Formal Verification Final Project

Ilya Lipovan 208669101

Tomer Lahav 208394452

GitHub link for the codes:
https://github.com/elemenopi/Sokoban_nuXmv.git

Part 1

1. Define an FDS for a general $n \times m$ Sokoban board. Use XSB format to describe the board.
2. Define a general temporal logic specification for a win of the Sokoban board.

1.

Symbol	Definition
@	warehouse keeper
+	warehouse keeper on goal
$	box
*	box on goal
#	wall
.	goal
-	floor

FDS for Sokoban board:

$D = \{V, \theta, \rho, J, C\}$

$V = \{@, \$_{\{1-n\}}\}$

$\theta = \{@ = (x_{@}, y_{@}), \$_1 = (x_{\$1}, y_{\$1}) \dots \$_n = (x_{\$n}, y_{\$n})\}$

$\rho = \{(@' = (x_{@+1}, y_{@}) \,||\, @' = (x_{@-1}, y_{@}) \,||\, @' = (x_{@}, y_{@+1}) \,||\, @' = (x_{@}, y_{@-1}) \;\&\&$
$(\$_t = \$'_t \,||\, \$'_t = (x_{\$t+1}, y_{\$t}) \,||\, \$'_t = (x_{\$t-1}, y_{\$t}) \,||\, \$'_t = (x_{\$t}, y_{\$t+1}) \,||\, \$_t = (x_{\$t}, y_{\$t-1}) \,||\, \$'_t = *_t)$
$for\ t\ \{1 \dots n\}\}$

עבור רו –
עבור השחקן – הוא זז למעלה/למטה/ימינה/שמאלה.
עבור כל כופסא – או שהיא זזה (בגלל שהשחק הזיז אותה) או שהיא נמצאת על מטרה.

$J = \{\$_{1-n} = *,\ \$_{1-n} \neq \#,\ \$_{1-n} \neq @,\ @ \neq \#,\ for\ i\ and\ j: \$_i \neq \$_j\ \}$

כל הכופסאות אין סוף פעמים נמצאות על נקודת מטרה,
אף פעם לא יהיה מצב ש2 כופסאות / כופסא ואיש / כופסא וקיר / איש וקיר על אותה המשבצת בלוח.

$C = \{< @' = \#, @' = @\ >, < @' = \$_i \;\&\&\; \$'_i = \#, @' = @\ >,$

$< \$'_i = \#, \$'_i = \$_i\ >, < \$'_i = \$_j, \$'_i = \$_i\ >\}$

אם האיש הולך לכיוון הקיר – הוא נשאר במקום, כנ"ל לגבי אם הוא מנסה לדחוף כופסה אבל היא צמודה לקיר.
בנוסף אם הכופסה צמודה לכופסה אחרת או כופסה צמודה לקיר – היא נשאר במקום.

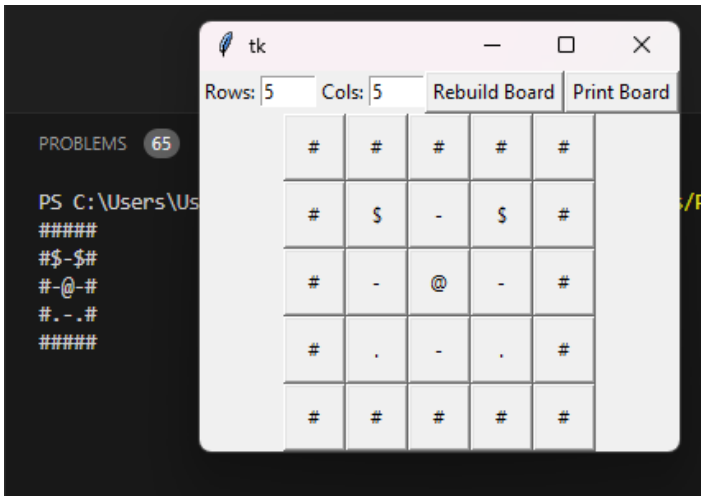
2.
Win condition for Sokoban board:

$$F\big(G(\$_{1-n} = *)\big), n = num\ of\ boxes$$

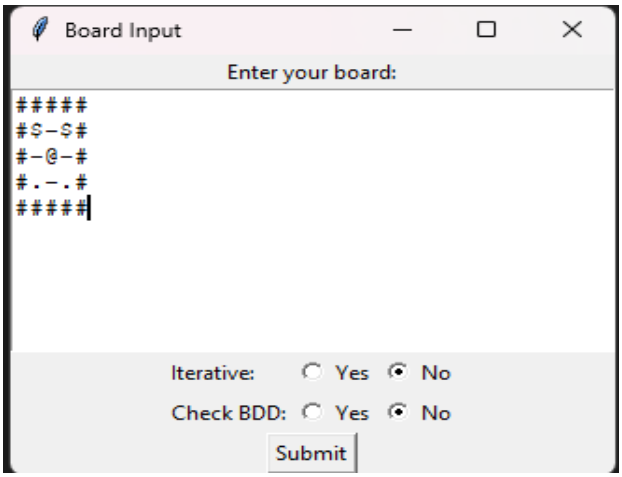
משלב מסויים, תמיד – כל הכופסאות ישבו על מטרה.
כלומר כל הפרמוטציות של כופסא כלשהי שיושבת על אחת המטרות כך שעל כל המטרות יש כופסאות מספקת את התנאי לנצחון.

מחלק זה אנחנו מבצעים אוטומציה ע"י פייתון –
ראשית נגשנו למשימה בכך שניסינו לייצר קוד ללוח אחד של סוקובן ב smv על מנת לייצר את הבסיס לתנאים והחוקים של המשחק ולאחר מכן ביצענו הכללה ואוטומציה בפייתון כך שבנהתן לוח כלשהו הקוד שלנו ידע לקחת ממנו את המיקומים של כל האלמנטים ולייצר קובץ smv.
לאחר כתיבת ה smv, הקוד יריץ את הקובץ ב nuXmv ויראה לנו את התוצאות.

על מנת להקל על כתיבת לוח משחק והכנסתם כאינפוט יצרנו קוד נוסף, שקראנו לו board_to_XSB_gui , קוד זה פותח לוח בגוי בו אפשר למקם כל אלמנט (אדם, כופסא, מטרה) על הלוח וכנלחץ print הוא ידפיס את הלוח כפלט של פייתון.



