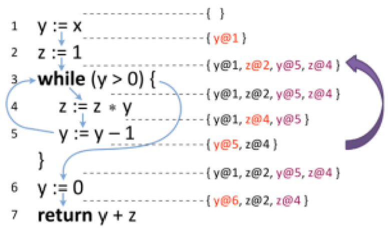
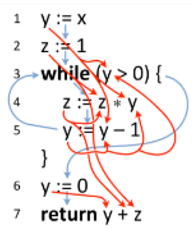
# אנליזת Dataflow

10:55 PM Wednesday, October 7, 2020

## דוגמה לניתוח באמצעות אנליזה סטטית - Reaching Definition:

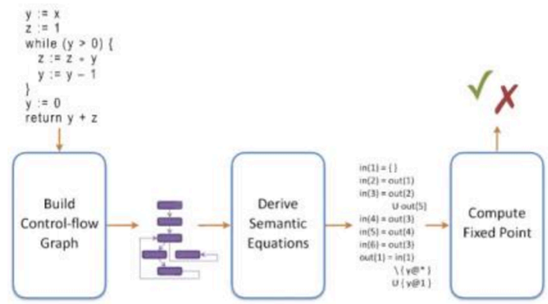



קבוצת ההגדרות שיכולות להגיע למיקום מסוים. למשל - y שהוגדר בשורה 1.



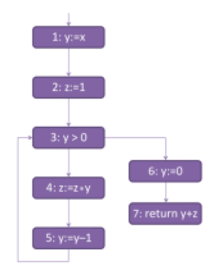

* קונספט של הגדרה ושימוש: $x = y+2$
  - שורה זו היא **הגדרה** של x.
  - שורה זו היא **שימוש** ב- y,2.
* הגדרה **מגיעה** לשימוש.
  - אם ישנו ערך שנכתב בהגדרה.
  - ישנו מסלול ריצה בו אין דריסה של הערך ובסופו נקרא הערך.

## אנליזת זרימת נתונים - מבט על:



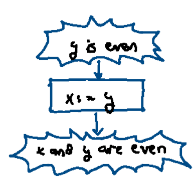

* בנינו את ה-Control Flow Graph.
* לכל בלוק ב-CFG הגדרנו מה המצב בכניסה וביציאה ממנו כאשר ה-in של שורה הוא בפועל השורה שלפניה כולל בקפיצות.
* ממשיכים באנליזה עד שאין שינויים-fixed point.

## גרף זרימת הבקרה:




## פונ' מעבר:

* בהינתן מצב של קוניקת S, נגדיר **פונ' מעבר** $T_S$ שמקשרת בין המשפטים שנכונים לפני המצב לבין המשפטים שנכונים לאחר המצב.




$$T_{x:=y}\begin{bmatrix} x(?) \\ y\ \text{even} \end{bmatrix} = \begin{bmatrix} x\ \text{even} \\ y\ \text{even} \end{bmatrix}$$

# יחס סדר חלקי

10:55 PM　　Wednesday, October 7, 2020

* בהינתן קב' $P$
* ובהינתן יחס בינארי $\sqsubseteq$ ("קודם או שווה ל") המקיים לכל $x,y,z$:
  - $x \sqsubseteq x$ (רפלקסיביות)
  - $x \sqsubseteq y$ וגם $y \sqsubseteq x$ גורר $x = y$ (אנטי-סימטריות)
  - $x \sqsubseteq y$ וגם $y \sqsubseteq z$ גורר $x \sqsubseteq z$ (טרנזיטיביות)
* אזי $\sqsubseteq$ הינו יחס סדר חלקי
* וקב' $P$ שעבורו קיים יחס כזה הינה קב' סדורה חלקית (partially ordered set) ובקיצור poset.
* בניגוד ליחס $\leq$ (למשל שבו יחס סדר מלא).
* יחס סדר חלקי משמש להגדיר:
  - גבול עליון ותחתון.
  - גבול עליון מינ'.
  - גבול תחתון מקס'.

## דוגמאות ל - Poset:

* תתי-קב' של קב' מסוימת (Powerset):
  - קב' $P$ = כל תתי הקב' של $\{x,y,z\}$ (נסמן $\mathcal{P}\{x,y,z\}$).
  - אופרטור היחס $\sqsubseteq$ הינו אופרטור תת-הקב' $\subseteq$.
  - $\{y\} \sqsubseteq \{x,y\}$, $\{x,y\} \not\sqsubseteq \{y\}$
  - $\emptyset \sqsubseteq \{x,y\} \sqsubseteq \{x,y,z\}$, $\{y,z\} \sqsubseteq \{y,z\}$
  - $\{x\} \not\sqsubseteq \{y,z\}$, $\{y,z\} \not\sqsubseteq \{x\}$ (לא כל האיברים בקב' ניתנים להשוואה)
* יחס חלוקה במס' טבעיים:
  - הקב' $P$ = כל המס' הטבעיים ($\mathbb{N}$).
  - אופרטור היחס $\sqsubseteq$ הינו "מחלק ב" $|$.
  - $7 \sqsubseteq 21$, $21 \sqsubseteq 7$
  - $17 \sqsubseteq 51 \sqsubseteq 255$, $19 \sqsubseteq 19$
  - $5 \not\sqsubseteq 13$, $13 \not\sqsubseteq 5$
* מחלקות בעץ ירושה:
  - הקב' $P$ = קב' טיפוסים (למשל במימוש קונקרטי).

