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

1

Prof. Tiziana Calamoneri
Network Algorithms
A.y. 2021/22

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.

THE PROBLEM (2)

- Sensor networks are dense wireless networks of small, low-cost sensors, which collect and disseminate environmental data.
- Sensor networks are rapidly growing for their large applicability to various purposes.

THE PROBLEM (3)

- 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.

THE PROBLEM (4)

- 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

THE PROBLEM (5)

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.

THE PROBLEM (6)

Another approach:

Exploit the mobility:

- A data mule is a mobile node that has:
  - wireless communication capabilities
  - 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 arrives at short distance from it; later it deposits all the data to the base station.

THE PROBLEM (7)

- 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 to the base station.
- As data mules return to the base station after the travel, energy issue is usually not critical for data mules.

THE PROBLEM (8)

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.

THE PROBLEM (9)

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.

THE PROBLEM (10)

Data mule scheduling problem (cntd)

- 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.

THE PROBLEM (11)

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

THE PROBLEM (12)

Data mule scheduling problem (cntd)

More in detail:

1. Path selection is to determine the trajectory of the data mule in the sensor field. To collect data from each sensor, the data mule needs to go within the sensor's communication range at least once.
2. …

THE PROBLEM (13)

Data mule scheduling problem (cntd)

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.
   More technical than algorithmic.
3. …

THE PROBLEM (14)

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.
   It can be reduced to classical scheduling problems
   → possible students' lesson

THE PROBLEM (15)

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

- Consider a sensor network that has sensor nodes operating at different sampling rates (*e.g.* in 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.
- …

THE PROBLEM (16)

- Problem: 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.

THE PROBLEM (17)

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 the other nodes are visited depending on the strictness of its deadline *i.e.* frequency of sampling.
- …

THE PROBLEM (18)

In fact, MES problem and TSP are different (cntd):

- In TSP the optimization function minimizes the cost of the the tour and costs are fixed in the time.
- In MES problem, 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.

THE TRAVELING SALESMAN PROBLEM

21

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.

22

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 mathematicians 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.

23

THE TRAVELING SALESMAN PROBLEM (3)

- **Th.** TSP is NP-complete.
- Proof.

1. 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.
2. 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 $E' \setminus E$

24

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 in K_n since they are all in G.
- So, if G has a HC then K_n has a TS tour of cost n.
- Conversely, if K_n 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. ■

THE TRAVELING SALESMAN PROBLEM (5)

TSP can be formulated as an ILP [Dantzig, Fulkerson, Johnson '54]:

- Assume that the tour is oriented.
- We define:
  - boolean variables $x_{ij}=1$ iff the tour traverses oriented edge (i,j), $x_{ij}=0$ otherwise.
  - w_{ij} the weight of oriented edge (i,j).
- …

THE TRAVELING SALESMAN PROBLEM (6)

…

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

subject to: $\sum_{j=1..n} x_{ij}=1$, $\sum_{i=1..n} x_{ij}=1$ (only one arc from i and to j)

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 $=|S|$.

INAPPROXIMABILITY (1)

The following negative result holds:

Th. If there exists a polyomial time algorithm for TSP with any 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 $|V|=n$.

…

INAPPROXIMABILITY (2)

If there exists a polyomial 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 cost(H) ≤ $r\,n$.

…

INAPPROXIMABILITY (3)

If there exists a polyomial 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:

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

That is a contradiction.

Hence, to a solution of TSP with cost(H) ≤ $r\,n$ corresponds an HC in G. ■

APPROXIMATE ALGORITHMS (1)

Inapproximability result: Bad news!

How to manage the problem?

Some special cases…

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.

- Note. The weigth 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:

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

APPROXIMATE ALGORITHMS (3)

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

2-Approx-mTSP

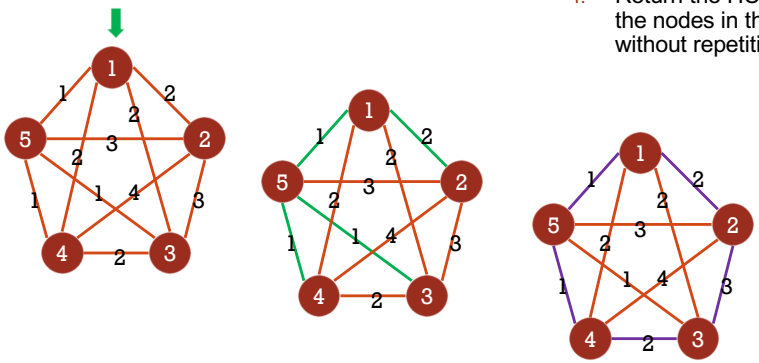
Input: $K_n(V,E)$
Output: a Hamiltonian cycle

1. Select a "root" node r of K_n
2. Compute an 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

APPROXIMATE ALGORITHMS (4)

1. Select a "root" node r of K_n
2. Compute an 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

example



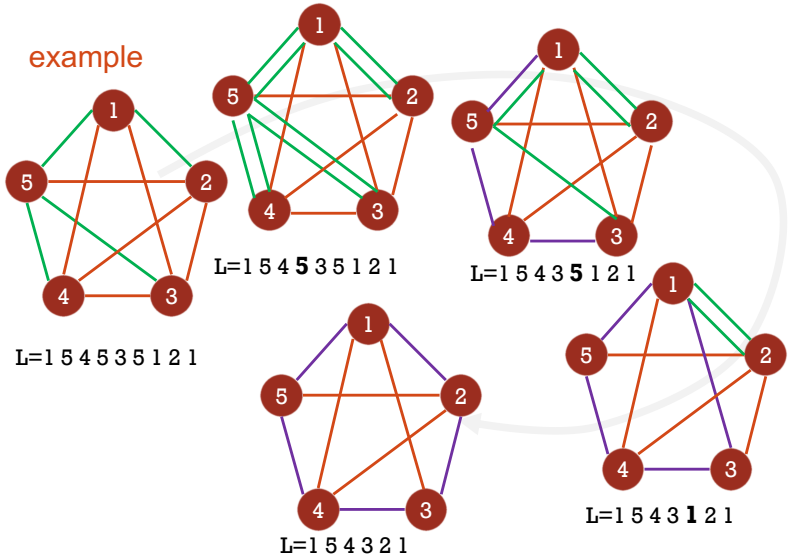
Weight=5
L=1 5 4 5 3 5 1 2 1

Weight=9

to pass from T (or from L) to a cycle…

APPROXIMATE ALGORITHMS (5)

example



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 $w(C)=2\,w(T)$.

