16.10.13

שיעור 1

מבוא לבינה מלאכותית

הקורס עוסק באלגוריתמים.
לא נדבר הרבה על הוכחות נכונות.
נדבר יותר על היוריסטיקה.

* חיפוש בלא הפרט לבינה מלאכותית.
יש מטרה, מרחב פתרונות, אך להגיע למטרה.

* חיפוש עם ידע - רוצים להגיע למטרה. יש מושגים שעזרה למצוא ואנחנו להגיע למטרה. בלא לבדוק מהם כל המשתנים.
* CSP - Constrain Satisfaction Problems
בעיות סיפוק אילוצים.
יש רשימת אילוצים. יש למצוא דרך לעמוד (לספק) כל האילוצים

* שליטה (הסקה שלב).
* תכנון. תכנון הדרך להגיע למטרה.
* הסתברות
* למידה מדגמאות

חיפוש

קונבליים ומסוכרים.

בקונבליים, במסוכרים נמצאים בתוך אחרי שכבר. שהם רוצים לבדוק את הנכון ולהשתמש אומרים סיבה אחת.
- הסיבה יכולה להיות שהם אוכלים לכל היותר.
- לעתים שיהיו יותר קונבליים ממסוכרים בכל מצב.

יש שני סוגי חיפוש
- חיפוש המלך הכולל. גם יודעים מה הדרך להגיע לנשיא.
- חיפוש מלך התהלך. לא יודעים מה המלך הכולל ומיהו הוא נראה

אלגוריתם לחיפוש הנתיב - כל אחד.

לפני החיפוש יש לחשב את המרחק
לפני האלגוריתם לקבלים והמרחקים המרחק בין
לחשב כמה קדקודים מושלמים, מסוגרים מושלמים
והיכן הסיר הה.
מבחוץ ניתן לחשב אותה מהלך יש לבדוק <u>אם</u> לאלו
מרבים אפשר לעבור מהמרחק הקודם.

1, 3, 2 הקדקוד

יש לבדוק: מהו המרחק ההתחלתי / מרחק כולל.
יש לבדוק אם המסלול הזה למרחק הנוכחי.
מרבים נסרקו ונחתכו (explored)
מרבים שנותרו חיפוש מהם (frontier)
בדרכה ההיוריסטי (heuristic ..)
לא באמת יודעים, נשתמש בהוריסטיקה שנותנת קירוב.

חיפוש עיוור - כאשר לא יודעים מספר על הבעיה. מעט
בלי מידע כוונים
חיפוש נרחב (חיפוש הוריסטי) - כאשר יש כוונים בכדי
לחיפוש

לסדר את המדבים:

מדבר ?	מטונים	מטונים	דגלים
מדבר התחלתי	3	3	L
מדבר סופי	0	0	R
	1	0	R L
	2	0	R L
	3	0	R L
	2	2	R L
	2	3	R L
	1	1	R L
	1	3	R L
	0	3	R L

סה"כ 18 מדבים אפשריים.
מכרעה המדבים / המעברים:
מדבר הסרה תלוי תמיד בין R ל L.

L	R
3, 3	0, 0
1, 0	1, 0
2, 0	2, 0
3, 0	3, 0
2, 2	2, 2
2, 3	2, 3
1, 1	1, 1
1, 3	1, 3
0, 3	0, 3

אין דרך המדבים הכללים סימטריים אם המעברים!
אפשר להתקדם: יש לסרוק את הגליל להפוך מדבר התחלה
למדבר הסופי.

יש שני עקרונות לחיפוש בגרף:

DFS - חיפוש לעומק.

בהתחלה ה- frontier יש 3,3,L.

באלגוריתם - מוציאים את ה-frontier של המצב הבא בו יש עדיפות. יש לפחות שמצב זה לשים explored.

יש לדחוף את כל המצבים שלידה ניתן להגיע מהמצב זה.

אבל המצבים הללו יש לשים ב- frontier (בלבנו) מעל המצב הקודם.

מוציאים את ה- frontier בצורה

ההבדל בין באלגוריתמים - בכיוון בחירת הצומת.

נשתמש ב- LIFO.

frontier

נראה לפחות ניתן להגיע מינו

3,3,L

1,3,R

2,3,L

0,3,R

1,3,L

1,4,R

2,4,R

3,4,R

3,5,L

2,5,R

1,5,L

1,6,R

2,6,R

3,6,R

3,7,L

2,7,R

1,7,L

1,8,R

2,8,R

3,8,R

3,9,L

2,9,R

1,9,L

1,10,R

2,10,R

3,10,R

3,11,L

2,11,R

1,11,L

1,12,R

2,12,R

3,12,R

3,13,L

2,13,R

1,13,L

1,14,R

2,14,R

3,14,R

3,15,L

2,15,R

נניח: b - פיצול

h - אורך הדרך לפתרון.

זמן הריצה: $O(b^h)$.

זיכרון: Frontier: $O(b \cdot h)$ מספר

explored: $O(b^h)$ כל המצבים בהם

סיירנו. גרוע. גם בעיה קשה.

בדר"כ ניתן לשפר את זמן הריצה ואת הזיכרון ע"י explored.

את זמן הריצה לא ניתן לשפר.

אם לא explored ניתן לעדכן רק חוסכים בזיכרון.

בעיה - יתכן ~~[illegible]~~ המצבים. לולאה אינסופית

בחיפוש בעומק - אין מצבים

חיפוש בגודל - יותר מצבים.

אם בכלל שבפתרון שמצאנו הוא הדרך הקצרה ביותר.

אנשר לדפסנות כ- FIFO בעומק LIFO.

BFS - חיפוש לרוחב

BFS מבטיח למצוא את הפתרון הקצר ביותר למטרה

הסיבה: בכל פעם אדרים כל המצבים שהם בעומק

במרחק x ידמ"ט.

~~[illegible]~~ זמן ריצה - כמו DFS. $O(b^h)$

לזיכרון - Frontier הוא מדד ההתחלה

בעיה ה Frontier הפך להיות גדול.

A* - exploited נמצא בעליה.
אם אין פתרון - נגיע אל אותו המצב לאורך
של ה- Frontier מהראשון ואז הגענו למטרה.

ההבדל בין אלגוריתמי החיפוש הוא במצב
ההתרחבות של ה- Frontier.

שלב ראשון שלב DFS ו- BFS.
יתרון DFS - צריכה.
יתרון BFS תמיד מוצא פתרון.
BFS - בתור FIFO
DFS LIFO
ID - נעשה DFS לעומק 1. אם נמצא פתרון - נעשה
כמו לעומק 2 אם אין נמצא פתרון - DFS לעומק 3.
אם ה- DFS עושים מיון מההתחלה.
צריך יכולת פתרון קרוב תוך כדי סריקה של ה- Frontier
של DFS.
הסבנה - קלקול זמן הריצה $b^1+b^2+b^3+\dots+b^h = O(b^h)$
על כל הרמות שסכומה: $\frac{b^{h+1}-1}{b-1}$

אם לכל מהלך יש משקל אחר, הפלגה מה- Frontier
הוא בעדיפות הנמוכה קודמות.
יש לעשות על הגרף מיון בעזרת. צריך מיון
אל ה- Frontier.

USC - אלגוריתם שמחזיר את הדרך קצרה ביותר.

בכל הזמן נשמרים מרחק הכי יש לבדוק אם הגיע
לא נמצא כבר.
בראש לפחות נצטרך אינדקס ולעדכנם הכל.

כל אלו הנחות אלוריתם. ולכן יש לשמור מרחק לגרף.

היוריסטיקה heuristic

על יודעים בערך המרחק קרוב למטרה. אפשר לדעת.
"בלי לדעת".

לפי מדד יודעים לדרוג את המרחק מהמטרה.
* הקונטור אומר לנו מהמבוך (Frontier)
את המצב שאנחנו מדרגים שבאמת הכי קרוב למטרה.
Greedy Best First Search - חיפוש חמדני:
וחמדן שהמרחק למטרה קרוב אליה הוא חתך מהבעיה.
לעל הטלה שנקבל את הדרך הקצרה ביותר.

A^* (A star)

אלגוריתם שמשתמש גם במרחק מההתחלה $g(n)$
וגם רק במרחק למטרה $h(n)$.
בגרור מצב בעל $h(n)+g(n)$ הכי נמוך.
$h(n)$ - פונקציה אריסטית. הערכה
$g(n)$ - מרחק מההתחלה. יודעים בחוזק.

הבעיה שמגבר בעל (ופא בחרל) מובטח ש- A^* ימצא
את הפתרון הטוב ביותר בתנאי ש- $h(n)$ קבלה / אופטימית.
כלומר: אם $h(n)$ אופטימי מובטח ש- A^* ימצא דרך
קצרה ביותר למטרה.
אופטימי - הערכה טובה יותר מהאמת.
קבילה - admissible
פונקציה קבילה תבוא בלי הקונבלים עבור A^*:
שאנשים תמיד ב.
מקבה - אופטימי בדרך לכל שני בחירות.

