעיה: בהינתן גרף G לא מכוון ושני קודקודים s,t ונרצה לברר האם קיים מסלול בין s לt בG.

מוטיבציה: כיום אנחנו יודעים לפתור את זה באמצעות BFS, DFS. מריצים מs ורואים אם פגשנו את t בדרך. זה אלגוריתמים שרצים מהר בזמן לינארי. אבל אנחנו רוצים משהו יותר טוב מזה.. הסיבה היא שבBFS DFS נצטרך להשתמש בהרבה זיכרון (הרי כל פעם נסמן לאן הגענו) אז מניחים שיש לנו גישת r/w לקלט – לסמן על קודקודים. ואפשר לחשוב על מקרים שזה לא מעט טוב. נרצה זיכרון נוסף כי אולי הקלט מאוד גדול ולנו אין הרבה זיכרון, אולי זה יהיה read only...

נרצה זיכרון לוגריתמי – כלומר נרצה לכתוב קוד שמשתמש במספר קבוע של משתנים.

השאלה היא איך בפועל נעשה את זה בלי לזכור הרבה דברים בזיכרון שלנו:

**

רעיון: נתחיל בs ונבחר את היעד הבא שלנו בצורה אקראית לפי שכן אקראי עד t. וצריך לקוות שזה יגיע לt בהסתברות גבוהה.

הערה -חשוב להדגיש שזה עובד לגרף לא מכוון.

דוגמה שדבר כזה לא יעבוד בגרף מכוון:



פה יש גרף מכוון, אבל מכל מקום במסלול יש קשת שחוזרת ל s. וכל פעם חוזרים להתחלה וצריכים לטפס את כל הסולם.

אם הגרף הזה היה לא מכוון אז לחזור לs זה לא כזה נורא, כי באיזשהו הסתברות נחזור לאיפה שהגענו.

האלגוריתם :

קלט: גרף לא מכוון G עם n קודקודים שבתוכם s,t

פלט: להגיד כן אם יש מסלול s לt. ולהגיד לא, אחרת.

צעדים:

נתחיל מs=u

תעשה פעמים (מספר שרירותי):  
2.1 תבחר שכן רנדומלי v מתוך u  
2.2 אם v=t תחזיר כן ותעצור  
2.3 אחרת, תמשיך ותשנה את u=v וממשיכים..

תחזיר לא – כעבור מספיק זמן וכי צריך לעצור מתישהו.

מה אפשר לטעון על האלגוריתם הזה?

* כמה משתנים יש באלגוריתם – u, v, counter to - סה"כ קל להשתכנע שהוא במעט זיכרון.
* נניח שאין מסלול בין s לt – מה ההסתברות שהאלגוריתם ימצא מסלול? 0. האלגוריתם תמיד יגיד לא.
* התהליך נקרא הילוך מקרי.
* אם יש מסלול מs לt- אז בהסתברות גבוהה התהליך מגיע לt וצריכים כלים לנתח את ההתפלגות שמתקבלת.

הנחות שיקלו עלינו :

* נניח שG הוא גרף D-רגולרי (הדרגה של כל קודקוד היא D). נניח את זה כי אנחנו תמיד יכולים להוסיף לולאות עצמיות. זה יקל עלינו להניח את זה.
* נניח גם שG הוא קשיר. אנחנו מנסים לנתח את המקרה שאפשר להגיע מs לt. כלומר שt הוא ברכיב הקשירות של s. מקרה כזה אפשר להניח שהגרף כולו הוא קשיר.

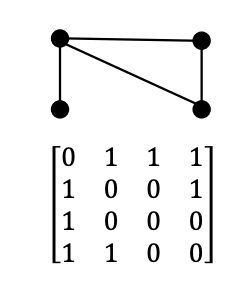
עכשיו אחרי ההנחות, אם נעשה הילוך מקרי שהוא מספיק ארוך, ההתפלגות תהיה קרובה להתפלגות האחידה.

בהתפלגות אחידה ההסתברות שנגיע לt היא

בפרט זה אומר שההתפלגות שלנו היא כמעט אחידה. ההסתברות שהגענו לt היא תהיה לפחות .

אפשר לחזור על האלגוריתם הרבה פעמים ולהגיע למצב שההסתברות שנגיע לt היא 0.999 אחוז – וזה יכול להיות קבוע אחר שקרוב ל1.

*מטריצת השכנויות שכבר הופיעה בעבר A:*



הפעם : השורות זה הקודקודים בגרף והעמודות זה הקודקודים בגרף. גרף בn קודקודים היא מטריצה n\*n .

והתא הi,j הוא 1 אם יש קשת בין iלj.

אם הגרף לא מכוון זה אומר שהמטריצה סימטרית (להתעלם מהתמונה של המטריצה פה)

אם הגרף הוא D-רגולרי, סכום כל שורה חייב להיות D.

הדבר שמעניין אותנו זה יהיה מספר הקשתות בין u לv. ההסתברות שנעבור מu לv הוא שזה מספר הקשתות שהולכות אליו.

המטריצה הזו תיתן לנו דרך אלגברית לחשב את הדרך שרצינו.

**טענה**: אם וקטור יש לו n קודקודים, אז זה ההתפלגות על ידי:

* + - 1. בחירת קודקוד ביחס ל
      2. לעשות הליכה אקראית מהקודקוד i צעדים.

**הוכחה באינדוקציה:** מה אנחנו רוצים להוכיח: התפלגות על הקודקודים, **וקטור ההתפלגות הזו שווה ל.**

בסיס: כאשר i=0 לא עושים שום צעדים בגרף.

צעד האינדוקציה: לכל קודקוד v, שווה להסתברות שהולכים מu לv.

בהנחת האינדוקציה זה בדיוק ההסתברות שעשינו את הניסוי הזה והגענו לu אחרי i-1 צעדים.

איך מחשבים את ? באופן כללי זה כפל של מטריצה בווקטור אבל האלגוריתם שלנו לא מחשב את זה. נרצה לדעת איך האלגוריתם שלנו נראה.

***וקטור עצמי***

מטריצה A בגודל

אז אומרים ש הוא וקטור עצמי של A אם קיים כך ש .

ואז מתקיים

ובעצם

מה עוד אפשר להגיד על A? היא סימטרית וסכום כל שורה הוא 1.

טענות:

* למטריצות סימטריות יש n ערכים עצמיים שכולם ממשיים.  
  כך ש הווקטורים העצמיים של הערכים האלה, אפשר לבחור אותם להיות בסיס אורתוגונלי. כלומר המכפלה הפנימית שלהם היא 0.
* אם סכום כל שורה היא 1 אז כל הערכים העצמיים הם בערך מוחלט לכל היותר 1.
* בגלל שכל סכום שורה הוא בדיוק 1 אז יש וקטור עצמי שהוא כולו אחדים. נקבל את הווקטור עצמי שמתאים לערך שהוא 1. אם נכפיל את A בווקטור שכולו 1, נקבל אותו דבר.
* ולסיכום

**הילוך מקרי עצל**

אמרנו שA מתאימה להילוך המקרי בG.

הילוך מקרי שהוא עצל היא מטריצה B. במקום שלב במקום לעבור לקודקוד אקראי, אנחנו בהסתברות חצי לא עושים כלום ובהסתברות חצי נעשה את ההילוך האקראי.

כלומר . זה לא משנה את הייסיגות רק יעכב אותנו.  
  
אם יש לנו ערך עצמי שהוא ווקטור עצמי שהוא .

אז

אז למה זה טוב לנו ?

כאשר אזי   
זה ענין טכני שמקל על הניתוח. אם זה עובד לפי B זה גם יעבוד לפי A.

עכשיו יש את ההבנה ששינינו קצת את האלגוריתם בהסתברות חצי לא עושים כלום ובהסתברות חצי כן עושים את זה.

ותכף נראה למה החסימה בין 0 ל1 יעזור לנו.

תורת הגרף הספקטרלית

עכשיו אנחנו נראה שיש קשר בין הערכים העצמים והווקטורים העצמים של המטריצה הזו לבין התכונות של הגרף.

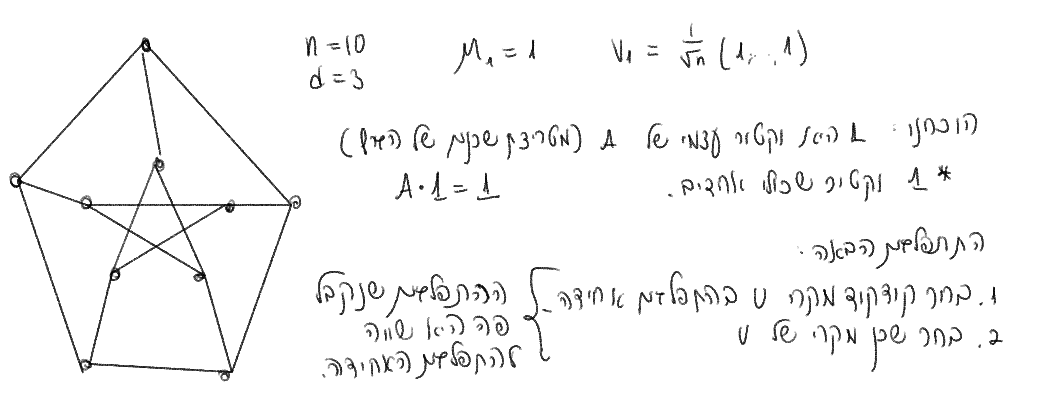
הערכים העצמים של A מקודדים למטריצה. לא נוכיח את זה בינתיים.

דוגמאות בגרפים רגולריים (כי זה יותר נחמד):

- אפשר להוכיח שG הוא גרף דו צדדי (אפשר לחלק את קודקודיו לשתי קבוצות שמאל וימין) אם

- G הוא גרף קשיר אם הערך העצמי השני הוא קטן מאחד.

התכונות היחידות שאנחנו נצטרף מתורת הגרפים הספקטרלית:

* הערך העצמי הוא 1 והוא מתאים לווקטור עצמי שכולו 1 : (נרמלנו ב1 חלקי שורש n כדי שסכום הריבועים יהיה 1). מתאים להתפלגות האחידה. בגרף רגולרי ההתפלגות האחידה היא וקטור עצמי.  
  הוכחה:  
  
* אם G הוא קשיר, אז נגיד על הווקטור עצמי השני הוא לא אחד הוא רחוק ממנו . זה לא הדבר הכי טוב שנוכל לכתוב פה אבל זה מה שנוכיח.

הוכחה בתרגיל בית.

אינטואיציה למה הולך לקרות:

בואו נזכר שיש לנו את  ההתפלגות ההתחלתית שנרשום כצירוף לינארי של הווקטורים העצמיים. אפשר לעשות את זה כי יש לנו n וקטרים עצמיים שהם בלתי תלוים ואורתוגונליים. לא יודעים מה המקדמים הם תלוים ב

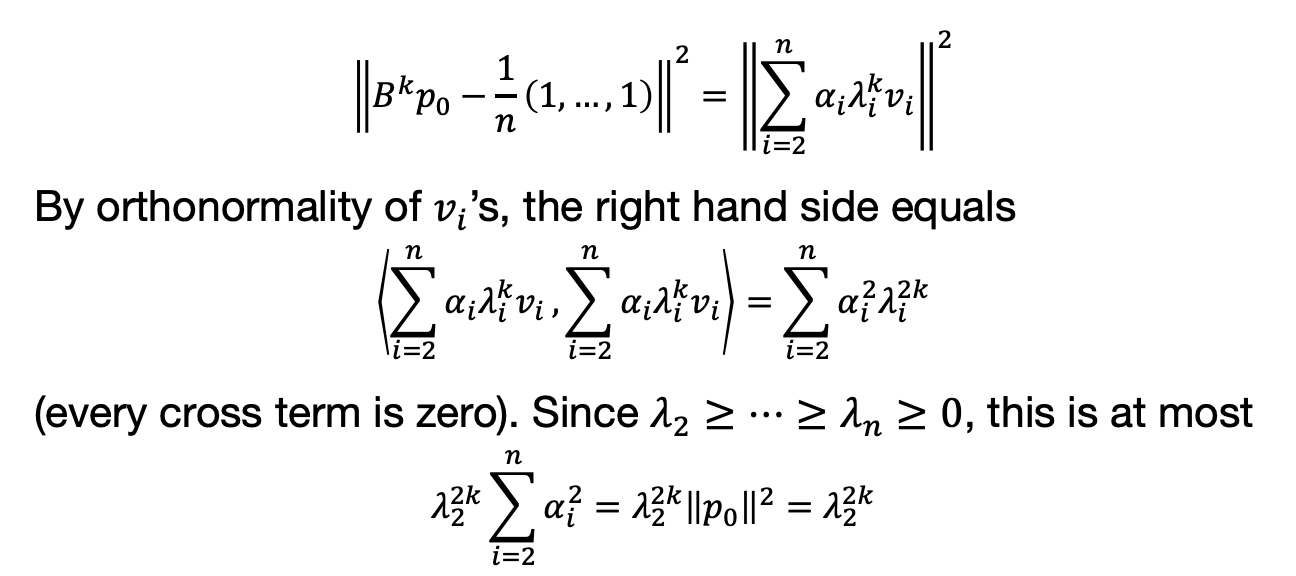
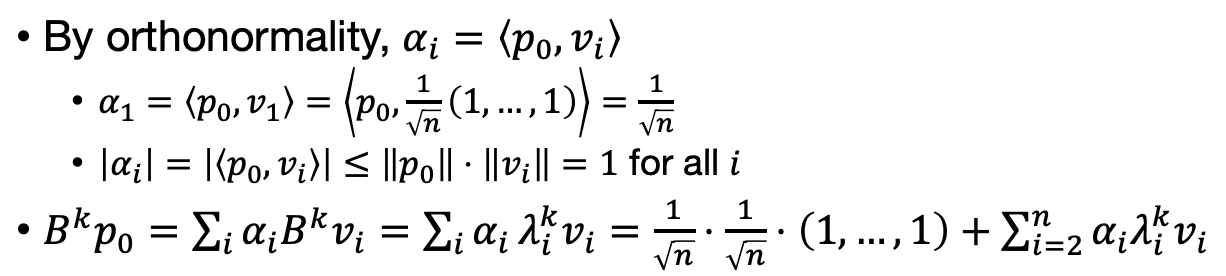
ובעצם נרצה לחשב את המטריצה שעברנו עליה B שזה וזה קצת מסובך אבל מלינאריות אנחנו נחשב ונדגים.

