<u>תרגול מס' 1 בקורס אלגוריתמים</u>

מתרגל: דן פלדמן

שעות קבלה: מתי שרוצים, בתאום מראש.

עזרה טלפונית: מתי שרוצים, 0528-550211

מייל: dannyf@post.tau.ac.il

אתר: http://www.cs.tau.ac.il/~dannyf

המחשב שלי ברשת של מדעי המחשב: Windows: Start->Run.. \\pc-sharir1

<u>הגדרה :</u> מעגל אוילר בגרף הוא מעגל (לא בהכרח פשוט) שעובר על כל קשת בדיוק פעם אחת.

<u>מסלול אוילר :</u> כנ"ל עם מסלול (לא בהכרח פשוט) במקום מעגל

<u>טענה :</u> גרף לא מכוון וקשיר $G=(V,E)$ מכיל מעגל אוילר <==> כל הדרגות בו זוגיות.

בנוסף, אם המעגל מכיל מעגל אוילר, ניתן למצוא אותו בזמן $O(|E|)$

<u>הוכחה :</u>

=> נתחיל לעבור על מעגל אוילר מקודקוד v כלשהו. לכל $u\neq v$, כל פעם שעוברים דרך u נכנסים אליו בקשת אחת ויוצאים ממנו בקשת אחרת, 2 קשתות בכל פעם, משמע $d(u)$ זוגי.

עבור v, הקשת הראשונה תורמת 1, האחרונה עד 1, וכל מעבר נוסף דרך v תורם 2 לדרגה. משמע: גם $d(v)$ זוגי.

<= נניח כל הדרגות זוגיות ונראה אלגוריתם למציאת מעגל אוילר.

א. נבחר קודקוד u כלשהו ונתחיל ליצור מעגל באופן הבא: נבחר קשת (u,v), נוציא אותה מהגרף ונוסיף למעגל. נעבור ל-v ונבחר קשת (v,w). נמשיך כך עד שנגיע חזרה ל-u.

ב. כל עוד נשארו קשתות, נבחר קודקוד על המעגל שנשארו לו קשתות, ונתחיל ממנו את (1) שוב. את המעגל שנקבל נצרף למעגל שהיה קודם. נחזור על ב. עד שלא יישארו קשתות.

<u>הוכחת נכונות :</u>

- בכל פעם שמתחילים את שלב א' אז כל הדרגות זוגיות ומתחילים מקודקוד שנשארו לו קשתות (הדרגות זוגיות כי הן זוגיות בהתחלה, ובכל פעם בשלב א' מורידים מהגרף מעגל ולכן הדרגות נשארו זוגיות.
- אם מתחילים את שלב א' מקודקוד u, אזי בכל פעם שמגיעים ל-$v\neq u$ ניתן להמשיך (כי לפני שהגענו ל-v הדרגה שלו היתה זוגית: נכנסנו דרך קשת אחת ולכן נשארה לפחות קשת אחת שעליה נוכל לצאת).
- כל עוד לא חזרנו ל-u ניתן להמשיך, ומכיוון שבכל צעד מנתקים קשת ומס' הקשתות הוא סופי, בשלב מסוים לא נוכל להמשיך, כלומר נסיים ב-u.
- בכל פעם שמסיימים את שלב א' יש לנו מעגל שעובר על כל קשת בדיוק פעם אחת.
- אם נשארו קשתות אזי יש קודקוד על המעגל שנשארו לו קשתות, כי הגרף קשיר. לכן נוכל להמשיך ולהגדיל את המעגל עד שנעבור על כל הקשתות => נקבל מעגל אוילר.

זמן ריצה:

כדי לנתח את זמן הריצה צריך להסביר את פרטי המימוש.

- נחזיק את הגרף ברשימת שכנויות עם מצביעים
- את המעגל שניצור נחזיק ברשימה מקושרת
- בכל צעד של שלב א' נבחר את השכן הראשון ברשימה ב-$O(1)$ זמן. בכל צעד מוסיפים קשת, ולכן סה"כ במשך ריצת האלגוריתמים זה ייקח $O(|E|)$.
- בשביל שלב ב' נחזיק מצביע שיצביע על קודקוד במעגל שממנו אנחנו יוצאים. בהתחלה נצביע לקודקוד u.
- כל פעם נבדוק אם לקודקוד שעליו מצביעים נשארו קשתות. אם לא – נקדם מצביע על המעגל עד שנמצא קודקוד שנשארו לו קשתות. את המעגל החדש נוסיף אחרי המצביע. חיפוש בשלב הבא יתחיל ממצביע זה.
- סה"כ מעבר על אחד כל מעגל האוילר ב-$O(|E|)$ זמן.

<u>טענה:</u> גרף לא מכוון וקשיר מכיל מסלול אוילר <==> בגרף 2 קודקודים מדרגה אי-זוגית והיתר מדרגה זוגית.

<u>הוכחה:</u> ==> כמו עבור מעגל, רק שכאן הצומת הראשון שונה מהאחרון ולכן דרגתו אי זוגית.

<== נניח u ו-v מדרגה אי זוגית – נוסיף קשת (u,v). כעת כל הדרגות זוגיות, לכן יש מעגל אוילר. אם נמחק ממנו את הקשת שהוספנו נקבל מסלול אוילר בגרף המקורי.

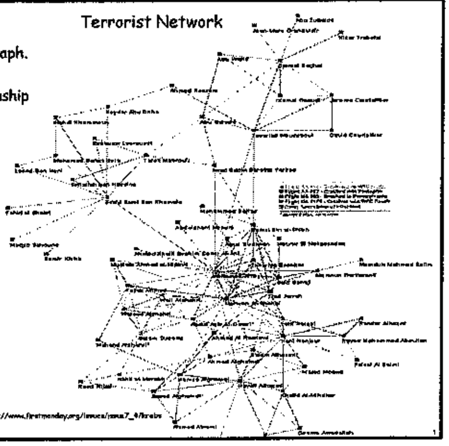
<u>טענה:</u>

1) גרף מכוון וקשיר מכיל מעגל אוילר <==> לכל v ב-V: $din(v)=dout(v)$.

