[redacted]

חלוקה אגליטרית של חפצים בדידים

Egalitarian Item Allocation

אראל סגל-הלוי

חלוקה אגליטרית

תזכורת: **חלוקה אגליטרית** = חלוקה שבה
הערך **הקטן ביותר** בין כל השחקנים
הוא **גדול ביותר** בין כל החלוקות:

$$\max_X \min_i V_i(X_i)$$

כשהמשאבים **רציפים**, וההערכות מנורמלות, כל
חלוקה אגליטרית היא **פרופורציונלית**.

כשהמשאבים **בדידים**, זה לא מתקיים.

דוגמה: 99 חפצים, שני אנשים. החלוקה האגליטרית
היא 49:50 - לא פרופורציונלית.

חלוקה אגליטרית היא "הכי קרובה לפרופורציונלית".

חלוקה אגליטרית - חישוב

כשהמשאבים **רציפים**, קיים אלגוריתם יעיל למציאת חלוקה אגליטרית.

משפט. כשהמשאבים בדידים, מציאת חלוקה אגליטרית היא בעיה NP-קשה.

הוכחה. רדוקציה מבעיית *חלוקת המספרים (Partition)*:
"נתונים m מספרים חיוביים שסכומם $2S$.
האם ניתן לחלקם לשתי קבוצות שסכומן S?"

בהינתן בעיית חלוקת מספרים P, נגדיר בעיית חלוקת חפצים Q, עם שני שחקנים המייחסים לכל חפץ j את המספר ה-j. התשובה לבעייה P היא "כן" אם ורק אם ערך החלוקה האגליטרית בבעייה Q הוא S. ***

איך פותרים בעיות -NPקשות?

	איכות הפתרון	זמן הריצה
אלגוריתם מדוייק	תמיד מיטבי	מעריכי במקרה הגרוע; מהיר בבעיות קטנות
אלגוריתם קירוב	לא מיטבי, אבל קרוב	תמיד פולינומיאלי

חלוקה אגליטרית - אלגוריתמים מדויקים

חיפוש במרחב המצבים

state-space search

מצב של חלוקה חלקית =: וקטור באורך $n+1$ (מספר החפצים שחולקו, והערך של כל שחקן).

המצב של חלוקה ריקה = (0; 0,...,0).

הרעיון:

- נתחיל מחלוקה ריקה;
- ניצור את כל n המצבים הנובעים ממצב קיים + חלוקת חפץ אחד;
- נמחק מצבים מיותרים (*גיזום*; פירוט בהמשך);
- מתוך כל המצבים הסופיים (m= חפצים חולקו), נבחר מצב עם הערך המינימלי הגדול ביותר.

חיפוש במרחב המצבים – דוגמה

	חפץ א	חפץ ב	חפץ ג
שחקן 1	11	11	55
שחקן 2	22	22	33
שחקן 3	33	44	0

מצב התחלתי: (0; 0,0,0).

נתינת חפץ א: (1; 11,0,0), (1; 0,22,0), (1; 0,0,33).

נתינת חפץ ב: (2; 11,0,44), (2; 11,22,0), (2; 22,0,0).

(2; 0,22,44), (2; 0,44,0); (2; 11,22,0).

(2; 0,0,77), (2; 0,22,33); (2; 11,0,33).

נתינת חפץ ג: 27 מצבים.

באופן כללי: n^m מצבים.

גיזום (pruning) - כלל א

כלל א: נמחק מצבים זהים.

מצב התחלתי: (0,0,0 ;0).

נתינת חפץ א: (11,0,0 ;1), (0,22,0 ;1), (0,0,33 ;1).

נתינת חפץ ב: (11,0,44 ;2), (11,22,0 ;2), (22,0,0 ;2).

(0,22,44 ;2), (0,44,0 ;2); ~~(11,22,0 ;2)~~.

(0,0,77 ;2), (0,22,33 ;2); (11,0,33 ;2).

נתינת חפץ ג: ~~27~~ **24** מצבים.

באופן כללי: לכל היותר $\mathbf{m*V^n}$, כאשר V הוא הערך הגדול ביותר של סל כלשהו לשחקן כלשהו.

--- לכל n קבוע, האלגוריתם **פסאודו-פולינומיאלי**.

גיזום – כלל ב (branch-and-bound)

כלל ב: נמחק כל מצב, שה**חסם האופטימי** שלו אינו טוב יותר מה**חסם הפסימי** הטוב ביותר שמצאנו.

