Memory Efficient Exploration in Anonymous Networks

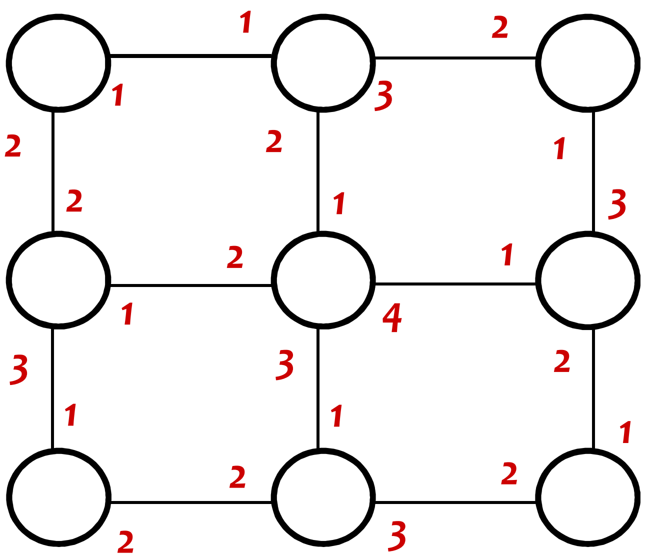
Leszek A. Gąsieniec

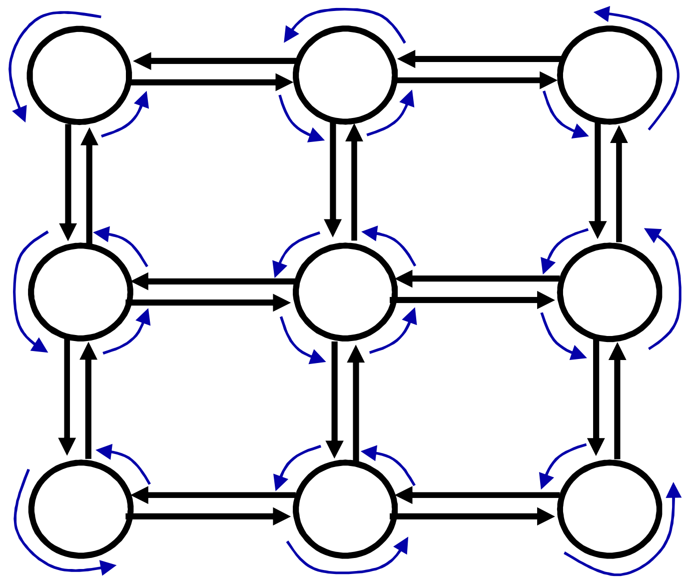
U of Liverpool

Special thanks go to: Evangelos Bampas, Petra Berenbrink, Andrew Collins, Jurek Czyzowicz, Stefan Dobrev, Robert Elsässer, Pierre Fraigniaud, Nicholas Hanusse, David Ilcinkas, Jesper Jansson, Ralf Klasing, Adrian Kosowski, Darek Kowalski, Arnaud Labourel, Gadi Landau, Yannis Lignos, Russell Martin, Alfredo Navarra, Andrzej Pelc, David Peleg, Tomasz Radzik, Kunihiko Sadakane, Wing-Kin Sung, and Xiaohui Zhang (among the others).

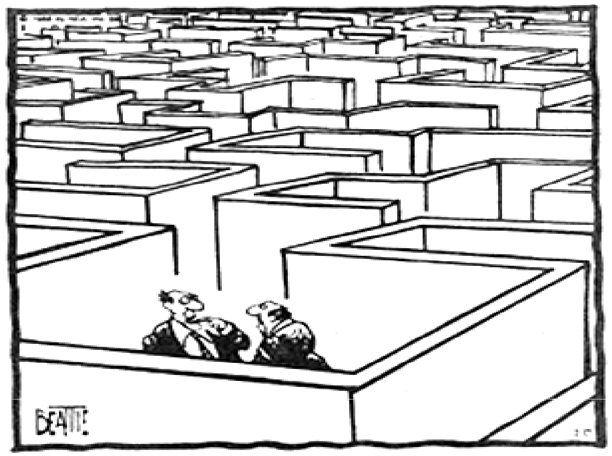
Anonymous Networks – labelled vs. implicit ports

- Equivalent definitions of anonymous graphs
  - with ***explicit*** and
  - ***implicit*** port ordering





Network/graph traversal problem



"The exit? Sure...take a right, then left, left again...no, wait...a *right*, then...no, wait...."

As *efficiently* as possible, typical complexity measures:

- ***memory utilization***,
- ***exploration time***,
- *use of* ***other resources*** *(markers, pebbles, colors, etc)*.

