**שאלה 2:**

1. ( => ) נסמן את הID של הroot בתור R, ולכן לכל Bridge אחר בעץ, יש ID=B ומתקיים ש-B>R. כל הודעת STP שתגיע לroot תהיה מן הצורה: (B,X,Y) ואנו יודעים שהאלגוריתם של הפרוטוקול "יפסול" את כל ההודעות האלו מפני שההודעה שתתחיל בR טובה יותר ולכן לעולם אףbridge אחר לא יעודכן להיות הroot port. מכאן נסיק כי אם הbridge הינו root אזי אין לו root port.

( <= ) נוכיח באינדוקציה על המרחק מן הroot (R) בעץ כי לכל bridge אחר במרחק גדול או שווה 1 יש root port:

בסיס:

כלbridge במרחק 1 מן הroot, מקבל ממנו הודעה מן הצורה (R,X,Y) ולכן הוא מעדכן את הport שממנו התקבלה ההודעה הזו להיות הroot port שלו, כי תמיד הID שלו יהיה גדול מR.

צעד:

נניח כי ההנחה מתקיימת עבור מרחק n, כלומר יש bridge שנסמנו בID M במרחק n מן הroot, ונוכיח את הטענה עבור bridge שנסמנו בID N שנמצא במרחק n+1 מן הroot.

מהנחת האינדוקציה, קיים root port עבור M (ששונה כמובן מהroot port של N, אחרת N היה במרחק n מן הroot ולא n+1). לכן בזמן הרצת האלגוריתם והתייצבותו, M קיבל הודעה מן הצורה (R,X,Y) דרך הroot port שלו וכך הוא יודע שR הינו הroot. לכן ניכר כי N יקבל הודעה מM מן הצורה (R,X,Y) וידע שR הינו הroot, כי לא משנה איזו הודעה אחרת הוא יקבל, R הינו הID הנמוך ביותר. לכן R הינו הroot של N וroot port שלו הינו הport שמחבר בינו לבין M.

קיבלנו כי לכל bridge שאינו root יש root port למעט הroot bridge.

1. נניח בשלילה כי קיים port שהbridge מגדיר אותו להיות גם designated וגם root.

נסמן את ההודעה הטובה ביותר שקיבל bridge זה ב-(X­min,Ymin,min). הודעה זו מקיימת שלכל הודעה אחרת, (Xi,Yi,i) מתקיים (X­min,Ymin,min)< (Xi,Yi,i). עבור i=j כזה, תהא ההודעה: (Xj,Yj,j) שהתקבלה על הפורט הנתון, מתוך זה שהפורט הינו designated. מפני שהport הוא גם root לפי ההנחה, נקבל כי הודעה זה מקיימת שלכל הודעה אחרת, (Xi,Yi,i) מתקיים (Xj,Yj,j)< (Xi,Yi,i), בפרט עבור

(Xi,Yi,i)= (X­min,Ymin,min)בסתירה.

**שאלה 3:**

נוסיף עוד שדה להודעה שיהיה השדה הרביעי שלה, שיציין את מס' הport ממנו הגיעה ההודעה. ההשוואה תתבצע כמו בSTP המקורי, כלומר שדה-שדה באופן לקסיקוגרפי, וכך בהינתן כמה הודעות, נבחר את השדה הרביעי בהודעה הטובה ביותר להיות הdesignated port (בהינתן כמובן שהbridge איננו root), ואת שאר הportים ניתן להגדיר כblocking.