- **חסם פסימי** = התוצאה המיטבית לא תהיה גרועה יותר .
  דוגמה: חלק את החפצים שנשארו באקראי.
- **חסם אופטימי** = התוצאה המיטבית לא תהיה טובה יותר .
  דוגמה: תן כל החפצים שנשארו לכולם.

המצב (0,0,0 ;0)

- חסם פסימי: **11** (1:א, 2:ג, 3:ב).
- חסם אופטימי: 77 (1:א+ב+ג, 2:א+ב+ג, 3:א+ב+ג).

המצב (22,0,0 ;2):

- חסם אופטימי: 0 (נותנים את חפץ ג לכולם).
- קטן מהחסם הפסימי שמצאנו ← נגזום את המצב הזה!

חסמים

- בבעיית **מקסימום**, חסם אופטימי = חסם עליון, חסם פסימי = חסם תחתון (אופטימי ≤ אמיתי ≤ פסימי).
- בבעיית **מינימום**, חסם אופטימי = חסם תחתון, חסם פסימי = חסם עליון (אופטימי ≥ אמיתי ≥ פסימי).
- האלגוריתם **מהיר** יותר ככל שהחסמים **הדוקים** יותר (= קרובים יותר לערך האמיתי).
- האתגר של מפתחי אלגוריתמים מדוייקים: למצוא חסמים הדוקים יותר.

חלוקה אגליטרית - אלגוריתמי קירוב

בעיית שיבוץ העבודות

צריך לבצע m עבודות-חישוב באורכים שונים.
יש n מחשבים זהים. צריך לשבץ עבודות למחשבים כך שזמן הסיום של העבודה האחרונה יהיה קצר ביותר.

דוגמה: 4 מחשבים, 9 עבודות עם זמני-ריצה (בשעות):

4, 4, 4, 5, 5, 6, 6, 7, 7.

- שיבוץ א: 4+4+7,   4+7,   5+6,   5+6,
  זמן סיום: 15.
- שיבוץ ב: 4+4+4,   7+5,   7+5,   6+6,
  זמן סיום: 12 - **מיטבי**.

- שיבוץ א הוא **קירוב 5/4** לשיבוץ המיטבי.

שיבוץ העבודות וחלוקה אגליטרית

בעיית **שיבוץ m עבודות על n מחשבים** שקולה לבעיית חלוקה אגליטרית של m מטלות (=חפצים עם ערך שלילי) בין n אנשים עם הערכות זהות.

דוגמה: 4 אנשים, 9 מטלות, ערכים (שליליים):

-4, -4, -4, -5, -5, -6, -6, -7, -7

- חלוקה א: -7-4-4, -7-4, -6-5, -6-5
  ערך מינימלי: -15.
- חלוקה ב: -4-4-4, -5-7, -5-7, -6-6
  ערך מינימלי: -12 – **חלוקה אגליטרית.**
- חלוקה א היא **קירוב 5/4** לחלוקה האגליטרית.

List Scheduling – שיבוץ רשימה

1. לכל עבודה j בין 1 ל-m:
2. תן את j למחשב עם זמן-סיום נוכחי קטן ביותר.

1. לכל מטלה j בין 1 ל-m:
2. תן את j לשחקן, שהעלות (=מינוס הערך) הנוכחית שלו קטנה ביותר (= קרובה ביותר לאפס).

דוגמה: 4 אנשים, 9 מטלות עם עלויות:

4, 4, 4, 5, 5, 6, 6, 7, 7.

שחקן ד	שחקן ג	שחקן ב	שחקן א
5 7	4 6	4 6	4 5 7

עלות מקסימלית: 16 (ערך מינימלי: מינוס 16).

אלגוריתם הרשימה – יחס הקירוב

משפט. אלגוריתם הרשימה לחלוקת מטלות מוצא חלוקה שבה העלות המקסימלית **קטנה מפי 2** מהעלות המקסימלית המיטבית (כלומר: הוא מוצא קירוב 2).

הוכחה. נסמן: T = העלות המיטבית. סכום העלויות של כל שחקן בחלוקה המיטבית לכל היותר T. לכן, העלות של כל *מטלה* לכל היותר T, ו*סכום העלויות* לכל היותר nT.

בכל סיבוב באלגוריתם, סכום העלויות של כל המטלות שכבר חולקו קטן מ-nT. לפי כלל שובך־היונים, העלות הקטנה ביותר של שחקן קטנה מ-T. לכן, סכום העלויות החדש של השחקן שקיבל מטלה קטן מ-2T. לכן, בסוף הסיבוב האחרון, העלות של כל שחקן קטנה מ-2T.