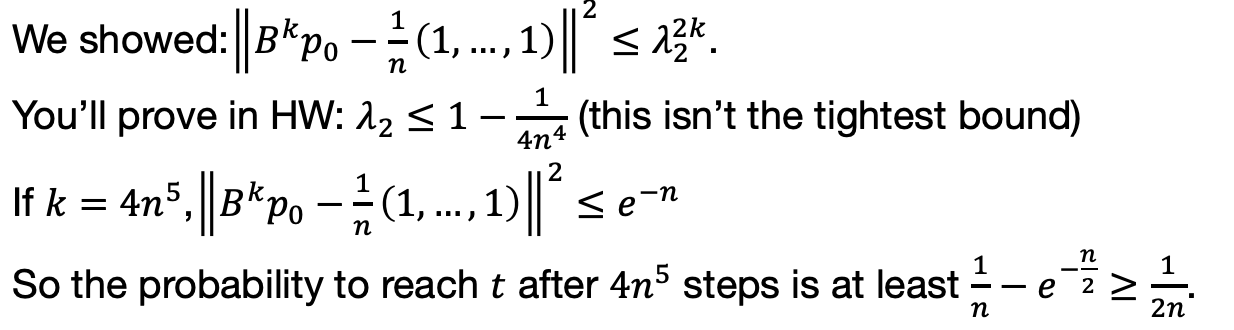
הנקודה החשובה היא שניקח k הולך וגדל. למשל יישאר 1 אבל ככל שk גדל זה ידעך וישאף ל0.

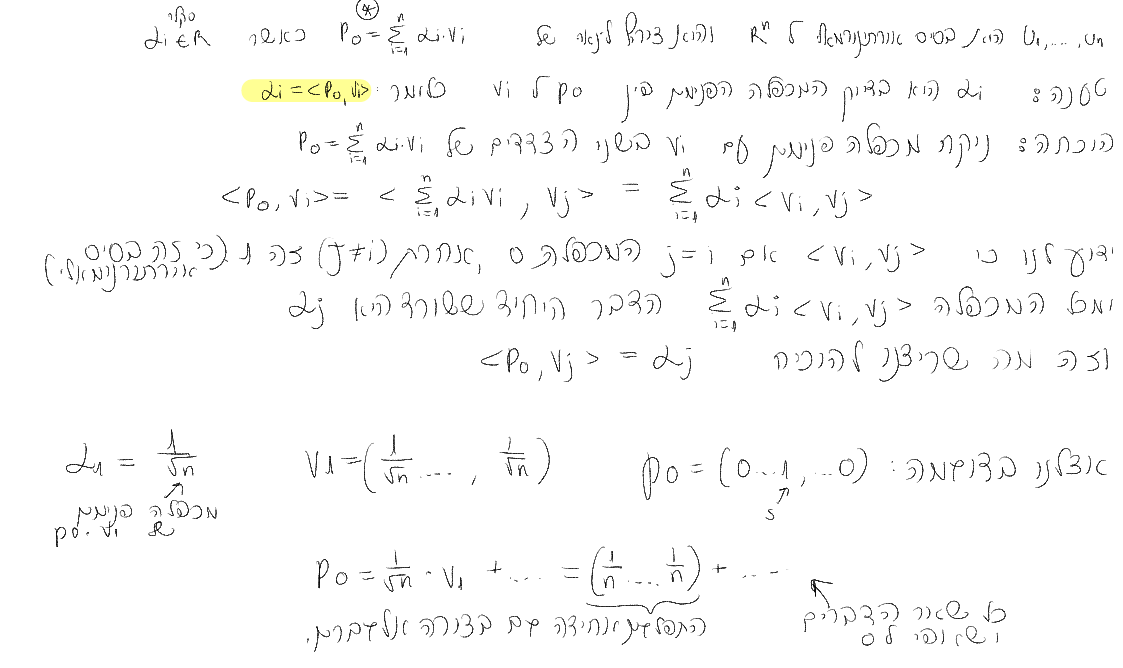
בסוף אם הכל קטנים ואפילו 0 אז נשאר עם ההתפלגות האחידה בלבד עד כדי שגיאה קטנה.

הוכחה בצורה פורמאלית:

נכתוב את ההתפלגות כצירוף של וקטורים עצמיים. נכתוב את זה כי הם אורתוגונליים, בלתי תלויים ויש n מהם כלומר הם פורסים את כל המרחב.(ההוכחה נמצאת בכתב היד שלי למטה):







הנקודה המרכזית היא הניתוח הספקטרלי. לקחת את מטריצת השכנויות ולהבין מה זה אומר על הגרף.

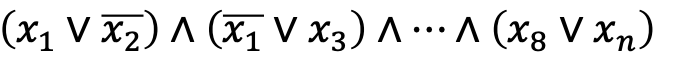
והסיבוכיות היא . אמנם זה יותר גרוע מהסיבוכיות קודם אבל זה לא אקספוננציאלי אלא פולינומי.

*שימושים: 2SAT-*

זה פסוקים בצורת cnf שנרצה לספק (למצוא השמה למשתנים שמספקת את הנוסחה הזו). לכל משתנה יש 0 או 1.

Sat היא בעיה np קשה. אי אפשר לבחור אותה בזמן פולינומי.

אבל ב2 sat היא בעיה קלה (יש אלגוריתם אקראי פשוט שמוכיח אותו).



קלט: נוסחה בצורת cnf עם 2 משתנים בכל פסוקית. יש בה n משתנים.

פלט: כן אם היא ספיקה ולא אם היא לא ספיקה.

הרעיון:

נבחר השמה מקרית למשתנים

נבצע פעמים:

2.1 אם היא מספקת נחזיר כן ועצור

2.2 אם היא לא מספקת אז יש איזשהו פסוקית שהיא לא מספקת עבורה, נסתכל עליה.

אם נחליף את הערכים לפסוקית שהיא כן מספקת בצורה אקראית ונמשיך.

נתייאש ונחזיר לא מספקת.

טענות לגבי האלגוריתם:

* הוא רץ בזמן פולינומי.
* אם הנוסחה לא ספיקה אז האלגוריתם יגיד שהיא לא ספיקה. ואז הלולאה תיקרא פעמים עד שתגיד לא.
* אם הנוסחה כן ספיקה, אז ההסתברות למצוא השמה מספקת היא דיי טובה ( רדוקציה על ידי המקרה הקודם- הילוך מקרי על גרפים).

הוכחה:

מניחים שהנוסחה ספיקה (יש השמה שמספקת אותה ).

אם ההשמה שלנו לא מספקת זה אומר שהיא לא ונקרא לה *.*

*נגדיר את המרחק בין ל* להיות:

כל פעם שמעדכנים את מה קורה ל?

אם נוריד את הגודל ב1 🡨 אנחנו נתקרב ל בהסתברות של חצי

אם נעלה את הגודל ב1 🡨 אנחנו נתרחק מ בהסתברות של חצי

וכמו בגרף אנחנו הולכים ימינה ושמאלה בהסתברויות שקטנות וגדולות מחצי.



וברגע שהdistance הוא 0 כלומר נדע שהגענו להשמה מספקת.

*למה זה לא עובד ב3sat ? זו בעיה קשה שאי אפשר לפתור אותה עם אקראיות. יכול להיות שמתקרבים בהסתברות שליש ומתרחקים ב2 שליש. כדי שנעשה n צעדים של התקרבות, ההסתברות היא אקספוננציאלי קטנה כי יש סיכוי יותר שנתרחק מאשר שנתקרב.*

**למת הבידוד:**

לכל שחקן נותנים ציון מצטבר שזה מספר בין 0 ל1000. יש 100 שחקנים ובקבוצה יש 11 אנשים. נסכום את הציונים של השחקנים בקבוצה ולמי שיש מספר מקסימלי של נקודות מנצח.

נניח שנארגן את הטורניר ונקבל פסים

מספר הקבוצות האפשריות הוא

מצד שני אולי יהיו יותר מדי פסים שיהיה להם את הציון המקסימלי.

הציון המינימלי 0 והציון מקסימלי 11,000. יש עיקרון שובך היונים, יותר משתתפים מתוצאות אפשריות ויש התנגשות.

אולי מספר הזוכים מתפלג אחיד ויהיו מספר זוכים בפרס.

למת הבידוד אומרת דבר כזה:

אם ניתן את הסיכוי לזכייה בצורה אחידה, בהסתברות גבוהה יהיה זוכה יחיד לפרס הראשון.

ואז נוכיח שהמקסימלי הוא יחיד בהסתברות גבוהה.

בצורה מתמטית:

כאשר יש לנו איברים ויש לנו פו׳ שנותנת לכל אחד מהאיברים משקל

האיברים היו השחקנים n=100 , משקל היה הציון שלהם =N1-1000.

קבוצת שחקנים זה בעצם סכום המשקולות.

אז יש לנו קבוצה כאשר זה קבוצה שמכילה 11 שחקנים.

והמשקלים של הקבוצה הוא:

למת הבידוד עצמה אומרת ניקח משפחה של תתי קבוצות של

לכל איבר בקבוצה נבחר את המשקל שלו בצורה אקראית, אחידה ובלתי תלויה בכל המשקלות האחרות.

אז הלמה אומרת שבהסתברות של לפחות יש ב קבוצה יחידה שיש לה את המשקל המינימלי. יש הרבה קבוצות שיקבלו משקליפ אבל המינימלית תהיה יחידה בהסתברות גבוהה.

הערות:

* שום דבר פה לא תלוי במשפחה ( כמה גדולה, קטנה, איך נראית תתי קבוצות שלה..) רק תלוי בn ובN.
* כמובן ש
* אפשר לנסח אותו דבר עם מקסימום

הוכחה של הלמה:

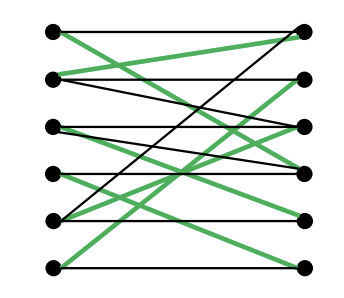
לכל איבר נגדיר פרמטר

-ניקח את כל הקבוצות במשפחה F שx לא שייך אליהן וביניהם ניקח את הקבוצה של המשקל המינימלי, פחות המינימום של כל הקבוצות שx כן נמצא אם מורידים אם x.

אם יש לנו שני סטים A,B שתי קבוצות עם משקל מינימלי, אז עבור מתקיים:

עכשיו שום דבר לא תלוי פה ב *וזה נבחר בהתפלגות יוניפורמי מתוך וההסתברות לכל x היא לכל היותר . ולכן הסתברות שקיים x שמקיים היא לכל היותר .*

איך למת הבידוד עוזרת לנו למצוא שידוך מושלם



ראינו אלגוריתם אקראי שמכריע אם יש שידוך מושלם שמשתמש בבעיית הפולינומים (חישבנו דטרמיננטה של מטריצת שכנויות ואם הוא היה שונה מ 0 יש שידוך, ואם 0 יש שידוך)

הפעם לא רק נכריע את השידוך אלא גם נמצא את השידוך.

הרעיון:

יש לנו קשתות ולכל קשת ניתן משקל אקראי ונבחר אותו באופן אקראי מהקבוצה.

\*10 כי זה צריך להיות משהו שגדול מ

למת הבידוד אומרת שבהסתברות לפחות של 90 אחוזים יש שידוך מושלם יחיד עם משקל מינימלי.

נסתכל על מטריצת השכנויות הדו צדדית. בתא ה *נשים את הקשת ו0 אם אין קשת.*

*קודם אמרנו שנשים מספרים אקראיים למשתנים, אז עכשיו נחליף את בערך כאשר זה המשקל.*

*ועכשיו נחשב את הדטרמיננטה של המטריצה. אם אין שידוך מושלם אז הדטרמיננטה הוא 0. אם יש שידוך יחיד ממשקל מינימלי אז הדטרמיננטה הוא משקל ששונה מ0.*

המשקלים אקראיים וזה אלגוריתם אקראי ולא דטרמיניסטי, במקום בעיית הכרעה זו בעיית חיפוש.

אם בחרנו משקולות טובות, אז הדטרמיננטה תהיה לא אפס.

עכשיו כשנחשב את הדטרמיננטה החישוב הוא :

כאשר אם ננתח זה יוצא +- וכל האיברים האחרים יהיו חזקות גבוהות יותר של 2(או 0).

האלגוריתם:

לכל קשת e :

* + - 1. להוריד את המשקל ב1
      2. מחשבים שוב את הדטרמיננטה
      3. בודקים האם המשקל המינימלי קטן.   
         אם כן הקשת משתתפת במינימום משקל לשידוך מושלם.

היתרון : אפשר למקבל לכל הקשתות, אין פה שום דבר שתלוי אחד בשני. נדבר על זה בהמשך הקורס. אבל בעיקרון סופרים זמן מקבילי. ואת הדטרמיננטה נחשב בזמן של .

