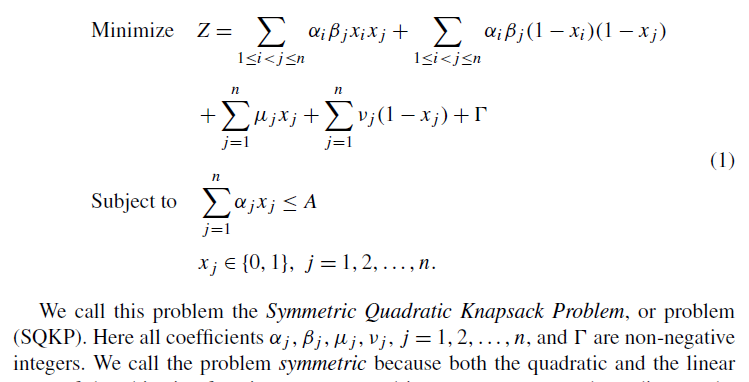
מימוש תכנון דינאמי עבור בעיית n-res

בשלב זה עסקנו במימוש של תכנון דינאמי עבור הבעיה של מכונה בודדת עם מטלות ממושקלות שבמהלך העבודה צריך שתהיה הפסקה ועבודה שהתחילה לפני ההפסקה לא יכולה להמשיך לאחריה אלא צריכה להתחיל מחדש, כפי שתכנו קלרר וסטרוסביץ'[[1]](#endnote-1).

המאמר מתחיל תחילה עם הבאת תכנון דינאמי לבעייה שהוא מכנה symmetric quadric knapsack problem שמנוסחת כך:



לבעייה זו מוצג תכנון דינאמי דומה לגירסא של תכנון קלאסי לבעיית התרמיל.

נציג גירסא זו ואז נראה את השינוי הנדרש:

בעיית התרמיל:

בהנתן ווקטורים (c מייצג ערך, w מייצג גודל) וW (גודל התרמיל) מקסם את תחת התנאי

התכנון:

ניצור טבלה (n+1)x(W) שכל התאים בה ריקים חוץ מ (0,0). כל שורה 1--(n+1) מייצגת חפץ.

עוברים שורה שורה ולכל תא לא ריק (i,j) שמים את מה שיש בו בתא (i+1, j) ואת מה שיש בו ועוד בתא (i+1,j + ) אם קיימים תאים כאלו, אם יש בהם כבר משהו שמים בהם את המקסימום בין מה שיש בהם למה שרוצים עכשיו להשים.

בסוף לוקחים את התא בשורה n שהערך בו הכי גדול ובודקים מה שיטת השמת החפצים דרכה הגענו לערך זה ומוציאים אותה כפלט.

במאמר הובאו שני דרכים איך לפתור את symmetric quadric knapsack problem בעזרת תכנון דינאמי, PDP,DDP

## PDP

הפתרון המוצע ממש דומה לתכנות הדינמי שהוצג לעיל עם שינויים מזעריים, קודם כל, שמים את המינימום בין שני אפשרויות השמה בתא כלשהו, כמו כן פונקציית הערך שונה: צריך ליצור כדלעיל טבלה בגודל A\*n נגדיר את השווי שהוכנס עד כה כ , המשקל של החפצים שעברנו עליהם הוא והמשקל של מה שהוכנס מתוכם הוא , אזי צריך להחליף את הפונקציה לחישוב השווי של הוספת החפץ ה k+1 לתרמיל להיות:

ואת הפונקציה לחישוב המקרה שלא הוספנו (פונקציה שקודם הייתה פונקציית הזהות, אבל כעת גם אפשרות זו מוסיפה לערך) להיות

.

ואז לכל תא בטבלה צריך לראות האם עדיף להגיע אליו עם להוסיף ערך או בלי.

המאמר הראה שאלו אכן הפונקציות הנדרשות כדי לחשב את הערך החדש

## DDP

הפתרון דומה לקודם אך שונה, כמו כן נראה שחסר חלק מהאלגוריתם במאמר (הוא לא טרח להציג אותו לגמרי אלא דילג על חלקים בטענה שזה כמו הקודם, אך נראה שיש כאן חלק קצת מורכב).

