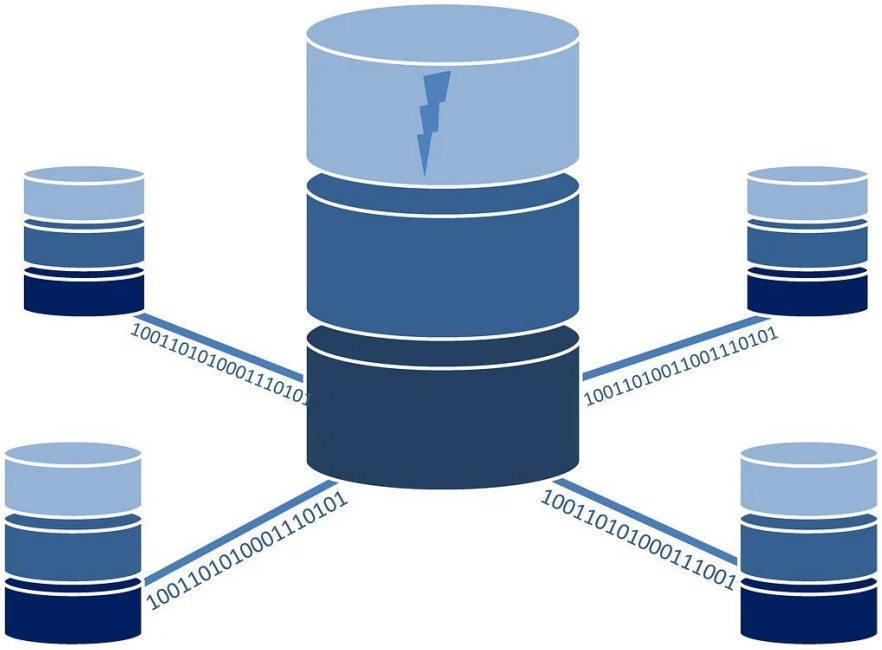
פרויקט במדידת חוסר עקביות במסדי נתונים

דו"ח סיום

Inconsistency Measures



מגישה: רינה קוצ'ירגן

תעודת זהות:316465707

תוכן עניינים

מבוא

הפרויקט נערך במסגרת מחקר[1] של בני קימלפלד ,אסתר ליבשיץ, איהב אליאס, סודיפה רוי ושגב צור העוסק במדידת אי עקביות במסדי נתונים.
במסגרת המחקר נחקרו 6 מדדים המשמשים למדידת חוסר עקביות במסדי נתונים , מטרת החקירה היא לאפיין כל מדד ולבחון את התנהגותו תוך כדי הכנסת הפרות מלאכותיות אל תוך מסד הנתונים.

ששת המדדים שמומשו ונחקרו הם :

1. **Drastic inconsistency value I_D** -

מדד בינארי שערכו 1 במידה והמסד איננו עקבי (קיימות הפרות במסד נתונים) ו-0 אחרת.

2. **Set of all minimal inconsistent subsets of D I_{MI}** –

מספר זוגות הרשומות המופיעות יחד בהפרה כלשהי במסד הנתונים.

3. **Problematic facts I_P** –

מספר הרשומות המשתתפות בהפרה כלשהי במסד הנתונים.

4. **Minimal cost of a sequence of operations that repairs the database I_R** –

המספר המינימאלי של רשומות שיש להסיר בכדי שמסד הנתונים יהיה עקבי. מדד זה מחושב באמצעות תכנון לינארי המוגדר באופן הבא :

$$minimize\ \{\sum_{i=1}^{n} x_i\} \qquad s.t \quad i\ represents\ a\ tuple\ in\ the\ database$$

$$x_i \in \{0,1\}$$

$$x_i + x_j \geq 1 \quad for\ each\ two\ tuples\ i,j\ that\ violate\ a\ constraint$$

5. **Linear relaxation of I_R I_R^{lin}** –

מדד זה כמעט זהה למדד הקודם מלבד הקלה על התחום האפשרי עבור כל x_i .

$$minimize\ \{\sum_{i=1}^{n} x_i\} \qquad s.t \quad i\ represents\ a\ tuple\ in\ the\ database$$

$$0 \leq x_i \leq 1$$

$$x_i + x_j \geq 1 \quad for\ each\ two\ tuples\ i,j\ that\ violate\ a\ constraint$$

[1] Principles of Progress Indicators for Database Repairing, Ester Livshits, Ihab F. Ilyas, Benny Kimelfeld, Sudeepa Roy, 13 Apr 2019

6. I_{MC} – **The set of all maximal consistent subsets of D**

מדד זה מתבסס על גרף הקונפליקטים של מסד הנתונים. גרף הקונפליקטים הוא גרף בו הצמתים מהווים רשומות וישנה קשת בין שני צמתים המייצגים רשומות שמשתתפות בהפרה כלשהי זו עם זו.

עבור הגרף המשלים לגרף הקונפליקטים – כלומר הגרף בו ישנה קשת בין שני צמתים המייצגים רשומות שאינן משתתפות בהפרה זו עם זו , המדד מחשב את המספר המקסימלי של קליקות בגרף.

- סביבת הפיתוח בה השתמשתי לכתיבת הקוד והרצתו היא **jupyter lab** גרסה 1.2.6

- הקוד נכתב בשפת פייתון גרסה 3.7

- לצורך מימוש המדדים I_R, I_R^{lin} השתמשתי בכלי המאפשר פתירת בעיות **בתכנות לינארי** הנקרא **Gurobi optimizer** גרסה 9.0

מהו **תכנות לינארי** ?

תכנות לינארי (Linear programming) הוא למעשה מציאת פתרון אופטימלי אשר מביא למינימום/ למקסימום את פונקציית המטרה של התוכנית.
בעיית תכנות לינארי מורכבת ממספר מאפיינים חשובים המאפשרים את פתרונה:

1. קבוצת משתנים (Variants) : $x_1, .., x_n$ כאשר לכל משתנה יש טווח (למשל: בינארי או רציף בתחום כלשהו או קבוצת ערכים סופית)

2. קבוצת אי שוויוניים (Constraints) מהצורה : $x_i + x_j + x_m + .. \leq Y$

3. פונקציית מטרה : הערך אותו אנו מעניינים להביא למינימום/למקסימום , למשל :

$$minimize \ \{\sum_{i=1}^{n} x_i\}$$

בכדי לפתור את בעיית התכנות הלינארי יש להגדיר כל אחד מהמאפיינים הללו באמצעות פונקציות ייעודיות של Gurobi ולבסוף ניתן לבצע את החישוב באמצעות פונקציית optimize .

- לצורך מימוש המדד I_{MC} המחשב את מספר הקליקות המקסימלי בגרף המשלים לגרף הקונפליקטים השתמשתי באלגוריתם parallel enum[2]
האלגוריתם מקבל כקלט גרף בפורמט NDE ומחשב את מספר הקליקות המקסימלי בגרף זה.

- ספריות נוספות שהשתמשתי בהן :

Pandas – ספרייה המשמשת לאנליזה ומניפולציות על מידע . מבנה הנתונים שמאחסן את המידע מכונה dataframe והוא מאפשר מגוון של פעולות על מידע.
באמצעות הdataframes טענתי את מסד הנתונים מהCSV לקוד ולאחר מכן השתמשתי בפונקציות הייעודיות להוצאת רשומות, עדכון רשומות , מניה ועוד.

