פתרונות נבחרים לחלק ראשון

<u>פרק רקע טכני:</u>

<u>תרגיל (1):</u> לאחר שראינו כי השפה Σ^* היא בת-מניה, נוכל להגדיר סדר על המילים ב- Σ^* (ובפרט לכל שפה L מעל הא"ב Σ שהיא תת-קבוצה של Σ^*) כדלקמן: יהיו $\alpha_1,\alpha_2 \in \Sigma^*$ ותהא f פונקציה חד-חד ערכית ועל מ-Σ^* לטבעיים (ראינו כבר כי קיימת f כזאת). נגדיר $\alpha_1 < \alpha_2$ אם $f(\alpha_1) < f(\alpha_2)$. מאחר ש-$L$ היא כאמור תת-קבוצה של Σ^* היא גם בת-מניה ונוכל "למנות" אותה בצורה האינדוקטיבית הבאה:

I. המילה הראשונה בשפה L, תהיה המילה הקטנה ביותר ב- L (לפי היחס "<").

II. המילה ה n-ית ב-L תהיה המילה הקטנה ביותר ב-L לאחר שהוציאו ממנה את $n-1$ המילים הראשונות.

כלומר, קיבלנו כי גם השפה L הינה בת-מניה.

<u>תרגיל (2):</u> נראה כי מחלקת כל השפות האינסופיות (מעל א"ב Σ) אינה בת-מניה, ע"י שנראה העתקה חח"ע ועל בין קבוצת השפות האינסופיות לבין המספרים הממשיים בקטע $[0,1]$ (שמספרם אינו בן-מניה) ביצוגם הבינארי. השפה האינסופית L תועתק למספר הממשי α, כאשר הספרה ה i-ית אחרי הנקודה במספר α תהיה 1 אם המילה ה i-ית בסדר הלקסיקוגרפי ב- Σ^* שייכת לשפה L. העתקה זו חח"ע, מכיוון שעבור כל שתי שפות שונות קיימת מילה הנמצאת רק באחת מהן, נניח מילה זו היא ה i-ית בסדר הלקסיקוגרפי, אז המספרים בקטע $[0,1]$ המותאמים לשתי שפות אלו שונים בספרה ה i-ית אחרי הנקודה. העתקה זו היא גם על, מכיוון שלכל מספר בקטע $[0,1]$ ביצוגו הבינארי המכיל מספר אינסופי של 1-ים (גם למספר בעל ייצוג של מספר סופי של 1-ים, יש ייצוג בעל מספר אינסופי של 1-ים. לדוגמא, המספר $\frac{1}{2}$ מיוצג ע"י $0.01111111\cdots$) ניתן להתאים את השפה המכילה בדיוק את המילים המתאימות ל 1-ים שבייצוג זה.

<u>הערה:</u> הטיעון תופס אפילו אם Σ מכילה אות אחת בלבד.

<u>תרגיל (3):</u> מחלקת כל השפות הסופיות היא בת-מניה. נגדיר עבור כל n טבעי קבוצת שפות (סופית) S_n כדלקמן:

$$S_n = \{\ n\ \text{קבוצת כל השפות שהמילה הארוכה ביותר בהן היא באורך}\ \}$$

מכיוון שלכל n טבעי, מספר המילים באורך n לכל היותר, הוא סופי (לכל היותר $\sum_{i=0}^{n}|\Sigma|^i$) אז מספר תתי-הקבוצות של מילים אלו (כלומר, שפות המורכבות ממילים אלו) הינו סופי, ולכן S_n (שהיא קבוצת כל השפות הנ"ל) היא סופית. לכן, נוכל "למנות" את אברי S_n בשלבים. בשלב ה i-י, נמנה את כל השפות ב- S_i (שהיא כאמור סופית לכל i). בכל שפה סופית L ישנה מילה ארוכה ביותר (נאמר באורך m) ולכן $L \in S_m$ ונמנה את L בשלב ה m-י. במילים אחרות, מחלקת השפות הסופיות היא איחוד של מספר בן מניה של קבוצות סופיות, ולכן גם היא בת מניה.

פרק 2

תרגיל 2.1: בנית M':

$$M' = \{Q', q_0', F', \Gamma', \Sigma, \flat', \delta'\}$$
$$Q' = Q \cup \{q_0'\}$$
$$F' = F \cup \{f'\}$$
$$\Gamma' = \Gamma \cup \{\sigma' : \sigma \in \Gamma\}$$

פונקצית המעברים δ' תהיה הרחבה של פונקצית המעברים δ, ותכיל בנוסף את המעברים הבאים:

$$\{\delta'(q_0', \sigma) = (q_0, \sigma', S) : \forall \sigma \in \Sigma\}$$
$$\{\delta'(q, \sigma') = (p, \tau', R) : \delta(q, \sigma) = (p, \tau, R)\text{, עבור כל מעבר ב–}\delta\text{ מן הצורה}\}$$
$$\{\delta'(q, \sigma') = (p, \tau', S) : \delta(q, \sigma) = (p, \tau, S)\text{ עבור כל מעבר ב–}\delta\text{ מן הצורה}\}$$
$$\{\delta'(q, \sigma') = (f', \tau', S) : \delta(q, \sigma) = (p, \tau, L)\text{, עבור כל מעבר ב–}\delta\text{ מן הצורה}\}$$

באופן מילולי, המכונה M' מסמנת את התא הראשון (תוך כדי שמירת אינפורמציה לשחזור ערכו). המכונה M', אינה מאפשרת צעד שמאלה מהתא הראשון (המכיל אות מתוך $\Gamma \setminus \Gamma'$). במקרה בו M היתה נופלת שמאלה מהסרט, M' עוברת למצב f' ועוצרת במקום (כלומר, הפלט הוא המילה הריקה כנדרש) ולכן $L(M) = L(M')$ (למען הדיוק, יש גם צורך להוסיף, במקרה של עצירה "נורמלית" של המכונה, גם: סימון המקום בו עצרה המכונה, הליכה לתחילת הסרט לשם החלפת התא הראשון לערך המתאים מתוך Γ וחזרה למקום העצירה. השלם את הפרטים בעצמך!).

תרגיל 2.2: תהי M מכונה שאינה נופלת אף פעם מהסרט (אחרת, לפי תרגיל 2.1 יש לה מכונה שקולה כזאת). נבנה מכונה שקולה M' כך ש- $L(M) = L(M')$, וכן הקונפיגורציות הסופיות אליהן היא מגיעה בסוף החישוב הן מהצורה $\beta p \flat^*$, כאשר $p \in F$.

