

SUMMARIZED BY NOAM KIMHI

noam.kimhi@mail.huji.ac.il

Introduction to COMMUNICATION NETWORKS

YEAR 3 SEMESTER A

COURSE NO. 67594

2025-2026



Course Summary

מרצה: יוסי גלעד

החומר המחייב הוא זה שנלמד בהרצאות וบทרגולים. סגל הקורס אינו אחראי לתוכן הסייעות או לנכונותו.

תוכן עניינים

3	הרצאה 1
3	הקדמה
4	רשת טלפון ני
7	הרצאה 2
7	Packet Switching
12	תרגול 1
12	חוור על הסתבות
17	הרצאה 3
17	מודל השכבות
21	תרגול 2
21	OSI Model
22	Random Access Protocols
23	ALOHA
28	שאלות תרגול
33	הרצאה 4
33	Random Access Protocols – ALOHA
35	Random Access Protocol – CSMA
38	Cתבות MAC
40	תרגול 3
40	Random Access Protocols
40	ALOHA – Poisson Approach
43	שאלות תרגול
50	הרצאה 5
50	Wi-Fi CSMA/CA ברשתות
52	Switch
56	STP (Spanning Tree Protocol)
60	תרגול 4
60	CSMA/CD (Carrier Sense Multiple Access/Collision Detection)
65	CSMA/CD – Questions
68	Errors Detection and Recovery

73	הרצאה 6
73	רשתות IP – IP Networks
79	פרוטוקול DHCP
82	תרגול 5
82	Spanning Tree
84	Spanning Tree Protocol
86	שאלות בנושא Spanning Tree Protocol

הרצאה 1

הקדמה

מצגת 1

מה Learned בקורס?

קונספטים של רשות, רעימות ועקרונות. נתעסק במורכבות שבתקשרות מגישות של פרטיות, אבטחת מידע, ביצועים וצדדים אחרים.

העלינו כדוגמה את נושא בעית טעית סרטוניים בראשת. דרך להתמודד עם הבעיה היא פרוטוקול שבו הלקוח מספר לשרת שהוא לא מצליח לטען את הסרטון וכן הלקוח יוכל לבקש מהשרת להוריד את איות הסרטון, זה נקרא (QoE) Quality of Experience).

אם הלקוח מנסה לקבל וידאו, השירות לפעמים לא יודע שהקלינט מסוגל להוריד מהר יותר. בשעה שלקוח עבר מקום, השירות צריך להתאים חזרה את קצב השידור שלו.

חלוקת לשכבות

אפיינו חלוקה ל-3 שכבות: אפליקציות, פרוטוקולי תקשורת, וטכנולוגיות חומרה.

יותר קל לבנות אפליקציות חדשות ולשנות חומרה (כמו שדרוגים לרואטורים) מאשר לשנות פרוטוקול תקשורת. לוקח זמן לפרוס פרוטוקולים חדשים, ושינוי פרוטוקולים חדשים (בוגמת השינוי בתקופה האחרונה מ-4-6 שנים ל-6-7 שנים) אוור שניות רבות, אבל שינויים קוררים. המטרה המרכזית היא להעביר מידע בין משתמשי קצה דרך רשת. אנחנו לא יכולים לבצע חיבור בין משתמשי קצה בצורה שהיא 1:1, אז יהיו צמתים שנינטו מידע והם מחוברים בלינקים (ראוטרים/סוויצ'רים). ניתן יכול להיות פייבר אופטי, wireless, בבל Ethernet ומשתמשי קצה יכולים להיות שרת, פלאפון נייד, מחשב נייד וכו'.

ברטיס רשת – נותן יכולת לנקודת קצה להתחבר לרשת.

שאוף 36

Jitter – האם יש שינוי ב-latency בין הודעות (ראשונה נמסרה ב-50ms, אחרת ב-70ms,...)

שאוף 37

מערכת מבוזרת – כמה מכשירי קצה שמחוברים לשרת.

רשות טלפון

שאוף 40

Circuit Switching

שני משתמשים רוצים לתקשר. השולח מקבל מסלול ברשות שМОקצת לשיחה שלו. הוא יתקשר עם צמתים ברשת, כל אחד יקצת לו משבאים ויאפשר לו להשתמש בlienק הבא. ייתכן מצב שאין משבאים פנויים ואז נקבל סטטוס "Busy". אם הצלחנו ליצור מעגל, השולח יוכל להתחיל לשלוח מסרים מעל המעגל הקיים אל משתמש הקצה. הוא לא צריך לכתוב لأن הוא הולך, איפה שהמעגל מסתיים – לשם התקשרות הולכת. בסיום יש פירוק של המעגל שעליו נוצר הקשה.

שאוף 42

בעבר היה חיבור אנושי לתקשורת טלפון.

שאוף 44-46

ב-Circuit Switching מתג = switch. כל ציר זו ישוט אחרת ברשות, ציר ע מתאר את הזמן. מתחילה בניסיון להקים מעגל. יש כמה סוג דילאי שהציגנו:

Propagation Delay – פער הזמן מהרגע שהבית הראשון של הודעה נשלח, ועד שהוא התקבל ביעד.

מדובר במאפיין של התווך שדרכו המידע זורם (פיבר, Ethernet).

ההודעה היא לא בית בודד, היא רצף של ביטים. לכן נדרש הגדרה של דילאי נוסף:

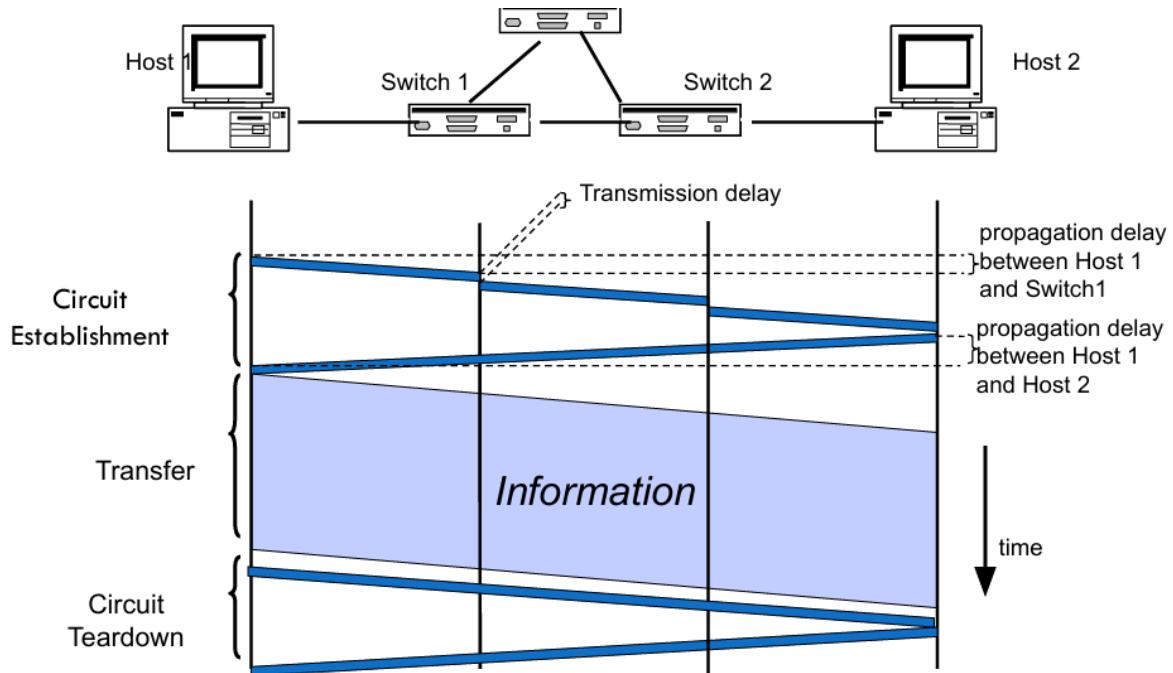
Transmission Delay – זמן שעבר מקבלת הבית הראשון ועד קבלת הבית האחרון.

זה מאפיין שישיר גם לתווך אבל גם לברטיס הרשות.

שאוף 47-50

ឧבשו 2 Host2 יכול לאשר ל-Host1 הודעה שהמעגל נוצר בהצלחה, ואז מידע עובר בחופשיות.

בסוף התקשרות צריך להרים את המעגל כדי לשחרר את משבבי הרשות, לספר להם שהפורטים למעשה בבר לא בשימוש.

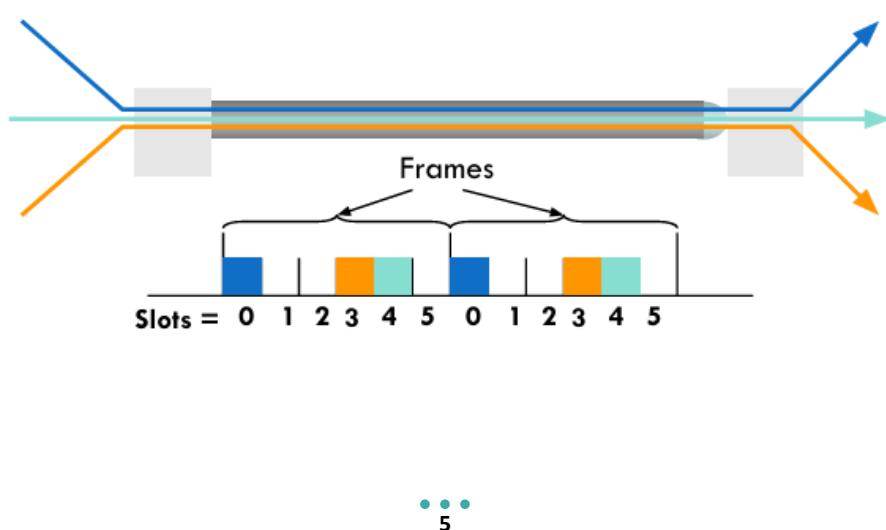


שאלה 51-53

יתכן שנרצה לחלק לינקים בין כמה שיחות. ישן כמה אפשרויות:

1. חלוקה בזמן – יש לינק, זמן השידור עליו רציף. נחלק את הזמן למקטעים. בכל מקטע ברשת קטע משיחה אחרת וכך נמשיך במעגל. איפה נמצאת השיחה האדומה בזמן שמקצת לשיחת הכתומה? נדרש סyncron בין כולם, או buffer שיחזיק את המידע.
2. להקצות מרוחבים שונים בתדר

ב-Time Slotting אם למשל יש 6 משתמשים אז כל אחד מקבל חתך בזמן (מודולו 6). חייב לסנכרון באופן המשילות, בעיקר לאור העובדה שאנו חnage לא בתבאים לא המידע ממעון. השיטה זו לא מתחשבת בכמות המידע של כל שיחה, יתכן שהשיחה הכחולה למשל עמוסה יותר ותתעכוב בלי סיבה מוצדקת בעקבות השימוש בשיטה זו.



יתרונות

- ברור במה slots שמורים לי, ובמה זמן ייקח להודיעו להגיע לצד השני.
- אין אובדן של מידע.
- ברגע שהקמתי מעגל השיחה תתקיים ללא תלות בעומס.

חסרונות

- **עמידות נמוכה בפני כשלים** – כשהקצו מעגל, חייב לעבור בו. אם לינק/צומת נופל בדרך, אין חלופה. הדרך היחידה לטפל בכך היא לבקש לסגור את המעגל הנוכחי ולהקם מעגל חדש.
- **בזבזני** – משלמים על סלוטים ומבדקים סלוטים שלא נעשו בהם שימוש מוצדק. למשל, אפליקציה שבשיא מגיעה ל-bandwidth בערך P , אבל בזמן מצהיר משדרת A, נהיה חיברים לשדרין עבורה P כדי לאפשר לה לشدד בזמן עומס, אבל בפועל נקבל ניצול נמוכה של המשאים שהוקטו עבורה, ביחס של $\frac{A}{P}$. אפליקציות קלות הן בדרך כלל ביחס $A:P$ של 1:3, אבל יש אפליקציות עם burst שmagiu לשיאים של למעלה מ-100. לסוגים כאלה של אפליקציות (עם קבוע שידור משתנים בKİצוניות), circuit switching לא מתאים.
- **התאמת לשמע** – את הסוג הזה של מיתוג בנו עבור רשתות טלפון ולבן יש אופטימיזציה ל-voice והגעה לפיקוד. זה נכון שיש מסלול קבוע לרשתות טלפון. באופן כללי, נרצה שהדרך שבה נעשה מיתוג בראשת יתמוך בהרבה סוג אפליקציות.
- **זמן setup גבוה** – נדרש לשמור משבבים עד לייצור מעגל, קבלת אישור וכדומה – מדובר ב-latency של תחילת התקשרות. קיימים מקרים בהם יש שאילותה קצרה, אבל במקרה זמן-h-*setup* הגבוה היא תלואה בתקופה גבוהה בשירות הנדרש (החלק העיקרי של התקשרות) הוא קצר מאוד.

הרצאה 2

מצגת 1
שקלף 62-66

אפשר לחלק תקשורת ל-2 סוגים:

1) Broadcast – כמה מחשבים על bus משותף, כולם יכולים וקוראים מאותו ערוץ תקשורת. אין דרך לשלוח למיisha הودעה מבלי שהיא יגיע לאחרים. Wi-Fi פועל באופן דומה. מה המוגבלות בשיטה זו? הגבלות טווח (יש עניין של קבוצה), בעיות בשייש כמה שירותי ליזום תקשורת באותו זמן על פני אותו bus, בעיות פרטיות – כולם מקבלים את כל ההודעות.

2) Switched – באמצעות ניתוב ברשת. דרך קבועה שהודעות יעברו בערות אד ולא באחרה. נתקלנו בו-. **Packet Switching** **circuit switching**

Packet Switching

שקלף 67-68

פקטה – פיסת מידע שנשלחת ברשת ומכליה תדר שמייע במיתוג – להביא את ההודעה ליעדה ולא לאחרים. מדובר בשינוי קונספטואלי-m-**circuit** ללא הכיל header מזהה.

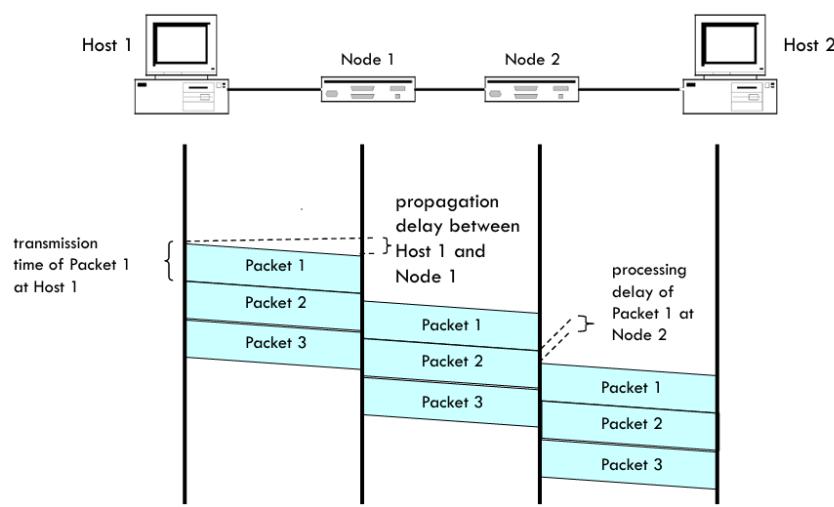
ב-Payload יימצא הדטה שאנחנו מעבירים, וב-Header הוראות לרשת איך לטפל ב-Payload. כל פקטה מתופל בנפרד על ידי הרשת. זה נכון כי לא צריך מראש להקצות משאבי ומסלול.

שקלף 70-72

באירו Host1 רוצח לשולח הودעה. אין אז זמן למזער propagation delay בין מדבר בזמן שבו לביטים לוקחים עברו לצד השני (בפיבר יהיה מעט, בלויין יהיה הרבה). כמו כן, קיימים עדין transmission delay על הזמן שלוקח לפקטה שלמה לעבור. השוני הוא שהוא עכשווי גם סוג דילאי נוסף.

זמן שלוקח לצומת בדרך לעבוד את חלקיה השונות של הפקטה. **Processing Delay**

בגלל צורת העטיפה של הפקטה, נדרש צומת שידע לעבוד את כל אחת מההודעות (לקריאה header וכדומה).



73-75

באיור A Host מעוניין לשלוח הודעה למשל ל-D Host. ב-circuit לא היה מותר לנו להשתמש בשני מסלולים שונים, אבל כאן מותר לבחור ביוטר מסלול אחד. זה עשוי ליצור בעיה שההודעות לא יגיעו לעדן בסדר הנכון ונctrar להתמודד עם זה.

בחירת פקטה

כמה ארכובה פקטה צריכה להיות? קטנה יחסית, כדי שאובדן של פקטה בודדת לא יהיה מאוד קריטי. לפעמים בראשת גם צריכים לקבל פקטה שלמה כדי להתחיל לבצע עיבוד, ואם הפקטה תהיה ארוכה מאוד יהיה "זמן מת" שבו מחכים לביטים שבסוף הפקטה – אולי כל פקטה הייתה גדולה היה לנו יותר delay transmission. מצד שני, הפקטה לא יכולה להיות ממש קטנה כי לכל payload נוסף גם header ועוד אם ה-payload קטן נקבל תקווה גבוהה על עיבוד headers.

76

בעובדה בפקטות לא שומרים על סЛОטים קבועים ל-flow מסוימים. אם אחד מה-hosts שולח הרבה הום פשוט יתתקבלו ב-burst בצד השני. יש בשיטה זו יכולת טובה יותר להתמודד עם bursts כי המצב דינמי ולא קבוע מראש.

מה קורה בשימושו שלוח הרבה מידע והוא מגע לצומת הבא?

אופציה 1: הצמת מספיק מהיר כדי לנטר את כל המידע לצומת הבא.

אופציה 2: יש עומס בנקודת מסויימת, אין ברירה וצריך להעביר את התקשות דרך אותו צומת. מותר לזרוק חלק מהפקטות ובתקופה השולח בין חלק מהמידע חסר ושלח אותו מחדש אם המידע שאבד היה חשוב. דרך טובה יותר היא הוספה buffer. אם יש עומס רגעי ההודעות יכתבו לתוך buffer ובשה-burst נגמר העודף ישתחר. חשוב לציין ש-buffer לא פותר את הבעיה לחלווטין, כי לכל buffer שנבחר תמיד יהיה burst שיגרם לו להתמלא ונאלץ לזרוק ההודעות. כמו כן, buffer יוצר בעיות נוספות נספפת כאשר נשלוח הודעות נשמרו בתוכו, למחרת שהצטמן שלහן כבר לא רלוונטי יותר ל-flow הנוכחי. זה נקרא **queuing delay**, כי ההודעה נתקעה מאחוריו תור ארוך.

84-86

אפשר לחלק frames ל-slots. נניח שיש flow של תקשורת עם שיא P וממוצע A . אם יש רק flow אחד נדרש להקצות (لتפס) P סLOTS, כי ה-buffer צריך להיות מסוגל לספוג מידע בגודל P . הבעיה היא שבה-peak יותר גודל מהממוצע, וזה מה שיקבע עלויות בספיות.

לעומת זאת, אם מתנסבים על הרובה s ows flow במקביל, אפשר להרוויח **מחוק המספרים הגדולים**. נדמיין מצב עם הרובה s ows flow שמתקשרים דרך אותו צומת וקצב בין 0 ל- P עם אותו ממוצע. אם כולם בלתי תלויים, אז סביר

להניח שלא בולם יגעו לשיא באותו נקודת זמן – זה מאפשר לנו לחסוך בגודל ה-buffer. בגדול, קיבל ערך שקרוב יותר לממוצע בכל שיש יותר flows, ועוד הדיק יהיה טוב יותר.

אם צריך לתקן צומת בראשת, נקצת קצת יותר מהממוצע $A \times N$, וזה נעשה מדויק יותר ככל ש- N (מספר ה-flows) גדול.

השוואות נוספות ל-[Packet Switching](#)

87-89

יתרונות של Circuit Switching

- הבטחה על bandwidth שנקלע
- ביצועים שנייתן לחזות
- אבטסרקציה פשוטה להבנה למי שבוטב את האפליקציה מעל הרשת אין בעית תזמון להודעות (הודעות יגעו לפי סדר שליחתן)
- Forwarding פשוט יותר עברו מי שמכין את ציוד התקשרות.
- בשנוץ מעגל הוא מקצה מקום – אין parsing וain buffers.
- אין header, כלומר תקורה נמוכהיחסית על ביטים.

חסרונות של Circuit Switching

- בזבוז של bandwidth – הקצינו מסלול עבור מישחו והוא לא תמיד יונצל.
- תעבורה ב-burst תזרוש הקצתת מקום לפי ה-peak.
- לא תמיד נוכל לאפשר תקשורת "סבירה" לאחד ו"טובה" לאחר.
- צריך לשמור מצב, זה גורם לבעה ולනפילות.
- בשנות נופל – כל מי שעבר דרכו בבעיה.
- תקלות לוקחות זמן ליזיהו ולתיקון (יצור מסלול חדש).
- יש setup delay שיכול להיות גבוה (מפרע בעיקר בשעה קרצה).

Packet vs. Circuit

אתם צמתים בראשת לא שומרים מידע לטווח ארוך על מה שעובר דרכם – אם צומת נופל הכל הוא לכל החיבור ב-[Packet Switching](#).

מבחינת יעילות משאבי – מנצלים ב-[Packet](#) נתונים סטטיסטיים כדי לתת ביצוע טוב יותר בצויה חסכונית.

מבחינת deploy – בזמנים שאנו מחוברים אליו нам צריך להבטיח דבר על האיכות מראש.

מבחינת Packet – ניתקל במצבים שלא יכולים להגיע אליהם לפניה, כמו עומס על צומת ספציפי, ובעומס

זה צריך לטפל. בעיות של חסור סדר מקצה לקצה (הודעה לא תמיד תגיע בסדר שבו נשלחה). בغالל ההקצתה

הдинמית הכל תלוי באיך אחרים משתמשים בראשת, לעומת [circuit](#) שבה יש בידוד מאחרים.

מצגת 2
שקלף 2

מטרת ה

על
 בקורס היא לבנות רשת גדולה ומורכבת. שיטת המיתוג היא עקרון בסיסי בראשת. צריך לבנות באופן מודולרי – כמו רכיבים. כל אחד יתן שירותים לרכיבים אחרים. זה שנטנן ורכיבים שונים למשימות שונות נס庭ר מורכבות, וגם יכול לשנות את המימוש הפנימי כל עוד נשמר על אותו ממשק.

שקלף 4

נחלק לשכבות – כל אחת תספק שירותים לשכבה שלמעלה. מוחות בתכנון. הבעה – יכול למנוע מאיתנו לבצע אופטימיזציות שהיינו יכולים לבצע אילו שבתנו את מודל השכבות. אבל, במערכת מורכבת כמו האינטרנט – זה ה-*tradeoff* שצורך לבצע.

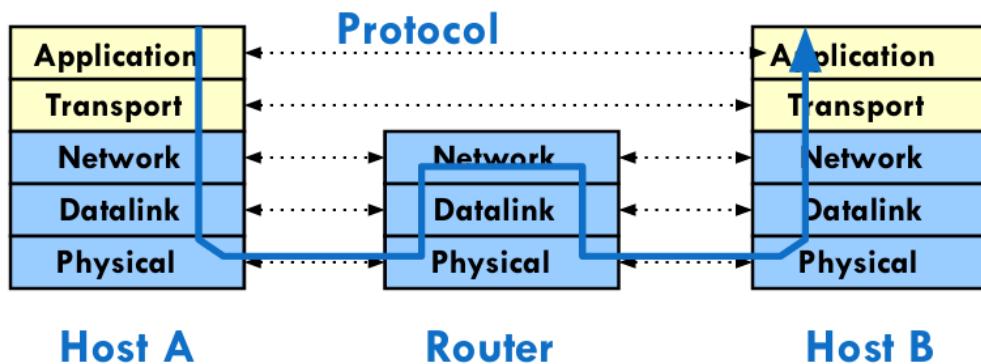
שקלף 7

Layering

אילו שימוש יש לנו? **מטרת ה

על** – העברת מידע שימושי בין משתמשי קצה. היכי בסיסי – העברת אלקטטרונים על ערוֹץ תקשורת, לפרש אותם כבאים, אחר כך כפקודות, ואוּתן נרצה להעביר לנקודות קרובות יחסית בראשת לוקאלית (LAN). בהמשך, העברת פקודות בין LANs וידוא שהמידע הגיע לצד השני (לטפל באובדן) ומטרת ה

על
 – ביצוע שימוש במידע.



שקלף 8

את המשימות שמנינו נחלק לשכבות שונות, כל אחת אחראית על 2-1 משימות. לכל שכבה הולך להיות ממושך לשכבה שלמעלה, ולא מסתכלים על איך השכבה הקודמת מומשה.

שקלף 13

איך זה עובד בתקשורת מחשבים? 2 אפליקציות של Hosts שונים מנוטות לתקשורת האפליקציה וה-transport. ממוממשות רק בקצוות, השאר גם בציר שבדרכן. כל שכבה מתקשרת עם השכבה המקבילה לה מצד השני. מבחינתם לא מעניין מה קורה במרכז. ב-*Transport* יש בקרת עומסים (לדעת אם הודעות הגיעו, לא הגיעו, להעלות קצב, להוריד..) כל שאר הדברים לא מעניינים אותה.

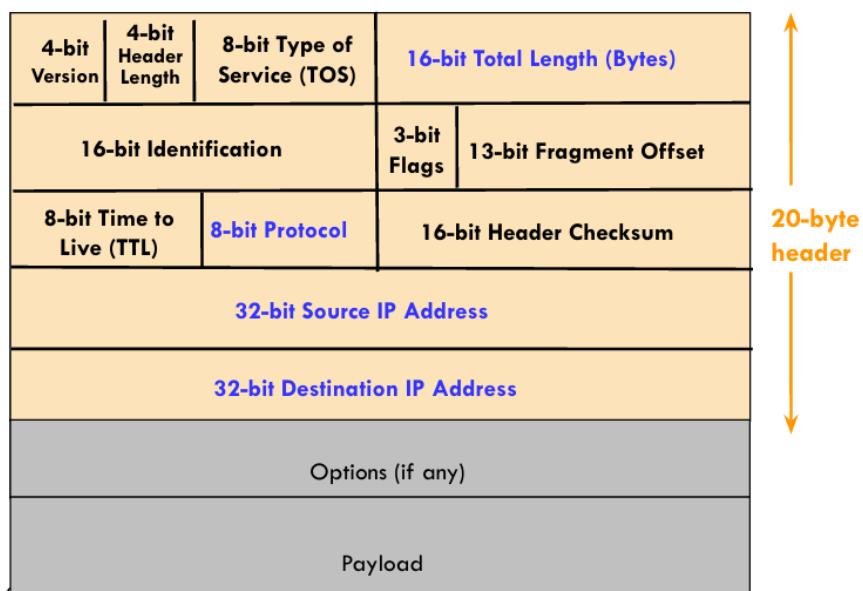
שקלף 14-17

התקשורת מתבצעת באמצעות פרוטוקול:

יש לפרטוקול צד סינטקטי – א' ביטים הראשונים מתייחסים לו.
סמנטי – מה עושים עם זה? שכבת Transport אותה על בעיה, מה עושים אליה זו סמנטיקה. יש פרוטוקולים
 שונים להתחמזרות, למשל הורדת קצב האיתות ובדומה.

שקלף 18
 אי אפשר לתקשר בדרך שאינה הפרוטוקול – יש מוסכמות.

שקלף 19
 מבנה פקעת IP (פרוטוקול של שכבת ה-Network-):



החלק הכתום הוא ה-header. בשראטור מקבל הודעה IP הראווט יכניס אותה למעין struct. המימוש של IP
 יסתכל על בתובת היעד וינסה להבין לאן לנתק אותה.

החלק האפור הוא מידע אופציוני שי יכול להוסיף ל-header, transport header, במשמעותם את החבילה הלאה
 הטעון שלה לא מעניין בכלל.

המטרה של שכבת ה-network זה להעביר את ה-payload שלה, של שכבת ה-link זה להעביר את
 שלה... וכך הלאה.

שקלף 21
 בغال שיש לנו ציוד תקשורת שמיוצר על ידי גופים שונים, ואפליקציות שנכתבות על ידי גופים שונים – חשוב
 לשומר על סטנדרט מסוים שיעזר לכולם לשומר על דיבור ב"אותה שפה". IETF – גוף תקינה שאחראי לייצר
 את הסטנדרטים הללו. הסטנדרטים מפורטים בתוכן (RFCs) (Request for Comments).

תרגול 1

חזרה על הסתברות

התרגול עוסק בחזרה על קונספטים חשובים מהסתברות שיישמשו אותנו לאורק הקורס, ולא חומר תיאורתי ברשות תקשורת. לאורק הקורס נשתמש בהסתברות לפתרת בעיות שונות. דוגמה פשוטה לכך היא הסתברות לשילוח פקטה או יעילות.

תכונות חשובות (תכונות נוספות במצגת)

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} \Leftrightarrow \mathbb{P}(A \cap B) = \mathbb{P}(A|B) \cdot \mathbb{P}(B)$$

$$\mathbb{P}(A|B) = \mathbb{P}(A) \stackrel{\text{מאורעות זרים}}{\Rightarrow} \mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B)$$

(אם המאורעות זרים, אפשר להחליף את \leq ב- $=$)

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(B|A) \cdot \mathbb{P}(A)}{\mathbb{P}(B)}$$

באשר $\Omega = \bigcup_{i=1}^n B_i$ נוכל להפעיל הסתברות שלמה ולקבל:

$$\mathbb{P}(A) = \sum_{i=1}^n \mathbb{P}(A|B_i) \cdot \mathbb{P}(B_i)$$

מניחים שפקטה "טובה" אם היא שורדת ברשת לפחות T שניות בהסתברות e^{-T} . פקטה "לא טובה" אם היא שורדת ברשת לפחות T שניות בהסתברות e^{-1000t} .

שאלות

1. נניח שפקטה בודדת נוצרה בזמן $t = 0$, מה ההסתברות שהיא עדין קיימת ברשת בזמן $t = ?$

פתרון:

נגיד מאורעות:

$A = \text{a packet survived for } t \text{ seconds}$; $B = \text{a good packet was created}$

$$\mathbb{P}(A) = \mathbb{P}(A|B) \cdot \mathbb{P}(B) + \mathbb{P}(A|\bar{B}) \cdot \mathbb{P}(\bar{B}) = e^{-t} \cdot p + e^{-1000t} \cdot (1 - p)$$

2. ידוע לנו שפקטה שורדה ברשת לפחות $t = T$ שניות. מהי ההסתברות שזו הייתה פקטה טובה?

פתרון:

נפעיל את כלל ביסוס ונקבל:

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(A|B) \cdot \mathbb{P}(B)}{\mathbb{P}(A)} = \frac{e^{-t} \cdot p}{e^{-t} \cdot p + e^{-1000t} \cdot (1-p)}$$

תוחלת ושונות

$$\mathbb{E}(X) = \sum_i x_i \cdot \mathbb{P}(X = x_i)$$

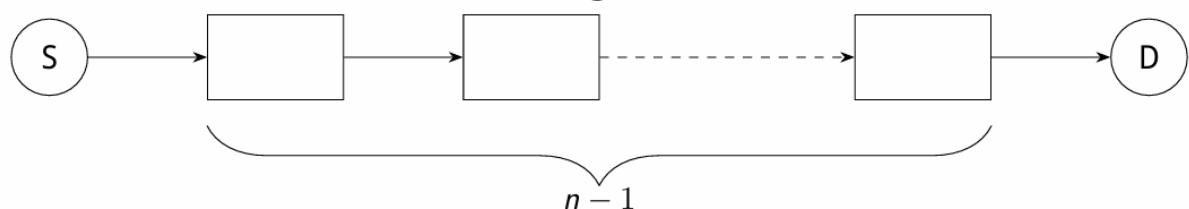
$$\mathbb{E}\left(\sum_i a_i \cdot x_i\right) = \sum_i a_i \cdot \mathbb{E}(x_i)$$

$$\text{Var}(X) = \mathbb{E}(X - \mathbb{E}(X^2)) = \mathbb{E}(X^2) - \mathbb{E}(X)^2$$

	Bernoulli	Binomial	Geometric	Poisson
Intuition	We make an experiment, and we could either fail or succeed	We make n independent Bernoulli experiments. We mark by X the sum of successes	We do independent Bernoulli experiments until we succeed. We denote by X the number of tries	Counting the number of events that occurred during some period of time
Probability mass function	$P(X = 1) = p, P(X = 0) = 1 - p$	$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$	$P(X = k) = (1 - p)^{k-1} p$	$P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}$
Notation	$X \sim Ber(p)$	$X \sim Bin(n, p)$	$X \sim Geo(p)$	$X \sim Pois(\lambda)$
Exp.	$\mathbb{E}(X) = p$	$\mathbb{E}(X) = np$	$\mathbb{E}(X) = \frac{1}{p}$	$\mathbb{E}(X) = \lambda$
Var.	$Var(x) = p(1 - p)$	$Var(x) = np(1 - p)$	$Var(x) = \frac{1-p}{p^2}$	$Var(x) = \lambda$

תפורה לשאלות

- We want to send a message from node **S** to node **D** which are connected by a chain of **n** links.
- The probability of any link to fail is p (independently).
- At time $T = 0$, **S** sends a message to **D**



שאלות

1. מה ההסתברות שהצומת D קיבלת את ההודעה?

פתרון:

מדובר בהסתברות שכל אחד מהצמתים הצליח להעביר את ההודעה. יש n צמתים, כל אחד יכול להיבש בהסתברות p בলומר להצלחה בהסתברות (המשלימה) של $p - 1$. סך הכל לאחר שהמאורעות בלתי תלויים אפשר לבסס אותם ונקבל שההסתברות שההודעה תגיע לעד היא $(p - 1)^n$.

2. הינו שאם ההודעה לא מגיעה ל-D, אז S תשלח אותה שוב ושוב עד להצלחה. מה תוחלת הפקודות שישילחו עד שפקטה תגיע לעד?

