



DATA MULE SCHEDULING IN  
SENSOR NETWORKS  
I.E.  
THE TRAVELING SALESMAN  
PROBLEM

Prof. Tiziana Calamoneri  
Network Algorithms  
A.y. 2017/18

1



THE PROBLEM

2

### THE PROBLEM (1)

- A **sensor** is a device that detects and responds to some type of input from the physical environment.
- The specific input could be light, heat, motion, moisture, pressure, or any one of a great number of other environmental phenomena.
- The output is generally a signal that is converted to human-readable display at the sensor location or transmitted electronically over a network for reading or further processing.
- **Sensor networks** are dense wireless networks of small, low-cost sensors, which collect and disseminate environmental data.

3

### THE PROBLEM (2)

- Sensor networks are rapidly growing for their large applicability to various purposes.
- From engineering perspectives, one of the most critical issues in sensor networks is energy, because:
  - sensor networks are often deployed in remote areas where line powers are hard to obtain. It forces the sensor nodes to rely on batteries, but replacing the batteries is also hard in many cases
  - sensor network applications often require long-term measurements over months.

4

### THE PROBLEM (3)

- So, improving the energy efficiency is mandatory for sensor networks.
- Wireless communication is one of the most energy-consuming operations on a sensor node



try to reduce communication

5

### THE PROBLEM (4)

A possible approach:

- Multi-hop communication to the base station.
- It can be inefficient depending on how densely the sensor nodes are deployed:
  - When the node deployment is very sparse, each hop distance becomes large and thus the large amount of energy is necessary for sending data over that distance.
  - When the node deployment is very dense, the nodes close to the base station need to forward the data from many remote nodes and thus tend to run out of energy soon.

6

### THE PROBLEM (5)

Another approach:

- Exploit the mobility.
- A **data mule** is a mobile node that has wireless communication capability and also a sufficient amount of storage to store the data from the (static) sensor nodes in the field. It can be used for data collection.
  - Data mule travels across the sensing field and collects data from each sensor node when the distance is short, and later deposits all the data to the base station.
  - Each sensor node can conserve a significant amount of energy, since it only needs to send the data over a shorter distance and has no need to forward other sensors' data all the way to the base station.
  - As data mules return to the base station after the travel, energy issue is usually not critical for data mules.

7

### THE PROBLEM (6)

**Data mule scheduling problem:**

- “how to control a data mule such that it collects data from all the nodes in the minimal amount of time?”
- We formulate it as a scheduling problem, since we view communication from each node as a job.
- We can control the movement of the data mule (path, speed) as well as its communication (i.e., which node it collects data from at certain time duration), where the latter corresponds to job allocation in classical scheduling problems.

8

## THE PROBLEM (7)

Data mule scheduling problem (cntd)

- Despite the similarities with a scheduling problem, data mule scheduling problem has both location and time constraints.
- Availability of each job is determined by the range of wireless communication, which primarily depends on the distance from a node and thus serves as a location constraint.
- On the other hand, by assuming the bandwidth of wireless communication is constant, we also have a time constraint for each node necessary for transmitting the data to a data mule.
- The movement of data mule determines how the location constraints map to time constraints and produces different real-time scheduling problems.

9

## THE PROBLEM (8)

Data mule scheduling problem (cntd)

Decompose the problem into the following three subproblems:

1. **Path selection**: which trajectory the data mule follows
2. **Speed control**: how the data mule changes the speed during the travel
3. **Job scheduling**: from which sensor the data mule collects data at each time point

10

## THE PROBLEM (9)

Data mule scheduling problem (cntd)

1. **Path selection** is to determine the trajectory of the data mule in the sensor field. To collect data from each particular sensor, the data mule needs to go within the sensor's communication range at least once.
2. **Speed control** is to determine how the data mule changes its speed along the chosen path. The data mule needs to change the speed so that it stays within each node's communication range long enough to collect all the data from it.
3. **Job scheduling**: data collection from each sensor is a job. Each job has one or more intervals in which it can be executed. Job scheduling is to determine the allocation of time slots to jobs so that all jobs can be completed.

11

## THE PROBLEM (10)

For what concerns the third subproblem, it can be reduced to classical scheduling problems

possible students' lesson

12

## THE PROBLEM (11)

We now focus on the first problem (path selection):

- We consider a sensor network that has sensor nodes in different areas operating at different sampling rates (e.g. in the case of pollution sensors).
- Each sensor has a finite buffer for storing the sensed values and a data mule (acting as a base station) does the job of the data gathering.
- Once the mobile element visits a sensor node, it transfers the data to its own memory and the sensor's memory is freed.
- ...