2) גרף מכוון וקשיר מכיל מסלול אוילר <==> קיימים 2 צמתים w,v כך ש-$din(v)=dout(v)+1$, $din(w)+1=dout(w)$ וגם לכל שאר הקודקודים $u\neq v,w$ מתקיים $din(u)=dout(u)$.

Terrorist Network

Social network graph.
- Node: people.
- Edge: relationship between two people.



Reference: Valdis Krebs, http://www.firstmonday.org/issues/issue7_4/krebs

---

Algorithms
Tirgul 1

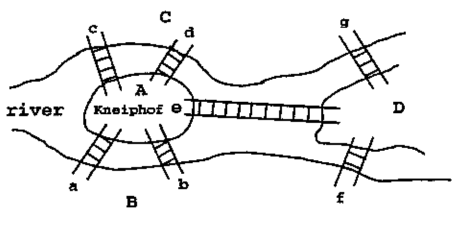
Euler Cycles/Trails

Sources:
academics.smcvt.edu/jellis-monaghan/combo2/class%20notes/tucker%202-1.ppt)
http://par.cse.nsysu.edu.tw/~cbyang/course/advprog/chap2graphs.ppt

11/6/2008 Tucker Sec. 2.1 2

יש ציור, האם ניתן לצייר אותו בלי להרים
את העיפרון או לעבור על אותה נקודה פעמיים
בצורה יותר פרקטית משתמשים בזה בעת
הדפסת כרטיסים אלקטרוניים (כדי לחסוך בזמן
המעבר במעגלים).
הבעיה: בהינתן מיליון דפים באינטרנט,
האם יש מסלול שעובר בין כל הדפים
מבלי לעבור על אותה הקשת פעמיים.

Koenigsberg bridge problem



- Is it possible to walk across all bridges exactly once from some land area and returning to the starting land area ?

11/6/2008 Tucker Sec. 2.1 3

גרף מורכב מ-2 קבוצות V ו-E.
V מכיל את כל הצמתים ו-E מכיל את
כל הקשתות (זוגות סדורים).
$E \subseteq V^2 = V \times V$.
אם הזוגות סדורים, נגיד שהגרף הוא מכוון.
אחרת (אם הסדר אינו חשוב) הגרף אינו מכוון
אנחנו נניח (x,x) לא קיים (אחרת) שהגרף
הוא גרף פשוט, כלומר, אין בו לולאות
או קשתות כפולות.
בעיית הגשרים - בהינתן הגרף, האם יש מסלול שמאפשר מעבר על כל הגשרים
מבלי לעבור על אותו גשר פעמיים.

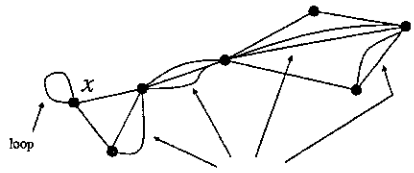
Definitions

Multigraph:

A graph in which;

a) multiple edges are allowed, that is, two or more edges can join the same two vertices.

b) loops are allowed, that is edges of the form (x, x).

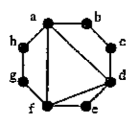


כאשר מדברים על מולטי-גרף
הכוונה היא שמתאפשרת חזרה
על קשתות.

Cycle:

A sequence of consecutively linked edges whose starting vertex is the ending vertex, in which no edge can appear more than once. However, vertices can be repeated.

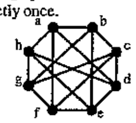
e.g.



$C = a\text{-}h\text{-}g\text{-}f\text{-}e\text{-}d\text{-}f\text{-}a$

Euler Cycle:

A path through a graph which starts and ends at the same vertex and includes every edge exactly once.

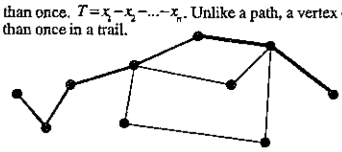


$E = a\text{-}b\text{-}c\text{-}d\text{-}e\text{-}f\text{-}g\text{-}h\text{-}a\text{-}d\text{-}h\text{-}e\text{-}b\text{-}g\text{-}c\text{-}f\text{-}a$

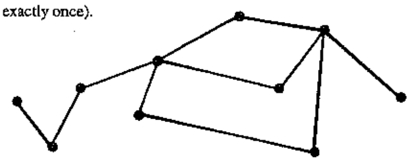
מעגל הוא מסלול בו המסלול ניתן
לעבור על אותה הקודקוד פעמיים רק
לא על אותה הקשת ושהוא מתחיל
ונגמר באותה נקודה.
מעגל אוילר - הוא מסלול שעובר
על כל קשתות הגרף

Trail: A sequence of consecutively linked edges in which no edge can appear more than once. $T = x_1 - x_2 - \ldots - x_n$. Unlike a path, a vertex can be visited more than once in a trail.

e.g.



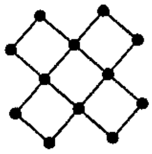
Euler Trail:- A trail that contains all the edges in a graph (and visits each edge exactly once).



מסלה - מאפשר מעבר בין הצמתים (וקודקוד
באותו הצומת פעמיים) רק אסור
לחזור על אותה הקשת פעמיים.
מסילת אוילר - ~~מ...~~ מסילה העוברת
על כל הקשתות בגרף
בשבוע נשאל איך לעשות את הדבר
על מסלול אוילר. (השאלה הבאה)

Theorem

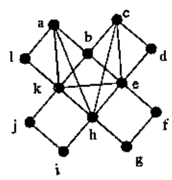
- An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.



