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" בהתחשב בחפצים הבדידים.

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 NP-קשות?

	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, הערך 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 – pruning; 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: (11,0,0; 1), (0,22,0; 1), (0,0,33; 1).

נתינת חפץ ב: (22,0,0; 2), (11,22,0; 2), (11,0,44; 2)
(11,22,0; 2), (0,44,0; 2), (0,22,44; 2)
(11,0,33; 2), (0,22,33; 2), (0,0,77; 2).

נתינת חפץ ג: 27 מצבים. באופן כללי: n^m.

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: (22,0,0; 2), **(11,22,0; 2)**, (11,0,44; 2)
~~**(11,22,0; 2)**~~, (0,44,0; 2), (0,22,44; 2)
(11,0,33; 2), (0,22,33; 2), (0,0,77; 2).

נתינת חפץ ג: ~~27~~ 24 מצבים.

באופן כללי: לכל היותר $m*V^n$, 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, ש**החסם האופטימי** שלו אינו טוב יותר מ**החסם 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).
- אפשר לגזום 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: 5+6, 5+6, 4+7, 4+4+7
  זמן סיום: 15.
- שיבוץ ב: 6+6, 7+5, 7+5, 4+4+4
  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: ‎-5-6, -5-6, -4-7, -4-4-7‎
  ערך מינימלי: ‎-15‎.
- חלוקה ב: ‎-6-6, -7-5, -7-5, -4-4-4‎
  ערך מינימלי: **‎-12‎ - חלוקה אגליטרית**.
- חלוקה א היא **קירוב 5/4** לחלוקה האגליטרית.

שיבוץ רשימה – List Scheduling

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

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.

הוכחה. נסמן: OPT = העלות המיטבית. נחלק את כל העלויות ב־OPT. לאחר החלוקה, סכום העלויות של כל שחקן בחלוקה המיטבית ≥ 1. לכן, העלות של כל מטלה ≥ 1, וסכום העלויות של כל המטלות ≥ n.

בכל סיבוב באלגוריתם, סכום העלויות של כל המטלות שכבר חולקו < n. לפי כלל שובך־היונים, העלות הקטנה ביותר של שחקן < 1. לכן, סכום העלויות החדש של השחקן שקיבל מטלה < 1+1 = 2. לכן, בסוף הסיבוב האחרון, העלות של כל שחקן < 2. ***

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
4 4 7	4 7	5 6	5 6

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. נחלק את כל העלויות ב־OPT כמו קודם.

נחלק את המטלות לשני סוגים: גדולות: עלות > 1/3 ; קטנות: עלות ≤ 1/3. בכל סל בחלוקה המיטבית יש ≤ 2 מטלות גדולות, ובסך־הכל יש ≤ 2n מטלות גדולות.

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 ≥ 1.

הוכחה: אם יש $n \geq$ מטלות גדולות, אז האלגוריתם החמדני נותן מטלה גדולה אחת בלבד לכל שחקן, וברור שהעלות ≥ 1.

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

בחלוקה המיטבית יש לפחות t סלים עם שתי מטלות גדולות, ולכן יש לפחות t זוגות של מטלות גדולות תואמות (מטלות $n+t, \ldots, n-t+1$). 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 1, ..., n, 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 1. ***

האלגוריתם החמדני – יחס הקירוב

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

הוכחה: סכום העלויות של כל המטלות שכבר חולקו $< n$. לפי כלל שובך־היונים, העלות הקטנה ביותר של שחקן כלשהו < 1. בתוספת מטלה קטנה אחת, העלות החדשה $< 4/3$. ***

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 > **פי 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.