לאחר שיש לנו פורמט מוכן של לוח נפעיל את הפייתון sokoban.py מ cmd בתוך התקייה בה נמצא nuXmv כדי ש nuXmv ימצא את קובץ ה smv ויצליח להריץ אותו.



נעתיק את הלוח לתוך הגוי ונבחר באפשרות שאנחנו מעוניינים להריץ.
(חשוב לזכור להיות על אנגלית אחרת זה לא ידביק את הלוח על המסך)

Part 2

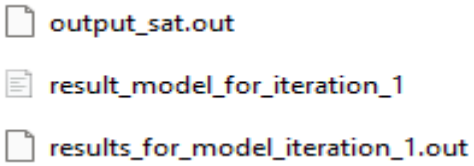
1. Using Python, or another coding language, and your answers to Part 1, automate definition of given input boards into SMV models. These models should contain both the model and the temporal logic formulae defining a win.
2. Run each of these models in nuXmv. Indicate the commands you used to run nuXmv, as well as screenshots of the nuXmv outputs.
3. For each board, indicate if it is winnable. If it is, indicate your winning solution in LURD format.

Part 3

1. Measure performance of nuXmv's BDD and SAT Solver engines on each of the models.
2. Compare the performance of the two engines. Is one engine more efficient than the other?

עבור חלק 2 ו3 נסמן check BDD כדי שנקבל כפלט את הזמנים גם של sat וגם של bdd,
כדיפולט עשינו הרצה של sat.
הגדרנו בדיקה עד ל40 צעדים במידה ולא מוצאים פתרון ללוח.

נקבל כפלט שלושה קבצים חדשים:

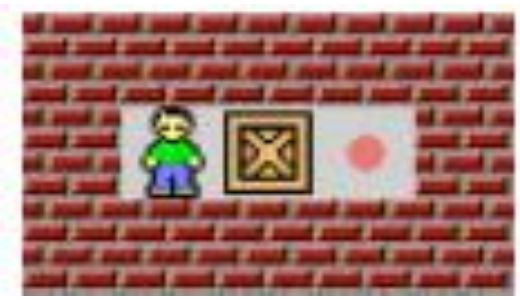


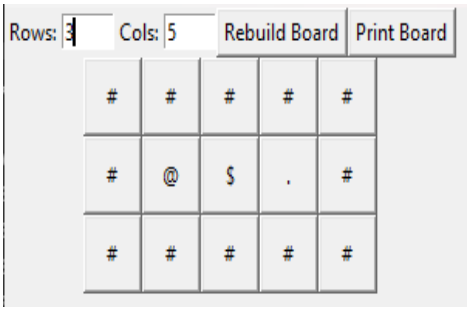
- קובץ result_model_for_iteration זה קובץ ה smv שאנחנו יוצרים אותו נריץ ב nuXmv.
- קובץ output_sat קובץ שמראה את פלט ה nuxmv אחרי ההרצה , כולל בסופו זמן ריצה (sat)
- קובץ result_for_model_iteration קובץ שמראה את תוצאת הריצה בפורמט LURD + זמני ריצה.

**בגלל שעשינו את כל החלקים 2,3,4 באותו הקוד שמות הקבצים כוללים iteration שיהיה רלוונטי בחלק 4.

להלן ההרצות והפלטים של חלק 2 ו3:

ריצה 1:





פלט nuxmv:

-- as demonstrated by the following execution sequence

Trace Description: BMC Counterexample

Trace Type: Counterexample

-> State: 1.1 <-

i_person = 1

j_person = 1

i_box1 = 1

j_box1 = 2

action_person = no-action

boxes_overlap = FALSE

box_on_wall = FALSE

man_on_box = FALSE

man_on_wall = FALSE

j_box_goal1 = 3

i_box_goal1 = 1

M = 5

N = 3

grid[0][0] = 1

grid[0][1] = 1

grid[0][2] = 1

grid[0][3] = 1

grid[0][4] = 1

grid[1][0] = 1

grid[1][1] = 0

grid[1][2] = 0

grid[1][3] = 0

grid[1][4] = 1

grid[2][0] = 1

grid[2][1] = 1

grid[2][2] = 1

grid[2][3] = 1

grid[2][4] = 1

-> State: 1.2 <-

j_person = 2

j_box1 = 3

action_person = right

-> State: 1.3 <-

j_person = 1

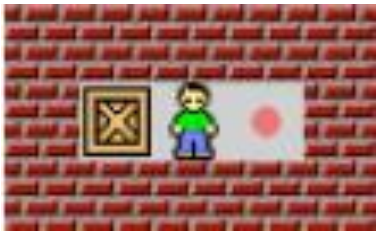
action_person = left

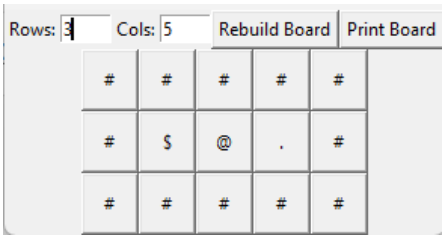
nuXmv > elapse: 0.02 seconds, total: 0.02 seconds

תוצאות:

```
results in lurd format : R
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 0.02 seconds (Total time: 0.02 seconds)
Last checked bound: 1

        runtime results BDD:

        Runtime after check_ltlspec: 0.04 seconds (Total time: 0.04 seconds)
```

ריצה 2:





פלט nuxmv:

nuXmv > nuXmv > nuXmv > nuXmv > -- no counterexample found with bound 0

-- no counterexample found with bound 1

-- no counterexample found with bound 2

-- no counterexample found with bound 3

-- no counterexample found with bound 4

-- no counterexample found with bound 5

……

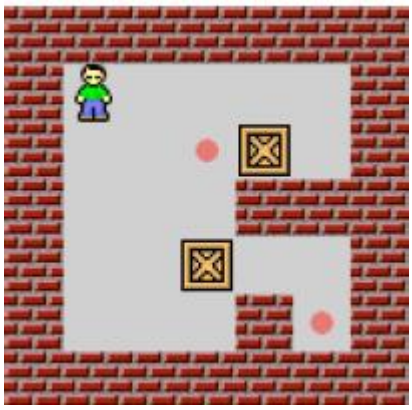
-- no counterexample found with bound 40