תוצאה שמוכיחה שמשהו NP קשה בעזרת למת הבידוד:

כמו למצוא את הקליק המקסימלי.

קליקה – קבוצה של קודקודים שכל זוג בהם מחושב בקשת. למצוא מה הקליק הכי גדול היא בעיה קשה.

אם הקליק הכי גדול הוא יחיד יש הרבה אילוצים שזה כבר נהיה קל.

נראה שבמקרה של שידוך מושלם – אם זה היה יחיד זה הקל עלינו למצוא אותו.

במקרה של קליק נראה שגם במקרה שהוא יחיד- יהיה קשה למצוא אותו.

נוכיח שזה קשה עם רדוקציה אקראית – רדוקציה שהחישוב שלה יתבצע עם מספרים אקראיים – לכן היא לא תמיד תעבוד אבל בהסתברות גבוהה היא תעבוד.

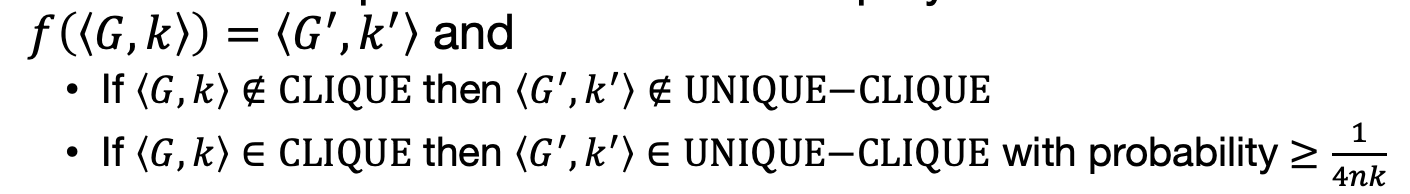
תזכורת

* **מחלקת NP-** מחלקה של הבעיות שאם נותנים לנו את הפתרון אפשר לוודא אותו בזמן פולינומי בצורה יעילה. *למשל האם קיים קליק בגודל K זו בעיה קשה. אבל אם נותנים לנו את הגרף בגודל K ושואלים אם הוא קליק זו בעיה NP.*
* **מחלקה NP complete**- היא גם np וגם יש רדוקציה מכל בעיה בnp לשפה הזו. היא הכי קשה בnp וחולקת את הסטטוס הזה עם עוד הרבה שפות אחרות.
* **רדוקציה מA לB**- יש פונקציה שאפשר לחשב בזמן פולינומי ומקיימת ש

התכנית היא להראות שאם היה אפשר לפתור את הבעיה של unique clique אז היה אפשר לפתור את clique.

פונקציית unique clique

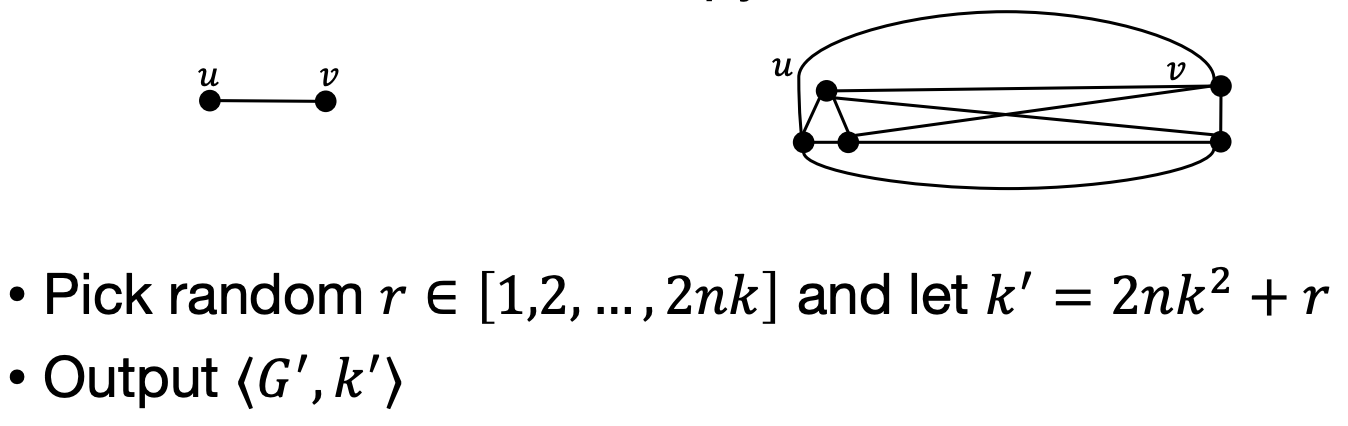
יש אלגוריתם אקראי שמקבל זוג ומוציא גרף אחר שמקיימים את הדברים הבאים:

  
נתחיל מלבחור לכל קודקוד בגרף משקל, מספר אקראי בין (המשקל המקסימלי צריך להיות גדול יותר ממספר האיברים).

אנחנו יכולים להגיד שכמה קליקים יהיו מספר מינימלי בהסתברות גבוה? קליק אחד ממשקל מינימלי יחיד. זו ההגדרה של למת הבידוד.

הקליק המקסימלי הוא יחיד.

דוגמה:



הr הוא הניחוש שלי למה הוא המשקל המינימלי.

אם אין קליק בגודל K, זה הגדול של כל קליק בG הוא לכל הפחות   
מספר שקטן מ ולכן אין קליק שגדול מזה.

אלגוריתמים אלגבריים:

מודל חישוב אלגברי  
 נסתכל על אלגוריתמים שיכולים לבצע חישוב אלגברי וכל פעולה כזו יכולה לבצע יחידה אחת ללא קשר לקלט.

נמדוד את מספר פעולות האריתמטיות שנבצע.

כפל מטריצות  
קלט: שתי מטריצות. מטריצה בגודל ומטריצה בגודל

פלט: מטריצה ששווה ל

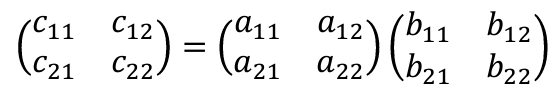
אלגוריתם קלאסי:   
עבורו נבצע כי נשתמש ב *פעולות כפל וב פעולות חיבור. ויש תאים במטריצה וכל אחד מהם אנחנו עושים בנפרד. ולכן מספר הפעולות הוא .*

*צריך להזכיר שחיבור מטריצות מתבצע בלפחות וזה במקרה שנעבור איבר איבר.*

*ואז נוצר אלגוריתם חדש:*

*שטרסן אלגוריתם*

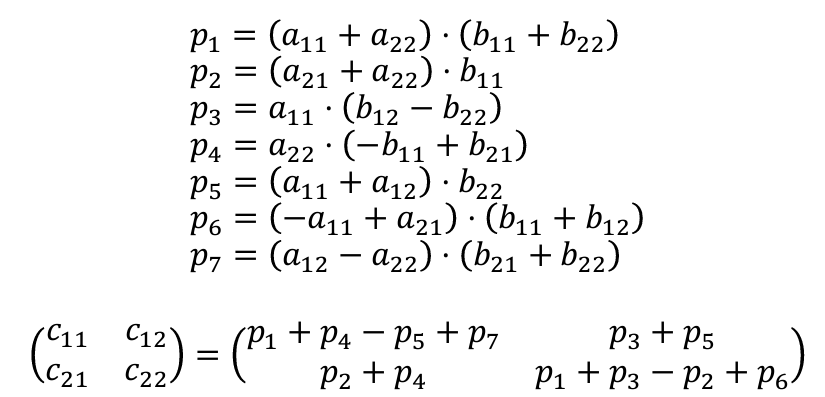
נניח ויש לנו מטריצת



באלגוריתם הקודם היינו צריכים 8 פעולות ושטרסן מצא דרך להשתמש ב7 פעולות כפל. אבל הוא משלם ביותר חיבורים וחיסורים(18).

אז האלגוריתם הולך ככה:

נחשב 7 כפלים:



איבר מהמטריצה A אל מול איבר מהמטריצה B

התועלת האמיתית שלו היא העובדה שניתן להשתמש בו רקורסיבית.

יש לנו מטריצה גדולה אז נחלק לארבעה בלוקים.

והרעיון להשתמש במטריצה של טרסן אבל כל פעם שצריך לכפול 2 בלוקים נעשה את זה בצורה רקורסיבית.

חיבור זו פעולה זולה אבל כפל זה מה שנשלם עליו ולכן העובדה שחסכנו בכפל היא הפקטור המשמעותי.

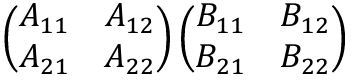
**משפט**: ניתן לבצע כפל מטריצות בגודל בזמן

שזה טוב יותר מ וזה שיפור

הוכחה:

נניח ש n הוא חזקה של 2 ( אחרת אפשר להוסיף שורות אפסים עד שזה יסתדר)

נחלק את המטריצה שלנו לבלוקים (במקום מטריצה זה בלוק בגודל



שטרסן מראה לנו איך להכפיל את המטריצות האלה עם 7 פעולות כפל ו18 חיבור. אבל הפעם זה לא מספרים אלא מטריצות.   
יהיו לנו 7 כפלים עבור כל מטריצה שבגודל וזה בגודל

ואז נקבל בצורה רקורסיבית:

כאשר זה מספר הפעולות ששטרסן מבצע

ולבסוף נקבל

מה בכל זאת אפשר לשפר?

הקבוע אומיגה 𝜔 שאנחנו לא יודעים מה הוא. אפשר להשיג רק חסמים.

אז תן ל𝜔 להיות ה כך שנוכל להכפיל מטריצות בזמן

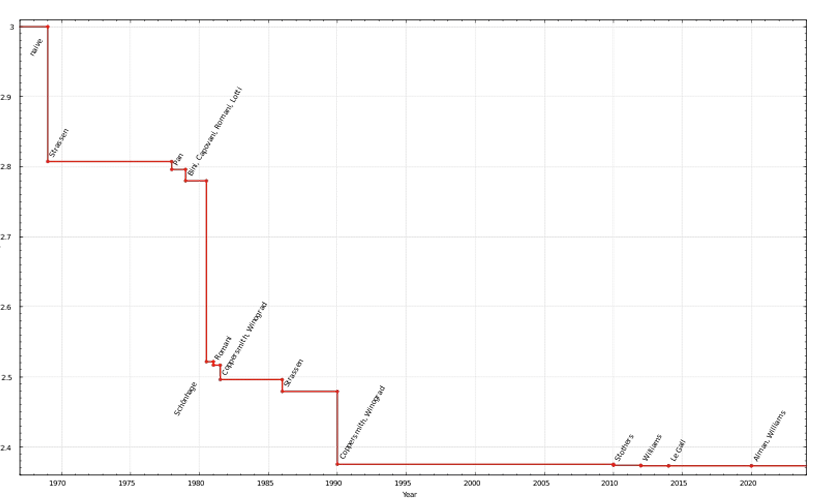
אם נוכל להכפיל מטריצה בגודל אם k מכפלות אז נוכל לטעון שהחסם העליון הוא

ואז מסתדר לנו ש

הרבה אנשים טוענים ש שאומיגה שווה ל2.

ואז הסיבוכיות היא

החסמים שאנחנו יודעים על אומגה לאורך השנים:



צריך לדעת על האלגוריתמים האלה שלאורך השנים פיתחו שיטות לכפול מטריצות בסיבוכיות נמוכה. ותמיד היו שיפורים בספרה השישית לאחר הנקודה. חוץ משטרסן אף אלגוריתם לא תוכנת. מה גם שנתנו שיפורים רק לn גדול מאוד.

נשתמש בבעיה אחרת כדי לפתור עם הכפל מטריצות.

אלגוריתם APSP- מציאת הדרכים הכי קצרות  
קלט: גרף קשיר לא מכוון ופשוט

פלט: את המרחק (האורך הכי קצר) בין שני קודקודים בין קודקודים

אלגוריתם קלאסי: אלגוריתם תכנות דינאמי פלויד-ורשל שמוצא n את המרחק בסיבוכיות של .

בעזרת כפל מטריצות נצליח לפתור את הבעיה בזמן של

איך נקשר בין גרפים למטריצות ? באמצעות מטריצת שכנויות.

בהינתן גרף כאשר והמטריצה A שבגודל כך ש אם כלומר יש קשת, ואם אין קשת אז - 0.