➢ This is a connected graph with all vertices of even degree, therefore it must have an Euler cycle.

אם יש גרף לא מכוון, אז קיים
מעגל אוילר אמ"מ יש לכל צומת
מספר זוגי של קשתות והוא קשיר.

Example



Multigraph H

➢ Here the multigraph H does not have an Euler cycle because there are 2 odd degree vertices, namely vertex 'k' and vertex 'a'.

➢ Therefore if and only if the edge (k , a) were completed then H would have an Euler Cycle.

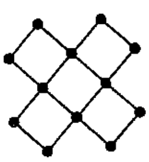
בדוגמה זו (במקור) לא קיימת קשת
בין a ל-k ולכן אין מעגל אוילר.
עם זאת, אם נוסיף את הקשת החסרה
הזו, נוכל למצוא מעגל אוילר
ולחזור במעבר נתונים

מ'

Proof

An Euler cycle ⟹ all vertices are connected and have even degree

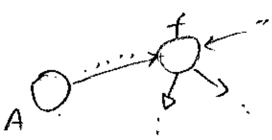
The Euler cycle connects all vertices, so the graph is connected. Every time the cycle enters a vertex it also exits, so the vertices have even degree in a whole complete cycle.



➢ This graph has an Euler cycle, therefore all the vertices are connected and have even degree.

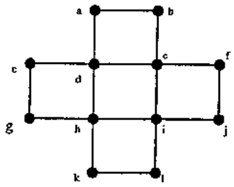
➢ We call a connected graph with all even degrees an ***Eulerian Graph.***

הוכחת הטענה
הנחה: יש מעגל אוילר לכן אני מתחילה
ומסיימת באותה נקודה (נסמן A).
נניח שקיימת צומת f שבה יש
מספר קשתות אי זוגי, אבל זה לא
אפשרי כי מכל צומת, אם נכנסו,
בטוח נצא (כי היא לא הצומת האחרון).
הצומת היחיד שבה זה לא קורה היא A אולי ~~...~~ יכולה
מתחילה מבלי להכנס (כי היא הראשונה) אבל
עם זאת, אנחנו בסוף גם נכנס אליה כי
חייבים (כי הוא גם אחרון).



Proof

All even and connected ⟹ Euler cycle

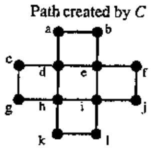


- Pick any point on G. (I will start with a)
- Since all vertices have an even degree, we are not forced to stop until we complete the cycle. (get back to the starting point)

הנחתי אם יש מספר זוגי של קשתות
בכל צומת, קיים מעגל אוילר.
נוכיח באמצעות אלגוריתם. ניקח
צומת כלשהי, תמיד תהיה לה קשת
כי סדרת הגרף אינו קשיר, ונמשיך לצומת
הבא, בטוח יש זוג קשת שמחוברת
אליה (כי בכל צומת מספר הקשתות זוגי)
לכן, לכל מקום שנגיע (פרט ל-a) תמיד נוכל לצאת למקום אחר.

Proof cont'd

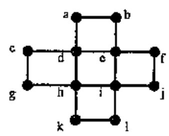
Path created by C



For example, let C equal the cycle created,

C = a-d-h-k-l-i-j-f-e-b-a

If there are any unused edges, and in this case there is, then create another cycle starting with any edge adjacent to C.



This is the cycle created by D in Blue

D = d-e-i-h-g-c-d

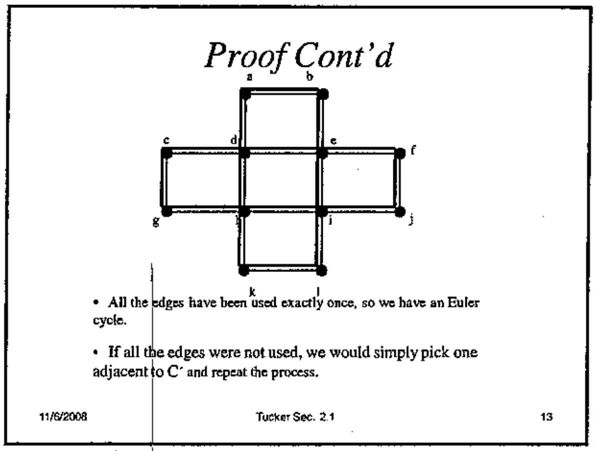
כדי לפתור את הבעיה שאנו רואים
להיתקע בה, נסתכל על הדוגמה הראשונה
שבה לא כל הקשתות מכוסות ונריץ
את האלגוריתם שוב. (מבלי לעבור על
הקשתות שעברנו עליהם קודם) נקבל מעגל
חדש שמתחיל ומסתיים ב- d. אם עדיין
לא עברנו על כל הקשתות (ממשיך הלאה.
לבסוף נשרשר את הכל ונקבל מעגל אוילר.

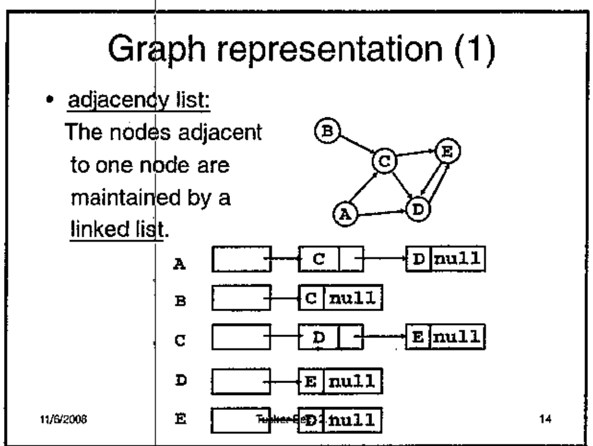
Proof cont'd

- Since C and D were originally connected, there must be at least one common vertex; pick one (point d)
- Build a new cycle C', by putting D into C at the common point to get a larger cycle.