13

## THE PROBLEM (12)

- A problem that naturally crops up is the scheduling of the visits of the data mule so that none of the sensor nodes' buffer overflows: **Mobile Element Scheduling (MES)** problem.
- Observe the similitude with the **Traveling Salesman Problem (TSP)**:

Given a set of cities, a salesman has to visit each one of the cities starting from a certain one (e.g. the hometown) and returning to the same city. The challenge of the problem is that the traveling salesman wants to minimize the total length of the trip.

14

## THE PROBLEM (13)

- In fact, MES problem and TSP are different:
  - In TSP, the goal is to find a minimum cost tour that visits each node exactly once.
  - In MES problem a node may need to be visited multiple times before all other nodes are visited depending on the strictness of its deadline i.e. frequency of sampling.
  - In addition, as soon as a node is visited, its deadline i.e. time before which it should be revisited to avoid buffer overflow is updated. Thus deadlines are “dynamically” updated as the mobile element performs the job of data gathering.
- Nevertheless, TSP seems very useful to solve MES problem.

15

## THE TRAVELING SALESMAN PROBLEM



16

## THE TRAVELING SALESMAN PROBLEM (1)

**TSP** (decisional version):

- Let  $K_n=(V,E)$  be a complete graph,  $w$  a non negative edge-weight function, and  $t$  a non negative real value.

**Def.** A Hamiltonian cycle is a cycle passing through all nodes exactly once.

**Question:** Does  $K_n$  contains a Hamiltonian cycle with cost not exceeding  $t$ ?

The problem of finding a Hamiltonian cycle (HC) is NP-complete.

17

## THE TRAVELING SALESMAN PROBLEM (2)

- The origins of TSP are unclear:
- A handbook for travelling salesmen from 1832 mentions the problem and includes example tours through Germany and Switzerland, but contains no mathematical treatment.
- It was mathematically formulated in the 1800s by the mathematician W.R. Hamilton and T. Kirkman.
- The general form has been first studied during the 1930s considering the obvious brute-force algorithm, and observing the non-optimality of the nearest neighbor heuristic.

18

## THE TRAVELING SALESMAN PROBLEM (3)

- Th.** TSP is NP-complete.

**Proof.**

- TSP belongs to NP: to check a solution it is enough to check that the tour contains each node exactly once. Then we sum the cost of all the edges in the tour and check that it is bounded by  $t$ .

- We reduce HC to TSP:

Assume  $G=(V,E)$  is an instance of HC and construct the complete graph  $K_n=(V,E')$ . Let  $t=n$  and function  $w$  is defined as follows:

- $w(i,j)=1$  if  $(i,j)$  is in  $E$
- $w(i,j)=2$  if  $(i,j)$  is in  $E' \setminus E$

19

## THE TRAVELING SALESMAN PROBLEM (4)

NP-completeness proof (cntd).

- Assume now that a HC  $C$  exists in  $G$ .
- All edges in  $C$  have weight 1 since they are all in  $G$ .
- So, if  $G$  has a HC then  $G'$  has a TS tour of cost  $n$ .
- Conversely, if  $G$  has a TS tour of cost  $n$  then all the used edges must come from  $G$  and hence the tour is a HC for  $G$ . ■

20

## THE TRAVELING SALESMAN PROBLEM (5)

TSP can be formulated as an ILP:

- Assume that the tour is oriented.
- We define boolean variables  $x_{ij}=1$  iff the tour passes from node  $i$  to node  $j$ ,  $x_{ij}=0$  otherwise.
- Call  $w_{ij}$  the weight of oriented edge  $(i,j)$ .

The objective is:  $\min \sum_{i,j=1..n} w_{ij} x_{ij}$

subject to:  $\sum_{j=1..n} x_{ij}=1$ ,  $\sum_{i=1..n} x_{ij}=1$

and the subtour elimination constraints (guaranteeing that a cycle cover is not a solution):

$\sum_{i,j \text{ in } S} x_{ij} < |S|$  for each  $S$  proper subset of  $V$ ,  
indeed if  $S$  forms a cycle the sum is  $=|S|$ .

21

## INAPPROXIMABILITY (1)

The following negative result holds:

**Th.** If there exists a polynomial algorithm for TSP with any constant approximation ratio  $r>1$  then  $P=NP$ .

**Proof.** We prove that if TSP is  $r$ -approximable, then there exists a polynomial time exact algorithm for HC.

Let  $G=(V,E)$  be an instance of HC and  $K_n$  a complete graph with  $|V|=n$ .

