13.4.2026	***מודלים חישוביים***

הרצאה 1: מעגלים

Based on previous iterations of this course, given by Nir Bitansky, Rotem Oshman, Iftach Haitner, and Omer Paneth.

מרצה: אורי שטמר

<u>על מה הקורס?</u>

אחת המטרות העיקריות של האקדמיה בכלל היא "להבין את העולם שבו אנו חיים/פועלים". למשל להבין את כח המשיכה בפיזיקה. בהקשר שלנו בתור מדעני מחשב, אנחנו רוצים להבין מזה "חישוב" ומהן היכולות והמגבלות שלו. זאת בדיוק המטרה של הקורס שלנו. לצורך כך נבחן כמה <u>מודלים לחישוב</u> וננסה להבין את היכולות שלהם.

<u>שאלה:</u> מזה "מודל חישוב"?
<u>תשובה:</u> בצורה לא פורמלית, "אלגוריתם" הוא מתכון שאומר למחשב מה לעשות. "מודל חישוב" אומר לנו איך המתכון הזה צריך להראות ומה מותר לו לכלול.

<u>נושאים:</u>

- מודל חישוב: מעגלים בוליאניים
  - אינטואיטיבית, אלו אלגוריתמים שאפשר לממש אותם על ידי "מעגל אלקטרוני", כמו למשל מעגל להכפלת 2 מספרים בני 64 ביט כל אחד.
  - מממשים פונקציה ספציפית עבור קלטים באורך ספציפי
  - מודל החישוב הזה הוא לא באמת ראליסטי אבל הוא עדיין מאוד שימושי להבנת נושא החישוב באופן יותר רחב (נדבר על זה היום).
- מודל חישוב: אוטומטים סופיים
  - מודל חישוב מאוד פשוט. אינטואיטיבית, זה ממדל אלגוריתמים עם זיכרון מאוד מאוד קטן, נגיד 10 ביטים של זיכרון.
  - למשל, קונטרולר של רמזור: זאת מערכת בעלת מספר סופי של מצבים (אדום, צהוב, ירוק) ומעברים ביניהם המבוססים על טיימרים או חיישנים.
- מודל חישוב: מכונות טיורינג (מ"ט)
  - זה מודל שהוא כבר מספיק אקספרסיבי כדי לתפוס אלגוריתמים כליים כמו שאנחנו מכירים. המודל הזה "שקול" (מבחינת החישובים שאפשר לבצע איתו) לפייתון או ל C++, אבל מבחינת התאור המתמטי שלו הוא יהיה יותר פשוט.
  - בחיים האמיתיים אף אחד לא כותב תוכניות בעזרת מכונות טיורינג, זה יהיה יותר נורא מאסמבלי... הייתרון של מ"ט הוא שהתיאור המתמטי של המודל עצמו הוא יחסית פשוט וזה יאפשר לנו להוכיח פורמלית דברים לגבי היכולות של המודל.
  - זה יאפשר לנו להוכיח שיש בעיות שמ"ט לא יכולות לפתור, ולכן גם אין לבעיות האלה אלגוריתמים בפייתון למשל. (אלו יהיו בעיות קצת "מוזרות"; לא באמת בעיות שאנחנו רוצים לפתור ביום יום).
- סיבוכיות
  - נדבר על בעיות אלגוריתמיות שבאמת היינו רוצים לפתור, אבל משום מה הן קשות.
  - הבעיות האלה הן בבירור "פתירות", אבל דורשות משאבים רבים (למשל זמן רב).
  - דוגמה: "תכננו את הטיול הזול ביותר שעובר בכל בירות אירופה".

מושגים בסיסיים

על מנת לדבר על בעיות ואלגוריתמים לבעיות האלה, נצטרך קודם להסביר איך מייצגים קלטים

הגדרה: אלפבית היא קבוצה סופית (לא ריקה) של תווים. נסמן Σ

דוגמאות:

$$\Sigma = \{0,1\}$$
$$\Sigma = \{a, b, c, \dots, z\}$$

הגדרה: מילה מעל אלפבית Σ היא שרשור של מספר סופי של תווים מ- Σ

סימונים:

- נסמן ב- Σ^* את אוסף כל המילים מעל Σ
- נסמן ב- Σ^n את אוסף כל המילים באורך n מעל Σ
- נסמן ב ε את המילה הריקה

