Minimum Spanning Tree - Prim Algorithm

האלגוריתם של פרים הוא אלגוריתם **חמדני** המשמש למציאת עץ פורש מינימלי בגרף משוקלל לא מכוון. האלגוריתם מתחיל את בניית העץ מקדקוד פתיחה שנבחר באקראי. בכל צעד האלגוריתם מוסיף לעץ את הצלע בעלת המשקל המינימלי מבין אלה היוצאות מקדקודי העץ ולא סוגרות מעגל,

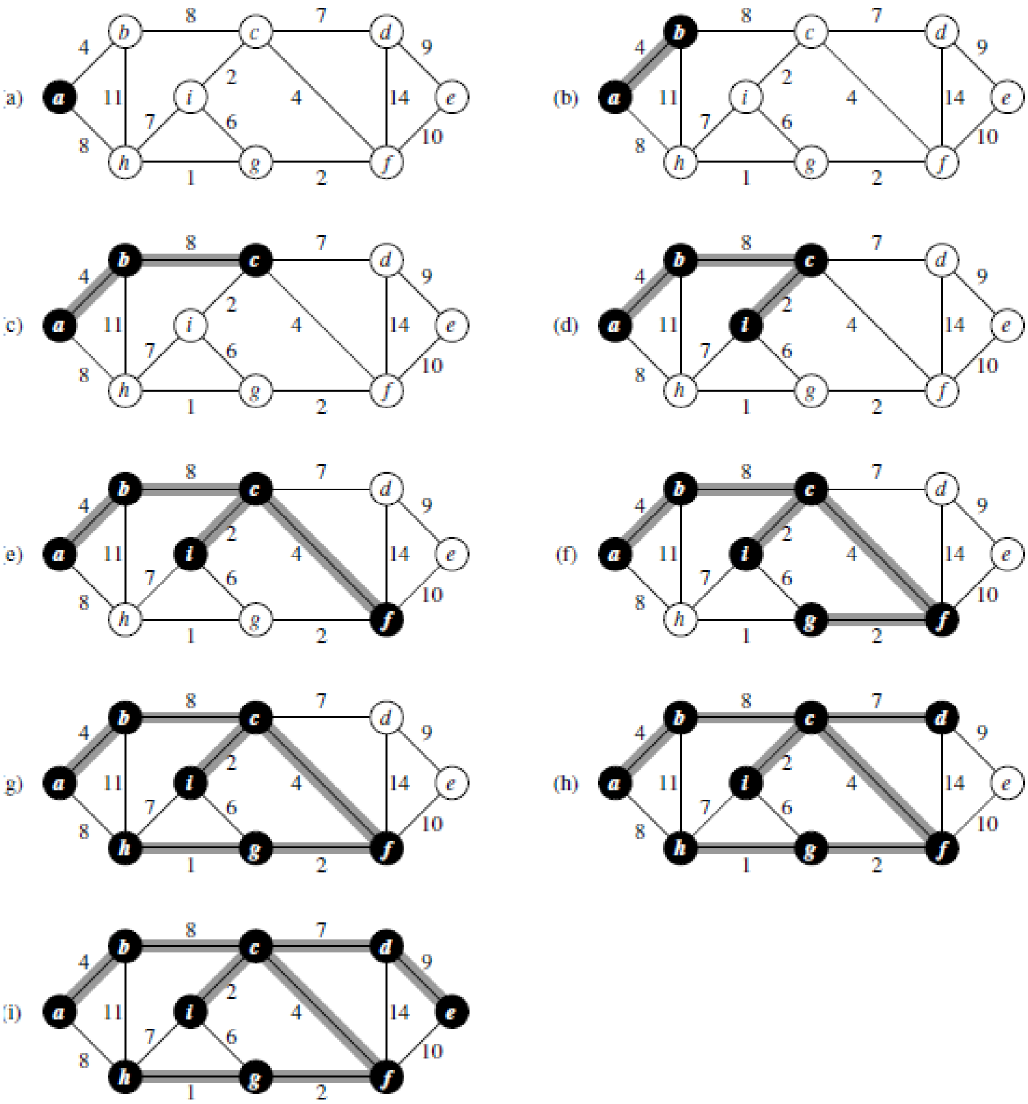
יותר פורמאלי:

1. בוחרים בקראי קדקוד כלשהו s ומוסיפים אותו לעץ:

Initialization: $T = \{s\}$

2. כול עוד T מכיל פחות מ- n-1 קודקודים מוסיפים ל-T צלע בעלת משקל מינימאלי, כך שבדיוק קדקוד אחד נמצא ב-T וקדקוד אחר לא ב-T.

$T = T \cup \{a,b\}$: $((a \in T \text{ and } b \notin T) \text{ or } (a \notin T \text{ and } b \in T)) \text{ and weight}(a,b) \text{->min}$



Prim pseudo code $(m = |E|,\ n = |V|)$

```
Prim(G, root)
      Edge T[n-1]<-empty tree (array of edges)
      numEdges = 0
      for each v in V(G, root) //O(n)
          visited[v] = false
          key[v] = infinity
          parent(v) = NIL
      end-for
      key[root] = 0
      Q <- V(G)// Q Min Heap, Q keyed by key[v] //O(n)
      while (Q != empty && numEdges<n-1) //O(m(log2(n))
          u = Q.extractMin() //O(log2(n))
          for each v in Adj(u)
              if (visited[v]==false && key[v] > weight(u,v))
                  key[v] = weight (u,v)
                  parent [v] = u
                  Q.decreaseKey(v, weight(u,v)) //O(log2(n))
              end-if
           end-for
           visited[u] = true
           x=Q.getMin()
           T[numEdges++]=(parent[x], x)
      end-while

end-Prim
```

שלבים של האלגוריתם בהתאם לדוגמא:

סימונים:

משבצת אפורה – קדקוד עזב את התור Q.

אות אדומה – מצב הקדקוד השתנה בהשוואה לשלב הקודם.

אתחול

קדקודים	Q	P	key
a	a	NIL	0
b	b	NIL	∞
c	c	NIL	∞
d	d	NIL	∞
e	e	NIL	∞
f	f	NIL	∞
g	g	NIL	∞
h	h	NIL	∞
i	i	NIL	∞

שלב 1

קדקודים	Q	P	key
a		NIL	0
b	b	a	4
c	c	NIL	∞
d	d	NIL	∞
e	e	NIL	∞
f	f	NIL	∞
g	g	NIL	∞
h	h	a	8
i	i	NIL	∞

שלב 2

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b	c	c
∞	NIL	d	d
∞	NIL	e	e
∞	NIL	f	f
∞	NIL	g	g
8	a	h	h
∞	NIL	i	I

שלב 3

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b		c
7	c	d	d
∞	NIL	e	e
4	c	f	f
∞	NIL	g	g
8	a	h	h
2	c	i	i

שלב 4

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b		c
7	c	d	d
∞	NIL	e	e
4	c	f	f
6	i	g	g
7	i	h	h
2	c		I

שלב 5

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b		c
7	c	d	d
10	f	e	e
4	c		f
2	f	g	g
7	i	h	h
2	c		i

שלב 6

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b		c
7	c	d	d
10	f	e	e
4	c		f
2	f		g
1	g	h	h
2	c		i

שלב 7

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b		c
7	c	d	d
10	f	e	e
4	c		f
2	f		g
1	g		h
2	c		i

שלב 8

key	P	Q	קדקודים
0	NIL		A
4	a		B
8	b		C
7	c		D
9	d	e	E
4	c		F
2	f		G
1	g		H
2	c		i