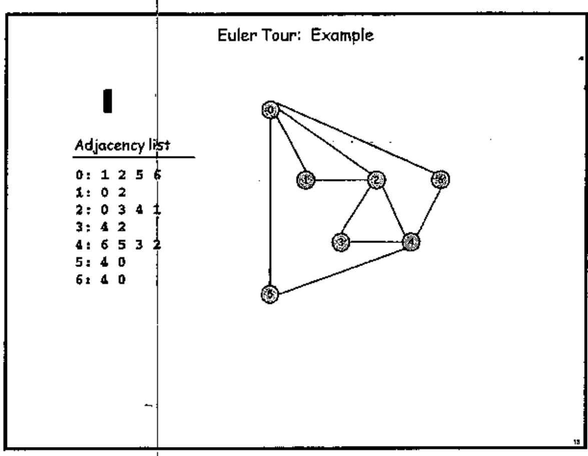
$$C' = D + C$$

C' is presented on the next slide

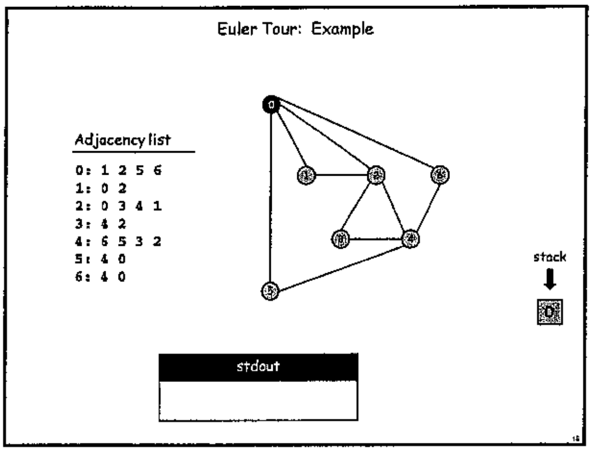


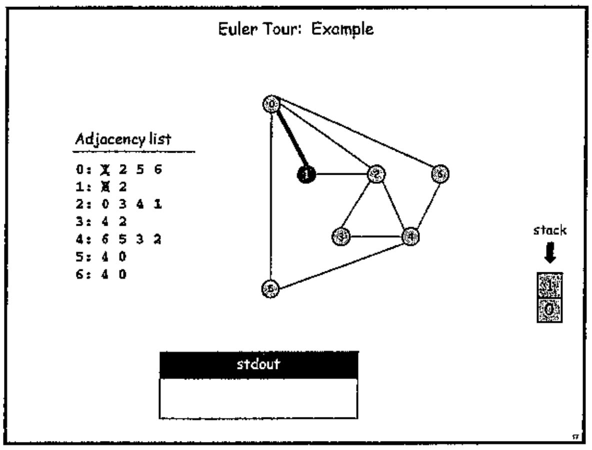


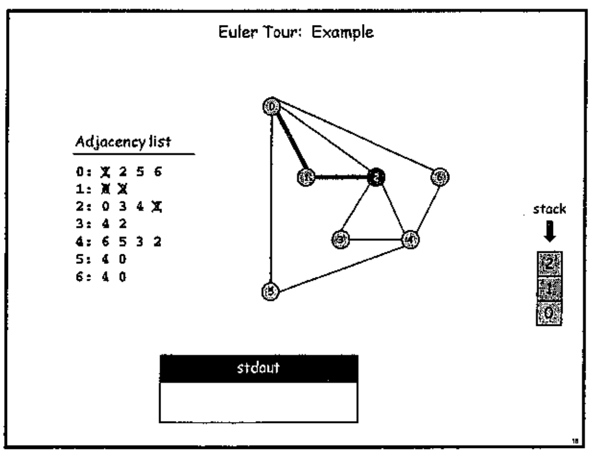
אלגוריתם המפרק את הבדיקה בצורה
יעילה. קודם כל כדי לייצג את הגרף נשתמש
במערך שייצג את כל הקודקודים ולכל קודקוד
יש לו רשימה מקושרת לשכנים שלו
השכנים שלו. יוצא אחד, הוא באמצעות מעריכה.