לדוגמה:

$$\{0,1\}^* = \{\varepsilon, 0,1,00,01,10,11, \dots\}$$

הגדרה: שפה מעל אלפבית Σ היא תת קבוצה של Σ^*

- עשויה להיות סופית (אפילו ריקה), אינסופית, יכולה להיות כל Σ^*.
- דוגמאות:
  - השפה האנגלית מעל הא"ב האנגלי
  - $\{0^n1^n : n \in \mathbb{N}\}$ = כל המחרוזות הבינאריות מהצורה 0000011111 כאשר מספר האפסים זהה למספר האחדים
  - $\text{Primes} = \{p \in \mathbb{N} : p \text{ ראשוני}\}$

בעיות הכרעה:

רוב הבעיות החישוביות שנעסוק בהן הסמסטר יהיו "בעיות הכרעה", כלומר בעיות מהצורה:

"האם קלט $x \in \Sigma^*$ שייך לשפה L?" (עבור שפה $L \subseteq \Sigma^*$ כלשהי)

במילים אחרות, בהינתן קלט x נרצה לקבוע האם x מקיים תכונה כלשהי.

למשל, בהינתן מספר $p \in \mathbb{N}$ נרצה לקבוע האם p ראשוני, כלומר האם $p \in \text{Primes}$

הגדרה (חצי פורמלית):
נאמר ש"אלגוריתם" מכריע שפה $L \subseteq \Sigma^*$ אם לכל $x \in L$ האלגוריתם מחזיר "כן" (נאמר שהאלגוריתם "מקבל" את x) ולכל $x \notin L$ האלגוריתם מחזיר "לא" (נאמר שהאלגוריתם "דוחה" את x).

דוגמה: האם אתם יכולים לחשוב על אלגוריתם המכריע את *Primes*?

<u>הערה:</u> לא כל הבעיות שמעניינות אותנו הן בעיות הכרעה. למשל בעיית חיפוש: "בהינתן p שאינו ראשוני, מצאו את גורמיו הראשוניים". מסתבר שבהרבה מקרים יש קשרים מאוד הדוקים בין בעיות הכרעה לבעיות חיפוש (יתבהר בהמשך הקורס).

מעגלים בוליאניים

- מעגל בוליאני C מחשב פונקציה $f:\{0,1\}^n \to \{0,1\}^m$ על ידי הרכבה של "פונקציות בסיס".

הפונקציה f הזאת יכולה להיות מאוד מורכבת, אבל הצורה שבה המעגל מחשב אותה זה ע"י "חזרה"/"שילוב" של פעולות מאוד בסיסיות. פעולות הבסיס העיקריות שנדבר עליהן אלו פעולות דה-מורגן:

- בסיס דה-מורגן: $B = \{\wedge, \vee, \neg\}$

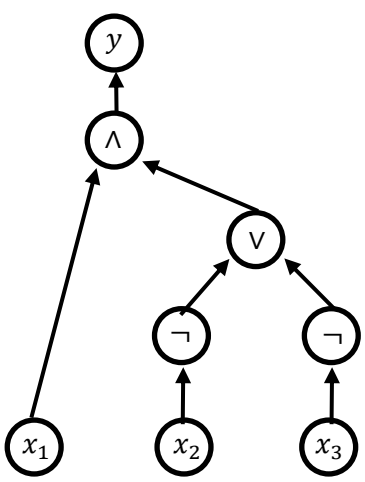
כשנבנה מעגל לחישוב איזושהי פונקציה f, ניקח הרבה "שערים" שכל אחד מהם מחשב אחת מהפעולות הבסיסיות האלה "ונרכיב" את השערים האלה כך שביחד יחשבו את הפונקציה היותר מורכבת f

x_1	x_2	$x_1 \wedge x_2$	$x_1 \vee x_2$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

x_1	$\neg$
0	1
1	0

דוגמה: מעגל המחשב את הפונקציה

$$f(x_1, x_2, x_3) = x_1 \wedge (\neg x_2 \vee \neg x_3)$$



שאלה: האם זהו המעגל היחיד שמחשב את הפונק' הנ"ל?
תשובה: לא, למשל ע"י שימוש בכלל דה-מורגן היינו יכולים לכתוב את הפונק' כך:
$f(x_1, x_2, x_3) = x_1 \wedge (\neg x_2 \vee \neg x_3) = x_1 \wedge \neg(x_2 \wedge x_3)$

דיון: למה מעגלים בוליאניים?

