

מבני נתונים ב07

תרגול 10
27/6/2007

קידוד/כיווץ/הופמן



מנהלה

□ תרגיל מעשי מס' 2

- הגשה נדחתה ליום ראשון
- אנא זכרו לבדוק הרשאות ולהריץ את תכנית הבדיקה
- שימו לב שהתרגיל עודכן, ניתן להניח בערמה לא יהיו יותר מ- 10000 פריטים (אם אתם משתמשים במערך ניתן להקצות ישר בגודל זה)

פתרו את הרקורסיה הבאה: $T(n) = 2T(n-1) + 1$ □

פתרון □

נוכיח ש- $T(n) = O(n)$ ברקורסיה ■

הבסיס ברור □

$$T(n) = 2T(n-1) + 1 = 2O(n-1) + 1 = O(n) \quad \square$$

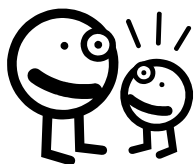
מה עשינו לא נכון? ■

$$\exists c \mid T(n) \leq c \cdot n \quad \leftarrow T(n) = O(n) \quad \square$$

$$T(n) = 2T(n-1) + 1 \leq 2 \cdot c(n-1) + 1 = 2cn - 2c + 1 \quad \square$$

ה- c לא יכול להשתנות □

מה הפתרון הנכון? בערך 2^n ■



חזרה קצרה: כיווץ/קידוד

- אלגוריתם כיווץ/קידוד מקבל מחרוזת S מעל א"ב בגודל k ומחזיר מחרוזת בינארית S'
- מה מעניין אותנו?
 - מה איכות הכיווץ? מה גודל S לעומת S' ?
 - כמה בכלל ניתן לכווץ את S ?
 - איזה מידע על S יכול לעזור לנו בכיווץ?

אנטרופיה

□ מהי אנטרופיה?

■ מידת ה"אי-וודאות" (uncertainty) של משתנה מקרי

□ דוגמאות

■ האנטרופיה של מחרוזת תווי ASCII הנבחרים באקראי הינה 7 ביטים לתו, מאחר שלא ניתן לנבא את התו הבא

■ מחרוזת ארוכה של 'Z' היא בעלת אנטרופיה 0 כיוון שכל תו צפוי מראש

■ האנטרופיה של טקסט באנגלית הוא בין 1 ל-1.5 ביט לתו (לא נוכיח)

אנטרופיה

□ אנטרופיה אמפירית מסדר 0

- בהינתן מחרוזת S באורך n מעל אלפבית $\{\alpha_1, \dots, \alpha_k\}$
נסמן ב- n_i את מס' ההופעות ב- S של התו α_i .
- האנטרופיה של S :

$$H_0(s) = \sum_{i=1}^k \frac{n_i}{n} \log_2 \left(\frac{n}{n_i} \right) = - \sum_{i=1}^k p_i \log_2(p_i)$$

p_i הן השכיחויות

$$p_i = \frac{n_i}{n}$$

למה מסדר 0? כל תו נבחר בלי תלות בתווים הקודמים

Prefix Codes

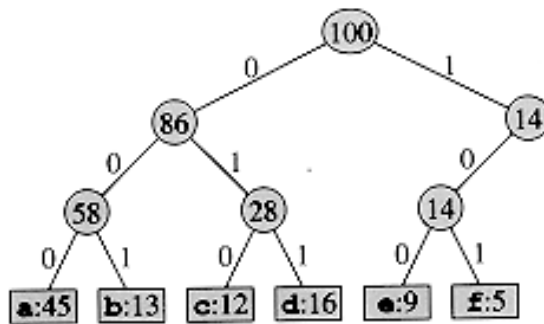
- נניח א"ב של 6 תווים שכל אחד מופיע בתדירות מסוימת
- קידוד אורך קבוע
 - אם נייצג כל תו במס' ביטים שווה נזדקק ל... 3 ביטים
 - עבור קובץ של 100000 תווים נזדקק ל...300000 ביטים
- קידוד אורך משתנה (variable length code)
 - תווים שחוזרים הרבה נייצג במעט ביטים (וההפך)

	a	b	c	d	e	f
Freq.	45	13	12	16	9	5
Fixed encoding	000	001	010	011	100	101
Variable encoding	0	101	100	111	1101	1100

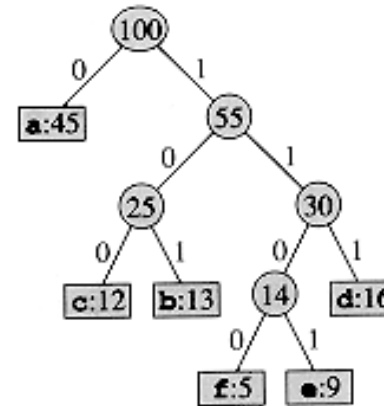
- עבור אותו קובץ נזדקק ל...224000 ביטים (25% חסכון)

