

תורת הקומפילציה

Data Flow Analysis

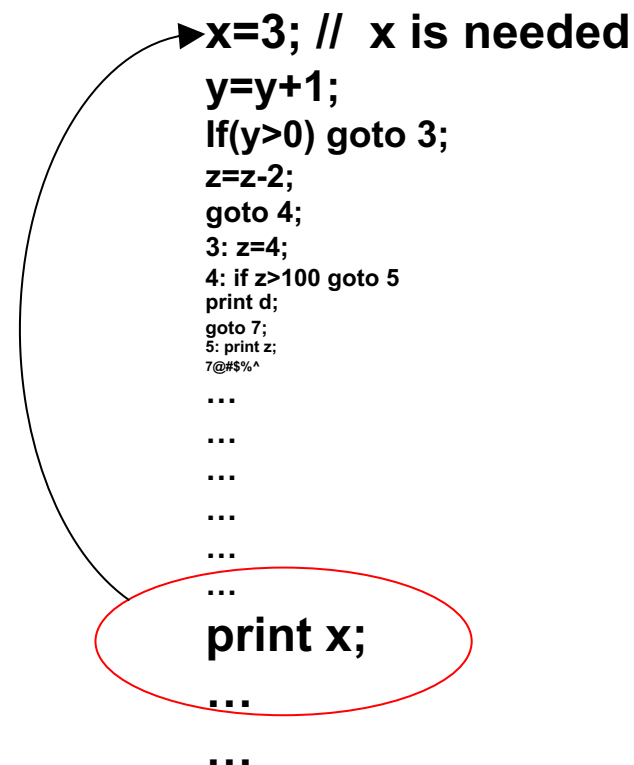
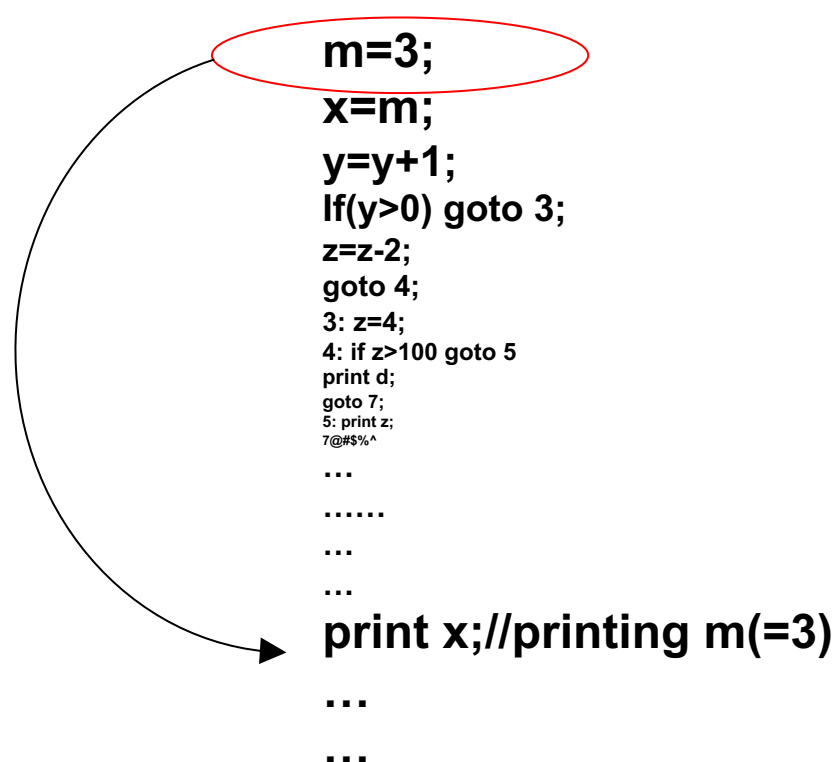
עודכן: חורף 2022/2023

מוטיבציה

- ראיתם בשיעור אנליזות DFA
 - reaching definitions
 - available expressions
- נתעמק ונגדיר פורמלית סוג ספציפי של אנליזות מסוג זה:
gen/kill analysis
- לאופטימיזציות רבות (שנלמד בהמשך), נדרש מידע על מצבים בתכנית כדי לזהות את כל הפקודות / משתנים / ביטויים אשר מקיימים תכונה מסוימת בנקודה זו.

מוטיבציה – דוגמה

ניתוח aliasing	ניתוח חיות	אנליזת DFA
Copy Propagation	Dead Code Elimination	אופטימיזציה



Control Flow Graph (CFG)

- גרף מכוון המייצג את זרימת בקרת התוכנית

– אחד הייצוגים הנפוצים של תכנית.

- צמתים: בלוקים בסיסיים

– קיים בלוק כניסה יחיד

- קשתות: יש קשת מבלוק A לבלוק B אם:

– A מסתיים בקפיצה (אולי מותנית) ל-B, או

– A מסתיים בקפיצה מותנית או ללא קפיצה ו-B מופיע אחרי A בקוד המקורי.

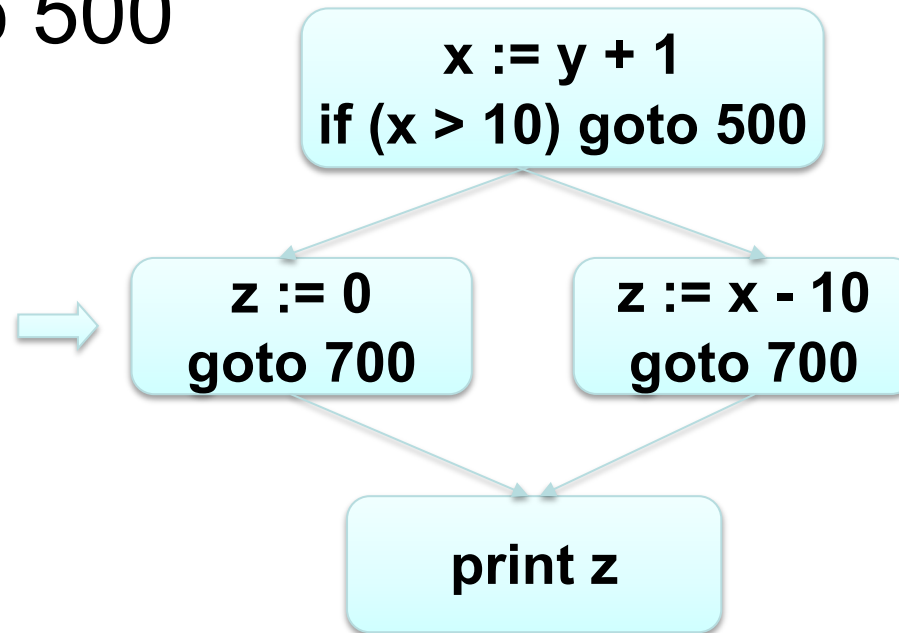
כלומר קיים ביצוע (אולי) שעובר מהשורה האחרונה בבלוק A לשורה הראשונה בבלוק B.