[2] Parallel enum algorithm GitHub

Pandasql - ספרייה המאפשרת הרצת שאילתות בשפת SQL על dataframes של Pandas. באמצעות כלי זה הרצתי שאילתות המוצאות את כל זוגות הרשומות המופיעות בהפרה זו עם זו וכל הרשומות המופיעות בהפרה כלשהי.

matplotlib – ספרייה המשמשת בתור כלי לוויזואליזציה של נתונים למשל: גרפים, תמונות, צורות ועוד. השתמשתי בכלי זה בפרויקט לצורך הצגת פלטי המדדים, הקוד מייצר הפרה במסד הנתונים ומעדכן את הרשומה עם ההפרה , לאחר כל שינוי המדדים מחושבים מחדש ולכל מדד שחושב התוכנית מייצרת גרף בו ציר ה – x הוא מספר השינויים במסד נתונים וציר ה – y הוא ערך המדד . ניתן לצפות בדוגמאות בגרפים ותוצאות

ipywidgets – היא ספרייה של Jupyter המאפשרת הוספה של ווידג'טים למשל: תיבת טקסט, בחירה מרובה, כפתורי רדיו, אנימציות, רכיבי HTML ועוד. באמצעות ספרייה זו יצרתי חווית קלט אינטראקטיבית למשתמש המעוניין להריץ את הקוד דרך הjupyter lab כפי שאציג בקלט לתוכנית

random – ספרייה של פייתון שבאמצעותה שגב ואני מימשנו את האלגוריתם שמייצר את ההפרות, כחלק מיצירת ההפרה התבקשנו להגריל ערכים לרשומות כדי ליצור את ההפרות.

Numpy - ספרייה המאפשר חישובים מדעיים באמצעות פייתון.

Re – ספרייה המאפשר פעולות על ביטויים רגולריים. השתמשתי בספרייה בכדי לבצע מניפולציות על מחרוזות ביתר קלות.

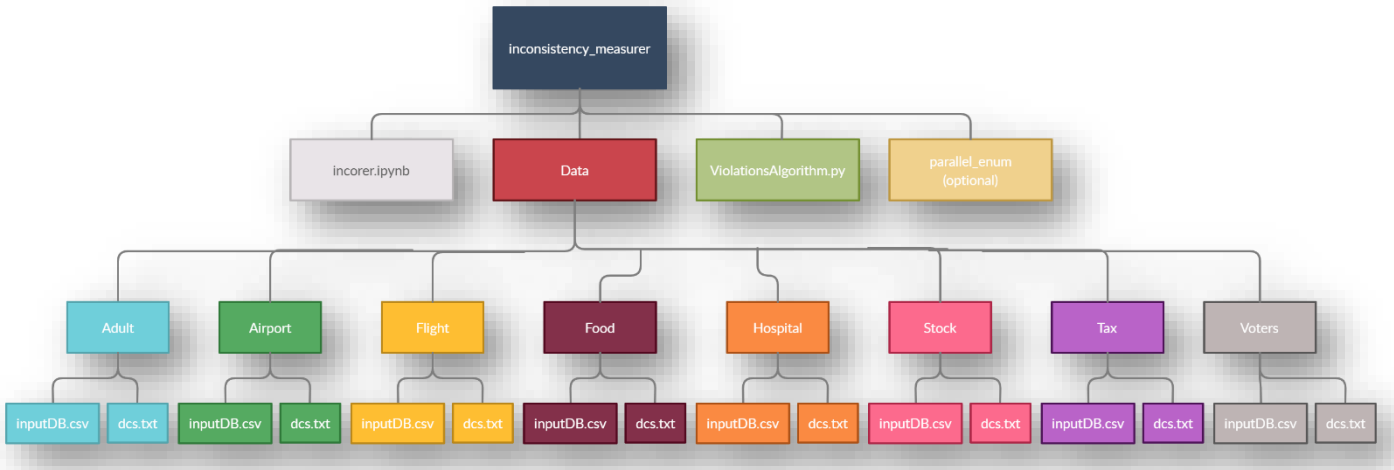
Os – ספרייה בפייתון המאפשרת מניפולציות הקשורות למערכת ההפעלה. השתמשתי בספרייה בכדי למצוא כתובות של קבצים בתיקיית הפרויקט במהלך הקוד ובנוסף כדי לקרוא , לכתוב , ליצור ולסגור קבצים.

Subprocess – ספרייה המאפשרת יצירת תהליכים חדשים והרצת תוכניות. השתמשתי בספרייה כדי להריץ את האלגוריתם parallel_enum[3] המחשב את המדד I_{MC} (מקסימום קליקות בגרף המשלים לגרף הקונפליקטים) לאחר ההרצה חילצתי את הפלט והחזרתי את ערך המדד.

Time – ספרייה בפייתון המאפשרת פעולות הקשורות בזמן. השתמשתי בספרייה כדי לחשב את זמני הריצה של כל אחד מהמדדים ובפרט זמן הריצה של חישוב המדדים הכולל על המסד הנתונים המסוים.

[3] Parallel_enum algorithm GitHub

הקלט לתוכנית



התוכנית מצפה להכיל בתיקיית הפרויקט בתיקיית Data/database_name שני קבצים עיקריים לכל מסד נתונים :

1. **מסד נתונים בפורמט CSV**

2. **קובץ תלויות על המסד**

כל תלות היא מסוג Denial constraint כלומר : $\forall t \neg(P_1, .., P_m)$

התלויות יופיעו בשורות נפרדות באופן הבא :

not(t1.City!=t2.City&t1.Location=t2.Location)

not(t1.Longitude!=t2.Longitude&t1.Location=t2.Location)

הקלט לתוכנית נוצר באמצעות HTML וסיפריית ipywidgets בכדי לספק למשתמש חוויה אינטראקטיבית ולבחור דרכים שונות לחשב את המדדים השונים .

כאן המשתמש בוחר את שם המסד שברצונו להריץ – אחד או יותר מהמסדים המסופקים עם הפרויקט או מסד אחר.

```
HelloNewUser()

Welcome to the inconsistency measurer

Please specify the databases you wish to compute the measures on (seperated by ',')
The databases included are: Adult,Airport,Flight,Food,Hospital,Stock,Tax,Voters
```

השלב הבא הוא לבחור האם להריץ את אלגוריתם ההפרות מספר פעמים ולחשב את המדדים לאחר כל הרצה.

```
Welcome to the inconsistency measurer

Please specify the databases you wish to compute the measures on (seperated by ',')
The databases included are: Adult,Airport,Flight,Food,Hospital,Stock,Tax,Voters
 Stock,Voters,Tax

Do you wish to run a simulation that introduces random violations in the database? [y/n]
Default value is: y
```

במידה והמשתמש בחר להריץ את האלגוריתם עליו לספק את מספר הפעמים שהאלגוריתם ייצר הפרות ויעדכן את מסד הנתונים

```
Welcome to the inconsistency measurer

Please specify the databases you wish to compute the measures on (seperated by ',')
The databases included are: Adult,Airport,Flight,Food,Hospital,Stock,Tax,Voters
 Stock,Voters,Tax

Do you wish to run a simulation that introduces random violations in the database? [y/n]
Default value is: y
 y

Please specify the number of iterations of the simulation
Default value is: 100
```

בעת, המשתמש יבחר את המדדים שברצונו לחשב . ההודעה המוצגת בהמשך מיועדת לאלו הרוצים להריץ את המדד I_{MC} שמריץ אלגוריתם לחישוב מספר קלילות בגרף .