פתרון:

נגידר את X להיות "מספר הפקודות שיש לשלוח מ-S עד שפקטה מגיעה ל-D בהצלחה". נבחן ש- X מתפלג גיאומטרית $(p - 1)^n$ כי כל פקטה אינה תלולה באחרות ולכן לפי תוחלת של מ"מ המתפלג גיאומטרית נקבל:

$$\mathbb{E}(X) = \frac{1}{(1 - p)^n}$$

3. העשוו, נניח שכל צומת ממשיר לשלוח ללא הפסקה את ההודעה לצומת הבא עד שההודעה מתתקבלת. הינו שザומת הראשונית (S) שולח פעם אחת בלבד. מה ההסתברות שההודעה תגיע ל-D?

פתרון:

אחר שכל צומת פרט ל-S שולחת את ההודעה עד שהיא תתקבל, ההודעה תתקדם לצומת הבא בהסתברות 1 (תמיד). התלות היחידה היא בזומת S, אשר מצליחה בהסתברות $p - 1$, וכן זו התשובה.

4. מה תוחלת מספר הפקודות שצומת בודד ישלח עד להצלחה?

פתרון:

נגידר מ"מ X שמתפלג גיאומטרית $(p - 1)^n$ Geometric וכאן לפי הנוסחה:

$$\mathbb{E}(X) = \frac{1}{1 - p}$$

משתנים רציפים

$$\mathbb{P}(a \leq x \leq b) = \int_a^b f(x)dx \quad ; \quad \int_{-\infty}^{\infty} f(x)dx = 1$$

$$\mathbb{E}(X) = \int_{-\infty}^{\infty} x \cdot f(x)dx \quad ; \quad \text{Var}(X) = \int_{-\infty}^{\infty} (x - \mathbb{E}(X))^2 f(x)dx$$

	Uniform	Exponential	Gaussian (Normal)
Notation	$X \sim Uni(a, b)$	$X \sim Exp(\lambda)$	$X \sim N(\mu, \sigma)$
probability distribution function	$f(x) = \begin{cases} \frac{1}{b-a} & x \in [a, b] \\ 0 & else \end{cases}$	$f(x) = \begin{cases} \lambda e^{-\lambda x} & x > 0 \\ 0 & else \end{cases}$	$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
Exp.	$\mathbb{E}(X) = \frac{a+b}{2}$	$\mathbb{E}(X) = \frac{1}{\lambda}$	$\mathbb{E}(X) = \mu$
Var.	$Var(X) = \frac{(b-a)^2}{12}$	$Var(X) = \frac{1}{\lambda^2}$	$Var(X) = \sigma$

תכונת חוסר זיכרון

למשתנים מקרים גיאומטריים וקספוננציאליים יש תכונת חוסר זיכרון:

$$\mathbb{P}(X > s + t | x > t) = \mathbb{P}(X > s)$$

כלומר לא משנה "כמה פעמים ניסינו".

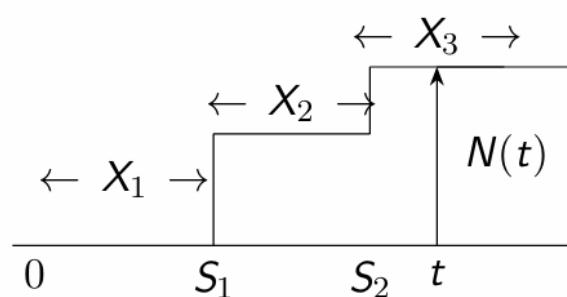
תהליך רנדומלי סטוכסטי

הטלת מטבע a פעמים זה תהליך סטוכסטי של אירועים ברנוליים. בזמן t אפשר להסתכל על זה ולהגיד שזה מתפלג כמו הסתברות בינומית. שימושי למשל עבור נתבים שמקבלים פקטה ולא ישר שלוחים אותה הלאה אלא מחכים כמויות זמן רנדומלית (מסיבות שנראה בהמשך). מבחינה ויזואלית:

Define

- $S_1 = X_1$
- $S_2 = S_1 + X_2$
- \vdots
- $S_i = S_{i-1} + X_i$

S_i is called an "arrival epoch" (the specific time at which an event occurs) $N(t) = n$ for $S_n \leq t < S_{n+1}$



תהליכי ספירה

תהליכי ספירה זה תהליכי רנדומליים שסופר את מספר ה-events עד נקודת זמן t . הוא מוגדר לפי $\{N_t | t \geq 0\}$ כאשר:

$$N_0 = 0$$

$$N_t \geq 0$$

$s \leq t \implies N_s \leq N_t$ (monotonicity) and if $s < t$ then $N_t - N_s$ is the number of events that occurred in $(s, t]$

דוגמה: מספר הלהקות שפגיעים לחנות.

תהליכי פואסוני

תהליכי פואסוני עם קבוע λ הוא תהליכי ספירה אבל בנוסף מקיים מספר תכונות:

- Independent increments: Number of events in any two disjoint intervals is independent
- Stationary increments: The number of events in any interval of length t is a Poisson random variable with parameter λt (depends on the interval's **length** and not on its timing). ie. $\mathbb{E}[N_t] = \lambda t$
- No bunching: The probability of > 1 arrivals in a tiny interval is negligible

האם מספר האנשיים שפגיעים לתחנת אוטובוס הוא תהליכי פואסוני? **לא**. כי בנקודת זמן אחת יכולים לעלות כמה אנשים לאוטובוס, ובשעות מאוחרות משמעותית פחות אנשים יעלו.

שאלה

בהתנחת טווח T שלוחים בו פקודות בהתקלגות פואסונית עם פרמטר $gT - gT$ זו ההסתברות שנשלח פקטה, T זה משך הזמן. מה ההסתברות שנשלח פקטה אחת?

פתרון:

תהליכי פואסוני מתפלג באופן הבא:

$$X_p \sim \text{Poi}(gt) \quad \text{and} \quad \mathbb{P}_{t=T}(X = i) = \frac{(gt)^i}{i!} e^{-gt}$$

$$\mathbb{P}_{t=T}(X = 1) = gt \cdot e^{-gt}$$

מודל השכבות

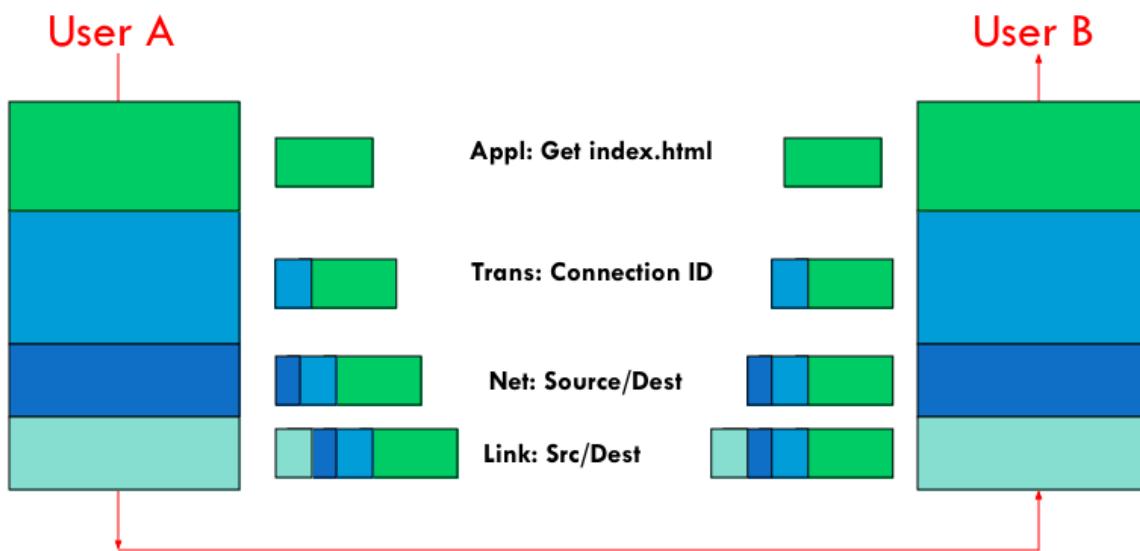
מצגת 2
שקי 13

דוגמה לIMPLEMENTATION של שכבות שונות. למשל יש שטחן מימוש של host client https וו הוא רוצה לדבר עם web server כדי לבקש עמוד web. מה שהדף (הAPPLICATION) עוזה זה לכתוב TCP. יוצר חיבור TCP בין הדף לשרת web.

כל שני צמחיים שכנים צריכים לתקשר באותה דרך, למשל Ethernet ל-Ethernet, אופטי לאופטי. ביל שכבה נוספת header משלמה.

שקי 23

כאן חאים שככל שכבה נוספת מוסיפה את ה-header שלה, נפוץ שורך כל ה-headers הוא 54 בתים. שרת ה-web למשל רואה רק אתappl: Get index.html, אבל ברטיס הרשות קורא שכבות נוספות.



Common case: 20 bytes TCP header + 20 bytes IP header + 14 bytes Ethernet header = 54 bytes overhead

שקי 24

כל שכבה מותנת שירות מסוים ו-interface לשכבה שמעליה לקרוא לה. פרוטוקול זו הדרך שבה אנחנו מתקשרים בין מופעים קרובים זה לזה של אותה שכבה. כך אפשר למשתמש בדרכים שונות בכל שכבה מבלי להשפיע על שכבות אחרות, כל עוד נצמדים לממשק זהה.

שאוף 29

השכבה הפיזית/Physical Layer – אחראית על העברת נתונים על אוט **wire**. פרוטוקולים אומרים איך ליצג את אותו בית, ויש פרוטוקולים שונים לסוגים שונים של שכבות פיזיות.

שכבה הLINK/Data Link Layer – מטרתה להעביר הודעות שנקרו להן **frames**. אלו כתובות לوكליות של מכשי רשת (כתובות ברשת הלוקאלית) בולם מוכרת רק בתוך הארגון ולא זמינות לפני חוץ. הפרוטוקול אחראי על (MAC) **Media Access Control** – נרחב על בר בהמשך.

שכבה הרשת/Network Layer (Inter) – תפקידיה להעביר מידע בין רשתות על פניו כמו **bus**-ים. כדי להעביר את ההוועה כל מכשיר רשת בדרך השתמש שוב בשכבה הLINK. מבחינת Network והלאה שלוחים תאות דבר, אבל ה-frame בכל פעם מעט שונה (**payload** זהה). ה-source הופך להיות הראטור הנוכחי והוא destination הцומת הבא (ראטור נוסף או היעד). למה אין כמה סוג פרוטוקול רשת? מדובר רק IPv4 (ובהמשך שנים מנסים להחליף ל-IPv6)?

Transport Layer – בתקשרות בין תהליכיים בתוך מחשב, בין אפליקציות אותה מערכת. השכבה מוסיפה מזהה להודעה ב-**scope** של אותו מכשיר (שלוח/מקבל), TCP ו-UDP, אבטחות מסוימת לקצה, התואששות מההודעות שהלכו לאיבוד, לדאוג שלא נשלח יותר ממה שה מקבל יודע לקבל או שהרשות מצילה להעביר. אבטורקציה **stream** של מידע. לא תמיד בתוך המפתח קל להעביר 1000 בתים אבל ואז 1000 בתים אחרים, נדרש מעטפת (ממשק **read/write**). כותבים לערך **buffer** ואז שכבת **transport** תחלק אותו לפקודות ותשלח, וכן גם מצד מקבל.

שכבת האפליקציה/Application Layer – כל שירות משתמש הקצה, והמשק הוא תלוי אפליקציה. דוגמאות לשכבה זו הן כל קטע קוד שמשתמש בשכבה **transport**, כולל כל אפליקציה שמשתמשה ברשת.

שאוף 30

ראים מיושמים שונים לשכבות שונות. שכבת ה-**Network** אין לה מגוון פרוטוקולים, מדוע? כי היא אחראית על אינטראקציה בין רשתות, בולם צריכים לדעת לדבר באותה שפה. מנסים כבר 20 שנה להחליף את IPv4 ל-IPv6 – זה קורה, אבל לאט מאד. כל פרוטוקול אחר מערב הרבה פחות גורמים בלתי תלויים, למשל **data link** זה מערב רק את הצומת הנוכחי והבא בתוור. מעבר בפרוטוקול **Network** קשה יותר – זה שהמחשב שלו ידבר בשפה IPv6 לא יעזר בלי שהראטור יודע לנtab אליו למקום הנכון.

שאוף 31-32

אתגר – שליחת הודעות מעלה רשת מקומית. נתחיל משליחת הודעות מעלה ערך אחד. אנחנו לא מדבר הרובה על שכבת האפליקציה ועל השכבה הפיזית. השיעור הזה נתמך ב-**Data Link**.

שאוף 33

במה מחשבים שמחוברים לכבל אחד ארוך.

שאוף 36

בשכבה 2 – ההודעות נקבעות על header link ו-link payload של frame.

שאוף 37-38

כל רואוטר מחובר ב망שך לרואוטר הבא בלינק. ברגע נתמךד בלינק בודד. רוצים לתאם שליחת על הילינק המשותף בין כל המחשבים שיש לו ברשות. לצורך כך נדרש MAC Address – כתובות בשכבה 2 שנקבעת על ידי ברטיס הרשות. לכל ברטיס יש כתובות MAC שנצרכה עליו, והכתובות הללו רלוונטיות רק לרשות המקומית. מה נרצה להשיג? נרצה לדאוג שפרויימים יישלחו כמו שצריך, ותחילה נרצה להימנע משגיאות. שגיאות עלולות להתרחש אם שני מחשבים שולחים מידע באותו הזמן, צריך לעת להימנע מקרים כאלה או לדעת לטפל בהם אם יקרה.

שאוף 39

השכבה ממומשת בברטיס הרשות שלהם. השכבות שמעל ממומשות ב-OS או באפליקציות שהיא מರיצה ומשתמשות ב-CPUs. את העבודה של שכבה 2 (שכבת הילינק) נבצע בברטיס הרשות שלהם. הוא יבצע הוספה header של הדאטעה ושליחת של ה-frame. אחריו על תקשורת מול ה-OS.

שאוף 40

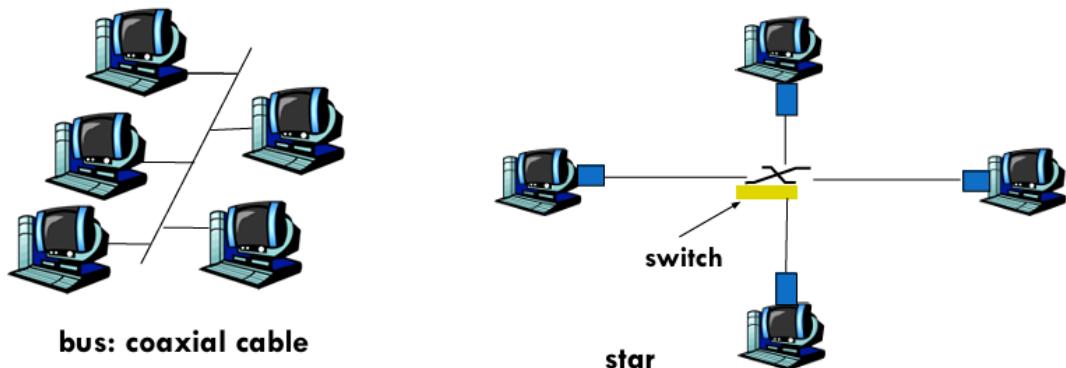
בצד המקבל קוראים את הפרויימים אחד אחרי השני באמצעות parsing של הפרויימים ובחינת כתובות. אם מופיעו אליו, עברו להלאה. הברטיס יכול לבדוק גם אם corruption במאזעות checksum – סכימת בתים ובדיקה האם במות הבטים שווה למה שמופיע בשדה checksum של ההודעה שהתקבלת. אם מספר הבטים לא תואם, ההודעה תיזרק.

שאוף 41

נכיה שיש ערוץ אחד והרבה שלוחים. איך נתאים את השילוחה ביניהם? שליחת מקבילית תהרוס את השידור. הקושי הוא שאין ערוץ נוסף למס' לתאם את התקשרות, הערוץ היחיד הוא זה שמעבריהם עליו את ההודעות. היינו רוצים פרוטוקול שיבטיח לנו כמה הודעות נצליח לשלוח בלי שתהיה התנגשות.

שקי 42

ברגע אנחנו מתעסקים ב-coaxial cable, אבל בהמשך נסתכל גם על switch שם יש מיתוג וכן נקבל פחות התנגשיות: מתאפשר מקובל בזכות ה-switch שדווג שهواتו שונות לא תשלבנה על אותו הקו באותו הזמן.



שקי 43

ב-wireless בנויגוד ל-wired הsenal הולך ודווער בתלות במרחב ומכשולים גיאוגרפיים.

שקי 44

למה אנחנו יכולים לקוות? נניח שיש לנו ערוץ עם capacity של R ביטים בשנייה. היינו רוצים שם יש לנו רק שלוח אחד אז הוא יכול לנצל את כל ה- R -capacity שלו ולשלוח בקצב R . אם יש יותר משולח אחד, למשל m שלוחים – אז שבל אחד יוכל לקבל $\frac{R}{m}$. השאייפה היא שזה יקרה בלי גורם שייהי אחראי על תיאום אלא באמצעות אלגוריתם לוקאלי שיגרום לכך שלא נקבל התנגשיות עם אחרים. כמובן, רצוי אלגוריתם פשוט – כי עליינו למשמש אותו פיזית בחומרה של ברטיס הרשות.

שקי 48

בשיעור הבא נדבר על משפחת פרוטוקולים Random Access Protocols שמשתמשים בהטלה מטבע בפרוטוקול שמטרתו להמודד עם התנגשיות.

תרגום 2

OSI Model

מודול(OSI) של 7 או 5 שכבות (תליי בצורת חלוקה) שמתאר מה התפקיד של כל שכבה, והתפקידים בכל שכבה בנפרד. מטרת המודל היא שמחשב אחד יתקשר עם השני, תוך אפשרות להטעם מהאחריות של בית אחד בתהליך (מידה מסוימת של אבסטרקציה). נניח שהשכבה התחתונה עובדת, ומושפעים ממנה.

#	Layer Name	Data Unit	Function
1	Physical	Bit	Transmit and receive raw bits over some physical medium (air, optics, etc.)
2	Data link	Frame	Transmission of data frames between two nodes
3	Network	Packet	Structure and manage multi-node network (Addressing, routing, etc.)
4	Transport	Segment	Transmission (reliable or not) of segments between points in the network (ack'ing, etc.)
5	Session	Data	Manage a continuous communication session
6	Presentation		Translation of data between a networking and an application (e.g., encryption/decryption)
7	Application		High level APIs between network applications (e.g., GET request)

(אנחנו בקורס נתעסק פחות בשכבה הפיזית קשורה יותר להנדסה ולפיזיקה).

נתחילMSC 2, **data link** שאומرت שיש כמה מחוברים ברשת אחת, אם מישו מדובר – בולם יודעים. השפה שידברו כדי לאפשר לכמה שיטור מידע לעבר נקרא פרוטוקול. מעלה יש את שכבת ה-**Network**. הם לא מחוברים ביחד ישר, אבל יש להם כתובות אחד של השני והם מעבירים הודעות דרך נתבים בכך. שכבת **transport** מסתכלת בראיה וחברה – לא תמיד ההודעות יגיעו כמו שצריך. בשכבת הדאטה לינק הנחנו שם מישו משדר ואין בעיות בפרוטוקול אז זה יגיע לצד השני. בפועל המצב אחר, הטיפול באבדות הוא באחריות שכבה זו. שכבות 5-7 מסתכלות על המידע בולו (לא בחלוקת לפקטה), ב-**session** יש לנו דוח-שיח מול אחר כמו למשל פייסבוק. ברגע שהתחילה session אנחנו יכולים להיבנים לכל האפשרויות בתוך פייסבוק מוביל לאמת בכל פעם מחדש את התקשרות בינינו. כדי שזה יקרה, יש ענייני הצפנה בהם נגע בהמשך. ב-**presentation** מתייחס להצפנה של הودעות כדי למנוע דליפת מידע בהאזנות, ובסוף שכבת אפליקציה שהוא הממשק בין הרשות לאפליקציות של המשתמשים.

דוגמאות של פרוטוקולים לשכבות שונות

שכבה האפליקציה: DNS, HTTP

שכבה תובלת: TCP, UDP :Transport

שכבה רשת: IP :Network

שכבה נתונים: MAC :Data Link

שכבה פיזית: Wi-Fi

Random Access Protocols

נניח שאנו בשכבה 2 (Data Link) וכולם מחוברים באותו Network. אני רוצה להתחיל לדבר עם אנשים, כਮון לא ארצה שיבזרו באותו זמן כמוי אחרתי אף צד לא בין את השני. יש כמה הנחות:

1. בתוך זהה אין רעש, כלומר כל עוד אני היחיד שմדבר – כולם שומעים.
2. אם שני אנשים מדברים באותו זמן – יש שיבוש בתטעות ההודעות.



הגדרות

1. $\frac{bit}{sec}$ – כמה ביטים אפשר לשלוח ביחידת זמן בלבד. סימן: B . יחידות מידע:

2. Throughput – החלק היחסית של bandwidth שנוצל לשילוח הودעות.

3. Goodput – החלק היחסית של bandwidth שנוצל לשילוח הודעות שהתקבלו בהצלחה.

$$\text{סומן ויחסב לפיה: } \eta = \frac{\text{total average effective bandwidth usage}}{\text{total bandwidth}} = \frac{\text{total average effective time}}{\text{total time}}$$

(נבחן כי $0 \leq \eta \leq 1$)

שאלה

אם יש לנו פקטה בגודל $30\ bits$, כאשר ה- $bandwidth$ הוא $5\ \frac{bit}{sec}$ נקבל שהזמן להעביר את הפקטה הוא:

$$T = \frac{|packet|}{B} = \frac{30}{5} \cdot \frac{bit}{bit/sec} = 6\ sec$$

לשים לב, יש חשיבות גם לחלוקת היחידות ולא רק לערבים – זה גם מנגנון לוודא שביצענו חלוקה נכונה (אכן קיבלנו שה היחידות הן שניות באשר נשאלנו במה זמן).

עכשו נניח שאנו רצים לשלוח 3 פקודות שאגודל כל אחת *bits* 30 בתוך 2 שניות, מה רוחב הפס הנדרש?

$$B = \frac{\#packets \cdot |packet|}{T} = \frac{3 \cdot 30}{2} \cdot \frac{bit}{sec} = 45 \frac{bit}{sec}$$

ALOHA

General Model

ALOHA זה פרוטוקול שמנסה למצוא מצב ביןים שבו הודעות בן יעברו בין אחד לשני, אולי לא כל הזמן, אבל לפחות חלק מהזמן.

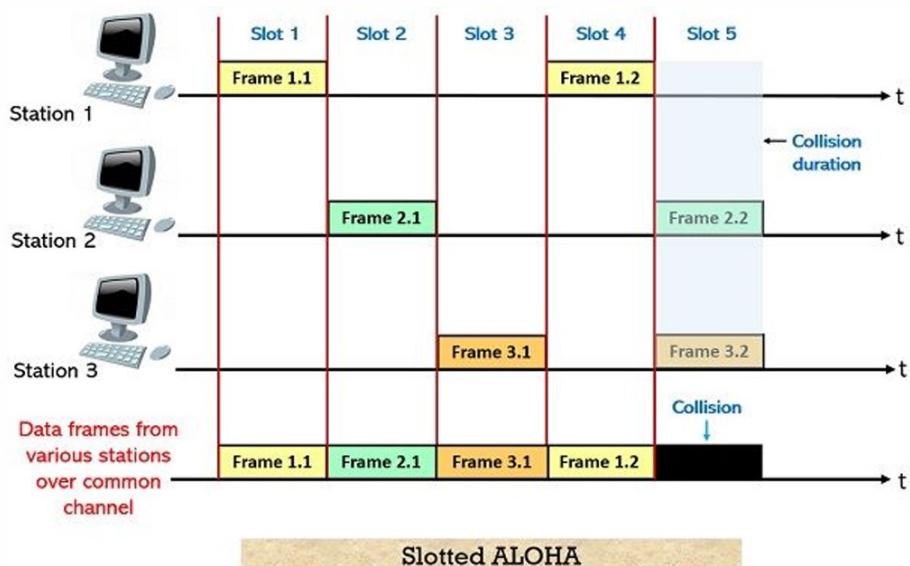
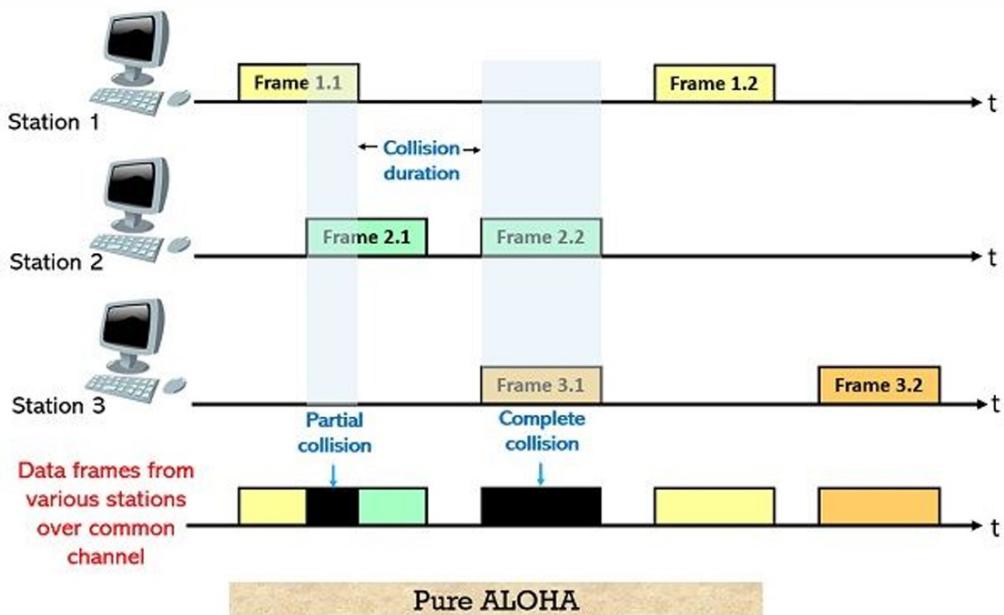
הנחה:

- יש N קודוקדים במערכת, ורrob ננתח את הפרוטוקול באשר $\infty \rightarrow N$.
- B – רוחב הפס
- L – כמה ביטים יש בתוך frame
- T – זמן,chioseb לפי גודל ה-frame חלקו רוחב הפס: $T = \frac{L}{B}$
- אם אנחנו שולחים את ההודעה בצורה תקינה ההנחה היא שאנו מוצלים את כל רוחב הפס לאורך כל ייחידת הזמן.
- לכל אחד יש slot זמן שבו הוא שולח הודעה, הודעות תשלchnה רק בתחילת slot. אם הודעה הגיעה באמצע slot בזמן היא תישלח בתחילת-slot הבא.
- באשר צומת מסימן לשולח הודעה, קיימת דרך לדעת שהייתה התנגשות בהודעות (ולשלוח אותה שוב בעתייד).
- שידור לוקח זמן, אבל אנחנו נתעלם מההפרשים האלה – הם זניחים לצורך החישוב.

Types of ALOHA

יש שני סוגי שוניים של ALOHA

- .1 – בצל צומת הוא בעל שעון משלו (סנכרון השעונים היא פעולה יקרה).
- .2 – כל השעונים מסוכברים.



ביחוח ביןומי לעומת ניתוח פואסוני:

הנחה היא שלכל מחשב ברשת יש משаг לשדר, והוא לא תמיד משדר את זה. מניחים שיש הסתברות p לשדר, ו- $(1-p)$ לא לשדר. מדובר בנוסי ביןומי ושאלים מה ההסתברות שימושו אחד לשדר, או שכמה ישדרו יחד. אם נרצה לספר כמה פעמים ניסוי ברכולי הצליח, זה מתפלג כמו משתנה מקרי ביןומי: $X_p \sim \text{Bin}(n, p)$.

אפשר להסכך על זה גם בצורה פואסונית – יש לנו n קודקודים, אבל n גדול מאוד ועוד λT – נזכיר שעבור N גדול ו- n קטן מתקיים $\text{Bin}(N, p) \approx \text{Poi}(\lambda)$ כאשר $\lambda = np$. אם נרצה למדל את המערכת עם **מעט קודקודים** نتيיחס להתפלגות **ביןומי**, ואם נרצה למדל עם **המוני (∞) קודקודים** نتيיחס להתפלגות **פואסונית**.

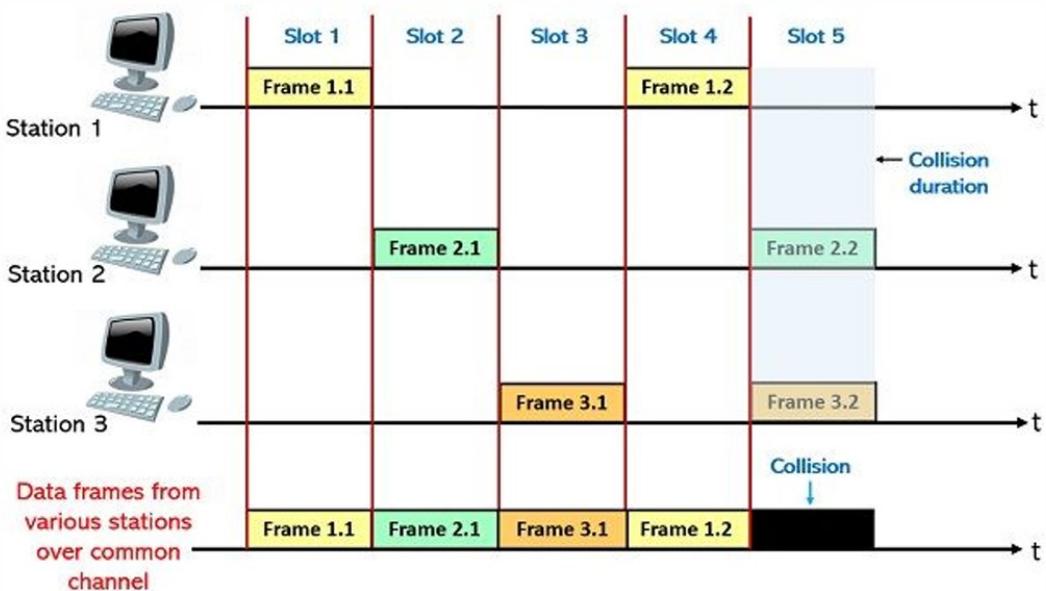
לכן יש 4 מודלים שאפשר לנתח:

Poisson	Binomial	
3	1	Slotted ALOHA
4	2	Pure ALOHA

בתרגול זהה נתמקד בניתוח הבינומי.

Slotted ALOHA – Binomial Approach

ב-slotted יש לכל תחנה slot מסוימת, ואין חפיפה ביניהם כלל.



Slotted ALOHA

בדוגמה זו אנחנו רואים שלמשל תחנה 1 הצליצה (בסיסי k) לשדר הוועה ב-1 slots, תחנה 2 ב-2, 3 ב-3 וכן הלאה. ב-5 slots אנחנו שמים לב שגם תחנה 2 וגם תחנה 3 הצליכו לשדר – וכך יש התנגשות. כל עמודה היא ניסוי ברנולי שאנו מגדירים בהצלחה אם קיבלנו את הערך 1 (רק תחנה אחת הצליכה לשדר הוועה). ב-5 slots קיבלנו את הערך 2 (כפי 2 תחנות שונות הצליכו לשדר) וכך הניסוי נכשל.

הנחות:

- כל מחשב מנסה לשדר משוחה, כל הזמן.
- רק בהסתברות p_{send} הצלחת יצליח לשדר הוועה.
- כל הוועה שהצליכה להתקבל השתמשה בכל ה-bandwidth.

שאלות לדוגמה:

1. נניח שככל ה-slots זהים, נסתכל על 1 slot. כמה זמן ננצל בתוחלת על שליחה מוצלחת?

פתרון:

נגדיר את X_p בתור מספר השליחות ביחידת זמן אחת, ונרצה למצוא את $\mathbb{P}_{suc} = \mathbb{P}(X_p = 1)$. נזכיר שאנו בניתוח הבינומי ולכן $X_p \sim \text{Bin}(n, p_{send})$.

- מה ההסתברות ל-frame מוצלח? מהנו סחה:

$$P_{bin} = \binom{n}{x} p^x (1-p)^{n-x}$$

נציב $1 = x$ ונקבל:

$$\mathbb{P}_{suc} = \mathbb{P}(X_p = 1) = n \cdot p_{send} \cdot (1 - p_{send})^{n-1}$$

נסמן ב- T_{suc} את הזמן שנצל על שליחה מוצלחת ב-slot יחיד. מה הוא $\mathbb{E}[T_{suc}]$?

$$\mathbb{E}[T_{suc}] = T \cdot \mathbb{P}_{suc} + 0 \cdot (1 - \mathbb{P}_{suc}) = T \cdot \mathbb{P}_{suc}$$

למה T בפועל \mathbb{P}_{suc} ? כי כשהוא מצליח הוא משדר על כל רוחב הפס במשך כל T השניות, ועוד ההסתברות שהוא נכשל בפועל הערך שמתקיים עבור כישלון – כלומר 0.

- מה ה-goodput של הרשת?

$$\eta = \frac{\mathbb{E}[T_{suc}]}{T} = \frac{T \cdot \mathbb{P}_{suc}}{T} = \mathbb{P}_{suc} = n \cdot p_{send} \cdot (1 - p_{send})^{n-1}$$

תוחלת של כמה שהצלחנו להעביר, חלקי זמן נתון.

2. מה היחס בין slots מוצלח ללא מוצלח? אפשר לשאול כמה *slots* בממוצע יש לנו עד שנקבל אחד מוצלח?