ראינו בהרצאה, בתרגול הבא
נראה אלגוריתם למציאת
בלוקים בסיסיים

כל מתודה בתכנית תיוצג ע"י CFG כאשר הבלוק ההתחלתי יהיה הבלוק המייצג את נקודת הכניסה של המתודה

CFG

100: $x := y + 1$
200: if ($x > 10$) goto 500
300: $z := 0$
400: goto 700
500: $z := x - 10$
600: goto 700
700: print z



אנליזה סטטית: תזכורת

- דומיין N שאיבר בו מתאר מצב בתכנית (פשוט יותר מהמצבים האמיתיים בתכנית)
- פונקציה f שעוברת ממצב למצב עבור כל משפט בתכנית
- ופעולות שעוזרות לנו לשלב בין מסלולי ריצה שונים בתכנית

דוגמה - Copy Propagation

- הרעיון: אחרי העתקה $x := y$, נרצה להחליף את x ב- y במקומות בהם משתמשים ב- x
- דוגמה:

$x := y$

$c := x + d$

$e := c * x$



$x := y$

$c := y + d$

$e := c * y$

- מה הרווחנו?
- אם x לא יהיה חי אחרי הפקודה $x := y$, נוכל למחוק את הפקודה

בעיות DFA – הגדרות

- **קלט**: CFG, כאשר
 - כל צומת הוא בלוק בסיסי, או
 - כל צומת הוא פקודה יחידה
- הופך את הניתוח ברמה של צומת לפשוט יותר
- מגדיל סיבוכיות זמן ומקום

בעיות DFA – הגדרות

- **קלט:** CFG, כאשר

- כל צומת הוא בלוק בסיסי, או

- כל צומת הוא פקודה יחידה

- הופך את הניתוח ברמה של צומת לפשוט יותר

- מגדיל סיבוכיות זמן ומקום

- **פלט:** ערכים $in(B)$ ו- $out(B)$ לכל צומת B ב-CFG

- $in(B)$ - מתייחסת לנק' בה האנליזה נכנסת ל-B

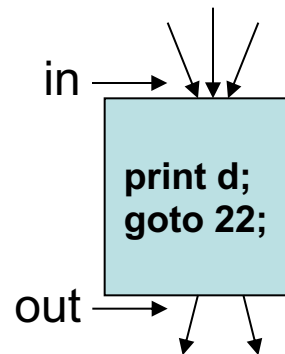
- $out(B)$ - מתייחסת לנק' בה האנליזה יוצאת מ-B

- בינתיים: לפני הפקודה הראשונה ואחרי האחרונה

- **מה הם הערכים?**

- איברים בדומיין של האנליזה שנריץ. מקרה פרטי מעניין הוא

שהם קבוצות.



Copy Propagation - דומיין

- עולם הבעיה הוא הצמדים (x,y) המציינים כי בוצעה השמה (ועדיין ניתן להשתמש בה) מ- y ל- x

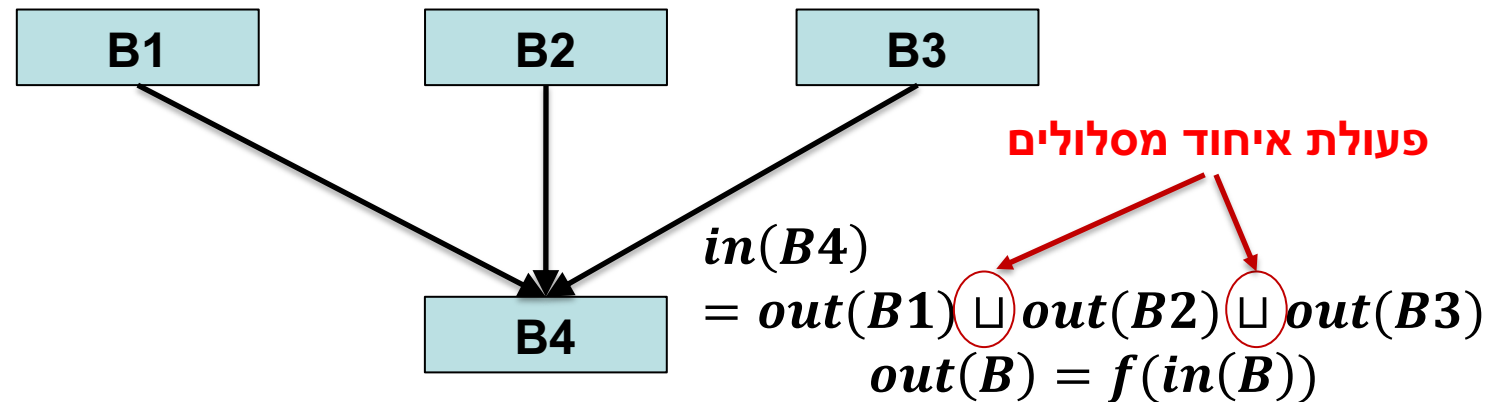
$$N = \{(x, y) | x \text{ and } y \text{ are vars}\}$$

הדומיין הוא 2^N (רושמים גם $\wp(N)$), כל תתי הקבוצות של N . איבר בדומיין מציין כי בנקודה מסוימת בתכנית אוסף הזוגות בתת הקבוצה עברו השמה.