...

22

## INAPPROXIMABILITY (2)

If there exists a polynomial algorithm for TSP with any approximation ratio  $r>1$  then  $P=NP$ . (proof cntd)

...

We define edge weights on  $K_n$  as follows:

$w(i,j)=1$  if  $(i,j)$  is an edge of  $E$

$w(i,j)=2+(r-1)n$  otherwise.

Then, a tour with cost  $n$  exists in  $K_n$  if and only if  $G$  has a HC.

In this case, if we assume there exists an  $r$ -approximation algorithm  $\mathcal{A}$  for TSP, if the cost is  $n$ ,  $\mathcal{A}$  will find a solution  $H$  with  $\text{cost}(H) \leq rn$ .

23

## INAPPROXIMABILITY (3)

If there exists a polynomial algorithm for TSP with any approximation ratio  $r>1$  then  $P=NP$ . (proof cntd)

...

If  $H$  contains an edge that is not in  $E$ , then:

$\text{cost}(H) \geq (n-1) + 2 + (r-1)n = rn + 1$

That is a contradiction.

Hence, to a solution of TSP with  $\text{cost}(H) \leq rn$  corresponds an HC in  $G$ .

...

24

## APPROXIMATE ALGORITHMS (1)

Inapproximability result: Bad news!  
How to manage the problem?

Some special cases...

25

## APPROXIMATE ALGORITHMS (2)

**Def.** Given any three nodes  $a, b, c$  if:

$$w(a,c) \leq w(a,b) + w(b,c)$$

then we say that  $w$  satisfies the triangle inequality.

- o **Note.** The weight of a minimum spanning tree  $T$  (MST) is a lower bound on the cost of an optimal traveling salesman tour. Indeed: let  $H^*$  be an optimal tour. A ST  $P$  can be deduced from  $H^*$  by deleting an edge. Moreover,  $P$  is a path. It holds that:

$$w(T) \leq w(P) \leq w(H^*).$$

26

## APPROXIMATE ALGORITHMS (3)

In the hypothesis of triangular inequality (metric TSP), using a minimum spanning tree we will create a tour the cost of which is at most 2 times the weight of the spanning tree.

**2-Approx-mTSP**

Input:  $K_n(V,E)$

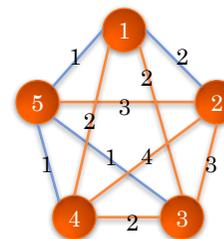
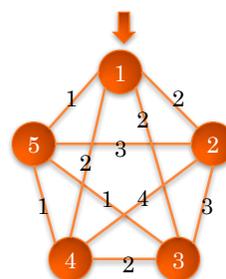
Output: a Hamiltonian cycle

1. Select a "root" node  $r$  of  $K_n$
2. Compute a MST  $T$  from  $r$
3. let  $L$  be the sequence of nodes visited in a preorder walk of  $T$
4. Return the HC that visits the nodes in the order  $L$  without repetitions

27

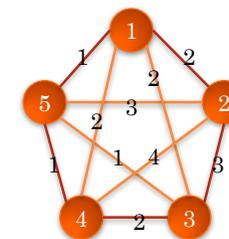
## APPROXIMATE ALGORITHMS (4)

example



Weight=5  
L=1 5 4 5 3 5 1 2 1

1. Select a "root" node  $r$  of  $K_n$
2. Compute a MST  $T$  from  $r$
3. let  $L$  be the sequence of nodes visited in a preorder walk of  $T$
4. Return the HC that visits the nodes in the order  $L$  without repetitions



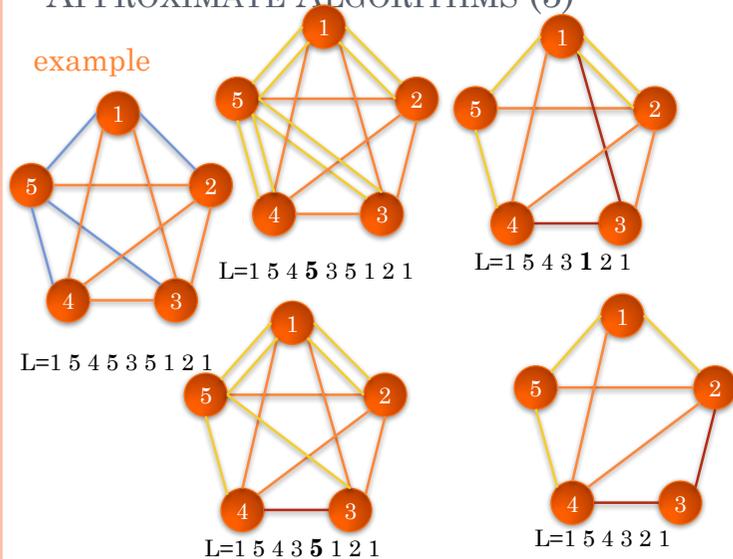
Weight=9

to pass from  $T$  (or from  $L$ ) to a cycle...