פתרון:

נגיד X_p מ"מ שמייצג את מספר הנסיכות. מתקיים: $\mathbb{E}[X_p] = \frac{1}{P_{suc}}$ וכן $X_p \sim \text{Geo}(\mathbb{P}_{suc})$. כמו כן, ה-

goodput *ychosab* לפיה:

$$\eta = \frac{T}{T \cdot \left(\frac{1}{\mathbb{P}_{suc}} \right)} = \mathbb{P}_{suc} = n \cdot p_{send} \cdot (1 - p_{send})^{n-1}$$

נרצה לדעת متى הערך הזה יהיה מקסימלי, שכן נגזר לפי p_{send} ונשווה לאפס. בסיום פיתוח מתמטי (הופיע בצורה חלקית בסיכון התרגול) קיבל $\frac{1}{n} = p_{send}$. משתמשים ב- p^* כדי לסמם את ההסתברות שambiliah את η לאופטימום, וב-*ALOHA slotted* זה יהיה $\frac{1}{n} = p^*$. אבל, אם נשאיר $\infty \rightarrow n$ נקבל:

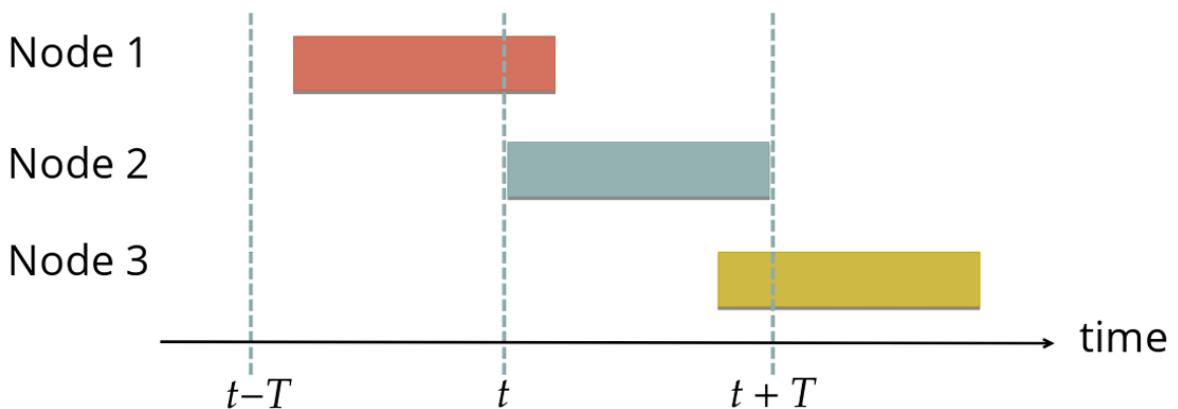
$$\lim_{n \rightarrow \infty} \eta(p^*) = \lim_{n \rightarrow \infty} (np^*(1 - p^*)^{n-1}) = \dots = \lim_{n \rightarrow \infty} \left(1 - \frac{1}{n}\right)^{n-1} = \frac{1}{e} \approx 0.37$$

מסקנה: אם יש לנו מערכת ב-*ALOHA slotted* ויש אינסוף צמתים – כולם משתמשים ב- p_{send} אופטימלי, אז שלישי מהזמן נצליח לשדר משחו – לא מאד אפקטיבי. בלי עוד הנחות זה התוצאה שנגיעה אליה.

Pure ALOHA – Binomial Approach

לכל תחנה יש slot משלה, אבל יש חפיפה בין slots של תחנות שונות. זה אומר שיכולה להיווצר התנגשות כמו ב-*ALOHA slotted*, אבל כאן יכולה להתறחש גם התנגשות חלקית. לשני הסוגים נתיחס באיזי הצלחה, גם אם הצלחנו להבין חלק מההודעה בעקבות התנגשות חלקית.

הקווי הנוסף כאן, הוא שלא מספיק לדאוג שלא יסדרו בתחרית frame של תחנה אחרת, באמצעות או בסוף – גם אם מישחו התחליל לשדר לפני sha-frame של תחנה אחרת התחליל, אבל משך השידור "גlesh" ל-frame אחר, עדין נקבל התנגשות. וזה יש זמן ברוחב T שיכול להיות בעייתי:



מה ההסתברות לשילוח מוצלח?

$$\mathbb{P}_{suc} = \mathbb{P}(\text{some node transmitted}) \cap \mathbb{P}(\text{no other node transmitted in } 2T)$$

ומאחר שמדובר במאורעות בלתי תלויים:

$$\mathbb{P}_{suc} = \mathbb{P}(\text{some node transmitted}) \cdot \mathbb{P}(\text{no other node transmitted in } 2T)$$

נציב ונקבל:

$$\mathbb{P}_{suc} = n \cdot p_{send} \cdot ((1 - p_{send})^2)^{n-1} = n \cdot p_{send} \cdot (1 - p_{send})^{2 \cdot (n-1)}$$

n – כי כל אחד מהצמתים היה יכול להיות זה שהצליח.

הרכיב – כי אנחנו צריכים שב-2 slots אף אחד לא ישדר.

1 – כי אספנו על כל השאר הצמתים לשדר.

מה ה-Goodput של המערכת הזאת?

נגדיר משתנה מקרי:

$$T_{suc} = \begin{cases} T & \text{successful slot} \\ 0 & \text{else} \end{cases}$$

$$\mathbb{E}[T_{suc}] = T \cdot \mathbb{P}_{suc}$$

$$\boxed{\eta} = \frac{\mathbb{E}[T_{suc}]}{T} = \frac{T \cdot \mathbb{P}_{suc}}{T} = \mathbb{P}_{suc} = \boxed{n \cdot p_{send} \cdot (1 - p_{send})^{2 \cdot (n-1)}}$$

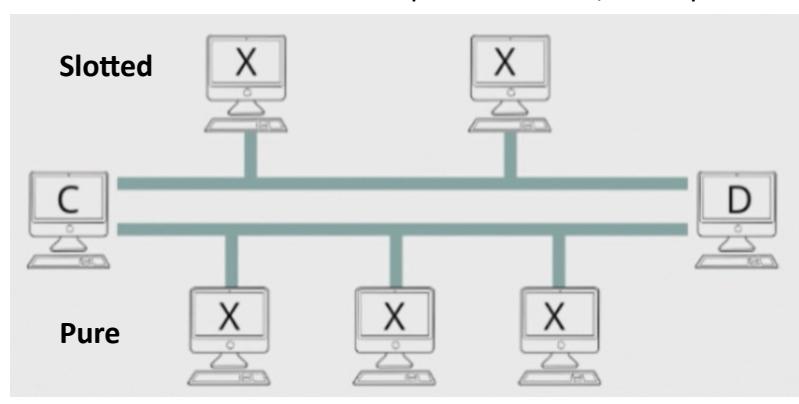
כמו מקודם, כדי למצוא את p_{send} אידיאלי נגזר ונשווה לאפס, נקבל $\frac{1}{2e} = \eta$, בדיק

חץ מה-goodput של slotted ALOHA.

שאלות תרגול

שאלה 1

בהתנן מערכת-slots hosts שנכרים D,C שמחברים דרך שתי מערכות נפרדות:



הרשת העליונה עם שני צמתים וועבדת לפי ALOHA slotted, התחתונה עם שלושה צמתים וועבדת לפי pure ALOHA. C אף פעם לא שולח הודעה. D מחייב להשתמש ברשות העליונה בהסתברות q , ושולח הודעה בהסתברות q . הצמתים X פשוט עושים רעש, ובהתברות q שולחים משהו.

א. C מעוניין לשלוח ל-D. מה ההסתברות שההודעה תגעה?

פתרון:

נחלק למקירם לפי הרשות שבה הודעה נשלחה, ומאחר שהמארעויות זרים, איחודם שווה לסכום ההסתברויות.

ברשות העליונה (slotted):

$$\mathbb{P}_{\text{slotted_suc}} = \mathbb{P}(\text{C send a message}) \cdot \mathbb{P}(\text{the others don't send}) = q \cdot (1 - q)^2$$

ברשות התחתונה (pure):

$$\mathbb{P}_{\text{pure_suc}} = \mathbb{P}(\text{C send a message}) \cdot \mathbb{P}(\text{no other transmissions for two slots}) =$$

$$= q \cdot \mathbb{P}(\text{no other transmissions for two slots}) = q \cdot ((1 - q)^2)^3 = q \cdot (1 - q)^6$$

סך הכל:

$$\mathbb{P}_{\text{total}} = \mathbb{P}(\text{use slotted ALOHA}) \cdot \mathbb{P}_{\text{slotted_suc}} + \mathbb{P}(\text{use pure ALOHA}) \cdot \mathbb{P}_{\text{pure_suc}} =$$

$$= q(p(1 - q)^2 + (1 - p)(1 - q)^6)$$

ב. C מעוניין לשלוח בתחילת כל slot. מה ה-goodput של הרשות העליונה?

פתרון:

כאשר הרשות העליונה נבחרה, יש 3 צמתים שמתקשרים (C, X, X). לכן:

$$\mathbb{P}_{\text{suc}}^1 = \binom{3}{1} \cdot q \cdot (1 - q)^2$$

יתכן גם ש-C יבחר לדבר ברשות התחתונה, ואז ברשות העליונה יש רק 2 צמתים שמתקשרים (X, X). לכן:

$$\mathbb{P}_{\text{suc}}^2 = \binom{2}{1} \cdot q \cdot (1 - q)$$

C בוחר את הרשות העליונה בהסתברות p , אך סה"כ:

$$\eta = p \cdot \mathbb{P}_{\text{suc}}^1 + (1 - p) \cdot \mathbb{P}_{\text{suc}}^2$$

שאלה 2

ברשת יש $2k$ קודקודים.

ה- k הראשונים משתמשים ב-ALOHA pure עם סלוטים בגודל T .

ה- k האחרונים משתמשים ב-slotted ALOHA עם סלוטים בגודל $2T$.

כל קודקוד משדר בתחלת כל slot בהסתברות p , וההודעה המשודרת לוקחת את כל ה-slots.



א. נבחר slot רנדומלי. מה ההסתברות שצומת שMRIAH pure יכול לשלוח הודעה בהצלחה

ב-slot זהה?

פתרון

צריכים שצומת הספציפי ישדר – הסתברות p

שאף צומת slotted ALOHA אחר לא ישדר – הסתברות $(1-p)^{n-1}$

שאף צומת ALOHA pure לא ישדר ב-3 slots, כי נשים לב לנตอน slotted ALOHA עם slot באורך $2T$,

ולכן ל-Pure T יש אחד לפני תחילת השידור שאסור לשדר בו, ועוד $2T$ זמן השידור ממש – הסתברות

$$(1-p)^{3n}$$

$$p \cdot (1-p)^{n-1} \cdot (1-p)^{3n}$$

ב. נבחר slot רנדומלי. מה ההסתברות שצומת שMRIAH pure יכול לשלוח הודעה בהצלחה ב-

slot זהה?

פתרון:

"תבנו שני מקרים – או שההודעה חופפת עם slot ייחד של ALOHA slotted, או שהיא חופפת עם 2

slots (שוב, לאחר slotted-slots ברוחב $2T$).

• חפיפה עם slot ייחד:

הסתברות שהוא ישדר: p

כל slotted ALOHA אחר לא ישדר: $(1-p)^n$

כל ALOHA pure אחר לא ישדר במשר 2 slots: $(1-p)^{2(n-1)}$

סה"כ מקרה ראשון: $p \cdot (1-p)^n \cdot (1-p)^{2(n-1)}$

• חפיפה עם שני slots:

הסתברות שהוא ישדר: p

כל ALOHA slotted לא ישדר בטווח של שני slots: $(1-p)^{2n}$

כל ALOHA pureACK אחר לא ישדר במשך 2 slots (אין שוני מהמקרה הראשון כי בכל מקרה ב-pure

$$(1 - p)^{2(n-1)} : \text{slots}$$

$$\text{סה"כ מקרה שני: } p \cdot (1 - p)^n \cdot (1 - p)^{2(n-1)}$$

ג. בסעיף זה נניח שבכל n הצמתים שהריצזו pureACK עבשו רצים ב-slotted ALOHA slots עם slots באורך T (האחרים עדין בגודל $2T$).

מה ה-Goodput של המערכת?

נסתכל על טווח של $2T$ ונבחן את כל המקרים שיוגדרו בהצלחה:

A: פקטה slotted מוצלחת.

B: פקטה pure מוצלחת בסלוט הראשון.

C: פקטה pure מוצלחת בסלוט השני.

D: 2 פקודות pure מוצלחות, אחת ב-slots הראשון והשנייה ב-slots השני.

נחשב אוטם:

A: כדי שתהייה פקטה slotted מוצלחת, בדיק אחד מתוך n הцמתים של "slotted $2T$ sized" יצליח לשדר:

$$\binom{n}{1} \cdot p \cdot (1 - p)^{n-1} = np(1 - p)^{n-1}$$

כל שאר ה-"pure T sized" לא ישלחו: $(1 - p)^{2n}$

האירועים ב"תולב" ובכפוף להסתברויות: $np(1 - p)^{n-1} \cdot (1 - p)^{2n} \cdot (p - 1)^n$

D: כדי שתהיינה 2 פקודות pure מוצלחות, צריך שיתקיים:

בדיקת "pure T sized" אחד יצליח לשדר ב-slots הראשון:

$$\binom{n}{1} \cdot p \cdot (1 - p)^{n-1} = np(1 - p)^{n-1}$$

חישוב זהה עבור ה-slot השני.

כל ה-"slotted $2T$ sized" לא ישדרו בכלל, בהסתברות: $(p - 1)^n$.

סה"כ נקבל: $n(p - 1)^n \cdot np(1 - p)^{n-1}$

B: כדי שתהייה פקטה pureACK מוצלחת בסלוט הראשון, צריך לבדוק אחד "pureACK" ישדר

$$\binom{n}{1} \cdot p \cdot (1 - p)^{n-1} = np(1 - p)^{n-1}$$

שבכל ה-"slotted $2T$ sized" לא ישדרו, בהסתברות $(p - 1)^n$.

לא להצליח בסלוט השני, בהסתברות $(1 - p)^{n-1}$ (n המשלים של להצליח).

סה"כ נקבל:

$$np(1 - p)^{n-1}(1 - p)^n(1 - np(1 - p)^{n-1})$$

C: המקרה שקול ל-B.

עכשווי לחישוב ה-Goodput:

במקרים שבהם ניצלנו רק חצי מהזמן ($2T$), כלומר מקרים B ו-C, علينا לכפול בחצי. נקבל:

$$\eta = A + \frac{1}{2}B + \frac{1}{2}C + D$$

הרצאה 4

הקדמה וחזרה

בשיעור ש做过ת התחלנו לדבר על שכבת ה-datalink והתפקיד שלה. ערכנו בבעיה שיש כמה צמתים שרוצים לתקשר והם חולקים לינק משותף – ערוץ תקשורת אחד עם כמה שלוחים פוטנציאליים. בשיעור זהה נראה פרוטוקולים שמטרתם לסנכרן את הגישה למשאב המשותף, ולתת "זכות דיבור" למחשב אחד בכל פעם. אלו פרוטוקולים מבזרים, אין ישות שאומرت למי יש זכות לשולח והם גם לא מבצעים חלוקה א-פרורית (כמו חלוקה בזמן או בתדר). הפרוטוקולים האלה יאלצו לסנכרן בעלי המחשבים מבליהם להשתמש בערוץ המשותף, ואת זה יבצעו באמצעות הסתרות, כלומר האלגוריתמים אינם דטרמיניסטיים – וזה מה שמבטיח שחלק מההודעות יעברו לעדן.

מצגת 2

שקלף 42-43

ברשתות wired יש שני סוגי של רשתות, bus משותף או switch (בו מתמקד בחלק השני של הרצאה זו). חלוקה נוספת יותר היא בין wireless ל-wired. ברשת אלחוטית יש אטגרים נוספים, למשל שכולם שומעים את ההודעות.

שקלף 44

אידיאלית, היינו רוצים פרוטוקול שייחלק בצורה הוגנת את השילוח בערוץ המשותף בין כל השולחים. לא נרצה לקבל סנכרון בין הצמתים, ונרצה להתחשב בכמות השולחים.

Random Access Protocols – ALOHA

שקלף 45

Slotted ALOHA

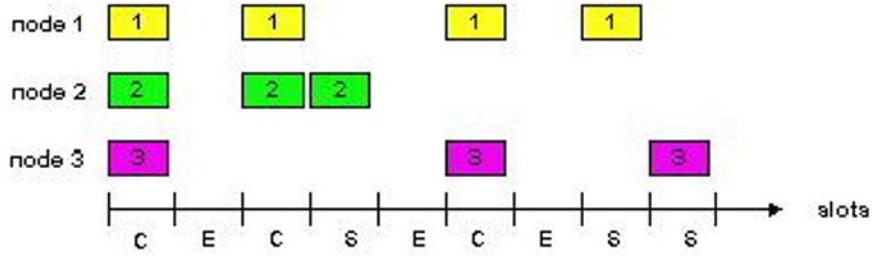
בפרוטוקול נניח שכל ההודעות באותו אורך, ונחלק את הזמן ל-slots שבהם שכל slot אפשר לשולח הודעה. נניח שהצמתים מסונכרים, כלומר שהם יודעים מתי מתחילה כל slot באותו הזמן ייחד. נניח שגם יש שניים שמשדרים הם מודעים להتنגשות.

איך ה프וטוקול עובד?

באשר אני כצומת קיבלתי הודעה לשולח מהשכבה הקודמת, אנסה לשולח אותה ב-slots הבא, ואבחן אם הייתה התנגשות או לא. אם לא הייתה התנגשות ויש לי משהו אחר לשולח, אעשה זאת. אחרת אם הודעה התנגשה, אני צריך לשולח את הודעה מחדש ובקופה הזאת אני יודע שיש שלוח נוסף, לכן כדי להתנתק מהסנכרון ממנו אגריל מספר שיקבע אם אשלח ב-slots הבא או לא.

שקלף 46

בדוגמה כאן יש 3 שלוחים שמצוירים את ההתנגשות, והගרת המספרים הובייה לרצף שאנו חנו רואים בדוגמה: עד שכולם הצלicho לשולח:



זה מדגים שהפרוטוקול לא פותר לגמרי התנגשויות, אנחנו כן נבזבז slots. יש שני סוגים של בזבוז – או שיתור מאחד שלח וזה הייתה התנששות (ההודעות לא הגיעו), או שאף אחד לא שלח (לא נצל slots בכלל). משני המצביעים הינו רוצים להימנע בשאיפה. **היתרון** ב프וטוקול, ביחס לפונקציונליות האידיאלית נבחן האם יש שלוח בודד הוא תמיד יוכל לשלוח. כמו כן, אין הרבה סyncron בין הצמתים אבל הוא כן דורך סyncron שעונים.

שאוף 72

נרצה לנתח את ה-protocol. אנחנו שואפים למקסם את ה-**Goodput**, בולמר למקסם את מספר הسلطים שהשתמשנו בהם ללא התנגשויות. לעומת זאת **Throughput** מייצג את %-ה-slots שימושו שלח בהם. בדוגמא, אם ב-slots מסוימים שני צמתים שלחו הودעה זה נחשב גבוה מבחינת **Throughput**, אבל מבחינת **Goodput** לא.

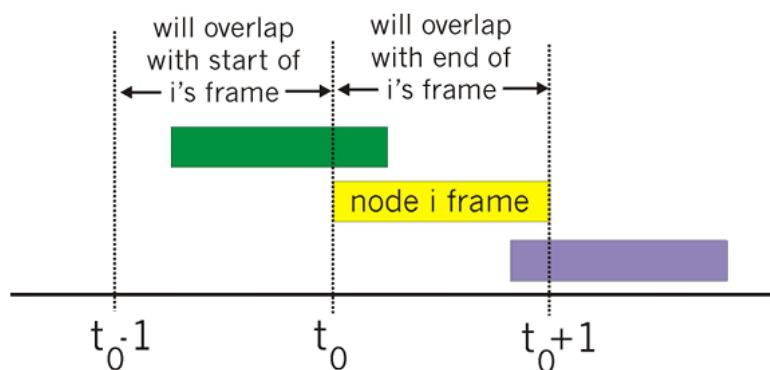
שאוף 73

נניח שיש לנו N צמתים שתמיד יש להם מידע לשלוח. כל אחד מהם מנסה לשלוח בהסתברות p . מה ההסתברות שיש צומת שמציל שלח? הסיכוי הוא שהוא בן שלח, וscalar $1 - N$ האחרים לא מצילו. לשלח, בולמר: $\binom{N}{1} \cdot p \cdot (1 - p)^{N-1}$ (התפלגות בינומית, אפשר לבחור צומת 1 מתוך N במקרה, וכן ה-choose). נרצה למצאו את ה- p האופטימי במונחי N . באופטימום מגיעים לניצולת $\frac{1}{e}$ (פיתוח מלא בתרגול).

שאוף 74

Pure ALOHA

השוני הוא שעכשו הצלמים לא מסונכרנים על מתי מתחילה ונגמר כל slot, אבל הספירה עדין באותו קצב בין צמתים שונים. נצפה שהוא יגרום לניצולת לדחת, מדוע? בדוגמה הבאה נניח שהסימונים של ה-slots הם על פי הספירה של:



נכיה ששלוחתי ב- t_0 . מאחר שה-slots של הצמתים האחרים לא מסונכרים עם שלי, אז **הירוק** התחילה לשЛОח קצר אחרי t_0 – t_1 וכן השליחה שלי הפרעה להודעה שלו לקרות הסוף. אותו דבר קרה עם השולח **הסגול**. בעצם בשאיון סנכדרו בהתחלה ובסוף של-slots הסיכון להתקנשות הוא בפוי.

שקלף 75

נסתכל על מישחו בודד ונבחן מה הסיכוי שהוא הצלח לשדר בלי הפרעה. זה מורכב ממספריו שהוא בן הצליח לשדר, והחלק השני זה שאף צומת אחר לא שידר בין זמן $t_0 + 1$ לזמן t_1 , **בפוי זמן** לעומת **ALOHA**:

$$p \cdot (1 - p)^{N-1} \cdot (1 - p)^{N-1} = p \cdot (1 - p)^{2(N-1)}$$

(עבור צומת כלשהו נדרש להכפיל ב- N).

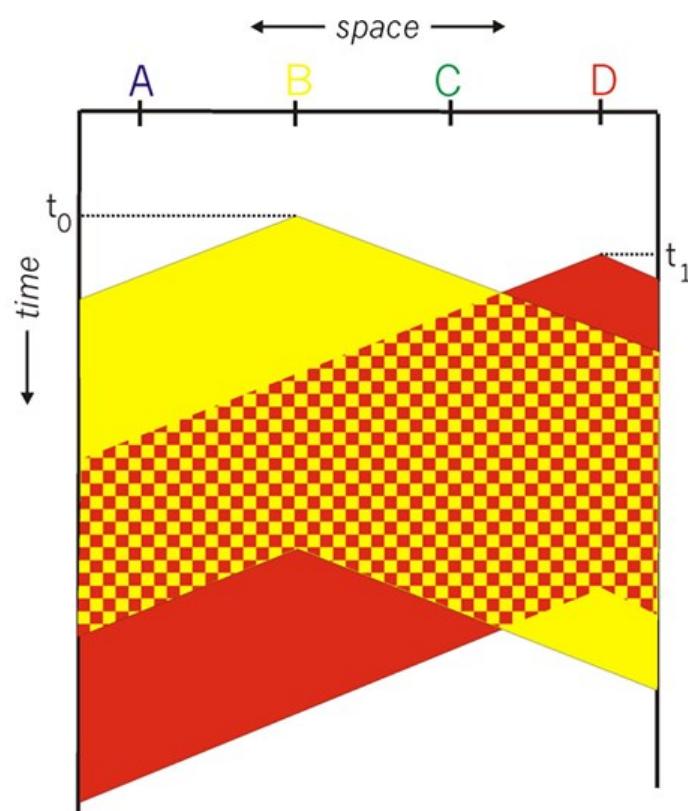
אחרי שמוסאים $\frac{1}{2e}$ אופטימלי מקבלים ניצולת אפילו פחות טובה מאשר Slotted CSMA, שהוא $\frac{1}{2e}$.

Random Access Protocol – CSMA

שקלף 76-77

CSMA (Carrier Sense Multiple Access)

שאר ה프וטוקולים שנציג היום בשיעור שייכים למשפחה שנקראת CSMA. הרעיון בגישה הוא שלפני שנסלח הודעה, נסתכל האם העוזץ פניו (אך אחד לא משדר ברגע). למה זה לא מונע התנגשיות של חלוטין? בתרשים הבא יש 4 שולחים וציר ע' הוא הזמן:



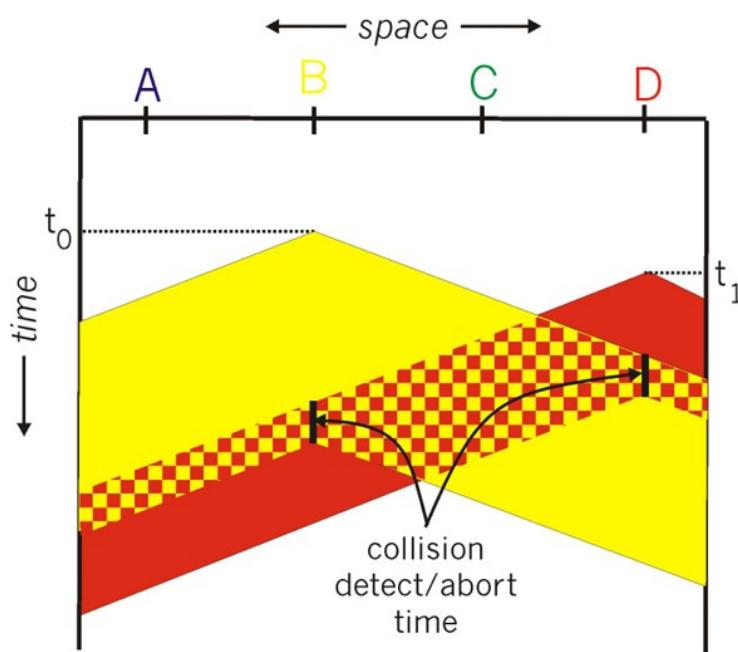
בזמן t_0 צומת B מנסה לשלוח. להודעה שלו יש propagation delay, אם נסתכל על הבית הראשון שהוא שלח, הוא הגיע ל-A ול-C באותו זמן, אבל D רחוק יותר (פיזית) ולכן לוקח לו זמן לקבל את הבית. B הסתכל בזמן t_0 על העරוץ והוא היה פנוי. בזמן t_1 D עושה את אותו הדבר, וגם הוא רואה שהעරוץ פנוי (כי הבית הראשון של ההודעה של B עוד לא הגיע). קיבלנו התנגשות.

שאוף 78

CSMA/CD (Collision Detection)

כיצע שדרוג למודל שראינו קודם, שנוסף לו זיהוי התנגשות. CSMA מנצל את יכולת הסתכל על הערוץ ולראות האם יש שלוח. CSMA/CD עומד לקרוא מהעරוץ ולראות האם יש שידור של מישחו אחר גם בזמן שאינו שלוח. ההבדל הוא ש-CSMA הבחן שהשידור שלו מתנגש עם מישחו אחר ובכל זאת המשיך (חלוקת המשובץ בגרף לעיל). CD בא לנצל את המידע הזה.

הערה: בדרך כלל ב-CD מדברים על LANs חוטיים, כי מאד קשה לעשות collision detection בցורה wireless (אולי אינני שומע אותו).



D משדר ובהתחלה הוא לא מזהה שהוא מתנגש עם מישחו אחר, כי המידע מ-B עדין לא הגיע אליו. ברגע שהוא מזהה שיש שלוח אחר D מפסיק לשולח בעצמו, וכך עושים גם B. אם מסתכלים על כמה זמן בזבז, זה ממשמעותית פחות מהמקרה הראשון, כי שני הצמתים ידעו להפסיק את השילוח שלהם.

שאוף 80

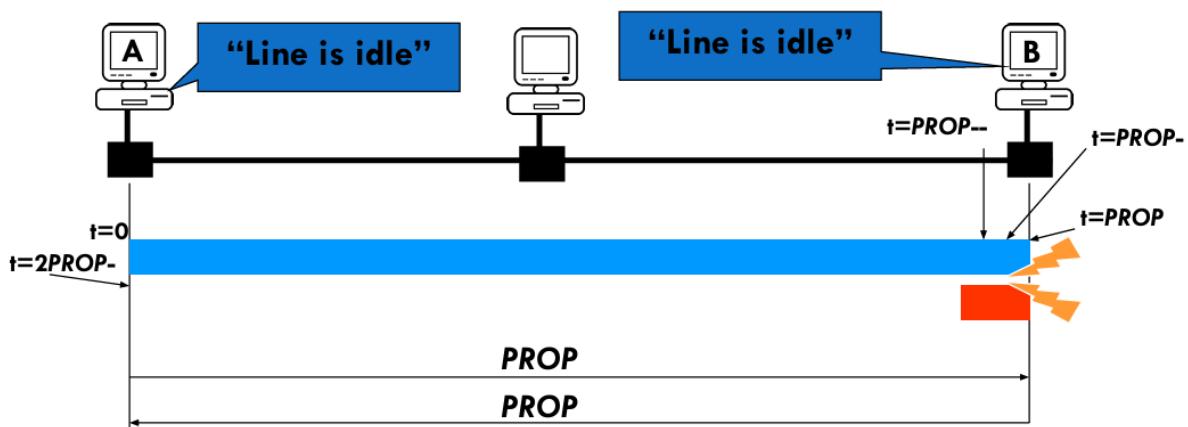
סיבום – carrier sense – בລומר בודקים את הקו לפני תחילת השילוח, collision detection משמעו זיהוי התנגשות במה שייותר מוקדם, ועצירת השידור בעת הזיהוי.

אם הייתה התנגשות נחבה זמן אקריא ואז נחזור ל-.carrier sense

שאוף 1

נראה עכשווי תכונה אינהרטית של פרוטוקול CSMA/CD. ננסה לראות מה מושך הזמן המקסימלי שיבול לעבור עד שאני אדע שהייתה התנגשות עם השידור שלי. מדוע זה משנה לי מה הזמן המקסימלי שצריך לעבור עד שאדע שהייתה התנגשות עם השידור שלי? כי זה תוחם **ליאת הזמן המינימלי שאינו צריך לשדר כדי לתפוס התנגשות בשידור.**

לשם בוחינת התכונה הזאת, ניקח את הנитוח לקיצון – בשקף כל "-ε" מייצג ε. בזמן $t = \text{PROP}$ ההודעה ממש במעט מגיעה ל-B, ואז B מחליט שהוא רוצה לשדר. B מקישב במשך ε שניות על הקו, רואת שהוא פמי, ואז מתחילה לשדר בעצמו בזמן $t = \text{PROP} + t$. המידע שלו יעבור על הקו ורואים התנגשות. A עומד לקבל את הביט הראשון של-B והספיק לשדר בעוד PROP , סה"כ כעבור 2PROP .



בלומר A שידר, אבל זיהה שהשידור נבשל רק 2 propagation time מאוחר יותר. מה אפשר להסיק מזה?

שאוף 2

זמן לשלח הودעה (TRANSP) צריך להיות לפחות פעמיים מה-propagation delay, אחרת נסימן לשלח את ההודעה מבלי לדעת שהיא הייתה בכל התנגשות. לכן הדרישה: $\text{TRANSP} > 2 \times \text{PROP}$.

שאוף 3-8

נכיה שיש לנו slots בצורה לוקאלית, הזמן לשלח הודעה הוא $\text{PROP} \times 2$ וההסתברות לשילחה היא α . ננסה להבין מה ה- $\eta_{\text{CSMA/CD}}$ של הפרוטוקול (כמה זמן שלוחתי מייד, חלקי הזמן ששלחתי מייד ועוד זמן לא מנצל):

$$\eta = \frac{\text{Time taken to send data}}{\text{Time taken to send data} + \text{overhead}}$$

באמצעות חישוב דומה למה שראינו קודם, הניצולות המקסימליות תהיה $\approx 40\%$, בלומר נצפה שעיל כל slot מונצל (שבו התקבלה הودעה) יהיו בממוצע 1.5 slots שלא הצליחו. נשתול בנוסחה ונקבל:

$$\eta_{\text{CSMA/CD}} = \frac{\text{TRANSP}}{\text{TRANSP} + \text{E}[time wasted]} = \frac{\text{TRANSP}}{\text{TRANSP} + \text{A}[2 \times \text{PROP}]} \\ = \frac{\text{TRANSP}}{\text{TRANSP} + [3 \times \text{PROP}]} \quad \dots$$

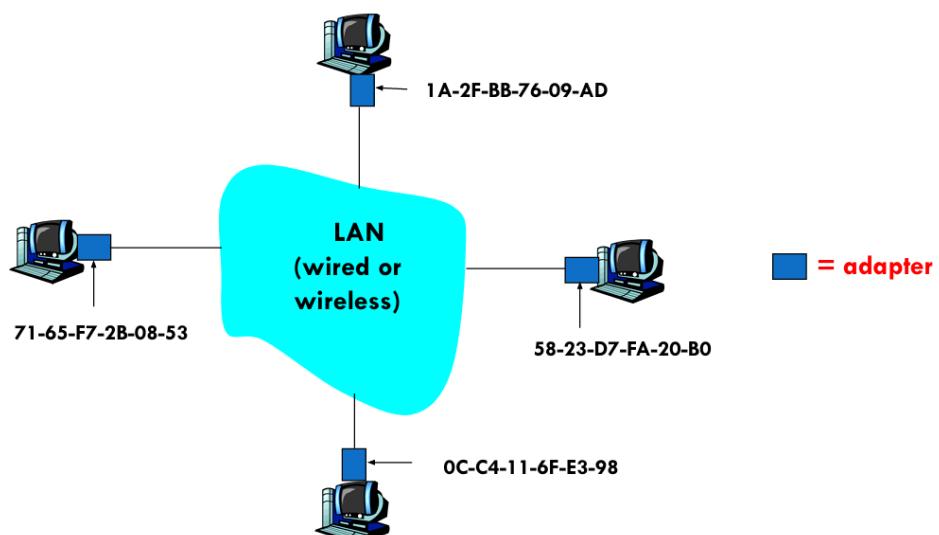
- מספר-slots שבזבוזו, כפול האורך של כל slot.
- $A=1.5$ – כמות-slots שנכשלים, וזמן השילחה שמתבצע זה פעמיים propagation time.

זה יותר טוב מ-ALOHA-slots כי עבשו אנחנו תליים ב-time propagation שהוא בתקווה זניח לעומת הזמן שנקוק למשדר את ההודעות.