- אופרטור היחס ⊑ הינו "הינו או שהוא צאצא של".

- המלך דוד ⋢ ישי, ישי ⊑ המלך דוד

- שלמה ⊑ שלמה, אברהם ⊑ יצחק ⊑ יעקוב

- עשיו ⋢ רות המואביה, רות המואביה ⋢ עשיו

* יכולת השגה (reachability) בגרף:

- הקבי ⊑ = ב הצמתים V ב-DAG $G=(V,E)$.

- אופרטור היחס ⊑ הינו "בר השגה מ-" (ב-0 או יותר צעדים).

- למה זה G לפיות מכוון וחסר מעגלים? ← אם הגרף עם מעגל $x \rightleftarrows y$ או שאינו מכוון $x - y$
אזי מתקיים: $x \sqsubseteq y$ וגם $y \sqsubseteq x$ אבל $x \neq y$

- למשל: עץ משפחה (של יעקב).

# גבול תחתון ועליון

10:55 PM Wednesday, October 7, 2020

## חסם עליון:

* בהינתן $S \subseteq P$
* $x \in P$ הינו חסם עליון של $S$ אם $\forall y \in S$, $y \sqsubseteq x$. (x לאו דווקא שייך ל-S)
* $x \in P$ הינו חסם עליון מינ' של $S$ אם:
  - $x$ חסם עליון של $S$, וגם
  - $x \sqsubseteq y$ לכל חסם עליון $y$ של $S$.
* $\sqcup$ - Join, חסם עליון מינ', lub, סופרימום, Sup.
  - $\sqcup S$ הינו חסם עליון מינ' של $S$.
  - $x \sqcup y = \sqcup\{x, y\}$
  - בד"כ נרשם כ-$\vee$.

## דוגמאות לחסם עליון:

* צאצאים בעץ ירושה:
  - הקב' P=כל הדמויות התנכיות.
  - {יעקב, ראובן, שמעון, לוי, יהודה, ..., יוסף} ← יעקב ו-יב בניו
  - ... וגם $x \sqsubseteq$ שרה וגם $x \sqsubseteq$ יעקב $\Leftarrow x \in S$
  - לכן יעקב הינו חסם עליון של S, שרה הינה חסם עליון של S,
    אברהם הינו חסם עליון של S, חווה הינה חסם עליון של S.
  - $\sqcup S$ = יעקב. ← יעקב הינו חסם עליון מינ' של S מכיוון
    אין חסם עליון b המקיים $b \sqsubsetneq$ יעקב.
  - ישי = {ישי, שלמה}$\sqcup$ = ישי $\sqcup$ שלמה.
  - פרץ = {עובד, אבשלום, שלמה}$\sqcup$ = עובד $\sqcup$ אבשלום $\sqcup$ שלמה
* חלוקה במס' טבעיים:
  - הקב' $P = \mathbb{N}$.
  - $S = \{1, 2, 3\}$
  - ... גם $x \sqsubseteq 100$ גם $x \sqsubseteq 6$ $\Leftarrow x \in S$.
  - לכן 6 חסם עליון של S, 18 חסם עליון של S, 24 חסם עליון
    של S, 120 הינו חסם עליון של S, ...
  - $\sqcup S = 6$. ← 6 הוא חסם עליון מינ' של S כי לכל חסם
    עליון b של S מתקיים $6 \sqsubseteq b$.

- $4\sqcup 12=\sqcup\{4,12\}=12$
- $3\sqcup 7=\sqcup\{3,7\}=21$

## חסם תחתון:

* בהינתן $S\subseteq P$.
* $x\in P$ הינו חסם תחתון של $S$ אם $\forall y\in S$ $x\sqsubseteq y$.
* $x\in P$ הינו חסם תחתון מקס' של $S$ אם:
  - $x$ חסם תחתון של $S$, וגם
  - $y\sqsubseteq x$ עבור כל החסמים התחתונים ב-$S$.
* $\sqcap$ - Meet, חסם תחתון מקס', גלב, אינפימום, inf.
  - $\sqcap S$ - חסם תחתון מקס' של $S$.
  - $x\sqcap y=\sqcap\{x,y\}$
  - בד"כ נרשם כ-$\wedge$.