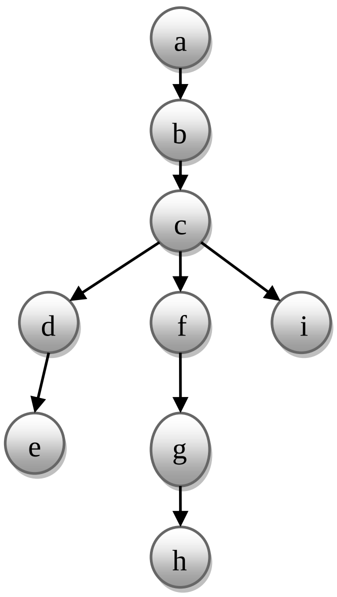
שלב 9

key	P	Q	קדקודים
0	NIL		a
4	a		b
8	b		c
7	c		d
9	d		e
4	c		f
2	f		g
1	g		h
2	c		i

<u>התוצאה</u>:

קבלנו עץ פורש מינימאלי שמשקלו 37 (ניתן לבדוק לפי האיור או לפי הסכום של שדות key).

לפי מערך P של קדקודי אבות (parent vertices) ניתן לבנות עץ:



סיבוכיות:

בעת מימוש האלגוריתם נעשה שימוש בערימה שמתוכה מוציאים בכל פעם את הצלע המינימלית. אם משתמשים תור עדיפויות (ערימה בינומית) סיבוכיות האלגוריתם תהיה $O(|E|*\log_2|V|)$, (כאשר |E| הוא מספר הצלעות ו-|V| הוא מספר הקדקודים).

באופן כללי היעילות של האלגוריתם של פרים טובה מזו של האלגוריתם של קרוסקל. למרות זאת, אם הקלט כבר ממוין לפי משקלי הצלעות או כאשר ניתן למיין אותם בזמן לינארי, אזי האלגוריתם של קרוסקל יהיה מהיר יותר. הסיבוכיות של אלגוריתם של קרוסקל ללא מיון היא $O(m \cdot \alpha(n))$.

הסיבוכיות של אלגוריתם של קרוסקל עם מיון היא תלויה בסיבוכיות של מיון ו שווה ל- $O(|E|\log_2(|V|))$.

הוכחת נכונות של אלגוריתם של פרים.

נוכיח שבכל שלב שאנו מוסיפים צלע חדשה ל-T אנו מקבלים תת-עץ של עץ פורש מינימאלי כלשהו M.

<u>הוכחה באינדוקציה.</u>

א) בסיס. בשלב ראשון של האלגוריתם אנו מוציאים מתור עדיפויות Q (min heap) את שורש העץ (root), שהוא שייך לכל עץ פורש מינימאלי.

ב) נניח שבשלב כלשהו של פרים קבלנו T תת-עץ של M ( ).

ג) בשלב הבא של **פרים** אנו מוסיפים צלע $e = (a,b)$ ל-T. אם $e \in M$ אז גם $T \cup \{e = (a,b)\} \subseteq M$ ובזה סיימנו את ההוכחה. נניח ש-$e \notin M$. נשים לב כי לפי אלגוריתם של פרים מוסיפים צלע שרק קצה אחד שלה שייך ל-T, ללא הגבלת ככליות נניח כי $a \in T$ ו-$b \notin T$. נוסיף e ל-M, נקבל מעגל. במעגל זה יש מסלול P שמחבר את קדקוד a עם b ובמעגל זה קיימת צלע $g = (x,y) \in P$ שרק קדקוד אחד שלה שייך ל-T. אלגוריתם של פרים היה יכול להוסיף g ל-T, אבל הוא בחר ב-e, כלומר weight(e) ≤ weight(g). נבנה עץ חדש $M' = M \cup \{e\} - \{g\}$ שמשקלו קטן או שווה למשקל של M. אבל M הוא בעל משקל מינימאלי, לכן weight(M) = weight(M') ו-M' הוא גם עץ פורש מינימאלי שמכיל את T: $T \cup \{e\} \subseteq M'$. מש"ל.