The random walk procedure

- The ***random walk***, is a mathematical formalization of a trajectory that consists of taking ***successive steps*** in *random directions*.
- A fundamental model for a ***random process*** in time. E.g., the following processes can be modeled as random walk
  - path traced by a molecule in a liquid or a gas (***Brownian motion***),
  - search path of a foraging animal,
  - price of a fluctuating stock and
  - financial status of a gambler, …
- A random walk on a graph is also a special case of a ***Markov chain***

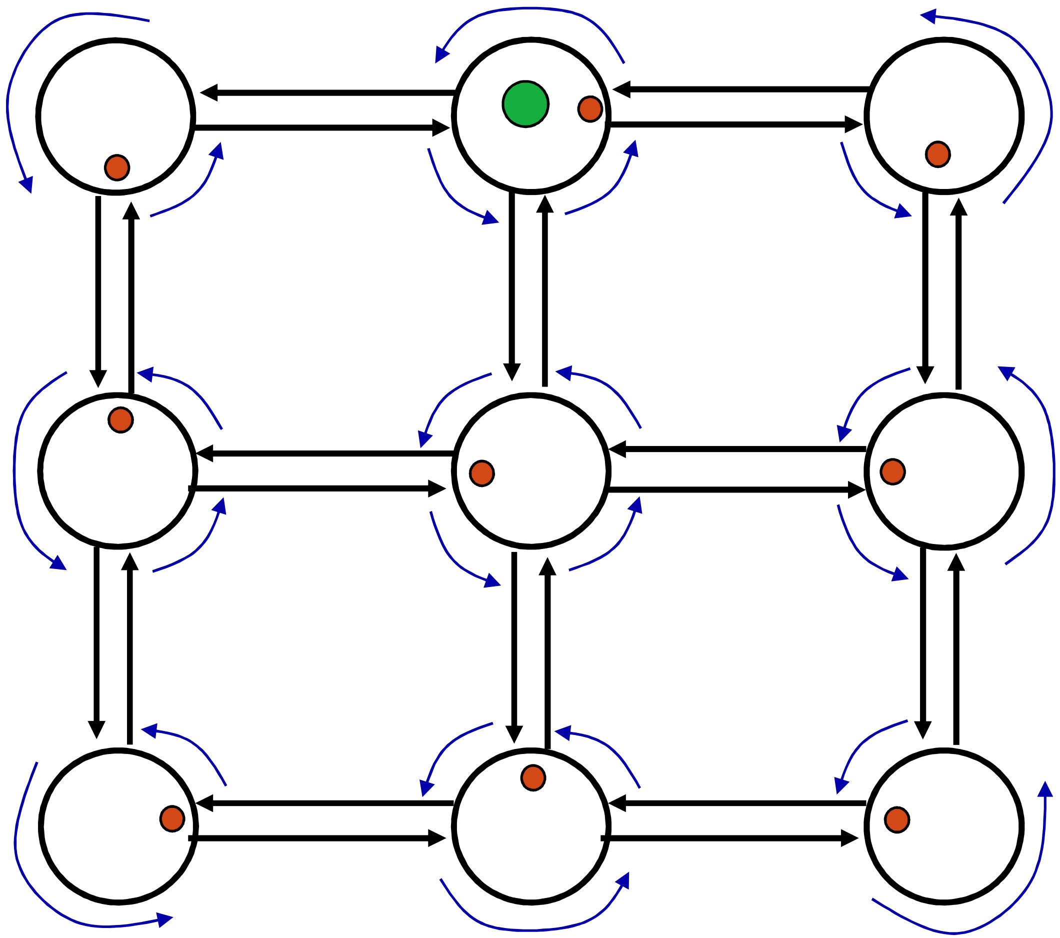
Basic results on the random walk

- Robot performing a random walk in an arbitrary graph of size ***n*** visits ***all nodes*** in the graph in (expected) time $O(n^3)$
    - R. Aleliunas, R.M. Karp, R.J. Lipton, L. Lovasz, and C. Rackoff, *FOCS'79*
- Robot performing a random walk in expected time:
    - complete graphs $O(n \log n)$
    - lines, trees $O(n^2)$
    - torus, 2D-grids $O(n \log^2 n)$
        - *(this can be improved to $O(n \log n)$ if **n** is known)*
- Robot performing a random walk in an arbitrary graph of size ***n*** visits ***all nodes*** in the graph in (expected) time $O(n^2 \log n)$ if we give preference to neighbours with lower degrees
    - S. Ikeda, I. Kubo, N. Okumoto, and M. Yamashita, *ICALP'03*

Deterministic counterparts for RW

- Traversal based on the random walk is virtually ***memory-less***, however it requires a ***large volume of (pseudo) random bits***
- There has been already a substantial attempt to study ***deterministic alternatives*** to the random walk
- Several models have been proposed and studied including:
  - ***the rotor-router mechanism*** and
  - ***the basic walk procedure***
- However, only a few results are known and further studies in the field would be highly appreciated