$$in(B) \in 2^N, out(B) \in 2^N$$

מסלולים שונים בתכנית

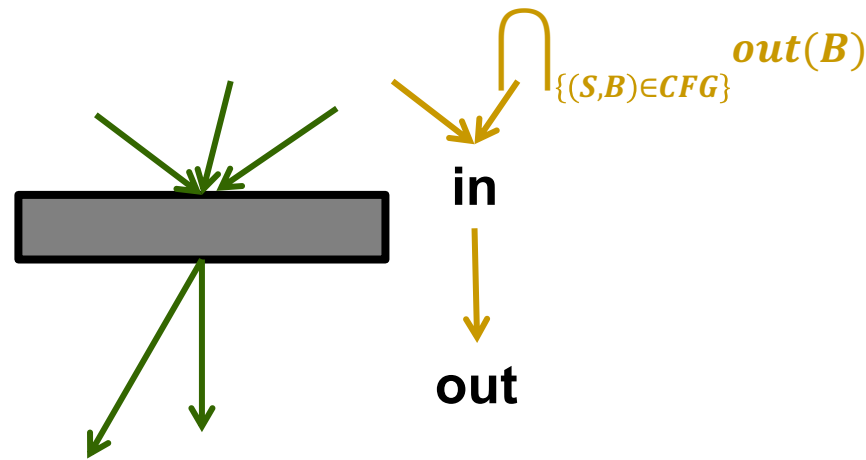
- כאשר מספר מסלולים בתכנית מתאחדים נרצה לשמר מידע מכל המסלולים



- צריך לבחור את המימוש של \sqcup , פעולת האיחוד בסריג

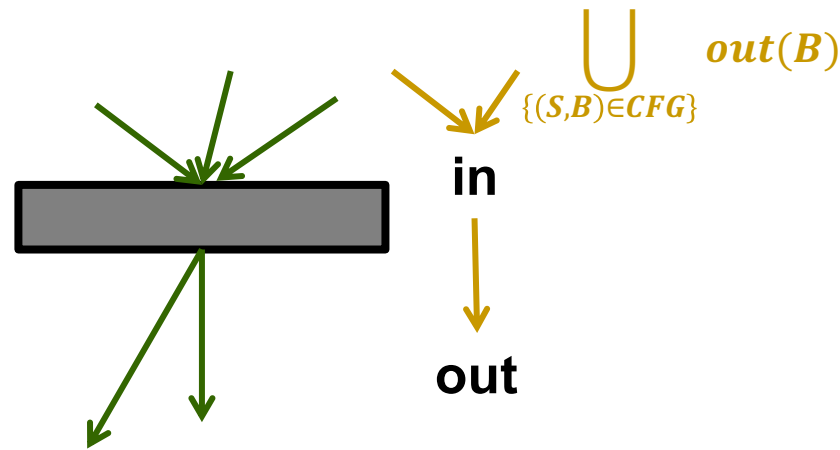
אופציה 1: חיתוך קבוצות

- אם הדרישה צריכה להתקיים על כל המסלולים עד לנקודה הנוכחית
- בדומיין powerset, נרצה להשתמש בחיתוך, כלומר: $\sqcap = \cap$
- לכן נגדיר את יחס הסדר החלקי להיות $\sqsubseteq = \supseteq$ (הכלה בכיוון ההפוך)



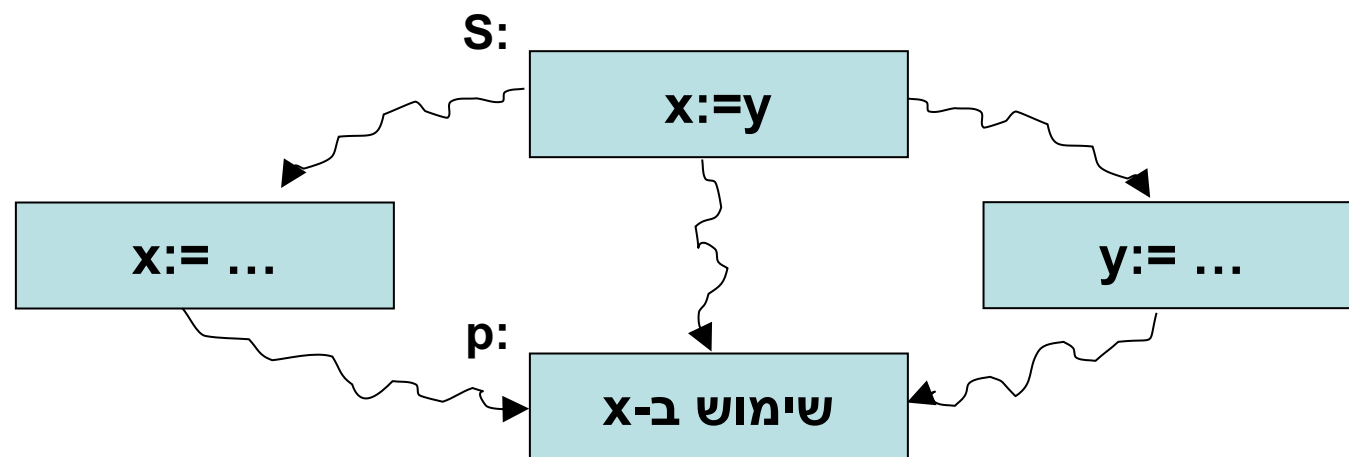
אופציה 2: איחוד קבוצות

- אם הדרישה צריכה להתקיים על מסלול כלשהו עד לנקודה הנוכחית
- בדומיין powerset, נרצה להשתמש באיחוד, כלומר: $\sqcup = \cup$
- לכן נגדיר את יחס הסדר החלקי להיות $\sqsubseteq = \subseteq$



מתי אסור לבצע את ההחלפה של x ב- y ?

- נניח כי כל צומת ב-CFG מכיל פקודה יחידה



1. יש הגדרה של x שאינה ב- s שמגיעה ל- p . • x לא בהכרח שווה ל- y .
2. יש מסלול בין s ל- p שעליו y מוגדר מחדש. • y השתנה.