*השאלה היא מה קורה אם מסתכלים על ?*

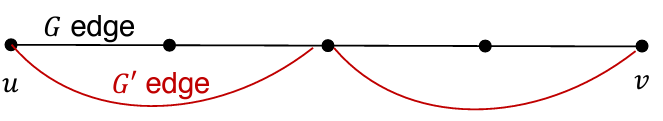
*אם ורק אם יש מסלול באורך 2 או לכל היותר 2.*

*ונבנה מטריצה B שעבורה מתקיים:   
 אם , כלומר או שיש מרחק לפחות 2 ואז הערך הוא 1 ויש קשת*

*או , אז או שאין מרחק כזה ואז הערך הוא 0 ואין קשת*

*ונתן להיות הגרף של המטריצת שכנויות B*

*על הגרפים*



*המחק בין קודקודים v לu בגרף G מסומן כd.*

*אם d הוא זוגי: המרחק בגרף הוא כי כל שני צעדים אפשר להחליף בצעד 1*

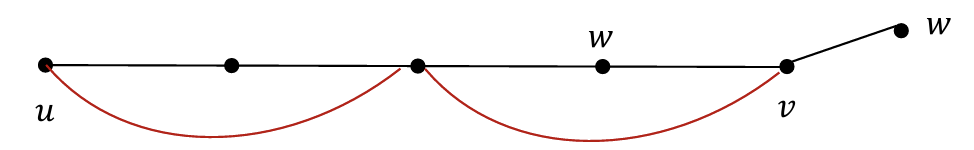
*אם d הוא אי זוגי: המרחק בגרף הוא*

*דרך אחרת לכתוב את זה אז אפשר להגיד כאשר זה המרחק ב*

*צריך להחליט אם אנחנו באחד משני המקרים :*

*במקרה שהמרחק בG זוגי:*

*נסתכל על שכנים של v ונטען שבמקרה הזה לכל שכן w של v מתקיים שהמרחק מu לw גדול או שווה מהמרחק בין u לv . בשפה אלגברית:*

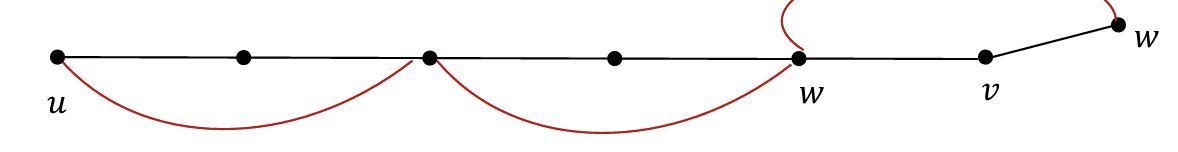
**

*במקרה הזה, כל השכנים של v הם רחוקים יותר.*

*או במילים אחרות אם נסכום את כל השכנים (את המרחקים בין u לw) זה לפחות גודל הקבוצה בין u לv.*

*אז הסכום גדול מזה.*

*במקרה שהמרחק בG אי זוגי :*

**

*לכל שכן w של v, המרחק בין uלw קטן מהמרחק בין u לv.*

*סיכום של המרחקים:*

*כאשר צריך לחשב את המרחק נצטרך לחשב את סכום השכנים שלו של המרחקים מu לw*

*מהנחת האינדוקציה יש לנו מטריצה T שעבורה מתקיים*

*ואז מתקיים*

*הסבר: נכפול את מטריצה A במטריצה T המקורית של G ונראה מה נקבל בתא u,v – נקבל את סכום הw עדv שזה באמת 1 אם v שכן של u ו0 אחרת.*

*שאלה: אז מתי האינדוקציה נגמרת?*

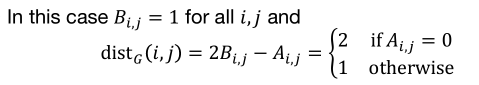
*תשובה: שימו לב שאנחנו רק מוסיפים קשתות , תמיד נשאר עם nקודקודים. בסופו של דבר יהיה עם כל הקשתות.  
אז נעצור כאשר יהיה הגרף המלא והמרחקים לכל היותר 2.*

*במקרה זה קל לחשב את המרחק מi לj*

*אם אין קשת המרחק הוא לכל היותר 2.*

*אם הוא שווה 1 אז המרחק הוא 1.*

*בצורה אלגברית :*

**

*הערה- זו מטריצת המרחקים לא השכנויות אז המרחק כן יכול להיות 2.*

*אלגוריתם סופי שנקרא APD*

*קלט: מטריצת A שכנויות של גרף פשוט, קשיר ולא מכוון.*

*מהלך:*

*נחשב את*

*נגדיר מטריצה B שהיא בעצם   
עבורה מתקיים אם אז   
אחרת ואז B היא מטריצת שכנויות של*

*אם עבור כל , החזר וזה בעצם התנאי הבסיסי לאינדוקציה.*

*בצורה רקורסיבית נחשב את מטריצה T שהיא מטריצת המרחקים של B באופן הבא:   
T זה בעצם המרחק בין i ל j בגרף*

*חשב את המטריצה החדשה שמאפשרת לנו להבחין בין מקרה זוגי לאי זוגי.*

*ועכשיו בהינתן x אפשר לחשב את המרחק. אם*

*החזר את D*

*זמן ריצה:*

*ישנם שלבים  
בכל שלב יש :   
2 שלבים של הכפלת מטריצות ( לחישוב וגם ) שלוקחים*

*כל שאר השלבים לוקחים*

*ואז בסה"כ:*

דברים שאפשר לעשות עם כפל מטריצות:

1. DIRECT SUM

*קלט: מטריצה A בגודל ווקטור עמודה v באורך n*

*פלט: נרצה לחשב את*

*האלגוריתם הנאיבי ייתן לנו את הפתרון בגודל כי יש פעולות עבור n פעמים.*

*ויש דרך טובה יותר על ידי מה שלמדנו.*

*נחפש את מטריצה B כמו שעשינו באלגוריתם של כפל מטריצות..*

*לשים לב שיש תרגיל כזה בשיעורי הבית עם מטריצת טופוליץ\**

1. וידוא כפל מטריצות  
   לא יודעים אם אנחנו יכולים להכפיל מטריצה בזמן . אבל אנחנו יכולים בהינתן שלוש מטריצות האם הן מכפלה אחת של השנייה בזמן של .

*קלט: מטריצות A,B,C בגודל*

*פלט: כן האם , אחרת לא  
רעיון:*

*אם אז עבור כל v כל זה* בזמן .

*אם אז ואז v הוא הקרנל. במטריצה לא אפסית*

*אלגוריתם :*

1. *נבחר באופן אקראי את v כך שונחשב את*
2. *אם החזר כן, אחרת החזר לא.*

*ברור שהאלגוריתם מבצע*  פעולות, ואם הוא תמיד יחזיר כן.

נשאר להוכיח מה קורה במקרה שבו , האלגוריתם מחזיר :לא , עם סיכוי של לפחות

**משפט**: נרצה להוכיח שכאשר המטריצה היא לא 0,

באמצעות למה נראה שאם D היא לא 0, אז עבור לפחות חצי מהנקודות ב

*הוכחה:*

*נניח ש i היא שורה של לא אפסים בD אז ניקח שורה במטריצה- הקורדינטה הi של Dv -   
 זהו פולינום לא אפסי במשתנים .   
 למה הוא לא אפס? כי יש איזשהו שהוא לא אפס זו הייתה ההנחה.*

*על ידי הלמה של שוורץ זיפל עבור d=1 ו s זו הקבוצה שממנה נבחר את המשתנים*

*נטען ש:*

*\*זו בעיה קצת מלאכותית אבל כדי שנזכור שיש אלגוריתם אקראי והוא קיים*

חישוב דטרמיננטה  
קלט: מטריצה A בגודל

פלט: חישוב דטרמיננטה

סכום על כל הפרמוטציות על n איברים וניקח את האלכסון המוכלל שמתאים לפרמוטציה.

האלגוריתם הנאיבי מחשב את הדטרמיננטה בזמן של

דרך אחת זה לדרג את המטריצה- לעשות פעולות אלמנטריות על השורות. ובאמצעות סדרת פעולות כאלה נעביר את המטריצה למדורגת והיא משולשת תחתונה או עליונה. והסיבוכיות של הדבר הזה הוא

עבור מטריצה מדורגת זה פשוט מכפלת האלכסון.

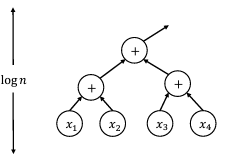
הבעיה עם התהליך הזה זה שהוא סדרתי. אי אפשר למקבל.   
נתעניין באלגוריתמים מקביליים : יש לנו מספר פולינומי של מעבדים שיכולים לפעול במקביל אבל אם הקלט שלהם תלוי בשלבים הקודמים, השלבים הקודם הושלם.

*דוגמה לחישוב מקבילי:*

*בהינתן חישוב סכום*

*נרצה לחשב :* :

אפשר לחשב באופן מקבילי עבור n אלמנטים בזמן של



נסתכל על חזקה גדולה של A

**טענה**: אפשר לחשב את זה בזמן של

דרך שאפשר לעשות את זה:

 וזה נקרא חישוב מרובע

באופן דומה אפשר לחשב את כל החזקות של 

נראה את הדטרמיננטה שגם מחשבת בזמן מקבילי ומשתמש במספר פולינומי של מעבדים.

ערכים עצמיים:

תזכורת – הקשר בין דטרמיננטה לערכים עצמיים הוא שמכפלת הערכים העצמים שווה לדטרמיננטה

אם הערכים העצמיים הם הדטרמיננטה בדיוק שווה למכפלה שלהם.

הוכחה:

משתמע מיידית מהפולינום במשתנה x הפולינום מעלתו היא לפחות x

ואז

המקדם של הוא 1.

המקדם של הוא

Tr זה סכום האיברים באלכסון

המקדם ההפוך הוא

וזה מינוס 1 אם n הוא זוגי ואחרת 1

אם היינו רוצים לחשב את סכום הערכים העצמיים:

לסכום את האיברים באלכסון. בזמן

המקדם של הוא

סכום של כל תתי הקבוצות ומכפילים את הלמדות ששייכות לקבוצה הזו.

1 אם n זוגי ומינוס 1 אם n אי-זוגי.

הראינו אלגוריתם שמחשב את המקדמים של פולינומים בזמן מקבילי

*הפולינום הסימטרי האלמנטרי הk:*

*סימטריה- אם נחליף מקדמים זה יהיה אותו פולינום.*

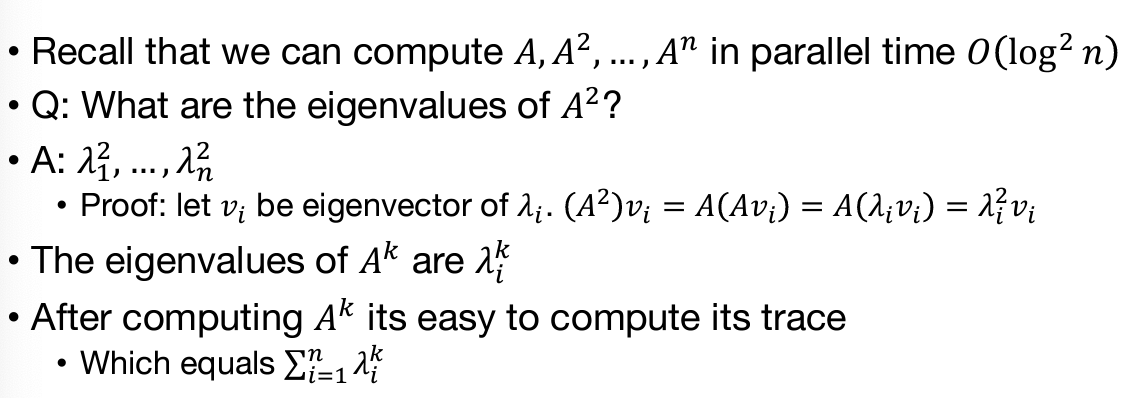
*יהיה לנו אלגוריתם שיחשב את כל הפולינומים הסימטריים האלמנטריים בפרט שמסתכלים על אחד מהם וזה בדיוק הדטרמיננטה.*

*את המקדם של קל לחשב הוא תמיד יהיה 1.*

*המקדם של הוא מינוס טרייס הוא גם קל לחשב.*

*לפני שנחשב אותם, נחשב משהו אחר, פונקציות סימטריות אחרות ב ומהם נחשב משהו אחר.*

*אנחנו יודעים לחשב את כל החזקות של A.*

**

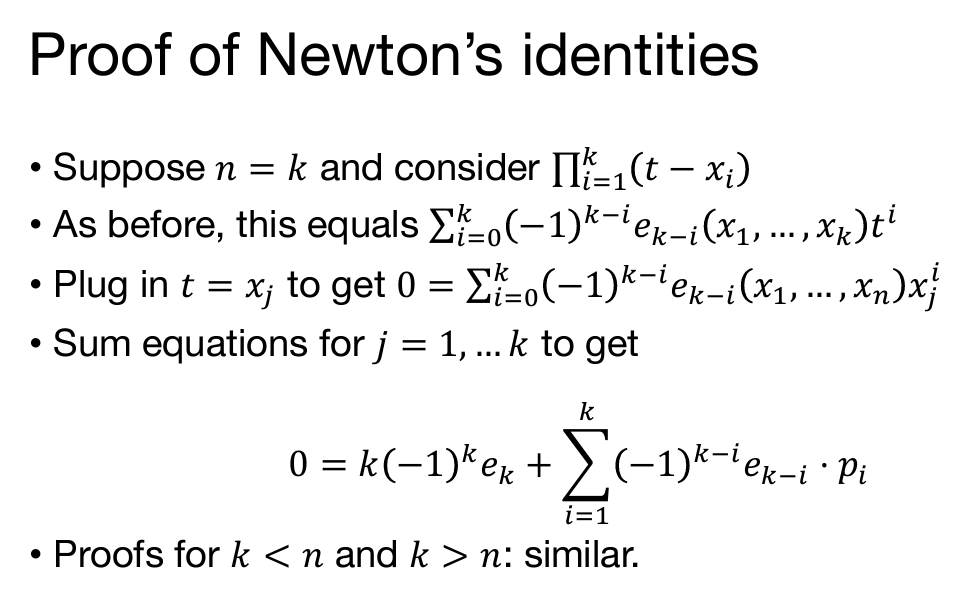
*הדבר הזה נקרא*

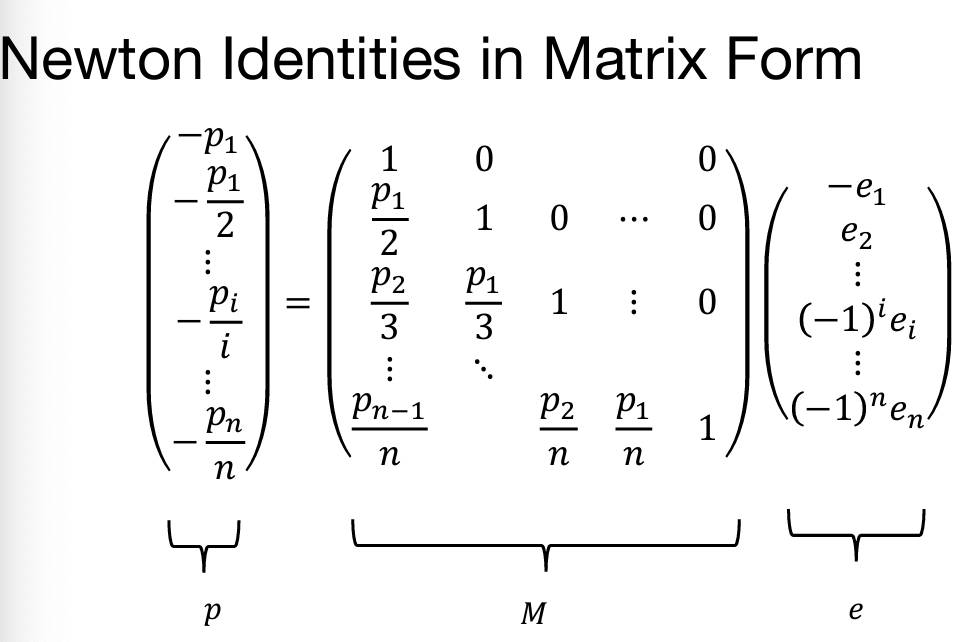
*בעצם נוכל לחשב את עבור בזמן מקבילי של*

*הפולינום p הוא סכום החזקה הפולינומית. מסתבר שהם קשורים לפולינומים e באמצעות זהויות ניוטון.*