23.10.13

מפגש 2

חיפוש:
מצב - מצב התחלה
פעולה מצב הלא מצב מטרה?
המצבים - הבאים
הדרכה הלוגית
בגלל למצב מרגיקים frontier ו-explored - אם צריך.
הרעיון הבסיסי של חיפוש - לעבור A כל המצבים לפי סדר
לרבות למצב הסופי.
ההבדל בין האלגוריתמים השונים = סדר הישקיר.
DFS - מרחיבים (frontier), frontier סביר.
BFS תור. מבטיח שנמצא את המסלול הקצר ביותר.
כי מקבל הסיומת (כמה בנים במעלה לכל בן)
l - עומק
עומק BFS - יש צורך ב b^l עומק ה frontier

DFS - מבטיח שימצא את המסלול הקצר ביותר.
אין הרבה סדר למרות שיש רעיון על חיפוש.

נניח פעל ל 3

$1+3$
$1+3+3^2$
⋮

$$\sum_{i=0}^{l} 3^i = \frac{3^{l+1}-1}{2} \approx 1.5\,3^l$$

$1+3+3^2+\dots+3^l$

אין הרבה שום סדר בקבוצה
זה בדיוק עומק הרבה האחרונה
יש מעמקן לבדוק לגה את הבנים שלו הסריקה
הקודמת - הכפלה הקבוע.

אלו חיפושים עלולים לא מתאימים במיוחד
A הבעיה. מתאים למצב ובודקים אם זו המטרה.

האלגוריתם שומר את המצב ההתחלתי (נניח ב- frontier.
כל פעם frontier לא ריק:
מוציאים איבר מה- frontier
שולחים אותו לכל הבנים. בודקים אם כבר מגענו
מגיעה. אם לא - מכניסים ל- frontier. אם כן - סיים.

Uniform - Cost

מעניין - הוא כמעט האלגו פשוט כל מהלך שונה.
יש מהלכים יקרים ומהלכים זולים יותר
הולכים קודם כל לאלגוריתם שם המעגל הקל
ותר.
מעניין - המון קדימויות. מוצאים בכל פעם
את המצב הכי קרוב להתחלה.
כמו אנרג מהאלגוריתמים יש דרך גם בטבלה גיבוי
כבר לבדוק אם המצב נמצא ב- frontier אבל
אין דרך לחפשו בפה נוספת.
באופן תאורטי Uniform Cost יש לחשוב אם נכון
להכניס לסדים מתוים בדרך צלמה יותר.

תיכנון היוריסטי. h(n)

מנסים לבחור לכיוון כדאי ללכת.
אחרי פונקציה היוריסטית
הטיבה היוריסטית - הטיבה המרחק למטרה

אלגוריתם Best first

מבנה נתונים - תור קדימויות לפי h(n), המרחק
אל המטרה.

A* - לפי g(n) + h(n)
מרחק מהמטרה + מרחק מהמצב ההתחלה (מהעבר)
בכל פעם g(n) + h(n) קטן יותר ← טוב.
אם h(n) קבלה (אופטימלי. h(n) ≤ המרחק הלא מזיז)
מובטח שנמצא מעגל קל ביותר.

לא כל היוריסטיקה שמורידים המרחק היא טובה
אם אמת - אם $h(n)$ מקרב בדיוק שמרחק מהצומת
קרוב ביותר
לפחות מקומית.

$$n \xrightarrow{c} n'$$

$$h(n) \le c + h(n')$$

אם הגרף מקיים אז היא הכי קרוב.
מאוד

מרחק מטרה:

$$h(n_2) \le c_2 + h(n_1)$$
$$\le c_2 + c_1$$

$$h(n_1) \le c_1$$

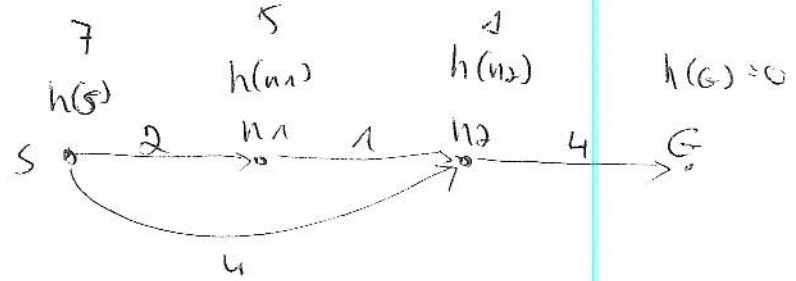
אפשר להראות מרחק רק שהגרף
~~[illegible]~~
מרוחקים הגרף שהתהיה אלגוריתם אין לו מקרים.
הקרוב הכלל אלגוריתם.

מה קרה עם A^*?

Frontier:		פלט אחרון
S		
n_1 2	n_2 3	S
n_1 2	G 5	n_2
G 5	n_2 4	n_1

נתקע שם אל n_2 ל- אחרון

מגיעים אל G לפחות.
מרחק G. 5+0
n_2 6=2+4
הגרף לא מקרים ואז A^* נתנה תוצאה נכונה



אלגוריתם A* לא יעיל. הסיבה? מרקלה? $h(s)=7$ לעומת $4+h(מ2)$

Frontier		מלבן
s		
מ1 2	מ2 3	s
מ1 2	G 8	מ2
G 8	מ2 3	מ1
G 7		מ2

s נכנס ל-frontier
שכנים של s: מ1, מ2
עדיפות מ1: $2+5=7$
מ2: $1+4=5$
מכניסים את G לסוף לתור
$4+4$.

מלבן של מ1: $2+5 < 8+0$
מ1 6
מלבן
מ2 מעדכנת את G ל-7.

תיכנון מרקלי:

לפעמים לא יודעים איך נראה המטרה.
לדוגמא - כדורגל. יודעים את היתרונות אך לא יודעים
איך נראה המטרה. יודעים רק לעבור לזה
מאיזה אי הלא המטרה אז לא.
לכן תיכנון בדרך הנשמה הפירון לפי הדרך.
לכן נסמוך רק מצב אחרון הוא בן-כלון.
מצב הלא מצב שלם (complete) - בגלל
שלמה יהיו כל הרכיבים אולם לא בסדר הכלון.
יש לבצע שינויים קטנים במצב כדי לעשות אותו
לא כלום למצב המטרה.
יש שינויים רבים שאפשר לעשות אבל שינוים
לבצע?

hill climbing - שיפור בלב מקומי

בעיה - לפעמים נתקעים לאקסימום מקומי אך לא ניתן לדעת לגבי המלכה. לא ניתן לנסות מחדש עוד.

בעיה נוספת: בלוקים מישורים

אלגו בסיס - hill climbing

כדי ניתן לצאת מזה simulated annealing

בשביל נתקעים בבלוקים אל מקסימום מקומי מבצעים שינוי גדול.

לפעמים יש "טמפרטורה" שהולכת ויורדת בכל צעד קיים סיכוי (שנגדל לפי הטמפרטורה) שהפתרון ישתנה ויהפוך גרוע יותר (שגדל לפי הטמפרטורה)

k-beam Search

לשמור k מצבים.

אלגוריתמים גנטיים

ניתן לייצג את הגנים בדרכים מתחרות.

לכל מתחרה ניתן לתת כמה הוא קרוב למלוכד הפתרון. fitnes - כמה המלכה קרוב למטרה.

מגרילים, בהתבסס אקראי, ל מתחרה לפתרון.

לכל מתחרה מתייחסים אל ה fittest.

מגרילים זוגות של מתחרים, זוגות במספר המתחרים. (מתוך המתחרים הישנים).

בתהליך עושים חצייה.

בחתיכה נקבע חיתוך לכל זוג מתחרים.

ניתן להחליף בין הזוגות את החלקים שחתכנו.

מבצעים מוטציה קלה בכל שלב מ- את המתחרים החדשים.

בטורנירים שמדמים את ה"הישרדות של ה- fittest
הגדלה בגודל – ומתנסים בו.
המטרה – להגיע לרמה מקסימלית.

משתמשים בדבר הזה אשר אין אלגוריתם יעיל ולא להפוך.

תיפוש נגד יריב – adversary

שמות למושגים.

מינימקס – רעיון של מנצח בשם פון נוימן.
יש לבנים שבה מבלק נגדה לשחק מה שהכי טוב
לבנים להגיע ישמה מה שהכי טוב עבורן.

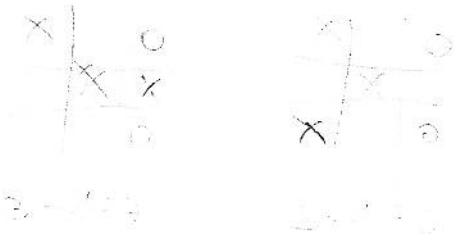
בגדר הכלי יש עלים.
1- הפסד
1 ניצחון
0 תיקו

מלא בתחילה הלוח. בכל פעם שמלא רושמים בתוך
את תור המינימקס. כלומר שני הצדדים אותו הדבר יקרה.
כאשר תגיע לעשרים נבדוק אם הלוח שלבו מקסימלי.

צריך לא מתאים לעשרים. מצב. בדיוק.
מה קורה כאשר יש הרבה שרוקנים, אבל מה זה?
ובמקום מצב כזו מה שרוקנים מתאחדים עם שרוקן אחר.
צריך מתאים לעשרים. ללא אלמנט של מצל לה ה שרוקנים
בלבד.

הבעיה בשלבים - המאבל שלה.
מוצרים את הלוח הזאבה סביר בלשכל.
בשקרה נמצא נצרה בשאר המעשרים דרוך פונק דיה בלוגיקה
שרשרת את מצב המעשרים.
נמשיכים את הכלים לפי הפונקציה בלוגיקה ומעבירים
הרשאה את המצבים להתחרות

פונקציה בלוגיקה טוב מעשרים o x.
כמה אפשרויות לעשות טבלה - מה נשמר אפשרות לירה



x - הלוח האחרון שלו.

