תוכן עניינים

[חלק תאורטי 2](#_Toc405456753)

[1. הוכחת אי שלמות 2](#_Toc405456754)

[מדוע לא תיתכן פגיעה בקבילות ללא פגיעה בשלמות: 2](#_Toc405456755)

[2. אלג' שלם וקביל לחיפוש מסלול פיסיבילי 2](#_Toc405456756)

[הסבר מילולי 2](#_Toc405456757)

[פסאודו קוד 2](#_Toc405456758)

[הוכחת שלמות וקבילות 3](#_Toc405456759)

[3. מציאת מספר מסלולים לצומת מטרה 3](#_Toc405456760)

[חלק רטוב א' 4](#_Toc405456761)

[4. קבצי ה-csv 4](#_Toc405456762)

[Israel.csv 4](#_Toc405456763)

[Lights.csv 4](#_Toc405456764)

[7. פונקציית המחיר והפונקצייה היוריסטית 4](#_Toc405456765)

[8. גרף להרצת 4](#_Toc405456766)

[9. הסבר למימוש lights 4](#_Toc405456767)

[10. השוואה לחיפוש עם משקול לרמזורים וללא משקול 5](#_Toc405456768)

[חלק רטוב ב' 5](#_Toc405456769)

[13. שלמות וקבילות אלגוריתם לניווט מובטח 5](#_Toc405456770)

[16. הצגה גרפית של Assured Paths 6](#_Toc405456771)

# חלק תאורטי

## הוכחת אי שלמות

הסטודנט הציע להוסיף בדיקה על התנאי של המסלול לפני החזרת הפתרון.

בפעולה זו נגרמה פגיעה בשלמות האלגוריתם. דוגמה:

5

2

2

3

אנו מתחילים בגרף בצומת הירוק ומעוניינים להגיע לצומת הכתום. אורכי הקשתות מסומנות עליהן (היוריסטיקה 0), ונאמר כי מסלול פיסיבילי אם הוא מכיל את הקשת הכחולה. האלגוריתם כולו עובד כ-A\*, ולכן הצומת יגיע לראש התור Open בפעם הראשונה כאשר f(t)=4. בשלב זה נבדוק אם המסלול פיסיבילי ונראה כי הוא אינו. הפעם הבאה שנגלה אותו היא כאשר נפתח את הצומת הכחול משמאל, אך נראה שה-f כעת (8) לא טוב מ-4, ולכן לא נחזיר את הצומת הכתום ל-Open.

סה"כ נסיים את האלגוריתם מבלי למצוא מסלול אף על פי שהוא קיים, ולכן זוהי פגיעה בשלמות.

### מדוע לא תיתכן פגיעה בקבילות ללא פגיעה בשלמות:

מקבילות A\* כאשר צומת מטרה מגיע לראש התור הוא בעל f(s)=C\* (כאשר C\* המחיר האופטימאלי), זאת כיוון שאחרת היינו מחזירים מסלול לא אופטימלי (בסתירה לקבילות A\*).

לכן אם האלגוריתם מחזיר פתרון (שלמות) – כאשר צומת המטרה הגיע לראש התור הוא היה בעל f(s)=C\* מינ' ללא תלות בפיסיבליות הדרך, ובפרט הפתרון הוא גם האופטימלי מבין הפתרונות הפיסיביליים.

הערה: הדבר נכון כיוון שהנחנו קיום צומת מטרה יחיד.   
אם ניתן היה להוסיף צומת נוסף, בדוגמה יכולנו לחבר את צומת המקור (ירוק) לצומת מטרה נוסף בקשת כחולה בודדת במשקל 10.   
כיוון שצמתי מטרה יוצאים מ-Open פעם אחת בלבד עם המרחק המינ' אליהם, לא היינו מבקרים בצומת הכתום עם המסלול השמאלי (מאורך 8), אך היינו מגיעים לצומת החדש (עם מסלול מאורך 10) ומחזירים אותו כפתרון. כך היינו מקבלים שלמות (החזרנו פתרון פיסיבילי) אך לא קבילות (הפתרון לא אופט').