- גישה מאוד ישירה לחישוב. הפעולות של "וגם", "או", "שלילה" הן פעולות מאוד בסיסיות.
- מדוע בוליאני? משום שכך מממשים מעגלים במציאות. יש לנו מעגלים אלקטרוניים עם חוטי מתכת כאשר "0" מיוצג על ידי מתח חשמלי נמוך ו- "1" מיוצג על ידי מתח חשמלי גבוה.
- איפה משתמשים במעגלים אלקטרוניים במציאות? בשביל לבצע משימות בסיסיות שאנחנו מבצעים באופן חוזר. למשל המעבד במחשב שלנו הוא איזשהו מעגל אשר יודע לבצע אוסף של פעולות בסיסיות ואנחנו מריצים אותו המון פעמים ובעזרת הפעולות האלה אנחנו יודעים לעשות פעולות מורכבות יותר. אז הדבר הזה חשוב בחומרה.

עכשיו נעבור להגדרה הפורמלית

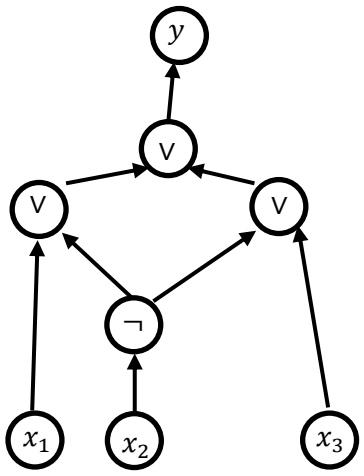
הגדרה: תהי B קבוצה של פונקציות בוליאניות (כלומר מעל 0,1). מעגל בוליאני מעל B עם ביטי קלט $x_1, \dots, x_n$ וביטי פלט $y_1, \dots, y_m$ הוא גרף מכוון חסר מעגלים מכוונים כאשר:

- כל צומת מסומן על ידי: פונקציה $g \in B$, או קלט x_i, או פלט y_i
- לכל ביט פלט y_i יש בדיוק צומת אחד המסומן ב y_i. דרגת הכניסה של צומת זה היא 1 ודרגת היציאה שלו היא 0.
- דרגת הכניסה של כל צומת המסומן בביט קלט x_i היא 0.
- לכל צומת המסומן בפונקציה $g \in B$, אם g מוגדרת על $\{0,1\}^k$ אז ישנן k קשתות הנכנסות לצומת עם סדר המסומן ב $1,2,\dots,k$. עבור פונקציה $g \in B$ סימטרית (כלומר סדר הקלטים לא משנה) אין צורך בסימון סדר הקשתות הנכנסות.

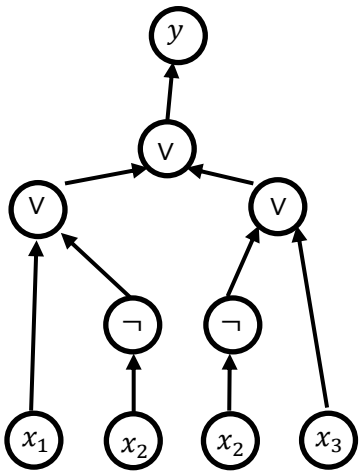
מונחים:

- צומת המסומן בפונקציה $g \in B$ נקרא שער (gate)
- הקשתות נקראות חוטים (wires)
- ה fan-out של מעגל הוא דרגת היציאה המקסימלית של שער (gate) כלשהו במעגל
- תת הקבוצה של המעגלים בהן ה fan-out הוא 1 נקראים גם נוסחאות. הסיבה היא שאפשר לקחת מעגל כזה עם n פעולות ולכתוב אותו באופן טבעי כנוסחה בעזרת אותן n פעולות בדיוק (עם סוגריים במקום חוטים)

דוגמה: $f(x_1, x_2, x_3) = (x_1 \vee \neg x_2) \wedge (\neg x_2 \vee x_3)$



מעגל: fan-out=2



נוסחה: fan-out=1

באופן כללי, כאשר מדברים על סיבוכיות, מעגלים זה דבר יותר אקספרסיבי מנוסחאות, במובן הזה שישנם פונקציות שכל נוסחה שמתארת אותם היא מאוד גדולה בעוד שאפשר לתאר אותם ע"י מעגל שהוא יחסית קטן.

<u>מאמר מוסגר:</u> למשל לפונק' PARITY אפשר לבנות מעגל בגודל לינארי אבל אפשר להוכיח שנוסחה חייבת להיות בגודל ריבועי. אם תנסו לבנות נוסחה עבור PARITY אתם תראו שאתם צריכים "לשכפל" הרב החלקים מהנוסחה שתבנו, מה שיוביל לגודל ריבועי:

$$\phi(x_1) = x_1$$
$$\phi(x_1, x_2) = (x_1 \wedge \overline{x_2}) \vee (\overline{x_1} \wedge x_2)$$
$$\phi(x_1 \dots x_n) = \left(\phi(x_1 \dots x_{n/2}) \wedge \overline{\phi(x_{n/2} \dots x_n)}\right) \vee \left(\overline{\phi(x_1 \dots x_{n/2})} \wedge \phi(x_{n/2} \dots x_n)\right)$$

