# בעיית שני הגנרלים

## תיאור הבעיה

שתי צבאות נלחמים זה בזה. הצבא הכחול, החזק יותר, בעמדה נחותה, ואילו הצבא האדום, החלש יותר, פיצל את כוחותיו לשתי עוצבות בגבעות שונות המכתרים את הצבא הכחול. לכל עוצבה יש גנרל. הצבא האדום ינצח רק אם שני הגנרלים של שני העוצבות יצליחו לתאם ביניהם לתקוף באותו הזמן. לשם כך הם מתקשרים ביניהם באמצעות יונות דואר. אמנם הצבא הכחול רואה את היונים ומנסה לירות בהן כדי לשבש את התקשורת של הצבא האדום. כיוון שהמרחק בין העוצבות גדול, גנרל ששולח יונה לא מצליח לראות האם היונה הגיע לצד השני, וכן הגנרל בצד השני לא יודע כאשר יונה נשלחת. הבעיה היא כיצד הגנרלים יכולים לתקשר ביניהם במצב כזה כדי להבטיח תקיפה של שניהם באותו הזמן שתוביל לניצחון?

הבעיה המתוארת לעיל היא אנלוגיה של הבעיה הבאה במערכות מבוזרות: עבור שני צמתים במערכת מבוזרת **מסונכרנת** שבה חלק מההודעות הנשלחות יכולות ללכת לאיבוד (Omission failure) וכל אחד מהצמתים יכול לקבל כקלט 0/1 ולפלוט 0/1, האם קיים אלגוריתם עבור שני הצמתים שיקיים את שלושת הדרישות הבאות:

1. Agreement - שני הצמתים צריכים להחזיר את אותו הפלט (0 או 1).
2. Termination - שני הצמתים מסיימים את פעולתם במספר סופי של סיבובים (rounds).
3. Validity - אם שני הצמתים קיבלו אותו קלט וגם לא היה איבוד של הודעות בשני הכיוונים אזי שני הצמתים יפלטו את הקלט שלהם. אם אחד משני התנאים לא מתקיים אין חובה כזו.

נשים לב שללא Validation יש אלגוריתם מאוד פשוט שפותר את הבעיה - אלגוריתם שתמיד מחזיר 0. בעיית שני הגנרלים מייצגת בעיה כללית יותר של כיצד תהליכים מגיעים להסכמה במערכת מבוזרת שבה לא בטוח שכל ההודעות מגיעות ליעד. ישנם מקרים רבים בהם תהליכים צריכים להגיע להסכמה, כמו לדוגמה: מי מוביל חישוב, כיצד לחלק סדר פעולות, האם לבצע פעולה או לא, מי הראשון שניגש למשאבים, וכו'.

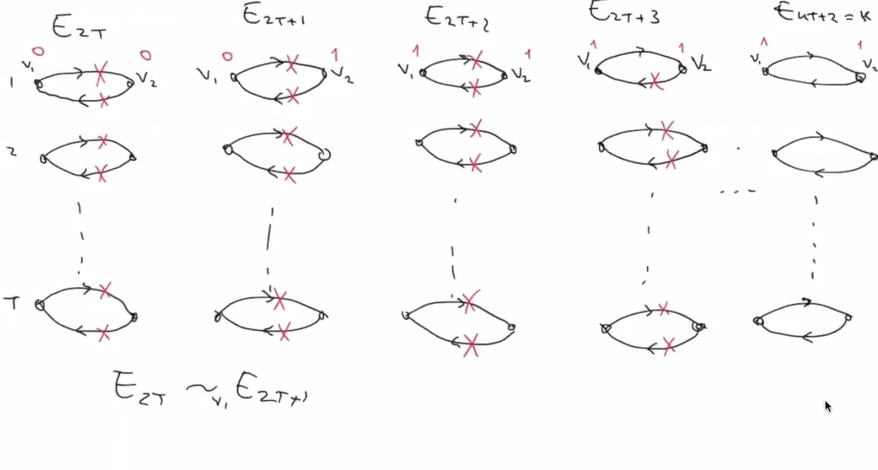
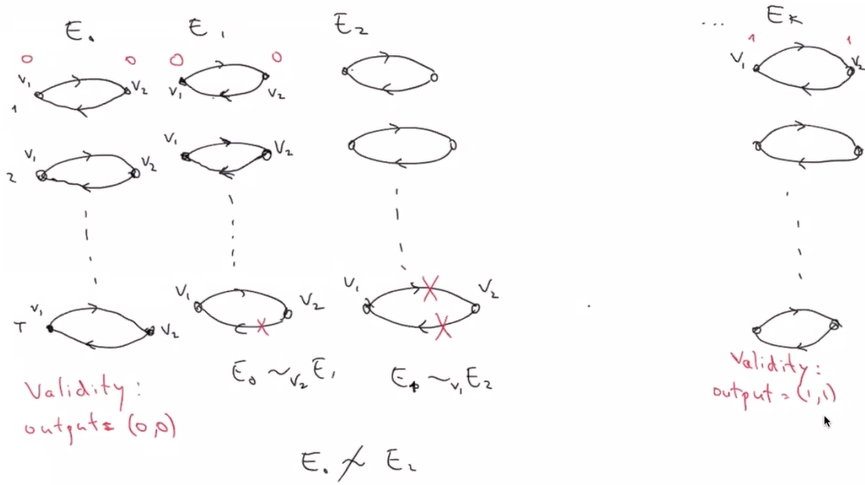
## הוכחת אין פתרון