הפונקציה הלוגיקה - כזה.
תוצאות המעשרים - נשיקה.

30.10.13

שיעור 3

A*

A* בדיק לקבל פונקציה קבילה או מונוטונית
קבילה - אופטימלי. בעיה טובה יותר מבחינה
מונוטונית - $h(n) \le c + h(n')$

$h(n) \le c + h(n')$

אם פונקציה מונוטונית היא תמיד קבילה.
$h(G)=0$ בשביל G היא בדיוק האמיתית.

A* - פונקציה קבילה.
לא מונוטונית. בקטע SA: $7 \le 4+1$
אפשר נכון ולכן ~~הקטע~~ הפונקציה לא מונוטונית

יש שני סוגי חיפושים: חיפוש בעץ וחיפוש בגרף.
חיפוש בעץ - אין explored.
חיפוש בגרף - יש explored.

אלו נסמלים, לא בהכרח כל הנתונים.
אם עושים חיפוש בעץ המשמעות הפונקציה קבילה -
מובטח לנו שנקבל פתרון קצר ביותר.
חיפוש בגרף (עם explored) בדרך הפונקציה מונוטונית -
חיפוש בגרף בשביל הפונקציה לא מונוטונית - ~~הפונקציה~~ A*
~~אם~~ עלול לפספס.

חיפוש בגרף

Frontier

[illegible]

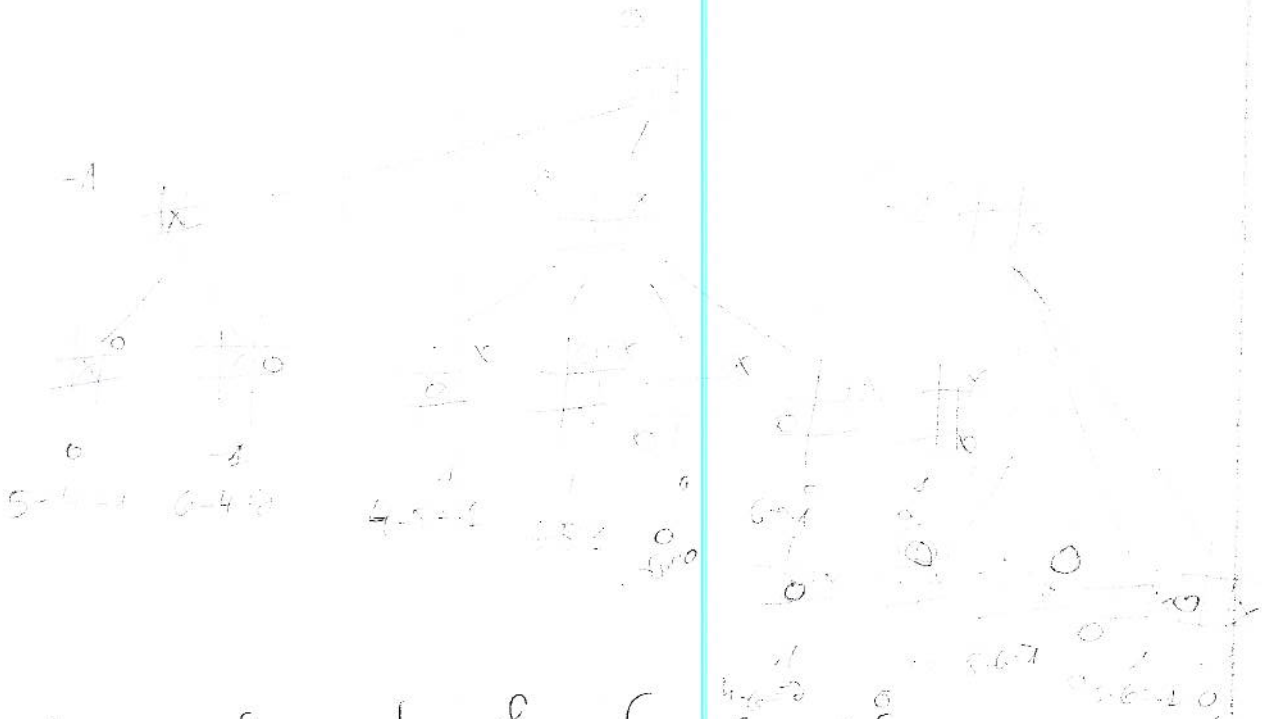
יש לבחור בין A ל- B. $f(n)=g(n)+h(n)$ שהוא שווה.
נניח ש- A יוצא בו הכי הרבה קדקוד לעיגול
מגיעים מהמלל מ- S ל- G ט- $f(n)=g(n)$. זה לא
המסלול הקצר ביותר.
לפעמים B יוצא. יש לשים explored ~~[illegible]~~ לכן חוזר
ש- A כבר היה לכן לא מוסיפים אותו פעם
נוספת.
המסלול שיבחר הוא S-A-G זה לא המסלול
הקצר ביותר.
האינטואיציה מבטיחה שנמצא קודם כל את המסלול
הקצר למטרה.

חיפוש נגד יריב

למשחקים. שני שחקנים משחקים בתורות.
מנצחים סכום אפס. אחד מנצח השני מפסיד
אין תיקו.
מנצח = 1, מפסיד = -1, תיקו = 0.
מנצחים מיצג מצב.
מיצג לא מצב - משחק, מעגל, קדקוד, שחקן רוצה
רק את הקלפים שלו ולא גם היריב.
~~[illegible]~~
נניח אי - אקראיות.

עקרון - מינימקס

ב- טא, לשחקן הראשון ש
שלושה מהלכים אפשריים (הצעדים)
יש סימטריה בין המהלכים.



אם נרצה היה לצייר את כל הפלט אז בעיה היינו
מבחינה לכל נקודה כדי את התוצאה: 1-, 1, 0.
(הכוונה מנקודה מאז כל הX, שמשתמש הושמן).
X - שריקן מקסימום (נקודות = 1)
O שריקן מינימום (נקודות = 1-)
כל שריקן בתור בתצוגה הטוב ביותר שבורן.

קרמת מצב - השריקן בתור את המינימום מכל הבנים
שלו. זה שריקן המינימום שבתור.
שריקן המקסימום בתור את הדיק המקסימום שקבל
הרמות מתחלפות בין שריקן המקסימום לשריקן המינימום.

ההנחה היא ששריקן היריב מלמד בתור תמיד
את האלגוריתם הטוב ביותר שבורן.

בעיה כל מינימקס. אז ניסיון אז ניתן לצייר
את כל הפלט.

פונקציה ואריבלים לשימוש ב-OX:
מספר המשתנים הפנימיים ל X בתור שאנחנו בתור ל O.
וסיתים אב המשתמש בדי לדעת שאפשר (נצחון
מוגרה + תיקו) ואז נשתמש בפונקציה ואריבלים

שמתחת הסיבה לעדיפות.
האלגוריתם מתנשק באותה שיטה כמו עד עתה.
לגבי שריקן מניחים - האלגוריתם בוחר את הגרף
עם הפרק המינימלי.
שריקן מקסימום - בוחר את הפרק המקסימלי מבין
כל הפרקים.

המשרתים צריך בעצם להשתמש ב DFS
BFS - מוצא את המסלול הקצר ביותר. עדיף רלוונטי
לסמן.
BFS עדיף הרבה בזכרון. עדיף מתאים למשרתים.
DFS - פחות זיכרון. המסלול הקצר עדיף רלוונטי ומבין
גם אלגוריתם טוב למשרתים.
יש שני אלגוריתמים. אחד עדיף רוצה ה-A לעשות
והשני עדיף רוצה ה-B לעשות.
אלגוריתם של ה-A קרוב יקר מאוד. לכן הבנים שלו
עם האלגוריתם של ה-B ---

DFS הוא אלגוריתם טוב.
מגדיר קודם על עד שאין מסלים. בעיה יש בגרף
שנקבע עי הגרף הרחבי.
האלגוריתם מחפש את הדרך לפי הבנים הללו.
בעיה לא נשמרים בזכרון אלא מחושבים בעזרת
רקורסיה.
בעיה - רוצים לראות באמת נשלח עמוק בהמשך.
יש פירוט מאוד לכן עדיף הרבה זיכרון.
כבר נתן לחבר את המשמר?
יש להקטין את הרחבה אולי יש בעיה שלו
בגלל לסמן אותם.

אלגוריתם גיבוי $\beta\alpha$

בין הראשון - מרוויח 1 לכל
שחקן מקום.
בין שני, נבדוק ראשון מרוויח
-1. אין טעם לבדוק אל
מול הנכבדים. שהו נקבל טריק
גדול מ- -1, כי קטן מ- 1
ולכן נפריקן מקום לא יהיה בכך.
כעת עלינו בין 3.

β α ∞ $-\infty$ לא להכניס טריק קטן מ- ∞ או גדול מ- β.

מקום קיים לימין.
מרוויח 1.
$\beta\alpha$ כופר לתת 1 ∞-
1 מרוויח לשאר ט' 1.
הבא נכנס אל 1 כ-
מבקש הגדול גדול מ- 1
הוא מכניס לפני השני את הטריפ ∞, 1
לפני השני אין טריפים בין ∞, 1 לכן לא רלוונטי.
בלומר כלשהי יש בדיוק מקום מ- ∞ אין
טעם לבדוק אם שאר הפנים.