לאחר מכן יש ללחוץ על המשך בכדי להריץ את המדדים

```
Please choose the measures you wish to compute:
    ☐ I_D
    ☐ I_MI
    ☐ I_P
    ☐ I_R
    ☐ I_lin_R
    ☐ I_MC

This message only applies in case you wish to compute the I_MC measure

Please make sure :
1. To build the parallel_enum project inside the current folder
2. that the file text_ui created by parallel_enum is located in /parallel_enum/build/
For additional instructions regarding the parallel_enum algorithm, please visit:
parallel_enum github
✔ Proceed
```

כאשר התוכנית מתחילה לרוץ מודפסת הודעה למסך יחד עם מד התקדמות

```
---
Starting tests ['Stock', 'Voters', 'Tax'] from database inputDB.csv
with the following measurers: ['I_D', 'I_MI']; iterationsNum = 100
---
Test Stock : running 100 iterations; startTime:1601661709.264454
Computing...
```

לאחר סיום הרצת המדדים על כל מסד ,הפלט מופיע בתיקייה עם שם המסד תחת השם מצוין . בתיקייה ניתן למצוא את הגרפים שנוצרו יחד עם קבצים המתארים את זמני הריצה ותוצאות המדדים.

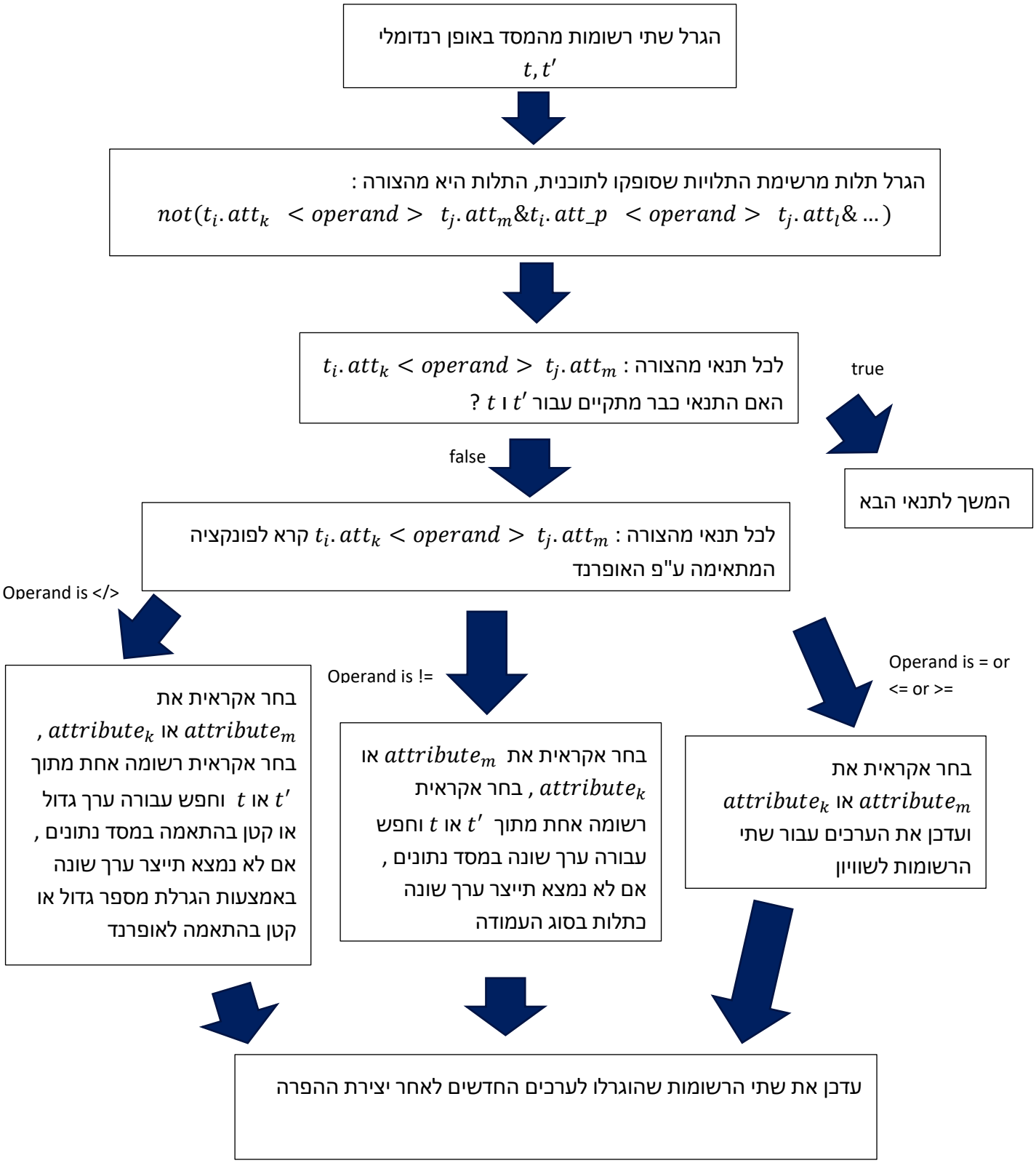
```
---
Starting tests ['Stock', 'Voters', 'Tax'] from database inputDB.csv
with the following measurers: ['I_D', 'I_MI']; iterationsNum = 100
---
Test Stock : running 100 iterations; startTime:1601661709.264454
Computing...
Test Stock : runTime = 1601661729.3201776
Test Stock finished, preparing the results.
End of test Stock; total time = 20.420215368270874
Computation finished, outputs can be found in Data/Stock/1601661709.264454_results

Test Voters : running 100 iterations; startTime:1601661729.6866636
Computing...
Test Voters : runTime = 1601662057.3623223
Test Voters finished, preparing the results.
End of test Voters; total time = 328.203813791275
Computation finished, outputs can be found in Data/Voters/1601661729.6876616_results

Test Tax : running 100 iterations; startTime:1601662057.8924754
Computing...
Test Tax : runTime = 1601662095.2310407
Test Tax finished, preparing the results.
End of test Tax; total time = 37.607834815979004
Computation finished, outputs can be found in Data/Tax/1601662057.8934703_results
```

אלגוריתם יצירת ההפרות

לצורך חישוב המדדים נדרשנו לייצר הפרות במסד הנתונים , לצורך כך שגב צור ואני כתבנו אלגוריתם המייצר הפרות ומכניס אותן למסד הנתונים .
(האלגוריתם נמצא בקובץ ViolationsAlgorithm.py)



מימוש המדדים

בחלק זה אסביר את האופן שבו מימשתי את המדדים השונים

ראשית, אתאר שתי פונקציות מרכזיות המאפשרות את מציאת ההפרות במסד הנתונים ומשמשות לחישוב המדדים בהמשך :

1. **build_dynamic_queries** -

הפונקציה מקבלת קבוצה של מחרוזות כאשר כל מחרוזת מהווה תלות כלשהי (constraint) מקובץ התלויות. כל תלות היא מהצורה :

$$not(t_i.att_k \ < operand > \ \ t_j.att_m \& t_i.att_p \ < operand > \ \ t_j.att_l \& \dots)$$

$$i, j \in \{1,2\}, operand \in \{=, !=, <, >\}$$

ובנוסף הפונקציה מקבלת את ה - dataframe של מסד הנתונים.

הפונקציה "תנקה" את המחרוזות של התלויות מתווים מיותרים ותבנה שתי שאילתות דינאמיות.

1. **unionOfAllPairs** – מחרוזת של שאילתת SQL שמבצעת איחוד בין כל זוגות הרשומות המשתתפות בהפרה כלשהי זו עם זו.