28

## APPROXIMATE ALGORITHMS (5)

example



29

## APPROXIMATE ALGORITHMS (6)

**Th.** *2-Approx-TSP* is a 2-approximation algorithm for TSP if  $w$  satisfies the triangle inequality.

**Proof.** Let  $H^*$  be an optimal tour and  $T$  a MST. We know that  $w(T) \leq w(H^*)$ .

In  $L$  every edge appears exactly twice, so the tour  $C$  deduced by  $L$  is such that  $w(C) = 2 w(T)$ .

Unfortunately,  $C$  is not a tour, since some nodes are repeated. We can erase some visits without increasing the cost: if a node  $a$  appears for the second time in the full path between  $b$  and  $c$ , we can go from  $b$  to  $c$  directly. In this way we get a tour  $H$ . By the triangle inequality:

$$w(H) \leq w(C) = 2 w(T) \leq 2 w(H^*).$$

30

## APPROXIMATE ALGORITHMS (7)

### Euclidean TSP

- When the cities that the salesman has to visit lie in the Euclidean plane, the problem is called Euclidean TSP.
- Like the general TSP, Euclidean TSP is NP-hard in either case.
- Euclidean TSP is a particular case of metric TSP, since distances in a plane obey the triangle inequality  $\rightarrow$  2-approximation algorithm.
- Christofides [C76] improves the previous approximation to  $3/2$  by exploiting a max matching when passing from the spanning tree to the cycle
- Can we do better?

31

## APPROXIMATE ALGORITHMS (8)

### Euclidean TSP(cntd)

- Yes: in general, there is a polynomial-time approximation scheme (PTAS) i.e. for any  $c > 0$ , if  $d$  is the number of dimensions in the Euclidean space, there is a polynomial-time algorithm that finds a tour of length at most  $(1 + 1/c)$  times the optimal for geometric instances of TSP in time

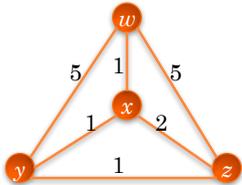
$$O\left(n(\log n)^{O(c\sqrt{d})^{d-1}}\right),$$

- Sanjeev Arora and Joseph S. B. Mitchell were awarded the Gödel Prize in 2010 for their concurrent discovery of a PTAS for the Euclidean TSP.
- In 2013, Bartal and Gottlieb improved the time complexity of the PTAS
- But in practice, simpler heuristics with weaker guarantees continue to be used...

32

## SOME GENERALIZATIONS (1)

- We can drop the condition that the travelling salesman should visit each city exactly once, so that we now consider not Hamiltonian cycles anymore, but simply closed walks containing each node at least once.
- If the problem is metric, any optimal tour will be also an optimal solution, but this is not true in general.



$w-x-y-z-x-w$  is a shortest closed walk (of length 6), but the shortest tour  $w-x-y-z-w$  has length 8.

33

## SOME GENERALIZATIONS (2)

### Asymmetric TSP

- Instead of  $K_n$ , we consider the complete directed graph  $K'_n$  on  $n$  nodes.
- This problem contains the usual TSP as a special case, and hence it is likewise NP-hard (indeed, notice that symmetry simply halves the number of possible solutions).

34

## SOME GENERALIZATIONS (3)

- We may also consider an arbitrary connected graph  $G$  with some length function  $w$  instead of  $K_n$ .
- In this case, it is not at all clear whether any tours exist: we need to check first whether  $G$  is Hamiltonian. As we know, this feasibility question is already an NP-complete problem in itself.

35

## A CURIOSITY

- The TSP, in particular the Euclidean variant of the problem, has attracted the attention of researchers in cognitive psychology.
- It has been observed that humans are able to produce good quality solutions quickly [Macgregor, Ormerod '96].
- These results suggest that computer performance on the TSP may be improved by understanding and emulating the methods used by humans for these problems, and have also led to new insights into the mechanisms of human thought.

36