## אלג' שלם וקביל לחיפוש מסלול פיסיבילי

### הסבר מילולי

נבצע חיפוש IDA\* (כלומר חיפוש עד לעומק f\_limit, כאשר בכל איטרציה נגדיל את f\_limit).

נוסיף כעת לבדיקה של האם הצומת הבא הוא צומת מטרה את הבדיקה של פיסיביליות המסלול.

### פסאודו קוד

DFS-f (state, g, path, f-limit, goal-p, feasibility)

new-f ← g + h(state)

if new-f > f-limit then

new-limit ← min(new-limit, new-f)

return NIL

if goal-p(state) and **feasibility(path)**

return path

for c in SUCC(state)

sol ← DFS-f (c, g + COST(state,c), path || c, f-limit, goal-p)

if sol then return sol

return NIL

IDA\*(problem)

new-limit ← h(problem[init-s])

loop while resources are available

f-limit ← new-limit

new-limit ← infinity

sol ← DFS-f(problem[init-s], 0, NIL, f-limit, problem[goal-p], **problem[feasibility]**)

if sol then return sol

### הוכחת שלמות וקבילות

שלמות: אם קיים פתרון פיסיבילי, נניח שמחירו C\*, אזי מהעובדה כי g חסומה מלמטה בהכרח נגיע למצב בו f-limit>C\*, ולכן נעבור על המסלול (אלא אם מצאנו מסלול קודם).

קבילות: נניח ומצאנו פתרון. כיוון שאנו מגדילים את f-limit להיות ה-f המינ' הבא, אם היה מסלול קצר יותר מהמסלול הנמצא היינו מגלים אותו באיטרציות קודמות של IDA\*, ולכן המסלול שנמצא הוא גם המינ' ב-f, כלומר במחירו (h של צומת מטרה = 0).

## מציאת מספר מסלולים לצומת מטרה

נציע לחפש מסלולים לפי פונקציות משקל ויוריסטיקות שונות, עם אופציה לפרמטריזציה שלהן.  
נמצא זוגות של פונקציות משקל ויוריסטיקות קבילות וע"י פרמטריזציה שלהן נקבל פונקציות מחיר ויוריסטיקות קבילות תואמות.

הוכחה:

נניח פונק' משקל ויוריסטיקות קבילות תואמות. יהי צומת s, ומסלול P ממנו למסלול מטרה. נסמן *כמשקל המסלול.  
כיוון שהיוריסטיקות קבילות מתקיים . פרמטריזציה של פונק' המשקל והיוריסטיקות יהיו .  
מכאן שיתקיים , ובסה"כ היוריסטיקה החדשה אופטימית, משמע קבילה ולכן גם A\* עם פונק' המשקל היוריסטיקה הנ"ל יהיה אלגוריתם קביל.*

*כעת בסה"כ עלינו להציג זוגות של פונק' משקל ויוריסטיקות תואמות, וכך נקבל אוסף גדול של פרמטריזציות שלהן. שיקולים לדוגמא שמעניינים בבחירת המסלולים הם זמן הנסיעה, מרחק הדרך, מספר רמזורים, כבישי אגרה וכו'. ע"י פרמטריזציה של הפונק' משקל ויוריסטיקות ניתן להגיע למסלולים שמשקפים הכי נכון את בחירות המשתמש (כי יכול להיות למשל שהוא לא רוצה כבישי אגרה, אבל עדיין רוצה להגיע בזמן המינימלי ולכן צריך שילוב של שתי הפונקציות).*

*נבחר להיות המרחק ביניהם, ו- המרחק המינ' ממנו לצומת שכן.*

*נבחר*  *להיות הזמן להגיע מהצומת הראשון לשני, ו- הזמן המינ' להגיע מהצומת הראשון לצומת שכן.*

*ניתן למשל להימנע מכבישי אגרה ע"י מתן משקל אינסופי לכבישים אלו (הפונק' היוריסטית תהיה 0).*

# חלק רטוב א'

## קבצי ה-csv

### Israel.csv

כל שורה מייצגת נקודה על המפה, וכל שורה מחולקת למספר עמודות:

1. אינדקס הצומת
2. קו ה-lat של הצומת
3. קו ה-lon של הצומת
4. שאר העמודות מתארות קשרים לצמתים אחרים ומחולקות ע"י @ ל-3 מספרים. אינדקס צומת היעד, המרחק וסוג הכביש.