השאילתה תהיה מהצורה :

$$SELECT\ t1.rowid\ as\ t1ctid\ , t2.rowid\ as\ t2ctid$$

$$FROM\ df\ t1, df\ t2$$

$$WHERE\ t_i.att_k < operand > t_j.att_m\ AND\ t_i.att_p < operand > \ \ t_j.att_l\ AND$$

$$\dots AND\ t1.rowid! = t2.rowid$$

$$UNION\ SELECT\ t1.rowid\ as\ t1ctid\ , t2.rowid\ as\ t2ctid\ FROM\ df\ t1, df\ t2$$

$$WHERE\ \ t_i.att_x < operand > t_j.att_y\ AND\ t_i.att_z < operand > \ \ t_j.att_t\ AND$$

$$\dots AND\ t1.rowid! = t2.rowid$$

...

כלומר לכל תלות השאילתה מחשבת את כל הרשומות המשתתפות זו עם זו בהפרתה ולבסוף עושה איחוד עם כל הרשומות שחושבו עד כה עבור שאר התלויות.

2. **unionOfAllTuples** - מחרוזת של שאילתת SQL שמבצעת איחוד בין כל הרשומות המשתתפות בהפרה כלשהי.
השאילתה תהיה מהצורה:

$$SELECT\ DISTINCT\ t1.*\ FROM\ (SELECT\ *$$

$$FROM\ df\ t1, df\ t2$$

$$WHERE\ t_i.att_k < operand > t_j.att_m\ AND\ t_i.att_p < operand > \ t_j.att_l\ AND$$

$$\ldots AND\ t1.rowid! = t2.rowid$$

$$UNION\ SELECT\ *\ FROM\ df\ t1, df\ t2$$

$$WHERE\ \ t_i.att_x < operand > t_j.att_y\ AND\ t_i.att_z < operand > \ t_j.att_t\ AND$$

$$\ldots AND\ t1.rowid! = t2.rowid\ \ldots)\ AS\ A$$

כלומר לכל תלות השאילתה מחשבת את כל הרשומות המשתתפות בהפרתה ולבסוף עושה איחוד עם כל הרשומות שחושבו עד כה עבור שאר התלויות, בסופו של דבר נרצה לסנן כפילויות ולהציג כל רשומה רק פעם אחת לכן נשתמש בdistinct .

שתי השאילתות הדינמיות מהוות את פלט הפונקציה .

2. **constraints_check** -
הפונקציה מקבלת את הdataframe של מסד הנתונים, קבוצת המחרוזות המהוות את התלויות ואת שתי השאילתות הדינאמיות שחושבו בפונקציה **build_dynamic_queries.**
הפונקציה תריץ את השאילתות הללו באמצעות הפקודה sqldf מהספרייה pandasql ותחזיר את הפלט של שתי השאילתות במשתנים violatingPairs,violatingTuples יחד עם זמני הריצה של כל אחת מהשאילתות.

כעת, לאחר שהצגתי את שתי הפונקציות הללו אציג את המימושים של המדדים :

1. **first_measurer_I_D :**
הפונקציה תקבל את הdataframe המכילה את כל הזוגות המשתתפות בהפרה כלשהי, במידה והdataframe ריק המשמעות היא שהמסד עקבי לכן יוחזר 0 . אחרת , המסד איננו עקבי ויוחזר 1.

2. **Second_measurer_I_MI :**
הפונקציה תקבל את הdataframe המכילה את כל הזוגות המשתתפות בהפרה כלשהי ותחזיר את מספר הרשומות בטבלה זו. (מספר הרשומות מהווה את מספר הזוגות)

3. **third_measurer_I_P :**
הפונקציה תקבל את הdataframe המכילה את כל הרשומות המשתתפות בהפרה כלשהי ותחזיר את מספר הרשומות בטבלה זו.

4. **fourth_measurer_I_R :**
הפונקציה תקבל את הdataframe (uniquePairsDF) המכילה את כל הזוגות המשתתפות בהפרה כלשהי ותשתמש בכלי לתכנון הלינארי Gurobi optimizer בכדי למצוא את המספר המינימאלי של הרשומות שיש למחוק מהמסד כדי שהמסד יהיה עקבי.
הגדרת בעיית התכנון הלינארי מתבצעת באופן הבא :

variants : לכל רשומה i נגדיר משתנה בינארי (בעל ערך 0 או 1) x_i ייחודי .

constraints : לכל שתי רשומות i, j כך ש $i, j \in uniquePairsDF$
(כלומר הרשומות משתתפות בהפרה כלשהי זו עם זו) נגדיר constraint מהצורה :

$$x_i + x_j \geq 1$$

objective function : פונקציית המטרה היא $\sum_{i=1}^{n} x_i$ כאשר n הוא מספר הרשומות במסד הנתונים והמטרה היא להביא למינימום פונקציה זו.
הפונקציה תקרא לפונקציית optimize של Gurobi ותחזיר כפלט את הערך שהתקבל יחד עם זמן הריצה של הפונקציה.

5. **fifth_measurer_I_lin_R :**
הפונקציה הזו כמעט זהה לפונקציה המחשבת את המדד I_R מלבד הגדרת ה variants שיוגדרו להיות רציפים בין 0 ל -1 כלומר: $0 \leq x_i \leq 1$ ולא בינאריים.
באותו אופן כמו הפונקציה הקודמת, הפונקציה הזו תחזיר את הערך האופטימלי שחושב וזמן הריצה של הפונקציה.

6. **sixth_measurer_I_MC** :
הפונקציה מקבלת את הכתובת המלאה לתיקיית הפרויקט ואת הdataframe (uniquePairsDF) המכילה את כל הזוגות המשתתפות בהפרה כלשהי .
הפונקציה תיצור את הגרף המשלים לגרף הקונפליקטים כלומר כל צומת מהווה רשומה כלשהי במסד הנתונים וישנה קשת בין שני צמתים רק במידה והם לא משתתפים יחד בהפרה . הגרף הנוצר יהיה בפורמט nde (node degree edge):

$$number\ of\ nodes\ (i.e\ \ numer\ of\ \ tuples\ in\ the\ databse)$$
$$node\ \ degree\ (for\ each\ node\ in\ a\ seperate\ row)$$
$$node1\ node2\ (edge\ between\ the\ nodes)$$

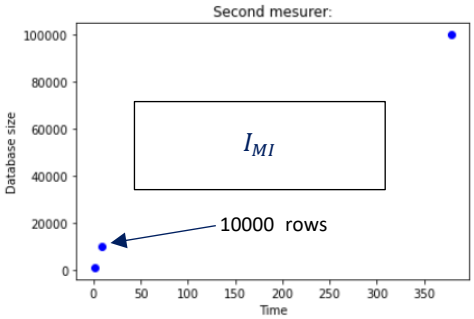
גרף זה יודפס אל תוך קובץ בעל סיומת nde ולאחר יצירתו הפונקציה תריץ את האלגוריתם parallel_enum ותחלץ את הפלט שהתקבל ותחזירו מהפונקציה יחד עם זמן הריצה הכולל.