**משפט:** לבעיה המוצגת לעיל, למרות שנראית פשוטה, לא יכול להיות פתרון!

**הוכחה:** נניח בשלילה שיש אלגוריתם כזה שעבור כל הרצה אפשרית עונה על כל דרישות הבעיה. נגדיר רצף של הרצות (Executions) כך שכל שתי הרצות סמוכות [דומות](#_משפט_אי_ההבחנה), כלומר לכל קיים צומת עבורה . נניח בנוסף שכל הרצה מחזירה פלט לאחר בדיוק T סיבובים. נוכל להניח זאת שכן ההרצות חייבות להסתיים לאחר מספר לא סופי של סיבובים ותמיד ניתן להוסיף סיבובים שלא משפיעים על הפלט.

אם נצליח להוכיח שעבור הרצה שני הצמתים חייבים להחזיר 0 ועבור הרצה שני הצמתים צריכים להחזיר 1 סיימנו, שכן אם בהרצה שניהם החזירו 0 אזי גם בהרצה שניהם יהיו חייבים להחזיר 0, שהרי ב- אחת הצמתים בוודאי החזירה 0. כך נמשיך ברצף עד שנגיע ל- שגם בו שני הצמתים יהיו חייבים להחזיר 0. אמנם כיוון שנוכיח שחייבים להחזיר 1 נקבל שהאלגוריתם יכשל עבור הרצה זו, בניגוד להנחה.

נוכיח שקיים רצף של הרצות כזה. בהרצה שני הצמתים יקבלו כקלט 0 וכל ההודעות יצליחו. הרצה זהה ל- אלא שההודעה האחרונה מ- לא הגיעה ל-. מבחינת אין הבדל ולכן ההרצות דומות. כך נמשיך כל הרצה לבטל את ההודעה האחרונה עד שבה כל ההודעות לא נשלחו. ב- נשנה רק את הקלט של ל-1, וב- נשנה את הקלט של ל-1. מ- כל הרצה נוסיף הודעה שבהרצות הקודמות נכשלה. בהרצה ה- נקבל ששני הצמתים קיבלו כקלט 1 וכל ההודעות נשלחו, לכן שניהם חייבים להחזיר 1. אמנם ה צריכים להחזיר 0. סתירה!



## תיאור חדש לבעיית שני הגנרלים

כיוון שהוכחנו שלבעיה זו אין אלגוריתם שפותר אותה נשנה את הבעיה מעט. ניתן לפתור את בעיית שני הגנרלים אם:

1. אנו מרשים לגנרלים להטיל מטבע ולקבל החלטות על סמך התוצאה. מספיק שגנרל אחד יקיים תנאי זה.
2. אנו מסופקים גם אם תנאי Agreement (שני הצמתים צריכים להחזיר את אותו הפלט) מתקבל בהסתברות .