המכונה M' תפעל כמו M, פרט לכך שיהיה לה תו נוסף, $\flat'$, אשר היא תדאג במשך כל החישוב, שכל ה $\flat$-ים שהיא רשמה אשר נמצאים מימין לראש הקורא/כותב, יוחלפו ב- $\flat'$-ים, ו $\flat$-ים שמשמאל לראש הקורא/כותב ישארו כ $\flat$-ים. עם סיום החישוב, M' תמחק (ע"י הדפסת $\flat$-ים משמאל לימין) את תוכן הסרט שמימין לראש עד ה- $\flat$ הראשון, ותחזור לקצה הפלט.

בנית M':

$$M' = \{Q', q_0, F', \Gamma', \Sigma, \flat', \delta'\}$$
$$Q' = Q \cup \{q_f, q_R, q_L\}$$
$$\Gamma' = \Gamma \cup \{\flat'\}$$
$$F' = \{q_f\}$$

בנית δ' תעשה בשלבים:

(1) $\delta \leftarrow \delta'$ (בשלבים הבאים תשונה δ' מספר פעמים בשלבים).

(2) כל חוק ב-δ' מהצורה $\delta'(q,\sigma) = (p,\not b,L)$, יוחלף בחוק $\delta'(q,\sigma) = (p,\not b',L)$ (נבדיל בין $\not b$ מקורי ל-$\not b$ שנכתב ע"י המכונה ונמצא מימין לראש הקורא/כותב).

(3) לכל חוק מהצורה $\delta(q,\not b) = (p,\sigma,L/R/S)$, נוסיף חוק $\delta'(q,\not b') = (p,\sigma,L/R/S)$ (נגרום למכונה שלא "להבדיל" בין $\not b$ לבין $\not b'$).

(4) כל חוק ב-δ' מהצורה $\delta'(q,\sigma) = (p,\not b',R)$, יוחלף בחוק $\delta'(q,\sigma) = (p,\not b,R)$ (נדאג לכך שלא יהיו $\not b'$-ים משמאל לראש הקורא/כותב).

(5) בנוסף:

לכל $f \in F$ ולכל $\sigma \in \Gamma' \setminus \{\not b\}$	נוסיף את החוק	$\delta(f,\sigma) = (q_R,\sigma,S)$	(q_R מעבר למצב)
לכל $f \in F$	נוסיף את החוק	$\delta(f,\not b) = (q_f,\not b,S)$	(אין צורך במחיקת $\not b'$-ים)
לכל $\sigma \in \Gamma' \setminus \{\not b\}$	נוסיף את החוק	$\delta(q_R,\sigma) = (q_R,\not b,R)$	(מחיקה מימין לפלט)
	נוסיף את החוק	$\delta(q_R,\not b) = (q_L,\not b,L)$	(סיום מחיקה מימין)
	נוסיף את החוק	$\delta(q_L,\not b) = (q_L,\not b,L)$	(חזרה שמאלה לקצה הפלט)
ולכל $\sigma \in \Gamma' \setminus \{\not b\}$	נוסיף את החוק	$\delta(q_L,\sigma) = (q_f,\sigma,R)$	(חזרה למקום העצירה)

<u>תרגיל 2.3:</u> פעולת המכונה המחשבת את הפונקציה shift:

(1) המכונה תסמן לעצמה את תחילת הקלט.

(2) המכונה תנוע ימינה עד לסוף הקלט, תזהה את σ_n ותזכור אותה במצב המכונה, q_{σ_n} (כל אינפורמציה <u>סופית</u> ניתן לזכור במצב מכונה, ע"י הוספת מספר מצבים <u>סופי</u> למכונה).

(3) המכונה תנוע שמאלה עד לתחילת הקלט, תדפיס σ_n ותזכור את σ_1.

(4) המכונה תנוע ימינה עד לסוף הקלט, כאשר בכל צעד תדפיס בתא ה-i את σ_{i-1} ותזכור (בעזרת מצב המכונה) את σ_i.

(5) כאשר המכונה תגיע לתא הראשון המכיל $\not b$ היא תעצור במקום.

<u>תרגיל 2.4:</u> חישוב הפונקציה succ, במקרה בו אורך המילה העוקבת אינו גדל (כלומר, $|\mathrm{succ}(x)| = |x|$) זהו למעשה חיבור של 1 למספר הנתון בבסיס Σ, והחישוב יעשה במעבר על הקלט מימין לשמאל (שים לב כי יש להמשיך במעבר עד אשר אין carry). במקרה בו אורך המילה העוקבת גדל ב-1 המילה העוקבת מכילה כולה את האות הראשונה ב-Σ (אפסים במקרה הבינארי).

להלן פרטי פעולת המכונה המחשבת את הפונקציה succ, על קלט השייך לא"ב Σ:

(1) המכונה תסמן את תחילת הקלט בסימון מיוחד.

(2) המכונה תזוז ימינה עד לתא המכיל את התו הימני ביותר בקלט.

(3) עבור האות σ, שעליה ניצב הראש הקורא/כותב, המכונה תחליפה באות "העוקבת מודולו Σ" ל-σ, שנסמנה ב-τ (שים לב לשני מקרים: אם σ אינה האות הגדולה ביותר ב-Σ אז τ הינה האות העוקבת לה, בעוד שאם σ הינה האות האחרונה ב-Σ אז τ הינה האות הראשונה ב-Σ. לדוגמא: אם $\Sigma=\{0,1,2,3\}$, אז האות העוקבת של 1 היא 2 בעוד האות העוקבת של 3 היא 0).

(3.1) אם τ אינה האות הראשונה ב-Σ, המכונה תחזור לתא הראשון בסרט, תמחוק את הסימון המיוחד, תחזור לסוף הקלט ותעצור (החישוב הסתיים).

(3.2) אם τ הינה האות הראשונה ב-Σ והמכונה אינה עומדת על התא הראשון בסרט (המסומן בסימון מיוחד) המכונה תזוז תא אחד שמאלה בסרט ותחזור לבצע את שלב (3).

(3.3) אחרת (זהו התא הראשון בסרט ו-τ הינה האות הראשונה ב-Σ) המכונה תחזור ימינה תוך כדי החלפת כל האותיות בדרך עד ה-$\not b$ הראשון (כולל) באות הראשונה ב-Σ ותעצור (זהו המקרה בו $(|\alpha| < |succ(\alpha)|$).

