



על הקצה

הטור הקודם (**תוכנית החלוקה**) עסק בטכניקת הבדיקה של מחלקות שקילות. הפעם אתמקד בטכניקה הקשורה לה: בדיקת ערכי גבול, וגם כאן יתברר שהחיים יותר מסובכים מאשר התרגילים שפותרים בקורס בדיקות.



מיכאל שטאל

ארכיטקט בדיקות תוכנה באינטל, ישראל, עוסק בעיקר בבדיקות מערכות משובצות מחשב. במסגרת תפקידו, מיכאל מגדיר שיטות בדיקה ומתודולוגיות עבודה, עוסק בהדרכה ולפעמים אפילו מרשים לו לבדוק משהו (שזה הכי כיף).

מיכאל מציג תכופות בכנסים בארץ ובחו"ל ומלמד בדיקות תוכנה בפקולטה למדעי המחשב באוניברסיטה העברית. ניתן לראות חלק מהמצגות והמאמרים שלו באתר

www.testprincipia.com



שקילות על ידי משפטי תנאי (if-else), ואז קל לזהות את מקרי הגבול של התנאים. כיוון שממילא צריך לבחור נציגים מכל מחלקה לצורך כיסוי כל המחלקות, וכיוון שהחיים מראים שיש שכיחות טעויות בקצוות של ערכי הקלט – אז הגיוני לבחור את ערכי הגבול בתור הנציגים של מחלקות השקילות. גם כיסינו את המחלקה, וגם בחרנו ערכי קלט שיש יותר סיכוי שימצאו באג.

ייש קשר הדוק בין מחלקות שקילות וערכי גבול"



הצד הפחות חביב של הקשר למחלקות השקילות הוא שהאתגרים הקיימים בחלוקה למחלקות שקילות משפיעים באופן ישיר על נכונות בחירת ערכי הגבול: אם טעינו בחלוקה למחלקות, הטעות תיגרר להגדרת ערכי הגבול. זה גם מעלה את מספר הבדיקות הנדרש: אם עבור כיסוי מחלקות שקילות הסתפקנו בנציג יחיד מכל מחלקה ולכן מקרה בדיקה יחיד, לכיסוי גבולות של טווח מסויים (למשל... דוגמא מקורית מאוד: ערכים מ-1 עד 10), צריך כבר ארבע מקרי בדיקה (0,1,10,11) וזה כשמשמשים בשני ערכי גבול. אם משתמשים בשלושה ערכי גבול אז כבר מדובר בששה טסטים (0,1,2,9,10,11).

תזכורת: בדיקות ערכי גבול

טכניקת הבדיקה של "ניתוח ערכי גבול" מבוססת על כך שסיבות נפוצות לבאגים הן תופעות שקשורות לגבולות. ישנם כמה מקרים שבהם אפשר לדבר על "גבול": המקרה הפשוט ביותר הוא כאשר התוכנה מצפה לקבל קלט בטווח מסויים. למשל - שדה החדוש בתאריך: הציפיה שיהיה בין 1 ל-12. קצת יותר מסובך: כשת-טווח של הקלט קובע את התנהגות התוכנה: מגיל 0 עד 2 לא משלמים על כרטיס טיסה; מגיל 2 עד 12 משלמים 75%, ומגיל 12 עד 120 – מחיר מלא.

REMINDER



במקרים אלה ישנן מספר טעויות אפשריות:

- (א) המפתחים לא הבינו נכון את הדרישה (למשל: חברת התעופה "מבינה" שגיל 0 עד 2 כולל ילדים שעברו את גיל 1, אבל אינם עדיין בני שנתיים; המפתחים הבינו שזה כולל ילדים שעברו את גיל שנתיים אבל אינם עדיין בני שלוש)
- (ב) המפתחים עשו טעות בקוד. במקום לכתוב " $1 >=$ " הם כתבו " $2 >=$ "
- (ג) המפתחים לא ציפו שיש בכלל אפשרות לקבל קלט מעבר לטווח המצופה. אם נניח שהטיפול בערך החדוש הוא על ידי פקודת switch (או case בשפות אחרות), אז אין קוד default שמטפל במקרים שבהם הקלט אינו חוקי (למשל 13).

מכאן ההמלצה לתכנן בדיקות שהקלט שלהן מכיל ערכי גבול. סביב כל גבול נריך ערך אחד שהוא בטווח, אבל האחרון שעדיין שם (ילדים בני 0 ו-1; חודשים 1 ו-12) וקלט ש"בדיק" כבר לא בטווח (ילדים בני 2; חודשים 0 ו-13).

זה כל כך פשוט!