כלומר כדי לבטא את $\phi(x_1 \dots x_n)$ אנחנו צריכים פעמיים את $\phi(x_1 \dots x_{n/2})$ ופעמיים את $\phi(x_{n/2} \dots x_n)$.
אם אתם רוצים לבנות נוסחה אז אנחנו לא יכולים "למחזר" ולכן נוסחת הנסיגה עבור הגודל היא:

$$F(n) = 4 \cdot F(n/2) + O(1)$$

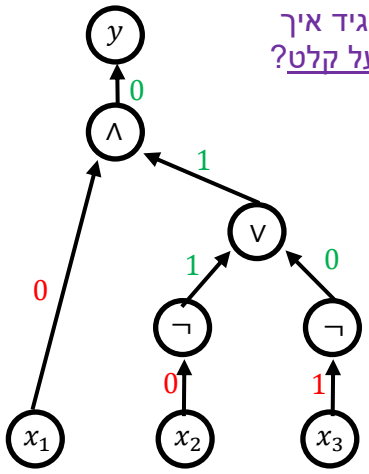
פתרון נוסחת הנסיגה הזאת (לפי משפט המאסטר) נותן $F(n) = \Theta(n^2)$.

אבל אם אתם רוצים לבנות מעגל עם fan-out=2 אז אין לכם בעיה "למחזר" ואז נוסחת הנסיגה עבור הגודל תהיה:

$$F(n) = 2 \cdot F(n/2) + O(1)$$

הפעם, הפתרון לנוסחת הנסיגה הזאת יהיה $F(n) = \Theta(n)$.

אז הגדרנו איך נראה מעגל. בשביל להשלים את המודל החישובי, כלומר להגיד איך הדבר הזה מתפקד כאלגוריתם, אנחנו רוצים להסביר <u>איך משערכים מעגל על קלט</u>?



למשל, איך נעריך את המעגל הבא על הקלט $v = 001$?

- תחילה נבצע השמה של ערכים לחוטי הקלט המתאימים
- נחפש שער שכל החוטים שנכנסים אליו הם כבר עם השמה כלשהי, נשערך אותו, ונמשיך

שערוך מעגל על קלט:

- בהינתן מעגל C עם n ביטי קלט וקלט $v \in \{0,1\}^n$
- מציבים ערכים לחוטים באופן איטרטיבי:
  - <u>צעד ראשון:</u> לכל $i \in [n]$ מציבים ערך v_i לחוטים היוצאים מצומת הקלט המסומן ב- x_i
  - <u>צעד איטרטיבי:</u> מוצאים שער שכל הכניסות שלו חושבו (והיציאה טרם חושבה). מחשבים את הפונק' g המתאימה על ערכי הכניסה ומציבים לחוט היציאה.
  - חוזרים על הצעד האיטרטיבי כל עוד אפשר.
  - הפלט $C(v)$ הנו הערכים שהוצבו לכניסות של $y_1, \dots, y_m$

<u>טענה:</u> התהליך מוגדר היטב, כלומר לכל חוט הצבה אחת ויחידה.

<u>הוכחה:</u> תרגיל (השתמשו באציקליות של הגרף).

הגדרה:
נאמר כי מעגל C <u>מחשב פונקציה</u> $f:\{0,1\}^n \to \{0,1\}^m$ אם לכל קלט $v \in \{0,1\}^n$ מתקיים $C(v) = f(v)$.

אז עכשיו אנחנו יודעים מזה מעגל ואיך מחשבים באמצעות מעגל. עכשיו ננסה להבין מה מעגלים יכולים לעשות

<u>אלו פונקציות ניתנות לחישוב ע"י מעגל?</u>

הנה משפט שכנראה נתקלתם בו בקורס בלוגיקה:

משפט (אוניברסליות של דה-מורגן):
לכל פונקציה $f:\{0,1\}^n \to \{0,1\}^m$ קיים מעגל מעל בסיס דה-מורגן המחשב אותה.

<u>הוכחה (סקיצה):</u>

נתחיל מהמקרה הפשוט יותר שבו יש לנו רק ביט אחד של פלט, כלומר $m = 1$

נתבונן בטבלת האמת של f.