***זהויות ניוטון***

**

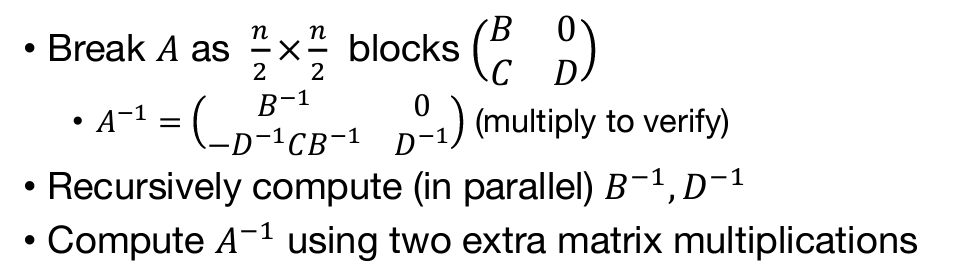
*הוכחה של זהויות ניוטון:  
  
דוגמה*

**

***המטרה: להפוך מטריצה משולשת תחתונה***

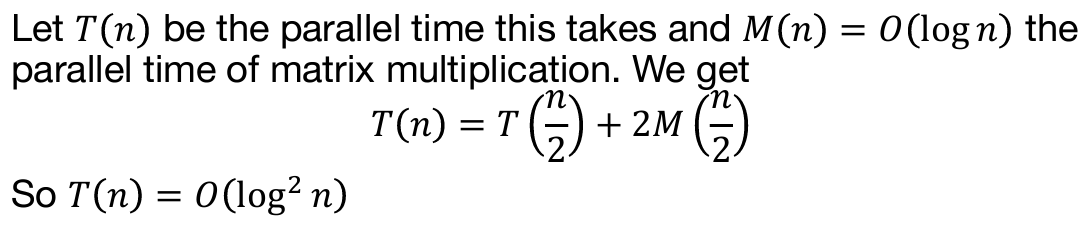
***למה****: אפשר להפוך מטריצה משולשת תחתונה בזמן מקבילי*

*\*זה לא יוסיף למה שכבר שילמנו.*

*איך עושים את זה ? בעזרת רקורסיה, נסתכל על A היא מטריצה משולשת תחתונה  
*

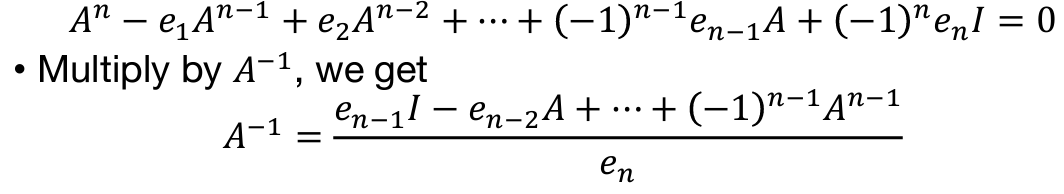
*אנחנו יודעים שגם B  וD הן גם מטריצות משולשות תחתונה. זה עוזר לנו כי ההופכית של A זה בדיוק המטריצה הזו שמצורפת שם.*

*כדי להפוך מטריצה הזמן הוא:*

**

*איך הופכים מטריצה כללית שהיא לא משולשת באותו זמן של לוג בריבוע?*

*הרעיון הוא להשתמש במשפט שלמדנו באלגברה לינארית- קייל המילטון*

**

*אם נעשה את זה נקבל את מטריצת האפס, מטריצה n\*n שהיא אפס.*

*שe הוא מספר, איבר בשדה וx הוא משתנה.*

*אם אפשר להפוך מטריצה אז אפשר גם לפתור מערכת משוואות.*

*יש אלגוריתמים שהם שימושיים:*

פורייה FFT  
איך מכפילים מספרים גדולים שלא נכנסים במשתנה?  
  
צריך אלגוריתם שיעזור לנו. גם בעבור חיבור.

עבור n ספרות חיבור הוא בגודל

כפל של ספרה בספרה זה בגודל

יש אלגוריתם יותר טובים שמכפילים אבל כרגע נלמד על FFT בגודל .

קלט: שני פולינומים ממעלה לכל היותר של n

פלט: חישוב הפולינומים ממעלה של 2n

*בכמה פעולות אנחנו יודעים לעשות את זה?ֿ*

*מה קורה היום:*

*בואו נפרק את הפולינומים וגם.*

*הפלט צריך להיות מחושב ככה של החיבור :*

*הפלט צריך להיות מחושב ככה של הכפל :*

*המעלה צריכה להיות קטנה מ n . כי המעלה יכולה להיות לכל היותר n-1*

*השאלה האם אפשר לעשות יותר טוב? התשובה היא כן*

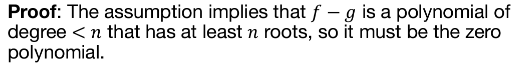
*האלגוריתם FFT שיעזור לנו לבצע כפל בצורה מהירה.*

*בזמן האחרון דיברנו על ייצוג פולינומים על ידי שערוך של הפולינום בn נקודות.*

*הבחנה: ניקח n-1 איברים שונים- מספרים מרוכבים  
 , אם יש שני פולינומים שהשערוך שלהם במספרים האלה אותו דבר אז הם אותו פולינום.*

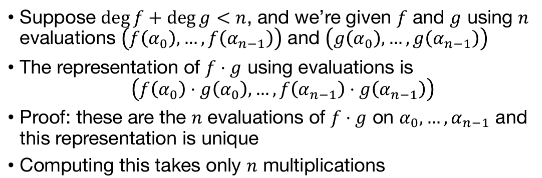
*כלומר אם עבור כל אז נגיד ש*

*הוכחה*

**

*הפונקציה ששולחת פולינום לוקטור באורך n שמכיל את השערוכים שלו לn נקודות שונים, זו פונקציה חח״ע. כלומר זה אותו פולינום = f נקבע ביחידות.*

*טענה: בייצוג הזה דווקא מאוד קל להכפיל כי הכפל של שני פולינומים הוא פשוט כפל של קורדינטה כפול קורדינטה.*

**

*יש דברים שקשה לעשות עם הייצוג הזה. אם נותנים את המקדמים פשוט אפשר להציב..*

*בייצוג שערוכים איך עושים את זה?*

*אלגוריתם כפל המהיר:*

1. *מעבר מייצוג מקדם לייצוג מעריכי*
2. *השתמש בn מכפלות כדי לחשב את f\*g בייצוג מעריכי.*
3. *אינטרפולציה חזרה לייצוג מקדמי של f\*g*

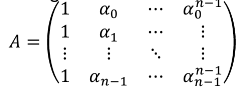
*אנחנו צריכים את שלושת השלבים שיקרו בפחות מ .*

*הסוד הוא לעשות את זה בצורה מהירה- לבחור את הנקודות האלה   
 בצורה חכמה עם תכונות שיעזרו לעשות את זה בצורה מהירה.*

*הדבר הראשון שנשים לב אליו הוא ששערוך זו פעולה לינארית על מקדמי הפולינום.*

*וקטור השערוך של f+g זה וקטור השערוך של f ועוד וקטור השערוך של g. ככה גם על כפל.*

*פונקציה לינארית, לכל פעולה לינארית יש מטריצה שמייצגת אותה שנקראת* ***ואנדרמונטה****:*

**

*צריך לבחור את הa כך שנוכל לחשב את ההפיכות שלה בקלות.*

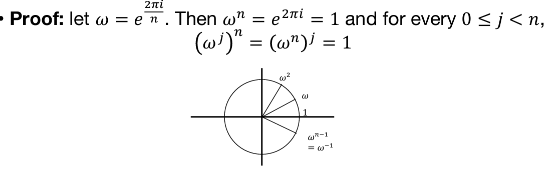
*הפתרון : נבחר את a כך שהמטריצה תהיה דומה להופכית ושתיהן יקיימו את התכונות שאנחנו רוצים.*

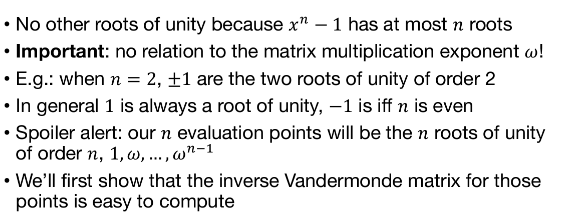
***בואו נכיר את שורשי היחידה:***

*שורש מרוכב מסדר n הוא כאשר*

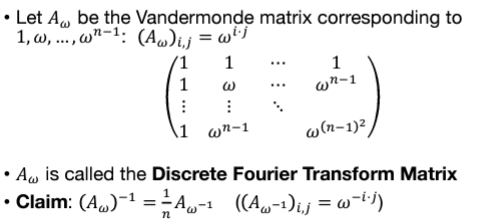
*במספרים המרוכבים, לכל n יש בדיוק n שורשי יחידה שכולם שונים.*

*הוכחה:*

**

**

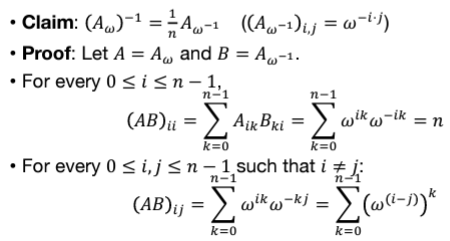
***עכשיו נציב את את זה ונקבל מטריצת DFT***

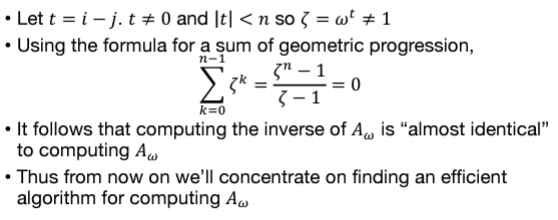
**

***הטענה****: ההפכי שלה זה בדיוק כמו לקחת את A ובכל מקום שכתוב אומגה- יהיה אומיגה מינוס אחד. וזה יהיה ההפכי.*

*יש גם נרמול קטן של חלוקה בn.*

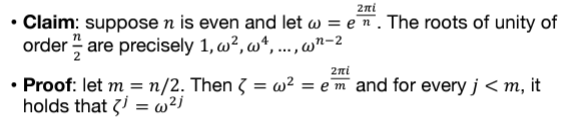
*הוכחה של הטענה*

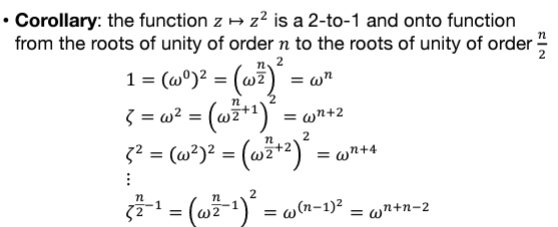
**

**

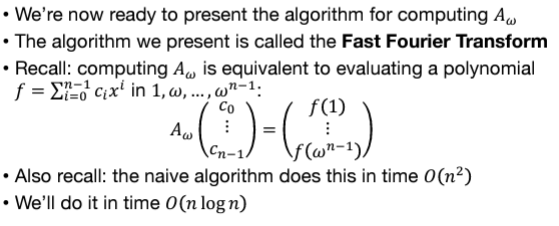
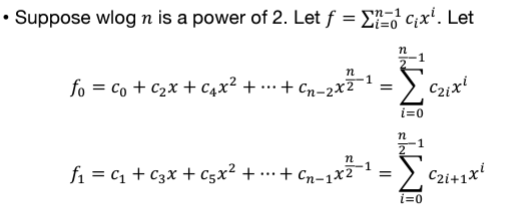
*צריך עוד תכונה אחת לשורשי היחידה שתעזור לנו לחשב :*

*נטען ששורשי היחידה מסדר n/2 הן בדיוק החזקות הזוגיות של אומגה*

**

**

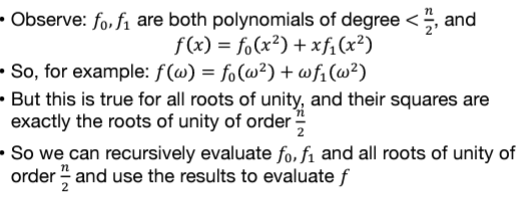
*עכשיו נראה איך לחשב את הטרנספורמציה הלינארית של DFT*