(β, ∞) בדיוק מקומות. בודקים שהטריק לא
גדול מ- ∞.
מעדכנים אם β למקום בין β לשאר על הבדיוק.
בדיוק מקומות שונים בודקים הבדיוק.

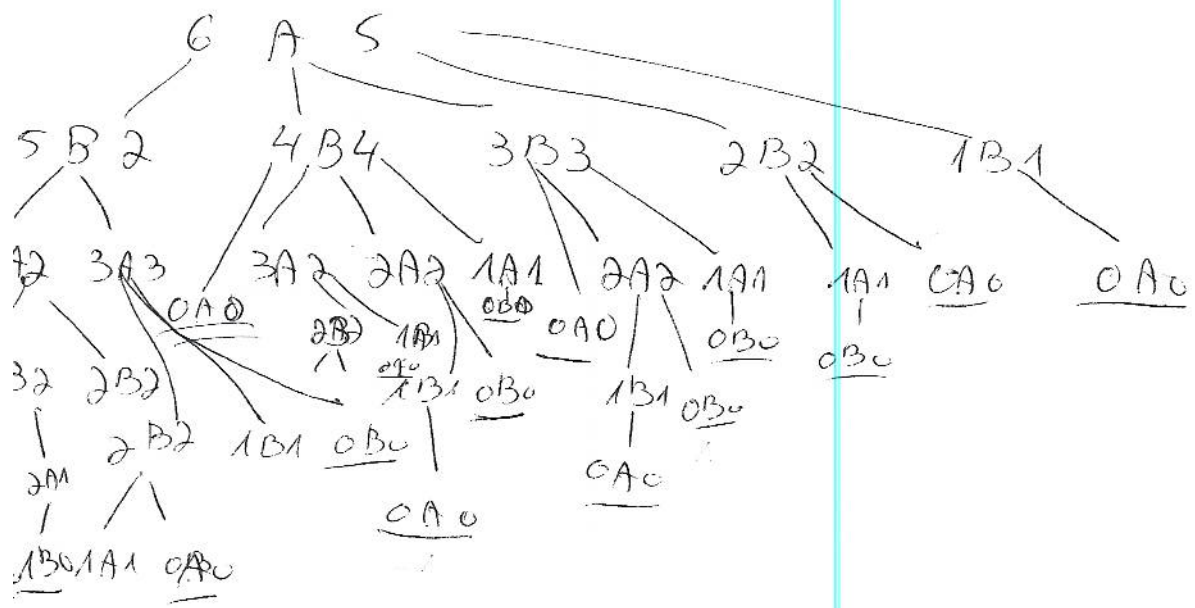
בדומה הגיבוים - הבדומה בה הינו וקבלנו
טרקאתך. אל נשאר הסיבוכים הבנים אז ביקנו
אולם הלמידה הא לא נשגיחנו אותם
אלא אל הבאל נשלחים.

הנוי של בונקרי מנוחה לרגיים לאלגי מסודרים
מבקיין לגדולי (פרט לבדומה הכי שמאלי בכל רמה).
באשר הם מסודרים ניתן לגצו אחר הבאה.

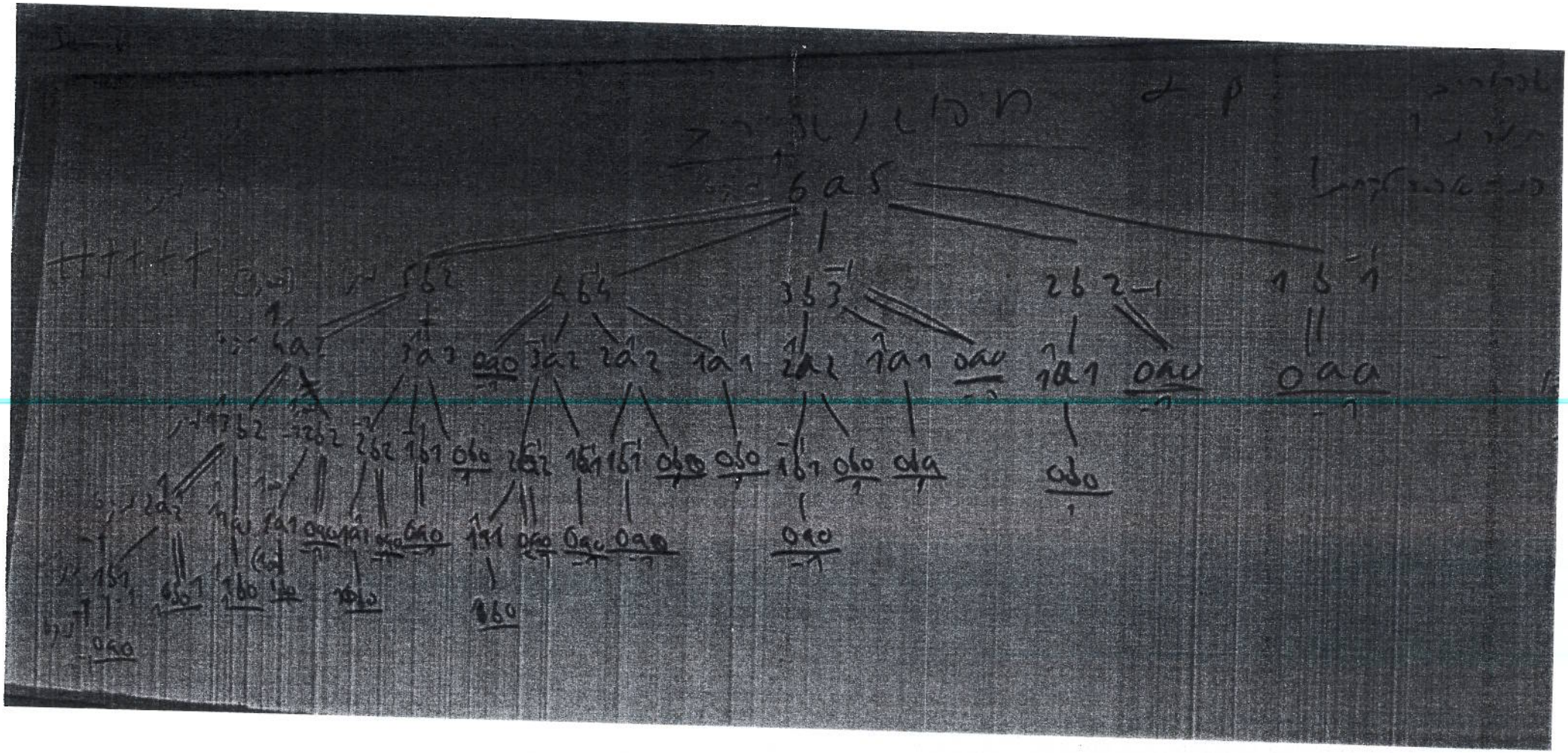
מעשיים

יש שישה אפרורים. כל שריקן לוקח בתור אפרורים
מהמלרומה, לפחות 1 ולכל הכי כבוד ממה שלקח יריבו
בתור הקודם.
הראשון יכול לקחת עד 5 אפרורים.
מי שלקח את האפרור האחרון מנצח.

"לוח מדבר": תור של מי, כמה אפרורים, כמה אפשר לקחת.
כמה נשאר תור אפרורים



נגדר:
A - מנקה
B מנגי
ונגיע לגלגל מרבי



מסקנה מהתרגיל:

שריקן	1	1
	2	1
	1	1
	2	1 או 2
	1	ינדרה

מי נגד מי טבלה כמעט

נתונה פונקציה הלוריסטית הבאה:

$$f(n) = (2-w)\,g(n) + w\,h(n)$$

שאלות:

א) הפונקציה $h(n)$ קבילה. טבלה אלגו טוב w, $f(n)$ תביא אל A^* לפתרון אופטימלי.

קבילה ← תיפלש מלא.

טבלה	שיש	נקבל
$f(n) = 2g(n)$		$w=1$
$f(n) = g(n)+h(n)$		$w=2$
$f(n) = 2h(n)$		

$$f(n) = (2-w)g(n) + wh(n)$$

$$(2-w)\left(g(n) + \frac{w}{2-w}h(n)\right)$$

$$0 < \frac{w}{2-w} \le 1$$

$w \ne 2$ $\qquad$ $0 < w \le 2-w$

$2-w > 0$ מן זה $\qquad$ $w \le 2-w$

$w < 2$ $\qquad$ $0 < w \le 1$

אם $2-w < 0$

$w > 2$

$0 > w \ge 2-w$

$w \ge 1$

לכן $0 > w \ge 1$ לא ייתכן

ולכן הפתרון אפשרי רק עבור $0 < w \le 1$

שאלה:

5		2
4	3	1

בכל שלב ניתן להזיז את הריבוע הריק לתפוס עמדה:

5		3
4	1	2

אין מבלבל - כאשר כולם A^*.

$h(n)$ קבועה - מספר הספרות שלא במקומן.

קבוע כי בכל מהלך ניתן לכל היותר להביא ספרה אחת למקומה

$n' \le n$

הוא הפונקציה מונוטונית? (מקבל)

$h(n) \le c + h(n')$

$h(n) \le 1 + h(n')$

explored - תיאור מרחב.