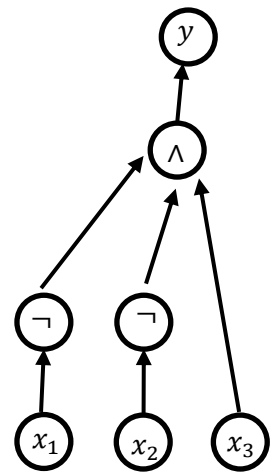
לדוגמה

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

לכל שורה $v \in \{0,1\}^n$ כך ש- $f(v) = 1$ נבנה מעגל I_v "המזהה" את השורה הזאת, כלומר מעגל I_v כך ש- $I_v(x) = 1$ אם ורק אם $x = v$.

אנחנו משתמשים באות I כאן כי אנחנו חושבים על המעגל הזה כמו indicator עבור הקלט המתאים

לדוגמה, עבור השורה 001 המעגל I_{001} יחשב את הנוסחה $\neg x_1 \wedge \neg x_2 \wedge x_3$
הנה המעגל:



שימו לב שעשינו כאן קצת abuse of notation: הכנסנו לשער "וגם" שלושה ערכים במקום שני ערכים.

<u>טענה:</u> ניתן לממש שער "וגם" של n משתנים ע"י n-1 שערי "וגם" של שני משתנים (על ידי עץ). אותה טענה גם נכונה לשערי "או".

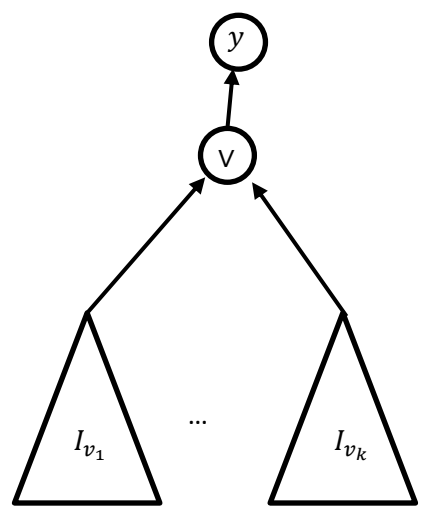
באופן כללי:

$$I_v(x) = \bigwedge_{i:v_i=1} x_i \bigwedge_{i:v_i=0} \neg x_i$$

כעת, המעגל (למעשה נוסחה) המחשב את f יהיה:

$$f(x) = \bigvee_{v:f(v)=1} I_v(x)$$

בציור:



$\forall v_i \in f^{-1}(1)$

אינטואיטיבית זה מאוד ברור שמעגל הזה אכן מחשב את הפונקציה f.
פורמלית, אנחנו צריכים להראות שלכל קלט v מתקיים $C(v) = f(v)$.
אז יהי v קלט כלשהו.
אם $f(v) = 1$ אז תת המעגל המתאים I_v יקיים $I_v(v) = 1$ ולכן שער ה $\vee$ העליון יחזיר 1.
אם $f(v) = 0$ אזי אף אחד משערי I_{v_i} הנ"ל לא יחזיר 1 ולכן גם שער ה $\vee$ העליון לא יחזיר 1.

מה לגבי המקרה $m > 1$, כלומר פונקציה עם יותר מביט אחד של פלט?

אנחנו יכולים לחשוב על כל ביט של פלט כפונקציה נפרדת:

$$f_1(x)\,, f_2(x)\,, \dots, f_m(x)$$

לכל אחד מביטי הפלט אנחנו יודעים איך לכתוב מעגל. נשים את המעגלים האלה אחד ליד השני.

מ.ש.ל.

אוקיי, אז ראינו שכל פונק' אפשר לתאר עם מעגל. זה נשמע טוב, זה נשמע מודל מאוד אקספרסיבי, למה אנחנו לא משתמשים בו יותר בחיים? הסיבה היא שמעגלים כאלה הם מאוד שונים מאלגוריתמים כפי שאנחנו מכירים אותם, במובן שנקרא "חוסר יוניפורמיות".

<u>מעגלים הם מודל לא-יוניפורמי</u>

- מעגל נתון מחשב פונקציה על אורך קלט <u>מסוים</u>
- על מנת לפתור בעיה על קלטים באורך שרירותי נצטרך מעגל נפרד לכל אורך קלט
- זה שונה מאיך שאנחנו בדרך כלל חושבים על אלגוריתמים, כאשר אותו אלגוריתם עובד עבור קלטים בכל אורך (למשל האלגוריתם לבדיקת ראשוניות עליו דיברנו)

ישנם שני סוגי מודלים עבור חישוב של פונק' $f: \{0,1\}^* \to \{0,1\}^*$ באורך שרירותי:

- **יוניפורמי:** אלגוריתם יחיד לכל אורך קלט
- **לא-יוניפורמי:** אלגוריתם שונה לכל אורך קלט. כלומר, לחישוב $f: \{0,1\}^* \to \{0,1\}^*$ בעצם נדרשת "משפחה" של מעגלים, כי היינו צריכים לכל אורך קלט לתאר מעגל נפרד. זה לא מודל ריאליסטי.