בתובות MAC

שאוף 91-92

איך הפרוטוקולים ממומשים בפועל? דיברנו בשיעור שעבר על בתובות MAC, לכל ברטיס רשות יש בתובת MAC שהוטבעה בו על ידי היצרן – 48 ביטים שהם הזיהוי שלו. הביטים אלו מזהים אותו ללא תלות ברשת אליה התחברה.

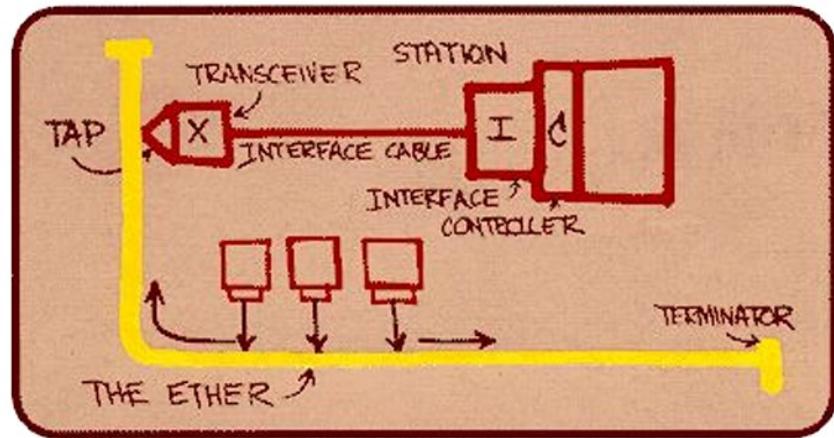


שאוף 93-94

כשאנחנו שולחים הודעה, בתובת MAC מצורפת להודעות שלנו, היא חלק מה-header של ה-frame.(frame header) בשים מחשבים שמחברים על סט משותף. כשאנחנו מקבלים הודעה אפשר לקרוא את ה-frame ולפי ה-destination MAC address אפשר לדעת האם זה מיועד אליו (ואז ברטיס הרשות יעלה את זה אליו שבסה אחת למעלה) או יזרוק אותה. בשיעור הבא נדבר על רשתות switch,switched, ושם ה-switch יעדיר בכתובת כדי לנתק אותה ליעדה.

שאוף 95

הטכנולוגיה הדומיננטית לחיבור רשתות **wired** היא Ethernet. הטכנולוגיה יחסית פשוטה, והוא עדין תקפה מהירות הגלישה של היום.



שאוף 96

Ethernet CSMA/CD

איך ה프וטוקול עובד מכך? הדבר היחיד שלא תארנו הוא איך עובד נושא האקראיות. אם זה היה תנטש, איך אדע לעזר?

בניגוד ל-ALOHA שם היה פרמטר k שהגדיר מה הסיכוי להמשיך, באן הסיכוי שלמו להמשיך לשילוח או לא תלוי בהיסטוריה של כמה פעמים הייתה התנטש. בשעשים ניתוח ל-ALOHA Slotted וראינו שהוא-* p האופטימלי תלוי ב- N מספר השולחים, וקשה לקבוע אותו בזמן קידוד. המנגנון שהבנינו ב-CSMA/CD הוא exponential backoff שמטרתו להתאים את ההסתברות לשילחה לפי מספר השולחים. ככל שהוא יותר התנטשויות עם ההודעה שלו, עולה בממוצע את הזמן שאחבה עד השילוח הבא. בהtanגשות ה- m אסתבל על כל המספרים בטוווי $\{0, \dots, 2^{m-1}\}$ ומתוכם אבחר באקראי מספר k שהוא יגדר כמה אני עומד לחכות.

המטרה: להגיע למצב שבכל שהtanגשו בי יותר אחבה יותר, ואז נצליח לפזר טוב יותר בזמן את שליחת ההודעות.

תרגום 3

Random Access Protocols

חזרה והקדמה

רוב הפרוטוקולים בשכבה 2 (דאטא לינק) עובדים באמצעות Random Access Protocols. מודמים מרכיבים עם המונ שחקנים שיכולים לצאת ולהיכנס במקומות שונים בזמן, ויש קושי להגדיר גוף אחד שייהי אחראי על התזמון של כלם. ב프וטוקולים אלה אם אחד נופל, אין השפעה רחבה על כלל המערכת. אנחנו עובדים ברגע על תווך אחד – אם שניים משדרים על אותו התווך במקביל, השידור בושל.

הבהרות:

1. Bandwidth אינו שקול ל-Propagation Speed, מדובר בכמה זמן לוקח לשידור מעבורתו (בתרגום נתיחס לזמן זהה באפסי, לא תמיד זו תהיה ההנחה).
2. גם ב-ALOHA יש slots. תמיד אפשר לשלוח רק בתחילת slot, פשוט ה-frame של קודקוד אחד לא מסונכרן בהכרח עם ה-frame של קודקוד אחר, וכך "מתלבשים" בצורה מושלמת מעל-slots.
3. במודל הגנרי של ALOHA ולצורך התרגום הזה כאשר מוסיפים collision detection, הצומת ידע על collisions רק בסיסם השידור.

ALOHA – Poisson Approach

בתרגול הקודם דיברנו על הגישה הבינומית, היום נדבר על הגישה הפואסונית (מעבר Pure Slotted ומעבר Slotted):

$$X_p \sim \text{Poi}(gT)$$

עם קצב שליחה g וזמן T בלבד, למשל זמן של slots כפול קצב השילחה ועוד נדע כמה אמורים לשלוח באותו slots – אם זה יהיה בדיק 1 אז הצלחנו לשדר בהצלחה אחרת לא.

תזכורת – תהליך Poisson

תהליכי ספירה – בהינתן שתי מאורעות שונים טווח הזמן שלהם לא חופה. מה שיעניין אותנו זה אורך הטווח (בניגוד למשל למידול כמה אנשים עולים על אוטובוס בשעה מסוימת שהוא אינם תהליכי פואסוני).

תבונה חשובה: סכום של משתנים מקרים פואסוניים הוא משתנה מקרי פואסוני.

Notation	Explanation	Units
g	Rate of <i>total</i> messages (both new and failed messages) that are up for transmission	$\frac{1}{\text{sec}}$
$G = gT$	Average number of transmitted (successful and failed) messages in a time interval T	None
P_{suc}	Probability of a (single) successful transmission (i.e., no collisions occurred)	None

Slotted ALOHA – Poisson Approach

בעזרת נוסחה של משתנה מקרי פואסוני נקבל:

$$X_p \sim \text{Poi}(gT) \quad \text{then} \quad \mathbb{P}_{t=T}(X = i) = \frac{(gT)^i}{i!} e^{-gT}$$

זה טווח הזמן, המשתנה המקרי X שווה לבמה הודעות נשלחו באותו טווח זמן (אם שווה 1, הצלחנו). לכן:

$$\mathbb{P}_{suc} = \mathbb{P}_{t=T}(k = 1) = gT \cdot e^{-gT} \stackrel{gT := G}{=} G \cdot e^{-G}$$

וה-**Goodput** יהיה פשוט $G \cdot e^{-G}$, כמו במודל הבינומי (לפי קצב שליחה והסתברות הצלחה). תזכורת מה חישוב במקרה הבינומי:

$$\eta = \frac{\mathbb{E}[T_{suc}]}{T} = \frac{T \cdot \mathbb{P}_{suc}}{T} = \mathbb{P}_{suc}$$

דרך נוספת לבצע את הניתוח זהה הוא **מפרנסקטייה של פקטה**. מהי הסתברות ההצלחה אם ההודעה נשלחה?

$$\mathbb{P}_0 = \mathbb{P}_{t=T}(k = 0) = e^{-gT} = e^{-G}$$

מהו ה-**Goodput**? כמו קודם. יש קצב שליחה בלבד G . אם נסתכל מנקודת מבטו של פקטה, הסיכוי שהוא קיים הוא G (לפי קצב השליחה) לכן נקבל:

$$\eta = G \cdot \mathbb{P}_0 = G \cdot e^{-G}$$

הערה: חישבנו כאן כמובן שיש N קודקודים, אבל אם משתמשים על פקטה שנשלחה מקודקוד מסוים היינו צריכים להתחשב רק ב- $1 - N$ קודקודים. לאחר שימוש ב-ALOHA על המקורה שבו $\infty \rightarrow N$ ההבדל זניח.

מהו ה-**goodput** המקסימלי? אחרי גזירה מקובלם שהוא יתקבל עבור $1 = *G$. קיבל $0.37 \approx 0.37$. בדיק כמו בתרגול הקודם עם משתנה מקרי ביןומי, זה הגיוני כי מדובר במקרה מאוד דומה – פשוט במקום להסתכל על כל קודקוד במשתנה מקרי שונה אנחנו משתמשים על המערכת כולה בכלל.

Pure ALOHA – Poisson Approach

הчисוב יהיה מאוד דומה למקרה של Slotted:

$$\mathbb{P}_{suc} = \mathbb{P}_{k=0}(t = T) \cdot \mathbb{P}_{k=1}(t = T) = e^{-gT} \cdot gTe^{-gT} = gT \cdot e^{-2gT} = G \cdot e^{-2G}$$

כפי אנחנו דורשים שבמשך זמן T שקדם לנקודת השילוח שלנו לא ישלח כלום ($k = 0$) וشبនקודת הזמן שלנו תשלח בדיקת הודהה אחת ($k = 1$).

מה יהיה ה-**Goodput**?

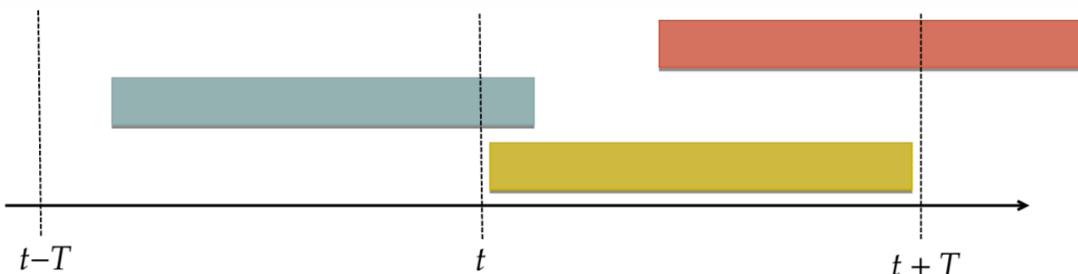
$$\eta = \mathbb{P}_{suc} = G \cdot e^{-2G}$$

כמו קודם, אפשר לחשב מנקודת מבט של פקטה ולהגיע לתוצאות דומות. אנחנו יודעים שהפקטה נשלחה, אך נדרש שבשלוט הקודם ובשלוט הנובייחי אף אחד אחר לא נשלח. נקבל:

$$\mathbb{P}_{suc} = \mathbb{P}_{k=0}(t = T) \cdot \mathbb{P}_{k=0}(t = T) = e^{-gT} \cdot e^{-gT} = e^{-2gT} = e^{-2G}$$

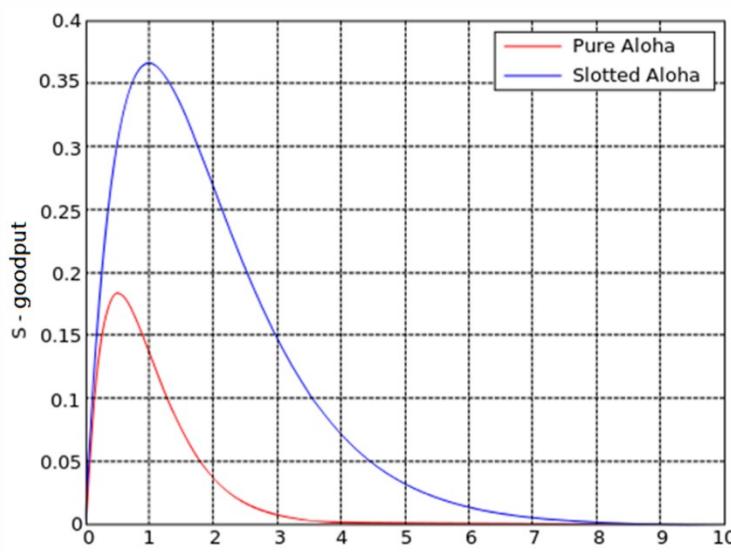
ואז ה-**Goodput** יהיה זהה גם כן, רק נדרש להכפיל בהסתברות שהפקטה נשלחה:

$$\eta = G \cdot e^{-2G}$$



שוב, נרצה למקסם את ה- G עבור ה-**Goodput**. נקבל $\eta_{G^*} = \frac{1}{2e}$ כמו בגרסת הבינומית.

השוואה בין Pure לבין Slotted Aloha



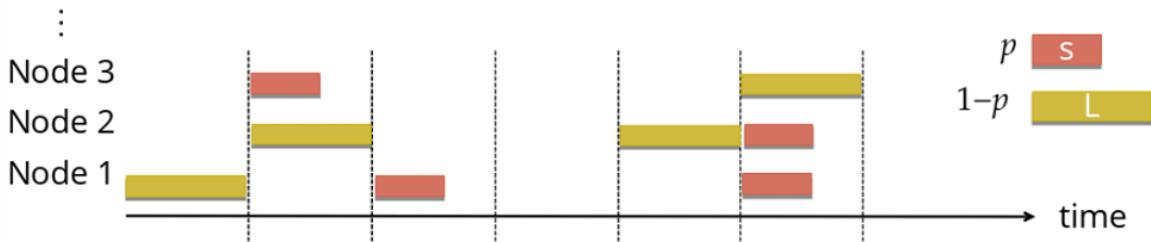
אם ה- G שנחבר לשים (ציר א) יהיה גדול יותר התנגשויות ולכן goodput צונח. באופן דומה עבור G קטן מדי, יותר מדי קודקודים "חושיים" לשולח, וגם בכך זה נראה דעיכה מאוד גדולה ב-goodput.

שאלות תרגול

שאלה 1

נתונה מערכת Slotted ALOHA שבה הודעות נשלחות בתהיל פואסוני עם קצב שליחה g . ישן שתי סוגים של הודעות: פקודות עם S ביטים שמאזועות בהסתברות p , ופקודות עם L ביטים בהסתברות $1-p$. נתון גם $> L$.

קצב שליחה הוא $1 \frac{\text{bit}}{\text{sec}}$ וגודל slot הוא L שניות ($T = L$).



מה ההסתברות לשליחה מוצלחת?

נזכור כי מתקיים: $X_p \sim \text{Poi}(gT)$ then $\mathbb{P}_{t=T}(X = i) = \frac{(gT)^i}{i!} e^{-gT}$

$$\mathbb{P}_{suc,L} = (1-p) \cdot \mathbb{P}_{t=L}(k=1) = (1-p)gL e^{-gL}$$

כלומר ההסתברות לשולח הודעה באורך L , כפול ההסתברות שבתווח זמן L רק אחד ישלח. עבור S :

$$\mathbb{P}_{suc,S} = p \cdot \mathbb{P}_{t=L}(k=1) = pgLe^{-gL}$$

מהו ה-goodput?

נגדיר משתנה מקרי שיעיל לנו לחישוב ה-goodput. הוא יעבד בראוי של "כמה ביטים של הודעות מוצלחות הצלחתי לשדר":

$$T_{suc} = \begin{cases} L & \text{successful L packet} \\ S & \text{successful S packet} \\ 0 & \text{else} \end{cases}$$

$$\mathbb{E}[T_{suc}] = L \cdot \mathbb{P}_{suc,L} + S \cdot \mathbb{P}_{suc,S}$$

$$\eta = \frac{\mathbb{E}[T_{suc}]}{L} = \frac{L}{L} \cdot \mathbb{P}_{suc,L} + \frac{S}{L} \cdot \mathbb{P}_{suc,S} = gLe^{-gL} \left(\frac{S}{L}p + \frac{L}{L}(1-p) \right)$$

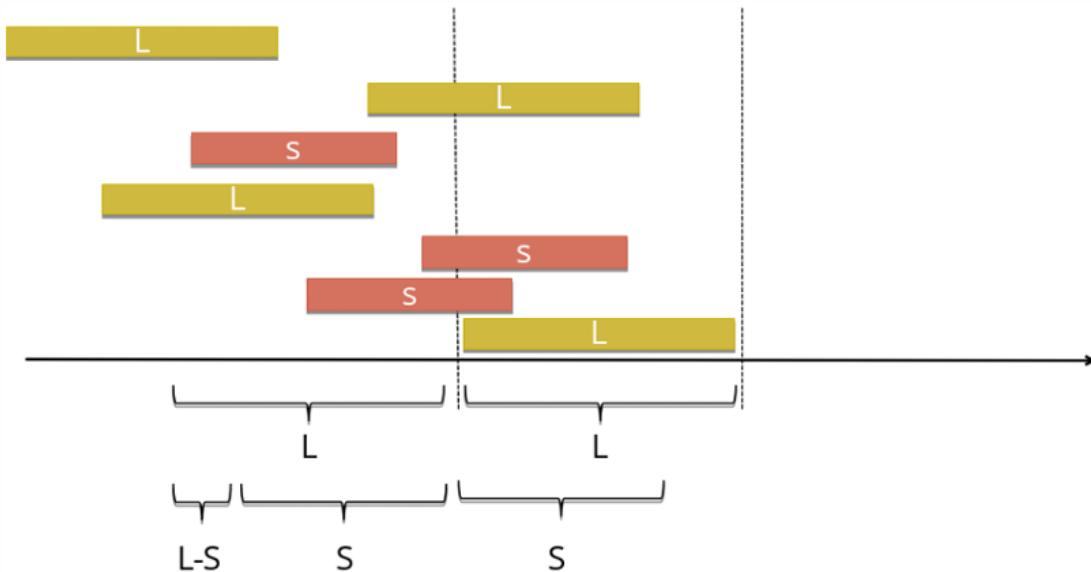
$$\eta = gLe^{-gL} \left(\frac{S}{L}p + 1 - p \right)$$

שאלה 2

נסתכל על נטווי השאלה הקודמת, אבל הפעם בהנחה שהמערכת היא Pure ALOHA.

מה ההסתברות לשילוח מוצלח?

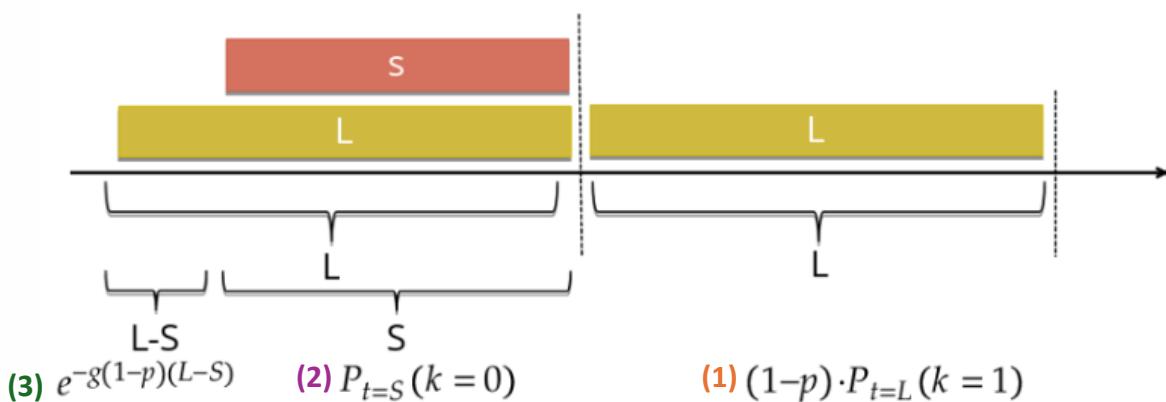
בגל שזה Pure, נדרש לעשות חישוב איפה יכולות להיות התנגשויות:



הודעה מסוג L יכולה להפריע בטעו של $(L, +L)$. הודעה מסוג S יכולה להפריע בטעו של $(S, -S)$. נגידר את זמני ההגעה לפוי סוג:

$$g_S = p \cdot g \quad ; \quad g_L = (1 - p) \cdot g$$

סיכוי ההצלחה עבור הودעה מסוג L מתחילה ל-3 מקרים:

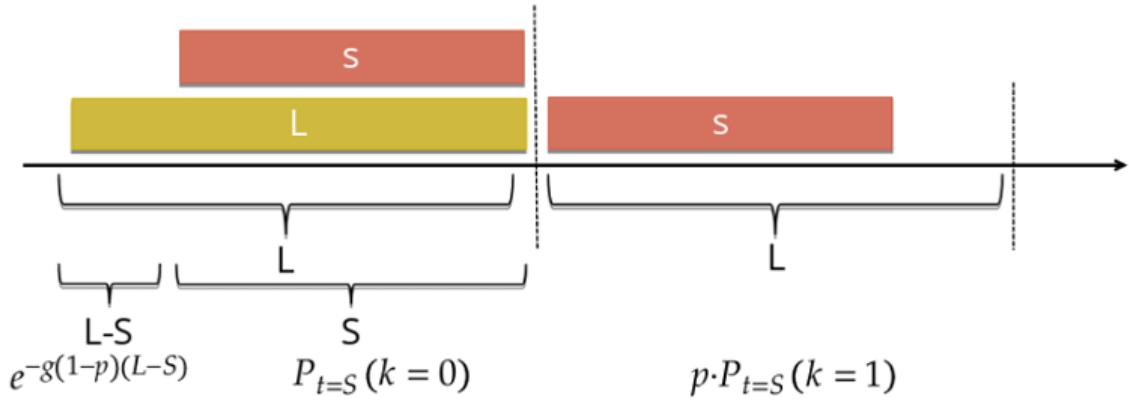


קודם כל (1), נדרש שזמן L שב-tots שבו התחלנו לשדר – אף אחד אחר מלבדנו לא ישדר ($k = 1$). כמו כן (2) בין $(0, S, -)$ אנחנו נדרש שאף סוג לא ישלח (לא S ולא L). לבסוף (3) נרצה בכך של השילוח רק של הודעה מסוג S (לא יתנגש), אבל לא נרצה L.

שלושת המאירועות האלו זרים, لكن ככפוף אוטם כדי לדרש את שלושתם ונקבל:

$$\mathbb{P}_{suc,L} = (1-p)e^{-g(2L-p(L-S))}$$

סיכוי ההצלחה עבור הודעה מסוג S מחלק ל-3 מקרים, נחשב באופן דומה:



עם הסברים דומים לחישוב סיכוי ההצלחה עבור הודעה מסוג L. ככפוף ונקבל:

$$\mathbb{P}_{suc,S} = pe^{-g(L+S-p(L-S))}$$

שוב נגדיר משתנה מקרי:

$$T_{suc} = \begin{cases} L & \text{successful L packet} \\ S & \text{successful S packet} \\ 0 & \text{else} \end{cases}$$

$$\mathbb{E}[T_{suc}] = L \cdot \mathbb{P}_{suc,L} + S \cdot \mathbb{P}_{suc,S}$$

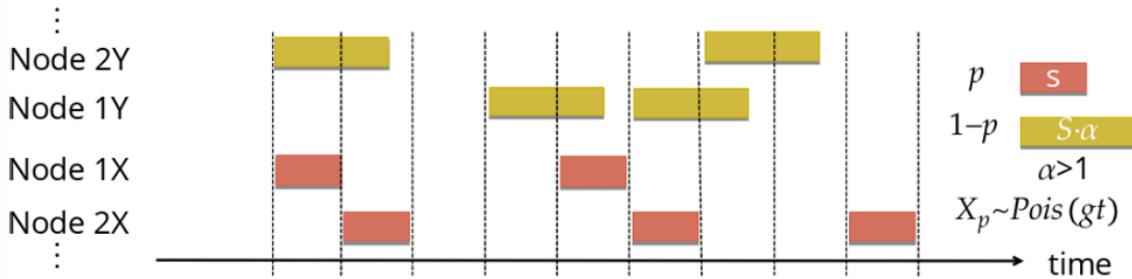
$$\boxed{\eta = \frac{\mathbb{E}[T_{suc}]}{L} = \frac{L}{L} \cdot \mathbb{P}_{suc,L} + \frac{S}{L} \cdot \mathbb{P}_{suc,S} = (1-p)e^{-g(2L-p(L-S))} + \frac{S}{L}pe^{L+S-p(L-S)}}$$

שאלה 3

במערכת Slotted ALOHA עם גודל slot של S יש שני קווים:

X שמכיל $N \cdot d$ קודקודים ששולחים הודעות בגודל S ומשתמשים ב- $\frac{1}{4}$ מרוחב הפס.

Y שמכיל $N \cdot (1-p)$ קודקודים ששולחים הודעות בגודל $S \cdot \alpha$ ומשתמשים ב- $\frac{3}{4}$ מרוחב הפס.



X ו-Y לא מפריעים אחד לשני, אפשר לדמיין את זה כשתי מחשבים שמחוברים על ידי bus. או שני שולחים הודעות על bus יחיד, או על 3 slots ביחד (כדי להמיחס את הרענון). אם שולחים הודעה על הפס היחיד זה לא יתנגש בהודעות שנשלחו על 3-slots האחרים.

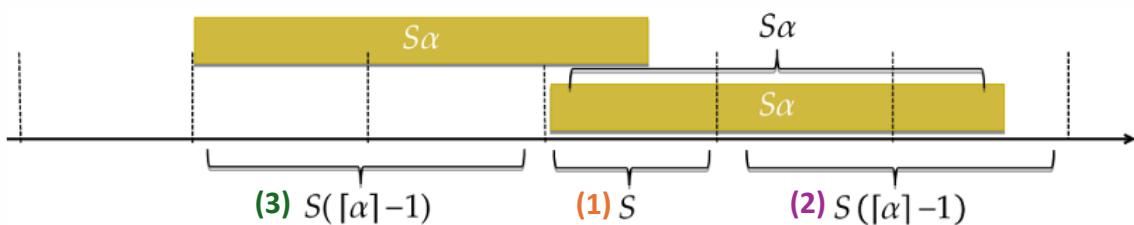
נגידו את זמני ההגעה לפי סוג:

$$g_{S_\alpha} = (1-p) \cdot g \quad \Rightarrow \quad S_p^\alpha \sim \text{Poi}((1-p)g)$$

$$g_S = p \cdot g \quad \Rightarrow \quad S_p \sim \text{Poi}(pg)$$

מה ההסתברות לשליחה מוצלחת?

עבור הودעה מסווג S_α :



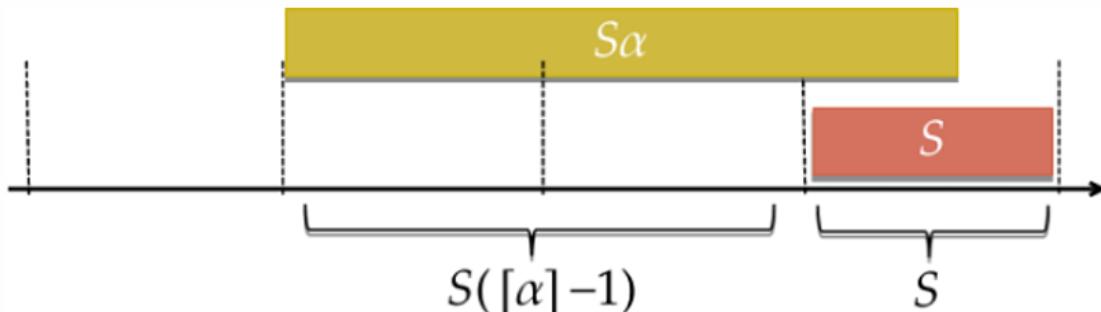
מתי עלולה להיות בעיה עם הودעה מהסוג? בדרכ ביל-ב-Slotted אנחנו צריכים להסתבל רק על ה-slots שלו, אבל מאחר שההודעה שאנו שולחים היא יותר מאורך של slot, גם ה-slots הקודמים עלולים להפריעו. מה שיעניין אותנו זה הערך $\lceil \alpha \rceil$, כי בפועל אם זה מתנגש בחלק מה-slots זה גם יהרום את השידורה נחלק ל-3 מקדים.

במקרה (1) אנחנו מסתכלים על התחלת באורך S שבה נרצה שידור יחיד, ואז ב-(2) מסתכלים על כל השאר פחותה-slots הראשונים. המקרה ה-(3) דומה כי גם בו לא נרצה אף שידור:

$$\mathbb{P}_{suc,S_\alpha} = \mathbb{P}_{k=0}^{S_\alpha}(t = S([\alpha] - 1)) \cdot \mathbb{P}_{k=1}^{S_\alpha}(t = S) \cdot \mathbb{P}_{k=0}^{S_\alpha}(t = S([\alpha] - 1))$$

$$\mathbb{P}_{suc,S_\alpha} = g(1-p)Se^{-g(1-p)S(2[\alpha]-1)}$$

עבור הودעה מסוג S :



כאן לא נוצרת אותה בעיה כמו קודם שפתקות α לא מתערבות בשידור של פתקות מסווג S , ומאחר שזה Slotted ALOHA לא יכולה להישלח עוד הודעה מסוג S במקביל:

$$\mathbb{P}_{suc,S} = \mathbb{P}_{k=1}^S(t = s) = gpSe^{-gps}$$

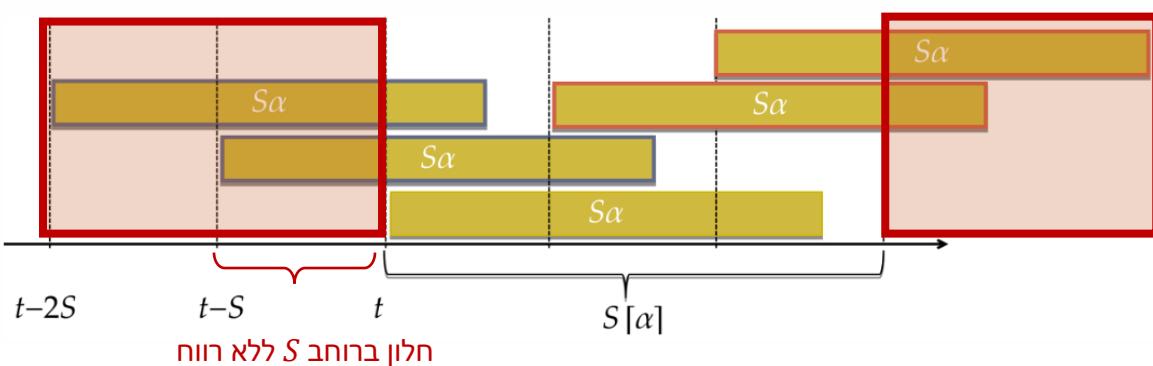
מהו Goodput ?

נשים לב שההודעות מסווגים שונים לא מפריעות אלו לאלו, אז نتيיחס לכל מקרה לגופו.

עבור הודעות S_α :

נסתכל על טווח $[S_\alpha]$. ונרצה לבדוק כמה ביטים נרוויים בו.

עבור הודעה שנשלחה בהצלחה בזמן t נרוויים $[S_\alpha]$. אם מישהו שלח לפני זה, גם נרוויים ביטים – אבל רק בחילון שבו אנחנו מודדים (כלומר לא מה שמסומן באדום בתמונה):



עבור המקרה שבו שלחנו בדיק בזמן t נרווי $\alpha \cdot S$ ביטים.

⋮

עבור המקרה שבו שלחנו בזמן $S - t$ נרווי $\alpha \cdot S = S(\alpha - 1)$ ביטים. חיסרנו S בגלל **חלון ברוחב S** **לא תוקן**, כי הרוחח התקבל לפני זמן המדייה.

⋮

עבור המקרה שבו שלחנו בזמן $S + t$ נרווי $[\alpha] \cdot S = S([\alpha] - 1)$ ביטים, שוב מאחר שהתחלנו למדוד S שניות לאחר זמן t מטור החילון עליינו אנחנו מסתכלים שהוא בגודל $[\alpha] \cdot S$ אז נותרו רק $S - [\alpha] \cdot S$ ביטים. מאחר שהזמן הוא רציף, יש הרבה מקרים בין לבין.

נדיר משתנה מקרי:

$$T_{suc,S\alpha} = \begin{cases} S\alpha & \text{successful packet that starts at } t \\ S(\alpha - 1) & \text{successful packet that starts at } t - S \\ S(\alpha - 2) & \text{successful packet that starts at } t - 2S \\ \vdots & \vdots \\ S([\alpha] - 1) & \text{successful packet that starts at } t + S \\ S([\alpha] - 2) & \text{successful packet that starts at } t + 2S \\ \vdots & \vdots \\ S \cdot 1 & \text{successful packet that starts at } t + S([\alpha] - 1) \\ 0 & \text{else} \end{cases}$$

נחשב את התוחלת ונקבל:

$$\begin{aligned} \mathbb{E}[T_{suc,S\alpha}] &= P_{suc,S\alpha} \left(S\alpha + \sum_{k=1}^{[\alpha]-1} S(\alpha - k) + \sum_{k=1}^{[\alpha]} S([\alpha] - k) \right) \\ &= P_{suc,S\alpha} S \left(\alpha + \sum_{k=1}^{[\alpha]-1} \alpha - \sum_{k=1}^{[\alpha]-1} k + \sum_{k=1}^{[\alpha]-1} k \right) \\ &= P_{suc,S\alpha} S (\alpha + ([\alpha] - 1)\alpha) \end{aligned}$$

$$\mathbb{E}[T_{suc,S\alpha}] = P_{suc,S\alpha} S \alpha [\alpha]$$

את **הירוק** פיצלנו ל-2, ובתכלית הפכנו את סדר הסכימה. קיבלנו את כמהות הביטים שנתקבל. נעשה את אותו דבר להודעות מסווג S , ושם (למזהלנו) הסיפור הרבה יותר פשוט:

$$T_{suc,S} = \begin{cases} S & \text{successful S packet} \\ 0 & \text{else} \end{cases} \implies \mathbb{E}[T_{suc,S}] = S \cdot P_{suc,S}$$

מה ה-Goodput?

$$\begin{aligned}\eta &= \frac{3}{4}\eta_{S_\alpha} + \frac{1}{4}\eta_S = \frac{3}{4} \frac{\mathbb{E}[T_{suc,S_\alpha}]}{S[\alpha]} + \frac{1}{4} \frac{\mathbb{E}[T_{suc,S}]}{S} \\ &= \frac{3}{4} \frac{S\alpha[\alpha] \cdot P_{suc,S_\alpha}}{S[\alpha]} + \frac{1}{4} \frac{S \cdot P_{suc,S}}{S} \\ &= \frac{3}{4}\alpha P_{suc,S_\alpha} + \frac{1}{4}P_{suc,S}\end{aligned}$$

נבחן שרווח הפס בשאלה היה מחולק ל $\frac{1}{4}$ ול $\frac{3}{4}$, ולכן גם יש צורך להבדיל בין Goodput שעובר בכל אחד מהם לפי הקבוע המתאים.