המצב שמכיל את כל המסלולים (לבעיה זו): המצב שמכיל רק את מה שמגיע מכולם

Copy Propagation - המשך

- מהו סוג הבעיה? (איך נגדיר את \sqcup)
– חיתוך קבוצות, כי מספיק שיהיה מסלול אחד "רע" כדי שלא נוכל לבצע החלפה
- כל המסלולים צריכים לקיים את הדרישות
- משוואות הזרימה:

$$\text{out}(B) = f_B(\text{in}(B))$$

$$\text{in}(B) = \bigcap_{(S,B) \in \text{CFG}} \text{out}(S)$$

הערת צד: may/must

- דרישה על כל המסלולים מכונה בעיית must, ודרישה על מסלול כלשהו מכונה בעיית may.
- בסמסטרים קודמים היה דגש על זיהוי הבעיה כאחת מהאופציות האלו, אבל בדומיינים שאינם powerset ההבדלה הזו פחות שימושית ואף מבלבלת.
- במקום זאת, נתמקד בהגדרת הסריג, כלומר יחס הסדר החלקי ופעולת ה-join.

הפונקציה f בבעיות gen/kill

$$f(B) = (in(B) \setminus kill(B)) \cup gen(B)$$

- Gen – איברים חדשים שנודעו לנו מתוך הבלוק
- Kill – איברים שאינם רלוונטיים יותר בגלל מה שמבוצע בבלוק
- כל עוד gen ו- $kill$ מסתמכים רק על הבלוק עצמו (ומתקיימים שאר התנאים של ה-monotone framework), האנליזה תהיה מונוטונית, ולכן מובטח שתתכנס.

Copy Propagation - המשך

• מהו $?gen(B)$

– הזוג (x,y) אם מופיעה ב- B פקודות העתקה $x:=y$
ואחריה אין אף כתיבה ל- x או ל- y

• מייד אחרי הפקודה $x:=y$ מותר להחליף את x ב- y

$$gen(B) = \{(x,y) \mid x := y \in B\} -$$

• מהו $?kill(B)$

– הזוג (x,y) אם:

• B מכיל כתיבה (אחרת) ל- x (מקרה 1)

• B מכיל כתיבה ל- y (מקרה 2)

$$kill(B) = \{(x,y) \mid x := ? \in B\} \cup \{(x,y) \mid y := ? \in B\} -$$

הגדרה אלטרנטיבית לפונקציות

- בשקף הקודם ראינו הגדרה מסורבלת מעט ל-kill:
$$kill(B) = \{(x,y) \mid x := ? \in B\} \cup \{(x,y) \mid y := ? \in B\}$$
- זה עדיין סביר, אבל אם נרצה לכלול יותר תבניות של פקודות בשפת הביניים, נקבל ביטוי ארוך ולא קריא. לצורך קריאות ניתן להגדיר את פונקציית המעבר באמצעות טבלה, בדומה לזו שראינו בהרצאה:

Statement	$kill(B)$
$x := expr$	$\{(x,y) \mid y \in Vars\} \cup \{(y,x) \mid y \in Vars\}$

אתחול

מאתחלים את $\text{in}(B)$ או $\text{out}(B)$ בצורה שמרנית:
נשתמש ב- \perp של הסריג

- **נימוק לאופציה ראשונה (איחוד) –**

- מה שנכנס באתחול עלול להישאר בתוצאה סופית
- אומדן שמרני – \emptyset – קבוצה ריקה (כי רק מוסיפים דברים)

- **נימוק לאופציה שנייה (חיתוך) –**

- מה שלא נכנס באתחול עלול לא להופיע בתוצאה סופית
- אומדן שמרני – N – כל העובדות האפשריות בתכנית (כי רק מורידים דברים)

לעיתים הבלוק הראשון או האחרון בתכנית צריכים טיפול מיוחד

Copy Propagation - המשך

- איך נאתחל את הקבוצות?

ה- \perp בסריג הוא N , ולכן:

– לכל בלוק פרט ל- B_0 :

$$\text{in}(B) = \text{out}(B) = \perp = N$$

– עבור B_0 :

$$\text{in}(B_0) = \emptyset$$

$$\text{out}(B_0) = \text{gen}(B_0) = (f_{B_0}(\emptyset))$$

– מספיק לאתחל רק אחת מבין הקבוצות in ו-out

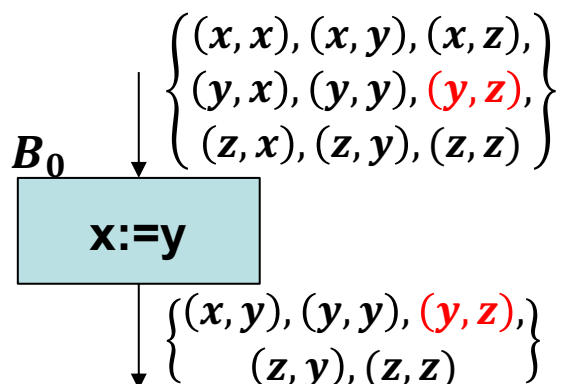
Copy Propagation - המשך

• מדוע יש טיפול מיוחד בבלוק הראשון?