<u>תרגיל 2.5:</u> פעולת המכונה המחשבת את הפונקציה order (משלב זה ואילך, נרשה לעצמנו לוותר על תאור סימון התא הראשון בתחילת החישוב ומחיקתו עם סיום החישוב ונניח כי פעולה זו נעשית):

(1) המכונה תשווה את ארכי הקלטים (ההשוואה תעשה באיטרציות, כאשר בכל איטרציה תסומן אות נוספת בכל אחד משני הקלטים בצורה דומה לזו שסימנו את התא הראשון) אם אחד מהם ארוך יותר, הוא גם גדול יותר והמכונה תענה בהתאם (ע"י חזרה לתחילת הקלט וכתיבת התשובה המתאימה בתא הראשון בסרט).

(2) עבור קלטים באורכים שווים, המכונה תעבור באיטרציות על שני הקלטים משמאל לימין (כאשר בכל איטרציה נסרקת אות נוספת מכל אחד מהקלטים) ותשווה בכל איטרציה בין שני הערכים הנוכחים. אם התגלה שוני בין שני הערכים, הקלט המכיל את הערך הגדול יותר הוא גם גדול יותר והמכונה תענה בהתאם. אם הערכים שווים המכונה תעבור לאיטרציה הבאה. אם נגמרו כל האיטרציות (הקלטים נסרקו עד סופם) ולא נתגלה שוני, הקלטים שווים והמכונה תענה בהתאם.

<u>תרגיל 2.6:</u> כדי לחשב את הפונקציה index, כל מה שיש לעשות הוא להוסיף 1 בתחילת הקלט (בדוק כי טענה זו נכונה). לכן, כל מה שעל המכונה לעשות הוא לכתוב 1 בתא הראשון ולבצע הזזה אחת ימינה של הקלט המקורי בדומה לביצוע הפונקציה shift מתרגיל 2.3.

<u>תרגיל 2.7:</u> בכדי לחשב את הפונקציה ההפוכה, יש לבצע הזזה אחת שמאלה של הקלט המקורי. שים לב כי במקרה בו הקלט אינו מתחיל באות 1 הפונקציה אינה מוגדרת ולכן המכונה אינה עוצרת. ניתן לעשות זאת ע"י ריצה

אינסופית ימינה למשל.

פרק 3

תרגיל 3.3:

א. כפי שהגדרנו צעד חישוב בהגדרה 2.3, אנו רואים כי כל קונפיגורציה במהלך החישוב, אשר במעבר אליה לא ביקרה המכונה בתא חדש, שווה בגודלה לקונפיגורציה הקודמת לה, ואילו קונפיגורציה אשר במעבר אליה מבקרת המכונה בתא חדש (כלומר, תא מימין לקלט המקורי שהמכונה עדיין לא ביקרה בו) גדולה ב-1 מקודמתה. הנקודה העיקרית היא שקונפיגורציה אף פעם אינה קטנה מקודמתה כי היא מכילה גם את תוכן התאים המכילים $\flat$ אשר המכונה ביקרה בהם (אם קיימים כאלו). לכן, באופן אינדוקטיבי, כל ביקור בתא חדש מגדיל את הקונפיגורציה ומכיוון שהמכונה מבקרת בכל התאים המכילים את הקלט שלה, גודל הקונפיגורציה הסופית הוא כמספר התאים בהם ביקרה המכונה בנוסף למצבה (מכאן בא ה-1–).

ב. המכונה M' תפעיל תחילה את המכונה M בצורה שתאפשר להבדיל בין תו $\flat$ מקורי, לתו $\flat$ שהודפס ע"י המכונה במהלך החישוב, כלומר תא בו ביקרה המכונה. בנוסף, המכונה M' תדאג לסמן את התא הראשון, על מנת למנוע נפילה מהסרט. כאשר המכונה M תגיע למצב סופי או כאשר המכונה M תעמוד על התא השמאלי ביותר ותצטרך (לפי החישוב המקורי) לצעוד שמאלה, תדפיס M' את התו 1 במקום בו M עצרה (עבור המקרה בו M עצרה על $\flat$ ולא החליפה אותו ל- $\flat$ החדש). לאחר מכן תכנס M' למצב חדש בו היא תלך שמאלה (אם צריך) עד לתא הראשון וממנו תחזור ימינה (תוך כדי הדפסת 1-ים בדרך) עד שתגיע ל- $\flat$ הראשון בו המכונה M לא בקרה. לסיום, תזוז M' צעד אחד שמאלה (עבור התא שסימנה) ואז תעצור. שים לב שבמקרה בו M אינה עוצרת, גם M' לא תעצור.

פרק 4

תרגיל 4.1: תהא M מכונה בעלת k סרטים, כאשר סרט מספר 1 משמש לקלט וסרט מספר k משמש לפלט. נבנה מכונה M' בעלת $k+2$ סרטים אשר בסרט הקלט שלה (סרט מספר 1) ובסרט הפלט שלה (סרט מספר $k+2$) הפעילות מוגבלת לפי התרגיל. המכונה M' מעתיקה את סרט מספר 1 שלה לסרט מספר $k+1$ שלה (ההעתקה נעשית במעבר אחד ימינה על הקלט, ויותר לא יעשה שימוש בסרט הקלט) ומפעילה את M על הסרטים $k+1,2,3,\ldots,k$ במקום הסרטים $1,2,3,\ldots,k$ בהתאמה. עם תום החישוב, מעתיקה M' את סרט הפלט של M', לסרט הפלט שלה (סרט מספר $k+2$).