## כיסוי:

* $x\sqsubset y$ אם $x\sqsubseteq y$ וגם $x\neq y$.
* $x$ מכוסה ע"י $y$ ($y$ מכסה את $x$) אם:
  - $x\sqsubset y$, וגם
  - לא קיים $z$ כך ש- $x\sqsubset z\sqsubset y$.
* קונספטואלית- $y$ מכסה את $x$ אם אין איברים בין $x$ לבין $y$.
* דוגמה:
  - $P=\mathbb{Z}$, $\sqsubseteq=\leq$
  - 5 מכסה את 4.
  - 5 אינו מכסה את 3.

# שריגים

10:55 PM Wednesday, October 7, 2020

## שריגים - Lattice:

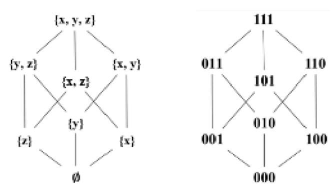
* אם קיימים $x \sqcup y$ וגם $x \sqcap y$ לכל $x, y \in P$ אזי $P$ שריג.
* אם קיימים $\sqcup S$ וגם $\sqcap S$ לכל $S \subseteq P$ אז $P$ שריג שלם.
* משפט: כל השריגים הסופיים שלמים.
* דוגמה לשריג שאיננו שלם:
  - $\mathbb{Z}$ מס' שלמים.
  - מקס' = $\sqcup$, מינ' = $\sqcap$
  - אבל, $\sqcup\mathbb{Z}$ ו- $\sqcap\mathbb{Z}$ לא קיימים ולכן לא שלם.
  - מצד שני $\mathbb{Z} \cup \{+\infty, -\infty\}$ כן שריג שלם.

## דוגמה:

* $P = \{000, 001, 010, 011, 100, 101, 110, 111\}$
* & = bitwise and → $x \sqsubseteq y \Leftrightarrow (x \& y) = x$

שריג בוליאני מנדרכי
נקרא גם hypercube

* דיאגרמת הסה Hasse:
  - אם $y$ מכסה את $x$:
  - ישנו קו בין $x$ ל-$y$.
  - $y$ נמצא מעל $x$.

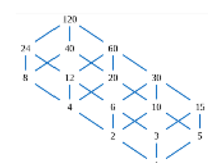



* הקבו' הסדורה חלקית איזומורפית לחזקת-קבו' של קבו' בת 3"

שרטוט קודם

## דוגמה נוספת:

* $P$ = קבו' כל המס' השלמים שמחלקים ב-120
* $\sqsubseteq$ = מתחלק ב.




## ראש וזנב (Top & Bottom):

* האיבר הגבוה ביותר ב-$P$ (אם קיים) נקרא top ($\top$).

האיבר הנמוך ביותר ב-$P$ (אם קיים) נקרא bottom ($\bot$).

$\top = \sqcup P \quad \bot = \sqcap P$

## שריג מכפלה:

* בהינתן שריג $L$ ושריג $Q$ ניתן להפוך את מכפלתם לשריג:

$(l_1,q_1) \sqsubseteq (l_2,q_2) \Leftrightarrow l_1 \sqsubseteq l_2$ וגם $q_1 \sqsubseteq q_2$

* עבור וקטורים של $L$, ניתן גם להגדיר אותם כשריג:

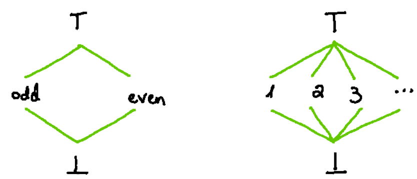
$\langle l_1,l_2,\ldots,l_k\rangle \sqsubseteq \langle t_1,t_2,\ldots,t_k\rangle \Leftrightarrow \forall i\in[1,k]\ l_i \sqsubseteq t_i$

# ניתוח סטטי ע"י שריגים

11:07 PM Wednesday, October 7, 2020

## שריגים של מאפייני התוכנית:

* מאפיינים שמעניינים אותנו ניתנים הרבה פעמים לסידור בסריג.
* למשל: סריגים של ערכים



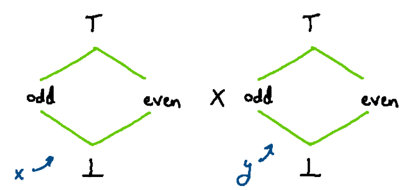

* כאשר הערך של כל משתנה הינו סריג, מצב התוכנית הינו סריג מכפלה של מצבי כל המשתנים.