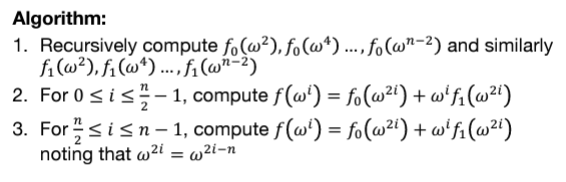
*  
*

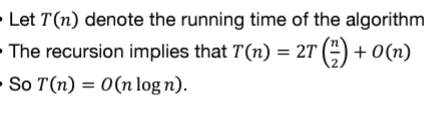
*מתאימים לחזקות זוגיות ולאי זוגיות.*

*רוצים להוריד את המעלה של הפולינומים לf/2*

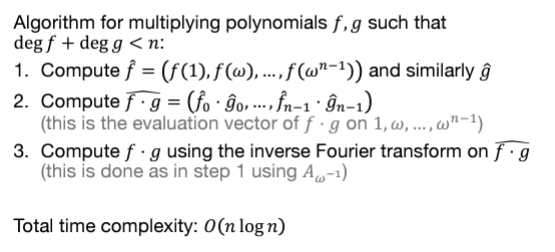
*המשך שלא היה לי כוח לתמלל:*

**

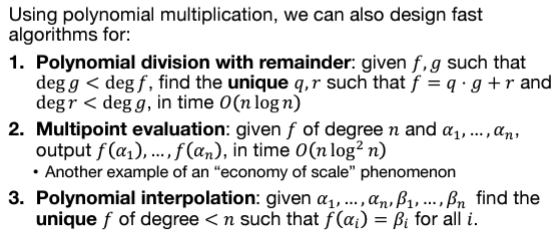
**

**

***ובחזרה למכפלת פולינומים:***

**

*טריקים נוספים לפולינומים:*

**

תכנון לינארי:

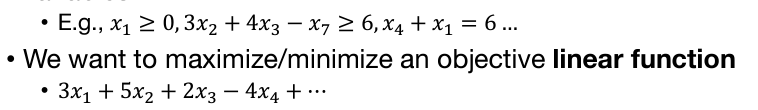
הפעם ניקח בעיה אחת ונחפור בה יותר, נלמד כלי חזק שאפשר לפתור איתן הרבה בעיות.

מה זה תכנון לינארי?

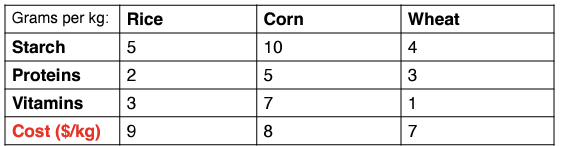
למקסם או לנרמל פונקציה לינארית תחת אילוצים לינארים.

נראה אלגוריתם שזמן הריצה שלו אקספוננציאלי שעובד מאוד טוב, ונראה גם אלגוריתמים אחרים שמבחינה תיאורתית אפשר לתת חסם טוב יותר, וגם אפשר להראות אופטימיזציות לא לינאריות וזה יהיה שימושי בהמשך הקורס.

אופטימיזציה לינארי  
 נסתכל על משתנים בהינתן אילוץ לינארי



*דוגמה- בעיית הדיאטה*



יש דרישות לתזונה היומית

נרצה 16 גרם עמילן, 20 פרוטאין ו3 ויטמינים.

נרצה **למנם** את העלות של התפריט היומי שלנו.

בואו נייצר ממנו בעיה של אופטימיזציה תחת אילוצים לינארי או במילים אחרות תכנון לינארי.

צריך לחשוב מהם המשתנים שלנו.

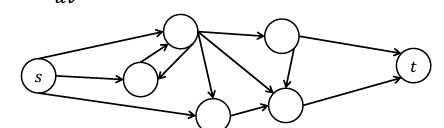
ישנם משתנים ונרצה למנם את העלות שלנו.

דרישות:

באופן דומה נרשום אילוצים לחלבון ולויטמינים.

בנוסף לא יכולים לאכול כמות שלילית ולכן שכל המשתנים האלה חייבים להיות אי שלילים.

*דוגמה נוספת בעיית זרימה בגרפים*



יש גרף מכוון שהוא רשת הזרימה שלנו G ומקור s והולכת לt ואנחנו רוצים להזרים כמה שיותר זרימה ברשת עם אילוצים כלשהם מן הסתם.

2 סוגי אילוצים:

* לכל קשת יש קיבול מספר ממשי גדול מ0 ואי אפשר להעביר בה יותר זרימה ממה שהיא מסוגלת להעביר.
* אילוצי שימור הזרימה- כל קודקוד פרט לs,t – הזרימה שנכנסת חייבת להיות הזרימה שיוצאת.

בואו נראה איך רושמים את הבעיה כבעיית תכנון לינארי:

לכל קשת יש כמה זרימה אנחנו מעבירים על הקשת

נרצה למקסם את סכום הקשתות שבהן עוברות הזרימה תחת האילוצים שבהם הקיבולת של כל קשת

. לכל קשת יש קיבולת כמה היא מסוגלת להעביר וזה האילוץ.

בעיית הזרימה בעצם עבור כל

*פתרון אופטימלי= זרימה אופטימלית.*

*יש כל מיני מניפולציות שאפשר לעשות לאי שוויונות ונדבר עליהם עכשיו :*

*כשיש מערכת משוואות לינאריות , כזו אפשר להציג גם בצורה כזו*

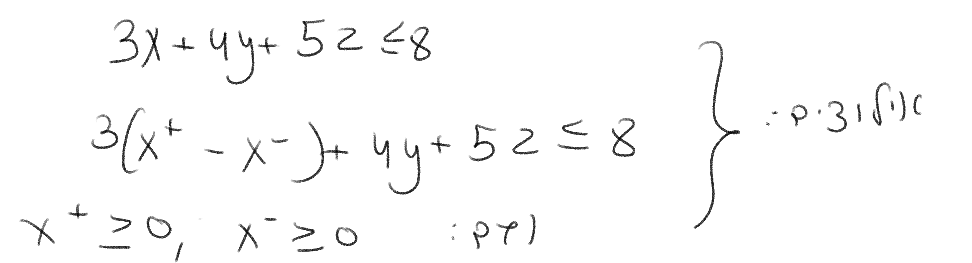
*במקום זה אפשר לקחת את המינימום של אחד הצדדים או המקסימום של אחד הצדדים. ואפשר תמיד לעבור מאחד לשני. אנחנו נסתכל בינתיים על המינימום.*

*אם זה מתקיים אז אפשר להגיד ש וגם וגם הפוך זה מתקיים.*

*נוכל להניח שכל המשתנים יהיו אי שליליים.*

***נוכל להניח שכל התכנון הלינארי שמעניין אותנו יראה ככה- מינימיזציה לפונקציה לינארית והאילוצים***

*דוגמה עם מספרים רק כדי שאני אבין:*

**

***אנחנו צריכים את הדבר הזה***

***כדי שנניח שהמערכת שהנתונה לנו, נוכל לעשות מינימייז לזה להמיר את זה ל***

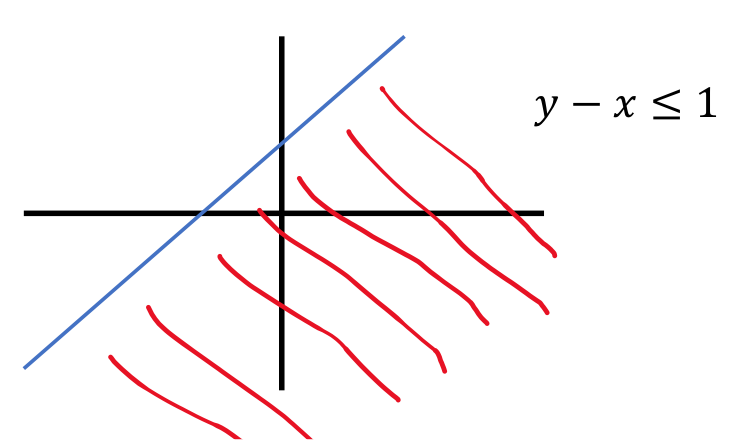
*מוסכמות: וקטורי עמודה כולם, אלא אם זה נרצה וקטורי שורה ואז זה טרנספוז.*

*אפשר להניח שתמיד יש בעיית מינימיזציה.*

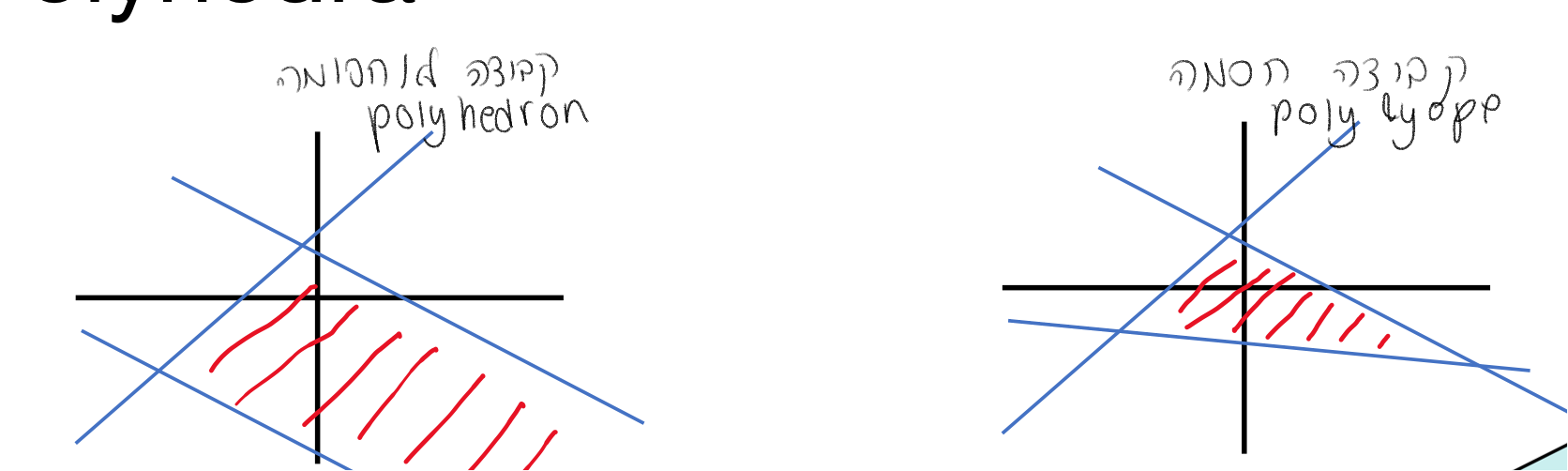
על מישור Hyperplane

*עד עכשיו זה היה אלגברי, אבל לתחום הזה יש צד גיאומטרי ואי אפשר להבין את אחד בלי השני.*

*בדוגמה הבאה זה מערכת משוואות יחידה:*

**

*לקבוצה שמוגדרת על ידי מערכת של אי שוויונות נקראת פיאון Polyhedron*

**

*המוטיבציה לעשות את זה:*

*Polyhedron- קבוצת הנקודות הפיזיביליות*

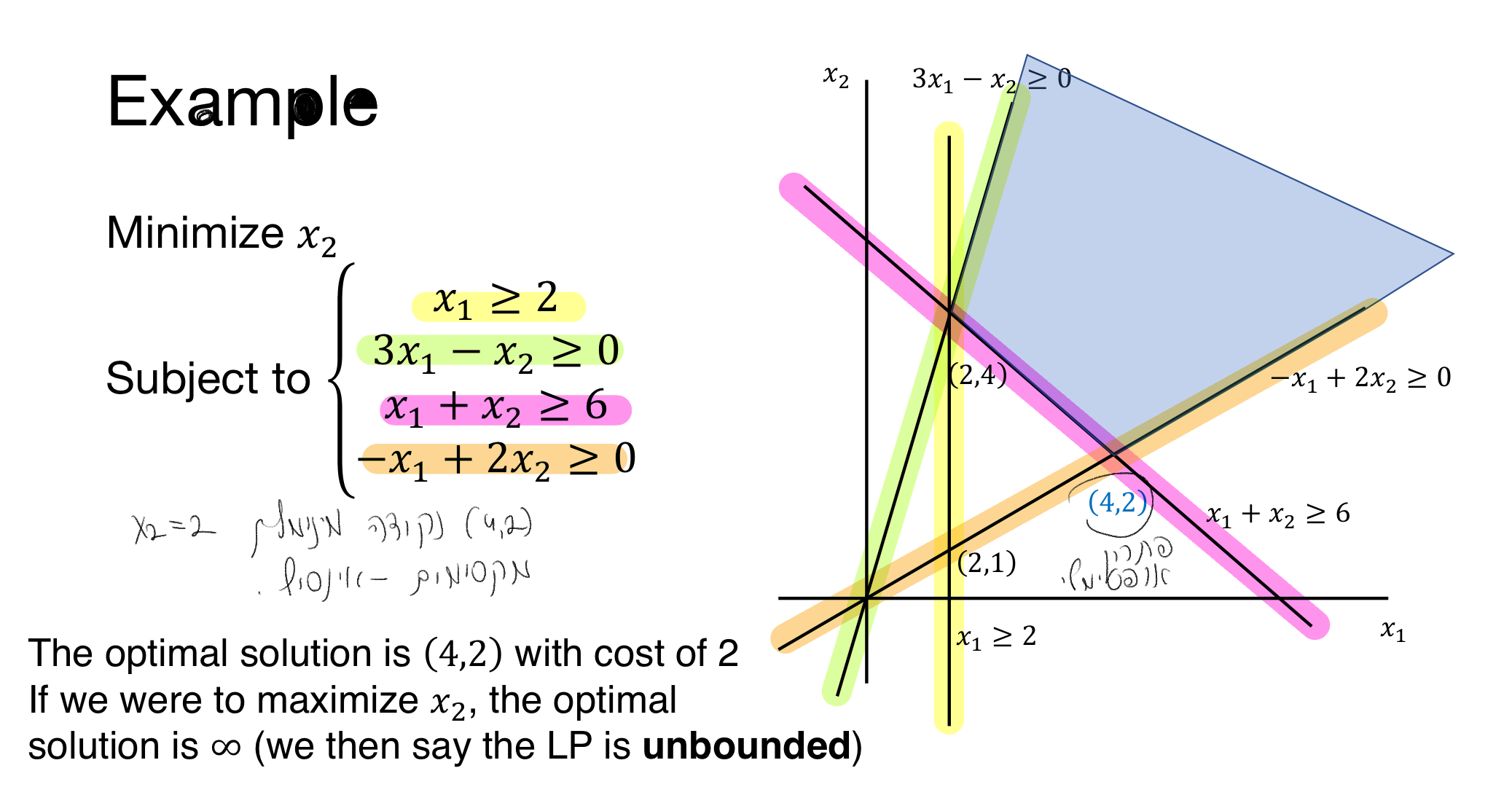
*אם יש וקטור x שמקיים את כל האילוצים של בעיית תכנון לינארי הוא נקרא פתרון פיזיבילי ( אפשרי). כלומר הוא מקיים את האילוצים הוא לא בטוח אופטימלי.*