<u>תרגיל 4.2:</u> נבצע סימולציה של פעולת מכונה, M, בעלת k הסרטים, בעזרת מכונה M בעלת סרט יחיד המחולק (מחשבתית) ל-$2k$ ערוצים שתפקידם מתואר בהדרכה. עבור כל צעד חישוב של המכונה M, תחזור M' לתחילת הסרט שלה, תעבור עליו משמאל לימין (עד שתעבור את כל k הראשים) ובחזרה לתא הראשון. במעבר משמאל לימין, M' "תזכור" (בעזרת מצבי המכונה) את האותיות הנמצאות תחת k ראשי המכונה. בשלב זה המכונה M' "יודעת" (שוב בעזרת מצביה) איזה חוק יש להפעיל מחוקי M. כעת, M' תחזור שמאלה (אל התא הראשון בסרט) כאשר בכל פעם "שיתגלה" ראש קורא (אחד או יותר) על אחד הערוצים, M' תבצע את השינויים הנדרשים בתא זה ובתאים השכנים לו על-פי פונקצית המעברים של M. שים לב כי ניתן לזכור את האותיות שתחת k הראשים בעזרת מצבי המכונה, מכיוון ש-k הינו קבוע אשר הינו ידוע מראש. כמו כן "ידיעת" חוקי M בעזרת מצבי M' אפשרית מכיוון שמספר החוקים ב-M הינו סופי. יש כמובן צורך בתחילת החישוב להפוך את הסרט לסרט המחולק ל-$2k$ ערוצים, אשר ערוצו הראשון מכיל את הקלט וכל הראשים נמצאים על התא הראשון בערוצם. כמו כן, עם סיום החישוב יש לחזור לתחילת הקלט, להחליף תוכן כל תא בתוכן הערוץ האחרון עד למקום הראש ולעצור.

<u>תרגיל 4.3:</u> המכונה תשתמש בשלושה סרטים ותעבוד באיטרציות שמספרן הוא כאורך הקלט (הניתן ביצוג בינארי על הסרט הראשון). המעבר על הקלט יעשה משמאל לימין החל מהתא הראשון בסרט. בכל איטרציה תבדוק המכונה את תכנו של התא שעליו עומד הראש בסרט הראשון. אם תו זה שונה מ-$\not b$, המכונה תבצע הכפלה ב-2 (בייצוג אונרי) של תוכן הסרט השני (שהינו ריק בתחילת החישוב). לאחר מכן, המכונה תשרשר לסרט השני 1 אמם תוכן התא שעליו נמצא הראש הראשון הוא 1, ותזוז צעד אחד ימינה על הסרט הראשון. שים לב כי הכפלה ב-2 בייצוג אונרי, היא שרשור התוכן הנוכחי לעצמו. ההכפלה תעשה ע"י העתקת תוכן הסרט השני לסרט השלישי, והעתקתו חזרה מימין לסרט השני. כאשר יתגלה התו $\not b$ על הסרט הראשון, המכונה תעתיק את תוכן הסרט השני אל הסרט השלישי (סרט הפלט) ותעצור. שים לב כי למעשה השתמשנו פה בשיטת Horner (המשמשת לחישוב פולינומים).

<u>תרגיל 4.4:</u> המכונה תשתמש בשני סרטים ותעבוד באיטרציות שמספרן הוא כאורך הקלט (הניתן ביצוג אונרי על הסרט הראשון). בכל איטרציה יוגדל תוכן הסרט השני (שיאותחל ע"י המכונה ל-0) ב-1 ביצוג בינארי, והראש על הסרט הראשון יזוז תא אחד ימינה. כאשר הראש הראשון יגיע לקצה הקלט, המכונה תעצור. שים לב כי הגדלת תוכן הסרט השני ב-1, נעשית בצורה דומה לחישוב הפונקציה succ (מתרגיל 2.4) המחזירה את המחרוזת העוקבת בסדר הלקסיקוגרפי הקנוני, פרט לכך שבמקרה בו $|\alpha| < |\mathrm{succ}(\alpha)|$ (מקרה (3.3)) יש לשנות את תוכן התא הראשון (בסרט השני) מ-0 ל-1.

<u>תרגיל 4.6:</u> המכונה שאינה עומדת במקום תבנה מהמכונה המקורית, כאשר קבוצת המצבים תהיה: $Q' = Q \bigcup \{q' : q \in Q\}$, ואילו פונקצית המעברים δ תהיה גם היא מורכבת מ-δ המקורית ע"י החלפת כל חוק מהצורה $\delta(q,\sigma) = (p,\tau,S)$ בחוק $\delta(q,\sigma) = (p',\tau,R)$, וכמו כן, לכל $p \in Q$ ולכל $\tau \in \Gamma$, יוסף החוק $\delta(p',\tau) = (p,\tau,L)$.

פרק 7

תרגיל 7.1: תהי M מכונת טיורינג אשר F היא קבוצת המצבים הסופיים שלה. נחליף כל חוק מהצורה, $\delta(q,\sigma)=(p,\tau,D)$ (כאשר $D\in\{L,R,S\}$ ו- $p\in F$) בחוק $\delta(q,\sigma)=(p',\tau,D)$ כאשר $p'\in\{q_{ACC},q_{REJ}\}$ (בחירת q_{ACC} או q_{REJ} היא שרירותית בלבד). קל לראות כי המכונה החדשה שנוצרה מחשבת בדיוק את אותה פונקציה כמו M כי אין חשיבות לשם המצב בו היא עוצרת.

תרגיל 7.2: ללא הגבלת הכלליות, נניח כי M אינה נופלת מסרט העבודה במהלך החישוב. המכונה M' תפעל בדיוק כמו המכונה M, אך בסוף חישוב שהיה אמור לעצור במצב q_{REJ} היא תמשיך בריצה אינסופית ימינה ולא תעצור. פורמלית, כל חוק מהצורה, $\delta(q,\sigma)=(q_{REJ},\tau,D)$ (כאשר $D\in\{L,R,S\}$) יוחלף בחוק $\delta(q,\sigma)=(q_R,\tau,R)$ (כאשר q_R הינו מצב חדש) ולכל $\sigma\in\Gamma$ יתווסף החוק, $\delta(q_R,\sigma)=(q_R,\sigma,R)$.

תרגיל 7.4: המכונה $\overline{M}$ שמכריעה את $\overline{L}$, בהנתן קלט x, תריץ את המכונה M שמכריעה את L על הקלט x. אם $\overline{M}$ דחתה את x אז M תקבל אותו, ואם $\overline{M}$ קיבלה את x אז M תדחה אותו (שים לב כי מ- $L\in\mathbf{R}$ נובע כי בכל מקרה $\overline{M}$ עוצרת).

תרגיל 7.5: המכונה המכריעה את L, בהנתן קלט x, תריץ "במקביל" את M (המכונה המקבלת את L) ואת $\overline{M}$ (המכונה המקבלת את $\overline{L}$) על הקלט x. אם M תקבל את הקלט אז גם המכונה המכריעה תקבל אותו, ואם $\overline{M}$ תקבל את הקלט אז המכונה המכריעה תדחה אותו (שים לב כי בכל מקרה אחת המכונות חייבת לקבל את הקלט).