## דוגמה:

```
x := 0;
y := 6;
while (x < 10) {
    x := x + 2;
    y := y + x;
}
assert (y is even);
```

x =
- T — יכול להיות זוגי או אי-זוגי
- odd — בהכרח אי-זוגי
- even — בהכרח זוגי
- ⊥ — לא משנה

← סריג מכפלה של 2 סריגים, אחד לכל משתנה




* סריג של פרדיקטים:
* $\langle x \in \{\bot, even, odd, T\}, y \in \{\bot, even, odd, T\} \rangle$
* למשל: $\langle x=even, y=odd \rangle \sqsubseteq \langle x=T, y=odd \rangle \sqsubseteq \langle x=T, y=T \rangle$

## דוגמה נוספת:

* ערכי סריג לא חייבים להיות בקשר ישיר לערכי התוכנית.
* למשל: האם אובייקט יכול לברוח מפונק'?


```
f() {
  var a, b;
  a := new A()
  b := a.val;
  return b;
}
/* a does not escape f */
```

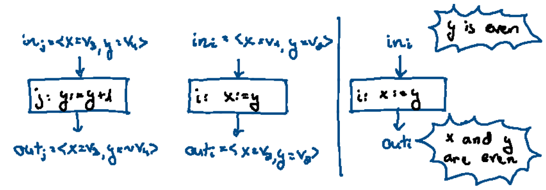
* אובייקט בורח מפונק' אם מצביע (בר השגה) אליו קיים גם לאחר היציאה מהפונק'.
* למשל: ב-C++ בפונקציות אין בריחה (לכן ניתן להוציא warning שחסר delete), אבל ב-C# a.val יכול להיות פונק' שמחזירה את this ולכן b מחזיק את המצביע ל-a וזו אינה שגיאה.

# חישוב פונ' המעבר:

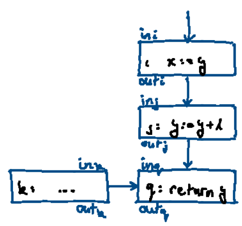
* עלינו לקבוע בצורה נוקשה את פונ' המעבר הספציפית לשריג.
  – לפעמים, קיים פשרה בין הדיוק של פונ' המעבר לבין כמה קל לחשב אותן.
* ניתן לבנות שריגים עבור עובדות שרירותיות על התוכנית.
  – עלינו לוודא שפונ' המעבר אלו "מתנהגות יפה" (נגדיר התנהגות טובה מאוחר יותר).
* משמעות פונ' המעבר: בהינתן קלט, המצב לפני ומה הפקודה שאנו מבצעים; מה המצב אחרי.

# מ-CFG למשוואות:

* אנו בעצם נבנה את ה-CFG למע' משוואות.
* לכל בלוק, נגדיר משתני מצב in ו-out: $out_i = T_i(in_i)$




$\sim odd = even$
$\sim even = odd$
$\sim \top = \top$
$\sim \bot = \bot$

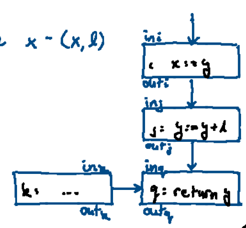



* אם i הקודם המיידי של j: $in_j = out_i$
* נשתמש ב-חיבור ($\sqcup$) אם מס' קשתות נכנסות לאותו בלוק: $in_q = out_j \sqcup out_k$

# הרחבה ל-Reaching Definitions:

* נגדיר את פונ' המעבר הבאה: $out_i = in_i \setminus (x,*) \cup \{(x,i)\}$

$(x,\ell)$ - x שהוגדר בפקודה $\ell$.

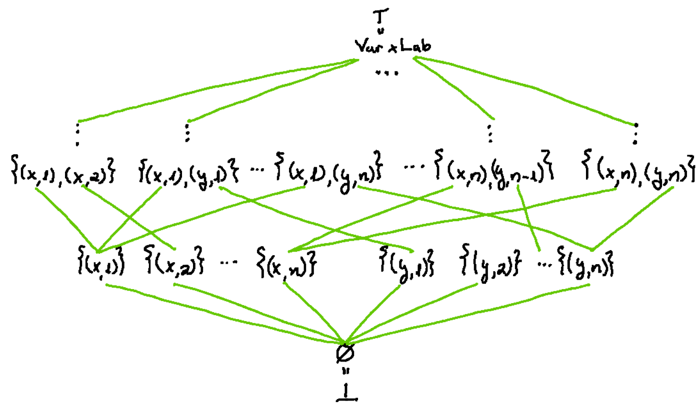



* כאשר:
  – x המשתנה אליו מקצים ב-i.
  – $(x,*) = \{(x,\ell) \mid \ell \in Lab\}$.
  – Lab קבו' כל ה-labels של הפקודות.
* כלומר, לוקחים את המצב הקיים ($in_i$), מורידים ממנו את כל ההגדרות הקודמות של x $(x,*)$ ומוסיפים את ההגדרה של x בשורה זו $(x,i)$.

# שריג ה-Reaching Definitions:

* עבור כל נק' בתוכנית, נחשב את קבו' ההגדרות המשתנים שעד' מושגות עבורם (משנים אותם).

$L = P(Var \times Lab)$, $\sqsubseteq = \subseteq$, $\sqcup = \cup$.



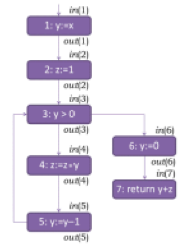

* זה נקרא שריג קב' החזקה (power set lattice).