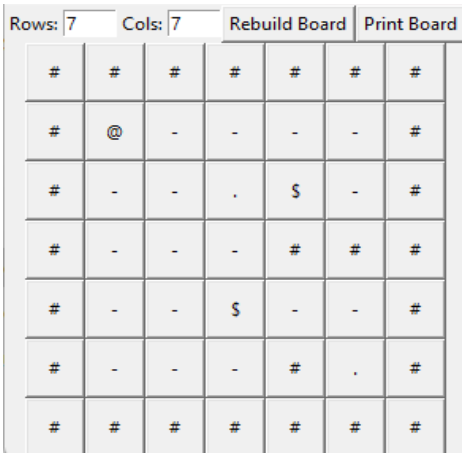
nuXmv > elapse: 1.28 seconds, total: 1.28 seconds

תוצאות:

```
results in lurd format :
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 1.28 seconds (Total time: 1.28 seconds)
Last checked bound: 40

        runtime results BDD:

        Runtime after check_ltlspec: 0.02 seconds (Total time: 0.02 seconds)
```

ריצה 3:





פלט nuxmv:

nuXmv > nuXmv > nuXmv > nuXmv > -- no counterexample found with bound 0

no counterexample found with bound 1 --

no counterexample found with bound 2 --

no counterexample found with bound 3 --

no counterexample found with bound 4 --

no counterexample found with bound 5 --

……..

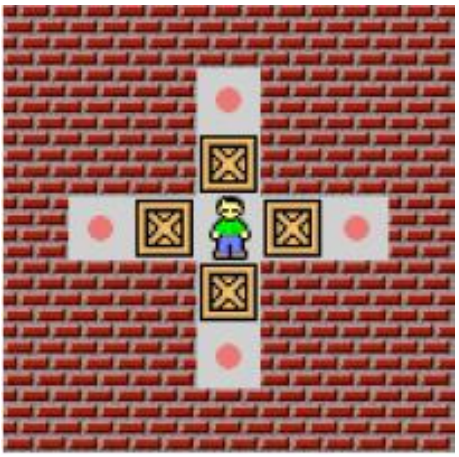
no counterexample found with bound 40 --

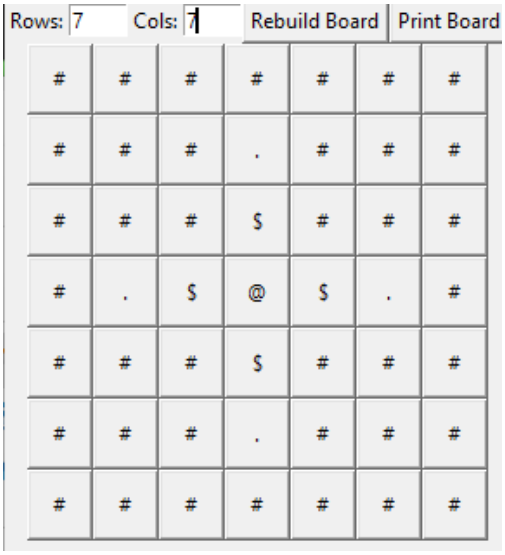
nuXmv > elapse: 136.72 seconds, total: 136.72 seconds

תוצאות:

```
results in lurd format :
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 136.72 seconds (Total time: 136.72 seconds)
Last checked bound: 40

        runtime results BDD:

        Runtime after check_ltlspec: 0.66 seconds (Total time: 0.66 seconds)
```

ריצה 4:





פלט nuXmv:

Trace Type: Counterexample

-> State: 1.1 <-

i_person = 3

j_person = 3

i_box1 = 2

j_box1 = 3

i_box2 = 3

j_box2 = 2

i_box3 = 3

j_box3 = 4

i_box4 = 4

j_box4 = 3

action_person = no-action

boxes_overlap = FALSE

box_on_wall = FALSE

man_on_box = FALSE

man_on_wall = FALSE

j_box_goal4 = 3

i_box_goal4 = 5

j_box_goal3 = 5

i_box_goal3 = 3

j_box_goal2 = 1

i_box_goal2 = 3

j_box_goal1 = 3

i_box_goal1 = 1

M = 7

N = 7

grid[0][0] = 1

grid[0][1] = 1

grid[0][2] = 1

grid[0][3] = 1

grid[0][4] = 1

grid[0][5] = 1

grid[0][6] = 1

grid[1][0] = 1

grid[1][1] = 1

grid[1][2] = 1

grid[1][3] = 0

grid[1][4] = 1

grid[1][5] = 1

grid[1][6] = 1

grid[2][0] = 1

grid[2][1] = 1

grid[2][2] = 1

grid[2][3] = 0

grid[2][4] = 1

grid[2][5] = 1

grid[2][6] = 1

grid[3][0] = 1

grid[3][1] = 0

grid[3][2] = 0

grid[3][3] = 0

grid[3][4] = 0

grid[3][5] = 0

grid[3][6] = 1

grid[4][0] = 1

grid[4][1] = 1

grid[4][2] = 1

grid[4][3] = 0

grid[4][4] = 1

grid[4][5] = 1

grid[4][6] = 1

grid[5][0] = 1

grid[5][1] = 1

grid[5][2] = 1

grid[5][3] = 0

grid[5][4] = 1

grid[5][5] = 1

grid[5][6] = 1

grid[6][0] = 1

grid[6][1] = 1

grid[6][2] = 1

grid[6][3] = 1

grid[6][4] = 1

grid[6][5] = 1

grid[6][6] = 1

-> State: 1.2 <-

i_person = 4

i_box4 = 5

action_person = down

-> State: 1.3 <-

i_person = 3

action_person = up

-> State: 1.4 <-

i_person = 2

i_box1 = 1

-> State: 1.5 <-

i_person = 3

action_person = down

-> State: 1.6 <-

j_person = 2

j_box2 = 1

action_person = left

-> State: 1.7 <-

j_person = 3

action_person = right

-> State: 1.8 <-

j_person = 4

j_box3 = 5

-> State: 1.9 <-

j_person = 3

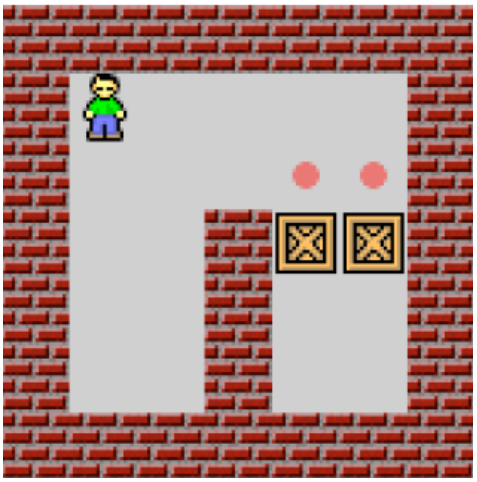
action_person = left

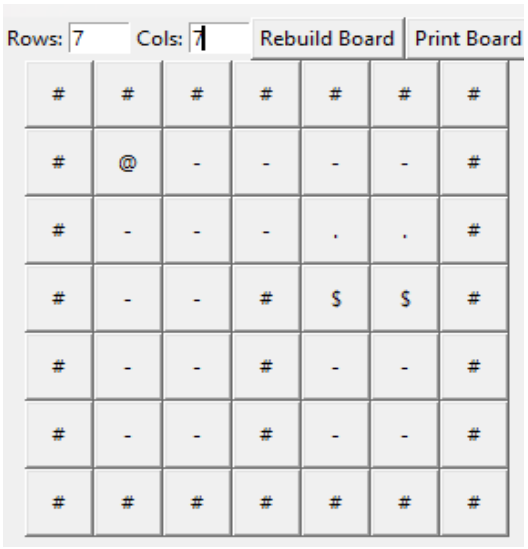
nuXmv > elapse: 0.19 seconds, total: 0.19 seconds

תוצאות:

```
results in lurd format : DUUDLRR
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 0.19 seconds (Total time: 0.19 seconds)
Last checked bound: 7

        runtime results BDD:

        Runtime after check_ltlspec: 0.86 seconds (Total time: 0.86 seconds)
```

ריצה 5:





פלט nuXmv:

nuXmv > nuXmv > nuXmv > nuXmv > -- no counterexample found with bound 0

-- no counterexample found with bound 1

-- no counterexample found with bound 2

-- no counterexample found with bound 3

-- no counterexample found with bound 4

-- no counterexample found with bound 5

……

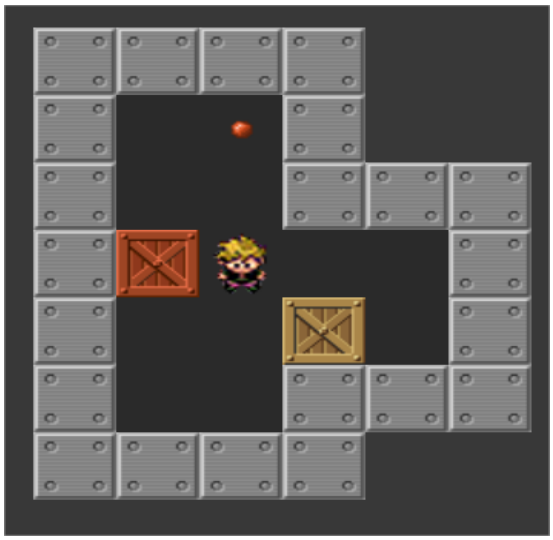
-- no counterexample found with bound 40

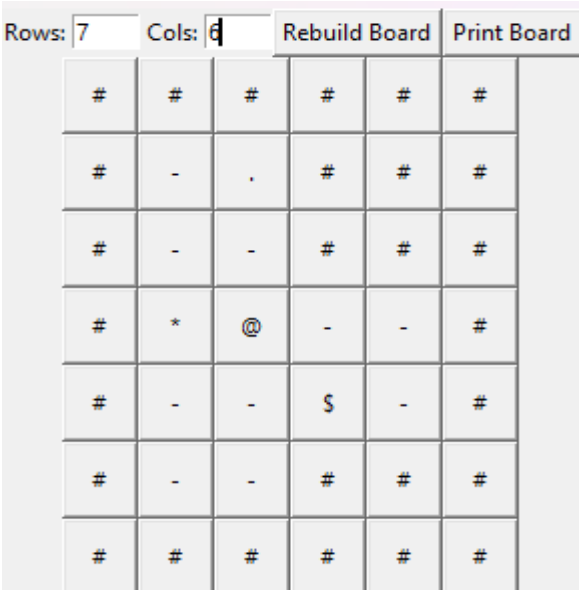
nuXmv > elapse: 246.91 seconds, total: 246.91 seconds

תוצאות:

```
results in lurd format :
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 246.91 seconds (Total time: 246.91 seconds)
Last checked bound: 40

        runtime results BDD:

        Runtime after check_ltlspec: 0.09 seconds (Total time: 0.09 seconds)
```

ריצה 6:





nuXmv output

Trace Type: Counterexample

-> State: 1.1 <-

i_person = 3

j_person = 2

i_box1 = 3

j_box1 = 1

i_box2 = 4

j_box2 = 3

action_person = no-action

boxes_overlap = FALSE

box_on_wall = FALSE

man_on_box = FALSE

man_on_wall = FALSE

j_box_goal2 = 1

i_box_goal2 = 3

j_box_goal1 = 2

i_box_goal1 = 1

M = 6

N = 7

grid[0][0] = 1

grid[0][1] = 1

grid[0][2] = 1

grid[0][3] = 1

grid[0][4] = 1

grid[0][5] = 1

grid[1][0] = 1

grid[1][1] = 0

grid[1][2] = 0

grid[1][3] = 1

grid[1][4] = 1

grid[1][5] = 1

grid[2][0] = 1

grid[2][1] = 0

grid[2][2] = 0

grid[2][3] = 1

grid[2][4] = 1

grid[2][5] = 1

grid[3][0] = 1

grid[3][1] = 0

grid[3][2] = 0

grid[3][3] = 0

grid[3][4] = 0

grid[3][5] = 1

grid[4][0] = 1

grid[4][1] = 0

grid[4][2] = 0

grid[4][3] = 0

grid[4][4] = 0

grid[4][5] = 1

grid[5][0] = 1

grid[5][1] = 0

grid[5][2] = 0

grid[5][3] = 1

grid[5][4] = 1

grid[5][5] = 1

grid[6][0] = 1

grid[6][1] = 1

grid[6][2] = 1

grid[6][3] = 1

grid[6][4] = 1

grid[6][5] = 1

-> State: 1.2 <-

i_person = 4

action_person = down

-> State: 1.3 <-

j_person = 1

action_person = left

-> State: 1.4 <-

i_person = 3

i_box1 = 2

action_person = up

-> State: 1.5 <-

j_person = 2

action_person = right

-> State: 1.6 <-

j_person = 3

-> State: 1.7 <-

j_person = 4

-> State: 1.8 <-

i_person = 4

action_person = down

-> State: 1.9 <-

j_person = 3

j_box2 = 2

action_person = left

-> State: 1.10 <-

i_person = 3

action_person = up

-> State: 1.11 <-

j_person = 2

action_person = left

-> State: 1.12 <-

j_person = 1

-> State: 1.13 <-

i_person = 4

action_person = down

-> State: 1.14 <-

i_person = 5

-> State: 1.15 <-

j_person = 2

action_person = right

-> State: 1.16 <-

i_person = 4

i_box2 = 3

action_person = up

-> State: 1.17 <-

j_person = 1

action_person = left

-> State: 1.18 <-

i_person = 3

action_person = up

-> State: 1.19 <-

j_person = 2

j_box2 = 3

action_person = right

-> State: 1.20 <-

i_person = 2

action_person = up

-> State: 1.21 <-

i_person = 1

-> State: 1.22 <-

j_person = 1

action_person = left

-> State: 1.23 <-

i_person = 2

i_box1 = 3

action_person = down

-> State: 1.24 <-

j_person = 2

action_person = right

-> State: 1.25 <-

i_person = 3

action_person = down

-> State: 1.26 <-

i_person = 4

-> State: 1.27 <-

j_person = 3

action_person = right

-> State: 1.28 <-

j_person = 4

-> State: 1.29 <-

i_person = 3

action_person = up

-> State: 1.30 <-

j_person = 3

j_box2 = 2

action_person = left

-> State: 1.31 <-

i_person = 4

action_person = down

-> State: 1.32 <-

j_person = 2

action_person = left

-> State: 1.33 <-

i_person = 3

i_box2 = 2

action_person = up

-> State: 1.34 <-

i_person = 2

i_box2 = 1

-> State: 1.35 <-

j_person = 1

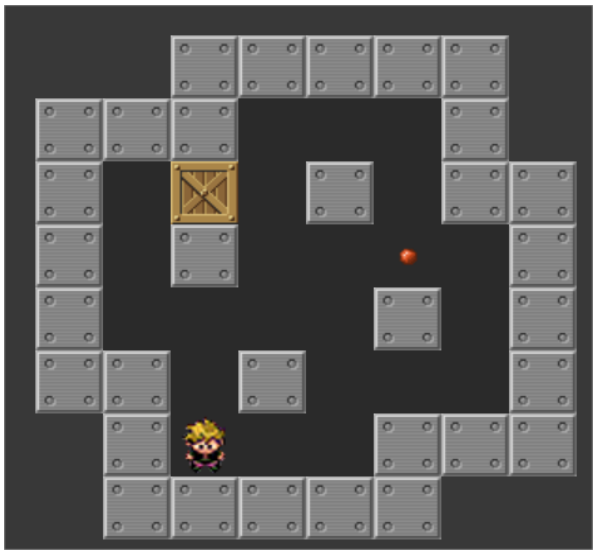
action_person = left

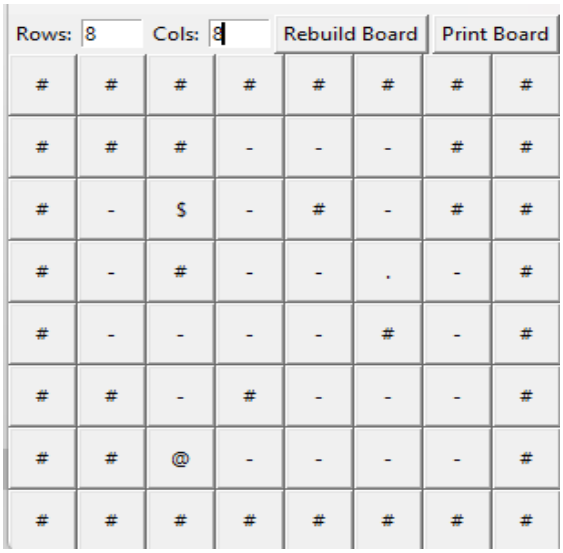
nuXmv > elapse: 43.16 seconds, total: 43.16 seconds

תוצאות:

```
results in lurd format : DLURRRDLULLDDRULURUULDRDDRRULDLUU
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 43.16 seconds (Total time: 43.16 seconds)
Last checked bound: 33

        runtime results BDD:

        Runtime after check_ltlspec: 0.18 seconds (Total time: 0.18 seconds)
```

ריצה 7:





:Output

Trace Type: Counterexample

-> State: 1.1 <-

i_person = 6

j_person = 2

i_box1 = 2

j_box1 = 2

action_person = no-action

boxes_overlap = FALSE

box_on_wall = FALSE

man_on_box = FALSE

man_on_wall = FALSE

j_box_goal1 = 5

i_box_goal1 = 3

M = 8

N = 8

grid[0][0] = 1

grid[0][1] = 1

grid[0][2] = 1

grid[0][3] = 1

grid[0][4] = 1

grid[0][5] = 1

grid[0][6] = 1

grid[0][7] = 1

grid[1][0] = 1

grid[1][1] = 1

grid[1][2] = 1

grid[1][3] = 0

grid[1][4] = 0

grid[1][5] = 0

grid[1][6] = 1

grid[1][7] = 1

grid[2][0] = 1

grid[2][1] = 0

grid[2][2] = 0

grid[2][3] = 0

grid[2][4] = 1

grid[2][5] = 0

grid[2][6] = 1

grid[2][7] = 1

grid[3][0] = 1

grid[3][1] = 0

grid[3][2] = 1

grid[3][3] = 0

grid[3][4] = 0

grid[3][5] = 0

grid[3][6] = 0

grid[3][7] = 1

grid[4][0] = 1

grid[4][1] = 0

grid[4][2] = 0

grid[4][3] = 0

grid[4][4] = 0

grid[4][5] = 1

grid[4][6] = 0

grid[4][7] = 1

grid[5][0] = 1

grid[5][1] = 1

grid[5][2] = 0

grid[5][3] = 1

grid[5][4] = 0

grid[5][5] = 0

grid[5][6] = 0

grid[5][7] = 1

grid[6][0] = 1

grid[6][1] = 1

grid[6][2] = 0

grid[6][3] = 0

grid[6][4] = 0

grid[6][5] = 0

grid[6][6] = 0

grid[6][7] = 1

grid[7][0] = 1

grid[7][1] = 1

grid[7][2] = 1

grid[7][3] = 1

grid[7][4] = 1

grid[7][5] = 1

grid[7][6] = 1

grid[7][7] = 1

-> State: 1.2 <-

i_person = 5

action_person = up

-> State: 1.3 <-

i_person = 4

-> State: 1.4 <-

j_person = 1

action_person = left

-> State: 1.5 <-

i_person = 3

action_person = up

-> State: 1.6 <-

i_person = 2

-> State: 1.7 <-

j_person = 2

j_box1 = 3

action_person = right

-> State: 1.8 <-

j_person = 1

action_person = left

-> State: 1.9 <-

i_person = 3

action_person = down

-> State: 1.10 <-

i_person = 4

-> State: 1.11 <-

j_person = 2

action_person = right

-> State: 1.12 <-

j_person = 3

-> State: 1.13 <-

j_person = 4

-> State: 1.14 <-

i_person = 3

action_person = up

-> State: 1.15 <-

j_person = 5

action_person = right

-> State: 1.16 <-

i_person = 2

action_person = up

-> State: 1.17 <-

i_person = 1

-> State: 1.18 <-

j_person = 4

action_person = left

-> State: 1.19 <-

j_person = 3

-> State: 1.20 <-

i_person = 2

i_box1 = 3

action_person = down

-> State: 1.21 <-

i_person = 3

i_box1 = 4

-> State: 1.22 <-

i_person = 2

action_person = up

-> State: 1.23 <-

j_person = 2

action_person = left

-> State: 1.24 <-

j_person = 1

-> State: 1.25 <-

i_person = 3

action_person = down

-> State: 1.26 <-

i_person = 4

-> State: 1.27 <-

j_person = 2

action_person = right

-> State: 1.28 <-

j_person = 3

j_box1 = 4

-> State: 1.29 <-

j_person = 2

action_person = left

-> State: 1.30 <-

i_person = 5

action_person = down

-> State: 1.31 <-

i_person = 6

-> State: 1.32 <-

j_person = 3

action_person = right

-> State: 1.33 <-

j_person = 4

-> State: 1.34 <-

i_person = 5

action_person = up

-> State: 1.35 <-

i_person = 4

i_box1 = 3

-> State: 1.36 <-

j_person = 3

action_person = left

-> State: 1.37 <-

i_person = 3

action_person = up

-> State: 1.38 <-

j_person = 4

j_box1 = 5

action_person = right

-> State: 1.39 <-

j_person = 3

action_person = left

nuXmv > elapse: 18.38 seconds, total: 18.38 seconds

תוצאות:

```
results in lurd format : UULUURLDDRRRURUULLDDULLDDRRLDDRRUULUR
runtime results SAT:

        Runtime after check_ltlspec_bmc -k 30: 18.38 seconds (Total time: 18.38 seconds)
Last checked bound: 37

        runtime results BDD:

        Runtime after check_ltlspec: 0.07 seconds (Total time: 0.07 seconds)
```

מסקנה:

בריצות ארוכות / כאלה ללא פתרון BDD יותר טוב מ SAT

בריצות קצרות ומהירות עם מס' צעדים קצר SAT טוב יותר מ BDD

Part 4

1. Break the problem into sub-problems by solving the boards iteratively. For example, solve for one box at a time. Indicate the temporal logic formulae used for each iteration.
2. Indicate runtime for each iteration, as well as the total number of iterations needed for a given board.
3. Test your iterative solution on larger more complex Sokoban boards.

נפתור איטרטיבית באופן הבא:
1.נריץ את המודל עם SPEC שתואם להצלחה בקופסא אחת בלבד
2.במידה וקיבלנו תוצאה, נזיז את השחקן בהתאם לתוצאה ונסדר
3.נקבל לוח חדש ניקח אותו ונגדיר שהוא הלוח
4.נחזור על צעד אחד עם הלוח החדש אבל עם 2 קופסות, וכן הלאה עד שנגיע לכל הקופסאות בתוצאה.

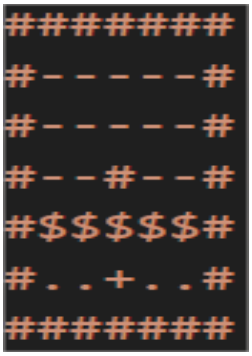
עבור כל הלוחות השתמשנו בSAT למקסימום 30 צעדים, השתמשנו בSAT מכיוון שכאשר היה מדובר במספר רב של צעדים הוא עבד יותר טוב מאשר BDD.
הרצנו את הפקודות הבאות:

```
Nuxmv -int “model_filename_{iteration}”.smv
Read_model -i “model_filename_{iteration}”.smv
Go_bmc
Check_ltlspec_bmc -k 30
```

לכל איטרציה מודל שונה לפי הפעולות להאיטרציה הקודמת ולכן ה{iteration}
נראה כיצד השיטה האיטרטיבית עובדת על הלוחות הבאים:
התנאי שלנו בסוף השתנה בהתאמם למספר האיטרציה.
בכל איטרציה אנחנו לקחנו לכל תת קבוצה אפשרית בגודל של מספר הקופסאות התואם לאיטרציה, את כל הפרמוטציות האפשריות בהתאם לתת קבוצה הזאת של BOX-GOALS.

כלומר בהנתן { boxn..., 3box,2box, 1box} ובהתאמה { goaln..., 3goal,2goal,1goal}
באיטרציה 1 ניקח את כל תתי הקבוצות בגודל 1:
{1goal},{2goal},{ 3goal},...{goaln} ונתאים אותן ל{ boxn..., 3box,2box, 1box} כך שלפחות אחת מהקופסאות שווה לGOAL.
באיטרציה 2 ניקח את כל תתי הקבוצות בגודל 2:
{goal1,goal2},{goal1.,goal3},,...{goaln-1, goaln }
ונתאים אותן ל{ boxn..., 3box,2box, 1box} כך שלפחות אחת מהקופסאות שווה לGOAL.
וכו.