נمشיך בנושא CSMA – איך לאפשר לבמה מחשבים (נקודות שידור) לתקשר כאשר הם חולקים main domain משותף. היום נדבר על CA\CSMA שזה וריאנט נוסף של CSMA לרשתות wireless, לאחר מכן נעבור לרשותות גדולות יותר, switches ואיך אפשר לחבר אותם יחד לרשת גדולה ומורכבת.

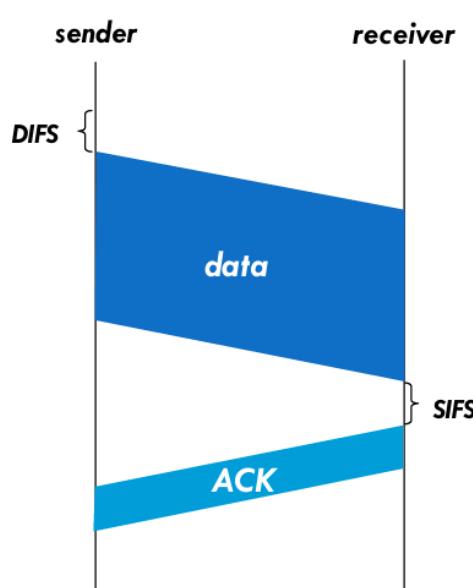
Wi-Fi ברשתות CSMA/CA

מצגת 2
שקל 99-100

תשכורת CSMA זו שיטה בה משתמשים על הערוץ לפני השידור, "מקשייבים" האם מישחו אחר מנסה לשדר. דיברנו על CSMA/CD שבו יש צפיפות התנגשויות. אם התנגשנו עם מישחו אחר נפסיק את השידור.

ברשתות Wi-Fi המנגנון לא עובד כל כך טוב, בעיקר כי קשה לוודת שימושו אחר משדר וגם כי קשה להקשיב בזמן שאחננו משדרים בעצמינו (בעיית חומרה טכנית). הציגו וריאנט אחרית שנקרא (Collision Avoidance) CSMA/CA (Collision Avoidance) שמטרתו לנסוט במקומות לזרות התנגשויות להימנע מהן מלכתחילה, למשל "להקדים תרופה למבה" (לא מובטח שייעבוד).

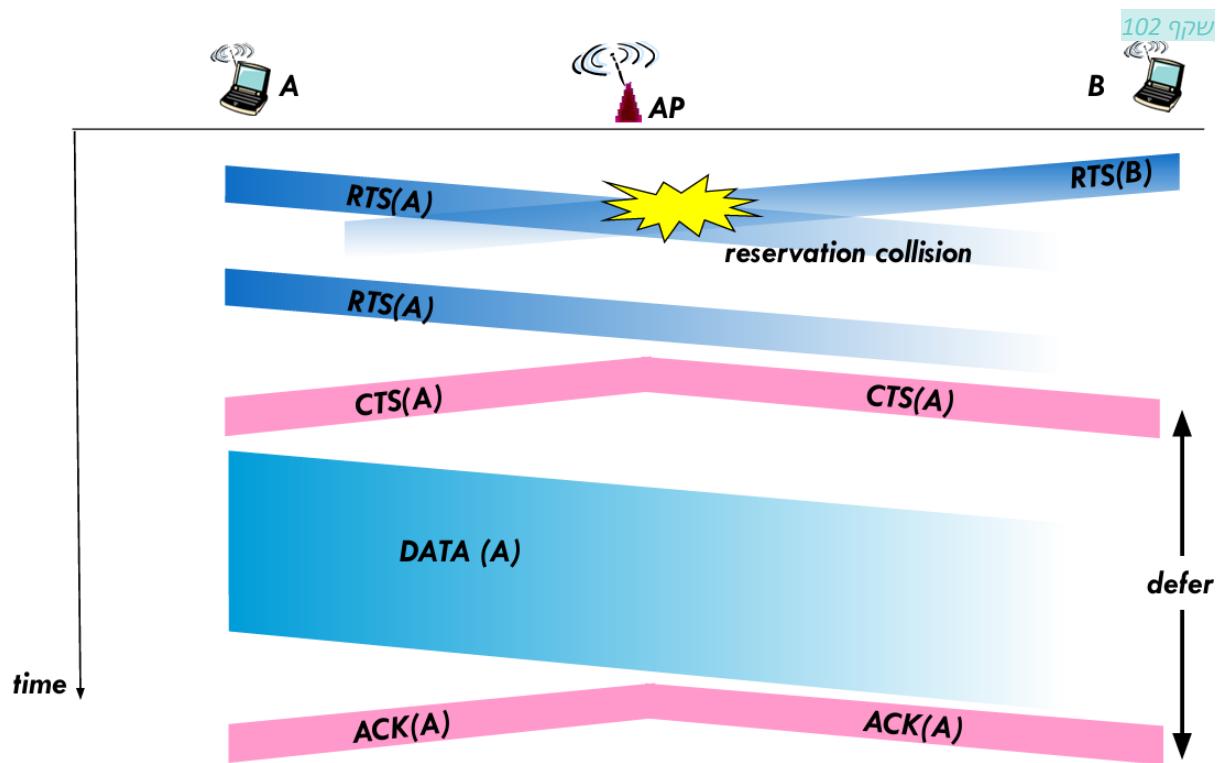
המנגנון: Nachka לעורץ שיתפנה זמן מסוים לפני שמתחלים את השידור (DIFS) ואם הוא לא פניו אז יש מנגנון Short backoff שמעבב אקראיות בדומה ל-CD, והרעיון הוא שנשלח ACK מהמקבל אחרי SIFS (interface space) שזה זמן מאד קצר שהמקבל מחליף לראות שהערוץ פניו לפני שהוא שולח את הה-acknowledgement שלו (הסיבה היא שה-ACK בעצמו זו הודעה, אז הוא מכחיה שהערוץ יתפנה כדי לא לדודס מישחו אחריו. אילוסטרציה:



נדגיש שוב, $DIFS > ACK$, והסיבה לכך היא שאפקטיבית זה נתן עדיפות ל-ACK על פני הودעות data (אם שניים מתחילה לחכות בערך באותו זמן, מי שרוצה לשלוח ACK יקבל עדיפות ובתקופה יגרום לשולח להתחיל את המנגנון מחדש במקומם להתגנש אותו).

שאוף 101

המנגנון הזה עובד אבל הוא לא מבטיח שלא יהיו התנגשויות – הסיכוי להתנגשויות יחסית גבוה. כדי למנוע את הסיכוי להתנגשות הרבה פרוטוקולים ברשתות wireless יש הרבה מכשירים שמחוברים ל-base station אחד, ומה שעושים הרבה פעמים זה להשתמש ב-CSMA/CA כדי לשמור רחוק פס (ברגע שאינו רוצה לשדר אני אשלח request to send לתחנת הבסיס שתפקידו לכל מי שייחולק אליו תוקן broadcast שזכות הדיבור עומדת להיות שלו). הסיכוי להתנגשות בהודעות הראשונות קיים, אבל אם קיבלתי מה-base station הודעה שמאשרת clear to send אני יכול להתחיל לשדר ואף אחד לא ישדר במקרה אליו.



זה מתאים למודל שבו התקשרות עוברת דרך נקודת מרכזית (כמו למשל Wi-Fi router או סלולר).

שאוף 102

xicom – אפשר לסוג את הסוגים של פרוטוקולים של גישה לעורוץ תקשורת משותף ל-2 סוגים:
 1. – בשמחלקים את העורץ לפי תדר או זמן (Circuit Switching).
 2. Random Access – פרוטוקולים לא דטרמיניסטיים שימושיים בהטלה מטבע כדי להוציא את המשתתפים ממצב של סינכרון ולהימנע ממצב שבו כמה משתתפים משדרים יחד (Variants).

Switch

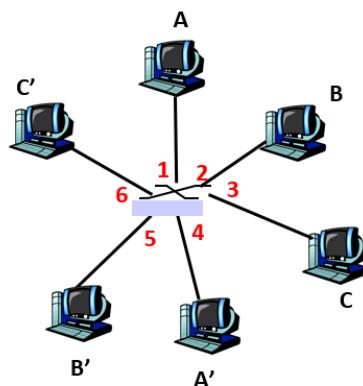
מצגת 3
שקי 3

נרצה לחבר **switches** בדרך יעליה – הדרך לעשות זאת היא באמצעות **broadcast domains**. מחבר במכשיר עם **ports** שמתמחברים אליו בצורה פיזית ויש לו טבלה שאומרת לו בעת קבלת הודעה לאיזו רג'ל/**port** להוציא אותה, ולכל רג'ל יש תור הודעה משלה. זה מאפשר לו לשלוח במקביל על כל ה-**ports** במקביל ו"לשבור" את התנטגשויות. ה-**switch** ממתג את התקשרות כי הוא יודע לאן היא צריכה הגיע.

כל אחת מהרגליים של ה-**switch** מחוברת ל-**broadcast domain** ורץ **CSMA/CD**. הסוייש' שוקף מבחינת התקשרות, מחשב הקצה לא שולח הודעה לסוייש', אלא הוא שם כדי להעביר את ההודעות. הסוייש' שולח הודעה גם בעצמו, אבל הן הודעה ללא תוכן אלא מטרתן למסכון הודעה עם סויישים אחרים (יפורט בהמשך). הסוייש' לומד את הטבלה שהזכירנו קודם מוקדם בלבד (שאומרת לסוייש' לאן להוציא כל הודעה), ככלומר יבין בלבד איפה נמצא כל מחבר קצה ולאן להעביר את ההודעות (אין צורך בקונפיגורציה כדי "להכשיר אותו" למשימה).

כעת נפרט את התכונות של הסוייש':

שקי 4
شبירות broadcast domains



בתמונה יש 6 מחשבים שמחוברים כל אחד לרג'ל אחרית של הסוייש'. זה מאפשר ל-A לדבר עם 'A' ול-B לדבר עם 'B' במקביל בלי שייפריעו זה לזה, בשונה מ-hub שפושט מחבר את כלם דרך צב משותף (ההודעות היי מגיעה לכלם). השימוש ב-**CSMA/CD** היה מזעער את זה, אבל עדין היו בעיות.

שקי 5
טבלת Switch

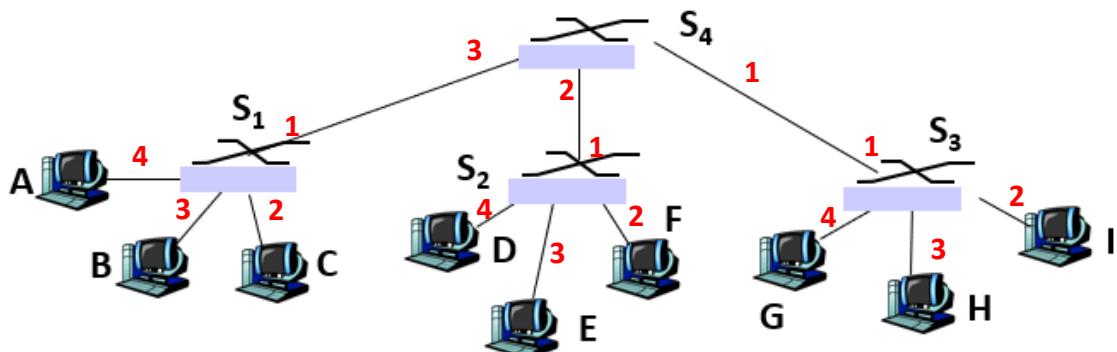
הטבלה פשוטה – כתוב בה MAC Address ואיזו רג'ל משובכת אליה, ו-timestamp שנutan אפשרות לנוקות את הטבלה מ-**entries** לא רלוונטיים. הזכירנו שסוייש' חדש לא דורש קונפיגורציה כדי להתחילה לפעול, אך אם ברגע בוונים את הטבלה? הסוייש' מסתכל על מקור ההודעות שמדובר באיזו וمبין שאותו מקור נמצא מאחוריו. הרג'ל שדרוכה הגיעו הודעה, ומתעד גם (Time to Leave) TTL שכחשה מגע ל-0 ימחק את ההודעה.

אם בתובת היד נמצא בטבלה: יתכן מצב בו המקור והיעד נמצאים באותו domain, מצב זה יתכן (למשל על ידי חיבור ב-hub) ובמקרה זה לא נעשה כלום – הסוייש' לא צריך לטפל בהודעה זו, ויזרק אותה (המקור והיעד מחוברים לאותה רgel). אחרת, יצא את ההודעה מהרגל הנכונה.

אם בתובת היד לא נמצא בטבלה: ההודעה תישלח לכל ה-gateways (כל הרגליים) חוץ מרגל של השולח (כי במקרה זה הוא ישמע את ההודעה בדיק במו שהסוייש' שמע). כשהיעד יגיב (זהו נפוץ) ישמור את הכתובת ויבן להבא מתחזק איזה interface הנקודה עומדת, יידע כבר להחזיר לבתובת המקורית כי התקבלה ממנה הודעה בעבר.

שרשור סוייש'

בארQUITטורה זאת אפשר לשדרר סוייש'. למשל בסוייש' S_4 מאחוריו כל רgel שלו מחוברים 3 מחשבים, למשל A,B,C יהו כלם מאחוריו אותה רgel בטבלה.



התוכנה של self learning נשמרת גם בשדרור הסוייש'. נמחיש באמצעות דוגמה כאשר A שולח הודעה לא-ויזרבן ל-A, ונניח שככל הסוייש' לא מקבלים אף מידע בטבלאות.

A שולח הודעה שמגיעה דרך ה-interface היחיד שיש לו – הסוייש' הראשון של S_1 . כך S_1 למד איפה A נמצא, אבל לא יודע איפה I נמצא וכן שולח אותה לכל הרגליים חוץ מרgel 4 (שמננו ההודעה הגיעה). המחשבים ברגליים 3 ו-2 מקבלים את ההודעה, מסתכלים על ה-MAC Address ורואים שזה ממוקן לא-ויזרבן (לא אליהם) וכן ברטיס הרשות זורק את ההודעה.

מה שמשמעותו קורה דרכן רgel 1, שמננה ההודעה מגיעה ל- S_4 . בבר מההודעה זו S_4 למד ש-A נמצא מאחוריו רgel מס' 3. הוא לא יודע איפה I נמצא וכן עושה שוב fludding להודעה לכל הרגליים חוץ מלזו שדרך הגיעה ההודעה (לא-ויזרבן). מכך גם S_2 וגם S_3 יודעים איפה A נמצא.

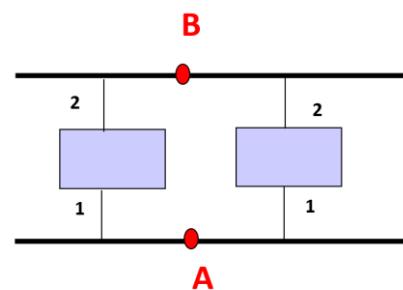
ההודעה מגיעה ל- S_3 שלא יודע איפה I נמצא וכן מבצע fludding גם הוא, G ו-H יזרקו את ההודעה כי היא לא ממענת אליהם – אבל I קיבל אותה ויגיב.

בתגובה, S_3 לומד ש- A נמצא מאחוריו רגל 2, והיעד של ההודעה היא A שאוטו הוא כבר מכיר – יכון לרגל מס' 1, זה הגיע ל- S_4 שילמד איפה A נמצא יכון ל- A ברגל 3, יעבור משם ל- S_1 שילמד איפה A נמצא יכון לרגל 4 (כי כבר מופיע לו A בטבלה).

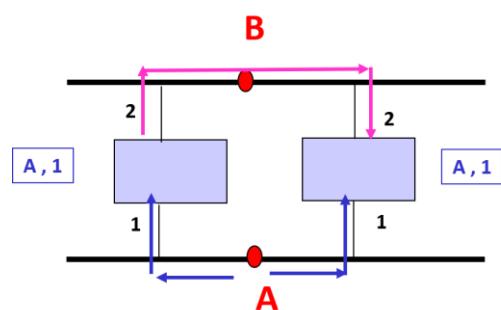
שקל 11 רשת ארגונית

אנחנו נראה מחשבים שמחוברים דרך סוויצ'ים, וסוויצ'ים שמחוברים לsns. אחרים. לחוב יהיה סוויץ' ראשי שמחובר לראטור שמוסמץ את הרשות לאינטרנט (לא הגדרנו עדין ראותר, נגיע לשם), ויכולים להיות שתתיים. בנגד לsns. ראותר איבט שקוֹף (יש לו בתבנית MAC), והוא "שבור" את הרשות שלנו, אך בשנדבר על רשות בשיעור הזה נדבר עד הראותר ולא החוצה ממנו.

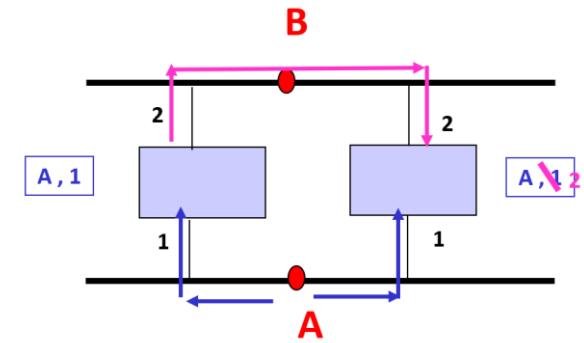
שקל 12 האם יכולות להיות לולאות? כן. נסתכל על רשת די פשוטה:



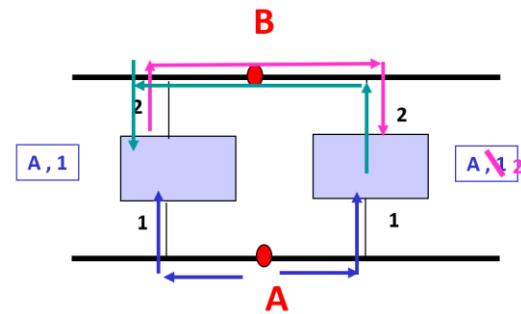
המלבנים הם sns. ולצדיהם מספרי ה-ports, A ו-B הן נקודות קצרה. מה הבעה שיכולה להזעך? A שלוח הודעה (בכחול) ש מגיע ל-2 sns. הsns. הראשון ינסה להזעך אותה דרך רגל השנייה (בורוד):



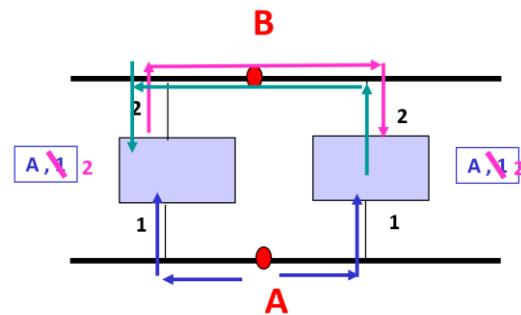
במלבנים אנחנו רואים את טבלת sns. כל sns. למד ש-A נמצא מאחוריו רגל מס' 1 (ה-TTL קיים אבל לא רלוונטי לצורך המבחן). הבעה היא שעכשיו sns. הימני חושב שהוא קיבל את ההודעה מ- A ו- A כנראה עבר מקום אז הוא מעדכן את המיקום שלו בטבלה:



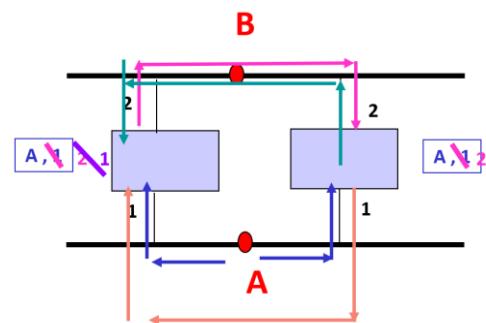
עבשו יש לו הودעה שהוא קיבל מקודם (הכחולה), והוא רוצה לשלוח אותה עבשו (בירוק):



עבשו היא מגיעה לסויץ' השמאלי, וגם הוא חושב ש-A עבר מקום אז מעדקן שהוא נמצא מאחרי רגל 2:



הסויץ' הימני קיבל הודעה (בווורוד), אז הוא צריך להוציא אותה מהרגל השנייה (בכתום) ושוב נעדכן בתובות:

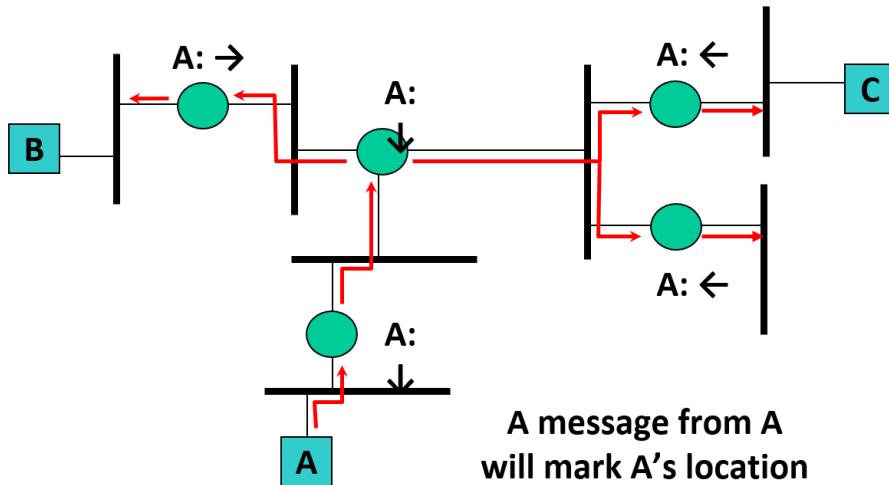


וכך קיבלנו הודעה שמסתובבת במעגל – בזבוז משאבי רשות.

STP (Spanning Tree Protocol)

שאוף 18

הינו רצים רשות שהוא עץ בו לא יכולות להיווצר מעגלים. העיגולים הם סוויצ'ים, הקווים המודגשים הם broadcast domains והריבועים מבשיiri קצה:



שאוף 19

המטרה היא לבנות עץ. איך נבנה את הטופולוגיה של הרשת כך שתיה בזו?

אפשרה ראשונה היא קיומם Network Administrator וUIL מודע עם צוות IT שנזהר לא לחבר בבלים במקומות שעולמים ליצור מעגל – קשה לביצוע, וגם אין בעז redundancy (ברגע ש-port מתנתק הרשת מתחלקת לשני חצאים). לא הינו וחיצם הגבלה פיזית על החיבורים, מעגלים יכולים לאפשר כמה דרכים להגעה באותו יעד. נרצה לשפר את הגישה זו.

נתאר פרוטוקול שהסוויצ'ים יריצו בין עצמם שטורתו קבוע איזה interfaces – בטל בכל סוויץ' – לא באמצעות ניטוק פיזי, אלא רק בזוורה לוגית נודא שכל סוויץ' יבבה חלק מהഫוטרים וכודא שלא יעביר תקשורת דרכם. תצורת הרשת תהיה גרף, אבל מבחינה לוגית יתקיים עץ פורש (אפשר להגעה מכל מקום לכל מקום).

שאוף 20,22

פרוטוקול עצים פורשים – Spanning Trees Protocol (STP)

הפרוטוקול זהה הוא לב רשתות Local Area. ב프וטוקול זהה הסוויצ'ים עצם שלוחים הודעות שטורתן היא לקנפוג את העץ. הסוויץ' ישלח הודעה שיעדתה הוא סוויצ'ים אחרים שמחוברים אליו. להודעה קוראים BPDU (Bridge Protocol Data Unit) ויש סטנדרט שמנדרט איך היא נראהות ואיך תטופל (IEEE 802.1D).

מטרות הפרוטוקול: קבוע אילו סוויצ'ים לבטל – כל סוויץ' ידע דרך איזה interfaces הוא לא יעביר דרכן הודעות למשתמשי קצה. כדי לעשות זאת, צריך לענות על 3 שאלות לפי הסדר הבא:

1. מי שורש העץ?

2. איך ליצור spanning tree של סוויצ'ים?

3. איזה סוויץ' יחבר את ה-interface? (כל האחרים שמחברים את הדומיין יבטלו את ה-interface)

שאוף 23

הפרוטוקול זהה הוא חלק ממשפחה גדולה של פרוטוקולים שנקראים Self-Stabilizing Protocols – פרוטוקול מבוזר שרצ על ידי מכשירים לא מתאימים ומתאים להוביל את הרשת למצב יחיד (למשל, לוודא שבכל הסוויצ'ים השתמשו באותו העץ).

בעת ננעה על השאלה:

שאוף 24

1. בחירת שורש לעץ

מניחים שלכל סוויץ' יש מזהה ייחודי. כל סוויץ' ישלח את ה-BPDUs דרך כל ה-interfaces שלו, ואחת השדות יציין מה ה-pid של הסוויץ' שלדעתו הוא שורש העץ. במקרים אחרים, נסכים למשל שה-pid הביא נマー ברשת הוא ה-root, ואני אשלח שאני השורש. אם אני מקבל מאחד השכנים שלי בתובת נמוכה יותר, מעתה אני עדכן שזה ה-root לדעתי. ה-pid של השורש עומד לחחל לכל הרשת עד שתתקבל הסכמה.

שאוף 25-30

2. חישוב עץ פורש של סוויצ'ים

אנחנו עומדים להשתמש **באלגוריתם בלמן-פורד**. המטרה היא שבל סוויץ' ימצא את ה-interface שמחובר למסלול הקצר ביותר לשורש. חוץ מלשלוח מה ה-pid של השורש, נשלח גם את המרחק שלו ממנו. ככלمر אם אני למשל חושב שאני השורש אשלח את ה-pid שלי ומרחק 0. אחרת אם קיבלתי ממישהו אחר pid יותר נמוך, עדכן את בתובת ה-pid של השורש להיות זו שראיתי עכשווי, ובמרחק אוסף +1. כל סוויץ' יקח את המינימום מכל השכנים שלו – וזה המרחק מהשורש. ה-interface שמננו קיבלתי את המרחק המינימלי יהיה ה-parent pointer שלו, המצביע לביון השורש. [דוגמה רצאה בשקפים 27-30](#).

הפורש של הסוויצ'ים (לפחות עד עדכון נוסף). לינקים שאינם מודגשים יכולים להתבטל (לא פיזית, רק לוגית).

שאוף 31

האלגוריתם עובד וכולם מתכנסים לאותה תוצאה אפילו כשהואעובד בצורה א-סינכרונית אצל כולם. לא משנה מה סדר ההודעות, כל סוויץ' ידע לזרות מה הלינק שלו לשורש.

שאוף 32

3. בחירת switches שיחובר ל-interface broadcast domain

לכל סוויץ' יש פוינטර אחד לשורש, אבל אם יש broadcast domain שמחובר לכמה סוויצ'ים עדין יש לנו מעגל (הודעה שתשלוח על אותו link). תצא החוצה דרך שני מסלולים נפרדים שמגיעים לשורש מאותו מחשב). המטרה היא בכל broadcast domain להشير רק port אחד בסוויץ' אחד שיחבר אותו הלהה לרשת. איך מבצעים זאת?

בשולחים UDPU אנחנו שולחים אותה לכל הרגליים של הסוויץ' – ככלומר אם יש שני סוויצ'ים שמחוברים לאותו broadcast domain הם ישמעו את ההודעות אחד של השני. לבן, נסיף להודעה גם את ה-pID של הסוויץ' השולח, וכל סוויץ' יבחר אם הוא צריך לנתק את הרגל שלו לאוטו חילוק broadcast domain לפי ההודעות שקיבל. אם הוא שמע הודעה מסויז' שמחובר במסלול קצר יותר – לשם יחבר את הרגל שלו. אם שניהם באותו אורך נחברו שוויין לפי pID ייחודי (נתעדף pID נמוך). יתכן מצב שבו 2 ports מאותו סוויז' מחוברים לאותו broadcast domain (ואז יש שוויין גם ב-pID) – במקרה זה נעדיף את ה-port עם ה-pID הנמוך יותר.

.broadcast domain – המונח שנבחר עבור ה-port שדרכו הסוויז' בחר להתחבר ל-chain **Designated port**

לכל broadcast domain נדרש להיות root port אחד, ולכל סוויז' נדרש להיות designated port אחד (מצביע אחד לשורש).

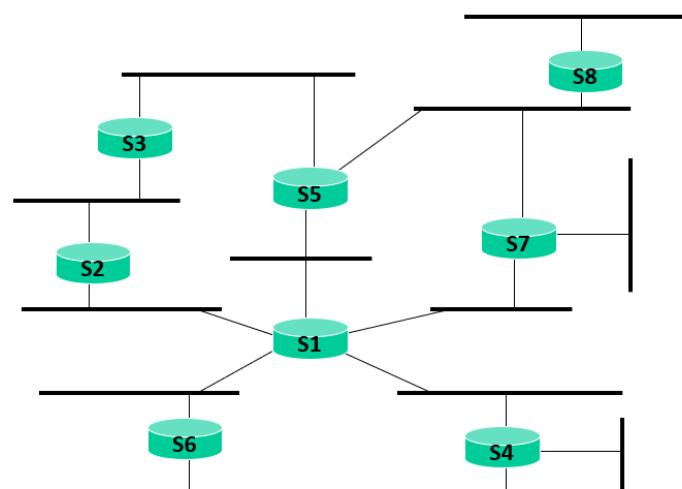
אם יש סוויז' ללא designated port (לא לחבר אף broadcast domain) אז אפשר לנתק את כל הסוויז' (לבבות את ה-root שמצויע לעבאי).

שקל 34

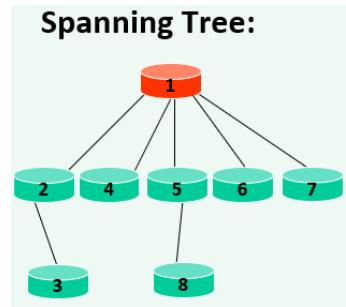
סיכום – לכל סוויז' חוץ מהסוויז' שורש יש port Root port (ה-port שמצויע לכך שורש העץ שנבחר, זה שהוא עם המסלול הקצר ביותר לשורש) ולכל broadcast domain יש port designated אחד (ה-port שמעביר את ההודעות מאותו broadcast domain לבניון השורש).

שקל 35-36

דוגמא: S_1 הוא הסוויז' עם ה-pID הנמוך ביותר, אך הוא יבחר להיות השורש:

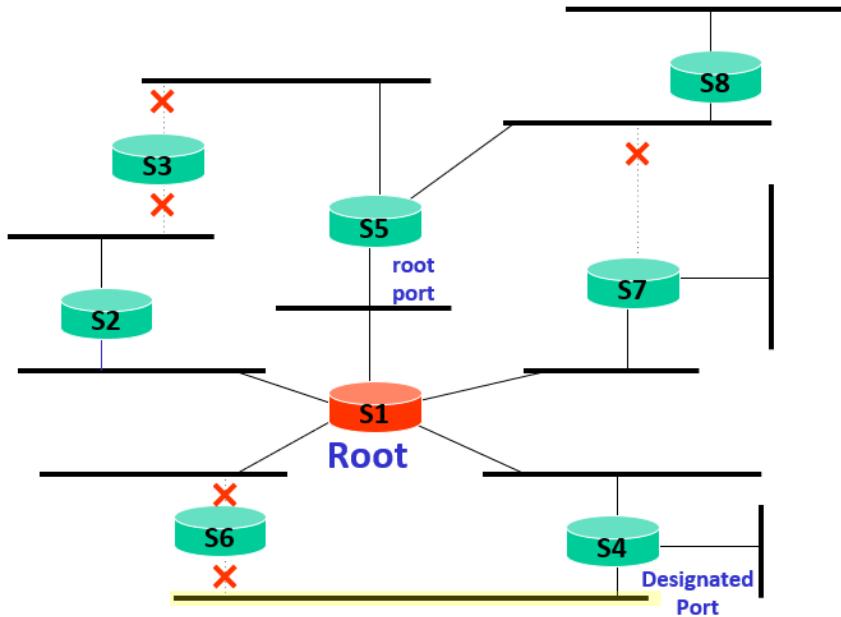


המטרה היא לקבל מבחינה לוגית עז שנראה כר:



שאלה 38

למשל ב-domain broadcast התיכון (מסומן בצהוב) אנחנו רואים שהוא מחובר על ידי 2 סוויצ'ים (S_4, S_6) שמחברים ישירות לשורש (S_4 ו- S_6 שניהם במרחק 1 מהשורש), אבל אחד מהם עם פו נמור יותר. לכן מי-שייבחר להיות designated port הוא S_4 , וסוויצ' S_6 ידע זאת ולכן יקבע את ה-port שלו שמחבר אותו לbroadcast domain. נוצר מצב של- S_6 אין יותר port designated, כלומר הוא לא מחבר אף אחד ולכן כל הפורטים שלו, כולל ה-port root, יכולים להיסגר.



בך לדוגמה נסתכל גם על S_3 , גם הוא ביטל את ה-port designated שלו, כי קיים מסלול קצר יותר לשורש שעובר דרך S_5 . למה את S_7 לא מבינן? כי הוא עדין עם port designated (ל广播 domain שנמצא מימין בתרשים). יש פרטום מהחת הרגילים שלו ולכן לא מתইיטה.

אם לסוויצ' אין designated port זה אומר שהוא לא יעשה תעבורת למכשירי קצה, אז אפשר להפסיק תעבורת מהסוויצ' לגמרי (ורק ישתתף בתעבורת הודעות בעץ). לא מנטקם לגמרי כי אם מצב הרשות ישנה בעתיד, אולי יהיה לו שימוש.

CSMA/CD (Carrier Sense Multiple Access/Collision Detection)

בעבר ראיינו 2 וריאציות של פרוטוקול ALOHA. נרצה לשפר את ה-goodput שקיבלנו עבור הוריאציות האלו.