הגדרה:
<u>משפחה של מעגלים</u> $\mathbb{C} = \{C_n\}_{n\in\mathbb{N}}$ היא אוסף אינסופי של מעגלים, כך ש- C_n מוגדר על קלטים באורך n

אז זה משפחה של מעגלים. זה לא משהו שמשתמשים בו במציאות בתור מודל חישוב, אבל במובן המתמטי אנחנו עדיין יכולים לדבר על "לפתור בעיות מסוימות בעזרת משפחת מעגלים".

הגדרה:
תהי $\mathbb{C} = \{C_n\}_{n\in\mathbb{N}}$ משפחת מעגלים עם ביט פלט יחיד ותהי $L \subseteq \{0,1\}^*$ שפה. נאמר כי <u>$\mathbb{C}$ מכריעה את L</u> אם לכל $n \in \mathbb{N}$ ולכל $x \in \{0,1\}^n$ מתקיים $C_n(x) = 1$ אם ורק אם $x \in L$.

<u>הערה טכנית:</u>
על-מנת ש- $\mathbb{C}$ תהיה מוגדרת גם במקרה בו $n = 0$ (כלומר הקלט הריק ε) נרשה שערים קבועים ZERO, ONE

אבל אם "משפחה של מעגלים" זה מודל לא ריאליסטי, למה בכל זאת אנחנו לומדים אותו ומתעניינים בו?
קודם כל, כמו שאמרנו, מעגלים הם כן שימושיים עבור מקרים בהם אנחנו יודעים מראש את אורך הקלט ורק האורך הזה מעניין אותנו, למשל CPU. אבל מעבר לכך, אנחנו מתעניינים במעגלים מסיבות תאורטיות: להבין מעגלים זה במובן מסוים יותר קל מאשר להבין תוכניות מחשב כלליות כי המודל הוא קומבינטורי ויחסית קל לטעון לגביו דברים. למשל, לנסות להבין כמה "זמן" לוקח לחשב משהו באמצעות מעגל יכול לעזור לנו להבין כמה זמן לוקח לחשב אותו באמצעות אלגוריתמים כלליים (למשל בפייתון).

<u>סיבוכיות מעגלים (על קצה המזלג)</u>

אחת מהשאלות המרכזיות שיהיו לנו בקורס היא:
בהינתן פונקציה f שמעניינת אותנו, כמה "קשה" לחשב אותה?

למשל, נחזור לשפה Primes. האם קיימת סדרת מעגלים שמכריעה את השפה הזאת?
כן – זה נובע מהמשפט שראינו – לכל אורך קלט ולכל פונק' עבור אורך הקלט הזה, קיים מעגל. לכן גם קיימת משפחה. לכן השאלה היא בעצם "כמה מסובכים" צריכים להיות המעגלים האלו. אחת ממידות הסיבוכיות העיקריות של מעגלים זה <u>הגודל</u> של המעגל (מספר השערים). זה במובן מסויים מהווה אנלוג של זמן החישוב, כי כדי לחשב את המעגל אנחנו צריכים לחשב כל שער במעגל. יש עוד מדדים מעניינים, למשל כמה המעגל רדוד, או מהו ה fan-in/fan-out. אנחנו נחשוב בעיקר על הגודל של המעגל.

האם אנחנו יכולים להשתמש במודל היחסית פשוט הזה של מעגלים כדי להוכיח משהו על אלגוריתמים כלליים, למשל בפייתון? מסתבר שכן! נראה עכשיו שאם קשה לחשב פונקציה מסויימת ע"י משפחה של מעגלים בגודל מסויים אז נוכל להגיד שגם אלגוריתמים כמו שאנחנו מכירים אותם דורשים הרבה זמן

הגדרה:
- <u>גודל מעגל</u> C הוא מספר השערים ב- C, מסומן ב- $|C|$.
- עבור משפחת מעגלים $\mathbb{C} = \{C_n\}_{n\in\mathbb{N}}$ ועבור פונקציה $S: \mathbb{N} \to \mathbb{N}$ נאמר <u>שגודל משפחת המעגלים</u> הוא לכל היותר S אם לכל $n \in \mathbb{N}$ מתקיים $|C_n| \le S(n)$.

<u>טענה:</u> כל פונקציה $f: \{0,1\}^n \to \{0,1\}$ ניתן לחשב ע"י מעגל בגודל $O(n \cdot 2^n)$.