תרגיל 7.6: המכונה M' המקבלת את השפה $\{x\#y : (x,y)\in R\}$, תשתמש במכונה M המובטחת (מהגדרה 7.6) מהיות בעיית החיפוש של R ניתנת לפתרון, אשר מוצאת לכל x את ה-y (היחיד) המתאים לו. בהנתן קלט $x\#y$, המכונה M' תריץ את M על x ואם התקבל כי $y=M(x)$ אז M' תקבל את הקלט $x\#y$ (אחרת היא תדחה אותו). מכיוון שקיים רק y יחיד המקיים $(x,y)\in R$, מתקיים: $y=M(x)$ אסם $(x,y)\in R$.

תרגיל 7.16: יהי A אנומרטור לשפה L, ותהי s המילה הראשונה ש-A מייצר. מכונת העוקב M, המשרה את L על הגרעין s, בהנתן קלט x, מפעילה את האנומרטור A עד אשר הוא מדפיס את המילה x ואת המילה הבאה אחריו, y, ואז מחזירה את המילה y כפלט.

יהי $s\in L$ ותהי M מכונת עוקב המשרה שפה L על הגרעין s. האנומרטור המייצר את השפה L רושם תחילה על הסרט את המילה s לאחר מכן את $M(s)$, את $M(M(s))$ וכו... כלומר, כל מילה מתקבלת ע"י הפעלת מכונת העוקב M על המילה הקודמת לה.

תרגיל 7.17: מכונת העוקב M_s תשתמש באנומרטור A (המובטח מתרגיל 7.16) ותחזיר, על קלט x את המילה

העוקבת ל-x שידפיס האנומרטור, פרט לשני מקרים חריגים:

1. מילת הקלט היא s - מכונת העוקב תחזיר את המילה s_0.
2. המילה העוקבת ל-x היא s - מכונת העוקב תחזיר את המילה העוקבת ל-s שידפיס האנומרטור.

<u>תרגיל 7.18:</u> מצב זה לא יתכן. נניח בשלילה כי ישנה מכונת עוקב M, המשרה שפה L על שני גרעינים שונים s_1 ו-s_2. מהגדרה 7.18 נקבל כי קיימים n ו-m כך ש- $s_1 = f_M^{(n)}(s_2)$ ו- $s_2 = f_M^{(m)}(s_1)$ וקבלנו בסתירה להגדרה 7.16 כי: $s_1 = f_M^{(m+n)}(s_1)$.

<u>תרגיל 7.19:</u> השתמש בדיוק בהוכחת תרגיל 7.16 תוך כדי שימוש באנומרטור קנוני ובמכונת עוקב מונוטונית. שים לב כי s היא המילה הקטנה ביותר בסדר הלקסיקוגרפי השייכת לשפה L.

<u>תרגיל 7.20:</u> תהי $L \in \mathbf{RE}$. נבנה אנומרטור קנוני, A', המייצר את L'. לשם כך נשתמש באנומרטור, A, המייצר את L (שקיומו מובטח מתרגיל 7.14). האנומרטור A', יפעיל את האנומרטור A. המילה הראשונה שייצר A תיוצר גם כן ע"י A'. לאחר מכן (בלולאה אינסופית) A' יפעיל את A עד אשר A ייצר מילה הגדולה בסדר הלקסיקוגרפי מהמילה האחרונה שיצר A'. האנומרטור A' ייצר מילה זו גם כן, ויחזור להפעיל את A. שים לב כי מכיוון שהשפה L הינה אינסופית, תמיד ייצר A מילה הגדולה (בסדר הלקסיקוגרפי) מהמילה האחרונה שייצר A' כי ישנו רק מספר סופי של מילים הקטנות (בסדר הלקסיקוגרפי) ממילה זו. מהבניה ברור כי האנומרטור, A', הוא קנוני, ולכן (לפי תרגיל 7.15) השפה L' שהוא מייצר (שהיא אינסופית ומוכלת ב-L) מקיימת $L' \in \mathbf{R}$.

<u>פרק 8</u>

<u>תרגיל 8.2:</u>

ב. הפלט זהה בשני המקרים, וזה נובע ישירות מסעיף (א) בשאלה, ומכך ש-M_U מחשבת את הפונקציה U.

ג. תוך שימוש בבניה 8.1 (בה מוגדרת הפונקציה האוניברסלית) ועל-פי הנתון נובע כי $f_M(<M'>,x) = U(<M'>,x)$. כלומר, M מחשבת את הפונקציה האוניברסלית U (ולכן היא מכונה אוניברסלית).

ד. הטענה אינה נכונה. למשל התבונן במכונה M אשר עבור כל מחרוזת x מחזירה פלט קבוע (כמו למשל המילה הריקה).

<u>תרגיל 8.3:</u> השתמש במשפט 8.2 לגבי המכונה M_{double} אשר בהנתן קלט y מחזירה פלט yy. חלקה השני של טענת התרגיל נובע מאותם שיקולים כמו במשפט 8.4.

<u>תרגילים 8.4-8.6:</u> תרגיל 8.6 למעשה כולל את תרגילים 8.4 ו-8.5. הוכחת התרגיל נובעת משימוש במשפט 8.2 לגבי המכונה, M_f, המחשבת את הפונקציה f (הניתנת לחישוב מהיותה רקורסיבית) ולאחר מכן מחזירה את $1^{f(n)}$ כפלט (המעבר לייצוג אונרי גם הוא רקורסיבי). שים לב לכך שייצוג קלט n, דורש $\log_2 n$ ביטים.

<u>תרגיל למתקדמים:</u>

א. נשתמש ברמז המוצע ונבנה מכונה (שקידודה כמובן באורך $O(1)$) אשר מקבלת כקלט מספר טבעי n (אשר בייצוג בינארי ארכו $\log_2 n$ ביטים) ומחשבת את x_n^f. קידוד המכונה ביחד עם הקלט ימלאו את דרישת התרגיל כקלט למכונה האוניברסלית. המכונה תסרוק את כל ה x-ים בסדר לקסיקוגרפי, עבור כל אחד מהם תחשב את $f(x)$ (שים לב כי מאחר ו- f מלאה וניתנת לחישוב, חישוב כזה צורך זמן סופי) וה-x הראשון שיקיים $f(x)>n$ הוא יהא הפלט.