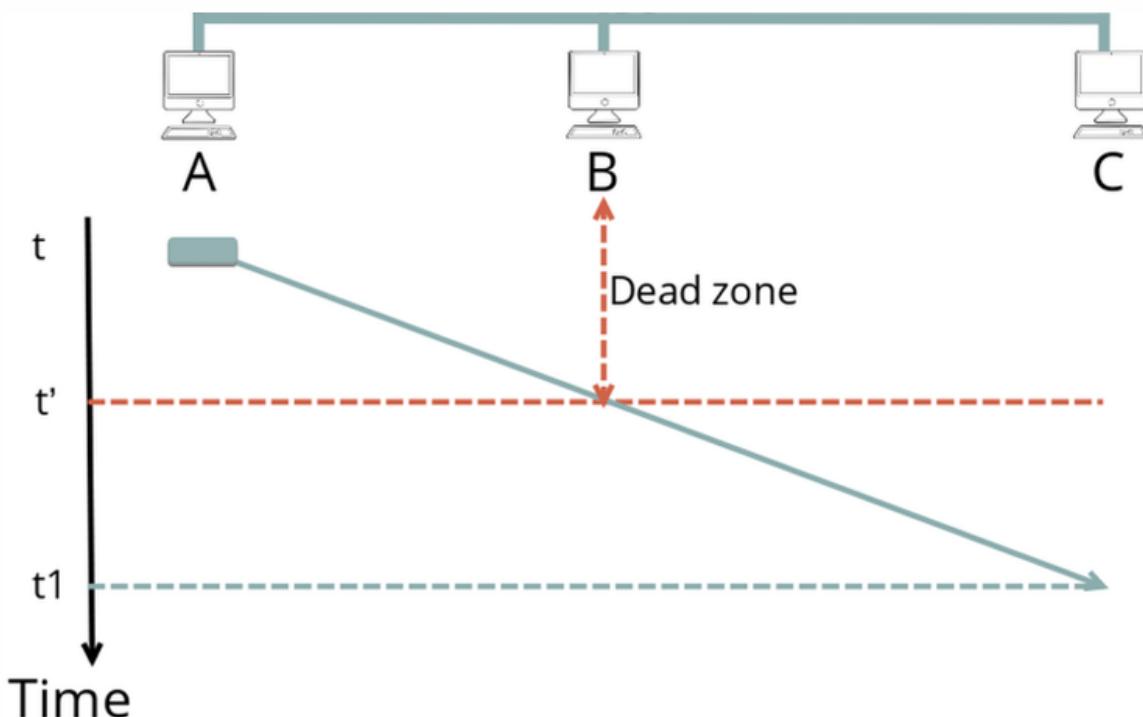
הציגת הפרוטוקול

נציג פרוטוקול חדש שנקרא CSMA. השינוי במסגרת הפרוטוקול זהה לעומת הקודמים, הוא שעכשו אונחנו מNICים שלצמתים יש אפשרות להקשיב לפני שהם מתחילה לשדר. אם הצומת שומע מישחו שמשדר אפשרותו לפעול בהתאם.

בהתאם להנחה הנוספת, איך יוכל להימנע?

- **לקשיב:** אפשר לבדוק האם התווך פניו טרם השידור.
- **لتזמון:** אפשר לקבוע תזמון מתי כל צומת תורשה להשתמש בתווך (למשל זוגי או-זוגי).
- **לחילק תדרים:** אפשר לחלק את התווך לבמה תדרים, לא להתעסק בשיטה זו.

במציאות בשימושו משדר לוקח זמן עד שהצד השני יקבל את הודעה. נניח ש-A משדר בזמן t , ו-B גם רוצה לשדר. B יקשיב, אבל בזמן t הוא לא ישמע שום דבר כי לוקח הודעה מעט זמן לעובר. הודעה מגיעה ל-B בזמן t' ורק בזמן t_1 מגיעה ל-C:



יש טווח שבו B מקשיב אבל לא יידע ש-A שידר (Dead zone).

אם כך, לא מספיק רק להקשיב לתווך ולשדר כשנראה שהכל פניו. נגדיר משנתנה T_{prop} שיסמן את הזמן הארוך יותר בין שני הקצאות (בתרשים למעלה, $t - t_1 = T_{prop}$). גם אם קודקוד מקשיב לפני שהוא משדר, במשך T_{prop} השניות הראשונות עלולה להיות בעיה.

גישה שונה לשילוח הودעות

נכיה שעבר T_{prop} שנויות וצומת רוצה לשדר אבל יש רוש, متى הוא ינסה לשדר?

Approach	Free?	Occupied?	Intuition
1-persistent	Send!	Wait until the channel becomes idle and then send immediately	High chance of collisions in loaded channels
non-persistent	Send!	Backoff and try again	Lower utilization on low load
p -persistent	Send with prob. p	Wait until the channel becomes idle and then send with probability p	A compromise between the other approaches

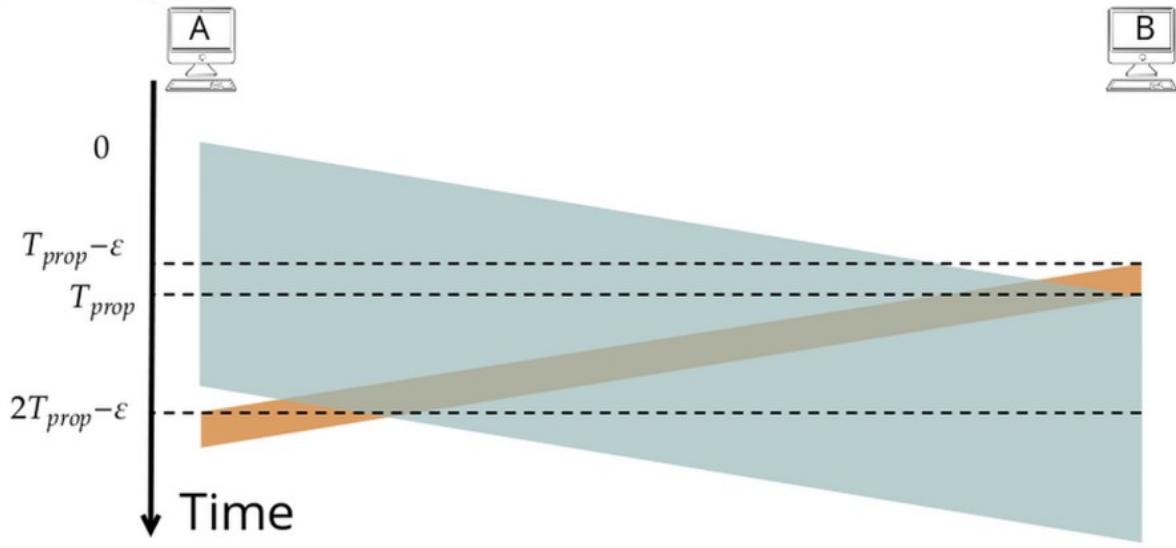
הצומת מקשיב – ברגע שהוא שומע שאין דיבורים ברשות הוא מיד ישדר. متى זה עיל? בשאי הרבה הودעות שרוצים לשלוח.

אם יש במאם קודקודים שרוצים לדבר תוק כדי אבל אין לנו דרך לדעת כמה ידברו במקביל. יש לנו שיאפשר לנו לחכות יותר זמן אם יש עומס, ולהחות פחות זמן כאשר אין עומס. מנגד, זה אומר שאם חיכינו הרבה זמן ואז נגמר העומס אז לא יהיה ניתן טוב של התווך.

יש לצומת הודהה לשלוח. ברגע הוא שומע שהתווך תפוס אז הוא ממתיין, אבל ברגע שהתווך פניו לא מיד נשדר, אלא נשדר בהסתברות p . יפורט עוד [במהשך](#).

Collision Detection

נסיף למגנון שציגנו גם CD. צומת מחכה ומזהה שהתווך פניו. ראיינו למעלה שזה לא מבטיח שלא תהיה התנגשות. ייקח לנו T_{prop} שנויות כדי להזות שהשידור שלנו התנגש עם שידור של מישהו אחר. לא מספיק שאנו נשים לב שיש התנגשות, אלא נצטרך להודיע לכלום על ההתנגשות. ננסה לשדר שוב אבל רק אחרי SCNODA שהתווך פניו.



הסבר לתרשים:

מחשב A מתחילה לשלוח בזמן 0 והשידור שלו מגיע למחשב B בזמן T_{prop} . נניח שבמקרה הגורע ביותר B הקשיב, ראה שהתווים פנו והחליט לשדר בזמן ϵ – T_{prop} . ייקח עוד T_{prop} שניות עד ש-A יידע ש-B שידר הודעה ויבין שהייתה התנגשות. סה"כ לוקח $2T_{prop}$ שניות מרגע השידור הראשון עד שהמערכת הצלילה להזזה שקרתת התנגשות.

מאחר ש-B גם מאזין, ברגע שהוא קיבל הודעה מ-A הוא יפסיק לשדר כי הוא יודע שאין סיבה להמשיך (ההודעות לא יעברו בהצלחה). נקודת המבט של A הוא שומע רעש אבל ברגע שהרעד נגמר הוא יכול לנסות להתחילה לשדר שוב (לא באופן מיידי, תלוי באיזה אלגוריתם להשתמש). ב-B לעומת זאת ייקח כמה זמן עד שambilינו התווים והזזה יהיה שוב ריק.

סיכום התרשים:

מניחים שהזמן שלוקח להזזה מעבר מ-A ל-B הוא T_{prop} שניות. A מתחילה לשדר בזמן 0, B מתחילה לשדר בזמן ϵ – T_{prop} . A יזהה את ההתנגשות רק בזמן ϵ – $2T_{prop}$, שכן כל הדיזה נמשך לפחות לפחות $2T_{prop}$ שניות. **מסקנה** – כדי שפרוטוקול CSMA/CD יעבוד צריך שא-frames יהיו באורך המקיים $T_{trans} > 2T_{prop}$, כלומר זמן השידור של ההודעות יהיה גדול יותר מהזמן שלוקח להזזה להתקבל מצד השני (ולחזר).

CSMA/CD – Jam Signal

אם אני מדבר ותווך כדי שידור שמעתי שהגע עוד שידור לא מוצלח, אני אדע שהייתה ההתנגשות. הבעיה היא שימושו שליש (ללא שידר, רק הקשיב) קיבל רצף של ביטים שהוא לא יודע אם השילוב שלהם תקין או שהוא שילוב של שתי הודעות.

לשם כך יש **Jam Signal** – דרך להודיע לצמותים אחרים שהתרחשה התנגשות, למשל בערוץ אחר. יישלח על ידי מזזה ההתנגשות. ברגע שהצומת שהתחילה את השידור שגרם להתנגשות קיבל את ה-Jam Signal הוא גם יפסיק לשדר, וגם יידע את כולם שהמידע שהם קיבלו הוא לא תקין.

Waiting (Backoff) Time

אחרי שהייתה ההתנגשות והבנו שקרה משהו לא תקין, משתמשים בגישה שנקראת "Exponential Backoff". נניח ששידרתי משהו וקיבלו Jam Signal, ניסיתי שוב וקיבلت עותה סיגנלשוב. עכשו בברआתחיל לחכות יותר זמן, ובכך אני מעלה את הסבירות שאחרים יצילחו לשדר ואז בשאני אשוב לשדר אני לא אפריע להם והם לא ל'.

ה-backoff תלוי בפערמטר c (לרוב $2 = c$). אחרי הניסיון ה- k להעבר הודעה, נבחר j באופן אחד מຕוך הטווות $\{1 - c^k, \dots, 0, 1\}$ וنمתיין $T \cdot j$ יחידות זמן. כאשר הצומת מצליח לשלוח את ההודעה הוא יאפשר את מספר הניסיונות שלו (k) ל-0. ה-counter של צמותים אחרים לא ישפיע מזזה.

הערות:

1. למה בוחרים מຕוך טווות $\{1 - c^k, \dots, 0, 1\}$ ולא פשוט מחייבים $1 - c^k$? כי אם שני צמותים A ו-B מנסים לתקשר באותו הזמן עם צומת C, לבוחר תמיד אותו ערך רק ימשיך את גלגל ההתנגשויות. בממוצע מחייבים יותר זמן, אבל מוגדים את הסיבוכו להתנגשות.
2. יש גם גישות אחרות לטיפול ב- k אחרי שליחת הודעה מוצלחת, אבל זו הנפוצה.

p-persist

נניח שכרגע זההiji יש התוווע ריק. במקום לשדר מיד, אני אטיל מטבע ואשדר בהסתברות p . אם זכיתי בהגירה אשדר, ואחרת אני אנסה שוב בסלוט הבא. אם יש הרבה קודקודים במערכת שמחכים לשדר אחרי שהוא שידור ארוך, יש עדין סיכוי שרק אחד מהם ינסה לדבר (לעומת persistent-1 שם אחרי שידור ארוך כלם ינסו לשדר יחד ואף אחד לא יצליח).

כאן לא יהיה את המושג של Exponential-Backoff.

סיכון הנחות והגדירות

הנחות הראשית:

T_{prop} – משך הזמן שלוקח להודעה להגיע מקודקוד אחד לקודקוד אחר למרחוק ביותר.
 N קודקודים במערכת.
 T_{trans} זמן שיודה.

הנחות פרוטוקול:

מניחים שהשלוט הוא באורך $2T_{prop}$ שניות.

תמיד יש לצמתים הודעה לשדר.

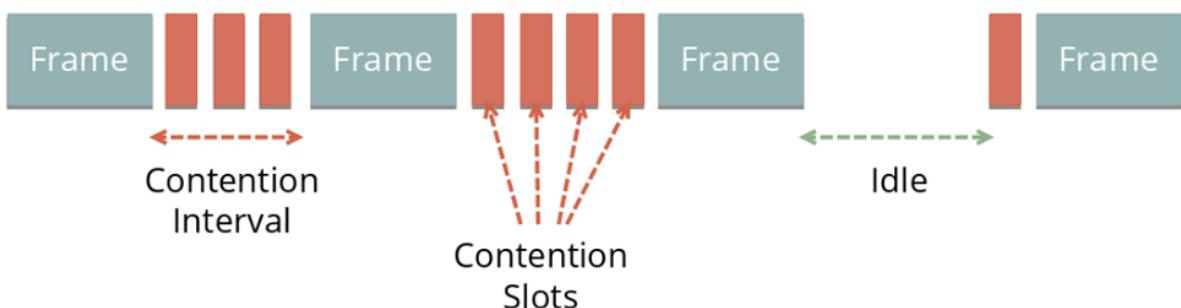
הסתברות שליחה μ .

CSMA/CD Analysis

בפרוטוקול p-persistent אנחנו נקייב ונכח עד לסלוט הבא אם ברגע התווך לא פנו, כאשר יתפנה נshedר בהסתברות μ , ובמהלך השידור אנחנו נקייב האם מתΚבל Jam-Signal (או להודעה של מישחו אחר אם הוגדר כך). ברגע שהבנו שהייתה התנגשות נפסיק לשדר, ובסלוט הבא נshedר שוב בהסתברות μ .

מהם מצביו המרכיבים האפשריים?

- Idle – אף אחד לא משדר.
- Transmitting – יש צומת שמצליח לשדר ללא הפרעות.
- Contention – יש שני קודקודים שמשדרים בהפרש קטן T_{prop} , וידעו על ההתנגשות אחרי כל היוטר $2T_{prop}$ ייחידות זמן.



чисובי goodput:

חישוב הסתברות הצלחה יהיה דומה לשל ALOHA : Slotted ALOHA

$$\mathbb{P}_{suc}(p) = \mathbb{P}(X_p = 1) = np(1 - p)^{n-1}$$

$$\max_{p \in [0,1]} \mathbb{P}_{suc}(p) = \left(1 - \frac{1}{n}\right)^{n-1} := S^* \text{ ו } p^* = \frac{1}{n} \text{ נקבל:}$$

בהינתן הנתון שגודל סלוט הוא $2T_{prop}$ קיבל תוצאה טוביה יותר לעומת Slotted ALOHA כי אכן אנחנו מקבלים דרך להפסיק שידור ברגע שזיהינו התנגשות.

מה ההסתברות שהיא k סלוטים בהם לא יהיה שידור מוצלח/התנשאות שלאחריהם הגיעו לשידור מוצלח?

זה מתפלג כמו משתנה מקרי גיאומטרי עם פרמטר S . לכן התוחלת תהיה $\frac{1}{S}$ כלומר $1 - \frac{1}{S}$ סלוטים מבוזבים.

בזמן שלא שידרנו הפסemo את האפשרות לשדר במשך $2T_{prop}$ שכנות (גודל סלוט). נחשב ונקבל:

$$\begin{array}{c} \text{זמן שנאבד} \\ \text{על שידור לא מוצלח} \\ \overbrace{2T_{prop}}^{\text{כמות סלוטים}} \quad \left(\frac{1}{S} - 1 \right) \\ \text{לא מוצלחים ממוצע} \end{array}$$

ולכן נוכל לחשב את ה-goodput בצורה הבאה:

$$\eta = \frac{\text{Transmission Time}}{\text{Transmission Time} + \text{Wasted Time}} = \frac{T_{trans}}{T_{trans} + 2T_{prop} \left(\frac{1}{S} - 1 \right)}$$

באשר נשאיף $\infty \rightarrow n$ נקבל את ה-goodput המקסימלי: $\left(\frac{1}{S} = e \right)$

$$\frac{T_{trans}}{T_{trans} + 2T_{prop}(e-1)} = \frac{1}{1 + \frac{2T_{prop}}{T_{trans}}(e-1)}$$

אנחנו רוצים להתקרב ל- $1-\text{goodput}$. איך נשיג זאת?

- להשאיף את T_{trans} לאינסוף (כלומר יש לי הודעה ארוכה מאוד לשולח, בהתחלה בזמן T_{prop} תהיה בעיה – אבל לאחר מכן ברגע שנעבור אותו אני יודע שאף אחד לא יתngeש בו).
- להשאיף את T_{prop} ל-0 (כלומר אנחנו מאוד קרובים אחד לשני ושנינו רוצים לשדר, יתכן שבדוק באותו זמן שנינו נרצה לשדר ואז תהיה בעיה. אם יש הפרש בלשחו והוא קטן מאוד אז נשמע שימוש אחר שידר ולא נתngeש, כלומר הזמן שיש בו סכמת התנשאות מזער).

מה קורה ב-ALOHA Slotted ללא CSMA/CD ?

לפעמים יעזר ולפעמים לא, נראה דוגמאות.

CSMA/CD – Questions

נתונים

יש לנו מערכת CSMA/CD עם רוחב פס propagation speed, 10Mbps של $\left[\frac{m}{s} \right]$. נוסף רשת דומה למרחק 20 km מזו שלנו עם תווך כמו זה שהגדכנו למערכת. בתוך אותה רשת הקודקודים יחסית קרובים אחד לשני.



שאלה 1

האם בהינתן הودעה בגודל **64Bytes** נוכל לנצל את התוכנה של CSMA/CD?

זמן שלוקח לبيט בודד לעבור בין שתי הרשותות (נוסחת זמן- מהירות- מרחק):

$$T_{prop} = \frac{\text{distance}}{\text{propagation speed}} = \frac{20Km}{2 \cdot 10^8 \frac{m}{s}} = 10^{-4}s = 100\mu s$$

זמן שלוקח להעביר 64B הוא:

$$T_{trans} = \frac{\text{bits}}{\text{bits per seconds}} = \frac{8 \cdot 64}{10 \frac{Mb}{s}} = \frac{8 \cdot 64}{10^7 \frac{b}{s}} = 51.2\mu s$$

נבחן שמתקיים:

$$T_{trans} < 2T_{prop}$$

כלומר לא מתקיים התנאי, ולא נוכל להשתמש בהודעה הזאת ברשות ולהשתמש ב-CSMA/CD בצורה שתתנו לנו את הבעיות המגויות עם הפרטוקול. יכול להיות מקרה שבו מישחו משדר משווה באורך 64B ורך בסוף השידור גילה שהיאיטה בעיה, אבל הינו רצים לעצור את השידור מיד.

שאלה 2

מה השינוי הקטן ביותר שאנו יכולים לעשות כדי לפתור את הבעיה הקודמת?

אנו צריכים שהפקטה תהיה מספיק גדולה כדי שייקח לי יותר זמן $m-\mu s$ $2T_{prop} = 200$ כדי לשדר אותה. נחזיר לנוסחה הקודמת אבל הפעם עם נעלם. אם נגדיל את גודל ההודעה פי 4 לגודל 256B נקבל $T_{trans} = .204.8\mu s$

שאלה 3

מה ישתנה בשאלת הקודמת אם רוחב הפס היה **100MBs**?

רוחב הפס גדול פי 10, אפשר לשדר פי 10 יותר בזמן נתון, כלומר T_{trans} קטן פי 10. אם בעבר התקיימנו $T_{trans} > 2T_{prop}$ נרצה לשמור את אי-השוויון הזה, אך נגדיל את גודל הפקטה פי 10, כלומר להיות 2560B.

שאלה 4



בשאלה זו יש 2 T_{prop} : אחד שאומר כמה זמן לוקח להגיע מ-A ל-C, והשני שאומר כמה זמן לוקח להגיע מ-A

$$\cdot B = 3 \left[\frac{mb}{s} \right] \cdot 10 \cdot 6 = \tau, \text{ ורוחב הפס הוא } \left[\frac{m}{s} \right]$$

מה גודל הפקטה המינימלית (x) ש-A יכול לשלוח ל-C?

אנחנו רוצים לנצל את CSMA/CD ולכן נרצה שאי השווין יתקיים.

$$T_{prop} = \frac{\text{distance}}{v_{prop}} = \frac{10km}{6 \cdot 10^7 \frac{m}{s}} = \frac{1}{6} 10^{-3}s$$

$$T_{trans} = \frac{x}{B} \geq 2T_{prop} \iff \frac{x}{3 \cdot 10^6} \geq \frac{1}{3} 10^{-3} \iff x \geq 10^3 bit$$

מה גודל הפקטה המינימלית (x) ש-A יכול לשלוח ל-C?

מדובר בשאלה מעט מכשילה, גם במקרה זהה התשובה צריכה להיות $10^3 bit$. הסיבה לכך היא שלמרות ש-

B קיבל את ההודעה קודם, C עדין יכול לשלוח הודעה ל-B שתפריע, או ש-A ישלח הודעה לכל המערכת

(בשבבה 2 הודעות נשלחות לבולם), בעוד B קיבל את ההודעה ויבין אותה, אבל בזמן זהה C אולי יתחל לשלוח

גם כן (לא יודע ש-A התחיל לשדר) ולכן יכול להפריע גם כן.

שאלה 5

יש לנו מערכת ש谋ריצה Slotted ALOHA ומקיימת את אי השווין $T_{trans} > 2T_{prop}$. גודל סלוט הוא

האם להוסיף CSMA יספר את ה-goodput?

לא. הסיבה לכך היא שהבדיקה שהתווך פנוי מתבצעת תמיד בתחילת-slots. נניח ש-A שידר. גם אם B שידר

וגם אם לא, להודעה של A בכל מקרה ייקח T_{trans} ייחיות זמן להגיע ל-B, זה לא יהיה עוזר לו להקשיב כי בכל

מקרה אם שניהם החליטו לשלוח הסלוט בוובז, יוכל הקששה לתווך לא תסיעו כאן.

האם להוסיף CSMA/CD יספר את ה-goodput?

לא. נניח ש-A התחיל לשדר ואחד מהם הבין שיש הודעה, שלח Jam Signal ואז A הפסיק לשדר. כך או כך,

הסלוט מבזבז והשידור שלו היה מתbezבז.

אילו המערכת הייתה Pure ALOHA, האם עבשו הוספה של CSMA תשפר את ה-goodput?

כן. לאחר שה솔טים לא מסונכרנים, הקשבה לתווך עשויה למנוע למשל מ-B להתחל שידור באמצעות שידור של A.

אילו המערכת הייתה Pure ALOHA, האם עבשו הוספה של CD\CSMA תשפר את ה-goodput?

גם כן. אם שניהם שידרו כי למשל B ראה שהתווך פנוי (ההודעה של A עוד לא הגיעה), אם הוא יספק לשולח Jam Signal לפני ש-A מתחילה lot הוא יוכל לנצל את הسلط הבא.

מסקנות

ב-CD/CSMA אנחנו מחכים שהתווך יהיה פנוי, מזהים התנגשויות ומודיעים עליהם באמצעות signal jam, נחכה זמן רנדומלי לפני ניסיון שידור חוזר. נשתמש ב-CD/CSMA כאשר זמן ה-(E2EP) (End-to-end propagation) קצר יותר מאשר הזמן transmission.

Errors Detection and Recovery



אנחנו שולחים פקטה עם ביטים, לעיתים יש שיבושים בדרך. עם coding טוב נוכל לזהות שהיא שיבוש, ולפעמים אפילו ל恢復 חלק מהנתונים. השיטה תהיה דומה למספרת ביקורת בתעודת זהות. ניקח את המידע שאנו רוצים לשלוח, נכין מזה מספר בקרה ונוסף אותו לסוף הפקטה. קיבלנו פקטה חדשה שמכילה את המידע בתוספת coding correction. בכל שносיף יותר ביטים לבקרה נקבל יותר תקורה.

Repetition code

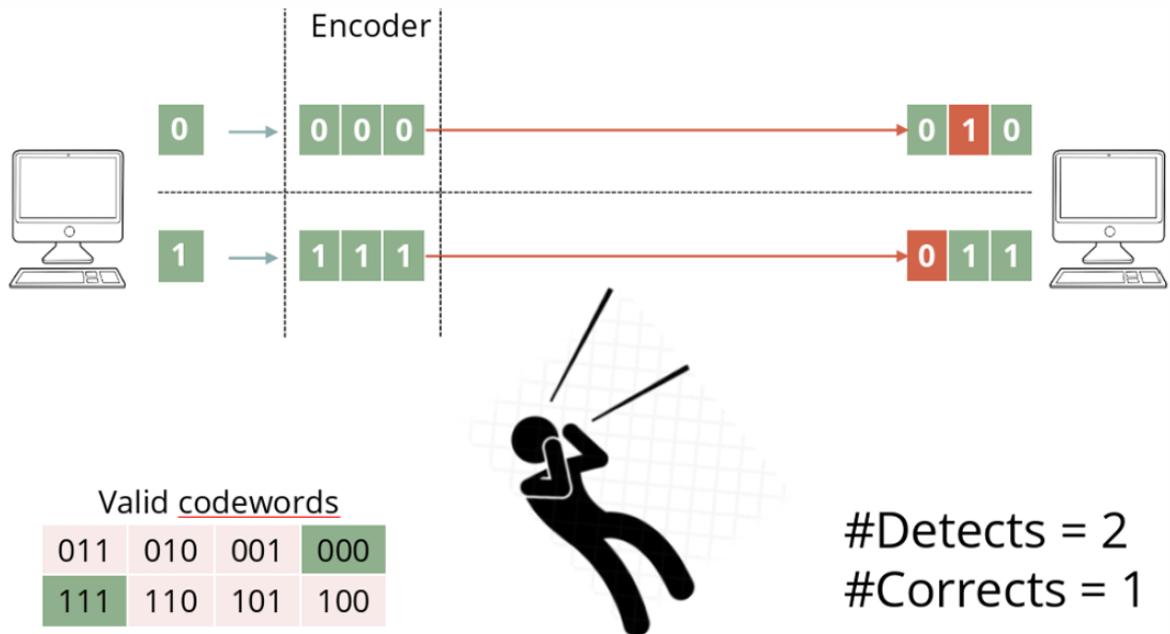
כל הודעה שנרצה לשלוח, במקומות לשלוח פעם אחת נשלח אותה 3 פעמים. אם היו א' ביטים בהתחלה עבשו יהיה א' 3 ביטים. חישוב התקורה יהיה באופן כללי:

$$\frac{\text{Actual} - \text{Original}}{\text{Original}} = \frac{\text{Actual}}{\text{Original}} - 1 = \frac{\# \text{ Redundant Bits}}{\# \text{ Original Bits}}$$

ועבור המקרה של 3-repetition

$$\frac{3N}{N} - 1 = 2$$

איך מתרבצע התיקון? לפי מה היה הרוב:



אילו 2 ביטים היו לא נכונים, היינו משחזרים בצורה לא נכונה.

1D Parity

עדין, מדובר על המונח **tosfot** בשיטה של repetition-3. מה שנבחר לעשות במקום היא אריתמטיקת ביטים על ידי ביצוע **checksum**. נסכום הכל ואז נוסיף את הסכום להתחלה/לסוף. השיטה היא בעצם לעשות xor משמאלי לימין. אנחנו לא נוכל לדעת אילו מבין הביטים התחלף (אפילו bit ה-parity עלול להיות שגוי). אם היה ביט שגוי איני אדע שההודעה שגואה, אבל אילו היה 2 שגויים איני יכול לקבל checksum נכון למרות שההודעה שגואה. לא נוכל לתקן, אבל התקורתה תהיה רק $\frac{1}{N}$.

הערה: ה-1 ב-1D הוא לא כי הוספנו רק ספרה אחת, אפשר להחליט להוסיף גם 5. בבין מה הסיבה שזה נקרא כך כשנגייע ל-2D.

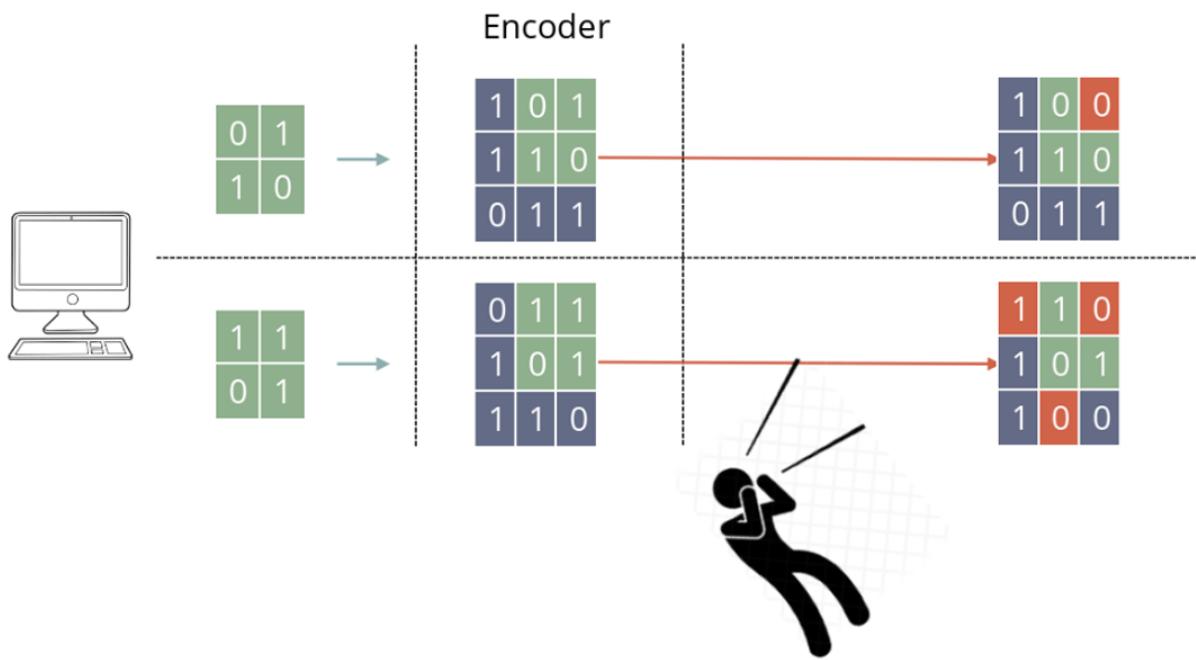
נרצה להיות מסוגלים לשחזר ולא רק לזיהות שגיאה:

2D-Parity

לא נסתכל על וקטור של הودעות, אלא על מטריצה. נניח שההודעה שלנו היא 9 ביטים, נחלק לריבוע של 3×3 (אפשר גם מלבן), נסכום כל אחד מהם ונוסיף bit parity לכל שורה ולכל עמודה, בתוספת בית בקורה לשורה ולעמודה:

$$\begin{array}{ccccccccc}
 d_{1,1} & \dots & d_{1,j} & & d_{1,1} & \dots & d_{1,j} & | & d_{1,j+1} \\
 d_{2,1} & \dots & d_{2,j} & \longrightarrow & d_{2,1} & \dots & d_{2,j} & | & d_{2,j+1} \\
 \dots & \dots & \dots & & \dots & \dots & \dots & | & \dots \\
 d_{i,1} & \dots & d_{i,j} & & d_{i,1} & \dots & d_{i,j} & | & d_{i,j+1} \\
 & & & & \hline
 & & & & d_{i+1,1} & \dots & d_{i+1,j} & | & d_{i+1,j+1}
 \end{array}$$

עבשו נוכל לזיהות שהייתה בעיה כמו מקודם:



בשזהים בעיה למשל בשורה הראשונה נוכל להצליב מול העמודה כדי להבין איפה הייתה הבעיה, ואז נוכל להחליף את הביט ולשחזר את הפעטה המקורית. בשמגדלים את ביות הביטים, עדין יתכן מצב שלא נצליח לשחזר.

ככל שנוסיף יותר ממדים נוכל לשחזר יותר מידע, הנוסחה לחישוב כמה ביטים אפשר לשחזר היא $\frac{\text{dimension}}{2}$.

מה תהיה התקורתה של 2D-Parity אם פקטה באורך 100 ביטים עם $i, j = ?$

$$\frac{i \cdot j + i + j + 1}{i \cdot j} - 1 = \frac{100 + 10 + 10 + 1}{100} - 1 = \frac{121}{100} - 1 = 0.21$$

סיכום

Overhead	Corrects	Detects	Code
2	1	2	Repetition (3 times)
$\frac{1}{N}$	0	1	1D Parity
$\frac{i + j + 1}{N}$	1	3	2D Parity

בעת נזכיר עוד שיטה, אך לא נרחב עליה הרבה:

Cyclic Redundancy Checks (CRC)

Ethernet משתמש ב-CRC-32. מוסיפים z ביטים, והסיבו להיבשל בזיהוי בעיה הוא $\approx 2^{-2}$. עברו z מאוד קטן יש סיכוי גבוהה יותר שפקטות שגויות יחודרו את המנגנון.

