5. מודלים אחרים

(לכל המדלג על קטעים של פרק זה, מומלץ לעיין בפיסקה המסיימת של הפרק).

בפרק הקודם ראינו שכוח החישוב של מכונת טיורינג איננו משתנה תחת וריאציות סבירות של המודל, במובן שפונקציה ניתנת לחישוב ע"י מכונה של מודל אחד אסם היא ניתנת לחישוב ע"י מכונה של המודל האחר. הוריאציות שסקרנו בפרק הקודם אינן ממצות את המספר הרב של וריאנטים שניתן למצוא בספרות, אך נראה לנו שהן ממצות את הענין בנושא. בפרק זה נעסוק במודלים סבירים אחרים של חישוב, ונראה כי הפונקציות הניתנות לחישוב על ידם, זהות לפונקציות הניתנות לחישוב ע"י מכונות טיורינג. המודלים שיוצגו בפרק זה, חבים את קיומם לגישות שונות לנסיון להגדרת חישוב.

5.1 המודל של מחשב גישה ישירה (Random Access Machine)

המוטיבציה להגדרה זו היא סכירת פיות "קטני-האמונה", הרגילים לזהות מחשב עם המודל של "גישה ישירה", אשר מכונת טיורינג הסדרתית היא לצנינים בעינהם. מי שלא בא מרקע של מדעי המחשב המעשיים, ימצא את האפולוגטיקה בסעיף זה מיותרת, וטוב יעשה אם ידלג עליה.

ראשית ראוי להגדיר מחשב גישה ישירה באופן מופשט. המילה מופשט באה לציין אך ורק את ההשתחררות מגדלים ספציפיים של רגיסטרים וזכרונות. מחשב גישה ישירה (מופשט) מורכב מתכנית סופית, אוסף אינסופי של תאי זכרון בעלי אורך אינסופי, מספר סופי של רגיסטרים בעלי אורך אינסופי ורגיסטר סופי מיוחד (הנקרא, Program Counter ומסומן PC) אשר ערכו מכתיב את שורת התכנית הבאה לביצוע. התכנית מכילה פקודות מהאוסף הבא:

- reset(r) - טען את רגיסטר r בערך 0.
- load(r_1,r_2) - טען את תוכן תא הזכרון המוצבע ע"י רגיסטר r_2, אל תוך רגיסטר r_1.
- store(r_1,r_2) - שמור את תוכן רגיסטר r_1, בתוך תא הזכרון המוצבע ע"י רגיסטר r_2.
- inc(r) - הגדל את התוכן של רגיסטר r ביחידה אחת.
- dec(r) - הקטן את התוכן של רגיסטר r ביחידה אחת.
- if r then goto l - אם תוכן רגיסטר r הוא אי-שלילי, טען את PC בקבוע l.

נציין כי לאחר ביצוע כל אחת מהפקודות הנ"ל מלבד האחרונה, מוגדל תוכן ה-PC ב-1. המודל נקרא מודל של גישה ישירה, משום שניתן לגשת אל תא הזכרון ה m-י מיד לאחר גישה אל התא ה n-י, ואין צורך לסרוק סדרתית את כל התאים שבינהם (כפי שהדבר מתבצע במכונת טיורינג). די בפקודות הנ"ל בכדי להוכיח שקילות המודל של גישה ישירה למודל של מכונת טיורינג, אולם מקובל להוסיף פקודות חזקות יותר מהסוג:

add(r_1,r_2) - חבר את תוכן הרגיסטרים r_1 ו-r_2 וטען את התוצאה ברגיסטר r_1.

mult(r_1,r_2) - הכפל את תוכן הרגיסטרים r_1 ו-r_2 וטען את התוצאה ברגיסטר r_1.