– ל- B_0 אין אבות ב-CFG

• כל דבר שנכניס ל- $\text{in/out}(B_0)$ באיתחול, ישאר שם עד הסוף ויחלחל לבנים

– דוג': תכנית עם $\text{Vars} = \{x, y, z\}$
נבחן אתחול עם N :



1. ההעתקה הלא קיימת (y, z) תישאר אחרי הבלוק ותחלחל לבנים

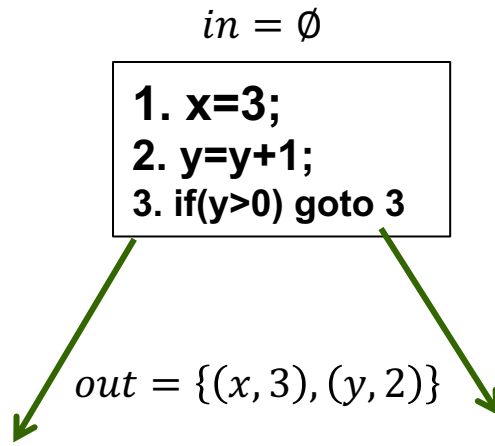
2. האנליזה לא נכונה: דברים לא נכונים לגבי החלפות אפשריות ב- B_0

– גם (1) וגם (2) לא יתוקנו במהלך הריצה

חישוב משוואות הזרימה

- המשוואות עשיות ליצור תלויות מעגליות
 - למשל, במקרה של לולאה
 - ראינו מקרים כאלה בעבר (first, follow)
 - כל עוד יש שינויים, מחשבים in ו-out מחדש לכל הבלוקים בתכנית לפי משוואות הזרימה
- לפי סדר כלשהו
- החישוב יתבצע בצורה איטרטיבית עד לנקודת שבת על ערכי in/out בגרף

כיוון זרימה



- צורת החישוב - אחת מהשתיים:

- סריקה קדמית - המידע "זורם" קדימה

- Copy propagation

- סריקה אחורית - המידע "זורם" אחורה

- למשל, בניתוח חיות

כיוון
זרימת/ריצת
התוכנית



כיוון זרימת
המידע

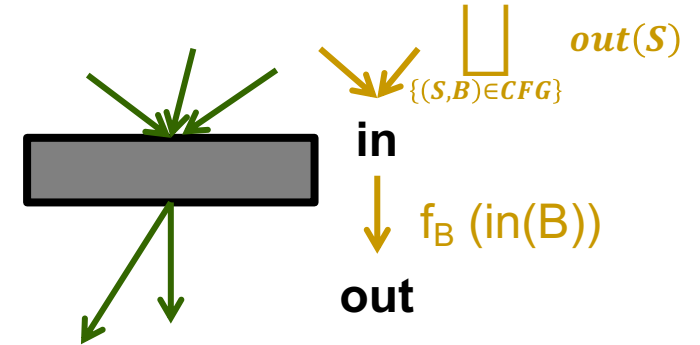


משוואות זרימה

$$\text{out}(B) = f_B (\text{in}(B))$$

$$\text{in}(B) = \bigsqcup_{(S,B) \in \text{CFG}} \text{out}(S)$$

סריקה קדמית



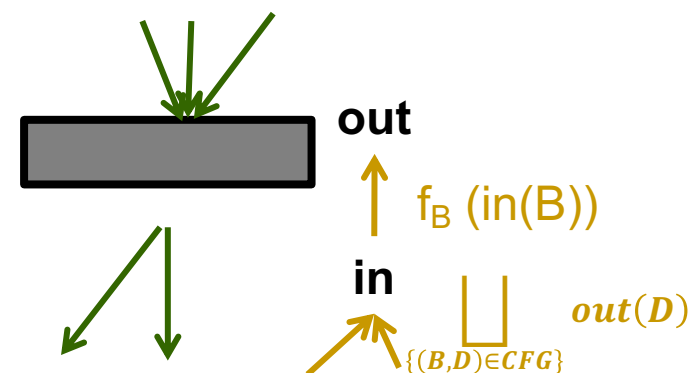
- מה שיוצא מ-B מתקבל מהפעלת f_B על מה שנכנס אליו
- מה שנכנס ל-B מתקבל מהאבות של B בגרף CFG – בהמשך נראה איך (חיתוך או איחוד)

משוואות זרימה

סריקה אחורית

$$\text{out}(B) = f_B (\text{in}(B))$$

$$\text{in}(B) = \bigsqcup_{(B,D) \in \text{CFG}} \text{out}(D)$$



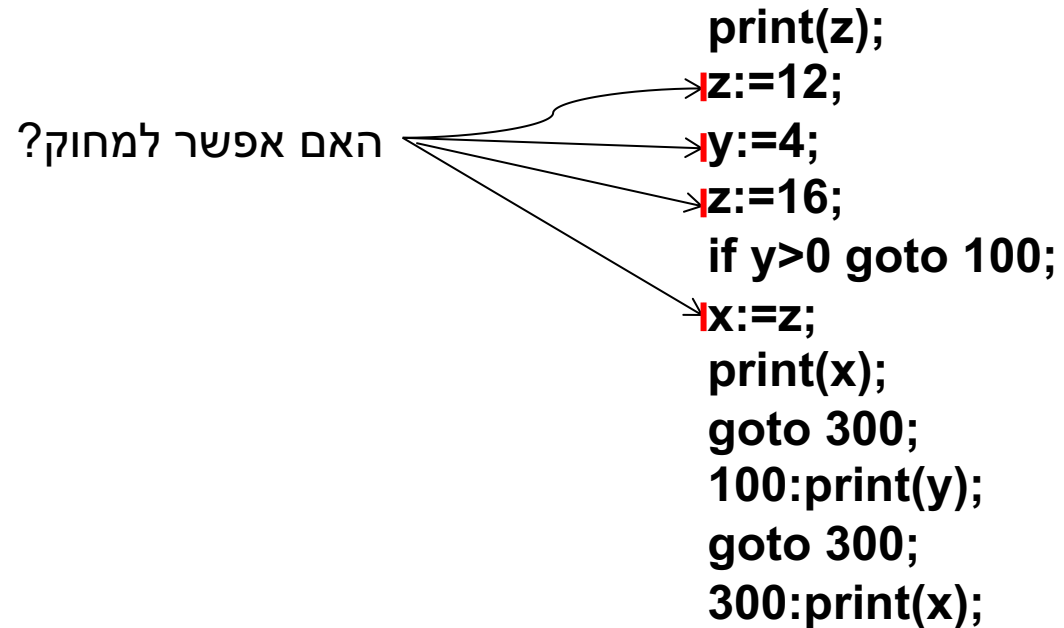
- בדיוק כמו בסריקה קדמית, רק ש-in ו-out של הבלוק מתחלפים

– הפעם, מסתכלים על הבנים של B ב-CFG

דוגמה 2: ניתוח חיות

- הגדרה: נאמר כי משתנה x הוא חי בנקודה מסויימת של תכנית אם ייתכן שיש בו שימוש (קריאת ערך) לפני שכותבים אליו מחדש.
- אם משתנה x אינו חי מייד אחרי פקודת השמה $x := \dots$, פקודה זו מיותרת.
- הצורך - אופטימיזציית Dead Code Elimination:
 - מוחקת פקודות השמה שאחריהן מובטח שאין שימוש למשתנה אליו כתבו.