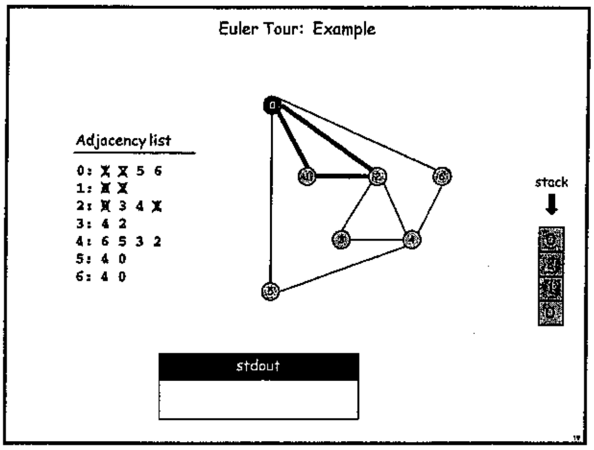


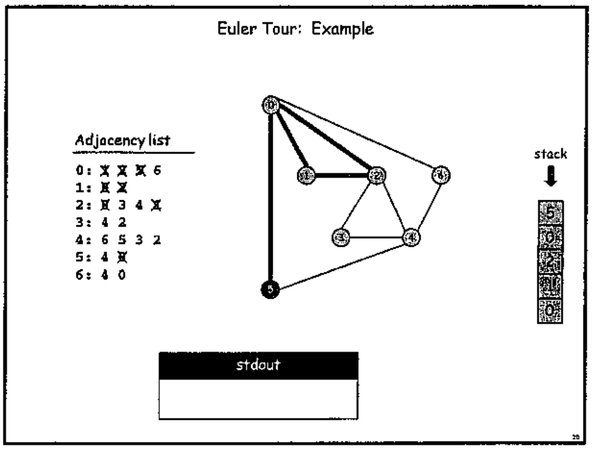
האלגוריתם שנציג ירוץ במהירות של $O(|V|+|E|)$.
השיטה היא באמצעות שימוש במחסנית.
נתחיל מהצומת הראשון ונלך אליו, נמחק
את הקופצת של הקשתות ונכניס דבר
זה נכניס למחסנית את הצומת. נמשיך
הלאה עד שנתקע עם רשימה ריקה.
בשלב הזה נתחיל לשחרר את המחסנית
ונדפיס כל אחד שיוצא. אם הצומת שהוצא של
המחסנית מכיל רשימה ריקה נדפיס אותו ונמשיך הלאה, אחרת נריץ שוב את הדבר
שתיארנו בהתחלה.

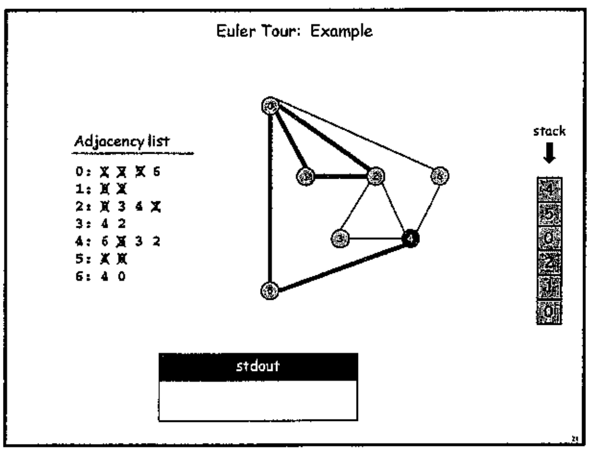

Euler Tour: Example
Adjacency list
0: 1 2 5 6
1: 0 2
2: 0 3 4 1
3: 4 2
4: 6 5 3 2
5: 4 0
6: 4 0
stack
0
stdout

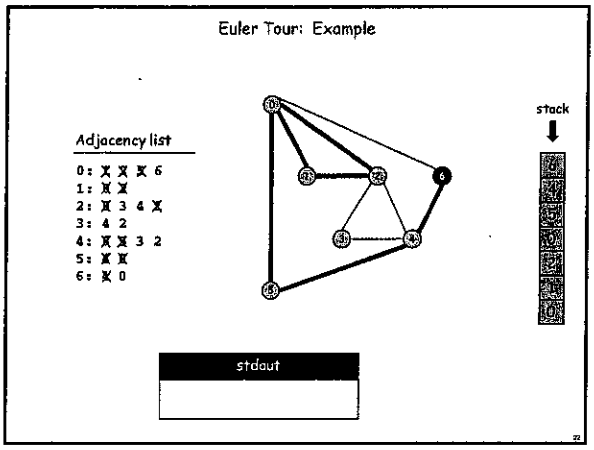

Euler Tour: Example
Adjacency list
0: ~~1~~ 2 5 6
1: ~~0~~ 2
2: 0 3 4 1
3: 4 2
4: 6 5 3 2
5: 4 0
6: 4 0
stack
1
0
stdout

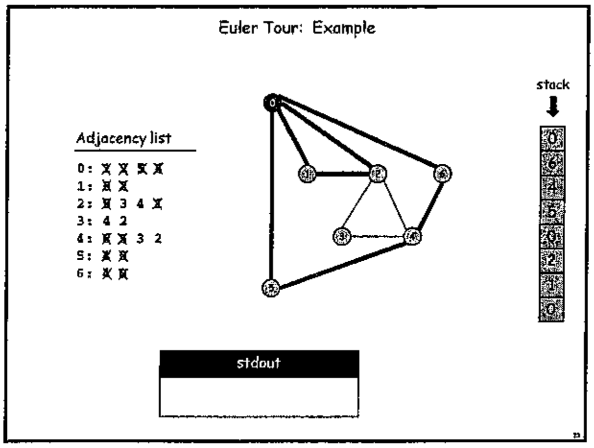

Euler Tour: Example
Adjacency list
0: ~~1~~ 2 5 6
1: ~~0~~ ~~2~~
2: 0 3 4 ~~1~~
3: 4 2
4: 6 5 3 2
5: 4 0
6: 4 0
stack
2
1
0
stdout

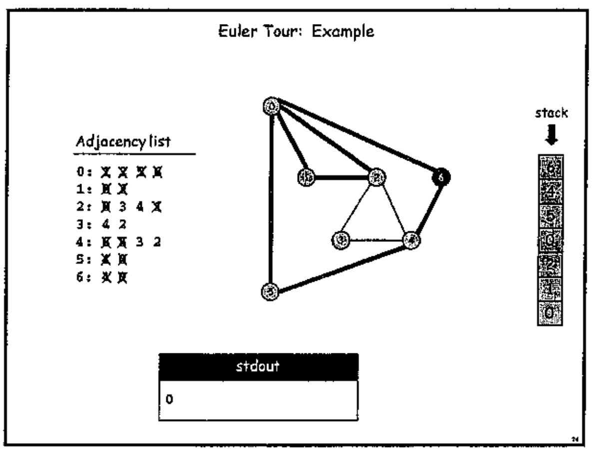

Euler Tour: Example
Adjacency list
0: X X 5 6
1: X X
2: X 3 4 X
3: 4 2
4: 6 5 3 2
5: 4 0
6: 4 0
stack
stdout