### Lights.csv

כל שורה מייצגת מיקום של רמזור, ומחולקות ל-2 עמודות

1. קו ה-lat של הרמזור
2. קו ה-lon של הרמזור

## פונקציית המחיר והפונקציה היוריסטית

פונקציית המחיר מקבלת שני אינדקסים של צמתים שיש בניהם כביש ומחזירה את זמן הנסיעה בניהם (מרחק הכביש בעזרת הפונקציה הניתנת ב-tools חלקי מהירות הכביש).

הפונקציה היוריסטית מקבלת אינדקס של צומת ומחזירה את המרחק האווירי מהצומת אל צומת המטרה חלקי המהירות נסיעה המקסימלית האפשרית (110 קמ"ש).   
בבירור פונקציה זו קבילה כי היא אופטימית (לא ייתכן מסלול יותר מהיר אל צומת המטרה מהמסלול האווירי במהירות המקסימלית).

## גרף להרצת

TODO

## הסבר למימוש lights

המימוש שלנו משתמש בפרמטר w שמייצג את הזמן שבו יעמוד המשתמש ברמזור – כלומר בכל דרך עם רמזור הוא יחכה עוד w זמן.

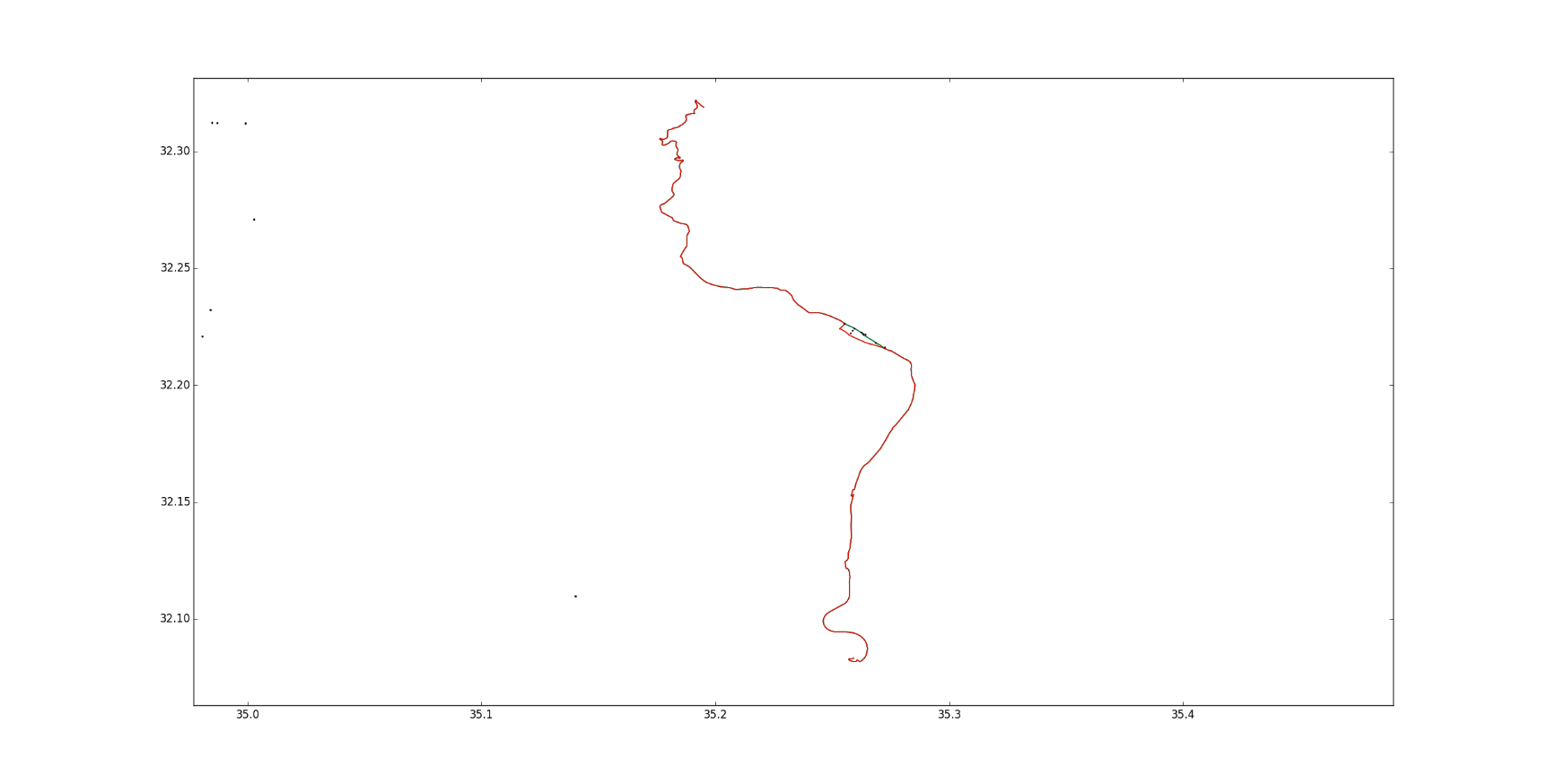
מימשנו פונקציית מחיר חדשה שערכה כמו פונקציית זמן הנסיעה בין הצמתים (כמקודם) אם אין רמזור בצומת, או ערך הפונקציה הקודמת + w אם יש רמזור בצומת.   
פונקציית היוריסטיקה נשארת זהה ובבירור היא נשארת קבילה כי היא מראש הניחה שלא עומדים ברמזורים.  
אין צורך למשקל את פונקציית זמן הנסיעה בין הצמתים, שכן הדבר היחיד המשפיע הוא היחס בין המשקלים שנתנו לפונקציות, וניתן להשיג כל יחס ע"י w בלבד (לא נרצה לתת משקל 0 לזמן הנסיעה בין צמתים, ולכן הכול בסדר).