6.11.13

שיעור 4

חיפוש הוא הבסיס לבעיות אלגוריתמיות ובחינה עיונית.

CSP

נתפשט "אחידות". אם לא הבעיות ניתן אלגוריתם - גם אלגוריתמי החיפוש יהיו פלומים.

CSP = Constraint Satisfaction Problems

בעיות סיפוק אילוצים

נסיים עם בעיה כאשר היא אופן של אילוצים.

בעיות ומלבנים

נשתמש לזה כאל בעיה כ המלבנים.

יש לעשות כ מלבנים בלוח דאג כך שכן לא יהיו אחד A השני. אז באותו שורה, טור או אלכסון.

איך נייצג את הבעיות?

מגדירה לכל תמונה שמכיל את מיקום התחלה בה נתחיל המלבן.

יפלו שאז יתכן עם מלבנים באותה תמונה.

לדעתי כן כדאי להשתמש ב DFS כי הוא
חסכוני בזיכרון.
ל BFS אין יתרון כי הגבלנו לו משקל, רק הגבלה
הסופית.

חיפוש עקלקל - מתבוננים עם הבדיל מלא = נשמים
את כל המלבנים A הלאה, בתנאים בלשכו
התקדמות לאורך הדרום: תצוגה של המלבנים
מציגים את המלבן עמקום שמתחבר אל מספר האיזומים
עד שנגיע ל 0 איזומים או לתוצאה עקלקל. מצב
שלא ניתן לשפר אותו.

בטור חיפוש אלגוריתם אותו נסתכלים על המלבנים
שיש לבן.
אלגוריתם - לקחת מתוך קבוצה של המשמעים להתבונן
אבל חייבים מותר להם לקבל.
נתקדם אל q1q2 ונראה אלו שורות הם כולם
לקבל

q1 q2
(1, 3)
(1, 4)
(1, 5)
(2, 4)
(2, 5)
(3, 1)
(3, 5)
(4, 1)
(4, 2)
(5, 1)
(5, 2)
(5, 3)

אלגוריתם כבר יש לרשום לכל צלע של המשמעים.

נניח שיש בתור פונקציה דרור Q_i, Q_j
כל הפעולות שמקיימים $Q_i \neq Q_j$
$Q_i - Q_j \neq i - j$
$Q_j - Q_i \neq j - i$ } שאז יהיו באלכסון.

בבעיה סיבוך אלגוריתם יש צורך במשתנים שמקבלים ערכים. X
Domain, תחום. הערכים שכל משתנה יכול לקבל. D
Constraint, אילוצים. C
צריך לדאוג לכל אילוצים בגרפים.
אילוץ שנותן מי בגרף ניתן להפוך אותו לבגרף.

פתרון בעיית 5 המלכות.
נקרא אל Q_1 ונכניס שם ערך $Q_1 = 1$.
אבל נצטרך לפחות עם $Q_1 = 1$ נגיע עם מתים אחרים
(כמו נשכן).
עכשיו $Q_2 = \{3, 4, 5\}$
$Q_3 = \{2, 4, 5\}$
$Q_4 = \{2, 3, 5\}$
$Q_5 = \{2, 3, 4\}$

arc-consistency
אם כל המשתנים בגרפים, נוכל לעבוד את הקשתות
בגרפים בלבד.
כל משתנה - קודקוד בגרף.
רושמים מעל אל תחום הערכים שלו.
קשת בין כל המשתנים + האילוצים.

בדיקה arc-consistency

נקיר כל צלע משמאל ונבדוק

אם $q_3 = 3$ יכול להיות לו ערך q_3 הפוך ל 2

$q_2 = 4$

$q_2 = 5$

הפלט אלגוריתם

בודקים כל צלע: (q_2, q_3), (q_2, q_4), (q_2, q_5), (q_3, q_5), (q_3, q_4), (q_5, q_4)

נבדוק את הצלע ונראה אלו ערכים נמחקים לאחרים

$q_2 = 4$, $q_4 \leftarrow 2, 5$

אם הבדיקה מבלבלים מול הערכים נמחקו

לאחר שסיימנו את q_1.

לאחר הבדיקה:

$q_2 = \{3, 4, 5\}$ $q_4 = \{2, \cancel{3}, 5\}$

$q_3 = \{2, \cancel{4}, 5\}$ $q_5 = \{2, 3, 5\}$

בינתיים את הערכים הכן יש לבדוק בדיקה נוספת שמתבצעת בתחום הערכים הרגיל.

הבדיקה יכולה תקנה

עכשיו יש לחזור משמאל לימין אחרי.

הוריסטיקה אחרת שיש לבחור משתנה ערך שלא

ערך יפה בתחום.

נבחר $q_3 = 2$.

$q_2 = \{4, 5\}$

$q_4 = \{5\}$

$q_5 = \{3\}$

בבדיקה אלגוריתם נקבל $q_2 = \{4\}$

והפתרון הוא:

אלגוריתם AC-3 מבוסס על בדיקת האילוצים

AC - מרחיב נאיבי של האלגוריתם הבא
הפה שיש שינוי בודק את הכל.
AC3 בודק שינויים.

AC-3 עובד מבנה נתונים תור.
1. תור עם כל האילוצים.
2. כל עוד התור לא ריק
הוצא אילוץ x_i, x_j מהתור.
לכל $a \in D_i$ בודק שיש $b \in D_j$
כך ש $(a,b) \in C_{x_i x_j}$

אם אין כזה מוחק את a מ D_i
אם D_i קטן, מוסיפים לתור את כל האילוצים (y, x_i)

יש d מצבים, c אילוצים.
מהו זמן הריצה של האלגוריתם? $O(cd^3)$

הבדיקה: הוצא אילוץ. $O(1)$
מה $a \in D_i$ כמו... $O(d^2)$
כל אילוץ שהוסרנו, ה domain של x_i קטן ב 1.
אי אפשר יותר מ d פעמים לעדכן כל האילוצים (c)
מקסימום d פעמים. לכן הפעולה: לוקח... $O(cd)$
וסה"כ זמן: $O(cd^3)$.

אלגוריתם הפרדת האילוצים הוא חלק מה DFS.

אלגוריתם AC-4 אותו לומדים לפי סדר מי הלכים שאותם הוא.

בשונה מהדרכים הנ"ל כל האילוצים נשמרים נפרד יש X אילוצים, גודל התור הוא X².
כלומר מספר אילוצים שבאלגוריתם לבדוק לדיבוק משתנה הבודדים לפחות כל אילוץ נמצא בפחות: (X_1, X_2) ו- (X_2, X_1).

אם בתור (X_1, X_2) האילוץ היה נבדק והכל תקין.
אם בתור יש (X_2, X_1)
נניח לתור אל 4 תור X_2.
מחפשים בתיכה ל- (X_1, X_2) לתור הבודקים שנמצא.

פונקציה לבדיקה אם קשת
4 אנשים נפגשים
יכולה כ:
1. הפגישה מושבה כל פעם
2. כל אדם נכון בפגישה בזמן רגיל.
3. Jones נכון בתחילת הפגישה
ו- White בסופה
4 Smith היה נוסה לבל הבא אחרי ש Jones
עבר אז עשינו אם עבר באותו זמן.
5. Brown, White ו- Smith נפגשו
הלאה ובלי להלוא שאין אוכל הפגישה?

נגדיר אל המשתנים: S, J, B, W, M
מי לפגישה.
לכל אחד מהם יש לפחות מי הבא אחרי עבר S, F.
S_S S_F J_S J_F B_S B_F W_S W_F M_S M_F

הסולמות:

$JS = MS$
$WF = MF$

כדי שנוכל להשתמש בזמנים בקלות יש להפוך אל המספרים.

מה שמתאים זה היחס בין האינטרוולים
המשמעות: אבל מה היחס בין האינטרוולים.
MJX זה משמעה שאותו מה היחס בין שתיהן לפגישה.

$XJM = \{starts, equals\}$
$XMJ = \{is started by, equals\}$ המשמעה ההפוך, הנגדי.
$XWM = \{Finishes, equals\}$
$XOM = \{starts, during, Finishes, equals\}$
(B S)

$XSJ = \{after, finishes, is finished by, equals\}$

סך הכל הגדרנו אל ה- Domain.

מהם האינטרוולים?
האינטרוולים הם היחס בין שלושת של אנשים.

X_{ab} X_{bc} X_{ac}
before before before

לדוגמא:

בעיית CSP מורכבת מ:

משתנים X

תחומים D

אילוצים C

באילוצים מופיעים אם כל הערכים עליהם צולבו של המשתנים שכולם לקבל - אילוצים בינאריים.

כל בעיית אילוצים ניתן לפרוק לבעיות אילוצים בינאריים.

ניתן לפתור את הבעיה בעזרת חיפוש (DFS) הפצר אילוצים.

נשתמש ב- arc-consistency בעזרת אלגוריתם AC-3

DFS יפרוס את הגן עם התחום הקטן ביותר. זה גורם שרק שגורם הפרות והיה הקטן ביותר הלפחות.

arc-consistency:

לכל אילוץ $C=(x_i, x_j)$

מתבוננים לכל ערך בדומיין של x_i יש ערך בדומיין של x_j שעומד באילוץ.

שמורה לה AC מבטיח שנקטין את הדומיין.