*אם יש פתרון פיזיבלי אז אומרים שהתכנון הלינארי הוא פיזיבילי.*

*פתרון אופטימלי הוא מינימיזציה של פתרון פיזיבלי.*

דוגמה:

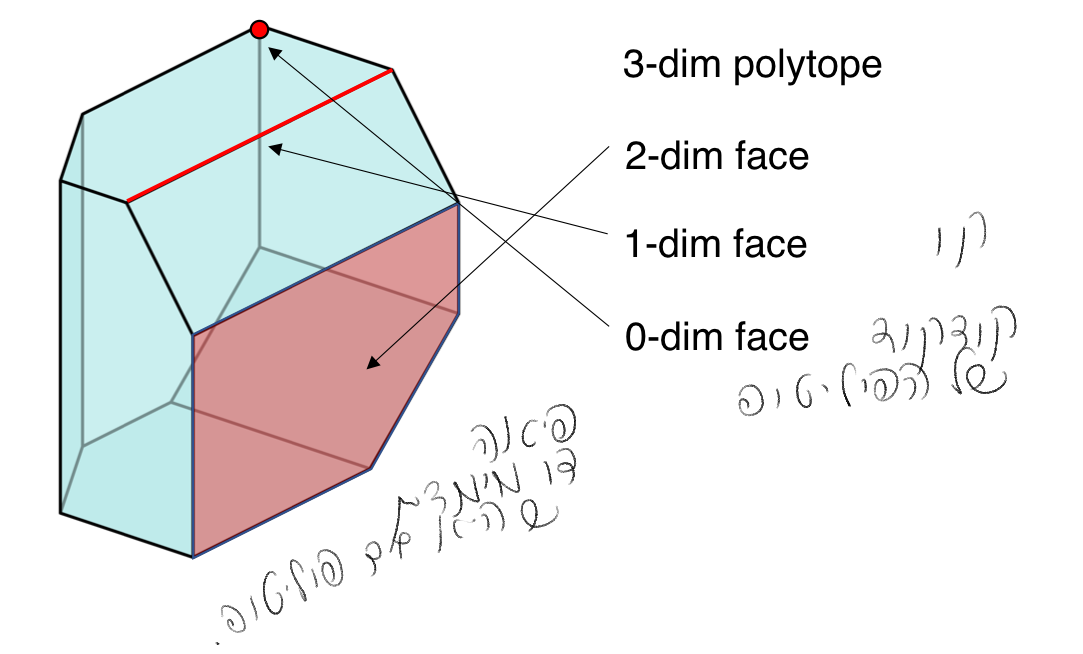
*נניח שיש לנו את הבעיה הבאה נרצה למינימייז את x2 תחת האילוצים הבאים:*

**

*נחפש חיתוכים של אילוצים. ותכף נראה למה.*

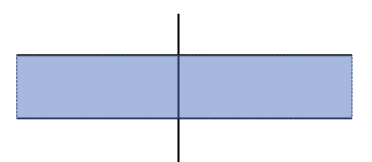
*משתנים בפוליטופס*

***פיאה*** *זה כאשר מחליפים אי שוויונות בשוויונות.*

**

***קודקוד -*** *הוא קודקוד x אם לא קיים כך שמקיים*

*לא לכל פוליהדרון יש קודקוד. דוגמה:*

**

*קבוצת הפתרונות האופטימלים של הפוליטופ היא תמיד פיאה.*

*אבל נוכיח שאם LP הוא בסטנדרט פורם שלנו ויש לה פתרון אופטימלי, אז הפתרון האופטימלי הוא מושג על ידי קודקוד של הפוליהדרון.*

***משפט****:*

*יש לנו P שהוא פוליטופ*

*נניח שהמינימום הוא סופי.*

*אז נאמר שעבור כל קודקוד בפוליטופ יש קודקוד כך שמתקיים*

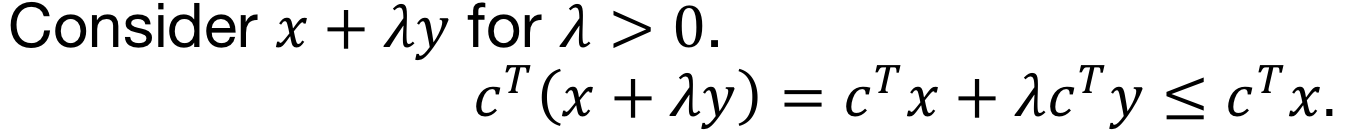
*הוכחה:*

*אם x הוא קודקוד סיימנו.*

*אחרת, ישנו כ כך שמתקיים   
זאת מכיוון ש וגם וגם*

*עכשיו נסתכל על קבוצה של וקטורים כי*

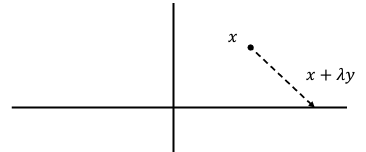
*פתרון אופטימלי עבור קודקוד*

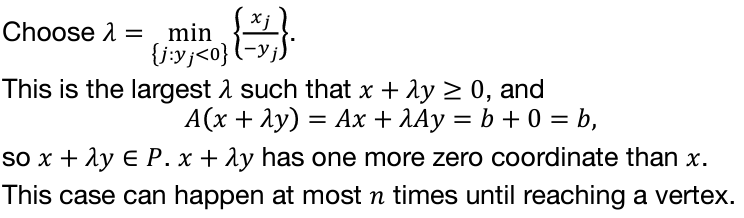
**

***במקרה הראשון :***

*כאשר יש כך ש כך ש מוגבר.*

*הקורדינטה הj מונמכת עד ש היא לא פיזיבילית יותר*

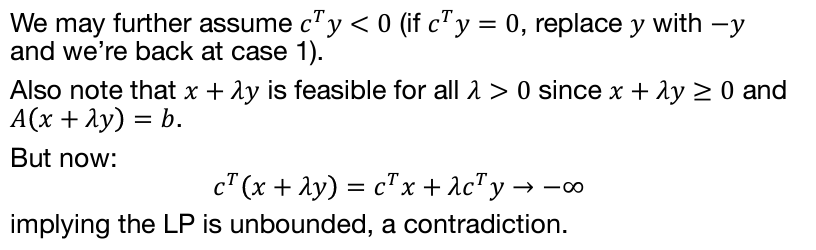
**

**

*לתמלל את זה\*\**

***במקרה השני:***

*כאשר עבור כל*

**

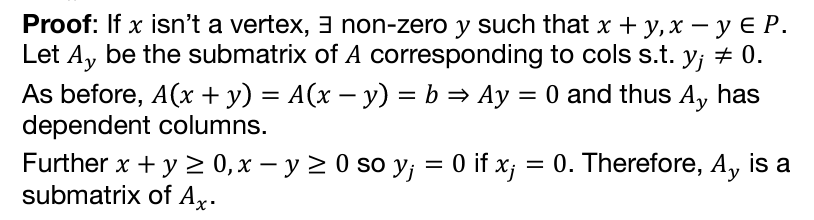
*לתמלל את זה\*\**

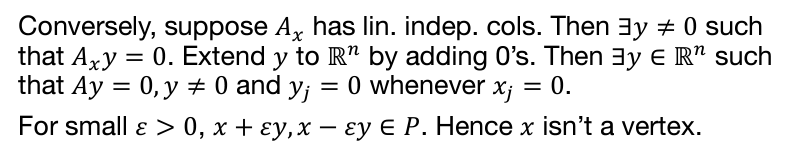
*קריטריון לינארי אלגברי לקודקודים*

***למה***

*יש לנו P שהוא פוליטופ עבור*

*תן ל להיות סאב מטריצה של A , אז x הוא קודקוד אם ורק אם יש לו עמודה מדורגת מלאה.*

*הוכחה:   
*

**

*לשים לב שכן תלוי ולא לא תלוי.*

*בP חייבים להיות וקטורים אי שלילים.*

*הנקודה המרכזית זו הלמה שאומרת לדעת אם אנחנו בקודקוד בפוליטופ, לוקחים מטריצה ומחשבים את הדרגה שלה.*

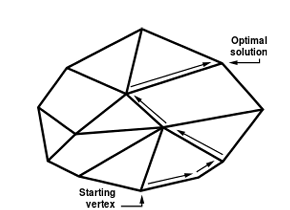
***איך האלגוריתם עובד? The simplex method***

*פותר בעיות של תכנון לינארי כלומר מוצא את האופטימום בפתרון אופטימלי. הוא מתחיל באיזשהו קודקוד ומנסה לשפר, מסתכל על השכנים ובודק אם יכול לשפר את מצבו ואם כן לעבור לקודקוד הסמוך.*

*אז הוא אומר זה האופטימום = אם אי אפשר לשפר.*

*אם הוא יכול לשפר = אז הוא משפר.*

*אולי אי אפשר לשפר אבל אפשר לעבור לפתרון טוב באותה מידה, והאם זה בכלל מסתיים בזמן סביר. נדבר על כל הנושאים הבאים. נדון גם בגודל הקלט וכו. מה השיטה הטובה, איך בוחרים מאיזה קודקודים להתחיל. על חלקן נענה ועל חלקן נגיד שאנחנו לא יודעים.*

*מתחיל באיזשהו קודקוד והולך ומוצא איזשהו משהו אופטימלי.  
*

*לפני שנציג את האלגוריתם צריך עוד קצת הגדרות ולהבין את היצורים האלה ( הפוליטופים)*

*הגדרות נוספות:*

* *X הוא קודקוד אם ורק אם בגודל m וגם :*

1. *הוא לא סינגולרי*

*נקרא לx הבייסיק פתרון פיסיבילי.*

*עכשיו יש לנו את כל הנתונים שאנחנו צריכים ונציג את זה בשבוע הבא את האלגוריתם הבא.*

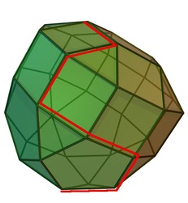
*המשך משבוע הקודם*

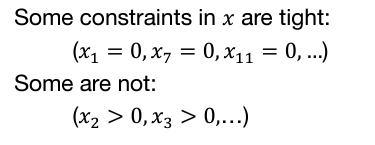
*לאיברים בB נקרא בסיסים ולאיברים שלא בB נקרא לא-בסיסים.*

*לגבי ההגדרות הנוספות זה שקול להגיד שהמטריצה ה היא הפיכה. וb חייב להיות אי שלילי*

*אם יש x שמקיים את כל התנאים שראינו אז הוא קודקוד.*

*בוא נניח שיש לנו בייסיק פיסיבל סולושן – פתרון אפשרי בסיסי, יש לנו קודקוד, נרצה ללכת על הצורה הזו מקודקוד לקודקוד ונראה איך אפשר לשפר אותו.*

**

*יש אילוצים שבהם הוא גדול ממש מאפס וקטן ממש מאפס, נבחר אילוץ הדוק כלשהו ונעשה לו רלקסציה, ננסה להגדיל את ויכולים לקרות כל מיני דברים.  
*

*למשל :*

*ננסה להגדיל את למשל אז הגענו למצב ש גדול מ0 שווה לאפס וזה בעצם ההליכה מקודקוד לקודקוד.*

*חוזרת מהתחלה:*

*יש אלגוריתם כזה שנקרא simplex שהולך על הקודקודים. וצריך להראות איך מראים את החישובים האלה מבחינה אלגברית.*

*צריך לזכור את העולם הגאומטרי.*

*האלגוריתם לא ירוץ בזמן פולינומי אבל בזמן יעיל.*

*יש את ההנחות שציינתי למעלה.*

*איקס שמקיים את זה נקרא בייסיק פיסיבל סולושן.*

*הוא יכול להיות אקספוננציאלי. זו תהיה בעיה שנתעלם ממנה כרגע.*

*איך עובד האלגוריתם ?*

*נניח שיש לנו x בייסיק פיסיבל סולושן, אז יש קודקוד ואנחנו רוצים ללכת על הפולהדרון מקודקוד לקודקוד ולראות אם אפשר לשפר.*

*יש אילוצים הדוקים ויש שלא הדוקים כלומר שגדול ממש מאפס:  
A picture containing text, font, white, typography