שיבוץ "המטלה הארוכה ראשונה"
Longest Processing Time First – LPT
נקרא גם: האלגוריתם החמדני - Greedy

1. סדר את העבודות בסדר יורד של זמן הריצה;
2. הפעל "תיזמון רשימה" על הרשימה המסודרת.

1. סדר את המטלות בסדר יורד של העלות שלהן;
2. חלק את המטלות בעזרת "אלגוריתם הרשימה".

דוגמה: 4 אנשים, 9 מטלות עם עלויות:

4, 4, 4, 5, 5, 6, 6, 7, 7.

שחקן א	שחקן ב	שחקן ג	שחקן ד
7 4 4	7 4	6 5	6 5

עלות מקסימלית: 15 (ערך מינימלי: מינוס 15).

האלגוריתם החמדני – יחס הקירוב

משפט. האלגוריתם החמדני מוצא חלוקת מטלות עם עלות מקסימלית **קטנה מפי 4/3** מהעלות המיטבית.

הוכחה. נסמן את הערך המיטבי ב-T כמו קודם.

נחלק את המטלות לשני סוגים: גדולות: עלות מעל T/3; קטנות: עלות לכל היותר T/3. בכל סל בחלוקה המיטבית יש לכל היותר 2 מטלות גדולות; בסה"כ לכל היותר 2n.

האלגוריתם החמדני מחלק קודם את כל המטלות הגדולות, ואז את כל המטלות הקטנות.

נוכיח את המשפט בשתי טענות־עזר: טענה א מתייחסת לשלב חלוקת המטלות הגדולות, וטענה ב מתייחסת לשלב חלוקת המטלות הקטנות ←

האלגוריתם החמדני – יחס הקירוב

טענה א: לאחר שהאלגוריתם סיים לחלק מטלות גדולות, העלות הכוללת של כל שחקן היא לכל היותר T.

הוכחה: אם יש לכל היותר n מטלות גדולות, אז האלגוריתם החמדני נותן מטלה גדולה אחת בלבד לכל שחקן, וברור שהעלות לכל היותר T.

נניח שיש n+t מטלות גדולות, עבור t בין 1 לבין n. נקרא לשתי מטלות גדולות **תואמות** – אם סכום העלויות שלהן קטן או שווה T.

בחלוקה המיטבית יש לפחות t סלים עם שתי מטלות גדולות, ולכן יש לפחות t זוגות של מטלות גדולות תואמות (מטלות n+t, …, n-t+1). לכן:

- מטלה n+t תואמת למטלה n-t+1.
- מטלה n+t-1 תואמת למטלה n-t+2.
- מטלה n+t–k+1 תואמת למטלה n–t+k לכל k בין 1 ל־t.

האלגוריתם החמדני מחלק את מטלות n, ..., 1, מטלה אחת לכל שחקן. ואז נותן את מטלה n+t–k+1 לשחקן שקיבל את n–t+k. הזוגות הללו תואמים לכל k, ולכן עלויות כל השחקנים לכל היותר T. ***

האלגוריתם החמדני – יחס הקירוב

טענה ב: כאשר האלגוריתם נותן מטלה קטנה לשחקן כלשהו, העלות החדשה שלו קטנה מ $4T/3$.

הוכחה: סכום העלויות של כל המטלות שכבר חולקו קטן מ-nT. לפי כלל שובך־היונים, העלות הקטנה ביותר של שחקן כלשהו קטנה מ-T. בתוספת מטלה קטנה אחת, העלות החדשה קטנה מ-$4T/3$. ***

האלגוריתם החמדני - המשך

- ניתחנו את האלגוריתם החמדני לחלוקת **מטלות** לשחקנים עם הערכות **זהות**.
- אפשר להשתמש באותו אלגוריתם לחלוקת **חפצים** לשחקנים עם הערכות **זהות**: הערך המינימלי גדול **מפי 3/4** מהערך האגליטרי. ההוכחה הרבה יותר ארוכה.
- לשחקנים עם הערכות **שונות**, הבעיה הרבה יותר קשה - נושא למחקר.

אלגוריתם מדויק + אלגוריתם קירוב

במציאות מקובל לשלב את שני סוגי האלגוריתמים:

- משתמשים באלגוריתם מדוייק – חיפוש במרחב המצבים;

- מחשבים חסמים פסימיים בעזרת אלגוריתם קירוב – כגון האלגוריתם החמדני שלמדנו.

- אם נגמר הזמן, והחיפוש במרחב המצבים עדיין לא מצא חלוקה מיטבית – מחזירים את החלוקה הכי טובה שהחיפוש מצא עד כה.

- החסם הפסימי מבטיח, שהחלוקה הזאת טובה לפחות כמו החלוקה של אלגוריתם הקירוב.