Path-Consistency טיפול מפלגה

לכל m משתנים $x_1 .. x_m$, לכל צמד ערכים קונסטלציה ל x_1, x_m הוא ערכים ל- $x_2 ... x_{m-1}$ שהם קונסטלציה לאורך המסלול

x_1, x_2 x_2, x_3 x_3, x_4 ... x_{m-1}, x_m

משמעות: אם יש רשימת ערכים, הצמד בסדר. אם אין Path.Con. אלא אומר שיש צמד ערכים שלא ניתן לבנות אותם.

בשונה משמרים על Path-Consistency ניתן לבנות אם הדומיין יהיה בשונה בעזרת arc-cons. אלגוריתם. PC-2.

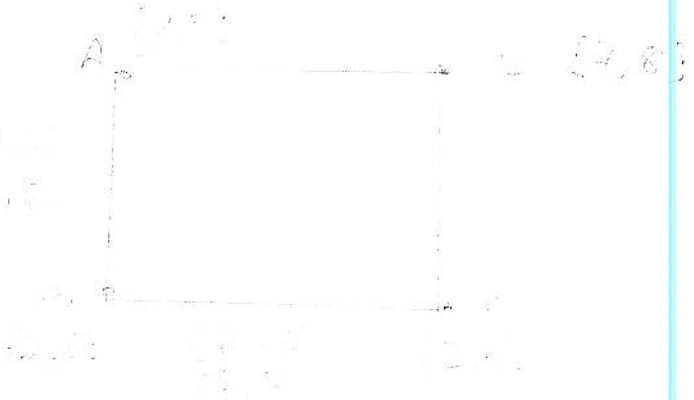
לכל 3 משתנים:
X_1 X_2 X_3
לכל צמד ערכים קונסטרינטים ל- X_1 X_3
יש ערכים ל- X_2 כך ש- X_2, X_3 X_1, X_2 קונסטרינטים.

Path-Con לכל 3 צמתים מתקיים משפט הבולדים
אינדוקציה על m: Path-consistency $\Leftarrow$ 3 צמתים
בסיס כאשר $m=3$ $\Leftarrow$ נכון שלכל.
3 מקרים שמתשמש מתקיים לכל $3 \le m < k$
נוכיח שמתשמש מתקיים ל- $m=k$.
$X_1, X_2, X_3 \dots X_m$

נסמן לכל: X_1, X_2, X_m

נניח X_2 כך ש- X_1, X_2, X_2, X_m קונסטרינטים.
נניקח $X_2 \dots X_m$. נבנה X_3 בתחילה, לאם X_m.
לפי הנחת האינדוקציה יש $X_3 \dots X_m$ קונסטרינטים בצורה
קבלנו $X_1, X_2 \dots X_m$ כנדרש.

דוגמא
ערך: יש לבדוק בגרף אינדוקציה כך שלא יהיה Path-Consistency
אלא יהיה פתרון.



אם בחרנו $A=1$, $C=7$ - אין פתרון
אם נבחר $A=5$, $B=6$, $C=7$ - יש פתרון
לפי Path-Consistency

13.11.13

תרגיל 5

מתמן 13

שאלה 3: אין הוכחה. אלגוריתם חמדני לאלגוריתם בינארי.

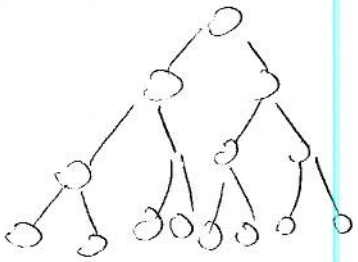
שאלה 1 - חיפוש נתיבים. גבולות β.

יש לקחת על. להגדיר את הגדרה כללית, מתאימה לכיוון.

לפרק מנימוקים.

להשתמש בגבולות β - כמה זה. הגדרה מתאימה

ב) β ו- δ הגדרה



אין גבולות - הבעיה עדיין מלאה β על הבעיה

שאלה 2: CSP. שינוי עדיפות

צמתים: 0, 1, 2

בגלל מסלולים. להרחיב β ה הבעיה. נניח B,C

א) המשתנה B,C: על הערכים (0,1,2)

ש B ו- C הבעיה לקבל.

אם B מתבטלה ב- 0, צמתים ב הוא

הקבוצה כאן הן בעיה B ו-

B, C

0, 1

0, 2

~~1, 0~~

1, 2

2, 0

~~2, 1~~ כל ה

כמו ל- A,C , A,B

ב. אם אלגוריתם נקרא לכל אחד מהמשפטים WFF הקיימים לפניו זה אלגוריתם.

ג. בדיקה בדידה - לבדוק קומבינטוריות של הקשרים הלוגיים לבדיקה.

ד. אם A לא שווה ל - לבדוק תקינות.

ו. יש לבדוק תקינות. לבדוק מה הנוסחה הזו הכי מעל תקינים.

~~בדיקה יש לה~~

בשביל מלתרים נוסחאות - בלוגיקה הנוסחאות מתם הרבה מעל עדכנים. מבנה הפנים - אל הדיק שמעל הכי מעל תקינים.

CSP

לוגיקה להגדרת הבעיה.
מנסים להגדיר שפה פורמלית.
לצורך מהו ה. אל"ב על הלוגיקה.
Well-Formed-Formula - איך נראה נוסחה נכונה בכמו נפלרים.
אקסיומות - WFF שמגדירים אותה לדוגמה אמיתי, שטוח יש להגדיר את אופן WFF שהם אקסיומות.
הגדרה כללי הסק.
כלל הסק - אם יש... WFF1, WFF2 אז אמיתי, ניתן לסמוך מהן נוסחה אחרת.
הבעיה: WFF1, WFF2.. ⊢ WFF

הבעיה מהן נסמוך - בהינתן WFF וכלל הסק, האם ניתן לסמוך נוסחה מסוימת?

דוגמאל.

אל

$0, t, p, S, !, C, >$

אטומים: 0

אם α ו- β הם אטומים אז גם $(\alpha t \beta>$

$(\alpha p \beta>$ $S\alpha$ הם אטומים

WFF

אקסיומות:

1. $(\alpha p 0> ! \alpha$
2. $(\alpha p S\beta> ! S(\alpha p \beta>$
3. $(\alpha t 0> ! 0$
4. $(\alpha t S\beta> ! ((\alpha t \beta> p \alpha>$

חוקי הסק:

1. אם α β ניתן להחליף הנחה. בכל נוסחה הם מופיעים
2. אם $\beta ! \alpha$ אז $S\beta ! S\alpha$
3. אם $S\beta ! S\alpha$ אז $\beta ! \alpha$

כל האקסיומות הן נוסחה שרירה לדוגמא.
כל נוסחה היא $\beta ! \alpha$ ובכל הנוסחאות יש סימן קריאה.

נוכיח את הנוסחה הבאה:

$(S0 p S0> ! SS0$

S

א) נוכיח את אקסיומה 2. נחליף:

$\alpha = S0$

$\beta = 0$

$(S0 p S0> ! S(S0 p 0>$

אם נוכיח $(S0 p 0> ! S0$ (לא להחליף $S0$ בנוסחה

ב) נחבר לפי $(S0 p 0> ! S0$ ונשתמש באקסיומה השלישית

ולכן: $\alpha = S0$ $A \times 1$

~~$(S0 p 0$~~

לפי אל + ר, רוק היסק 1 נקבל
(סס1<סקסס)
מ.ש.ל.

מסתבר שבעזרת לוגיקה + חוק היסק מתאים
ניתן לתאר כל תהליך "חישובי" כל אלגוריתם.

~~לכל~~ בא"ל ישלב אות סימן:
0, t, p, s, 1, c, <
( ) = סוגר + x אופס

פאקטורים:

1. (x + 0) = x
2. (x + (x+1)) = (x + x) + 1
3. (x × 0) = 0
4. (x × (y+1)) = ((x×y) + 1)

פרדיקט הפורקים, של לסיות.

בלוגיקה יש שירה a → b, אם a נכון, b נכון,
אפ a לא נכון, b יכול להיות כל דבר:
ולכן אין שווה ל- a∧¬b → b.

מדהים: כל הפורקים שרירותיים.
כי, אלוגשים את ההסתרה ואת הרבה פורקים שרירי
כל פורק שרירי ולפי יאלוגשים את האמיתיים.

~~לכל~~ כל מה שבא אם שרירי הוא לא פורק.
שני המשפטים שקולים. אולם הראה "מעשה"
השני אם מאלוגשים את המעשה האמיתי.

אלגוריתמים. כל מה שנעשה למדעי המחשב
לתחום בעצם לוגיקה.
שפת prolog ראשית נעשית לתכנות בעצם מושגים
לוגיים.
כל הקודמים כי prolog הם מושגים בלוגיקה.

לוגיקה שנבנה אלמנטרי
הנחות הפסוקים - חלשה
לוגיקה מחסיר אלמנטרי. - מורכבת

תחשיב פסוקים

יש אטומים ואופרטורים לוגיים.
אטומים - נרשם באותיות. $P, Q, \ldots A, B$
כל אטום יכול לקבל ערך True או False
אופרטורים - $\wedge \quad \vee \quad \neg \quad \rightarrow \quad \leftrightarrow$
and or not implication equivalence
שקילות גרירה

$P \wedge Q$ שניהם T
$P \vee Q$ לפחות אחד T
$\neg P$ F
$P \rightarrow Q$ זה שקול ל $\neg P \vee Q$
כאשר F ⊢ P
$P \leftrightarrow Q$ כאשר יש להם אותו ערך.