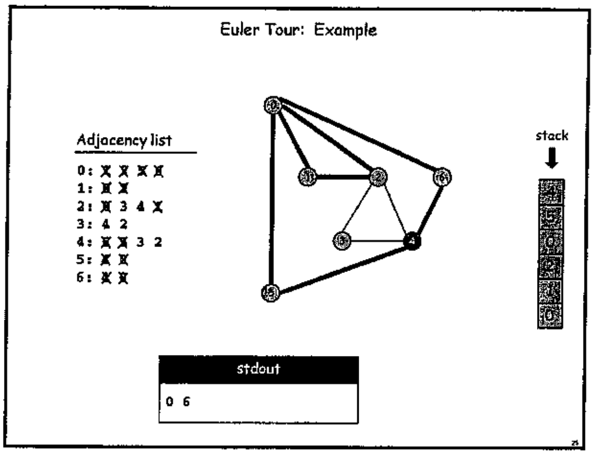

Euler Tour: Example
Adjacency list
0: X X X 6
1: X X
2: X 3 4 X
3: 4 2
4: 6 5 3 2
5: 4 X
6: 4 0
stack
stdout

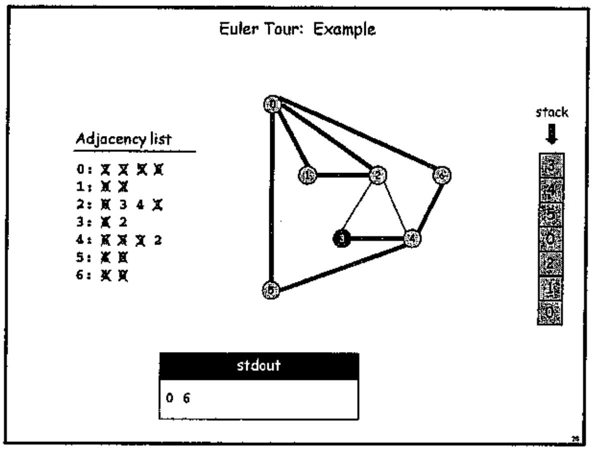

Euler Tour: Example
Adjacency list
0: X X X 6
1: X X
2: X 3 4 X
3: 4 2
4: 6 X 3 2
5: X X
6: 4 0
stack
stdout