ריצה ופלטים עבור חלק 4:

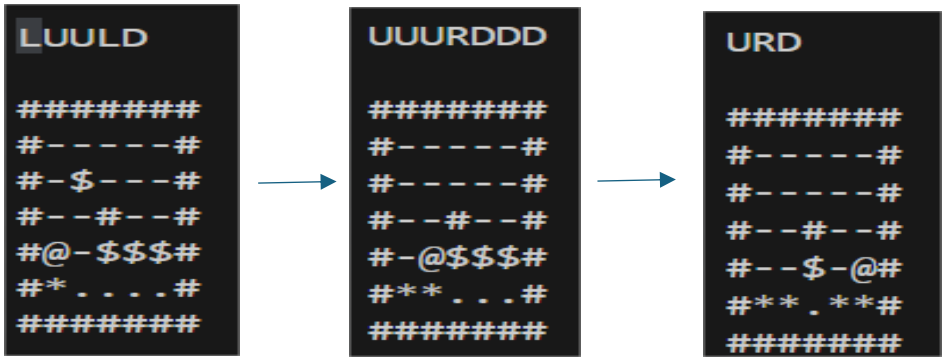


The Runtime for iteration 1:
Runtime after check_ltlspec_bmc -k 30: 0.41 seconds (Total time: 0.41 seconds)
The Runtime for iteration 2:
Runtime after check_ltlspec_bmc -k 30: 0.78 seconds (Total time: 0.78 seconds)
The Runtime for iteration 3:
Runtime after check_ltlspec_bmc -k 30: 1.92 seconds (Total time: 1.92 seconds)
The Runtime for iteration 4:
Runtime after check_ltlspec_bmc -k 30: 2.11 seconds (Total time: 2.11 seconds)
The Runtime for iteration 5:
stop

Formula for iteration 1:
LTLSPEC G!((!next(man_on_box) & !next(man_on_wall) & !next(box_on_wall) & !next(boxes_overlap)) & ((i_box1 = 5) & (j_box1 = 1)) | ((i_box1 = 5) & (j_box1 = 2)) | ((i_box1 = 5) & (j_box1 = 3)) | ((i_box1 = 5) & (j_box1 = 4)) | ((i_box1 = 5) & (j_box1 = 5)) | ((i_box2 = 5) & (j_box2 = 1)) | ((i_box2 = 5) & (j_box2 = 2)) | ((i_box2 = 5) & (j_box2 = 3)) | ((i_box2 = 5) & (j_box2 = 4)) | ((i_box2 = 5) & (j_box2 = 5)) | ((i_box3 = 5) & (j_box3 = 1)) | ((i_box3 = 5) & (j_box3 = 2)) | ((i_box3 = 5) & (j_box3 = 3)) | ((i_box3 = 5) & (j_box3 = 4)) | ((i_box3 = 5) & (j_box3 = 5)) | ((i_box4 = 5) & (j_box4 = 1)) | ((i_box4 = 5) & (j_box4 = 2)) | ((i_box4 = 5) & (j_box4 = 3)) | ((i_box4 = 5) & (j_box4 = 4)) | ((i_box4 = 5) & (j_box4 = 5)) | ((i_box5 = 5) & (j_box5 = 1)) | ((i_box5 = 5) & (j_box5 = 2)) | ((i_box5 = 5) & (j_box5 = 3)) | ((i_box5 = 5) & (j_box5 = 4)) | ((i_box5 = 5) & (j_box5 = 5)));

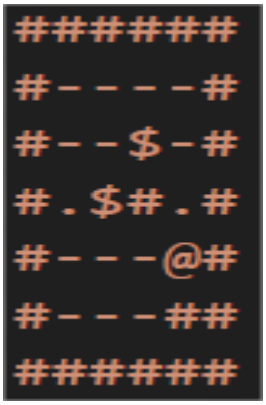
עבור איטרציות 2 ואלך, התנאים ארוכים מאוד בכדי לשים בדוח. מצורפות תוצאות למודלים בGITHUB.

כאן השחקן הגיע עד 4 קופסאות מכיוון שבהגעה ל2 קופסאות אחת מהקופסאות נתקעה בפינה.
סידור הקופסאות:



השוואה להרצת לוח בשיטה לא איטרטיבית:

אחרי ניסיון של להריץ את המודל עבור 5 קופסאות ישירות חיכינו שעה והמודל לא נתן פתרון עם זאת כאן יש סידור של 4 קופסאות.



The formula for iteration 1:

LTLSPEC G!((!next(man_on_box) & !next(man_on_wall) & !next(box_on_wall) & !next(boxes_overlap)) & ((i_box1 = 3) & (j_box1 = 1)) | ((i_box1 = 3) & (j_box1 = 4)) | ((i_box2 = 3) & (j_box2 = 1)) | ((i_box2 = 3) & (j_box2 = 4)));

The formula for iteration 2:

LTLSPEC G!((!next(man_on_box) & !next(man_on_wall) & !next(box_on_wall) & !next(boxes_overlap)) & ((i_box1 = 3) & (j_box1 = 1) & (i_box2 = 3) & (j_box2 = 4)) | ((i_box1 = 3) & (j_box1 = 4) & (i_box2 = 3) & (j_box2 = 1)));

Runtime iteration 1:

Runtime after check_ltlspec_bmc -k 30: 0.14 seconds (Total time: 0.14 seconds)
Last checked bound: 7

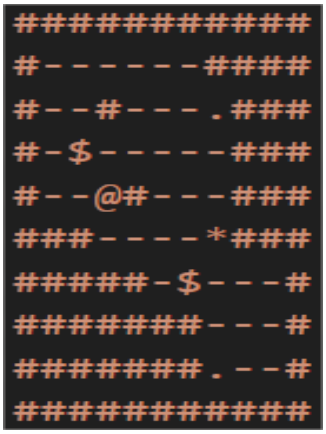
Runtime iteration 2:

Runtime after check_ltlspec_bmc -k 30: 23.36 seconds (Total time: 23.36 seconds)
Last checked bound: 23

השוואה להרצת לוח בשיטה לא איטרטיבית:

Runtime after check_ltlspec_bmc -k 30: 32.12 seconds (Total time: 32.12 seconds)
Last checked bound: 23

כאן השיטה האיטרטיבית עבדה יותר מהר 32.12>23.36+0.14



The formula for iteration 1:

LTLSPEC G!((!next(man_on_box) & !next(man_on_wall) & !next(box_on_wall) & !next(boxes_overlap)) & ((i_box1 = 2) & (j_box1 = 7)) | ((i_box1 = 5) & (j_box1 = 7)) | ((i_box1 = 8) & (j_box1 = 7)) | ((i_box2 = 2) & (j_box2 = 7)) | ((i_box2 = 5) & (j_box2 = 7)) | ((i_box2 = 8) & (j_box2 = 7)) | ((i_box3 = 2) & (j_box3 = 7)) | ((i_box3 = 5) & (j_box3 = 7)) | ((i_box3 = 8) & (j_box3 = 7)));

The formula for iteration 2:

LTLSPEC G!((!next(man_on_box) & !next(man_on_wall) & !next(box_on_wall) & !next(boxes_overlap)) & ((i_box1 = 2) & (j_box1 = 7) & (i_box2 = 5) & (j_box2 = 7)) | ((i_box1 = 2) & (j_box1 = 7) & (i_box2 = 8) & (j_box2 = 7)) | ((i_box1 = 5) & (j_box1 = 7) & (i_box2 = 2) & (j_box2 = 7)) | ((i_box1 = 5) & (j_box1 = 7) & (i_box2 = 8) & (j_box2 = 7)) | ((i_box1 = 8) & (j_box1 = 7) & (i_box2 = 2) & (j_box2 = 7)) | ((i_box1 = 8) & (j_box1 = 7) & (i_box2 = 5) & (j_box2 = 7)) | ((i_box1 = 2) & (j_box1 = 7) & (i_box3 = 5) & (j_box3 = 7)) | ((i_box1 = 2) & (j_box1 = 7) & (i_box3 = 8) & (j_box3 = 7)) | ((i_box1 = 5) & (j_box1 = 7) & (i_box3 = 2) & (j_box3 = 7)) | ((i_box1 = 5) & (j_box1 = 7) & (i_box3 = 8) & (j_box3 = 7)) | ((i_box1 = 8) & (j_box1 = 7) & (i_box3 = 2) & (j_box3 = 7)) | ((i_box1 = 8) & (j_box1 = 7) & (i_box3 = 5) & (j_box3 = 7)) | ((i_box2 = 2) & (j_box2 = 7) & (i_box3 = 5) & (j_box3 = 7)) | ((i_box2 = 2) & (j_box2 = 7) & (i_box3 = 8) & (j_box3 = 7)) | ((i_box2 = 5) & (j_box2 = 7) & (i_box3 = 2) & (j_box3 = 7)) | ((i_box2 = 5) & (j_box2 = 7) & (i_box3 = 8) & (j_box3 = 7)) | ((i_box2 = 8) & (j_box2 = 7) & (i_box3 = 2) & (j_box3 = 7)) | ((i_box2 = 8) & (j_box2 = 7) & (i_box3 = 5) & (j_box3 = 7)));

The formula for iteration 3:

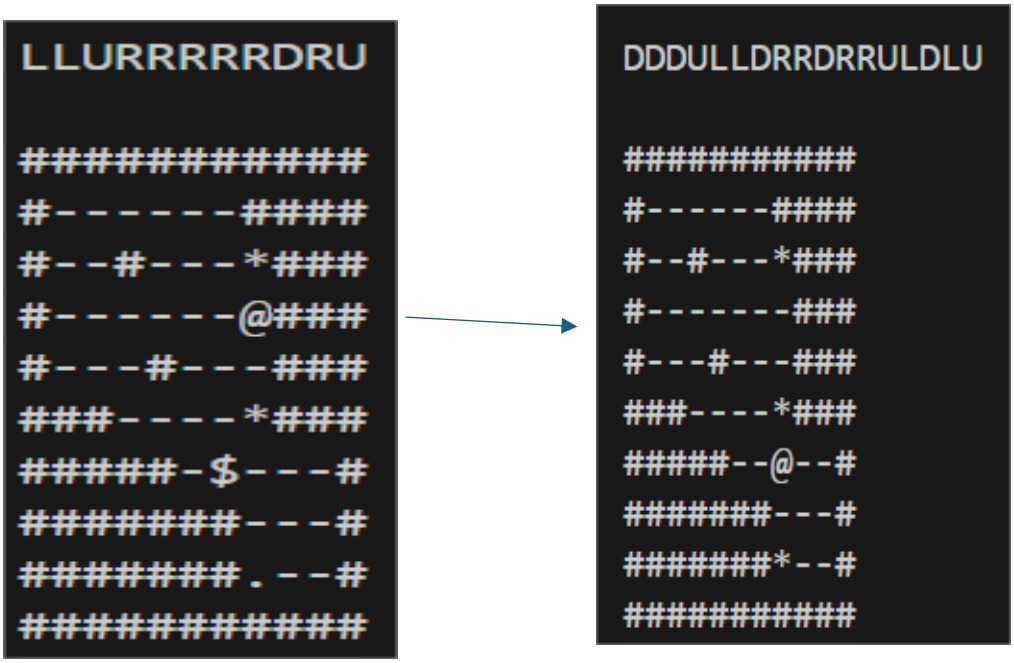
LTLSPEC G!((!next(man_on_box) & !next(man_on_wall) & !next(box_on_wall) & !next(boxes_overlap)) & ((i_box1 = 2) & (j_box1 = 7) & (i_box2 = 5) & (j_box2 = 7) & (i_box3 = 8) & (j_box3 = 7)) | ((i_box1 = 2) & (j_box1 = 7) & (i_box2 = 8) & (j_box2 = 7) & (i_box3 = 5) & (j_box3 = 7)) | ((i_box1 = 5) & (j_box1 = 7) & (i_box2 = 2) & (j_box2 = 7) & (i_box3 = 8) & (j_box3 = 7)) | ((i_box1 = 5) & (j_box1 = 7) & (i_box2 = 8) & (j_box2 = 7) & (i_box3 = 2) & (j_box3 = 7)) | ((i_box1 = 8) & (j_box1 = 7) & (i_box2 = 2) & (j_box2 = 7) & (i_box3 = 5) & (j_box3 = 7)) | ((i_box1 = 8) & (j_box1 = 7) & (i_box2 = 5) & (j_box2 = 7) & (i_box3 = 2) & (j_box3 = 7)));

Runtime iteration 1:
Runtime after check_ltlspec_bmc -k 30: 0.11 seconds (Total time: 0.11 seconds)
Last checked bound: 0

Runtime iteration 2:

Runtime after check_ltlspec_bmc -k 30: 1.44 seconds (Total time: 1.44 seconds)
Last checked bound: 11

Runtime iteration 3:
Runtime after check_ltlspec_bmc -k 30: 5.56 seconds (Total time: 5.56 seconds)
Last checked bound: 17



השוואה להרצת לוח בשיטה לא איטרטיבית:
אחרי הרצה בשיטה הלא איטרטיבית קיבלנו את התוצאות הבאות:

results in lurd format : LLURRRRRDRUDDDULLDRRDRRULDLU
runtime results SAT:

Runtime after check_ltlspec_bmc -k 30: 365.16 seconds (Total time: 365.16 seconds)
Last checked bound: 28

ניתן לראות בבירור שכאן הריצה הלא איטרטיבית עובדת בהרבה פחות טוב.