דוגמא פשוטה ונפוצה ביותר בכל מקום שבו מוסברת הטכניקה, היא עבור קלט שיכול לקבל ערכים בין 1 ל-10. ערכי הגבול הם 0, 1 עבור הגבול התחתון, ו-10, 11 עבור הגבול העליון. כבר כאן יש הצגה מאוד פשטנית של הטכניקה. קודם כל, לא תמיד הקלט הוא במספרים שלמים; מה הגבולות אז? לפעמים הקלט הוא מחזורי; מה הגבולות? כמו כן, יש מקרים שבהם ערכים מעל 10 או מתחת ל-1 אינם ערכים שגויים, אלא ערכים שעבורם המערכת מתנהגת אחרת. איך לקבוע את ערכי הגבול במקרים אלה? ישנם גם מקרים שבהם לא מדובר במשפט תנאי של "אם הערך גדול מ-X אז..." אלא הגדרת תנאי סיום ללולאות של for או while, ושם המצב יותר מסובך. תוסיפו על זה את האפשרות להגדיר שני ערכי גבול או שלושה ערכי גבול, וכבר מובן שיש מקום להיכנס קצת יותר לעומק.



ערכי גבול ומחלקות שקילות

נתחיל בזה שיש קשר הדוק בין מחלקות שקילות וערכי גבול. טכניקת הבדיקה של חלוקה למחלקות שקילות מציעה לבצע בדיקות עם ערכי קלט שמייצגים את מחלקות השקילות - לפחות נציג אחד מכל מחלקה. בדרך כלל הקוד מחלק את מרחב הקלטים למחלקות





שניים או שלושה (רמז: ארבע!)

יש סיבה טובה למה צריך יותר משני מקרי בדיקה כדי לכסות כל גבול. ניקח דוגמה של תוכנה למכירת ביטוח חיים¹. התוכנה מממשת כלל פשוט: יש אישור לבטח מי שמתחת לגיל 99. אפשר לנחש בוודאות גבוהה שאיפשהו בקוד יהיו שורות שיממשו משהו בסגנון הבא:

If age <= 99 then Accept; Else Reject

אם נשתמש בבדיקות ערכי גבול עם שני ערכים, נריץ שתי בדיקות:

גיל	תוצאה צפויה
99	Accept
100	Reject

אבל מה אם המפתח שגה, ובמקום " \geq " כתב " $<$ "? ברור שהתוצאות בפועל יהיו הפוכות לתוצאות הצפויות, הבאג ימצא, ואנו הבודקים, נסמן עוד חריץ על קת הרובה. אבל האם, בעזרת שתי בדיקות, נוכל לתפוס את כל השגיאות הלוגיות האפשריות? במאמר המקורי יש טבלה עם כל האפשרויות, וממנה מתברר שיש טעות אחת שלא ניתן לתפוס על ידי שתי בדיקות בלבד:

If age == 99 then Accept; Else Reject;

על מנת "לתפוס" גם את המקרה הזה, צריך להריץ עוד ערך מתוך מחלקת השקילות התקפה, שאינו ממש על הגבול. כל ערך יעשה את העבודה, אז נניח שבחרנו בערך 98.

אבל מה אם הקוד מומש ככה:

If age < 100 then Accept; Else Reject;

כמעט כל הטעויות הלוגיות יתפסו על ידי הרצת הערכים 98, 99, 100. חוץ מהטעות הבאה:

If age < > 100 then Accept; Else Reject;

בשביל לתפוס טעות זו, צריך להריץ ערך נוסף ממחלקת השקילות שאינה תקפה, שאינו על הגבול. גם כאן כל ערך ייתן את אותה תוצאה, ואם כך אפשר לבחור את 101.

אז הסברנו למה כדאי לבדוק ערכי גבול עם 4 ערכים. גם ההצדקה לבדיקות עם שני ערכים ברורה: זה תופס את רוב הטעויות, וודאי את ה"הגינות" שבהן (תסכימו שהמקרים שאינם נתפסים בעזרת שני ערכי גבול הם, איך נאמר... גבוליים). לעומת זאת, מאיפה ההמלצה לשלוש ערכי גבול... לא לגמרי ברור. אני אנסה להסביר את זה בניפנופי ידיים נמרצים, אבל מודע לכך שזה הסבר חלש.

בהרבה מקרים יש למחלקת שקילות שני גבולות: תחתון ועליון:



על פי העקרון של ארבע בדיקות בכל גבול, נריץ: ערך תקף על הגבול (ירוק); ערך לא-תקף על הגבול (אדום); ערך לא-תקף שאינו על הגבול (כתום). מה עם הערך הרביעי (ערך תקף שאינו על הגבול)? מסתבר שיש לנו אותו: עבור הגבול התחתון, הערך התקף של הגבול העליון הוא "ערך תקף שאינו על הגבול", וכנ"ל לגבי הגבול העליון.