אופטימיזציית Dead Code Elimination



שימוש ב DFA לניתוח חיות

- מטרת החישוב: בכל נקודה q בתכנית, נרצה למצוא את כל המשתנים החיים בה.

– כלומר: $N = Vars$

- עבור כל צומת B ב- CFG (בלוק בסיסי) נגדיר את **פריטי המידע הבאים**:

– $out(B)$ – כל משתני התוכנית שחיים לפני תחילת הבלוק B .

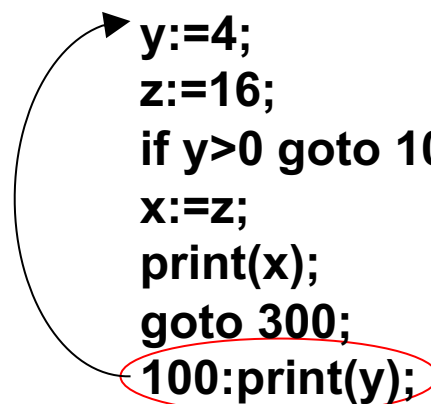
– $in(B)$ – כל משתני התוכנית שחיים מיד לאחר הבלוק B .

שימוש ב DFA לניתוח חיות

• מהו כיוון הזרימה?

– זרימה אחורית

```
print(z);  
z:=12;  
y:=4;  
z:=16;  
if y>0 goto 100;  
x:=z;  
print(x);  
goto 300;  
100:print(y);  
goto 300;  
300:print(x);
```



שימוש ב DFA לניתוח חיות - המשך

- הגדרת Gen/kill:

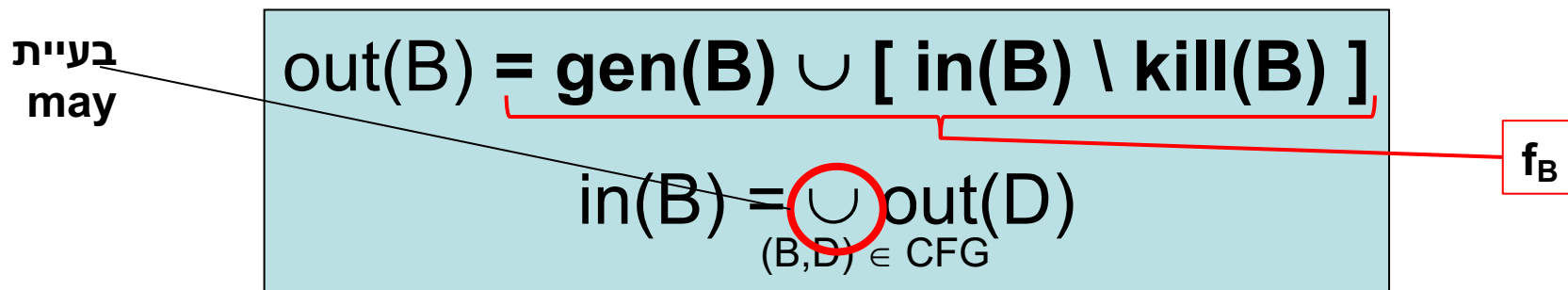
- מהו Gen(B)?

$\text{gen}(B) = \{ x \mid x \text{ משתנה שנעשה בו שימוש ב } B \text{ ואין כתיבה לתוכו לפני השימוש} \}$

- מהו Kill(B)?

$\text{kill}(B) = \{ x \mid x \text{ הוא משתנה אשר מבוצעת אליו כתיבת ערך ב } B \}$

- הגדרת \sqcup ומשוואות הזרימה:



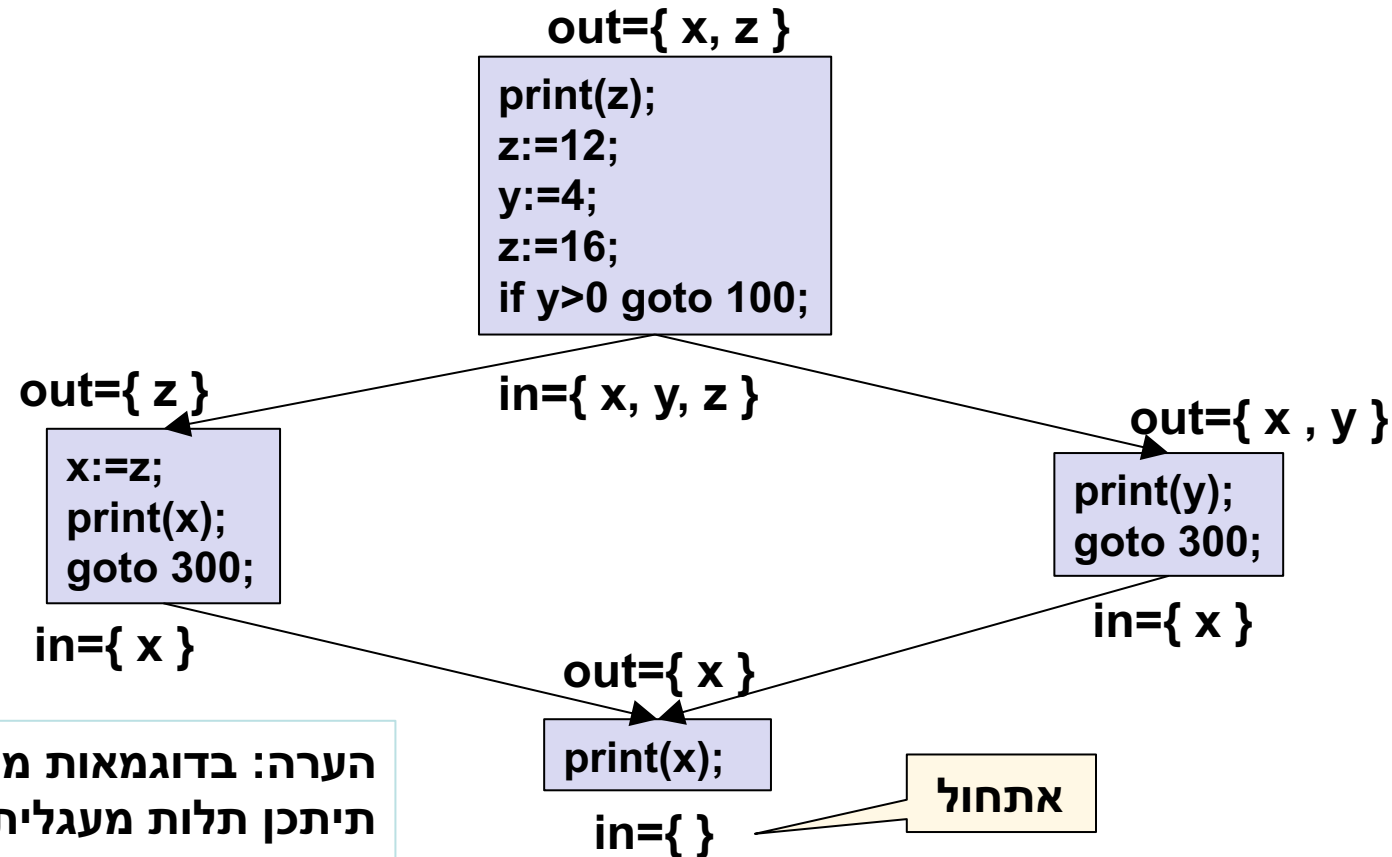
שימוש ב DFA לניתוח חיות – אתחול

- אתחול הפריטים $\text{In}(B)$, $\text{Out}(B)$ בצורה שמרנית:
 - במקרה שלנו, במהלך הסריקה האחורית פריטים מתווספים לקבוצה in (כי כך הגדרנו את \sqcup)
 - לכן עבור כל בלוק נאתחל: $\text{in}(B) = \emptyset$.
- בסוף התוכנית בוודאי אף משתנה אינו חי.
- גם אם כל התכנית היא לולאה, לא נרצה פריטים שאינם חיים ביציאה מהלולאה (עוזר להניח בלוק אחרון יחיד)
- נבצע איטרציות על כל הבלוקים עד להתייצבות.

שימוש ב DFA לניתוח חיות - דוגמא

$$\text{out}(B) = \text{gen}(B) \cup [\text{in}(B) \setminus \text{kill}(B)]$$

$$\text{in}(B) = \bigcup_{(B,D) \in \text{CFG}} \text{out}(D)$$

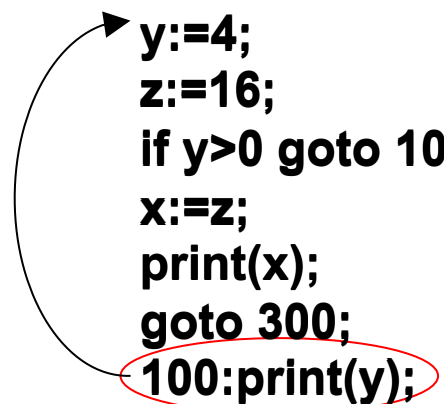


הערה: בדוגמאות מסובכות יותר
תיתכן תלות מעגלית. יידרשו
הפעלות חוזרות עד להתייצבות.

שימוש ב DFA לניתוח חיות

- כיוון הזרימה: אחורית

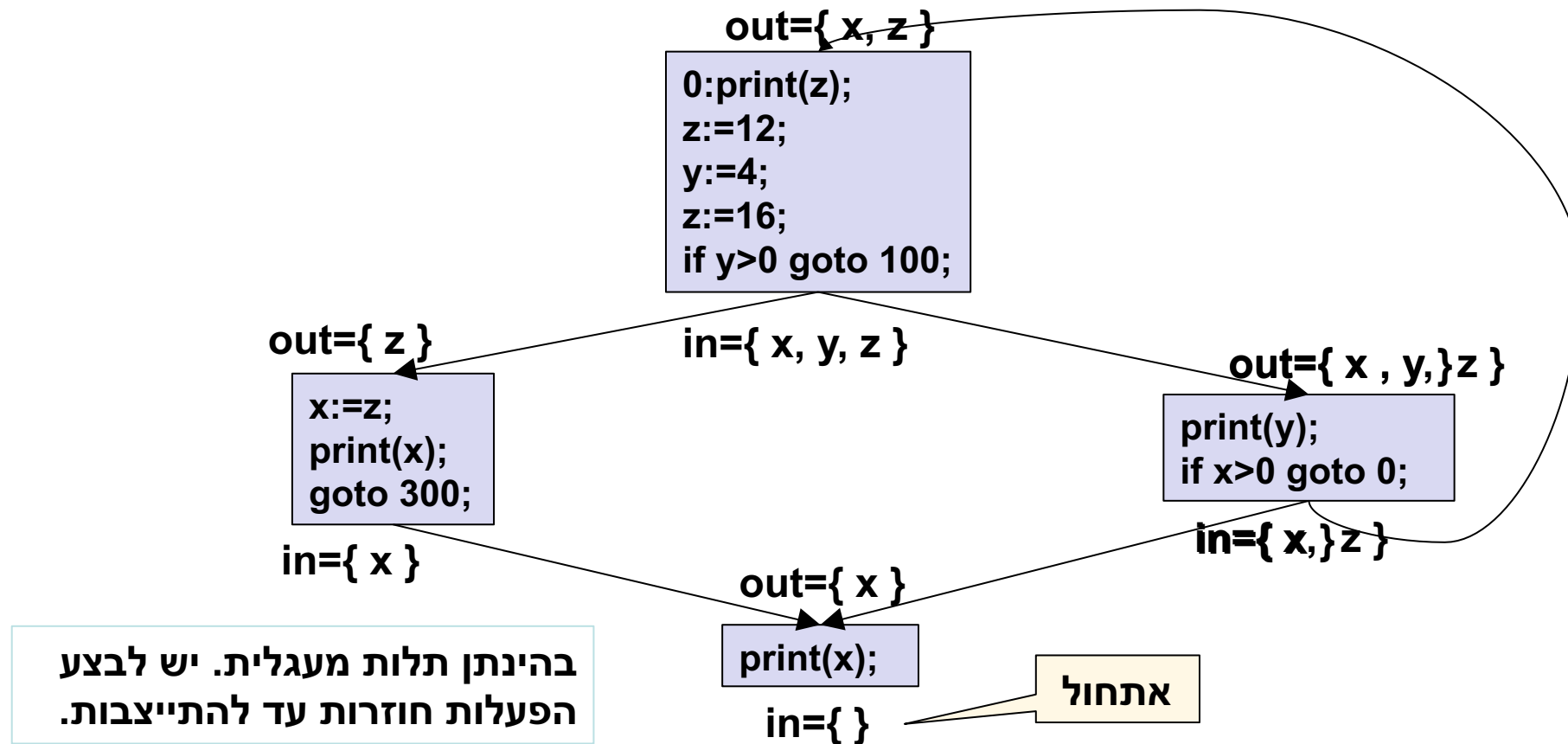
```
print(z);  
z:=12;  
y:=4;  
z:=16;  
if y>0 goto 100;  
x:=z;  
print(x);  
goto 300;  
100:print(y);  
if y>0 goto 0;  
goto 300;  
300:print(x);
```