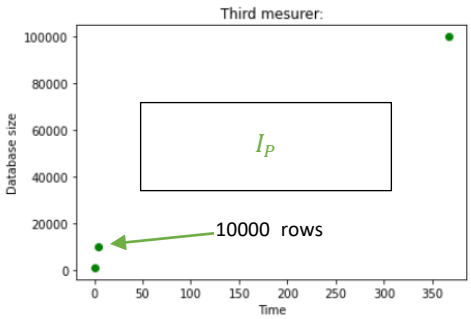
תחילה, הרצתי את הקוד על מסדי נתונים המכילים כמיליון רשומות.
אך נתקלתי בבעיה כאשר שמתי לב שהקוד איננו מסתיים. לאחר בדיקה לעומק של השאילתות שמוצאות את הרשומות המשתתפות בהפרה ייעלתי את ריצתן ע"י החלפת השאילתות משאילתות שמשתמשות ב or לשאילתות המשתמשות ב $union$ ועושות למעשה איחוד בין טבלאות רבות.
לאחר בדיקה ברחבי האינטרנט[4], התוודעתי לכך ש $union$ משפר את זמן הריצה של השאילתה ולעומת זאת שאילתה המכילה תנאים רבים עם or רצה יותר לאט.
בהמשך, ניסיתי גם להוסיף לשאילתות אינדקסים כדי לאפשר זמן ריצה מהיר יותר (אינדקסים עבור עמודות במסד הנתונים מאפשרים מיון של הערכים של העמודות הללו וכך תתאפשר גישה מהירה יותר בעת חיפוש) . הוספת האינדקסים אכן שיפרה במעט את זמן הריצה כפי שהוא מוצג ב cost (העלות המשוערכת של זמן הריצה של השאילתה) .

Data Output | Explain | Messages | Notifications

	QUERY PLAN text
1	Hash Join (cost=99831.05..308066618.99 rows=16498067180 width=150)
2	Hash Cond: ((t1.areacode)::text = (t2.areacode)::text)
3	Join Filter: (((t1.state)::text <> (t2.state)::text) AND (t1.ctid <> t2.ctid))
4	-> Seq Scan on inputdb t1 (cost=0.00..47487.02 rows=2000002 width=81)
5	-> Hash (cost=47487.02..47487.02 rows=2000002 width=81)
6	-> Seq Scan on inputdb t2 (cost=0.00..47487.02 rows=2000002 width=81)

ניתן לראות בגרפים אלו את ההבדלים בזמני הריצה של הקוד על מסד הנתונים Tax :
ניתן לראות כאן קפיצה משמעותית של זמן הריצה על 2 מדדים אלו בין מסד נתונים המכיל רק 10,000 שורות שרץ פחות מ10 שניות לבין מסד נתונים המכיל 100,000 רשומות שרץ ב400 שניות (כ- 6 דקות)





לאחר הוספת המדדים הנוספים לקוד זמן הריצה הפך להיות ארוך מדיי ולכן לא היה מנוס מלהקטין את מספר השורות במסד הנתונים ל 1000 שורות.

[4] Using Union Instead of OR , Posted on October 7, 2012 by Derek Dieter

גרפים ותוצאות

בחלק זה אציג את תוצאות המדדים של הרצת הפרויקט על כל מסדי הנתונים הנכללים בתיקיית הפרויקט המכילים 1000 שורות :

$Adult, Airport, Flight, Hospital, Stock, Voters, Tax, Food$

מספר ההפרות שהוכנסו הוא 50

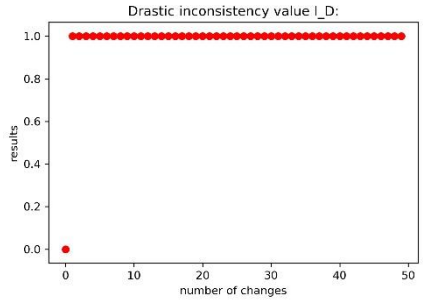
מקרא : $Drastic\ inconsistency\ I_D$

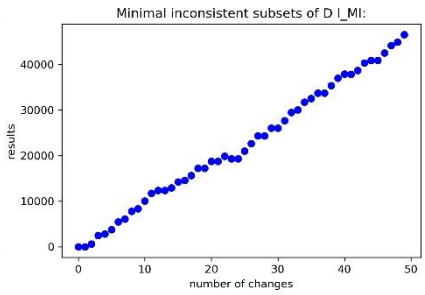
$Minimal\ inconsistent\ subsets\ of\ D\ I_{MI}$ $Problematic\ facts\ I_P$

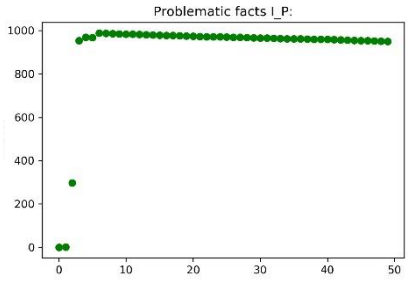
$Minimal\ tuples\ deletions\ I_R$ $Linear\ relaxation\ of\ the\ previous\ measurer\ I_R^{lin}$