במקום שכל עמודה מהA עמודות תייצג איזה משקל של חפצים החלטנו להכניס פנימה, היא תייצג

את המשקל של החפצים שבחרנו לא להכניס פנימה עד כה שיקרא  *שהוא שווה ל*

*כעת במקום שכל שורה מייצגת כמה משקל בחרנו להכניס עד כה, היא מייצגת כמה משקל בחרנו שלא להכניס עד כה.*

*הפונקציות של בחירה האם להכניס או לא להכניס קלות לחישוב כי כל לעיל מוחלף ב ולכן הן:*

*האלגוריתמים שקולים מבחינת התוצאה.*

*במאמר לא צוינו עוד הבדלים ונטען שם שהסיבוכיות של שני האלגוריתמים הוא . באלגוריתם הראשון קל לראות אכן שמדובר בטבלה בת A\*n עמודות שעוברים עליה פעם אחת, אך בשני נראה שאם השורות מייצגות את המשקל שבחרנו שלא להכניס עד כה אזי יכולות להיות הרבה יותר מA שורות והסיבוכיות יכולה להיות גרועה. לכן ננסה לפתור בעיה זו.*

*עבור כל טור שמייצג שהחלטנו לגבי k חפצים עד כה ייתכן שהחלטנו לא להכניס חפצים כלל (כי יכול להיות שנחליט אפילו לא להכניס חפצים כלל) ולכן מקסימום משקל החפצים שהחלטנו לא להכניס יכול להגיע עד , מצד שני, מאחר שאסור להכניס יותר ממשקל כולל של A אזי לא יכול להיות שהחלטנו לא להכניס פחות מ ולכן תחום המשקלים שחוקי לא להכניס הוא בגודל A+1. לכן הטור המינימלי ייצג שבחרנו לא להכניס* משקל של ווהמקסימלי ייצג שבחרנו שלא להכניס משקל של , שזה בעצם כל מה שהחלטנו לגביו עד כה.

## FPTAS

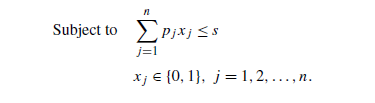
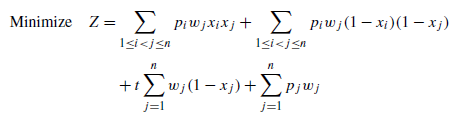
השלב הבא של המאמר הוא להראות אלגוריתם שמקיים את תנאי FPTS בהנחה שקיים אלגוריתם שנותן חסם עליון במכפלה של קבוע על התוצאה המיטבית.

בגדול האלגוריתם דומה לאלגוריתם המקביל בבעיית התרמיל ששם מקצרים את טווח המשקלים ביחס מסויים ואת הגודל המקסימלי באותו יחס ומעגל ואז שולחים לאלגוריתם הדינאמי. אך הוא הרבה יותר מורכב ולא התבקשנו לדון בו

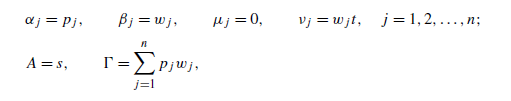
# הבעיה שלנו

הבעיה בה המאמר דן בפרק 3 היא בעיה בה יש מכונה אחת וכמות של משימות, לכל אחת מהן ידוע הזמן שהיא לוקחת - . ולכל אחד יש חשיבות - , והזמן בו מסתיימת כל משימה הוא , כמו כן יש אינטרוול זמן [s,t] בו המכונה מושבתת מעבודה. קיימת בעיה בה עבודה שהופסקה באמצע (crossover) ממשיכה לאחר מכן מהמקום שבו הייתה, בעיה זו נקראת res. הבעיה שלנו היא במקרה שבו עבודה שהופסקה באמצע צריכה להתחיל מההתחלה לאחר מכן. המטרה היא להביא למינימום את .

המאמר מוכיח שהבעיה שקולה לבעיה:



בעיה זו איזומורפית לבעיה הקודמת שהוצגה בצורה זו:



ולכן ניתן **וזה מה שנדרשנו** לממש את התכנון הדינאמי עבור בעיה זו.

לאחר מכן המאמר מביא אלגוריתם שמביא חסם עליון במכפלה של 4 על התוצאה המיטבית ובכך סולל את הדרך לאלגוריתם קירוב מצורת FTAS כמו שהבאנו קודם.

1. Kellerer, H., & Strusevich, V. A. (2010). Fully polynomial approximation schemes for a symmetric quadratic knapsack problem and its scheduling applications. *Algorithmica*, *57*(4), 769-795.‏ [↑](#endnote-ref-1)