שימוש ב DFA לניתוח חיות - דוגמא

$$\text{out}(B) = \text{gen}(B) \cup [\text{in}(B) \setminus \text{kill}(B)]$$

$$\text{in}(B) = \bigcup_{(B,D) \in \text{CFG}} \text{out}(D)$$



שימוש ב DFA לניתוח חיות

$$\text{out}(B) = \text{gen}(B) \cup [\text{in}(B) \setminus \text{kill}(B)]$$

$$\text{in}(B) = \bigcup_{(B,D) \in \text{CFG}} \text{out}(D)$$

- איך מקבלים את מה שנדרש בשאלה מ-DFA?
 - לכל בלוק B, $\text{in}(B)$ יכיל את קבוצת המשתנים החיים ביציאה מהבלוק. כעת ניתן עבור כל בלוק בנפרד, לבצע הרצה אחת כדי למצוא את המשתנים החיים בסיום כל פקודת השמה. (אנליזה פנימית לבלוק)
- היתרון שהושג:
 - קיבלנו את המשתנים החיים עבור כל הבלוקים ב CFG בבת אחת.

ניתוח 2-חיות

- תזכורת: משתנה x הוא חי בנקודה p בתכנית אם קיים מסלול $m-p$ בו יש שימוש ב- x (קריאת ערך x) לפני הגדרה מחודשת (כתיבה מחדש ל- x).
 - נגדיר: משתנה x הוא **2-חי** בנקודה p בתכנית אם קיים מסלול $m-p$ בו יש שני שימושים ב- x לפחות לפני הגדרה מחודשת.
- א. $in^2(B), out^2(B)$ – קבוצת המשתנים ה-2 חיים בכניסה/יציאה מהבלוק.
- ב. זרימה אחורית – בדומה לניתוח חיות.
- ג. בעיית may – מספיק מסלול אחד שיקיים את התכונה.

ניתוח 2-חיות - המשך

- אילו משתנים נרצה שיופיעו ב $?out^2(B)$

1. כל משתנה ב- $in^2(B)$ ללא הגדרה מחודשת ב-B.

2. כל משתנה עם לפחות שני שימושים ב-B לפני הגדרה מחודשת.

3. כל משתנה שיש לו שימוש ב-B, הוא חי ביציאה מ-B ולא הוגדר מחדש ב-B.

- מהו $?gen(B)$ 

– כדי למצוא את המשתנים החיים ביציאה מ-B נבצע הרצה מקדימה של ניתוח חיות.

- מהו $?kill(B)$

– כל המשתנים שנכתב אליהם ערך ב-B.

ניתוח 2-חיות - המשך

$\text{kill}(B) = \{ x \mid x \text{ משתנה שהוגדר מחדש ב- } B \}$

$\text{gen2}(B) = \{ x \mid x \text{ שני שימושים בבלוק לפני הגדרה מחודשת} \}$

$\text{gen1}(B) = \{ x \mid x \text{ יש שימוש בבלוק ואין הגדרה מחודשת} \}$

$$\text{out}^2(B) = \text{gen}^2(B) \cup [\text{in}^2(B) \setminus \text{kill}(B)] = \\ (\text{gen2}(B) \cup [\text{gen1}(B) \cap \text{in}^1(B)]) \cup [\text{in}^2(B) \setminus \text{kill}(B)]$$

הקבוצה in שהתקבלה מההרצה
המוקדמת של ניתוח החיות

$$\text{in}^2(B) = \bigcup_{(B,D) \in \text{CFG}} \text{out}^2(D)$$

ה. אתחול הקבוצות:

$$\forall B, \text{in}^2(B) = \text{out}^2(B) = \emptyset$$

סיכום

- מה משתנה בעקבות ההנחה על ייצוג צמתים ב-CFG (בלוק בסיסי / פקודה בודדת)?
 - × משוואות הזרימה
 - × אתחול הקבוצות in / out
 - ✓ פונקציות gen ו-kill
- הצורה בה פתרנו את השאלות לפי סדר השאלות ששאלנו הינה מתכון לפתרון שאלות ה-DFA

צריך לציין עבור פתרון בעיית DFA בצורת gen/kill

- מהי מטרת החישוב?
- מהו הסריג: האיברים, הסדר החלקי (\sqsubseteq) ופעולת ה-join (\sqcup)?
- מהו כיוון זרימת המידע? (קדמית / אחורית)
- מהו $gen(B)$? מהו $kill(B)$?
- משוואות הזרימה.
- איך נאתחל את הקבוצות?
- איך מקבלים את מה שנדרש בשאלה מהאנליזה?