# מערכת המשוואות עבור שריג Reachable Definitions

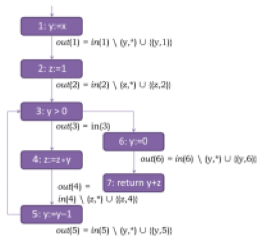
10:55 PM    Wednesday, October 7, 2020

קבוצת קלט/פלט:

```
1  y := x
2  z := 1
3  while (y > 0) {
4      z := z * y
5      y := y - 1
   }
6  y := 0
7  return y + z
```




פונ' המעבר:




$in(1) = \emptyset$
$in(2) = out(1)$
$in(3) = out(2) \cup out(5)$
$in(4) = out(3)$
$in(5) = out(4)$
$in(6) = out(3)$
$in(7) = out(6)$

מע' המשוואות:

$$
\begin{cases}
v_1 - in(1) = \emptyset \\
v_2 - in(2) = out(1) \\
v_3 - in(3) = out(2) \cup out(5) \\
v_4 - in(4) = out(3) \\
v_5 - in(5) = out(4) \\
v_6 - in(6) = out(3) \\
v_7 - in(7) = out(6) \\
v_8 - out(1) = in(1) \setminus (y,*) \cup \{(y,1)\} \\
v_9 - out(2) = in(2) \setminus (z,*) \cup \{(z,2)\} \\
v_{10} - out(3) = in(3) \\
v_{11} - out(4) = in(4) \setminus (z,*) \cup \{(z,4)\} \\
v_{12} - out(5) = in(5) \setminus (y,*) \cup \{(y,5)\} \\
v_{13} - out(6) = in(6) \setminus (y,*) \cup \{(y,6)\}
\end{cases}
$$

$F(v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}) =$
$= \langle \emptyset, v_8, v_9 \cup v_{12}, v_{10}, v_{11}, v_{10}, v_{13}, v_1 \setminus (y,*) \cup \{(y,1)\}, v_2 \setminus (z,*) \cup \{(z,2)\},$
$v_3, v_4 \setminus (z,*) \cup \{(z,4)\}, v_5 \setminus (y,*) \cup \{(y,5)\}, v_6 \setminus (y,*) \cup \{(y,6)\} \rangle$

* נשים לב שמע' כזו שהינה מערכת בגודל n.
* משוואות שלו משתנים שונ' מאז 13 משתנים $(in(1 \ldots 7), out(1 \ldots 6))$
* כל משתנה מקבל ערך מהסריג $L = \mathcal{P}(Var \times Lab)$.

$F : (\mathcal{P}(Var \times Lab))^{13} \to (\mathcal{P}(Var \times Lab))^{13}$

* פתרון $\vec{v}$ מקיים $F(\vec{v}) = \vec{v}$

# פתרון המשוואות:

* **בעיית נק' השבת:** בהינתן פונ' $F: L \to L$ נמצא $x \in L$ כך ש- $x = F(x)$.

* עם פונ' מעבר בדר"כ קיים יותר מפתרון אחד.

  - ספציפית כאשר בקוד יש לולאות.

  - נרצה את הפתרון **המדויק ביותר**. המצומצם ביותר, least upper bound

  - פתרון בנאלי: $F(\top) = \top$

# משפט Knaster-Tarski:

* **פונ' משמרת סדר (מונוטונית):** $x \sqsubseteq y \Rightarrow F(x) \sqsubseteq F(y)$

* יהי L סריג שלם ו- $F: L \to L$ פונ' מונוטונית. אזי, קב' נק' השבת של F הינה גם סריג שלם.

* **הגדרה:** **נק' השבת המינימלית ביותר $x_\perp$** הינה נק' שבת ($F(x_\perp) = x_\perp$) כך שלכל x אשר $F(x) = x$ אז $x_\perp \sqsubseteq x$.

# משפט נק' השבת של Kleene:

* בהינתן פונ' משמרת סדר (מונוטונית): $x \sqsubseteq y \Rightarrow F(x) \sqsubseteq F(y)$

* ובהינתן נק' שבת מינימלית ביותר $x_\perp$

  - בפרט $F(x_\perp) \sqsubseteq x_\perp$

* נבנה **סדרת** $x_0, x_1, \dots$ לפי: $x_0 = \perp$ , $x_{i+1} = F(x_i)$

  - באינדוקציה: $x_i \sqsubseteq x_\perp$

    בסיס: $x_0 = \perp \sqsubseteq x_\perp$ ($\perp \sqsubseteq x$ מתקיים $x$ לכל)

    צעד: $x_{i-1} \sqsubseteq x_\perp \Rightarrow F(x_{i-1}) \sqsubseteq F(x_\perp) \Rightarrow x_i \sqsubseteq x_\perp$

    הנחת האינדוקציה    F מונוטונית    הגדרת $x_\perp$, F

  - $x_i \sqsubseteq x_{i+1}$

    בסיס: $x_0 = \perp \sqsubseteq x_1$ ($\perp \sqsubseteq x$ מתקיים $x$ לכל)

    צעד: $x_{i-1} \sqsubseteq x_i \Rightarrow F(x_{i-1}) \sqsubseteq F(x_i) \Rightarrow x_i \sqsubseteq x_{i+1}$

    הנחת האינדוקציה    F מונוטונית    הגדרת F

  - אם קיים i עבורו $x_i = x_{i+1}$ ⇐ $x_i$ נק' שבת ⇐ $x_\perp \sqsubseteq x_i \sqsubseteq x_\perp$ ⇐ $x_i = x_\perp$

* אותו טריק עובד עבור חישוב **נק' שבת גבוהה ביותר $x_\top$**.