…

APPROXIMATE ALGORITHMS (7)

2-Approx-TSP is a 2-approximation algorithm for TSP if *w* satisfies the triangle inequality (proof cntd)

Unfortunately, *C* is not a tour, since some nodes are repetead. 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:

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

APPROXIMATE ALGORITHMS (8)

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.
- Euclidean TSP is a special case of metric TSP, since distances in a plane obey the triangle inequality

  → 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.

APPROXIMATE ALGORITHMS (9)

Euclidean TSP(cntd)

Christofides Algorithm:

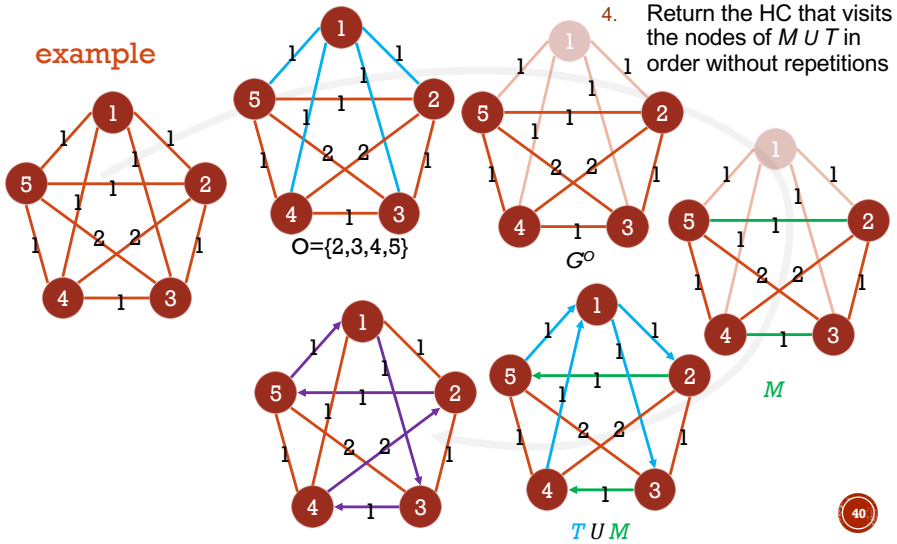
Input: $K_n(V,E)$

Output: a Hamiltonian cycle

- Compute an MST *T* of *G*.
- *O* = set of nodes with odd degree in *T*. |*O*| is even.
- G^O = subgraph induced by *O*.
- Find a min-weight perfect matching *M* in G^O.
- The graph induced by $M \cup T$ is a connected multigraph and each node has even degree.
- Form an Eulerian circuit.
- Return the HC that visits the nodes without repetitions.

APPROXIMATE ALGORITHMS (10)

1. Compute a MST *T*
2. Compute G^O
3. Find a min-weight PM *M*
4. Return the HC that visits the nodes of $M \cup T$ in order without repetitions



APPROXIMATE ALGORITHMS (11)

Euclidean TSP(cntd)

Th. *Christofides Algorithm* is a 3/2-approximation algorithm for TSP if *w* satisfies the triangle inequality.

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

Number the nodes of G^O in cyclic order around H^*, and partition H^* into two sets of edges: the ones with an odd number on the first node and the ones with an even number on the first node. Each set is a perfect matching: $M_o \cup M_e \subseteq H^*$.

…

APPROXIMATE ALGORITHMS (12)

Euclidean TSP: *Christofides Algorithm* is a 3/2-approximation algorithm for TSP if *w* satisfies the triangle inequality (proof cntd)

…

So, $w(M_o)+w(M_e) \leq w(H^*)$; wlog $w(M_o) \leq w(H^*)/2$.

M is a min-weight perfect matching of G^O so $w(M) \leq w(M_o)$.

Putting everything together:

$$w(H) \leq w(M)+w(T) \leq w(H^*)/2+w(H^*) \leq 3/2\ w(H^*).$$ ■

APPROXIMATE ALGORITHMS (13)

Euclidean TSP(cntd)

- There exist inputs to the travelling salesman problem that cause the Christofides algorithm to find a solution whose approximation ratio is arbitrarily close to 3/2.
- Can we do better?

APPROXIMATE ALGORITHMS (14)

Euclidean TSP(cntd)

- Yes: in general, there is a polynomial time approx scheme (PTAS) *i.e.* for any $c > 0$, if d is the dimension of 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

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

- Arora and Mitchell were awarded with the Gödel Prize in 2010 for their concurrent discovery of a PTAS for the Euclidean TSP.
- …

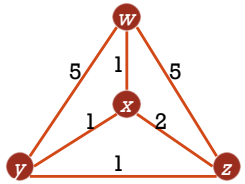
APPROXIMATE ALGORITHMS (15)

Euclidean TSP(cntd)

- …
- In 2013, Bartal and Gottlieb improved the time complexity of the PTAS
- But in practice, simpler heuristics with weaker guarantees continue to be used…

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:



Example:
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.

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).

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.

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.