Prefix Codes

- נגדיר קודים כך שאף קוד לא משמש כ-prefix לקוד אחר
- הדבר יקל על encoding/decoding
- ניתן לייצג קוד ע"י עץ בינארי
- קוד אופטימאלי תמיד מיוצג ע"י עץ בינארי מלא



(a)

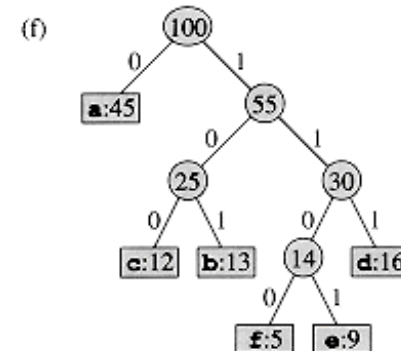
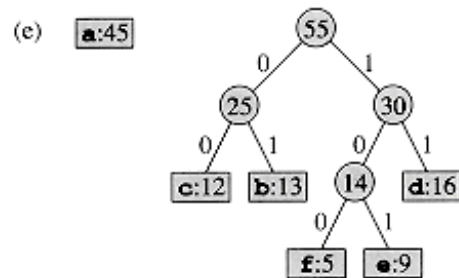
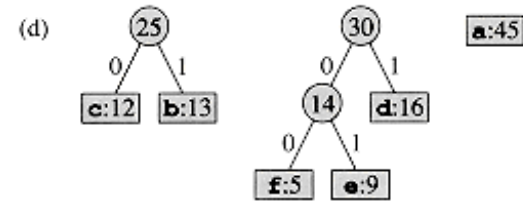
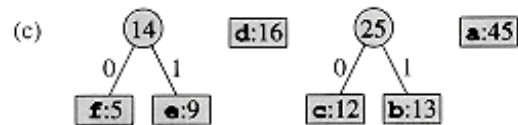
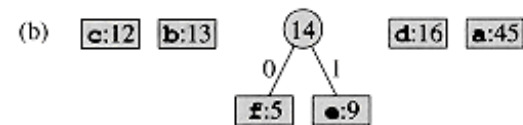


(b)

קוד הופמן

- הומצא ב-1951 ע"י הסטודנט David Huffman בתור מטלה לקורס תיאוריית המידע
- זהו אלגוריתם חמדני הבונה עץ של prefix code מלמטה למעלה

(a) f:5 e:9 c:12 b:13 d:16 a:45



קוד הופמן

□ משפט 1

■ אם S מחרוזת ו- C_H קוד הופמן שלה, ו- C הוא Prefix Code
אחר. אזי $|C_H(S)| \leq |C(S)|$



Huffman הוא קידוד Prefix Code אופטימאלי

"תרגיל"

□ בהינתן מחרוזת S באורך 800 מעל א"ב של 8 תווים
בתדירות זהה, מהו H_0 של S ?

□ פתרון

$$H_0(s) = \sum_{i=1}^8 \frac{n_i}{n} \log_2 \left(\frac{n}{n_i} \right) = \sum_{i=1}^8 \frac{100}{800} \log 8 = 8 \cdot \frac{100}{800} \cdot 3 = 3$$

□ אם היינו בונים עץ הופמן איך הוא היה נראה?

קוד הופמן ואנטרופיה

□ משפט 2

■ עבור כל מחרוזת S מתקיים

$$H_0(S) \cdot n \leq |Huff(S)| \leq (H_0(S) + 1) \cdot n$$

"גודל ממוצע של קידוד" הוא בין $H_0(S)$ ל- $H_0(S)+1$

□ משפט 3

■ עבור כל אלגוריתם כיווץ A תהיה מחרוזת S

$$|A(S)| \geq n \cdot H_0(S) \quad \text{כך ש}$$

אין אלג' כיווץ שמסוגל לכווץ כל מחרוזת לפחות מ- $nH_0(S)$

קוד הופמן ואנטרופיה

□ הוכחה (משפט 2)

■ בהינתן S נגדיר C Prefix Code כך ש- $|C(S)| \leq (H_0(S) + 1) \cdot n$

■ נגדיר את הקוד כך:

□ את התו a_i הקוד C יקודד בעזרת מחרוזת ביטים באורך $\left\lceil \log_2\left(\frac{n}{n_i}\right) \right\rceil$

■ נניח קיים קוד כזה, אזי:

$$|C(S)| = n_1 (\text{Coding Size } a_1) + \dots + n_k (\text{Coding Size } a_k)$$

$$= \sum_{i=1}^k n_i \left\lceil \log_2\left(\frac{n}{n_i}\right) \right\rceil \leq \sum_{i=1}^k n_i (\log_2\left(\frac{n}{n_i}\right) + 1)$$

$$= \sum_{i=1}^k n_i (\log_2\left(\frac{n}{n_i}\right)) + \sum_{i=1}^k n_i$$

$$= n \cdot H_0(S) + n = (H_0(S) + 1) \cdot n$$



$$|Huff(S)| \leq (H_0(S) + 1) \cdot n$$

קוד הופמן ואנטרופיה

□ טענה 1 (בדרך להוכחת משפט 3)