ב. מהגדרה 8.1 נקבל כי K מוגדרת על כל קלט ולכן היא פונקציה מלאה. במשפט 8.4 הוכח כי לכל n, מספר המחרוזות שסיבוכיותן אינו עולה על n הינו סופי (בפרט, $|\{x : K(x)\le n\}|<2^{n+1}$). לפיכך נובע כי K (המוגדרת על כל הקבוצה האינסופית של כל המילים) הינה בהכרח בלתי חסומה. נניח בשלילה כי K ניתנת לחישוב. נתבונן באברי הסדרה x_n^K, על פי הגדרת x_n^K מתקיים $K(x_n^K)\ge n$, אך על פי סעיף (א) מתקיים $K(x_n^K)\le \log_2 n + O(1)$. מתקבלת סתירה, ולפיכך נובע כי הפונקציה K אינה ניתנת לחישוב.

ג. בהנחה כי קיימת מכונה M כנ"ל, נחשב את הפונקציה K. בהנתן קלט x, נריץ את M על הקלטים $1,2,3,\cdots$ עד אשר קבוצת הפלט תכלול את x. בשלב זה נעצור ונחזיר את הקלט האחרון שעליו רצה המכונה M.

<u>פרק 9</u>

<u>תרגיל 9.1:</u> נשתמש ברמז על מנת להראות כי $L_U \in \mathbf{RE}\setminus\mathbf{R}$. ראשית, קיומן של מכונות אוניברסליות (המקבלות את השפה L_U) גורר כי $L_U \in \mathbf{RE}$. שנית, קיום מכונה, M_U, המכריעה את L_U (כלומר, $L_U \in \mathbf{R}$) יגרור קיום מכונה M_h המחשבת את הפונקציה h (אשר על-פי משפט 9.2 היא אינה ניתנת לחישוב): המכונה M_h תפעיל את M_U על הקלט $(<M>,x)$. אם M_U תקבל את הקלט אז M_h תחזיר 1, אחרת M_h תחזיר 0.

<u>תרגיל 9.2:</u> ראינו כי $L_U \in \mathbf{RE}$, כמו כן לו $\overline{L_U}$ היתה גם כן ב-**RE**, היינו מקבלים (מתרגיל 7.5) כי $L_U \in \mathbf{R}$ בסתירה לתרגיל 9.1.

<u>תרגיל 9.3:</u> נניח בשלילה כי $L \in \mathbf{RE}$ (כלומר, קיימת מכונה M_L המקבלת את השפה L) ונבנה מכונה M_U המכריעה את L_U. המכונה M_U, בהנתן קלט y $(y=(<M>,x))$ מפעילה "במקביל" שני עותקים של M_L על הקלטים $0y$ ו-

$1y$. מהגדרת L, בדיוק אחד מהקלטים שייך ל-L ולכן מובטח כי לפחות אחד העותקים יעצור ויקבל את הקלט שלו. המכונה M_U תקבל את הקלט y אם M_L קיבלה את הקלט $1y$. אחרת (M_L קיבלה את $0y$) תדחה את הקלט. שים לב כי טיעון דומה, תופס עבור $\overline{L} \in \mathbf{RE}$, משום ש- $\overline{L} = \{0y : y \in L_U\} \bigcup \{1y : y \notin L_U\} \bigcup \{\lambda\}$.

<u>תרגיל 9.4:</u> נבחר את המכונה M כמכונה אוניברסלית M_U (למשל זו שהוגדרה בתחילת הפרק) אשר עבורה, כלקלט y מהווה קידוד של $(<M>,x)$ כלשהם. נקבל:

$L_{M_U} \triangleq \{y : y$ עוצרת על $M_U\} = \{(<M>,x) : (<M>,x)$ עוצרת על $M\} = \{x : x$ עוצרת על $M\}$

כלומר, קיבלנו כי $L_{M_U} = L_U$ ומתרגיל 9.1 נקבל כי $L_{M_U} \in \mathbf{RE} \setminus \mathbf{R}$. כמו כן שים לב כי עבור כל מכונה M, מתקיים $L_M \in \mathbf{RE}$, והיא מתקבלת ע"י מכונה המקבלת קלט y אם M עוצרת על y.

<u>תרגיל 9.6:</u>

א. לא. $\mathbf{R}$ היא תת-קבוצה לא טריביאלית של $\mathbf{RE}$ (כלומר, $\varnothing \neq \mathbf{R} \subsetneq \mathbf{RE}$).

ב. כן. התשובה במקרה זה הינה תמיד לא. כלומר זוהי השפה הריקה שודאי ניתנת להכרעה.

ג. לא. במילים אחרות זוהי קבוצת קידודי המכונות המקבלות שפות אינסופיות, וקבוצה זו עומדת בתנאי משפט רייס.

ד. לא. המחלקה $\mathbf{RE} \cap \{L = L \circ L\}$ היא תת-קבוצה לא טריביאלית של $\mathbf{RE}$ כי היא מכילה את Σ^* למשל אך אינה מכילה את השפה $\{10^*\}$.

ה. לא. המחלקה $\mathbf{RE} \cap \{L = L^*\}$ היא תת-קבוצה לא טריביאלית של $\mathbf{RE}$ מאותם שיקולים של סעיף (ד).

ו. לא. (שים לב כי $L^* = \Sigma^*$ אם $\Sigma \subseteq L$).

ז. לא. נניח בשלילה כי קיימת מכונה M_c המכריעה שפה זו, ונבנה מכונה M_s המכריעה את בעית העצירה. המכונה M_s, בהנתן קלט $(<\Pi>,x)$, בונה קידוד של מכונה $<\Pi_x>$ המתעלמת מהקלט שלה ומריצה את Π על x. המכונה M_s תחזיר את תשובת M_c על $<\Pi_x>$. המכונה M_s אכן מכריעה את בעית העצירה מכיוון ש-Π עוצרת על x (לאחר מספר סופי של צעדים, שנסמנו ב-c) אם Π_x עוצרת על כל קלט לאחר c צעדים (שים לב כי בהתעלמות מהקלט אין כוונה בשום אופן למחיקתו. מחיקת הקלט, היתה מונעת מ-Π_x לרוץ על כל קלט זמן קבוע והיתה מוסיפה זמן ריצה לינארי באורך הקלט הניתן ל-Π_x).