  - במקרה זה נתחיל עם $x_0 = \top$.

# שרשראות:

* קב' $S \subseteq L$ הינה **שרשרת** אם: $x \sqsubseteq y$ או $y \sqsubseteq x$ $\forall x,y \in S$.

* ל- L **אין שרשראות אינסופיות** אם כל שרשרת ב-L היא סופית.

* במקרה זה, **מובטח** שנמצא את נק' השבת המינ' במס' סופי של צעדים.

## נחזור לדוגמה:

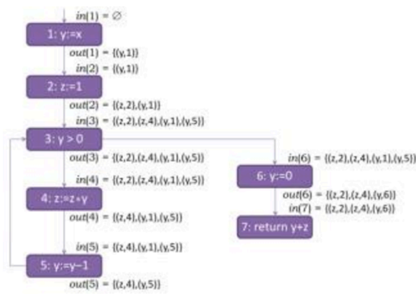
* הפונ': $F:(\mathcal{P}(Var\times Lab))^{13}\to(\mathcal{P}(Var\times Lab))^{13}$

* המשוואות: $in(1)=\emptyset$ $in(2)=out(1)$ $in(3)=out(2)\cup out(5)$ $in(4)=out(3)$
$in(5)=out(4)$ $in(6)=out(3)$ $in(7)=out(6)$
$out(1)=in(1)\setminus(y,*)\cup\{(y,1)\}$ $out(2)=in(2)\setminus(z,*)\cup\{(z,2)\}$
$out(3)=in(3)$ $out(4)=in(4)\setminus(z,*)\cup\{(z,4)\}$
$out(5)=in(5)\setminus(y,*)\cup\{(y,5)\}$ $out(6)=in(6)\setminus(y,*)\cup\{(y,6)\}$

| | ⊥ | F(⊥) | F(F(⊥)) | F(F(F(⊥))) | F(F(F(F(⊥)))) | F(F(F(F(F(⊥))))) | F(F(F(F(F(⊥))))) | $F^6(\perp)$ | $F^7(\perp)$ | $F^8(\perp)$ |
|---|---|---|---|---|---|---|---|---|---|---|
| in(1) | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ | ∅ |
| in(2) | ∅ | ∅ | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} |
| in(3) | ∅ | ∅ | {(z,2),(y,5)} | {(z,2),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} |
| in(4) | ∅ | ∅ | ∅ | ∅ | {(z,2),(y,5)} | {(z,2),(y,5)} | {(z,2),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} |
| in(5) | ∅ | ∅ | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4),(y,1),(y,5)} | {(z,4),(y,1),(y,5)} |
| in(6) | ∅ | ∅ | ∅ | ∅ | {(z,2),(y,5)} | {(z,2),(y,5)} | {(z,2),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} |
| in(7) | ∅ | ∅ | {(y,6)} | {(y,6)} | {(y,6)} | {(y,6)} | {(y,6)} | {(y,6)} | {(y,6)} | {(y,6)} |
| out(1) | ∅ | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} | {(y,1)} |
| out(2) | ∅ | {(z,2)} | {(z,2)} | {(z,2),(y,1)} | {(z,2),(y,1)} | {(z,2),(y,1)} | {(z,2),(y,1)} | {(z,2),(y,1)} | {(z,2),(y,1)} | {(z,2),(y,1)} |
| out(3) | ∅ | ∅ | ∅ | {(z,2),(y,5)} | {(z,2),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} | {(z,2),(z,4),(y,1),(y,5)} |
| out(4) | ∅ | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4)} | {(z,4),(y,1),(y,5)} | {(z,4),(y,1),(y,5)} | {(z,4),(y,1),(y,5)} |
| out(5) | ∅ | {(y,5)} | {(y,5)} | {(z,4),(y,5)} | {(z,4),(y,5)} | {(z,4),(y,5)} | {(z,4),(y,5)} | {(z,4),(y,5)} | {(z,4),(y,5)} | {(z,4),(y,5)} |
| out(6) | ∅ | {(y,6)} | {(y,6)} | {(y,6)} | {(y,6)} | {(z,2),(y,6)} | {(z,2),(y,6)} | {(z,2),(y,6)} | {(z,2),(y,6)} | {(z,2),(y,6)} |

...