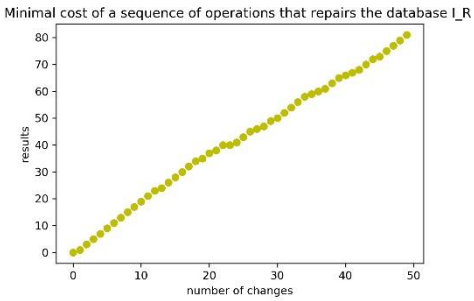
$Maximal\ cliques\ I_{MC}$

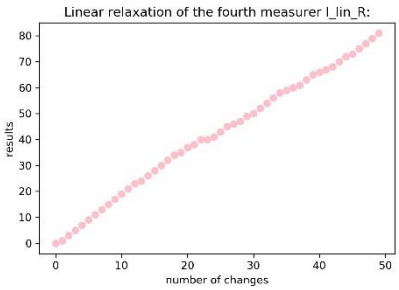
$Voters$

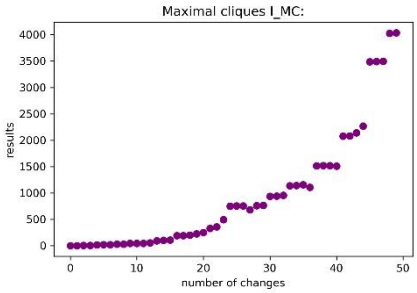








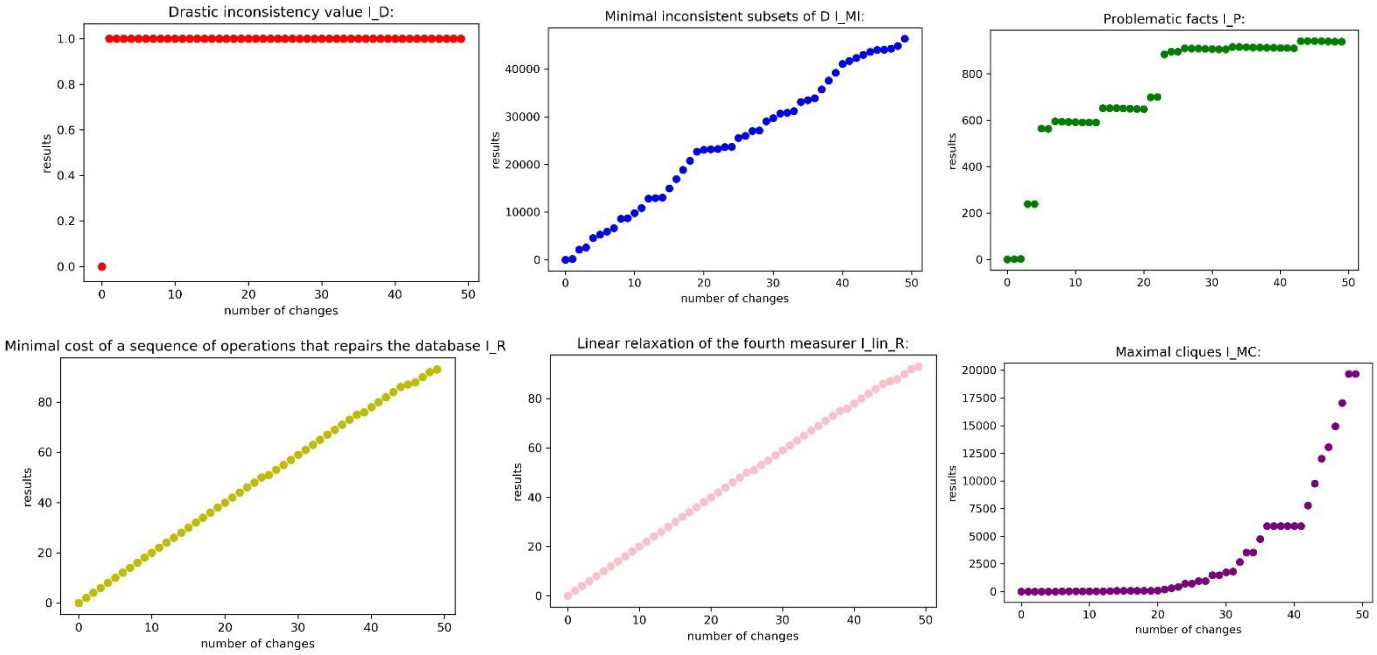




זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.4404	0.6293	1.2769	1.0027	3.6970	300.9803

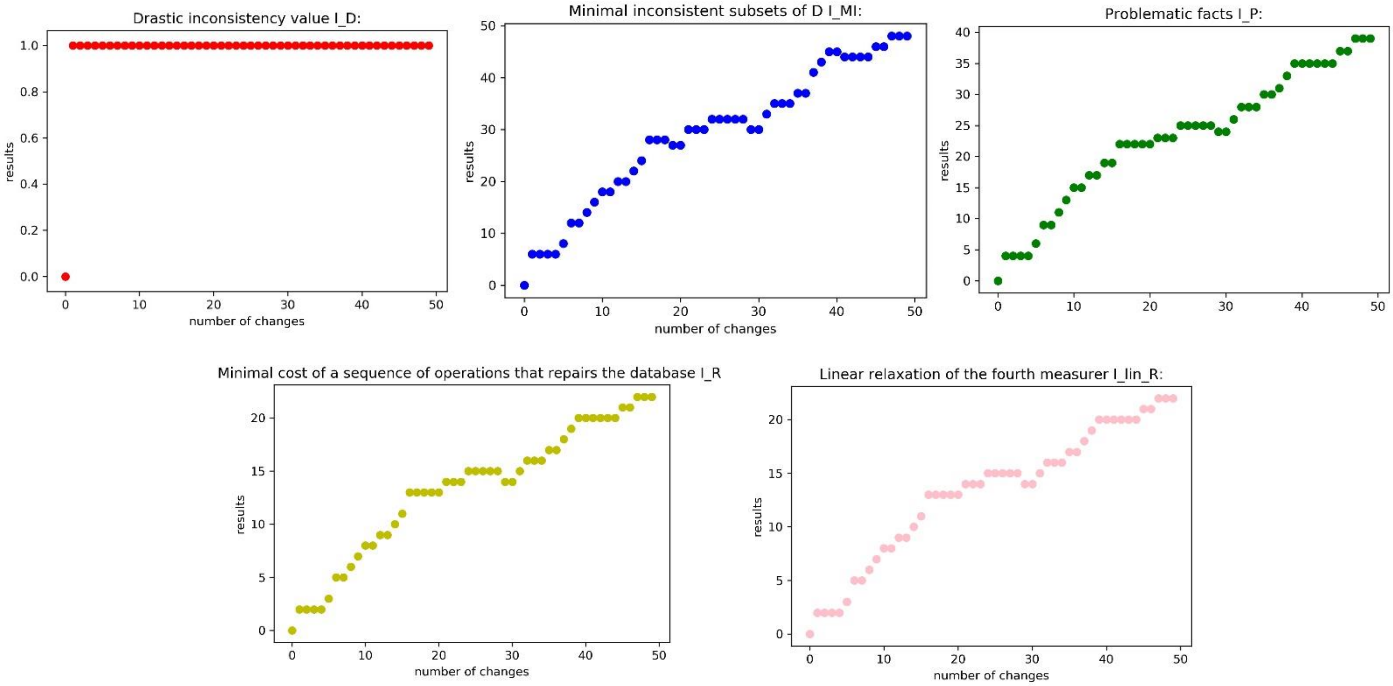
Adult



זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.2377	0.2905	1.1616	0.8595	9.6623	589.4423

Food

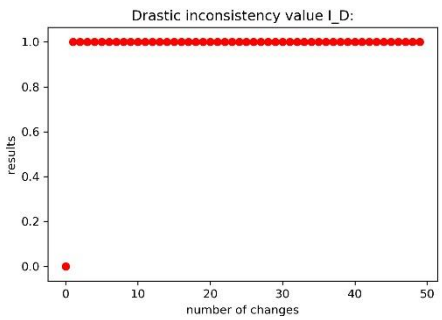


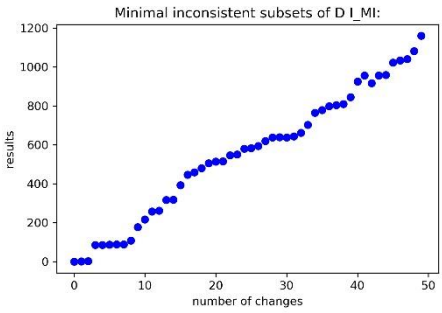
זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

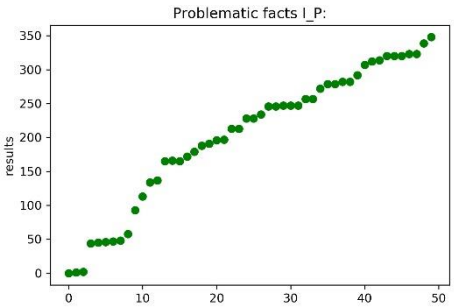
I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.1544	0.1566	0.1564	0.1555	$time\ out$[5]	22.6719

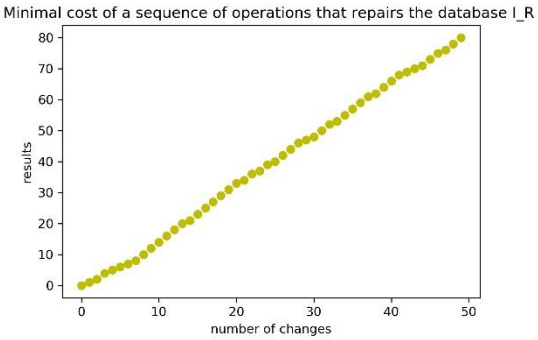
[5] Time out means that the running time is more than 24 hours