ענין זה של פקודות נוספות הוא עדין. כל זמן שהפקודות הנוספות הן "סבירות" (ז.א. מייצגות פונקציות הניתנות לחישוב) אין בעיה. אבל אסור להרשות במודל פקודות אשר מייצגות פונקציות שלא ניתנות לחישוב. תנאי זה (המתקיים כמובן במחשבים המעשיים) עושה את כל המודל מעגלי במובן הבא: זהו מודל סביר של חישוב בתנאי שהפעולות המוגדרות על הרגיסטרים שלו הן פעולות הניתנות לחישוב. חולשה זו של המודל, היא אחת הסיבות מדוע לא להשתמש בו לצרכי קורס זה ולמטרות דומות. אולם כל מה שרצינו לאמר הוא שהמודל הזה עם פעולות סבירות המתאימות לאלו שאנו מכירים ב"מחשבים אמיתיים", הוא שווה כוח למודל של מכונת טיורינג. לפני שנוכיח (למעשה לא נביא פה הוכחה מלאה, אלא נביא רק את דרך ההוכחה. ההוכחה המלאה תושאר כתרגיל לקורא) את הטענה נפנה את תשומת לב הקורא לעובדה שהאינסופיות של המודל אנלוגית לאינסופיות של מכונת טיורינג, ואיננה אלא אופן לדבר על גדול כנגד קטן מבלי להסתבך. כפי שנראה בהוכחת שקילות המודלים, מספר תאי הזכרון במודל RAM יתאים למספר התאים בסרט של מכונת טיורינג, וגודל הרגיסטרים (ותאי הזכרון) יהיה כזה שהם יוכלו להכיל כתובות בזכרון זה.

תרגיל 5.1: הראה שלכל מכונת טיורינג יש מחשב גישה ישירה השקול לה.

הדרכה: לכל אחד מתאי מכונת הטיורינג יותאם תא בזכרון של ה-RAM. אחד הרגיסטרים ב-RAM יכיל את מיקום הראש של המכונה לה עושים סימולציה, רגיסטר אחר יכיל את מצב המכונה. התכנית צריכה לאפשר את חישוב פונקצית המעברים δ. די מסורבל לממש זאת ללא פקודות נוספות כגון הזזה (shift) של תוכן רגיסטר או עיון בביט הראשון שלו. שים לב כי פקודות אלו ניתנות למימוש ע"י הפקודות האחרות. במילים אחרות, קל לכתוב בשפה עילית (כגון Pascal) תכנית המבצעת סימולציה של מכונת טיורינג. על תכנית זו ניתן להפעיל קומפיילר המתרגם משפה עילית לשפת ה-RAM (המצומצמת יותר שהגדרנו).

תרגיל 5.2: הראה כי לכל מחשב גישה ישירה (RAM) יש מכונת טיורינג השקולה לו.

הדרכה: הראה תחילה כי ניתן לבנות מחשב גישה ישירה שקול, אשר כל אחד מתאי הזכרון שלו הם מגודל סופי (נאמר מסוגל לאכלס אחד מתוך ארבע ערכים אפשריים). לשם כך הסתכל על הזכרון של המחשב המקורי כמערך אינסופי דו-מימדי של תאים סופיים, וממש אותו ע"י מערך אינסופי חד-מימדי. בצע סימולציה של מחשב גישה ישירה זה על גבי מכונת טיורינג רבת סרטים. סרט אחד יתאים לתוכן תאי הזכרון של ה-RAM וכל אחד משאר הסרטים יתאים לרגיסטר של ה-RAM. גישה לתא זכרון מסוים תמומש ע"י סריקה סדרתית של הסרט המכיל את תוכן תאי הזכרון (המכונה שנבנה צריכה להיות מסוגלת לספור, אולי בעזרת סרט עבודה נוסף). שים לב כי תוכן תאי הזכרון בכל שלב בחישוב, ניתן ליצוג ע"י מספר סופי של תאי הסרט של מכונת טיורינג.