הסיב יחד. אולם כל הנוסחאות נשארו ובעלים
שהם אמת.
מתוך הסיב היחד, בעצם הלוקח היסק, ונעשה
לפרשים נוסחאות תקשורתיות.
טאוטולוגיה - פסוק שהוא תמיד נכון, לא
משנה אילו ערכים נכניס בו למשל: $P \vee \neg P$

כתיבה - תמיד לא נכון. דוגמה: $p \wedge \neg p$
סופית - הסוף שנתן להגיד בו אפשר שיהיה אמת.

נגדה - כאשר מהלך הקודם נוסחאות נכון ניתנה ל.

נוסחאות $\models$ נוסחאות
$\{p, p \to q, q \to s\} \models s$
נכון - בכל הדרכה שנעשיה, כאשר הפסוקים יהיו אמיתיים, גם s יהיה אמיתי.

α נכון מ β $\equiv$ בכל הדרכה בה β שקר, α שקר גם הוא.

הסקה - מהלך הסק ניתן להסיק את α לפי β.

כל הדרכה = מודל.
הדרכה למודל תהליך שלם לוקחו את α מ β כאשר α נכון מ β. כלומר אלגוריתם יודע שימצאו הסקה.
נייצג את הקלדה האמיתי בצורה נוסחאות
מודל חוק הסק + אלגוריתם.
אם האלגוריתם ~~[illegible]~~ + חוקי היסק מגדה $\alpha \vdash \beta$ נרדה שי $\alpha \models \beta$
בצורה שפאלגוריתם מוכיח דברים נכונים, אמיתיים, נאותות.
נרדה גם שלמות - כל מה שנכון ניתן להסקה שלמות: אם $\alpha \models \beta$ אז האלגוריתם + כלל הסק יכולים להוכיח $\alpha \vdash \beta$

כלל היסק נלה - מישהו ניתן להסיק הכל.
" " נאות - כלל הסק ריק.

כדי נוסחאות הדיסק יהיה יציג אלו להיות נוחות ופשוט.
נעסה לאחדו חלקי הדיסק (אלגוריתמים)

CNF

נצטרך לסחאות מתרגשים הפסקים רק
מהצורה הבאה: $\neg R \vee P \vee \neg Q \vee S$
רק שם יש $\vee$, $\wedge$ אם
נוסחה בנויה בטבלה משמעה, הוא למשמעה אם

דוגמא

אם הרגיל קרן הולך מיקי אל הלא בן אלמוות.
אם הלא לא מיקי אל הלא בן מוות הלא יונק.
אם הלא בן אלמוות אל יונק אל יש לו קרן
אם יש לו קרן אל הלא קסום

נהפוך את הקשרות לתרגשים פסקים.

A - מיקי
B - בן אלמוות
C - יונק
D - יש לו קרן
E - קסום

$A \to B$
$\neg A \to \neg B \wedge C$
$B \vee C \to D$
$D \to E$

נהפוך את הפסקים ל- CNF. רק $\vee$, $\neg$
שמרגיל. $B \to$ ש שקול $\neg B \vee$
מוגדרים ניתן לפתרון.
בשביל יש לנו $\neg a$ בין שני משפטים ניתן להפריד
אל $\neg a$ לשני משפטים נפרדים.

$B \vee C \to D$

$\neg(B \vee C) \vee D$

$(\neg B \wedge \neg C) \vee D$

$\neg B \vee D \quad \neg C \vee D$

נפרק סוגרים ונקבל 2 פסוקים

$D \to E$

$\neg D \vee E$

קיבלנו את הפסוקים הבאים:

1) $\neg A \vee B$
2) $A \vee \neg B$
3) $A \vee C$
4) $\neg B \vee D$
5) $\neg C \vee D$
6) $\neg D \vee E$

אם הכול ב-CNF יש כלל היסק נאות ושלם.
כל מה שנגזר בעזרת יחסים בעולם האמיתי,
כל מה שיש בעולם האמיתי נוכל לגזור בעזרתו.

רזולוציה - כלל היסק.

אם יש $a_1 \vee a_2 \vee \dots \vee a_n$

$b_1 \vee b_2 \vee \dots \vee b_m$

a_i ו- b_j הם אותו משפט. אחד מ- not
והשני ללא not.

ניתן לגזור מה שגם בעולם בלי a_i, b_j.
הם מבטלים אחד את השני.
כלל היסק זה (רזולוציה) הוא נאות ושלם.

$a_1 \vee \dots a_{i-1} \vee a_{i+1} \dots \vee a_n \vee b_1 \dots \vee b_{j-1} \vee b_{j+1} \dots \vee b_m$

$\neg A \vee B$

נניח - הנחה קיימת הלא מוגדרת (A)
נשתמש הגדרה על הכל
בד"ר שקיים אי שקר המשפט.

7.(1,3) $B \vee C$
8(1,4) $\neg A \vee D$
9(3,5) $A \vee D$
10(3,8) $C \vee D$
11(4,6) $\neg B \vee E$
12(5,6) $\neg C \vee E$
13(5,7) $B \vee D$
14(5,10) D
15(6,8) $\neg A \vee E$
16(6,9) $A \vee E$
17(6,10) $C \vee E$
18(6,13) $E \vee B$
19(6,14) E

דוגמא:

$A \vee B \vee \neg C \vee \neg D$
$D \vee E \vee \neg F$

לפי הגדרה ניתן לכתוב

$A \vee B \vee \neg C \vee E \vee \neg F$

לעשות כך נכון.
ידוע כי $\neg \alpha \vee B$ ⇐ שקול $B \to \alpha$
לכן ניתן לכתוב:

$D \to (\quad)$
$\neg(\quad) \to D$

לפי הטרנזיטיבי

$\neg(\quad) \to (\quad)$
$(\quad) \vee (\quad)$

לפי הכלל:

המטרה - להבין כאן בעזרת הגדרות פסוקים מהם אלגוריתם לפיסקה.
נרצה אלגוריתם שתמיד יגלה לפסוק אותו דבר.
האלגוריתם יקבל אקסיומות ויסיק אותן דבר.
כאשר נרצה לדעת מתמטי נוסף נוסף לו, ϕ אל הטבלה שלו, כאקסיומה.
~~[illegible]~~ אם נשים בתירה המשפט נכון.
אם לא, לא הגלתנו לפלוגיה אחר.

נניח המשפט EVA, נוסף $\neg(EVA)$

לפי דה מורגן $\neg E \wedge \neg A$

נוסף לפנים היגד $\neg E$

$\neg A$

שיעור 6

20.11.13

לוגיקה מסדר ראשון.

תחביר הפסוקים

א, ב, ג קבועים. מקבלים ערכי אמת T או F

אופרטורים $\neg \quad \vee \quad \wedge \quad \rightarrow \quad \leftrightarrow$

$\neg p \qquad p \rightarrow$

$p \vee q \qquad \neg p \vee q \equiv p \rightarrow q$

$(p \rightarrow false) \rightarrow q$

כלל היסק.
הצלבה

$$\begin{cases} a_1 \vee a_2 \dots \vee a_m \\ b_1 \vee b_2 \dots \vee b_n \end{cases}$$

$a_1 \vee a_2 \dots a_{j-1} \vee a_{j+1} \dots \vee a_m \vee b_1 \vee \dots b_{i-1} \vee b_{i+1} \dots \vee b_n$

שלילה. אפשר להסיק אם כל מה שנכתב

$a \models b$

$\underbrace{p \rightarrow q,\ \neg q}_{a} \models \underbrace{\neg p}_{b}$

נכון - כאשר a מקבל ערך T אז גם b מקבל T.
זה קורה כאשר $q=F,\ p=F$

כלל הרזולוציה הוא שלם.
כלל הרזולוציה הוא גם נאות.
נאותות - כל מה שניתן להסיק זה נכון.

$a \rightarrow b \qquad \neg a \vee b$

$(\alpha) \rightarrow a$

$a \rightarrow (\beta)$

$(\alpha) \rightarrow (\beta)$

$(a_1 \vee \dots \vee a_{m-1}) \vee a$

$(b_1 \vee \dots \vee b_{n-1}) \vee \neg a$

לכתוב אל עבר המשפטים האלו בצורה:

$\neg(a_1 \vee \dots \vee a_{m-1}) \rightarrow a$

$a \rightarrow (b_1 \vee \dots \vee b_{n-1})$

ומכאן ניתן להסיק

$\neg(a_1 \vee \dots \vee a_{m-1}) \rightarrow (b_1 \vee \dots \vee b_{n-1})$

לפי $a \rightarrow b$ $\quad b \vee \neg a$

ניתן לכתוב אל המשפטים כ: $a_1 \vee \dots \vee a_m \vee b_1 \dots \vee b_{n-1}$

בהתשיב הפסוקים יכולים הבינוי מאוד גדולה.
לכן ניתן להגדיר כל פסוקים מורכבים. יש צורך בעפה לשירה יותר.

לוגיקה מסדר ראשון

העולם מורכב מעצמים. בדר נגדיר אותם באותיות גדולות
T/F

פונקציות עצמים בעולם <= עצמים בעולם.

Unicorn - עצם

Father-of(Unicorn) פונקציה שמעניקה אב האבא.

עצמים בעולם <= T/F

יחס - פונקציה שמקבלת עצם ומעניקה T/F לפי התשובה.

dead(Unicorn)

bigger(x, y) מגדיר T אם x גדול מ y.