בדרך זו ניתן להגדיל את w ולהימנע לפי הצורך מדרכים מרומזרות (למשל, לתת ל-w ערך גדול מאוד אומר ש- יחזיר דרך עם מעט רמזורים ככל הניתן, ומבניהן את המסלול הקצר ביותר).

## השוואה לחיפוש עם משקול לרמזורים וללא משקול

ביצענו חיפוש מצומת 171154 אל צומת 123198.   
הקו הכחול מתאר מסלול ללא מחיר לרמזורים, והקו האדום מתאר שקלול של מחיר של 30 שניות לרמזור.

ניתן לראות שאכן בחלק מהמסלול הכחול יש רמזורים, כך שלאחר הוספת המחיר לרמזורים בקו האדום בחר לעקוף אותם.



# חלק רטוב ב'

## שלמות וקבילות אלגוריתם לניווט מובטח

האלגוריתם אינו שלם (ומכך שגם אינו קביל).

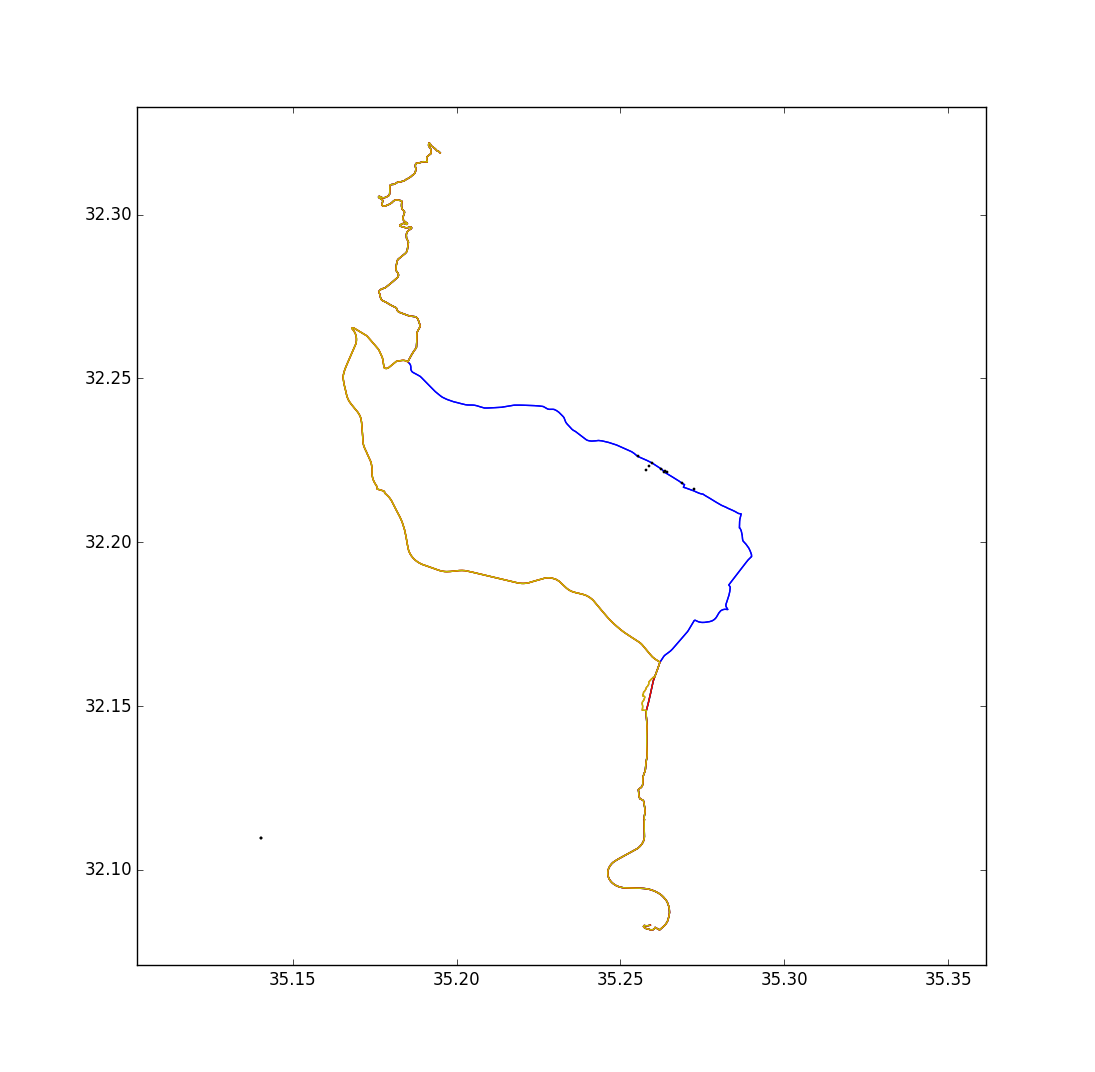
נניח ש-k נבחר להיות 1, לכן אנו בוחרים דגימת זמן אחת (נניח 0) ומוצאים את המסלול האופטימלי לפיה.

ייתכן כי ניווט זה כלל לא יהיה בטוח (למשל עבור כל דגימות הזמן אורכו גדול מ-T), אך דבר זה לא שולל אי קיום ניווט בטוח.  