5.2 אוטומט תאים אינסופי ומעבר לו

גישתנו בהגדרת מכונת טיורינג היתה מינימליסטית. ביקשנו לתפוס את המינימום הדרוש על מנת שנוכל לבצע חישוב כללי. ברצוננו לנסות לשכנע שגם בגישה מקסימליסטית סבירה, לא היינו מגיעים לכוח חישוב אחר. על מנת לבאר גישה זו נתחיל בדוגמא פשוטה. נסתכל על מערך אינסופי של תאים, אשר תכנם משתנה מדי רגע על פי תכנם ותוכן שכניהם. לשם פשטות נניח שכל השינויים מתבצעים יחדיו, במעין פעימה אחת. החוקים הקובעים את השינויים הם פשוטים ומהוים טבלה המגדירה את תכנו של התא ברגע הבא אם ברגע הנוכחי תכנו σ, תוכן שכנו השמאלי σ_L, ותכן שכנו הימיני σ_R. במילים אחרות, יש לנו פונקצית מעבר $f : S \times S \times S \to S$ כאשר S היא קבוצה סופית של ערכים אפשריים לכל אחד מהתאים. עבור התא הראשון אשר לו שכן אחד בלבד (נקרא לו השכן הימני), יש לנו פונקצית מעבר $f_1 : S \bigcup \{\#\} \times S \to S \bigcup \{\#\}$, כאשר $\# \notin S$. אם קיימת פעימה בה תוכן אף אחד מן התאים אינו משתנה, נאמר שהאוטומט האינסופי נמצא בקונפיגורציה עוצרת. באמצעות האוטומט מוגדרת העתקה של סדרות אינסופיות אל סדרות אינסופיות, כך שהסדרה $\vec{\alpha}$ מועתקת לסדרה $\vec{\beta}$, אם האוטומט המתחיל מקונפיגורציה $\vec{\alpha}$ נעצר בקונפיגורציה $\vec{\beta}$. תהי $\vec{\alpha}$ סדרה סופית. אם חישוב של האוטומט שתוכן תאיו הראשונים הוא $\#\alpha$ ומימינם כל התאים מכילים סימן מיוחד (נאמר $\not\#$) מסתיים בקונפיגורציה עוצרת, נגדיר את הפלט של האוטומט כתוכן התאים החל מהתא השני ועד (לא כולל) לתא שהחל ממנו תוכן כל התאים זהה. כעת יש טעם לדבר על פונקציות מעל מחרוזות סופיות הניתנות לחישוב ע"י אוטומטים אינסופיים אלו. די קל לראות שהן בדיוק הפונקציות הניתנות לחישוב ע"י מכונת טיורינג.

<u>תרגיל 5.3:</u> הראה כי לכל מכונת טיורינג קיים אוטומט תאים אינסופי אשר מחשב את אותה פונקציה.

<u>הדרכה:</u> תא האוטומט יכיל את תוכן תא הסרט המתאים וכן את מצב מכונת הטיורינג במידה והראש שלה נמצא בתא זה. פונקצית המעברים של האוטומט תהיה תלויה בפונקצית המעברים של מכונת הטיורינג ותגרום שינוי בתא רק באחד משני המקרים:

(1) הראש נמצא בתא זה (בתחילת הפעימה).

(2) הראש נמצא בתא השכן (בתחילת הפעימה) ונע לתא זה (בסופה).

תפקיד התא הראשון הוא להתחיל את החישוב. כלומר, בפעימה הראשונה, התא השני "יוסיף" לתכנו את q_0, והתא הראשון ישנה את ערכו מ-$\#$ (המסמן שזוהי פעימה ראשונה) לערך אחר.

<u>תרגיל 5.4:</u> הראה כי לכל אוטומט אינסופי ישנה מכונת טיורינג המחשבת את הפונקציה (על סדרות סופיות) שהוא מחשב.

<u>הדרכה:</u> שים לב כי הקלט של האוטומט האינסופי הינו $\#\alpha\not\#^*$, כאשר $\alpha \in S^*$. הסרט של מכונת הטיורינג יכיל את

תוכן התאים של האוטומט. כמו כן נסמן בכל רגע את גבול התפשטות השפעת הקלט על האוטומט. סימולציה של פעימה של האוטומט תתבצע ע"י סריקה של הסרט עד לגבול ההתפשטות, תוך רישום התוכן החדש לאחר הפעימה בערוץ (סרט) נוסף, הרחקת גבול ההתפשטות ביחידה אחת ימינה, וסריקה חוזרת בה ירשם התוכן החדש על גבי הישן. יש גם לשמור את תכנם של התאים שמעבר לגבול ההתפשטות, אך לשם כך מספיק תא אחד בלבד, כי לכולם תוכן זהה (אשר אינו תלוי בקלט α, אלא תלוי רק ב-$f(\sigma,\sigma,\sigma)$, עבור $\sigma \in S$).