(נובע מהבניה הכללית שראינו למעגל מעל שערי דה-מורגן)

שאלה: האם גם בפייתון אפשר לחשב כל פונקציה בזמן אקספוננציאלי לכל היותר? לא. בפייתון יש פונק' שאי-אפשר לחשב. אנחנו נקרא לפונק' כאלה "בלתי כריעות".

<u>בתרגול:</u> למעשה ב- $O(2^n)$.

משפט לופיאנוב (לא נוכיח): למעשה ב- $O(2^n/n)$.

משפט לופיאנוב הוא אולי קצת מפתיע: איכשהו קורה פה משהו שהוא פחות מלעבור על כל הקלטים...

טענה (שאנון): עבור n מספיק גדול, קיימות פונקציות שלא ניתנות לחישוב ע"י מעגלים בגודל $\frac{2^n}{10n} > S$

כלומר משפט לופיאנוב הדוק עד כדי קבועים

<u>הוכחה:</u> טיעון ספירה: נראה כי יש יותר פונקציות $f:\{0,1\}^n \to \{0,1\}$ מאשר מעגלים בגודל S.

- מספר הפונקציות: 2^{2^n} (כי יש לנו 2^n קלטים ולכל קלט אנחנו בוחרים אחד מ-2 פלטים אפשריים)

אנחנו רוצים להראות שעבור גודל $2^n/10n > S$, מספר המעגלים בגודל S הוא הרבה יותר קטן ממספר הפונקציות. זה אומר שיש פונקציות שהן לא מחושבות ע"י אף אחד מהמעגלים האלו. פשוטו אין מספיק מעגלים בגודל הזה כדי לחשב את כל הפונקציות.

- נראה חסם על מספר המעגלים בגודל S: (ספירת יתר, כלומר נחסום מלמעלה)
  - בחירת סוג לכל אחד מהשערים 3^S
  - בחירת לכל היותר שני ילדים לכל שער $(S+n)^{2S}$
  - בחירת ילד עבור הפלט $(S+n)$
  - סה"כ $3^S \cdot (S+n)^{2S+1}$

טענה: $\underbrace{2^{2^n}}_{\text{מספר הפונקציות}} \gg \underbrace{3^S \cdot (S+n)^{2S+1}}_{\text{חסם עליון על מספר המעגלים}}$ עבור $2^n/10n > S$ (כאשר n מספיק גדול).

משמעות?

- כמעט כל פונק' דורשת מעגלים אקספ'.

<u>שאלה:</u> כאן ספרנו (או יותר נכון חסמנו) את מספר המעגלים בגודל <u>בדיוק</u> S. מה לגבי מעגלים קטנים יותר? כלומר, האם מה שאנחנו צריכים להראות זה באמת $3^S \cdot (S+n)^{2S+1} \gg 2^{2^n}$ או שבעצם צריך להראות שמתקיים $\sum_{\ell=1}^{S} 3^\ell \cdot (S+n)^{2\ell+1} \gg 2^{2^n}$?

<u>תשובה:</u> מספיק להראות שמתקיים $3^S \cdot (S+n)^{2S+1} \gg 2^{2^n}$. זה מראה שיש הרבה פונקציות שאין להם מעגל בגודל בדיוק S. כעת נשים לב שאם לפונקציה מסוימת f אין מעגל בגודל S אז גם אין לה מעגל בגודל t עבור $t < S$. מדוע? אם היה לה מעגל בגודל t אז היינו יכולים "להרחיב" אותו בצורה פיקטיבית ע"י שערים נוספים שאין מסלול מהם לאף צומת פלט ולקבל מעגל בגודל S עבור f. לכן מספיק להראות שאין ל- f מעגל בגודל <u>בדיוק</u> S.

המשפט הבא אומר לנו שאם פונקציה מסוימת היא "קשה" למעגלים אז היא גם קשה לתוכניות כלליות. זה נותן מוטיבציה מאוד חזקה ללמוד על מעגלים.

משפט: קיים קבוע $0 < c < 1$ כך שהטענה הבאה מתקיימת:
תהי $f:\{0,1\}^* \to \{0,1\}$ פונקציה. אם כל משפחת מעגלים $\mathbb{C} = \{C_n\}_{n\in\mathbb{N}}$ עבור f היא בגודל לפחות $t(n)$ אז f כל אלג' בפייתון עבור f דורש זמן לפחות $t^c(n)$.