לדוגמה ייתכן ואם היינו לוקחים דגימת זמן אחרת, היינו מקבלים בה כי המסלול הקצר ביותר היה אכן בטוח (זמן הנסיעה בו קטן מ-T בלפחות מהצמתים).   
בנוסף ייתכן והמסלול הבטוח היחיד לא יהיה קצר ביותר באף אחת מדגימות הזמן (וגם אם נבחר לא נבדוק אותו).

כלומר ייתכן קיום מסלול בטוח אך אי מציאתו ע"י האלגוריתם. זו פגיעה בשלמות ובפרט פגיעה בקבילות.

## הצגה גרפית של Assured Paths

ביצענו חיפוש מובטח עם k=3,p=73,T=47 מהצומת 171154 אל הצומת 123198, והגרלנו את הדורות 21,34,90. קיבלנו את המסלולים הבאים בכחול, אדום וצהוב בהתאמה:



קיבלנו שהמסלול של דור 21 לא חרג מהזמן ב-48 מתוך 100 הדורות, המסלול של דור 34 לא חרג ב-82 מהדורות והמסלול של דור 90 לא חרג ב-79 מהדורות. סה"כ קיבלנו כי המסלול האדום והצהוב בטוחים והכחול לא. (האדום מתחבא מתחת לצהוב).