באופן דומה להגדרת האוטומט הנ"ל (שיקרא **אוטומט תאים חד–מימדי**) ניתן להגדיר אוטומט תאים דו-מימדי. באוטומט זה, התאים מסודרים בצורת מערך דו-מימדי ובכל פעימה תוכן כל תא משתנה בהתחשב בתוכן שכניו (מימינו, משמאלו, מעליו ומתחתיו) ובתכנו בתחילת הפעימה. באופן כללי יותר, ניתן לחשוב על גרף אינסופי חסום-דרגה (כלומר, קיים קבוע, d, כל שדרגת כל צומת בגרף הינה d לכל היותר) כלשהו, שתוכן התאים (הנמצאים בצמתיו) משתנה בכל פעימה בהתחשב בתוכן התאים בצמתים השכנים. שים לב כי מבנה הגרף צריך להיות "פשוט" (כלומר, ניתן לתיאור סופי). ניתן להגדיר גרף אינסופי "פשוט", ע"י פונקציה, $g : N \to N$, הניתנת לחישוב כך ש-$g(v)$ היא קבוצת השכנים של צומת v (כמובן שיש לדרוש, $u \in g(v)$ אמם $v \in g(u)$).

<u>תרגיל למתקדמים:</u> הגדר אוטומט תאים של גרף (פשוט וחסום דרגה) המוגדר ע"י פונקציה, g, כנ"ל וכן הגדר חישוב של אוטומט כזה (על סדרות סופיות). הראה התאמה בין חישובי אוטומט זה, לבין חישובי מכונת טיורינג (שים לב למכשלות כגון: "אוטומט הגרף" יכול לבצע סימולציה למכונת טיורינג רק אם יש בו רכיב קשיר אינסופי. שים לב לחשיבות היות הפונקציה, g, ניתנת לחישוב, בטיעון שמכונת טיורינג יכולה לבצע סימולציה של "אוטומט הגרף". תאר תחילה את הסימולציה ע"י אלגוריתם).

צעד נוסף מעבר לכל המודלים הללו שהם בעלי מבנה סטטי (קבוע) הוא מודל המבוסס על גרף בו מבנה הגרף עצמו משתנה במהלך החישוב (גרף דינמי). נתאר את המודל הנקרא מכונת מצביעים (pointer machine) ע"י תיאור של קונפיגורציה רגעית והקונפיגורציה הבאה אחריה. קונפיגורציה רגעית במודל היא גרף מכוון סופי בעל דרגת יציאה חסומה. קשתות הגרף צבועות בצבעים מתוך קבוצה סופית של צבעים, כך שאין שתי קשתות היוצאות מאותו צומת אשר צבועות באותו צבע.

בצעד חישוב אחד מסוגל כל צומת לבצע אחת או יותר מן הפעולות הבאות:

(1) השמטת קשת היוצאת ממנו.

(2) החלפת צבע קשת יוצאת.

(3) בנית צומת חדש עם קשת בצבע מסויים המכוונת אליו.

(4) "נסיון" לבנות קשת בצבע מסוים אל צומת הנמצא במסלול מכוון באורך 2 היוצא מן הצומת וצבוע בצבעים מסוימים (במידה ומסלול כזה לא קיים הקשת לא נבנית).

(5) בנית קשת יוצאת, צבועה בצבע מסוים, לצומת (היחיד) ממנו נכנסת קשת צבועה בצבע מסוים (כלומר, בנית קשת אנטי-מקבילה).

(6) בנית קשת עצמית צבועה בצבע מסוים.