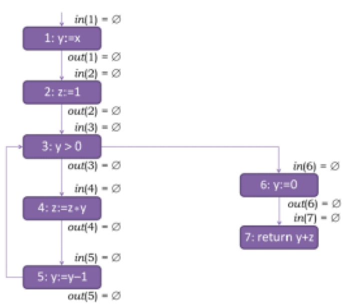
## תוצאת נק' השבת המינ':




## איטרציות כאוטיות:

* בד"כ הוקטורים של המשתנים מאוד גדולים.
* לכן כדי להימנע מחישוב חוזר של ערכים של משתנים:
  - נשמור רשימת עבודה (work list) של צמתי ה-CFG שיש לעדכן.
  - נבחר צומת יחיד בכל פעם.
  - נעדכן את out(u) מ-in(u).
  - אם out(u) השתנה, נעדכן את in(v) לכל השכנים v של u ונוסיף את v לרשימת העבודה.

## איטרציות כאוטיות - דוגמה:




* בהתחלה: work-list = {1}
* עדכונים:
$out(1)=in(1)\setminus(y,*)\cup\{(y,1)\}$
$out(2)=in(2)\setminus(z,*)\cup\{(z,2)\}$
$out(3)=in(3)$
$out(4)=in(4)\setminus(z,*)\cup\{(z,4)\}$
$out(5)=in(5)\setminus(y,*)\cup\{(y,5)\}$
$out(6)=in(6)\setminus(y,*)\cup\{(y,6)\}$

# שימוש במידע של Reaching Definitions:

* נזכיר: זהו קירוב-יתר. ייתכן והגדרה לא תגיע ל-use.

* ייתכן ונאמר, אך תמיד לחומרה. ← בצד שמרני

- ייתכן ונאמר שהגדרה יכולה להגיע לנק' בזיכרון כאשר אינה מגיעה.

- לעולם לא נפספס הגדרה שיכולה להגיע לנק' מסוימת.

* דוגמאות לשימוש:

- איתור שימוש לפני הגדרה אפשרי.

- שימושי לדיבאגינג ב- IDE.

- constant folding פשוט מאוד.

# ניתוח ביטויים זמינים

10:55 PM Wednesday, October 7, 2020

## אנליזת ביטויים זמינים:

```
1  x = a + b
2  y = a * b
3  while (y > a + b) {
4    a = a + 1
5    x = a + b
   }
```

ביטוי זה זמין
בשורה 3

* לכל נקודה בתוכנית, נמצא אילו ביטויים הם חייבים כבר להיות מחושבים,

  ולא שונו מאוחר יותר, בכל המסלולים המובילים לנקודה זו.

## נוטציות חשובות:

* מחלקות של ביטויים:
  - AExp - ביטויים אריתמטיים.
  - BExp - ביטויים בוליאניים.
* $FV : (BExp \cup AExp) \to \mathcal{P}(Var)$ - המשתנים שמופיעים בביטוי.
* AExp(a) - כל תתי-הביטויים (שאינם אטומיים) האריתמטיים של ביטוי אריתמטי a.
  - AExp(b) עבור ביטוי בוליאני b.

## אנליזת ביטויים זמינים:

* מרחב המצבים:
  - $L = \mathcal{P}(AExp)$, $\sqsubseteq = \supseteq$, $\sqcup = \cap$
  - $in, out : Lab \to L$

ממפים תווית פקודה לקבוצת ביטויים אריתמטיים שזמינים ב-(אחרי, לפני) הפקודה.

* משוואות בזרימת המידע:
  - משוואות זרימה: כיצד לבצע join על זרמים dataflow נכנסים.
  - משוואות אפקטים: בהינתן קלט של קבוצת ביטויים $in(i)$ מה האפקט של הפקודה ב-i?

* $in(\ell)$:

| | |
|---|---|
| אם $\ell$ תווית ההתחלה | $\emptyset$ |
| אחרת | $\cap\{out(\ell') \mid \ell' \in pred(\ell)\}$ |

* $out(\ell)$:

כל הביטויים האריתמטיים ש-x מופיע בהם

כל תתי-הביטויים של a שx לא מופיע בהם

| | |
|---|---|
| x:=a | $in(\ell) \setminus \{k \in AExp \mid x \in FV(k)\} \cup \{g \in AExp(a) \mid x \notin FV(g)\}$ |
| Skip | $in(\ell)$ |
| Cond | $in(\ell) \cup AExp(cond)$ |

## פונ' מעבר:

```
1  x = a + b
2  y = a * b
3  while (y > a + b) {
4      a = a + 1
5      x = a + b
   }
```

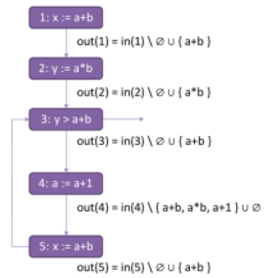
$in(1) = \emptyset$

$in(2) = out(1)$

$in(3) = out(2) \cap out(5)$

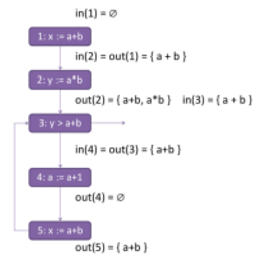
$in(4) = out(3)$

$in(5) = out(4)$




## פתרון:

* החישוב נעשה ע"י מציאת נק' השבת.