Description automatically generated*

*נבחר אילוץ הדוק ונעשה לו רילקסציה- ננסה להגדיל את אם נחשוב שזה עלול למקסם.*

*אם נגדיל את ונראה אם זה עוזר לאופטימום. כלומר האם זה עולה? ואז הגדלנו הכל וזה לא טוב ואז נשאיר כי זה מקרב אותנו לאפס.*

*נראה שזה מצב שהלכנו מקודקוד לקודקוד.*

*A picture containing text, font, white, receipt

Description automatically generated*

*הצבנו בפונקציה שנרצה לאפטמם ויש פה משהו שלא תלוי ב וחלק שכן תלוי בו.*

*הפונקציה שנרצה לאפטמם היא A picture containing font, typography, text, handwriting

Description automatically generated*

*המסקנה שהגענו היא שאנחנו רוצים לחשב את זה*

*A black text on a white background

Description automatically generated with low confidence*

*יש לנו כרגע פתרון בו הוא שווה לאפס.*

*אם הוקטור הזה הוא אי שלילי, אנחנו באופטימום.*

*כלומר אין דרך לשפר, אנחנו באופטימום.*

*כי כל y אחר יקיים גם את זה.*

*A picture containing text, font, white, receipt

Description automatically generated*

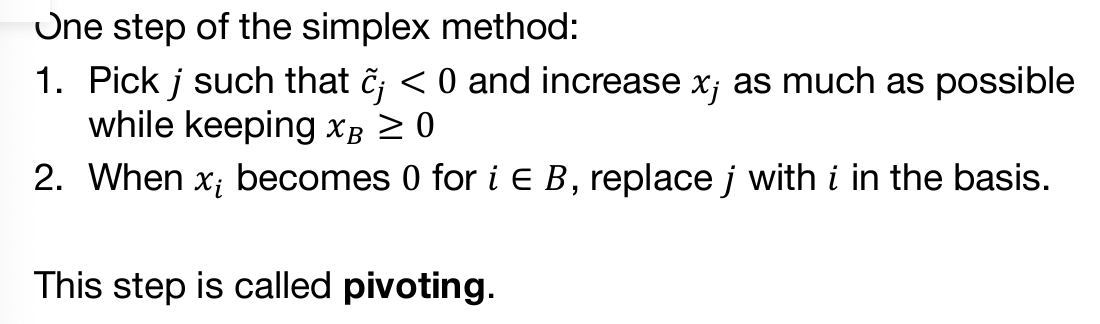
*אם הווקטור הוא שלילי,*

*A picture containing text, font, white, screenshot

Description automatically generated*

*בחירת j: יש כל מיני שיטות בחירה, נדבר עליהן אולי בהמשך.*

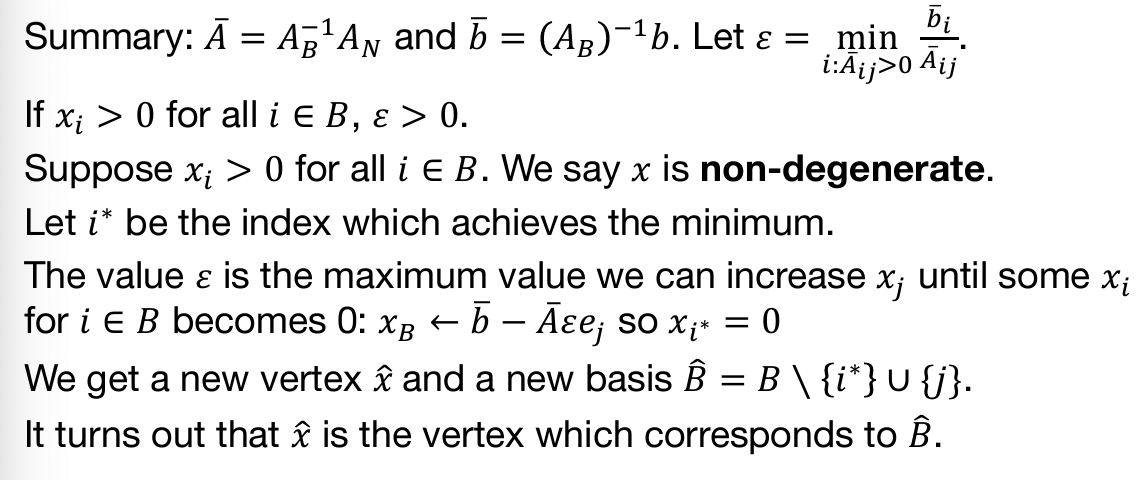
*צעד אחד כזה נקרא פיבוט:*

**

*צעד אחד של הסימפלקס ו למצוא j שישפר לנו . שנוכל להגדיל אותו עד שB י יהפוך ל0. ואז מחליפים, מוציאים את j ושמים את i.*

*A picture containing text, font, screenshot, white

Description automatically generated*

**

*הערות : אפסילון הוא חיובי – מה שהוספנו ל .*

*בהנחה שx אינו מנוון, אנחנו מתקדמים. ואין נתונים. צריך להראות שB כובע הוא בסיס כלומר שהעמודות שלו הן בלתי תלויות ושx כובע זה הקודקוד שמתאים לB סטאר:*

*A black text on a white background

Description automatically generated with low confidence*

*נתחיל בלהראות שB כובע הוא בסיס:*

*A picture containing text, font, screenshot, white

Description automatically generated*

*בעצם מה שהראינו שA כובע הוא מכפלה של שתי מטריצות הפיכות.*

*עוד שקופית שלא הקשבתי אליה:*

*A picture containing text, font, screenshot, white

Description automatically generated*

*השאלות שעכשיו צצות:*

1. *מאיפה מביאים את הקודקוד ההתחלתי? לא ברור בינתיים.*
2. *הפתרונות המנוונים – מה קורה אם אי אפשר להגדיל את ה. יכולים לצוץ מעגלים ולולאות אין סופית.*
3. *איזה j נבחר? יכולים להיות כמה j שיתנו לנו ערך אי שלילי.*
4. *מה הזמן ריצה של הסיפור.*

*חלק אין תשובות כלכך טובות.*

*זמן ריצה:*

1. *כל פיבוט אפשר לחשב בזמן פולינומי. אין לנו אף שיטה שמבטיחה שזה יהיה בזמן פולינומי. כי מספר הבסיסים יכול להיות אקספוננציאלי.*
2. *אם יש לו פתרון מנוון, זה יכול להיות זמן אינסופי אבל יש דרך להבטיח שזה בחיים לא ירוץ בזמן אינסופי.*

*הפתרונות המנוונים:*

*אם הפתרון מנוון, אנחנו לא משפרים.*

*נשנה את הבעיה כדי שנפתור בעיה טיפה אחרת:*

*A picture containing text, screenshot, font

Description automatically generated*

*נפתור בעיה שהיא טיפה שונה אבל היא קרובה לבעיה שלנו.*

*כדי לדעת אם יש פתרוונת אי שלילים למשוואה*

*למצוא התחלתי אי שלילי:*

*A picture containing text, font, screenshot, white

Description automatically generated*

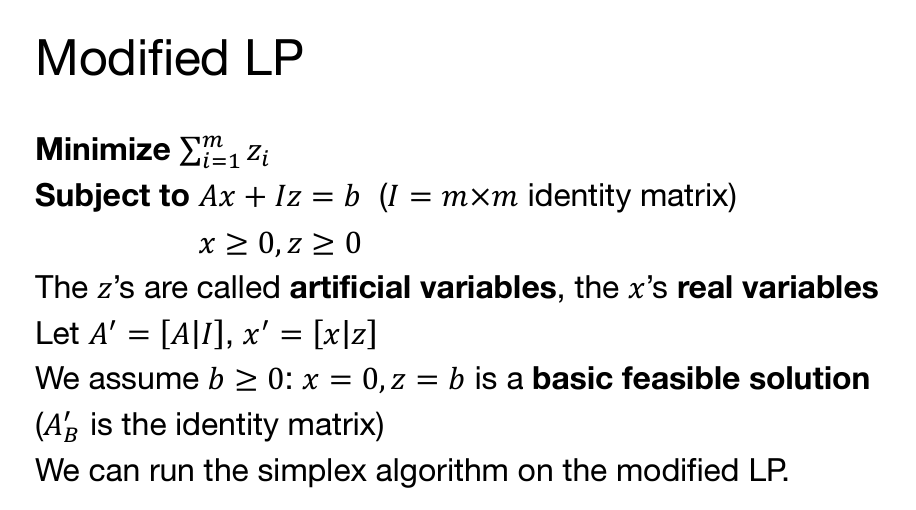
*חלק מהאינדקסים מתאימים לזד וחלק לאיקס- צריך לזכור את זה.*

*לפעמים יש לנו מערכת אילוצים שיש להם I .*

*הינה יש לנו בייסיק פיסיבל סולושן ועכשיו צריך לעבור לפיסיבל סולושן של המערכת המקורית אז לא סיימנו.*

*צריכים קודקוד לא סתם נקודה.*

*עוד לא ברור איך עושים את ההתאמות*

**

*לשים לב שתמיד הLP תמיד חסומה כי האילוץ שהוא גדול שווה לאפס.*

*נסתכל על כל המקרים אחד אחד:*

*A black text on a white background

Description automatically generated with low confidence*

*אם הוא גדול מאפס זה אומר שהמערכת המקורית היא לא פיסיבילית. כי אם היא הייתה פיסיבילית אז הערך היה אפס.*

*בתנאי שהוא אפס:*

*A picture containing text, font, screenshot, algebra

Description automatically generated*

*יש מקרה טוב ומקרה רע:*

*אפשר לשכוח מהמשתנים המלאכותיים כי הם כבר עשו את שלהם.*

*A picture containing text, font, screenshot, white

Description automatically generated*

*הוכחה של הטענות:*

*A picture containing text, font, screenshot, line

Description automatically generated*

*A picture containing text, font, screenshot, algebra

Description automatically generated*

*A picture containing text, screenshot, font, algebra

Description automatically generated*

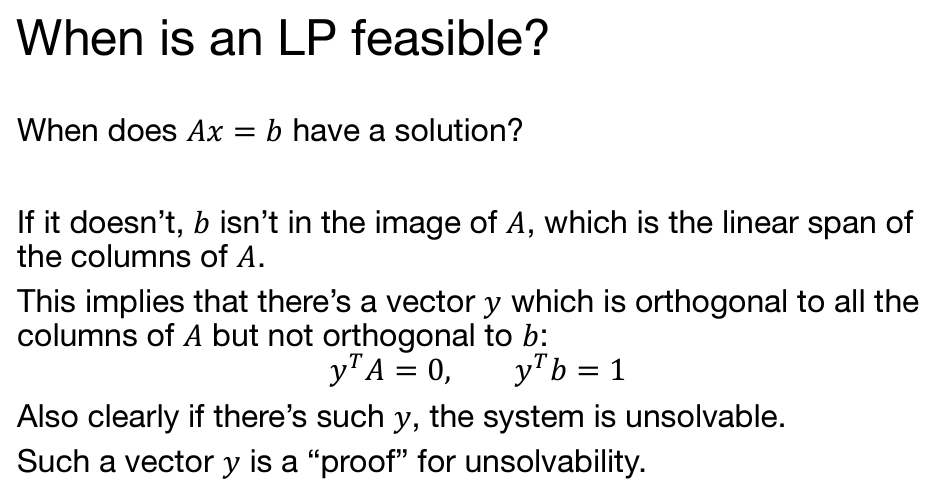
*A picture containing text, font, screenshot, white

Description automatically generated*

*האלגוריתם פשוט:*

1. *למצוא בייסיק פיסיבל סולושן – לתת. LP מלאכותית להריץ סימפלקס, למצוא פתרון עם איקסים, ואז יש לנו קודקוד של המערכת המקורית.*
2. *בשלב השני פיבוטינג, מנסים לשפר, מחשבים את הרידוס ורואים אם אפשר בכלל לשפר.*

*האם LP הוא פיסיבלי בצורה יעילה?*

**

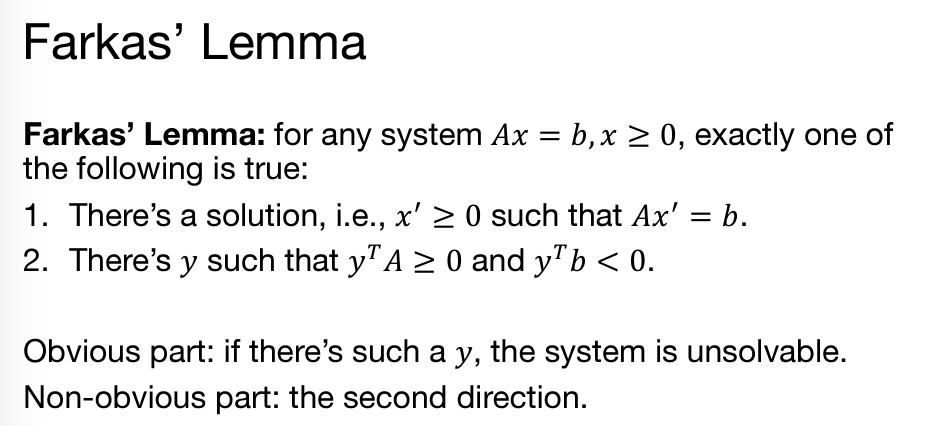
*מדברים פה על הוקטורים שניצבים לו – שהמכפלה הפנימית שווה 0*

*זה מתורת הקבוצות.*

*A picture containing text, screenshot, font, algebra

Description automatically generated*

*הלמה של פרקש:*

**

*A picture containing text, font, screenshot, white

Description automatically generated*

*A picture containing text, font, screenshot, white

Description automatically generated*

*בתרגיל הבית נראה על פרקש למה 1*

*פה יש לנו פרקש למה 2*