שים לב כי צמתי הגרף אינם מסומנים. הצומת מחליט אלו מהפעולות הנ"ל לבצע על פי צבעי הקשתות היוצאות ממנו ונכנסות אליו ותכנית קבועה (זהה לכל הצמתים) שפקודותיה מן הצורה: "אם קבוצת הצבעים של הקשתות היוצאות היא C_1 וקבוצת הצבעים של הקשתות הנכנסות היא C_2, השמט את הקשת היוצאת הצבועה בצבע ϕ_1 ($\phi_1 \in C_1$) ובנה צומת חדשה עם קשת בצבע ϕ_2 ($\phi_2 \notin C_1$) המובילה אליה". C_i היא תת-קבוצה של קבוצת הצבעים האפשריים $\{1,2,3,\ldots,d\}$, לכל זוג תתי-קבוצות כאלו תותאם לכל היותר פקודה אחת. דוגמא נוספת: "אם קבוצת הצבעים של הקשתות היוצאות היא C_1 וקבוצת הצבעים של הקשתות הנכנסות היא C_2, בנה קשת יוצאת בצבע ϕ_1 ($\phi_1 \notin C_1$) אל הצומת ממנה מגיעה הקשת הנכנסת (היחידה) הצבועה בצבע ϕ_2". דוגמא אחרונה: "אם קבוצת הצבעים על הקשתות היוצאות היא C_1 וקבוצת הצבעים על הקשתות הנכנסות היא C_2, בנה קשת יוצאת בצבע ϕ_0 אל הצומת שממנה מגיע מסלול באורך 2, שהקשת הראשונה בו צבועה ϕ_1 והשניה צבועה ϕ_2". הקונפיגורציה ההתחלתית של מכונת מצביעים היא מסלול מכוון סופי (עם צבעים על קשתותיו) ולולאה עצמית על הצומת הראשון במסלול. סדרת הצבעים של המסלול תקרא הקלט של המכונה (ראה ציור 5.1). הקונפיגורציה הסופית של המכונה היא קונפיגורציה אשר אינה משתנה בצעד חישוב נוסף. אם הקונפיגורציה הסופית מכילה מסלול פשוט מכוון יחיד הכולל את כל צמתי הגרף (יתכנו מסלולים נוספים שאינם כוללים את כל הצמתים) נגדיר את הפלט כסדרת הצבעים שעל המסלול (ראה ציור 5.2). בכל מקרה אחר, נגדיר את הפלט (של הקונפיגורציה הסופית) כריק.

<u>תרגיל 5.5:</u> הראה כי לכל מכונת טיורינג קיימת מכונת מצביעים אשר מחשבת את אותה פונקציה.

<u>פתרון:</u> מכונת המצביעים תבצע סימולציה של חישוב של מכונת טיורינג (ללא הגבלת הכלליות נניח שאינה עומדת במקום, שאינה עוצרת ע"י נפילה שמאלה מן הסרט ושאינה עוצרת במצב שאינו מצב מקבל) כאשר לכל $\sigma \in \Gamma$ נתאים צבע σ. בצעד האתחול מכונת המצבים תעבור ממסלול מכוון למסלול בעל קשתות אנטי-מקבילות הצבועות בצבע מיוחד, וכן תצבע את הלולאה העצמית בצבע q_0 (ראה ציור 5.3).

ציור 5.1: קונפיגורציה התחלתית עם קלט 0111.

ציור 5.2: קונפיגורציה סופית עם פלט 10.

ציור 5.3: צעד האתחול של מכונת המצביעים המבצעת סימולציה על קלט 011.

לאחר סיום צעד האתחול, הקונפיגורציה של מכונת המצביעים מקדדת את הקונפיגורציה ההתחלתית של מכונת טיורינג. הצעדים הבאים יהיו צעדי סימולציה, כאשר בכל שני צעדים מבצעים צעד יחיד של מכונת טיורינג המקורית. בצעד הסימולציה הראשון בודק כל צומת עבור כל $q \in Q \setminus F$ ולכל $\sigma \in \Gamma$, האם קבוצת הצבעים של הקשתות היוצאות ממנו מכילה את $\{q,\sigma\}$. במידה וכן, קיימים שני מקרים:

(1) $\delta(q,\sigma) = (p,\tau,R)$: הצומת מוחק את הקשת הצבועה q, ומחליף את צבע הקשת הצבועה σ לצבע (p,τ).