■ לא קיים אלגוריתם כיווץ A כך ש $\forall S, |S| = n : |A(S)| \leq n - 1$

□ הוכחה

■ אלגוריתם כיווץ הינו פ' חד-חד ערכית

■ נניח קיימת פ' כזו, אזי היא פ' חח"ע מקבוצה בגודל 2^n לקבוצה בגודל 2^{n-1}

■ אין פונקציה כזו ולכן לא קיים אלגוריתם כזה

קוד הופמן ואנטרופיה

□ הוכחה (משפט 3) - ההתחלה

■ נקבע n_1, \dots, n_k ואת n לערכים מסוימים

■ כמה מחרוזות כאלה יש?

$$X := \binom{n}{n_1, n_2, \dots, n_k} = \frac{n!}{n_1! n_2! \dots n_k!}$$

□ טענה 2

■ אין אלגוריתם שמכווץ כל מחרוזת מקבוצה בת X מחרוזות לפחות מ- $\log(X)$ ביטים

□ הוכחה (טענה 2)

■ הדבר יגדיר פונקציה חח"ע מקבוצה בגודל X לקבוצה בגודל $> 2^{\log X} = X$

■ בסתירה לטענה 1, ולכן הטענה נכונה

קוד הופמן ואנטרופיה

□ אזי $\log(X)$ חסם תחתון על הכיווץ

□ נחשב את $\log(X)$: נשתמש בהערכה ש- $\log(m!) \approx m \log m$

$$\log X = \log \frac{n!}{n_1! \dots n_k!} = \log n! - \log n_1! - \log n_2! - \dots - \log n_k!$$

$$\approx n \log n - n_1 \log n_1 - n_2 \log n_2 - \dots - n_k \log n_k$$

$$= (n_1 + n_2 + \dots + n_k) \log n - n_1 \log n_1 - n_2 \log n_2 - \dots - n_k \log n_k$$

$$= n_1 \log n + n_2 \log n + \dots + n_k \log n - n_1 \log n_1 - n_2 \log n_2 - \dots - n_k \log n_k$$

$$= n_1 \log \frac{n}{n_1} + \dots + n_k \log \frac{n}{n_k}$$

$$= nH_0(S)$$



הוכחנו את משפט 3. מש"ל

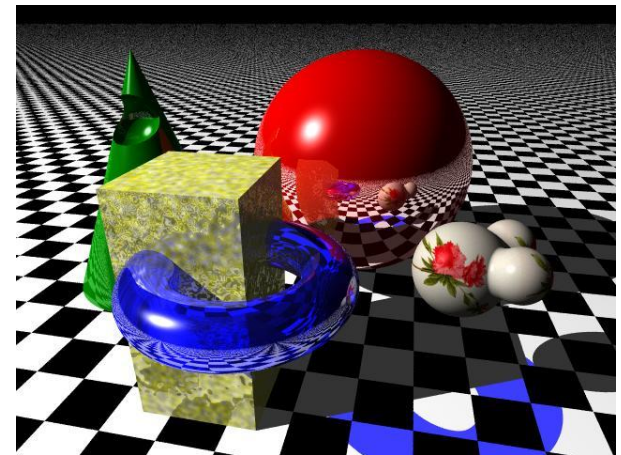
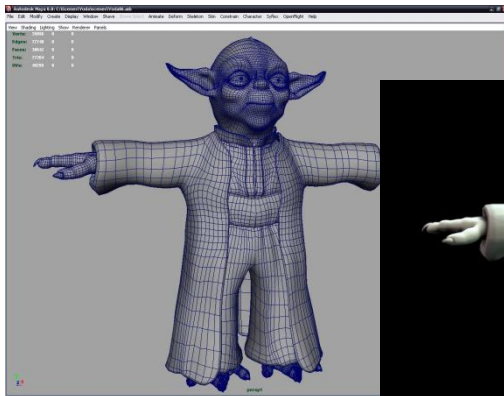
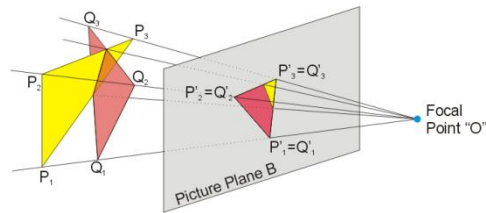
פרסום חסר בושה



□ קורס גרפיקה: סמסטר ב' שנה הבאה

□ מרצה: ליאור שפירא

□ תוכן: מאד מגניב



תם הטקס – נגמר הקורס - הסוף

בהצלחה
במבחן!!!