## Kill / Gen:

| | |
|---|---|
| $in(\ell) \setminus \underbrace{\{k \in AExp \mid x \in FV(k)\}}_{kill} \cup \underbrace{\{g \in AExp(a) \mid x \notin FV(g)\}}_{gen}$ | $x := a$ |
| $in(\ell)$ | Skip |
| $in(\ell) \cup \underbrace{AExp(cond)}_{gen}$ | Cond |

| gen | kill | |
|---|---|---|
| $\{g \in AExp(a) \mid x \notin FV(g)\}$ | $\{k \in AExp \mid x \in FV(k)\}$ | $x := a$ |
| $\emptyset$ | $\emptyset$ | skip |
| $AExp(cond)$ | $\emptyset$ | cond |

* $out(\ell) = in(\ell) \setminus kill(B^\ell) \cup gen(B^\ell)$, $B^\ell$ - פקודה או בלוק בקודקוד $\ell$.

## הגדרה ל- Reaching Definitions:

* $out(\ell)$:

| | |
|---|---|
| $in(\ell) \setminus \{(x,i) \mid i \in Lab\} \cup \{(x,\ell)\}$ | $x := a$ |
| $in(\ell)$ | skip |
| $in(\ell)$ | cond |

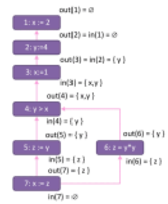
| gen | kill | |
|---|---|---|
| $\{(x,\ell)\}$ | $\{(x,i) \mid i \in Lab\}$ | $x := a$ |
| $\emptyset$ | $\emptyset$ | skip |
| $\emptyset$ | $\emptyset$ | cond |

* לכל נק' בתוכנית, אילו השמות יגיעו ועדיין לא שונו,

כאשר ריצת התוכנית מגיעה לנק' זו לפי מסלול כלשהו.

# משתנים חיים:

```
1: x := 2;
2: y := 4;
3: x := 1;
4: if y > x
5:     then z := y
6:     else z := y * y;
7: x := z
```




| gen | kill | |
|---|---|---|
| $\{FV(a)\}$ | $\{x\}$ | x:a |
| $\emptyset$ | $\emptyset$ | skip |
| $FV(cond)$ | $\emptyset$ | cond |

* עוברים מהסוף להתחלה. לכן in של בלוק זה בעצם ה-out שלו.

* לכל נק' בתוכנית, אילו משתנים יתכן והם חיים (כלומר יש בהם שימוש עתידי לפני השמה מחדש, עם מסלול כלשהו) ביצוע מנק' זו.

# Monotone Framework

10:55 PM Wednesday, October 7, 2020

## :Monotone Framework

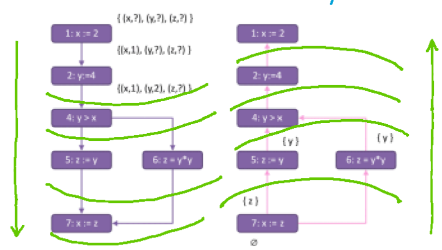
* תבנית שחוזרת על עצמה באנליזות שונות.

$$in(\ell) = \begin{cases} \text{Initial} & \ell \in \text{Entry labels} \\ \bigsqcup \{ out(\ell') \mid (\ell', \ell) \in \text{CFG edges} \} & \text{otherwise} \end{cases}$$

$$out(\ell) = f_\ell(in(\ell))$$

* $\cap$ או $\cup$ = $\sqcup$
* כיוון ההתקדמות על קשתות ה-CFG הינו קדימה או אחורה.
* Entry labels הינם קשתות ההתחלה של התוכנית או קשתות סיום של התוכנית (כאשר הולכים אחורה).
* Initial הינו מצב התחלתי (או סופי כשהולכים אחורה).
* $f_\ell$ הינה פונ' המעבר שמתאימה לבלוק $B^\ell$.

## עבודה קדימה לעומת אחורית:




## דוגמה - Reaching Definitions:

* $L = \mathcal{P}(Var \times Lab)$ סדורה חלקית ע"י $\sqsubseteq = \subseteq$.
* $\sqcup$ הינו $\cup$.
* $L$ ללא שרשראות אינסופיות כי $Var \times Lab$ סופי (עבור תוכנית נתונה).

## דוגמה - ביטויים זמינים:

* $L = \mathcal{P}(AExp)$ סדורה חלקית ע"י $\sqsubseteq = \supseteq$.
* $\sqcup$ הינו $\cap$.
* $L$ ללא שרשראות אינסופיות כי $AExp$ סופי (עבור תוכנית נתונה).