Lecture #16 10/07/2016

- Reminding:

  * We use "solution" and "tour" when refering to a trip starting and ending at a site and visiting all other sites exactly once.

  * Last time we learnt the Brute-Force Algorithm which consists of listing all possible Hamilton circuits (there are $(N-1)!$ of them, N = #sites), computing the total cost of each, and picking the cheapest ones

    Note: It is convenient to split all these circuits into pairs, by reverting direction of all arrows.

  ! Drawback: This is hardly done by hands for $N>5$, while for $N \geq 25$ no computer will do this in a reasonable amount of time.

* It turns out that there is no other simple way to find to find the optimal solution in a faster way.

  However, there are several approxmate algorithms, which provide a solution, which is not optimal, but in many situations quite close to the optimal.

  Two of such Algorithms were discussed last time:

  - the Nearest-Neighbor Algorithm
  - the Repetitive Nearest-Neighbor Algorithm.

[Ask audience if anyone remembers how these algorithms work]

Remark 1: The number $\varepsilon = \frac{\text{Cost(tour)} - \text{Optimal cost}}{\text{Optimal cost}}$ is called a relative error of a tour.

Remark 2: - Emphasize that both algorithms apply to the case where the starting point is fixed or not
- Discuss how we rewrite circuit from any vertex

- Before we consider an example on application of the (Repetitive) Nearest-Neighbor Algorithm, let me make a few comments on the hwk:
  - you should remember how to rewrite a Hamilton circuit starting from any vertex
  - the only reasonable way to get a Hamilton path out of a Hamilton circuit is by erasing one of the edges in circuit.
    ! This is referred to as breaking a Hamilton circuit.
  - a Hamilton path arises from broken Hamilton circuit iff, there is an edge b/w the starting & ending point of the path.
  - there are no criteria for the existence of Hamilton paths/circuits, unlike for Euler paths/circuits. Hence if you have a q-n asking you if there exists a Hamilton path or circuit:
    - draw one if it exists
    or
    - provide a simple argument why you think it doesn't exist

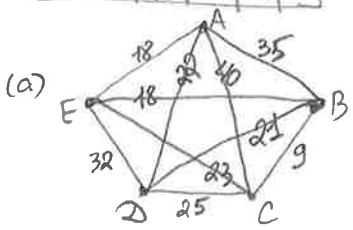
- Example 1: The following Table shows the distance b/w any two sites (out of 5 sites A, B, C, D, E)
  [Exercise 6.3.29]
  39

	A	B	C	D	E
A		35	40	22	18
B	35		9	21	18
C	40	9		25	23
D	22	21	25		32
E	18	18	23	32	

(a) Draw the corresponding weighted graph.

(b) Find the nearest-neighbor tour starting at A. Compute the total cost

(c) Find the nearest-neighbor tour starting at B and give the answer using A as the starting point. Total Cost = ?



(b) A-E-B-C-D-A. Total Cost: 18+18+9+25+22 = 92

(c) B-C-E-A-D-B. Total Cost: 9+23+18+22+21 = 93

A-D-B-C-E-A

! Actually, AEBCDA and its reverse ADCBEA are the optimal solutions

Lecture #16 10/07/2016

- Today we are gonna learn the last algorithm (also approximate!) [Section 6.5]

"The Cheapest-Link Algorithm"

Rmk: Here the word "Link" is a synonym of "edge", i.e. we could also call this algorithm as "edge of least cost algorithm".

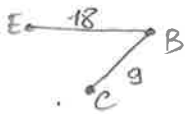
Let me explain how it works and then we will illustrate this by an example.

- Step 1: Pick the cheapest link (=edge) available.
  [if there are several of those, choose any]
  Highlight it in any visible color.
- Step 2, 3, ..., N-1: On each next step, we look for the cheapest link which has not been highlighted by now and which:
  (a) does not violate the partial-circuit rule
  (i.e. doesn't close a partial circuit)
  (b) does not violate the 3-edge rule
  (i.e. doesn't create three edges meeting at 1 vertex)
- Step N: Connect two vertices of the obtained Hamilton path to get a Hamilton circuit.

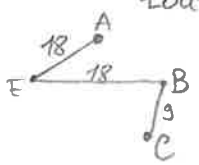
Example 2: For a weighted graph from Example 1, find the cheapest-link tour.

Step 1:

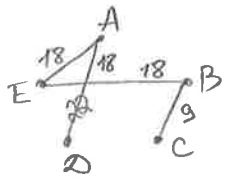
Step 2 (EB or EA)



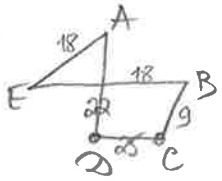
Step 3



Step 4



Step 5



Total Cost: 92

In this example, we obtained an optimal solution, but this not the case in general.

■