בשני תנאים אלו אנו גם מגדילים את היכולת וגם מקילים את הדרישה. נראה אלגוריתם מקרי (Randomized Algorithm) שפותר את הבעיה החדשה. אך לפני כן נצטרך להגדיר אלגוריתם עזר, שנקרא Level Algorithm, שנשתמש בו באלגוריתם המקרי שנציג.

### Level Algorithm

עבור שני צמתים u, v השולחים הודעות אחד לשני (לא בטוח שכל ההודעות מגיעות ליעד), נגדיר לכל צומת מספר הנקרא "הרמה (Level) של הצומת". מספר זה מייצג את מספר ההודעות שנשלחו וחזרו בהצלחה. נרצה אלגוריתם שבסוף ריצתו שתי הרמות של הצמתים יהיו בהפרש של לכל היותר אחד. Level Algorithm עונה על דרישה זו. הוא מבצע את הפעולות הבאות:

1. שתי הרמות מאותחלות ל-0.
2. בכל סיבוב שני הצמתים שולחים את הרמה הנוכחית שלהם אחד לשני.
3. כאשר צומת u עם רמה ℓu מקבל הודעה מצומת v, עם הרמה ℓv, אזי u מעדכן את רמתו להיות: ℓu = max {ℓu, ℓv+1}.

אבחנות:

* הרמה של צומת לעולם לא יורדת.
* בכל סיבוב הרמות של שני הצמתים יהיו בהפרש של לכל היותר 1.
* אם כל ההודעות מתקבלות אזי רמות שני הצמתים שוות למספר הסיבובים.
* הרמה של צומת u שווה ל-0 אם ורק אם u לא קיבל אף הודעה. לכן אם הרמה לכל הפחות 1, בהכרח u קיבל הודעה.

ההוכחה לכל אחת מאבחנות אלו היא באמצעות אינדוקציה פשוטה ונמצאת במצגת 2.

## Randomized Two Generals Algorithm

1. נניח שצומת u יכול להטיל מטבע ולקבל החלטות על סמך התוצאה, אך צומת v לא יכול.
2. צומת u בוחר מספר באופן אחיד ואקראי.
3. שני הצמתים מריצים את Level Algorithm למשך r סיבובים. בכל סיבוב כל אחד מהם שולח את ערך הקלט ההתחלתי שלו. צומת u בנוסף גם כולל את הערך t שבחר באקראי בכל הודעה.
4. צומת תפלוט 1 אם מקיימת שלושה תנאים:
5. הצומת יודע מהו t וגם ראה את הקלט של הצומת המקבילה.
6. הקלט של שני הצמתים הוא 1.
7. הרמה של הצומת היא לפחות t.
8. אחרת, הצומת תפלוט 0.

אבחנות:

* אם לפחות אחד הקלטים 0 שני הצמתים יחזירו 0.
* אם שני הקלטים 1 ואף צומת לא נאבדה - שניהם יחזירו 1.
* אם שני הקלטים 1 ויש הודעות שנאבדו - שני הצמתים יחזירו אותו הפלט אלא אם כן , שאז יחזירו פלט שונה והאלגוריתם יכשל. ההוכחה נובעת מהגדרת האלגוריתם (ניתן לראות פירוט בשקף 23).

**טענה:** האלגוריתם מקיים Agreement עם הסתברות לכל הפחות . ככל שנבחר r גדול יותר כך נוכל לקרב את ההסתברות ל-1.

**הוכחה**: האלגוריתם יסתיים לאחר r סיבובים ולכן מקיים Termination. בנוסף, משתי האבחנות הראשונות ניתן לראות שמקיים Validation. נוכיח את הטענה. נדמיין שהיריב מנסה להכשיל את האלגוריתם שלנו. הוא שולט בקלט ובאיזה הודעות ישלחו ואיזה לא. היריב יבחר קלט (1,1) אחרת האלגוריתם תמיד יצליח. כעת, הוא ירצה להכשיל מספר הודעות כך שהרמות של הצמתים יהיו , אחרת האלגוריתם שוב תמיד יצליח. אמנם היריב אינו יודע מהו t שכן נבחר באקראי, ולכן הוא חייב לבחור כלשהו. היריב אכן יצליח להכשיל את האלגוריתם בהסתברות .

\*להשלים סוף הרצאה 3 (15 דק אחרונות) וסוף תרגול 3 (בקובץ) הוכחת חסם תחתון על ההסתברות לטעות .