ניקח פקטה ונתיחס אליה באוסף מספרים. נתיחס אליהם במקדמים של החזקות בפולינום. יש למערכת פולינום גנרטור באורך המספר שבחרנו, ואת הפקטה שנרצה לשולח נחלק בגנרטור ונקבל שארית. השארית תהיה ערך CRC שאותו נשים בסוף הודעה. אופן שימוש:

- בשולח נחשב את CRC ונוסיף לפקטה
- מקבל ניקח את הפקטה שקיבלנו ונחלק בפולינום הגנרטור, נבדוק שארית. אם השארית 0, הכל תקין בהסתברות גבוהה. אם יש שארית נדע שהיא שגיאה.

דוגמה ל-CRC-3

נבחר גנרטור פשוט, $G = 1011$

הצד השולח:

. $D \cdot x^3 = 1101000$ המידע לשילחה הוא $1101 = D$. נוסיף לו 3 = r ביטים בסוף: 1101000 נחלק את 1101000 ב- $G = 1011$ בעזרת long division xor ונקבל שארית $R = 001$. ההודעה שנשלח היא השרשור: $D \| R = 1101\ 001$

הצד המקבל:

מחלק את ההודעה שהתקבל $R \| D$ בגנרטור G . לאחר חישוב השארית היא 000 ולכן לא זיהה טעות.

אם אחד הביטים היה משתנה בהסתברות 2^{-2} נצליח לזהות את הבעיה.
לא נתבקש לדעת לבצע חישוב כזה, אבל כן לדעת שהוא קיים.

הרצאה 6

תשכורת והקדמה

בשיעור הקודם המשכנו לדבר על שכבות הדאטה לינק והתחלנו לדבר על spanning tree protocol, עליון נמשיך לפרט היום. בשלב זה בהרצאה חזרנו על שקיים 38-24 במצגת 3.

מצגת 3
שקי 39-40

דרישות ה프וטוקול

לכל סוויז' צריך להיות פו ייחודי משלו (מסופק על ידי היצרן), לכל port בסוויז' צריך להיות פו גם כן, וזה גם קורה מהמפעל – שני אלו מספקים לנו יחד את הצירוף הייחודי של סוויז' + port החדש כדי לתעדף דברים באלגוריתם.

הפרוטוקול פשוט – כל סוויז' שולח בכל אינטראול את ה-`ip_root` שלו, מחיר לשורש, ואת ה-`ip` שלו. את ההודעה נשלח לכל השכנים. ברגע ששכן קיבל הודעה על סוויז' עם פו יותר נמוך הוא יפיז אותו בתור השורש שלו, ויעדכן את המחיר. ההודעה זו נקראת BPDU. דוגמה רצhaftה של הפרוטוקול בשקי 42.

שקי 41

לכל סוויז' תמיד `root` אחד, ולא תמיד יש לו `designated port`. כל ה-`ports` האחרים בהם לא `root` ולא מעבירים אף הודעה.

רשתות IP – IP Networks

מצגת 4
שקי 2-4

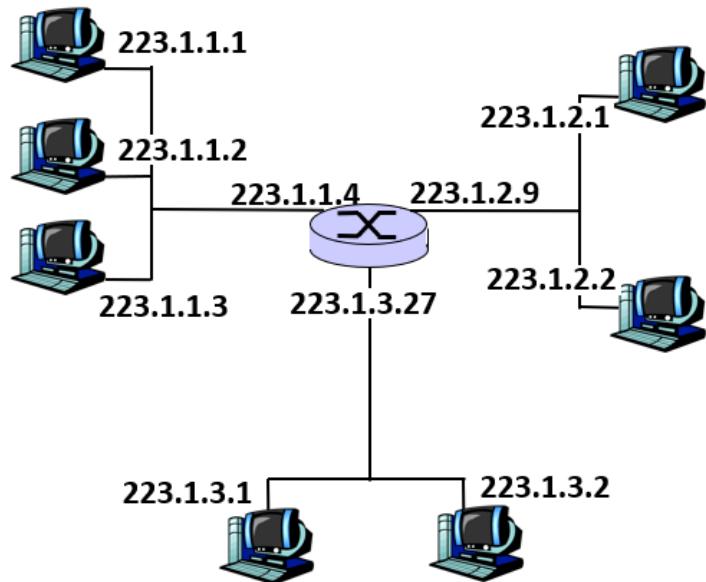
עבשוי אנחנו עולים שכבה לשכבות ה-Network. המטרה עבשוי היא לחבר LAN. מוטיבציה – למה צריך עוד שכבה? למה שלא יהיה סוויז' אחד גדול שיחבר את כל המחשבים בעולם? מעבר לסייעויות בנית העז היא מורכבת. עד עבשוי הסתמכו על כתובות MAC ללא סדר, היצרנית מטיבעה כתובות MAC על ברטייס הרשות, ובשהמחשב עובר בין רשתות הוא שומר על אותה כתובות. מה אם נעבד בראשת ענקית? נרצה סדר בכתובות, ולא שבל סוויז' יצטרכ לזכור טבלה עצקית של כתובות (ואם לא יזכור לאן – יעשה broadcast לכל מקום, מאוד לא יעיל).

שקי 5

נדרשת כתובות לוגית היררכית, זה מה שקורא לשכבות ה-Network. את השיעורים הקרובים נקידש לשכבה זו. נתמקד בפרוטוקולי ניתוב, בדבר על פרוטוקול IP ועל פרוטוקולים מסוימים כמו ICMP אשר משמש לדיבוג הרשת. לאורך ההסביר ניגע גם בשכבות הטרנספורט בגללה הבלתי נפרד משכבות ה-Network.

שקי 6

רוב הדוגמאות שניתן יהיו IPv4 שבו כתובות ה-4 בתים (4 מספרים שכל אחד מהם בטוחה 0-255), כאשר הכתיב המקביל הוא בצורה 223.1.1.2 למשל.



כאן יש לנו רשת עם מחשבים שמחוברים לראוטר שתפקידו לנ Abbott פקודות (סוויצ' שימש למיתוג פקודות בשכבה הדאטה לינק, רואוטר משמש לנ Abbott פקודות בשכבה ה-Network). לכל רואוטר יש כמה גלילים וכל אחד מהן בתובות IP משלו שנמצאת ב-LAN אחר. סוויצ' היה שקוּף בשכבה הדאטה לינק, לא הייתה לו בתובת MAC. הרואוטר לעומת זאת אינו שקוּף, וכך כל אחת מהרגליים שלו בתובות IP. כאשר אנחנו שלוחים הודעה בין מחשבים אשר עוברת דרך רואוטר, בעת ההגעה לרואוטר ה-source address משתנה מהמחשב (למשל 223.1.2.1 באירוע) להיות כתובת של הרgel בראוטר ממנו יצאתה הודעה (למשל 223.1.4.2 באירוע).

סיכום ההבדלים:

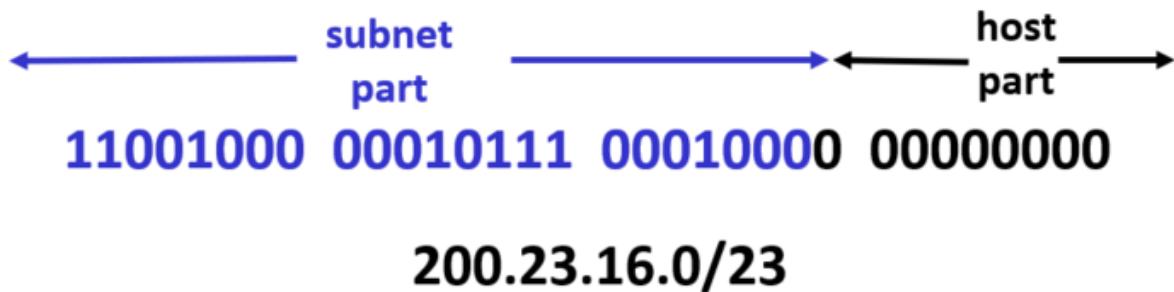
Switch	Router	
מיתוג פקודות בשכבה Data Link	מיתוג פקודות בשכבה ה-Network	שימוש
בתובות MAC	בתובות IP	דרך פעולה
שקוּף מבנית הרשת	לא שקוּף, לכל רgel בתובות IP	שיקיפות

שקלף 7

– תת רשת, כל חלק של הרשת עד הראטור נקרא net-sub. כל הכתובות באותו net-sub חולקות prefix משותף שמשיע לראים לבצע אגרגציה וכיוטוב.

את כתובת ה-IP אפשר לחלק לחלק של subnet (ביטים עליונים) ולחוק של host part (ביטים תחתונים).

שקלף 8



CIDR

מדובר בכתוב נפוץ לתאר כתובות רשת. הכתוב אומר כמה ביטים בכתובת מזינים את subnet. בדוגמה לעיל יש 4 מקטעים של 8 ביטים, וה-**23**/23 **הביטים העליונים** מזינים את הרשת, וה-9 התחתונים את hosts בתוך הרשת, לומר ברשת זהו יש 2^9 כתובות host. ככל שהיא שמאחוריה-/ גדול יותר ברשת קטנה יותר.

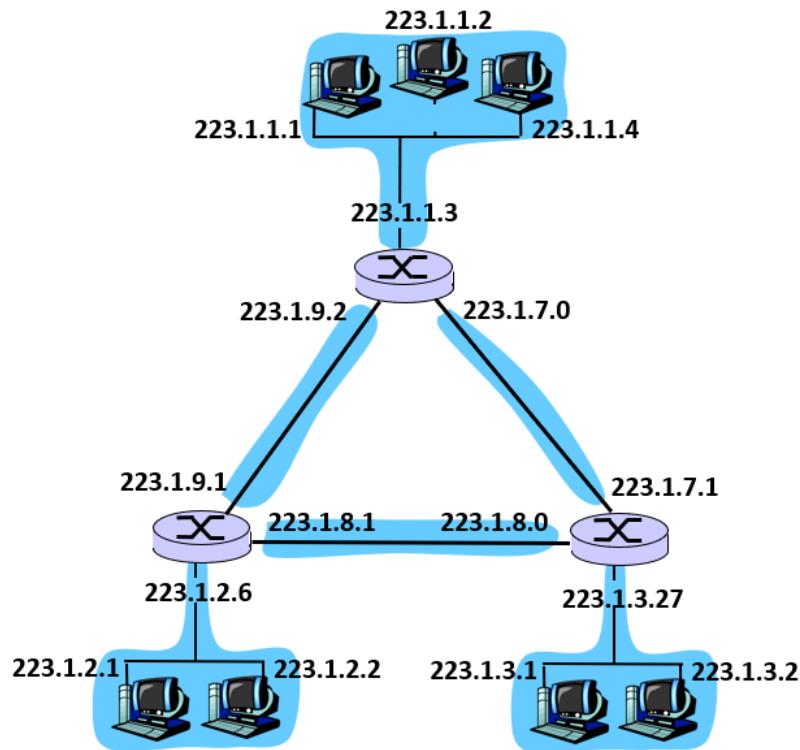
שקלף 9

Subnet Mask

אנחנו רוצים להבין אילו subnets יש ברשת. איך נדע כמה subnets יש באյור? מוצאים את הראטור מהטופולוגיה, ומסתכלים על ה"אימס" שנוצרם. כל אי כזה הוא subnet.

בדוגמה הספציפית זו subnet mask של תת-הרשת התחתונה הוא **24**/24 לומר byte התחתון בתת-הרשת מציין את host וכל השלושים האחרים את הרשת. למשל בתת-הרשת התחתונה כל hosts יתחילו מ-223.1.3 ואז הביט האחרון (המספר האחרון) שלהם ישנה. גם הרגל המתאימה של הראטור מקבל כתובות מתאימה (223.1.3 ואז מספר אחריו).

שאוף 10
תרגיל – כמה subnets בדיאגרמה?

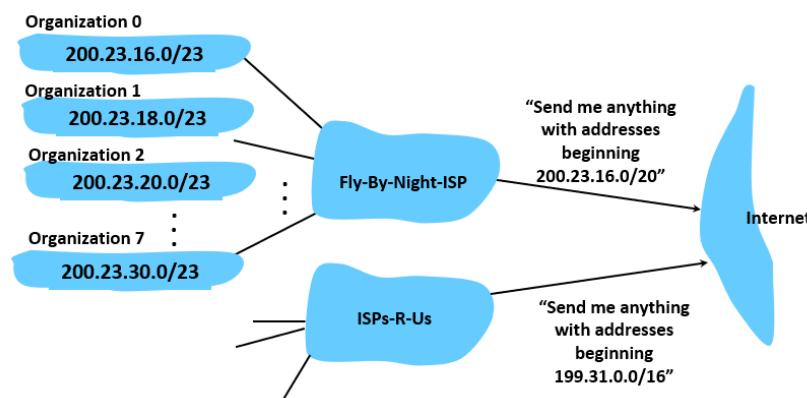


ונמצא את כל הראותרים ואז נשארים עם 3 subnets. אבל בלינק בין הראותרים הוא subnet בפני עצמו, כי הם לא שייכים לאף רשת אחרת. **שה"ב 6 subnets**.

שאוף 11

Route Aggregation

התחלנו את השיעור בדבר על למה האינטרנט הוא לא LAN אחד גדול, והסיבה לכך היא בעיית אגgregציה – לא נרצה בטבלה של הסוויץ' רשותה לכל כתובת האינטרנט בעולם. זה בן מתאפשר באמצעות כתובות IP. בעוד שכנתובת MAC מוטבעת במכשיר (גlobeלי), אבל שובר היררכיה), כתובות IP מתקבלות באופן דינמי מספקית האינטרנט.



כآن הספקית העליונה קיבלה הקצאה, והרשות שלה היא **20/200.23.16.0**, כלומר היא יכולה $2^4 = 16$ רשותות קטנות יותר (זו רשות גדולה, צ�ור לנו, אבל SHA-/ עם מספר קטן יותר בבה הרשות גדולה יותר). הספקית למטה קיבלה **16/**, יש לה רשות גדולה אפילו יותר.

הרשות העליונה ISP Fly-By-Night מחברת ארגונים, לכל אחד הייא הקצתה רשות, וכל רשות הייא **23/**. כמה יכולו הייא יכולה לספק? 20 ביתים מוקצים לרשות הפנימית, כלומר יש לה סך הכל 12 ביתים להוסטיהם, אז $2^9 = 512$ IP addresses. כל רשות של **23/** עם 9 ביתים להוסטיהם, כלומר $512 \div 4096 = 8$.

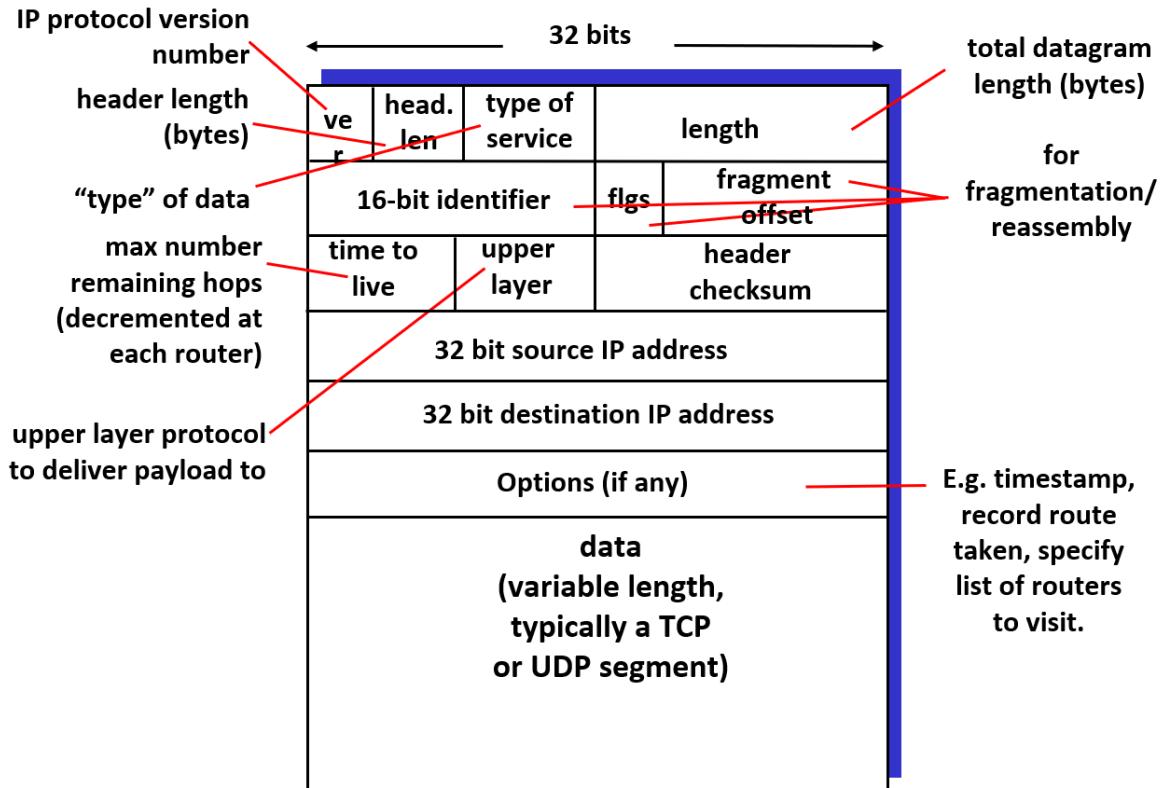
הספקית יכולה להודיע לספקית שלה לשלוח לה הודעות ש商量ונות לבתוות שמתחליה ב-**20/200.23.16.0**, כי זו הרשות שלה. כל רואו הבא בתור לא צריך לזכור את הרשותות הפנימיות (0-7 organizations), אלא רשומה אחת בטלת הנוטוב שלו שמכילה רק את ה-prefix הנדרש וכן יחליט לאן הודעה נשלחת להאה).

נניח שארגון אחד שקנה את כתובות שלו דרך ISP Fly-By-Night רצה לעבור לספקית השנייה, שלה יש בлок **16/200.23.18.0**. עבשו הספקית ISPs-R מפנה דרכה ניטובי שמתחלים או ב-**16/199.31.0** או **23/200.23.18.0**. החוק הוא להעביר לפי הכתובת הספרטטיבית ביותר שאפשר למצוא, וכן הודעה כזו לא תגיע לספקית העליונה אלא לתחתונה. מבחינת הספקית העליונה שום דבר לא השתנה, היא עדין מפרסמת את אותו הניטוב. כן נפגעה מה האgregציה.

שוף 13

[air מקבלים בлок כתובות באינטרנט?](#)

יש ארגון שנקרא ICANN שאחראי על חלוקת כתובות IP לארגונים, חלוקת resources באינטרנט. מהרגע ש-ISP קיבל כתובות IP זה הפך להיות הרכוש שלו, הוא יכול להשתמש למכור חוותיות מהנתה וכו'.



כל האזיה זו היא הדטה שנמצאת בתוך data link frame. את ה-source וה-destination ניתן לא רואים, הם נמצאים מעל האזיה שאנו רואים מולנו. מצד שני, גם ל-IP יש דטה, גם ל-IP ימוקם איפה שהשכבה הבאה מתחילה (כלומר ה-header של שכבת transport נמצא בתוך ה-data של IP), וה-data של שכבת יהיה בשכבת האפליקציה). בקיצור – השכבה התחתונה עוטפת את המידע שהגיע אליה מהשכבה שמעליה.

הרכיבים שאנו רואים בדיאגרמה:

יש מספר גרסה של הפרטוקול (ver), אורך של ההודעה (length), שדות שימושיים לפרגמנטייה (אם הפקטה ארוכה – IP תומך בפיזול וחיבור של הודעות). השדות הći חשובים הם שדה **source IP address** ו-**destination IP address**. מדובר בשדה של 32-bit ב-IPv4, והוא מזוהים host שלו ו-host מקבל, ואינם משתנים במהלך חייו הפקטה. אם שלחתי הודעה וכותבתי בה את ה-IP source שלו וגם source MAC address משתנה לבתוות ה-MAC של רgel הראור שעהירה את השלי, בכל פעם שנעבור router source MAC נשתאר זהה (בכל שלב יאתרו מה ה-longest prefix match של לבתוות היעד). ההודעה הלאה, אבל IP **source** נשאר זהה (בכל שלב יאתרו מה ה-longest prefix match של לבתוות היעד).

הדגשה: ה-**MAC address** רק משמש אותנו לצעד הקרוב הבא.

שאוף 17

Demultiplexing at the IP Layer

אחד היתרונות ש-IP מאפשר לעשות זה לכתוב מה פרוטוקול של השכבה הבאה (מעליו), האם זה פרוטוקול TCP או UDP של שכבת ה-Transport, או פרוטוקול ICMP שמשמש לאבחן בעיות ברשת. בשעה שלנו מגיעה למכוна ביעד, מكونת הקצה מסתכלת על ה-IP Header ומビינה מי הפרוטוקול של השכבה הבאה שאמור לטפל בהודעה (וכך המימוש המתאים יטפל בהודעה – TCP או UDP לפי מה שצוו).

ב Hodעות של שכבת הילינק אין הרבה שימוש לצוין הפרוטוקול של השכבה הבאה, כי מעל שכבת הדאטה לינק יש את שכבת ה-Network, שבה הפרוטוקול הוא תמיד IP, אין עוד אופציות.

(הצצה קידמה: גם בשכבת ה-Transport ניתן במנגנון דומים – ה-port יזהה את האפליקציה)

שאוף 18

נרצה לפטור כמה בעיות שאין להן כרגע מענה:

1. איך מקבלים בתובת IP? **DHCP**
2. איך אני מושג את בתובת IP של היעד שלו? **DNS**
3. ביצוע המרה מכתובת IP לכתובת MAC וחזרה. **ARP**

DHCP

אחד מהפרוטוקולים הבסיסיים שמשמשים hosts, ומטרתו להקצות ל-host בתובת IP. כדי לשלוח הודעה צריכה לכתוב את IP source שלו, מאיפה קיבלתי אותו בכלל? כאן DHCP נכנס לתמונה.

שאוף 19

בשונה מכתובת MAC, אני צריך לקבל בתובת IP בתוך הרשות, אחרת אשבור את המבנה ההיררכי. יש שני דברים לעשות את זה, אחת מהן היא באמצעות כתובת IP קבועה שה-administrator קינפג לתוך המערכת. כל מערכת הפעלה נותנת גישה קבועה בתובת IP של host שאפשר לקינפג עם הרשותות מתאימות. אם אותו מחשב יירצה אחרת וינסה לשולח הודעה עם בתובת שלא נמצאת ברשות הוא יהיה בבעיה, יותר חמור אם הכתובת הזאת כבר שייכת למי שהוא אחר ברשות.

הדרך השנייה לעשות את זה היא הקצתה הכתובת בצורה דינמית בעת התחברות לרשות באמצעות פרוטוקול (**DHCP**) **Dynamic Host Configuration Protocol**. לרוב ברשות יש שירות DHCP שמטרתו להקצות כתובות ברשות למחשבים חדשים שמתחברים.

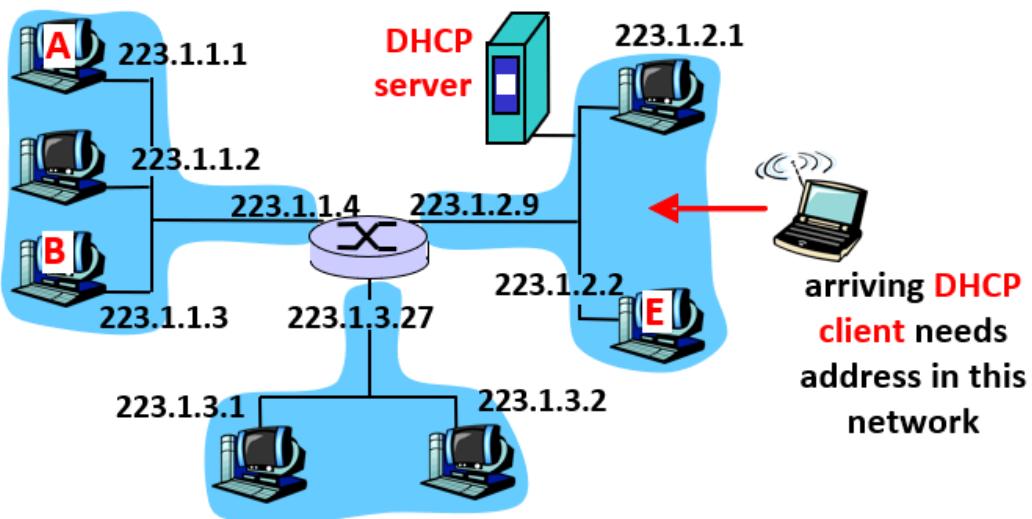
שאוף 20

לפרוטוקול כמה שלבים שנציג בהמשך. הפרוטוקול הוא מעין שיחה בין host לשרת ה-DHCP, אבל במקרה הזמן הזה host עדין אין כתובת IP והוא גם לא יודע מה כתובת IP של השירות DHCP لكن יש בעית תקשורת שצורך לפתור. במנוחה DHCP, השירות יתן lease לכתובת IP (סוג של "שכירות") למחשב החדש שהתחבר. נקרא lease כי הוא תחום בזמן (כמו חוזה שכירות), וכשפרק הזמן פוקע או שהמחשב מבקש

להאריך את lease שלו ולשמור את אותה כתובת IP, או שכותבת IP, חזרה לבנק הכתובות שהשתרת מתחזק. המטרה של הפיקעה היא שאם מחשב כביה נרצה לשחרר את מה שהקצינו לו בצורה אוטומטית.

שקייף 22

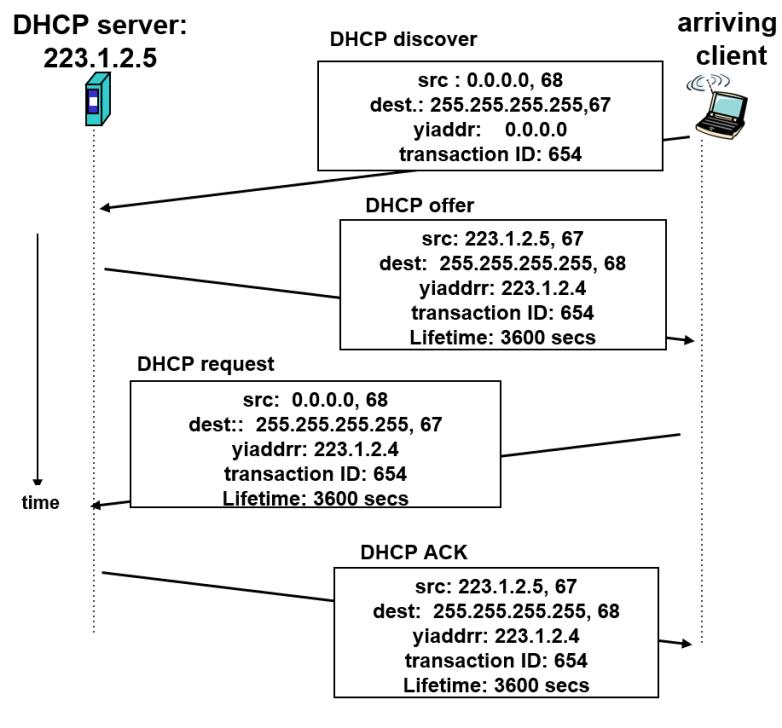
תרשים שמתאר את הרשות:



בשלקו מגע לרשות הוא מנסה להתחבר, ואז שרת ה-DHCP צריך להקצתו לו בכתובת. נראה בקרוב שהוא עושה קצת יותר מזה – נותן לו מידע שימושי על איך להשתמש ברשות (מה ה-subnet mask, מי ה-router).

שקייף 23

תרכיז שימוש בפרוטוקול DHCP



1. ל Koh מגיע והוא צריך לתקשר עם שירות DHCP שלוח ב-ץ על הראטור בחלק מהשירות הבסיסי-שהראטור מספק. אותו Koh שSEG אין לו בתובת IP והוא גם לא יודע את בתובת ה-IP של שירות ה-DHCP. הוא שולח הודעה שנקראת **DHCP Discover** source IP מה-IP שלה הוא כולל אפסים בבייטים destination IP הוא הכל 1 בבייטים (255.255.255.255), וה-IP (0.0.0.0) המשמש בתובת שכלה 1-im משמעו שהיא broadcast-subnet. בחלק מההודעה ה-IPdiscover נקבע גם ID transaction כדי שיכל לדעת לשני את התגובה של השירות אליו.
2. בתגובה הל Koh מקבל **DHCP Offer**, שמשודר ב广播 broadcastDHCP משרת ה-DHCP ומתייג על ידי אותו transaction ID, בתוספת lifetime שמצוין לבמה זמן הכתובות ניתנה לקליינט אם הוא ייקח אותה.
3. לאחר מכן הל Koh מבקש את הכתובות ב-**DHCP Request**. למה יש צורך בכך אם הרגעים הציעו את הכתובות? כי יכול להיות שברשת יש כמה שירותי DHCP שענו ל Koh במקביל, ולכן הוא צריך אקטיבית לבקש בתובת מסוימת.
4. לבסוף שירות ה-DHCP שלוח חזרה **DHCP ACK** שמאשר שהכתובות שלו, ומאותו רגע ל Koh יש בתובת IP בראשת.

בתרמונה – 67 ו-68 זה הפורטים של שירות DHCP, נגיעה לפורטים בהמשך בשנדייר על שכבה ה-Transport (לשני הודעה לאפליקציה שנקראת DHCP).

תרגול 5

Spanning Tree

Interconnecting Nodes

עד עכשווי, אמרנו שכל המחשבים מדברים על אותו תוון, ואנחנו מאפשרים את התקשרות בעזרת פרוטוקולים כמו ALOHA וمبرוצעים הנחוצים כמו איך הגיעו ההודעות מתפלגת. אם שני מחשבים מדברים באותו הזמן על אותו התוון, ההודעות לא עוברות הלאה. אנחנו עדין נמצאים ברמה 2, אבל נרצה לשפר את המודל שהציגנו (שכנן במציאות מחשבים לא מתקשרים על תוון בודד).

Broadcast domain – סט כל הצמתים שמקבלים את ה-frames (של שבבה 2) אלו של אלו.

נאמר שרשת מכילה אחד או יותר חסום broadcast domain. אם מחשב אחד שולח הודעה, כל המחשבים שומעים את אותה הודעה (במקרה של הודעה מוצלחת).

Collision Domain – עכשווי הוא ייה שונה מה-chain broadcast domain. יהיה אוצר שבו מחשבים יכולים לשימוש זה את זה, אבל גם אם ידברו באותו זמן זה לאו דווקא יגרום להתנגשות.

מבנה הרשת

בדרכם כלל לרשת יש טופולוגיה מאוד מורכבת.

הרשת מורכבת מצויד רשת, ונוהג לחלק אותו לפי מודל השכבות ISO שראינו בקורס:

- **שכבה פיזית** (שכבה 1 – ביטים): Hub (כניסה אחת וכמה יציאות) או Repeater (מעביר את ההודעה שהוא קיבל מבוחר)
- **שכבת-link** (שכבה 2 – MAC): Switch (להעביר הודעה בין מחשבים שונים, מכיר בתובעת MAC)
- **שכבת הרשת** (שכבה 3 – IP): Router (נפגש בשיעור הבא)

לכל המכשורים האלה בדרך כלל יש כמה יציאות ומקבלים רשת מסועפת.

– מכשור בסיסי, כמו מפצל. מקבל אינפוט אחד ומעתיק אותו לכמה outputs. יכול להיות שיבצע **Hub correction**, אבל אולי גם לא. כל מה שמחובר ל-Hub הוא chain Collision domain כי אם אחד משדר איזה שני בהברח שומע גם כן. אם שניהם ישדרו באותו זמן – הם יפריעו אחד לשני.



– יש רכיב זיכרון שזכור מידע ומבצע את ניתוב הפקודות בצורה חכמה יותר למקומות הנכונים. אם בינה הودעות מගיעות במקביל הוא יודע לטפל בהן ללא התנגשות.



אם הוא מקבל 2 פקודות באותו הזמן, לסויץ' יש באפר שמאפשר לו לשדר את השניה לאחר מכן. אם מחשב אחד שלוח פקודה למחשב אחר אך בהתחלה משדרים הכולם, אבל יש שלב שבו לומדים את המיקומים אחד של השני, ואז ההודעה מנוטבת וישורת לעדיה (לא ב广播 broadcast לכלום).

כדי ללמידה, צריך לשולח נתונים שימושיים לסויץ' להבין איפה המחשבים נמצאים. בשצומת שלוח frame (בשבה 2) הוא מציין מה ה-MAC Address של היעד. באשר switch מקבל פקודה (פריים) לאחד מה-ports (שלו הוא ישמר את הצמד SRC MAC, IN PORT) בטבלה.

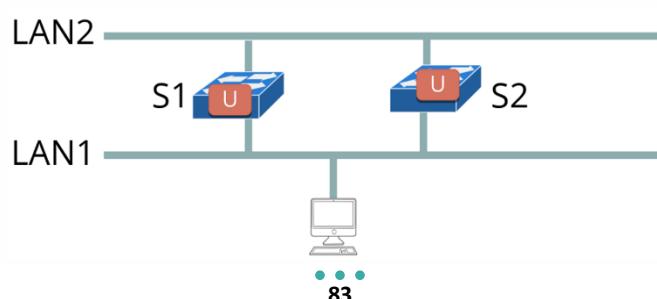
אם קיבלנו entry מהצורה (k, DST MAC, PORT k) אז נשדר את הפקודה דרך port k למעט במקורה שבו k הוא גם ה-port ממנו הגיעו ההודעה. למה? כי זה אומר שמחשב המקור והיעד שניהם נמצאים באותו domain, ולכן ההודעה בבר偈 הגיעה ללא התוצאות ה-switch. אחרת, שלוח את הפקודה לכל ה-ports חסוך מל-k.

עם הזמן, נמלא את הטבלה – אבל היא אינה סטטית.Entries בטבלה יש timeout ואם בתובת MAC לא הייתה בשימוש במשך x שניות, השורה תימחק מהטבלה. הסיבות לכך: אנחנו לא רוצים לנפגג את הסויץ' בצורה ידנית, מעדיפים שהכל יבוצע אוטומטית. סיבה נוספת – יכול להיות שדברים התקלקלו או עדיף לנתק איזור שלא יכולות להגיע אליו הודעות.