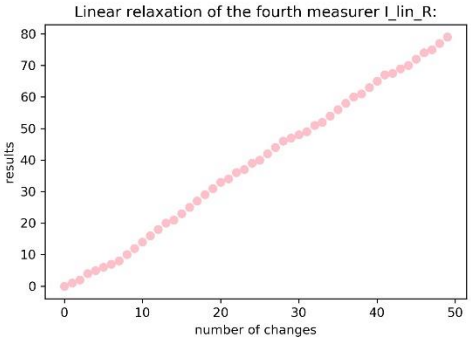
Tax









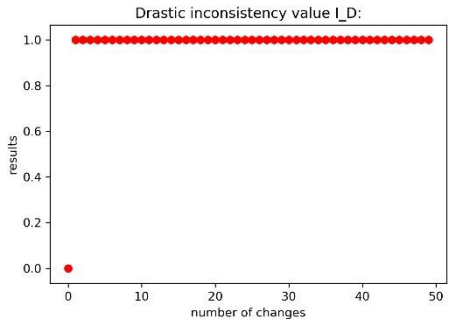


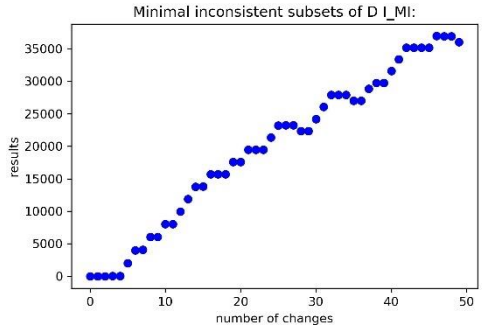
זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

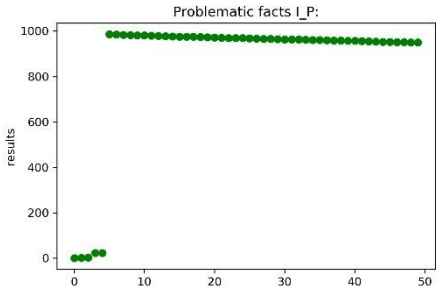
I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.0704	0.0738	0.0853	0.0841	$time\ out$[6]	15.4795

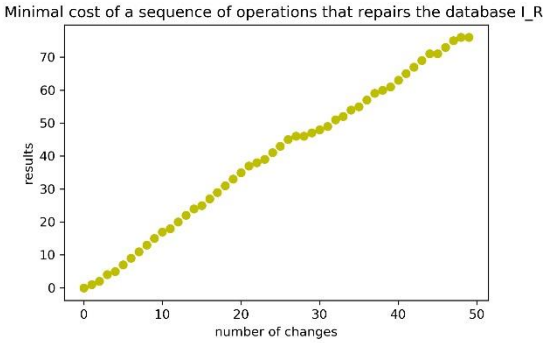
[6] Time out means that the running time is more than 24 hours

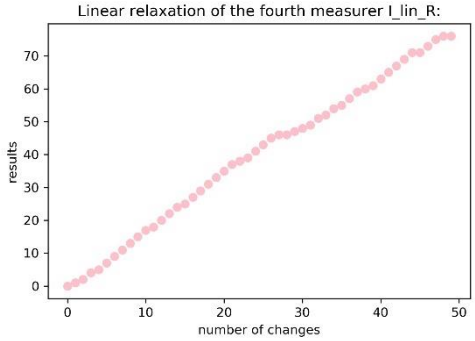
Airport









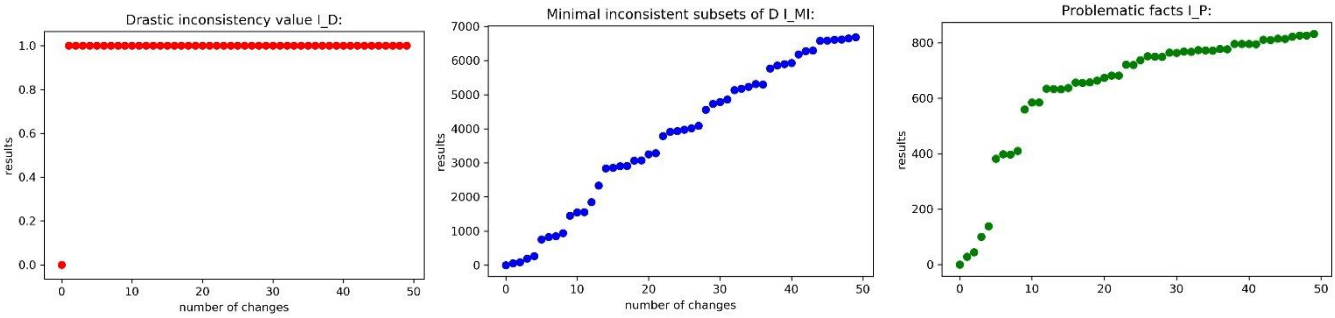


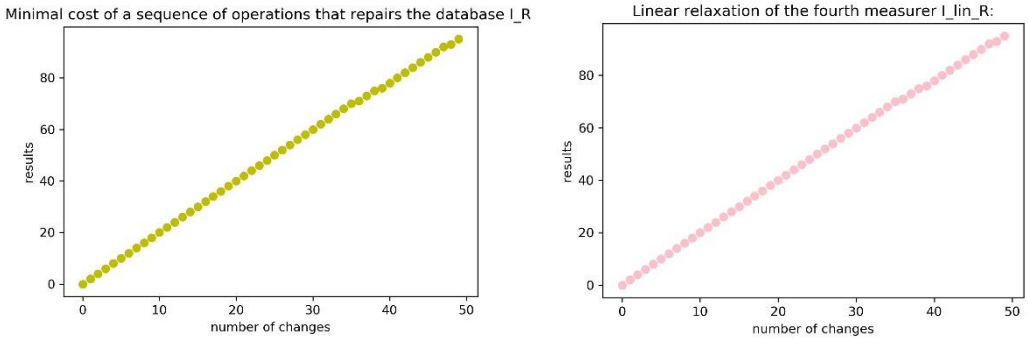
זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.2170	0.2795	0.9614	0.6876	$time\ out$[7]	94.8695

[7] Time out means that the running time is more than 24 hours

Flight



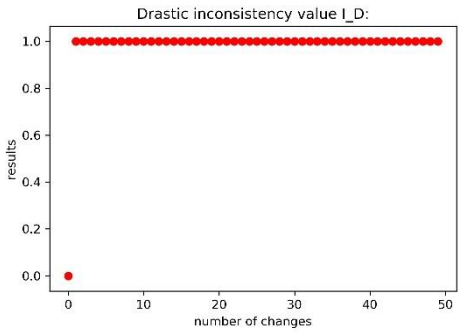


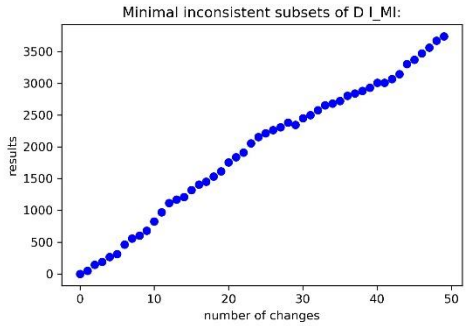
זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

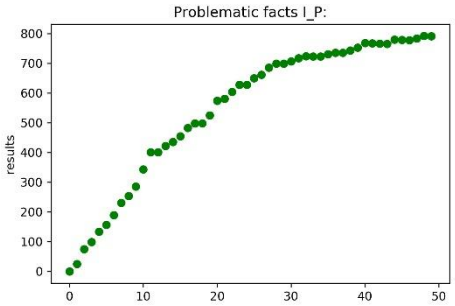
I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.1844	0.2094	0.3030	0.2738	$time\ out$[8]	37.5367