(2) $\delta(q,\sigma) = (p,\tau,L)$: הצומת מוחק את הקשת הצבועה q, מחליף את צבע הקשת הצבועה σ לצבע τ ואת צבע הקשת הצבועה "אחורה", לצבע "אחורה, p".

אם קבוצת הצבעים של הקשתות היוצאות מכילה צבע של מצב q, אך אינה מכילה אף צבע של אות σ, נפריד שוב לשני מקרים:

(1) $\delta(q,\text{ב}) = (p,\tau,R)$: הצומת מוחק את הקשת הצבועה q, ובונה קשת צבועה בצבע (p,τ) לצומת חדש.

(2) $\delta(q,\text{ב}) = (p,\tau,L)$: הצומת מוחק את הקשת הצבועה q, ובונה קשת צבועה בצבע τ לצומת חדש ואת צבע הקשת הצבועה "אחורה", מחליף לצבע "אחורה", p".

בצעד השני, צומת שקבוצת הצבעים של הקשתות היוצאות שלו מכילה (p,τ) מחליף את הקשת המתאימה לקשת הצבועה בצבע τ, וצומת שקבוצת הצבעים של הקשתות היוצאות שלו מכילה "אחורה, p" מחליף את הקשת המתאימה לקשת הצבועה בצבע "אחורה". צומת שקבוצת הצבעים של הקשתות הנכנסות שלו מכילה (p,τ) או "אחורה, p", בונה לולאה עצמית הצבועה בצבע p. צומת שקבוצת הצבעים של הקשתות הנכנסות שלו מכילה צבע אחד בלבד, השונה מהצבע "אחורה", בונה לה קשת אנטי-מקבילה הצבועה בצבע "אחורה".

<u>תרגיל 5.6:</u> הראה כי ניתן לממש החלפת צבע קשת ע"י שאר הפעולות המותרות במכונת מצביעים (רמז: השתמש בכללים (6) ו- (4)).

<u>תרגיל 5.7 (למתקדמים):</u> נגדיר את הפונקציה $f(x) \triangleq x'$ כאשר x' הוא הפרפיקס הארוך ביותר של x שארכו הוא חזקה של 2. הראה כי מכונת מצביעים מסוגלת לחשב את הפונקציה f במספר צעדים לוגריתמי באורך הקלט. מה מספר הצעדים המינימלי הנחוץ למכונת טיורינג על מנת לחשב פונקציה זו?

<u>תרגיל 5.8:</u> הראה כי לכל מכונת מצביעים קיימת מכונת טיורינג אשר מחשבת את אותה פונקציה.

<u>הדרכה:</u> החזק את הקונפיגורציה הרגעית של מכונת המצביעים כגרף מכוון עם צבעי קשתות. מעבר מקונפיגורציה רגעית אחת לקונפיגורציה רגעית אחרת היא פונקציה מסורבלת להגדרה אך לא קשה לחישוב. לשם האינטואיציה שכנע את עצמך כי הינך מסוגל לכתוב תכנית מחשב שתבצע סימולציה של מכונת מצביעים, ולפי סעיף 5.1 קיימת גם מכונת טיורינג כזו.

ראינו שקילות של מכונת טיורינג למכונת מצביעים, ומה יכול להיות "מקסימליסטי יותר" מאשר מחשב אשר כל אחד ממעבדיו יכול להתקשר בכל צעד עם מספר קבוע של מעבדים אחרים ולפעול על פי תכנם, ליצור ולהרוס מעבדים וכו...

זה המקום לחזור על התזה של צ'רץ': כל המודלים הסבירים של חישוב, בעלי כוח חישוב שווה. כלומר, כל מודל של חישוב אשר מרשה מצד אחד לחשב פונקציות שברור כי הן ניתנות לחישוב (ובפרט פונקציות הניתנות לחישוב ע"י מכונת טיורינג) ואינו מרשה פעולות פלאיות (שחוקיותן אינה ניתנת לתיאור סופי - ראה מבוא) מאפשר לחשב בדיוק את אותן פונקציות הניתנות לחישוב ע"י מכונת טיורינג.