ח. כן. קיימת מכונה המכריעה את S_c, בהנתן קלט $<M>$, מריצה את M על כל הקלטים שאורכם אינו עולה על c, בדיוק c צעדים (אלא אם כן M עוצרת קודם לכן) ומקבלת את $<M>$ אם M עצרה בכל אחת מריצות אלו. שים לב כי מספר הקלטים שארכם אינו עולה על c הוא סופי ולכן המכונה המכריעה את S_c תמיד עוצרת. כמו כן שים לב כי מספיק להריץ את M רק על קלטים מאורך שאינו עולה על c, מכיוון שב-c צעדים, המכונה M יכולה לסרוק לכל היותר את c התאים הראשונים שעל סרט הקלט שלה.

<u>תרגיל 9.8:</u>

א. [1] השפה המשלימה (שנסמנה $\overline{L}$) אינה שייכת ל-**RE**. נראה זאת ע"י רדוקציה מהשפה המשלימה לבעית העצירה (כלומר, השפה $\overline{L}_{\varnothing} = \{<M,x> : M \text{ אינה עוצרת על } x\}$) לשפה $\overline{L}$. נשתמש בדיוק באותה רדוקציה מהוכחת משפט 9.3. נזכר כי לכל קלט y מתקיים כי Π_x (מהוכחת המשפט) עוצרת על y אם המכונה M (Π בהוכחת המשפט) עוצרת על x. לכן, המכונה M לא עוצרת על x אם קיים y כך ש-Π_x לא עוצרת על y, ולכן הרדוקציה תקפה.

[2] גם השפה, L, עצמה אינה שייכת ל-**RE**. נראה זאת ע"י רדוקציה מהשפה $\overline{L}_{\varnothing}$ לשפה זו. שים לב כי ברדוקציה זו יופיע "טריק" חדש שלא נזכר עד כה. בהנתן קלט $<M,x>$, מכונת הרדוקציה תבנה קידוד של מכונה, $<\Pi_{M,x}>$ (המכילה קידוד של המכונה M ושל "הקלט" x) אשר פעולתה תתואר להלן: בהנתן קלט y, המכונה $\Pi_{M,x}$, תכתוב את x על סרט עזר, תריץ את M למשך $|y|$ צעדים על x, ותקבל אם M לא עצרה על x לאחר $|y|$ צעדים. במקרה בו M אינה עוצרת על x, נקבל כי לכל y המכונה M אינה עוצרת על x תוך $|y|$ צעדים, ולכן $\Pi_{M,x}$ תקבל כל קלט y (כלומר, $L(\Pi_{M,x})=\Sigma^*$). במקרה בו M עוצרת על x (לאחר n צעדים) המכונה $\Pi_{M,x}$ תדחה כל קלט y המקיים $|y|\geq n$ (כלומר, $L(\Pi_{M,x})\neq\Sigma^*$). ודא כי פונקצית הרדוקציה ניתנת לחישוב.

ב. השפה המשלימה שייכת ל-**RE**. מכונה $\overline{M}$ המקבלת אותה, היא מכונה המריצה "במקביל" את M על כל הקלטים ב-Σ^* (הכוונה פה היא לריצה מבוקרת, כמו בהוכחת משפט 7.5). אם מתקבל אחד הקלטים היא עוצרת ומקבלת את הקלט $<M>$. שים לב כי השפה (וכן השפה המשלימה) מקיימת את תנאי משפט רייס ולכן אינה ב-**R**. לכן מכיוון שהשפה המשלימה נמצאת ב-**RE** נקבל כי השפה בתרגיל אינה ב-**RE**.

ג. [1] השפה המשלימה (שנסמנה $\overline{L}$) אינה שייכת ל-**RE**. נשתמש באותה הרדוקציה שהשתמשנו בחלק [1] של סעיף (א), על מנת להראות שהשפה המשלימה אינה שייכת ל-**RE**. ודא את נכונות הרדוקציה גם כאן.

[2] גם השפה עצמה אינה שייכת ל-**RE**. נשתמש גם כאן באותה הרדוקציה שהשתמשנו בחלק [2] של סעיף (א) (על מנת להראות שהשפה אינה שייכת ל-**RE**) בשינוי קטן. השינוי יהיה שהמכונה $\Pi_{M,x}$ שאת קידודה מייצרת מכונת הרדוקציה, תדחה את 99 המילים הראשונות בסדר הלקסיקוגרפי, ועבור שאר המילים ב-Σ^* תפעל באופן זהה למכונה מסעיף (א). ודא גם כאן את נכונות הרדוקציה.

ד. השפה המשלימה שייכת ל-**RE**. מכונה $\overline{M}$ המקבלת אותה, היא מכונה המריצה "במקביל" את M על כל הקלטים ב-Σ^* וסופרת את מספר הקלטים שנתקבלו עד עתה. אם התקבלו 100 קלטים (לפחות) היא עוצרת ומקבלת את הקלט $<M>$. שים לב כי גם כאן, מאותם שיקולים שבסעיף (ב), השפה אינה שייכת ל-**RE**.

ה. לגבי C לא טריביאלי כלשהו, ברור ממשפט רייס כי מתקיים $L_C \notin \mathbf{R}$, אך קיימים שלושה מקרים אפשריים:

[1] השפה L_C נמצאת ב-**RE**: למשל עבור $L_C = \{<M> : L(M)\neq\varnothing\}$ (ראה הוכחה בסעיף (ב)).

[2] השפה $\overline{L_C}$ נמצאת ב-**RE**: למשל עבור $L_C = \{<M> : L(M)=\varnothing\}$ (ראה הוכחה בסעיף (ב)).

[3] השפות L_C ו-$\overline{L_C}$, אינן נמצאות ב-**RE**: למשל עבור השפה מסעיף (א).

<u>פרק 10</u>

<u>תרגיל 10.1:</u> לכל פונקציה הניתנת לחישוב, קיימת מכונת טיורינג (רגילה) המחשבת אותה. מכונת האוב שתחשב פונקציה זו, תשתמש במכונת הטיורינג הנ"ל, תתעלם מסרט האוב, לא תעבור למצב זימון האוב, וכך תחשב את אותה פונקציה.

<u>תרגיל 10.2:</u> בהנתן מכונת האוב M ומכונה M_f לחישוב הפונקציה f, נבנה מכונה M_g (המחשבת את הפונקציה g) כדלקמן: המכונה M_g תפעיל, על הקלט הנתון, את מכונת האוב M, ובכל פעם ש-M תכנס למצב זימון האוב, המכונה M_g תפעיל את M_f על תוכן "סרט האוב" ותחזור להפעיל את המכונה M ממצב הופעת האוב. מה שעשינו כאן, זוהי למעשה סימולציה לפעולת האוב (במכונה M) ע"י החלפתו במכונה M_f.