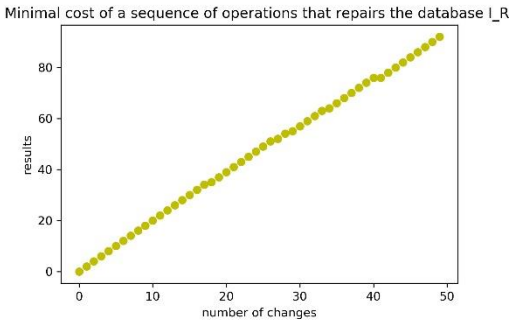
[8] Time out means that the running time is more than 24 hours

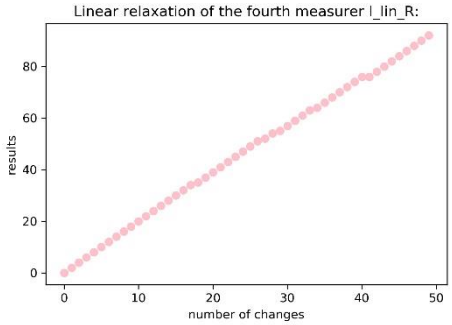
Hospital









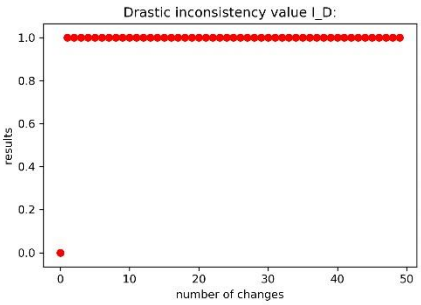


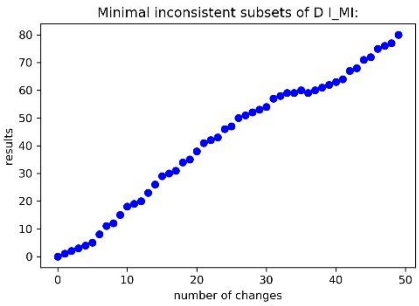
זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

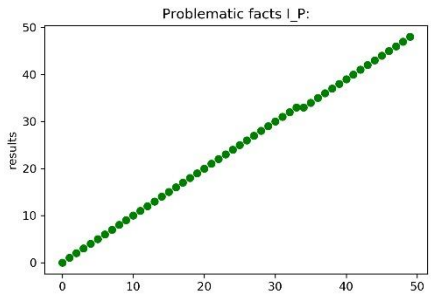
I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.0916	0.1051	0.1541	0.1385	$time\ out$[9]	22.2117

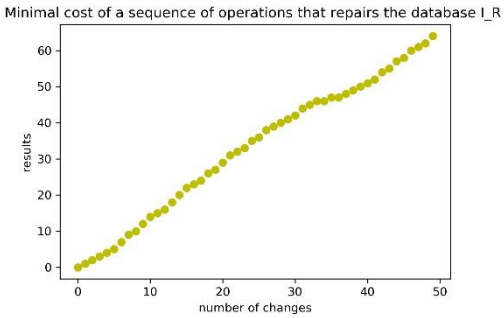
[9] Time out means that the running time is more than 24 hours

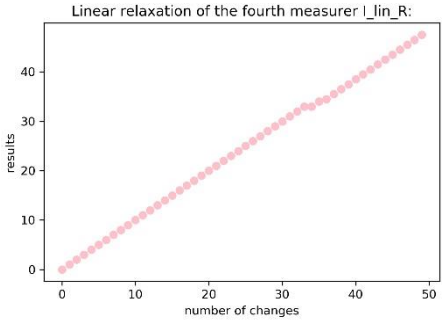
Stock











זמני ריצה ממוצעים (בשניות) עבור כל אחד מהמדדים וסך זמן הריצה הכולל :

I_{MI}	I_P	I_R	I_R^{lin}	I_{MC}	$Total\ run\ time$
0.0328	0.0313	0.0344	0.0342	$time\ out$[10]	9.8281

[10] Time out means that the running time is more than 24 hours

מתוך ההתנהגות של הגרפים ראינו שלכל מדד ישנה התנהגות שונה על פני מסדים שונים.
המדד הטריוויאלי ביותר I_D התנהג כמצופה והגיע לערך 1 לאחר הכנסת ההפרה הראשונה ונשאר יציב לאורך כל הריצה. הסקנו מכך שמדד זה לא מספק מספיק מידע על מידת חוסר העקביות במסד הנתונים .
המדד I_{MI} נטה להיות יותר לינארי ביחס למדדים האחרים.
המדד I_P לעומתו, בדרך כלל הגיע לרוויה (1000 רשומות המשתתפות בהפרה) מהר מאוד.
ניתן לראות שהמדדים I_R, I_R^{lin} לאורך כל ההרצות שמרו על תבנית כמעט לינארית ויציבה.
ולבסוף, המדד I_{MC} לא היה יציב במיוחד ולא הראה תבנית אחידה על פני המסדים שעליהם הוא רץ.

אך התנהגות הגרפים איננה הדבר היחיד שחקרנו, זמני הריצה של המדדים השונים הם חשובים לא פחות. זמני הריצה חשובים מאוד כדי להכריע איזה מדד מתאים לשימוש של המשתמש. למשל, אם המשתמש זקוק לנתונים אודות עקביות המסד באופן כמעט מידי חלק מהמדדים לא יספקו לו תוצאה בפרק זמן סביר.
מדד ה - I_{MC} כפי שניתן לראות בתוצאות, היווה את צוואר הבקבוק של זמן הריצה. על חלק ממסדי הנתונים , מדד זה רץ במשך שעות ואף יותר מיממה.
אחריו המדד I_R שאמנם לא היה ארוך משמעותית מבחינת זמן הריצה שלו אך ניתן לראות בכל הניסויים כי I_R^{lin} היה בעל זמן ריצה קטן יותר בהשוואה אליו, למרות ששניהם היו בעלי תבנית לינארית. נוכל להסיק כי ה relaxation של המדד I_R (כלומר העובדה ש"הקלנו" על הטווח האפשרי עבור המשתנים בבעיית התכנון הלינארי במדד I_R^{lin}) היה מוצלח והביא למדד שרץ בסיבוכיות פולינומיאלית.

המסקנה שניתן להגיע אליה מתוך תוצאות הניסויים היא שהמדד I_R^{lin} הוא מדד יציב, שואף להיות כמעט תמיד לינארי וניתן לחישוב בזמן פולינומיאלי ועל כן הוא נחשב מדד איכותי ומועיל למדידת חוסר עקביות במסדי נתונים.

ביבליוגרפיה

inconsistency-measurer
The project's Google Drive

Principles of Progress Indicators for Database Repairing
Ester Livshits, Ihab F. Ilyas, Benny Kimelfeld, Sudeepa Roy, 13 Apr 2019.

Gurobi optimizer site
Provided me with installation info and code examples for the LP tool.

Ipywidgets documentation
Provided me with usage examples of widgets in jupyter lab.

Python official site
Provided me with usage of all libraries presented in the report.

Pandas official site
Provided me with usage and code examples.

Matplotlib official site
Provided me with usage and code examples.

Parallel_enum algorithm GitHub
The algorithm calculating maximal cliques in a graph, is used for calculating the measurer I_{MC}.

Using Union Instead of OR
Article regarding time optimization of SQL queries using Union instead of OR

Stackoverflow
Provided me with solutions and information which was not found in the previous sources.