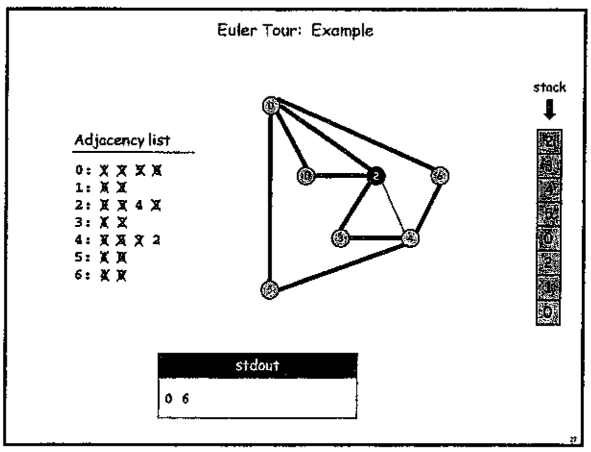

Euler Tour: Example
Adjacency list
stack
stdout

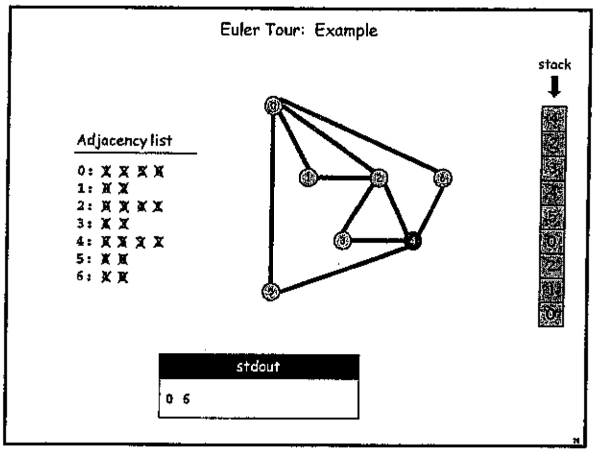

Euler Tour: Example
Adjacency list
stack
stdout

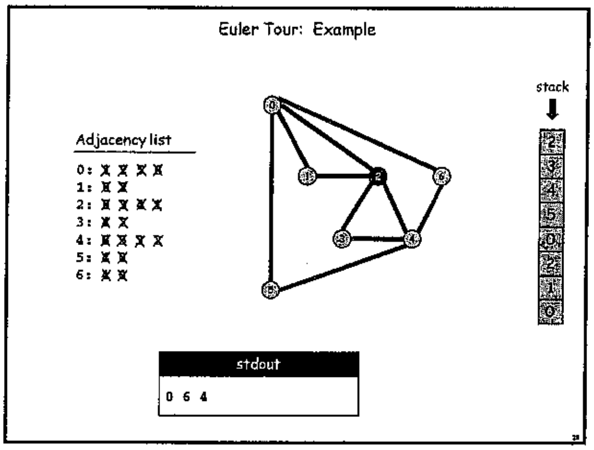

Euler Tour: Example
Adjacency list
stack
stdout
0

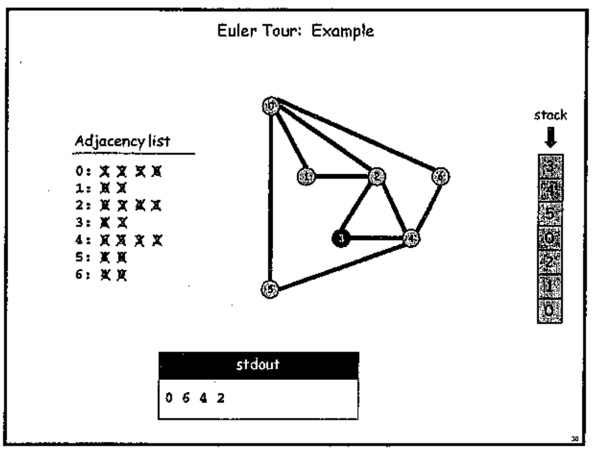

Euler Tour: Example
Adjacency list
0: X X X X
1: X X
2: X 3 4 X
3: 4 2
4: X X 3 2
5: X X
6: X X
stack
stdout
0 6

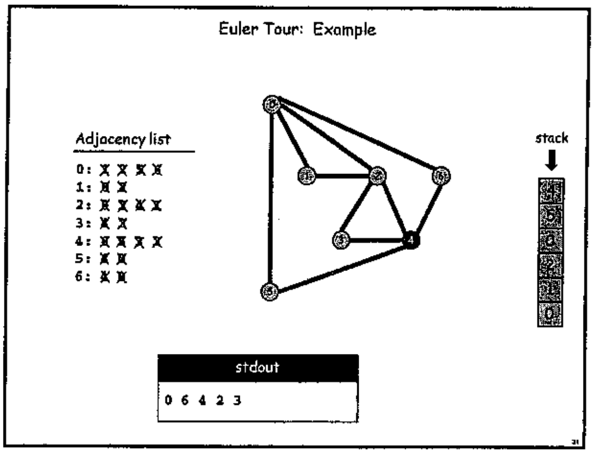

Euler Tour: Example
Adjacency list
0: X X X X
1: X X
2: X 3 4 X
3: X 2
4: X X X 2
5: X X
6: X X
stack
stdout
0 6

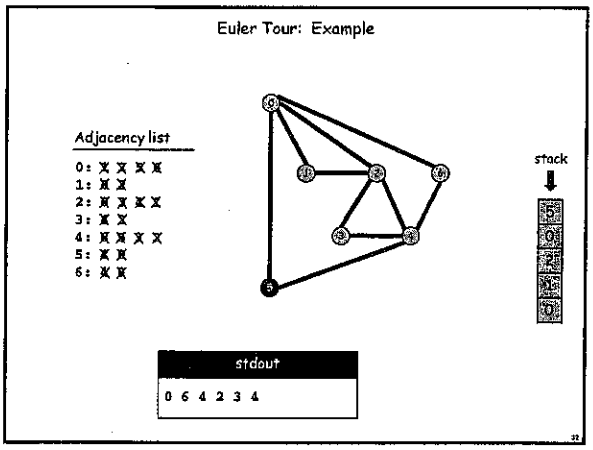

Euler Tour: Example
Adjacency list
0: X X X X
1: X X
2: X X 4 X
3: X X
4: X X X 2
5: X X
6: X X
stack
stdout
0 6