<u>תרגיל 10.3:</u> נבחר את הפונקציה f להיות הפונקציה g עצמה, והמכונה M פשוט תעתיק את סרט הקלט שלה אל סרט האוב, תעבור למצב זימון האוב ולבסוף תעתיק את תשובת האוב (תוכן סרט האוב) אל סרט הפלט שלה.

<u>תרגיל 10.4:</u> תהי h הפונקציה הניתנת לחישוב המובטחת מהגדרת רדוקציה ע"י העתקה, המקיימת $f_1(x)=f_2(h(x))$. נבנה מכונת טיורינג, M, המחשבת את f_1 עם אוב לפונקציה f_2. המכונה M, בהנתן קלט x, תחשב את $h(x)$. לאחר מכן המכונה M "תשאל את האוב" את השאלה $h(x)$ (ע"י העתקת $h(x)$ לסרט האוב, ומעבר למצב זימון האוב) ותחזיר את תשובת האוב על סרט הפלט שלה. תשובת האוב הינה $f_2(h(x))$ שהיא אכן $f_1(x)$ כנדרש.

<u>תרגיל 10.5:</u> פעולת המכונה להכרעת L_1, בהנתן קלט x, תהיה: חישוב $h(x)$ (h היא אותה פונקציה ניתנת לחישוב המובטחת מהגדרת רדוקציה ע"י העתקה) והפעלת המכונה להכרעת L_2 (שקיימת מההנחה $L_2\in\mathbf{R}$) על הקלט $h(x)$. מהגדרת רדוקציה ע"י העתקה, $x\in L_1$ אמם $h(x)\in L_2$, לכן מכונה זו מכריעה את השפה L_1 ולכן $L_1\in\mathbf{R}$.

<u>תרגיל 10.6:</u>

א. נראה רדוקציה מבעית העצירה (שכידוע אינה ב-**R**) לבעיה זו. בהנתן קלט לבעית העצירה, $\langle M,x\rangle$, נעתיק אותו לזוג, $\langle M_x,f\rangle$, כאשר f הוא מצב עצירה חדש אשר אינו בקבוצת המצבים של M, ו-M_x היא מכונה אשר פעולתה תתואר להלן: בהנתן קלט y, המכונה M_x מוחקת את y, כותבת במקומו את x, ועליו היא מפעילה אתהמכונה M, כאשר היא מוסיפה ל-M את כל החוקים מהצורה $\delta(q,\sigma)=(f,\sigma,S)$, עבור כל הזוגות (q,σ) ($q\in Q$ ו- $\sigma\in\Gamma$) אשר הפונקציה δ אינה מוגדרת עליהם. ברור כי העתקה זו ניתנת לחישוב,

וכן ברור כי המכונה M_x עוצרת (במצב f כמובן) אמם המכונה M עוצרת על הקלט x, כמו כן M_x עוצרת אמם היא נכנסת למצב f, ולכן ההעתקה הינה תקפה.

ב. רעיון הרדוקציה מבעית העצירה לבעיה זו, הוא: בהנתן קלט $(\langle M\rangle,x)$ לבעית העצירה, תועתק המכונה M, למכונה אשר מריצה את M על הקלט x, ועם סיום החישוב, מוחקת את הסרט שלה ועוברת למצב עצירה חדש. כמו כן α תהיה הקונפיגורציה ההתחלתית של הרצת M על x, ו-β תהיה הקונפיגורציה המתארת עצירה על סרט ריק במצב העצירה החדש.

ה. רעיון הרדוקציה מהבעיה המשלימה לבעית העצירה לבעיה זו, הוא: בהנתן קלט $(\langle M\rangle,x)$ לבעיה המשלימה לבעית העצירה, יועתק הקלט לקידוד של מכונה, $\langle M_x\rangle$. המכונה M_x, בהנתן קלט y, מוחקת את y ומריצה את M על x. כמו כן, המכונה M_x עושה צעד סרק לאחר כל צעד, פרט לצעד האחרון (שהוא עצירה) דבר שיבטיח מספר צעדים אי-זוגי במקרה בו M עוצרת על x. ודא כי המכונה M_x רצה על כל קלט מספר זוגי של צעדים אמם המכונה M לא עוצרת על הקלט x.

פרק 11

<u>תרגיל 11.1:</u> לשם פתרון בעית פוסט אשר בה לכל i מתקיים $|y_i| \geq |x_i|$, לא נוכל להשתמש בזוגות של מחרוזות המקיימות $|y_i| > |x_i|$ (כי במקרה זה המילה המורכבת מה-x-ים תהיה ארוכה יותר מהמילה המורכבת מה-y-ים). כלומר, נוכל להשתמש רק בזוגות של מחרוזות המקיימות $|y_i| = |x_i|$. לכן, כל מה שיש לעשות על מנת לפתור בעית פוסט זו, הוא לעבור על כל זוגות המחרוזות ולקבל את הקלט אמם קיים i כך ש- $x_i = y_i$.

<u>תרגיל 11.2:</u> ראשית נצדיק את הטענה כי קיימת מכונת טיורינג אשר זמן הריצה שלה על קלטים באורך n הינו 2^{2^n} לפחות. המכונה, M, אשר בהנתן קלט באורך n מחזירה את 2^{2^n} בייצוג אונרי, עונה על הדרישה מכיוון שרק הדפסת הפלט דורשת 2^{2^n} צעדי חישוב. כעת משפחת הקלטים לבעיית ה-PCP המבוקשת, תורכב מהקלט לבעית פוסט המתקבל מהרדוקציה שבטענה 11.6 עבור המכונה M ועבור הקלטים מהצורה 1^n. שים לב כי כל זוגות המחרוזות, פרט לזוג הראשון, אינם תלויים בקלט של המכונה M ולכן גודלם הוא קבוע (אשר תלוי ב-M בלבד). הזוג הראשון ישתנה על פי הקלט (ויהיה $(\#,\#q_0 1^n\#)$) ולכן אורך הקלט (לבעית פוסט) כולו יהיה $n+c$ (כאשר c הוא קבוע). פתרון לבעיה זו, אורכו כמספר צעדי המכונה M על הקלט 1^n, ולכן ארכו 2^{2^n} לפחות.