- נחזור למשפט הזה בסוף הקורס (כנראה בהרצאה 12).
- לפי משפט שאנון, זה אומר שכמעט כל פונקציה דורשת מעגל בגודל לפחות $\frac{2^n}{10n}$ ולכן דורשת זמן אקספוננציאלי בפייתון (זמן לפחות $2^{cn}/(10n)^c$).
- נקודת אור? משפט שאנון לא אומר שום דבר על אף פונק' מסויימת. יתכן שכל הפונק' "המעניינות" ניתנות לחישוב בזמן קצר.

<u>תרגיל כיתה:</u>
תכננו מעגל בוליאני (מעל בסיס דה-מורגן) לחיבור שני מספרים בני n ביטים כ"א (בייצוג בינארי).

<u>פתרון ראשון:</u> נחשוב על חיבור בשיטה הרגילה, ביט אחרי ביט:

$$\begin{array}{r} 1\,1\,1 \\ +\quad 1\,0\,1\,0\,1\,1 \\ 1\,0\,1\,1\,1\,0 \\ \hline 1\,0\,1\,1\,0\,0\,1 \end{array}$$

נרשום את זה בצורת מעגל ע"י כך שנחשב את הסכומים ואת ה-carry-ים:

$$\begin{array}{r} c_1 \\ a_{n-1} \cdots\cdots\cdots a_1 \;\; a_0 \\ +\quad b_{n-1} \cdots\cdots\cdots b_1 \;\; b_0 \\ \hline s_n \;\; s_{n-1} \cdots\cdots\cdots s_1 \;\; s_0 \end{array}$$

$\boldsymbol{c_0 = 0}$ כי אין לנו *carry* לספרה האפס

$\boldsymbol{c_1 = a_0 \wedge b_0}$ אם שניהם אחד אז יש לנו carry

$\boldsymbol{c_{i+1} = (a_i \wedge b_i) \vee (a_i \wedge c_i) \vee (b_i \wedge c_i)}$ אם לפחות 2 מקבלים 1 אז יש carry

$\boldsymbol{s_i = (a_i \wedge b_i \wedge v_i) \vee (a_i \wedge \overline{b_i} \wedge \overline{c_i}) \vee (\overline{a_i} \wedge b_i \wedge \overline{c_i}) \vee (\overline{a_i} \wedge \overline{b_i} \wedge c_i)}$ או ששלושתם מקבלים 1 או שבדיוק 1 מקבל 1

זה מגדיר מעגל לחיבור.

<u>חומר למחשבה:</u> המעגל שתיארנו כאן הוא <u>בעומק</u> $O(n)$, כאשר <u>העומק</u> של מעגל הוא אורך המסלול המכוון הארוך ביותר בין קלט לפלט. האם תוכלו לחשוב על מעגל עם עומק משמעותית קטן יותר?

כפי שאמרנו, אנחנו חושבים על <u>הגודל</u> כמדד לזמן חישוב. באופן דומה, <u>עומק</u> המעגל הוא מדד לזמן חישוב המקבילי.

<u>פתרון שני (טקסט בהיר):</u>

$$\begin{array}{r} 1\,\mathbf{0}\,1\,0\,1\,1 \\ +\ 1\,\mathbf{0}\,1\,1\,1\,0 \\ \hline \end{array}$$

אם בעמודה מסוימת אני רואה שני אפסים, אז אני יודע שמהעמודה הזאת לא יהיה *carry*.
אם אני רואה *1,1* אז יהיה *carry*.
ואם אני רואה גם 0 וגם 1 אז אני לא יודע אם יהיה *carry*, אבל אני כן יודע להגיד שהעמודה הזאת "מעבירה *carry*".

עבור $1 \leq i \leq n$ נגדיר:

$g_i = (a_i \wedge b_i)$ הביטים במקום ה- i יוצרים carry
$p_i = (a_i \vee b_i)$ הביטים במקום ה- i מעבירים carry

עכשיו נוכל לרשום:

$$c_{i+1} = g_i \vee (p_i \wedge g_{i-1}) \vee (p_i \wedge p_{i-1} \wedge g_{i-2}) \vee \cdots \vee (p_i \wedge p_{i-1} \wedge \cdots p_2 \wedge p_1)$$

לכן אנחנו יכולים לבנות מעגל בעומק קבוע עם הצורה הבאה:

- קודם מחשבים את כל ה g_i, p_i "במקביל". כלומר לכל g_i ולכל p_i יש לי מעגל בעומק קבוע ואני שם את כל המעגלים האלה "אחד ליד השני"
- אח"כ אנחנו מחשבים את כל ה-c_i-ים במקביל
- לבסוף אנחנו מחשבים את כל ה-s_i-ים (כלומר את תוצאת החישוב) במקביל, כאשר כל s_i מחושב מתוך a_i, b_i, c_i כמו קודם.