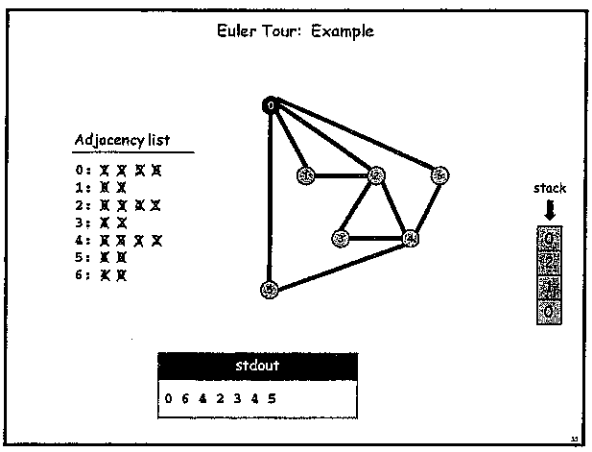

Euler Tour: Example
stack
Adjacency list
stdout
0 6

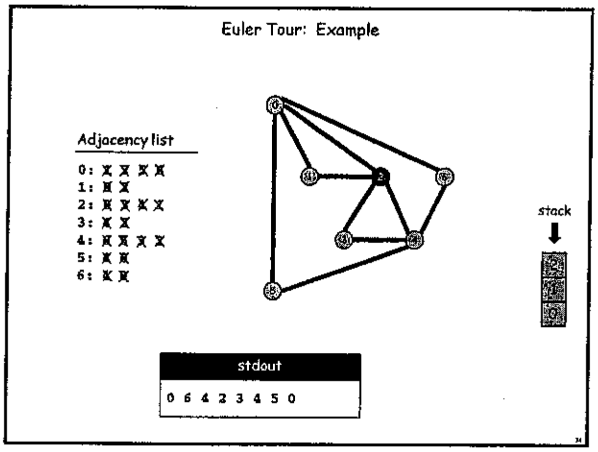

Euler Tour: Example
stack
Adjacency list
stdout
0 6 4

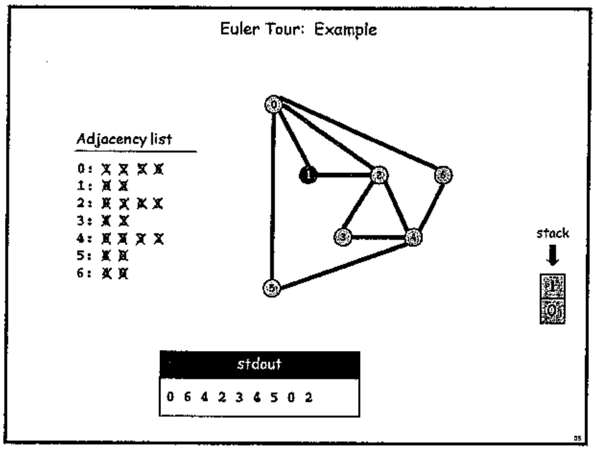

Euler Tour: Example
stack
Adjacency list
stdout
0 6 4 2

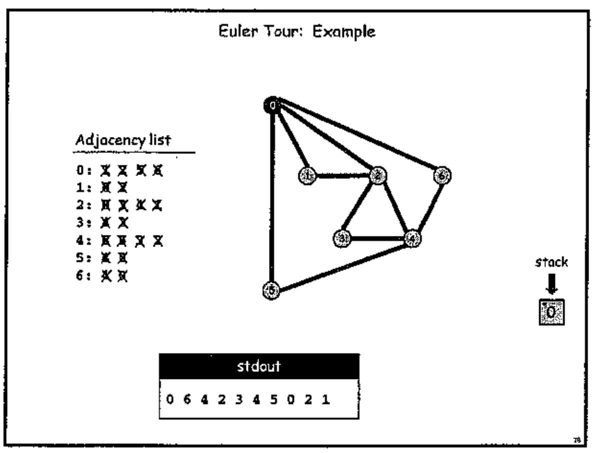

Euler Tour: Example
Adjacency list
stack
stdout
0 6 4 2 3


Euler Tour: Example
Adjacency list
stack
stdout
0 6 4 2 3 4


Euler Tour: Example
Adjacency list
stack
stdout
0 6 4 2 3 4 5


Euler Tour: Example
Adjacency list
stack
stdout
0 6 4 2 3 4 5 0

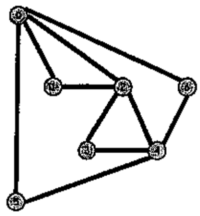

Euler Tour: Example
Adjacency list
stack
stdout
0 6 4 2 3 4 5 0 2


Euler Tour: Example
Adjacency list
stack
stdout
0 6 4 2 3 4 5 0 2 1

Euler Tour: Example

Adjacency list

0: X X X X
1: X X
2: X X X X
3: X X
4: X X X X
5: X X
6: X X



stdout

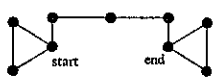
0 6 4 2 3 4 5 0 2 1 0

Corollary

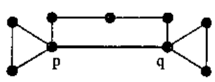
A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof

- Suppose a multigraph *H* has an Euler trail *T*, but not an Euler cycle.
- The starting and ending vertices of *T* must have odd degree while all other vertices have an even degree (by the same reasoning that showed all vertices with even degree in an Euler cycle). Also the graph must be connected.

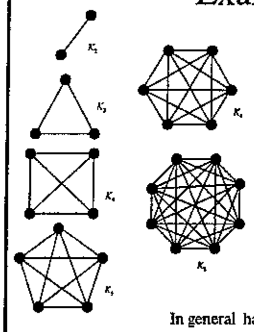


- On the other hand if a multigraph *H* is connected and has exactly two vertices, p and q, of odd degree, then to obtain a graph *H'*, add a supplementary edge (p,q) to *H*.



- *H'* is now connected and has all vertices of even degree. Therefore by the Euler cycle theorem, *H'* has an Euler cycle *C*.
- If you removed the edge (p,q) from *C*, this reduces the Euler cycle to an Euler trail that includes all of edges of *H*. Hence the corollary.

Examples



K_2: Euler trail – Yes- exactly two vertices of odd degree.
Euler cycle – No- vertices are not of even degree.

K_3: Euler trail – No- vertices are not of odd degree.
Euler cycle – Yes- even degree vertices.

K_4: Euler trail – No- more than two vertices of odd degree.
Euler cycle – No- there aren't any even degree vertices.

K_5: Euler trail – No- there are no vertices of odd degree.
Euler cycle – Yes- cycle is connected and even degree.

K_6: Euler trail – No- more than two vertices of odd degree.
Euler cycle – No- there aren't any even degree vertices.

K_7: Euler trail – No- more than two vertices of odd degree.
Euler cycle – No- there aren't any even degree vertices.

In general has K_n Euler cycle if n is an odd number.

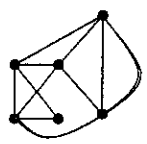
11/6/2008 Tucker Sec. 2.1 40

k_n הוא גרף עם n צמתים
כאשר בכל צומת יש קשתות לכל צומת
אחר.
כאשר n אי זוגי יש מעגל אוילר.

Class Exercise

Give an undirected 12 edge graph, that has an Euler cycle.

Answer



-All vertices have even degree
-Is connected

11/6/2008 Tucker Sec. 2.1 41

Class Exercises

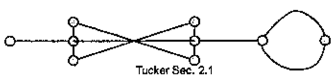
1. A) Can a graph with an Euler cycle have a bridge (an edge whose removal disconnects the graph)?

   B) Give an example of a 10-edge graph with an Euler trail that has a bridge.

Answers:

A) No. If you start on one side of the bridge and then you cross it the only way you can get back to where you started is to cross the bridge again. This means you have to travel the same edge twice and thus, a Euler cycle cannot exist.

B)



11/6/2008 Tucker Sec. 2.1 42

בנו גרף שקיים בו מסלול אוילר המקיים את התכונה הבאה:

– בבניית מסלול אוילר על-פי האלגוריתם לבניית מסלול אויילר, שלב ב' של האלגוריתם יכול להתבצע לפחות 3 פעמים.

11/6/2008 Tucker Sec. 2.1 43