תרגול בשקפים 22-11 מופיעה דוגמה. הרעיון הוא לנצל את מידת הכתובות בכל הדמנות – כאשר מגיע פריים ממוקור מסוים, כל סויץ' שהפריים עבר דרכו זכר גם את בתובת המקוון, למראות שהבעיה הנווכחית היא בביול מיצאת בתובת היעד. כל סויץ' זכר מאיזה port שלו הוא קיבל הודעה למשל מחשב A, ושם עבר את ההודעה שסמוענת ל-A תוך כדי שהוא זכר עבשו מי הוא זה שהшиб ל-A ומכוון לטבלה את המיקום שלו גם כן.

Forwarding Loops

כל הרעיון שתיארנו לעיל עובד כל עוד אין ללולאות:



במקרה זהה הסוויצ'ים יתבלבלו ולא ידעו לאיזה port לכוון הودעה שסמנועת למחשב למיטה, כי הם קיבלו ממנה הודעה בשני ה-*dst*-ports. יכול להיות שהוא ידרס את ה-*entry* בטבלה או שהוא ישדר כל פעם לכל הרಗלים (מעלה התנגשויות וכו').

האם נרצה להיפטר מlolaoת לגמרי? ממש לא, ליתירות חשיבות:

- יכול להיות שבעתיד תהיה בעיה, ונרצה למצוא דרך חלופית (הlolaoת תהפוך למסלול ללא מעגל כי יהיה ניתוק, והתקשרות תוכל להמשיך להתקיים).
- הסתגלות והתפתחות – עם הזמן נתבים מסוימים יתגלו בטוביים יותר אחרים יהיו פחות בשימוש, לאחר מכן דברים עשויים לששתנות. מטאפר בין היתר בגל הטבלה בסוויצ'ים שמוחקת רשומות אחת לבמה זמן. ברן עד לקבלת יציבות.

איך ניתן לדודע עם lolaoות?

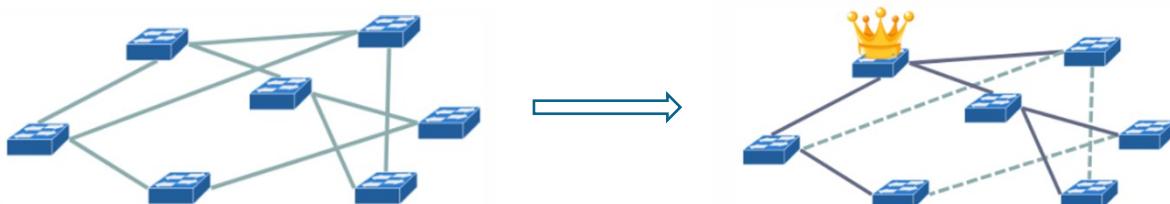
- הוספה נוספת לפקודות שירד עם כל ביקור ב-*switch*.
- **לבנות מסלול אחד שייהי בשימוש.**

Dealing with Forwarding Loops

נרצה לבנות מבנה טופולוגי בראשת שבו אין מעגלים, תוך הכרה בעובדה שתתכן יותר מדרך אחת בין מקור לעוד. אם בהמשך תהיה תקלה, המסלולים ישתנו בהתאם – ולכן חשוב לא לנתק שום דבר.

Spanning Tree Protocol

את הפרוטוקול הזה יריץ כל סוויצ'. המטרה:



זה קצת שונה מפרוטוקולים שראינו עד עכשיו – עד עכשיו כל מחשב עשה בדיק את אותו דבר כמו מחשבים אחרים. כאן בולם עדין ישמשו באותו פרוטוקול אבל יש שינוי קל: לכל סוויצ' יהיה *po* ייחודי (בערך) משלו, והמחשב עם ה-*po* הנמוך ביותר הוא ה-*root*.

בשאנחנו מייצרים את העץ אנחנו מתעלמים מהמחברים שמחוברים ומתיחסים רק ל-LAN שמחוברים לסוויצ'ים. המטרה היא שככל ה-LAN יהיו מחוברים אחד לשני בעזרת הסוויצ'ים. יש פרוטוקול אינטראקטיבי שלוקח לו כמה איטרציות להתייצב, אבל לאחר מכן יהיה שורש אחד שכולם מכירים, וכל LAN תהיה דרך אחת בלבד להגעה ל-*root*. כל סוויצ' שלא מחובר ל-LAN יכבה את עצמו (לא לחילוטין, אם יש תקלה כלשהו הם יתעורו אולי מחדש).

אופן ביצוע ה프וטוקול

אנחנו נניח שה-po הוא ייחודי. כל סוויז' ישלח פרימים שנקרא BPDU (מכונה גם "Hello") שכולל:

- ה-po המכני נמור בכל הרשת. מה אני חושב שהשורש, בהתחלה אחשב שהוא אני באופן דיפולטיבי.
- מרחק מהשורש. עלות המסלול מהסוויז' השולח אל השורש.
- ה-po של השולח.

כל סוויז' מבצע את אותו פרוטוקול במקביל לשארו: (DTR – Distance to Root , RID – RootID – RootID – RootID)

1. שולחים פרימים HELLO דרך כל ה-ports

2. אם קיבלתי פרימס HELLO:

עדכן במידת הצורך את RID ואת DTR

3. אם ציפיתי לקבל פרימס HELLO והרבה זמן לא קיבלתי (אנlich שהיה ניתוק ולבן):

אחשב RID מחדש

מי שיבחר להיות השורש הוא זה עם RID הנמור ביוותה. לפי פרימס HELLO אדע לאיזה port לטעוף. אם קיבלתי שתי פקודות מפורטים שונים: אחד אומר ש- $1=RID$ והשני אומר $3=RID$, אני אתעדף את המסלול שМОוביל אליו ל-1. מה קורה במקרה שוויז' אסתכל על האחד שהוא-DTR קצר יותר. אם גם הוא אותו דבר, אשבור שוויז' בדרך אחרת (למשל אתעדף את ה-number port הנמור יותר). כאן כבר אין עניין של ייעילות, רק קבלת החלטה.

סוגי Ports בתוצאה מהפרוטוקול

(PP) Root Port / Parent Port – בכל סוויז' נבחר (RP) Root Port כלומר ה-port שמוביל לשורש העז. ה-RP חשוב לבניית ה-tree, spanning, לא לצורך שידור הודעות (לשם כך יש את הפרוטוקול שראינו בתחילת התרגול היום). יהיה רק RP אחד לכל סוויז'.

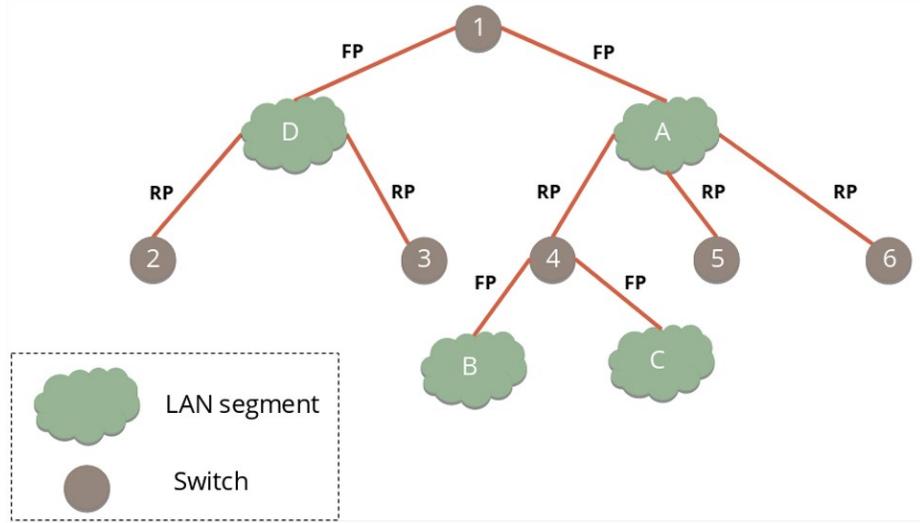
"זה ה-port מנקודות המבט של ה-switch: "מאייה port שלי אני אגיע הבן מהר ל-root?"

Forwarding Port (FP) / Designated Port – ה-port שבו ה-LAN משתמש כדי להעביר הודעות לסוויז' (לכל LAN יהיה אחד בזה, אבל בסוויז' אחד יכולם להיות כמה FPs ל-LANs שונים).

"זה ה-port מנקודות המבט של ה-LAN: "מ-port של לי switch אני אגיע הבן מהר ל-root?"

Blocked Port (BP) – אם יש port שאינו RP ואינו FP, הוא נחסם והוא פרט להיות BP. ה-port לא נבסה לגמרי, ועודין הסוויז' ייאזן להודעות HELLO שמתקבלות דרך הפורט הזה.

RPs יהיו חלק מהעץ הפורט. אם שלוחתי מסוויז' אחד HELLO למסוויז' אחר, הוא לא יעביר את אותה הודעה בדיק, אלא יכין הודעה חדשה בהתאם למידע שקיביל ב-HELLO (כלומר HELLO הוא בין שני LANs בלבד).



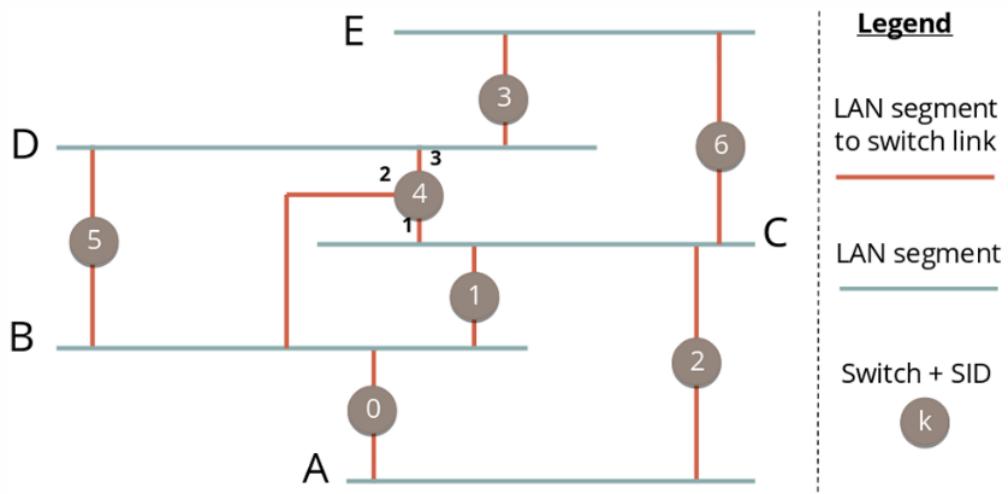
יכול להיות שיחסים מסוימים עם Bandwidths שונים. במקרה זה נתעדף BW יותר גדול ולשם כך משתמש באלגוריתם ממושקל (קיים אבל לא מתעסק בזיהה בקורס).

בשופטורים בעיות עם Spanning Tree Protocol, למרות שבפועל זה מתבצע במקביל בכל הסוויצ'רים לנו נוח דיווקא לעבוד בצורה איטרטיבית ולהתחל מסוייך. 0.

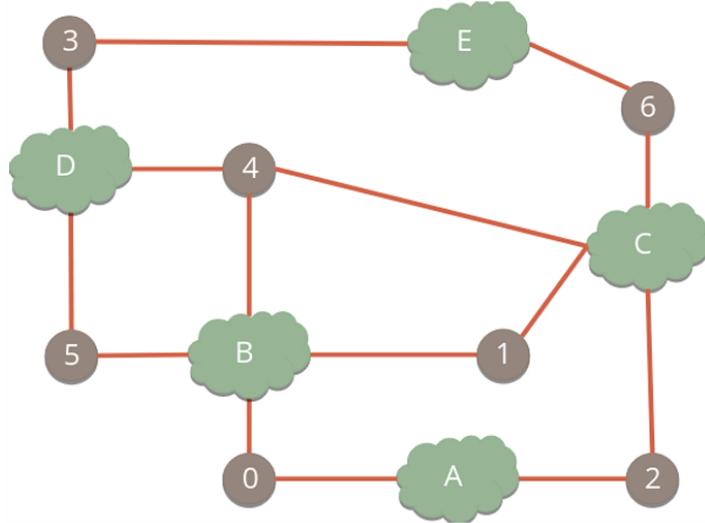
שאלות בנושא Spanning Tree Protocol

[פתרונות מבט על הרשת בולחן](#)

בתרגול עבדנו עם הדוגמה זו:



שיכולה להיות מומרת גם לתרשים הבא:



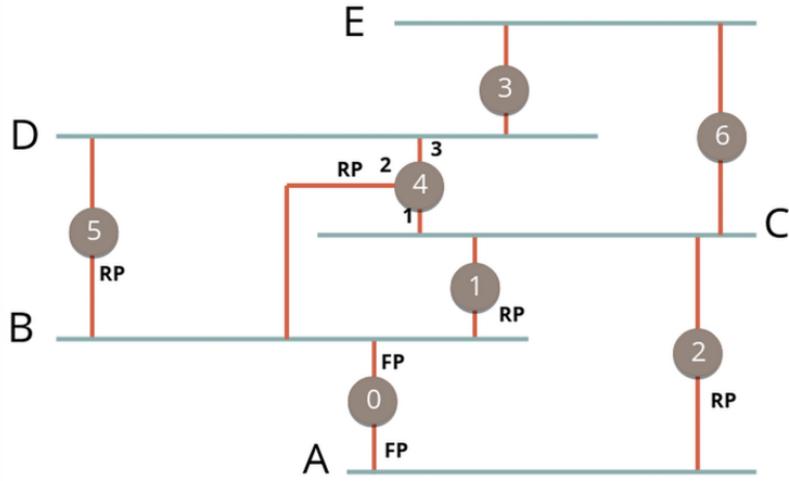
נוח לעבוד עם טבלה שבאייטרציה הראשונה נראה כך:

SID	RID	DTR
0	0	0
1	1	0
2	2	0
3	3	0
4	4	0
5	5	0
6	6	0

נתחיל עם 0 ששולח HELLO frame, ואז סוויצ' 2 וסוויצ'ים 1,4,5 שומעים את ההודעה ומעדכנים:

SID	RID	DTR
0	0	0
1	0	1
2	0	1
3	3	0
4	0	1
5	0	1
6	6	0

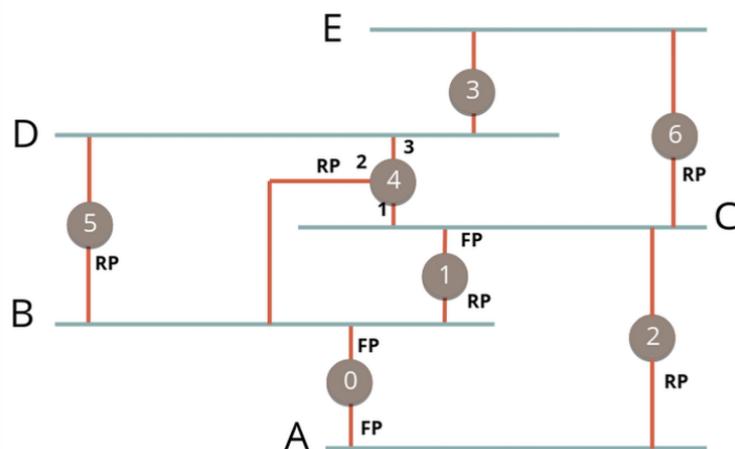
בשלב זה אפשר לעדכן מנוקודת המבט של כל LAN עליו לבחור איזה switch קרוב יותר כדי להגיע אל ה-root, ובבחירה את ה-FP שלו אל הסוויצ' הקרוב יותר.



עכשו נסתכל על C: הסוויצ'ים 2,4,6,0 שלחו לשם הודעות. בשלב זה סוויצ' 6 חושב שהוא root, כי הוא לא שמע שום דבר אחר. בש-6 קיבל את ההודעה והוא ידע שה-root הוא 0, ונעדקן:

SID	RID	DTR
0	0	0
1	0	1
2	0	1
3	3	0
4	0	1
5	0	1
6	0	2

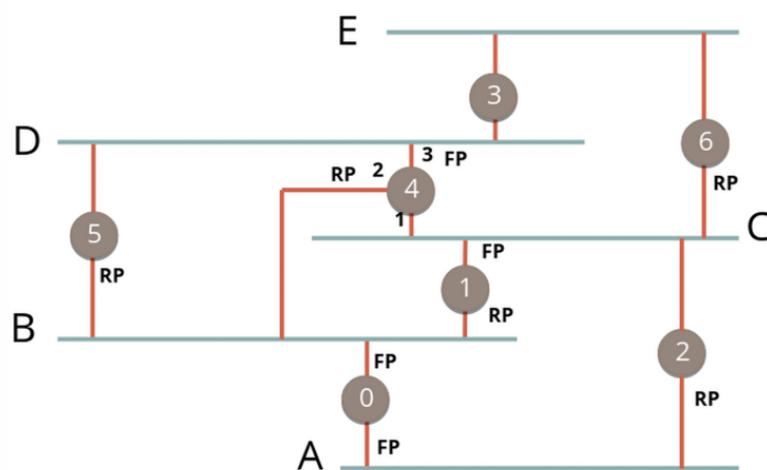
air 6 יבחר את ה-RP שלו? דרך 1,2,4 הדרך ל-root היא תmid 1+1, אז נשבר שוויון לפי ID Switch הנמור ביטור ונקח ש-1 יהיה ה-RP.
air C LAN יבחר את ה-FP שלו? הוא יכול דרך 1 או דרך 2, ולכן יבחר את 1.



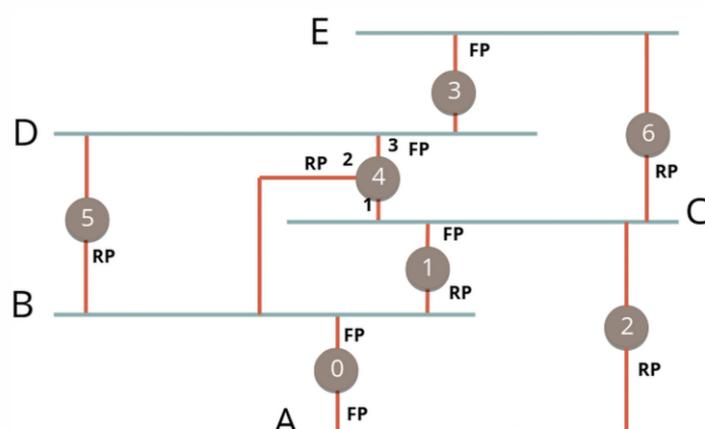
עכשו נבדוק את D. D מקבל הודעות מסוייצים 3,4,5. 4 ו-5 כבר מכירים את root אבל 3 לא. עכשו יכיר את root הנוכחי:

SID	RID	DTR
0	0	0
1	0	1
2	0	1
3	0	2
4	0	1
5	0	1
6	0	2

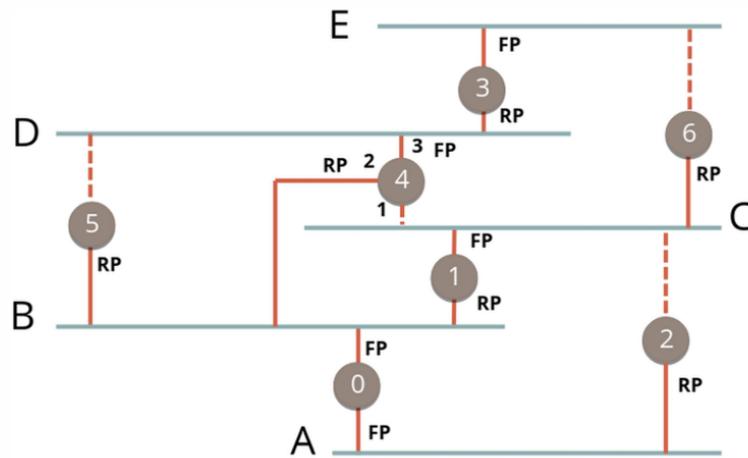
איך D יבחר את FP שלו? ההתלבבות היא בין סוייצ' 4 ל-5 כי שניהם במרקם 1 מהשורש (לעומת 3 שהוא במרקם 2 ולכן לא נשלול לגבי). נבחרו שווין ונבחר את 4 (המספר יותר).



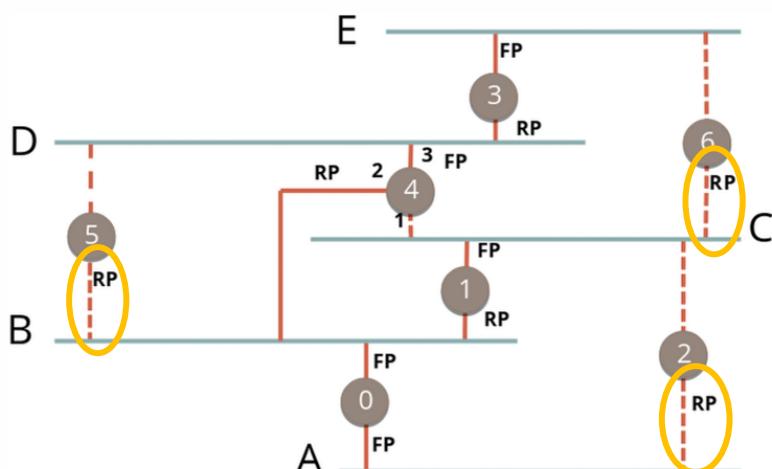
עכשו נותר רק E LAN. E קיבל הודעות מ-3 ומ-6, שניהם במרקם 2 מהשורש ולכן יבחר את port של סוייצ' 3 להיות FP שלו.



סיימנו לסמן את כל ה-FP וה-RP. עכשיו אפשר להתחיל לבנות את העץ. קודם כל נהפוך כל port שלא בשימוש (בולםר הוא לא FP ולא RP) ל-BP.

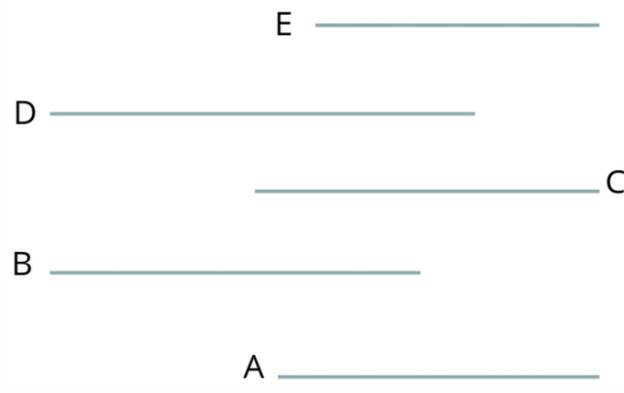


ובשוו, לא משנה לנו שה-switches יהיו מחוברים לרשת, מה שמשנה לנו זה LANs יהיו מחוברים לרשת. כל port שלא לחבר בין 2 LANs יתייחס, ונוכל לבטל גם אותו (מודגשים באיוור):

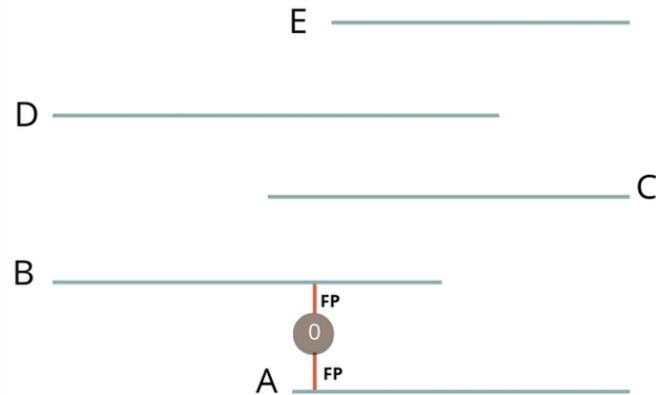


פרתון שמתחליל מנוקודת מבט על ה-LANs

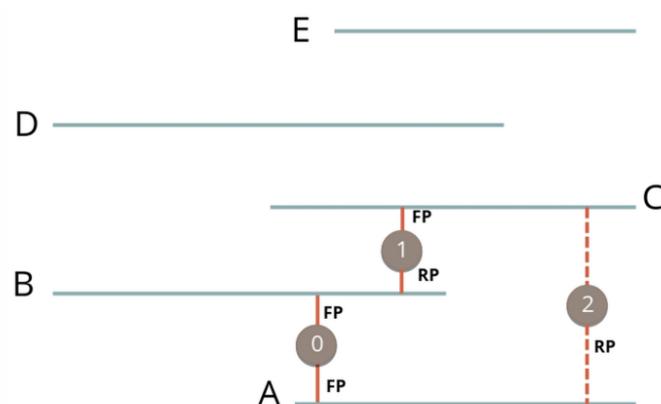
השיטה זו פחות משקפת את מה שתרחשת במציאות, אבל אפשר להחליט איזו שיטה יותר נוחה. מה שחשוב לנו זה לחבר את כל ה-LANs בצורה של עץ.



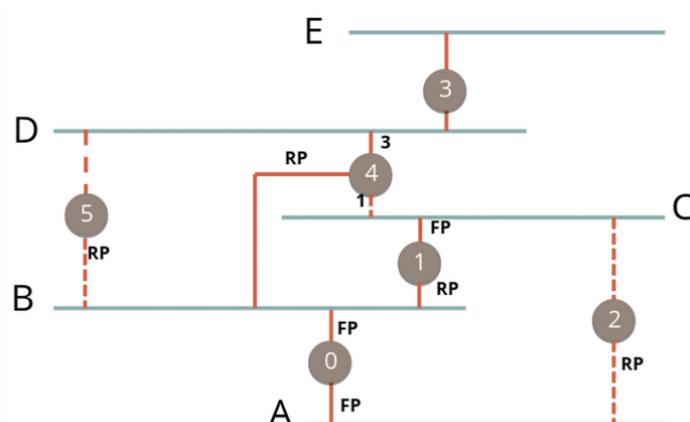
נתחיל מלחבר את השורש:



ל-0 נוכל לשתמש ב-1,2,4. נשבור שווין ונבחר ב-1. C-A בבר מחוברים אז 2 מיותר
עלינו אנחנו צריכים לחבר את C. יכול להגיע ל-B או ל-A בכמה דרכים, אבל כדי הגיע בצורה מהירה יותר



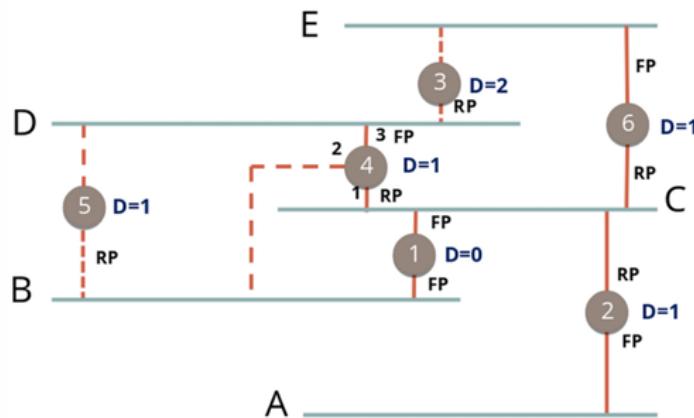
עבשו נחבר את D. C יכול להתחבר ל- switch root ? דרך 3,4,5. הוכיח מהירותה ביזור היא דרך ה-port השמאלי של 4 ונקבל:



אתם כבר חיכרנו דoor 3 כי המסלול דרך switch 6 יתאפשר רק אם switch 3 יתאפשר.

התמודדות עם תקלת

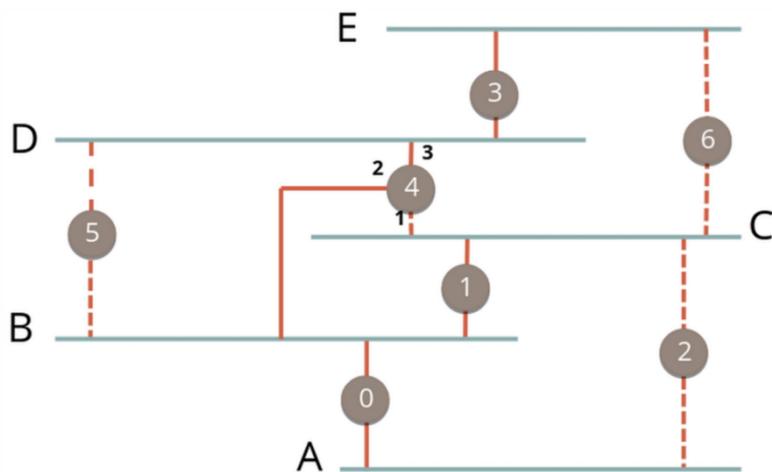
מה קורה אם 0 switch נופל?



אפשר להתחיל מהתחלה, אבל אפשר גם להסתכל מה הושפעו:

- עבשו 1 switch יփוך להיות השורש (הבא בתור עם ID נמוך ביותר).
- A היה מחובר דרך 0 switch, לכן עבשו יהיה חייב לעבור דרך 2 switch.
- ננתק את 2 port של 4 switch כי עבשו יהיה שווין באורך המסלולים דרך 1 port או 2 port, ולכן 1 port יבחר להיות RP.
- כדי להגיע מ-E ל-1 switch יותר מהר מאשר דרך 6 switch וולכן נקבע את 3 switch.

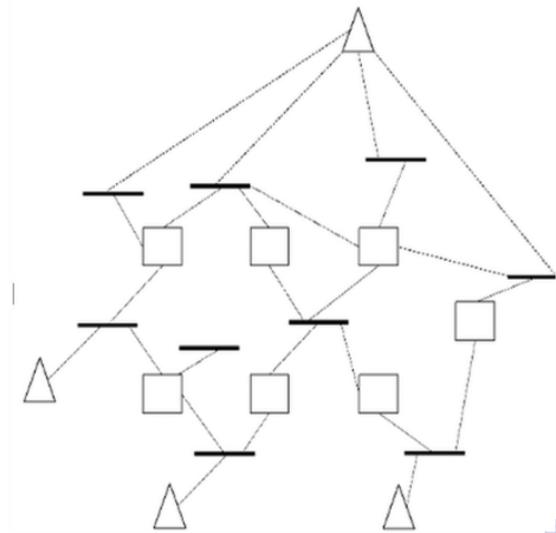
מה קורה אם 1 switch נופל?



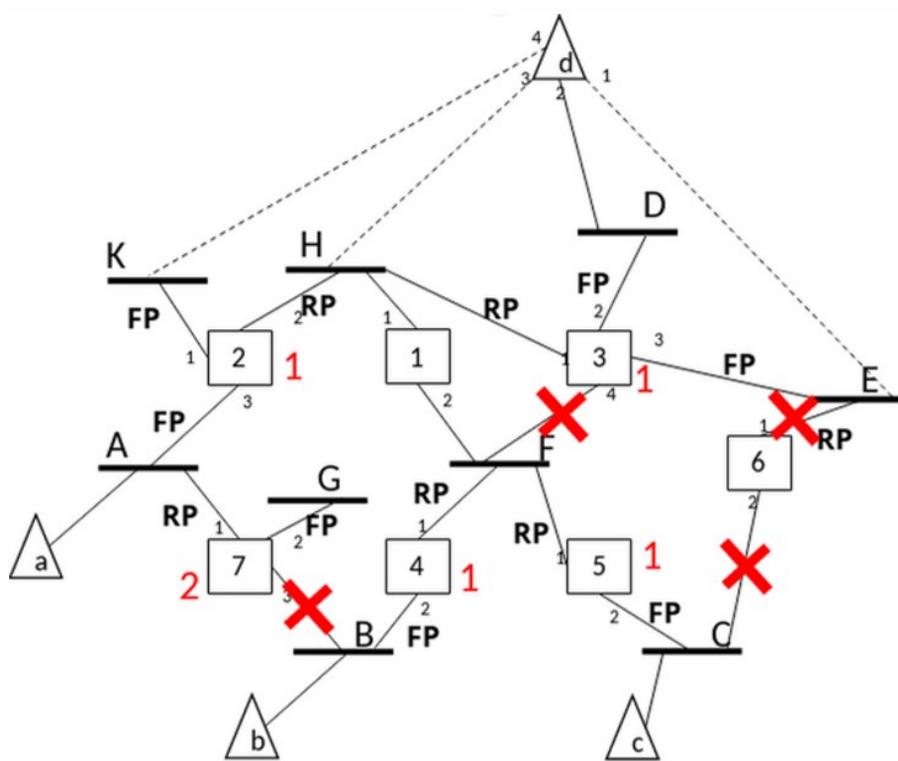
שאלה מבחן

נתונה רשות הבניה באופן הבא:

כל קו בתשיים הוא LAN, המשולשים הם מחשבי קצה, והרביעים הם switches.



בלל אחד מהקודקודים בתובת MAC, והם מחוברים ל-LANs או חלום דרך כמה LANs. אם למחשב ס (כמו למשל המשולש העליון) יש כמה פורטים, נסמן את כתובות ה-MAC של הפורט כר: (3,b,3). **ה:right STP**
כasher הקווים המהווים קווים מנוטקים.



עכשו הריצו את STP שוב, אבל כשהקווים מחוברים.

התשובה היא שלא יהיה הבדל לעומת הרצאה הקודמת, כי המחשב לא switch, הוא לא עבר הודעות להלאה – אם קיבל הודעה שסמנעת אליו יתיחס אליה, אחרת יתעלם.

מלאו את הטבלה עבור 1 switch (בנהנזה שהטבלאות בסו�יצ'ים כבר מלאות):

MAC Address	MAC(a)	MAC(b)	MAC(c)	MAC(d,1)	MAC(d,2)	MAC(d,3)	MAC(d,4)
Port							

נចטרך להחליט דרך أي port נגיע לכל אחד מכתובות ה-MAC. נשתמש בעץ שיצרנו:

MAC Address	MAC(a)	MAC(b)	MAC(c)	MAC(d,1)	MAC(d,2)	MAC(d,3)	MAC(d,4)
Port	1	2	2	1	1	1	1

באומרנו אופן, מילאנו את הטבלה עבור 2 switch:

MAC Address	MAC(a)	MAC(b)	MAC(c)	MAC(d,1)	MAC(d,2)	MAC(d,3)	MAC(d,4)
Port	3	2	2	2	2	2	1