"אם טעינו בחלוקה למחלקות, הטעות תיגרר להגדרת ערכי הגבול"

זכרו גם שאת הערכים שאינם על הגבול אפשר לשים צמוד לערך הגבול:



קיבלנו משהו שאכן נראה כמו שלוש ערכים לכל גבול. מ.ש.ל.

יש רק בעיה קטנה: על פי התיאוריה של שלושה ערכי גבול מדובר בערך שעל הגבול, ושני ערכים משני צידיו. כלומר, בעצם צריך להסביר בחירת בדיקות כזאת:



יתכן שההסבר נעוץ בדרך שבה מתארים את הבעיה. הרי זה לא באמת כמו הציורים למעלה. אם הטווח המותר הוא "בין 1 ל-10" אז התיאור הנכון עבור בדיקות עם שני ערכי גבול הוא שהערך התקף יושב ממש "על" הגבול:



וזה קצת מציק בעיני. נראה איכשהו לא סימטרי. ומכאן הדרך סלולה למשהו שנראה יותר טוב (אם כי לא בטוח כלל שנותן בדיקות חזקות יותר):



לא כולם ביחד!



העולם כידוע יותר מסובך מקלט אחד בטווח של 1 עד 10. בהרבה מקרים הקלט מכיל יותר ממשנתה אחד ולכל משנתה יש, מה לעשות, ערכי גבול. איך נממש את הטכניקה של בדיקות ערכי גבול במצב זה?

לצורך ההמשך, נניח שאנו מפתחים תוכנה שמזהה עצמים בתוך תמונה. התוכנה משתמשת בחומרה שעושה את רוב העיבוד, ועל מנת שהחומרה תהיה פשוטה יותר, הוחלט שהתמונה שתישלח לחומרה תהיה תמיד באותו גודל: 1500x1000 פיקסלים. התוכנה עצמה מקבלת תמונות בגדלים מ-150x100 ועד 9000x6000 פיקסלים, וכל תמונה שתקבל תעבור המרה לגודל 1500x1000 לפני שתועבר לחומרה. תמונות שבהן היחס בין הגובה לרוחב אינו 1:1.5, ידחו כלא חוקיות.

נתחיל בזה שעכשיו יש לנו ארבעה פרמטרים שאפשר לנתח אותם מבחינת מקרי גבול:

גובה התמונה: 100 עד 6000

רוחב התמונה: 150 עד 9000

שטח התמונה: 150x100 עד 9000x6000

יחס גובה/רוחב: 1:1.5

מפתה להחליט שנבדוק את כל ערכי הגבול יחד:

גבול תחתון	גבול עליון
149x99	9001x6001

¹ דוגמה זו וההסבר שלה לקוחים ממאמר שהתפרסם במגזין Testing Experience בשנת 2008: The Boundary Value Fallacy, René Tuinhout, Testing Experience Magazine, Sep 2008 https://www.istqb.org/images/Articles/konushin_personnel%20evaluation%20based%20on%20tests.pdf



התמונה. לעומת זאת, על פי הגדרה זו, נוצר לנו פרמטר נוסף (גודל האובייקט) שגם עליו צריך לבצע ניתוח ערכי גבול, ולבדוק בהתאם. נשמע קל, אבל הגדרות מסוג זה הן מאוד קשות, כי הרי זה לא רק גודל העצם; זה הזווית בה הוא צולם; הקונטרסט בינו ובין הרקע; חדות הצילום; וכו' וכו'. לעיתים בלתי אפשרי להגדיר ממש את הגבול, אלא הגדרות הרבה יותר מעורפלות. אז מריצים את התוכנה עם הרבה מקרים, מסתכלים על התוצאה ומחליטים אם היא "טובה מספיק".

יחס הרחבולובה בתמונה גם הוא מקרה מעניין שכן מדובר בפרמטר עם ערך שאינו שלם. מה מקרה הגבול? האם תמונה של 8999 600 (יחס: 1:1.4998333) נחשבת מספיק קרובה ל-1:1.5? אם לא נגדיר את הדיוק שאנו דורשים (למשל: +/- 0.002 (1:1.5) הרי שהתוצאה תהיה תלויה במערכת ההפעלה ובחומרה עליה רצה התוכנה - כי הם קובעים את רמת הדיוק בייצוג מספרים ממשיים. זה יכול להיות בעייתי אם זו תוכנה שתותקן במערכות שונות ומגוונות, או סביר לגמרי אם התוכנה מיועדת למימוש מאוד ספציפי. בכל מקרה, עלינו כבודקים להבין מהו הדיוק הנדרש, על מנת להגדיר את מקרי הגבול, ומשם הדרך למקרי בדיקה קלה.