T/F <= עצמים - אין צורך בכך.

T/F <= T/F $\quad \neg \wedge \vee \rightarrow \leftrightarrow$ אופרטורים.

בלוגיקה מסדר ראשון ניתן להגדיר משתנים של עצמים בעולם.

לא ניתן להגדיר משתנה שהערך שלו הוא יחס, T/F אופרטור

משתנים x, y, z

כמתים $\forall x$ לכל x $\exists x$ קיים x.

בוליאנים מפרק.

קטס פרדיקטים מתאר את היחסים
$Parent(x,y)$ מקבל שני טיעונים. מתקיים T אם x אב של y.

משפט א: אם y הורה של x ו- y מתחתן עם בן הזוג
של y הוא הורה של x.

שבעה המשפטים הראשונים מתארים את הטוטלה.
יש לבנות את המשפט ג.
משפט ד: מתקיים לכל x ולכל y. $\forall x \forall y$

יש לבטא את המשפטים ל- FMC כדי שיהיה לנו כלי
רציונלי.
כיצד נראה FMC לוגיקה מסדר ראשון?
$\forall x \forall y \forall z \; Rel(\ldots) \vee Rel_2(\ldots) \vee \ldots \neg$
את כל $\forall x$ לא מתיחס כאן, לגרוע הקיצור.

משפט ד: הלבבים מגוירה ל v.
משפט ה: מגוירה + דה מורגן.

$$P(z,x) \leftarrow P(y,x) \wedge m(y,z)$$
$$P(z,x) \vee \neg(P(y,x) \wedge m(y,z))$$
$$P(z,x) \vee \neg P(y,x) \vee \neg m(y,z)$$

משפט ו - דרוש הוכחה (בגלל אותו מלא לבוא לבוא)
לבסוף כדי לא שלילה.
אם מתקיים P אם $\neg P$ = סתירה.

כאשר יש שני יחסים, אחד עם ¬ ואחד בלי, ניתן לבדוק אם שאר הפרמטרים ולהתלק אל היחס הזה יש לוודא שהפרמטרים בתוך היחסים מתאימים.

Unification, האלגוריתם - בדיקה שקולים בתוך הביטויים שלהם.

¬ Rel(,)
Rel(,)

א) יש לבדוק שמאחר באותו היחס.
בדיקה שאותו מספר פרמטרים קיים בשניהם.
ב) האלגוריתם בין הפרמטרים של היחס.
פרמטר יכול לפעול עצמו, משתנה או קבוע או פונקציה.

משתנה מול משתנה. x, y.
הצבה x = y

ניתן לבטא באמצעות קיום כאן לכל. $\exists x\, p(x) \equiv \neg \forall x \neg p(x)$

משתנה וקבוע: בחירה של קבוע נכון הצבה.
x bamba
נמצא x = bamba

¬ Rel(x, Father-of(x))
Rel(bamba, bisli)

x = bamba
¬ Rel(bam, Father-of(bam))
Rel(bam, bis)

קבוע מול קבוע. אם הם שווים ניתן להמשיך בהאלגוריתם.
פונקציה מול משתנה: פונ' x, ניתן לשלוח בין אם הם שווים: פונ' = x

אני (I) התחתנתי עם אלמנה (W). אבי (F) התחתן עם בתה (D).
צריך להוכיח שאני הסבא של עצמי.

1. parent (F,I)
2. parent (W,D)
3. married (I,W)
4. married (F,D)
5. married (y,x)←married(x,y)
6. parent (z,x)←parent(y,x)∧married(y,z)
7. grandparent(z,x)←parent(z,y)∧parent(y,x)
8. **? grandparent(I,I)**

ב-CNF:

1. parent (F,I)
2. parent (W,D)
3. married (I,W)
4. married (F,D)
5. married (y,x)∨¬married(x,y)
6. parent (z,x)∨¬parent(y,x)∨¬married(y,z)
7. grandparent(z,x)∨¬parent(z,y)∨¬parent(y,x)
8. ¬grandparent(I,I)

ההוכחה:

(3,5) 9. married(W,I)
(9,2,6) 10. parent(I,D)
(1,4,6) 11. parent(D,I)
(11,10,7) 12. grandparent(I,I)
(12,8) 13.----

ההפרכה.

3. married(I, W)

5. $m(y, x) \vee \neg m(x, y)$

3,5 → 9

$x = I$

$y = W$

$m(W, I)$

נבנה הפרכה.

9. $m(W, I)$

2. $P(W, D)$

6. $P(z, x) \vee \neg P(y, x) \vee \neg m(y, z)$

הפרכה:

$y = W, \quad z = I$

$x = D$

10. $P(I, D)$ למרבל

1. $P(F, I)$

4. $m(F, D)$

6. $P(z, x) \vee \neg P(y, x) \vee \neg m(y, z)$

11. $P(D, I)$

$y = F, \quad x = I$

$z = D$

כאשר מגיעים לשורה ריקה ← הפרכנו את הטענה.

בעיה נוספת.

יש לבנות את המשפטים הלוגיקה מסדר ראשון.

1. פרד כלב קולי. פרד = שם עצם, קולי - ציר.

collie(Fred)

2. כל. קבוע, שם עצם. בלתי. יחס.

6. אם יש שני משפטים לא קשורים שם ∧.

ספניאלים הם כלבים טובים.

$\forall x \; spaniel(x) \rightarrow good(x)$

$(\forall x\ \neg spaniel(x) \vee good(x)) \wedge (\forall y\ (\neg collie(y) \wedge trained(y)) \vee good(y))$

נרצה לבצע את המעבר הזה לפי לוגיקה.

לפני אלגוריתם לצורה קנונית.

ניזכר מהגדרה: $a \to b \quad \neg a \vee b$

נשתמש בפה מרכאן כדי לבטל את $\wedge$

$(\forall x\ \neg spaniel(x) \vee good(x)) \wedge (\forall y\ \neg collie(y) \vee \neg trained(y) \vee good(y))$

עתה יש לבטל מהכמתים אל נוכל לבטל את הכמתים הבאים $\forall x\ \forall y\ (\ldots)$

מתוך הנחה $\wedge$ באינטגרל - ניתן לבטל שני משפטים שונים.

ד. אם כלב הוא לא מאולף אז הוא יתנהג טוב בבית.

$\forall x\ (good(x) \wedge \exists y\ Master(y,x) \to Loc(y,x))$

בכל מקרה אחרת שונה.

נרחיב את הגדרה ואז נבטל זה מרכאן.

$\forall x\ (\neg good(x) \vee \neg \exists y\ Master(y,x) \vee Loc(y,x))$

נבטל את $\exists y$ ל $\forall y$

$\forall y\ \neg master(y,x)$

ועתה ניתן לבטל מהכמתים

$\neg good(x) \vee \neg master(y,x) \vee Loc(y,x)$

נשאלה: הוא פרד בחוץ אמן? הנתון מעל פרד בחוץ אמן טוב נוכל בהכרחה.

הכלכלה בין מדען של סבר. לפי כלכלנים, קל לבלבל אם ניקח מטפל

שמתאחד עליו ונעשה 16 הפלגה עם המשפטים שלפני.

מתחילים ממשפט 11 בכפוף הכללים לעילה.

17 (11, 8) $\neg g(c) \vee \neg m(c,d) \vee \neg Loc(d,e) \vee$
$\underbrace{Loc(c,e)}\ \underbrace{\neg Loc(f,m)}$

$c=f\ ,\ e=m$

$\neg g(f) \vee \neg m(f,d) \vee \neg Loc(d,m)$

18 + 7 : $\neg c(\overset{F}{b}) \vee \neg t(b) \vee \overset{F}{g(b)}$
$\neg g(f) \vee \neg m(f,d) \vee \neg Loc(d,m)$

16: $\neg Loc(s,m)$
10: $\neg day(sat) \vee W(sat) \vee Loc(s,m)$
$\downarrow$
17. $\neg day(sat) \vee W(sat)$
נלקח גם 3 ו- 4 ונישאר עם משפט ריק.

שאלה.

הילד לבד בבית ובשלון מרקים:
אם הבית שלו בשלון אל הילד שבבית.
מרקים: בבית, אינו מרקים.

$\forall x,y,z$ אינו בבית מרקים $\quad Loc(x,y) \wedge Loc(z,y) \wedge Mas(x,z) \rightarrow Day(SAT)$

נרשום בסגנון אחר:

$$\underbrace{\left(Mas(x,z) \wedge Loc(x,y) \rightarrow Loc(z,y)\right)}_{\downarrow} \rightarrow Day(SAT)$$

$$\neg Mas(x,z) \vee \neg Loc(x,y) \vee Loc(z,y)$$

בשביל מתמטיקה בשביל הכול T, מה שהוא T.
אולם בשביל הפרדוקס F ← לא צריכים.

נלך אל הנשלה:

$$\neg(\forall x,y,z \quad Mas(x,z) \wedge Loc(x,y) \rightarrow Loc(z,y)) \rightarrow day(sat)$$

$$\neg((\forall x,y,z \quad \neg Mas(x,z) \vee Loc(x,y) \vee Loc(z,y)) \rightarrow day(sat))$$

$$\neg(\neg(\forall x,y,z \quad \neg Mas(x,z) \vee Loc(x,y) \vee Loc(z,y)) \vee day(SAT))$$

נשלם ל- ¬ הפנימי