לסיכום, ישנם מקרים (רבים) שבהם בדיקות מקרי גבול ברורות ומיידיות. אבל יש גם לא מעט מקרים שבהם העקרון לא ממש עובד, או קשה ליישם אותו. במקרים אלה, העקרון של "מקרה גבול" עוזר לנו לחשוב על בדיקות שכדאי להריץ, גם אם הם לא ממשות את הטכניקה לפי הספר. רוח הדברים של הטכניקה מנחה אותנו לחפש מתי התוכנה עוד מתפקדת ומתי מרימה ידיים, ולו גם ברמה אבסטרקטית של חיפוש מקרים ש"כמעט נכשלים" ומקרים ש"כמעט עוברים".

9000x6000	150x100
8999x5999	151x101

אבל יש סיבה טובה למה לא: הקוד. מן הסתם, בדיקת גודל התמונה מתבצעת מיד בהתחלה. משהו בסגנון:

If (height < 100 or height > 6000 or width < 150 or width > 9000) then Error

"רוח הדברים של הטכניקה מנחה אותנו לחפש מתי התוכנה עוד מתפקדת ומתי מרימה ידיים"

כשקוד זה עובר קומפילציה, התוצאה מכילה Short Circuiting. זוהי אופטימיזציה שהקומפיילר מבצע על מנת לחסוך זמן עיבוד בשעת ההרצה. הבה ננתח: כאשר הגובה קטן מ-100, כבר התנאי הראשון מתקיים. כיוון ששאר התנאים הם "או" (OR), אין שום טעם להמשיך ולבדוק את שאר התנאים. ברור שהתוצאה הסופית של כל הביטוי הלוגי תהיה "אמת", ולכן אפשר לנטוש את הביצוע כבר אחר שהתנאי הראשון מתקיים. מבחינה מעשית זה אומר שאם נשתמש בתמונה בגודל 149x99 על מנת לבדוק את הגבול התחתון, גם הקוד הזה ייתן את התוצאה הצפויה:

If (height < 100 or height > 6000 or width > 9000) then Error

ולמעשה המקרה של $width > 150$ כלל לא נבדק.

ההמלצה לגבי בדיקות מקרי גבול כאשר יש יותר ממשתנה אחד היא לבדוק כל משתנה בנפרד (להציב מקרי קצה למשתנה זה) תוך שאנו מציבים ערכים נומינליים (שאינם בקצוות) לכל המשתנים האחרים. מעבר לזה, מומלץ להריץ בתור התחלה מקרה שבו כל המשתנים בערכים נומינליים. אם בדיקה זו תעבור, ואחרות לא, זו אינדיקציה מצויינת שהבעיה היא בגלל מקרה קצה. אם גם בדיקה זו לא תעבור, אז כנראה אף בדיקה לא תעבור כי לפחות אחד המשתנים מטופל לגמרי לא נכון. בהנחה שיש ח משתנים, שנבחר לבדוק עם שני ערכי גבול, ושלכל משתנה יש גבול עליון ותחתון, מדובר ב-4n בדיקות, ועוד בדיקה אחת שבה כל המשתנים בערכים נומינליים.

כיוון שגם שטח התמונה הוא פרמטר, הרי יש גם הצדקה להריץ את התמונה הגדולה ביותר שאפשר לעבד בתוכנה שלנו: 9000x6000. לעומת זאת, אין טעם לנסות תמונה בגודל גדול יותר (למשל 9000x6001) כי ברור מבדיקות מקרי הגבול של גובה רוחב שקלט זה ידחה. גם אין משמעות לבדוק תמונה בגודל 8999 6000 או 9000x5999, שהרי מטרת הבדיקה עם תמונה בעלת שטח מקסימלי היא לבדוק שהמערכת מקצה מספיק משאבים לקלוט כמות כזו של נתונים (54 מיליון פיקסלים). אם התמונה הגדולה ביותר מטופלת היטב, הרי שיש במערכת מספיק משאבים לכך, וודאי שיש מספיק לתמונות קטנות יותר.

גבולות מעורפלים

נשארנו עם שני פרמטרי קצה: התמונה הקטנה ביותר, והיחס גובה/רוחב.



איזה מקרה גבול מייצג את התמונה הקטנה ביותר? זה לא חד משמעי כלל, ויתכן שהתמונה הקטנה ביותר אינה מקרה גבול לשום דבר. בדוגמה שלנו התוכנה מזהה עצמים בתמונה. נניח שהתוכנה מזהה עצמים אם הם בגודל של לפחות 150x150 פיקסלים. במקרה כזה, תמונה של 100x150 כנראה ממש לא מעניינת אלא רק כמקרה בדיקה שלילי להראות שכשאף חפץ לא מזהה - או שבתמונה יש רק חלק מעצם - התוכנה לא קורסת. אם הגודל המינימלי של עצמים הוא 98x98 פיקסלים, אז יש (כמעט במקרה) משמעות לגודל המינימלי של



LOADING WEEKEND

MONKEYUSER.COM