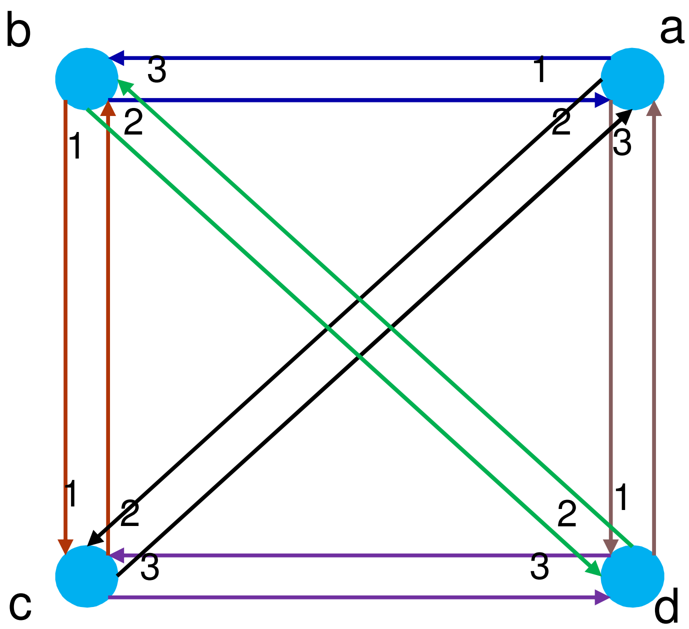
Rotor-router mechanism

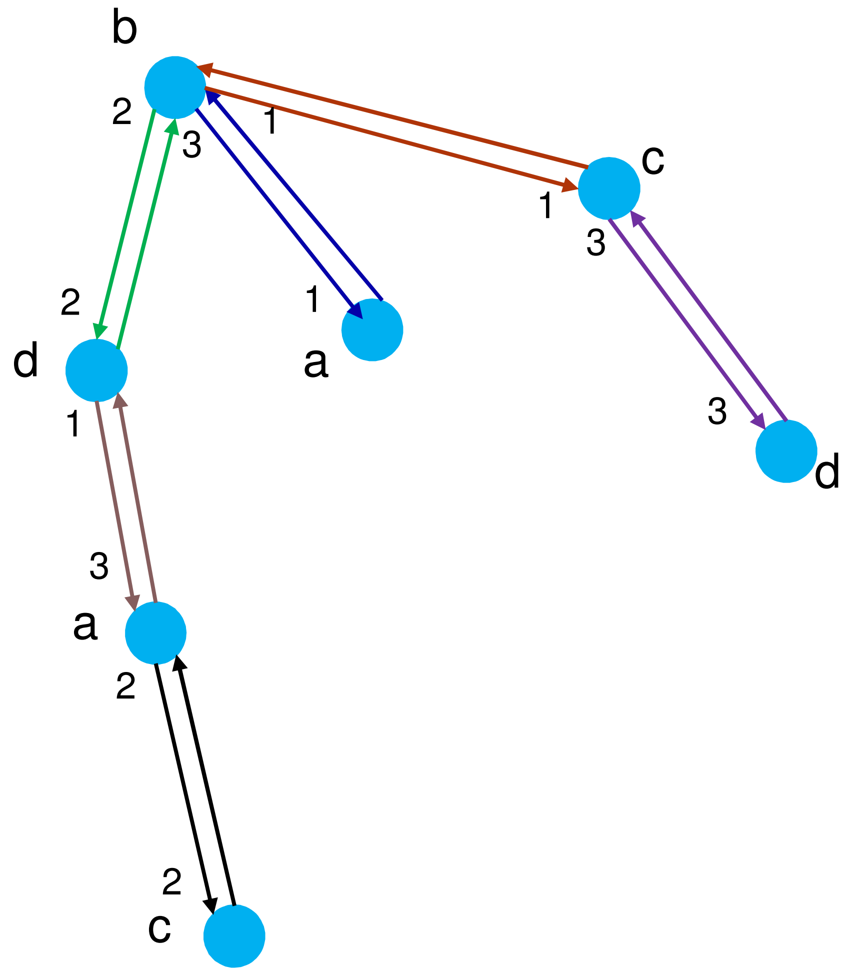


Traversal in rotor-router mechanism

- ***Robot locks*** in an Eulerian cycle in $O(V \cdot E)$ steps
  - S. Bhatt, S. Even, D. Greenberg, and R. Tayar,
    J. of Graph Algorithms and Applications'02
- ***Robot locks*** in an Eulerian cycle in $2 \cdot E \cdot D$ steps
  - V. Yanovski, I.A. Wagner, and A.M. Bruckstein,
    Algorithmica'03
- There is more and more work on comparison of performance of random walk and rotor-routers, e.g., in load balancing mechanism, infinite graphs, etc

Rotor-router model – Euler cycle





111$\boxed{\text{21223231133}}$21223231...

abc$\boxed{\text{bdacadbabcdc}}$bdacadba...

Traversal in rotor-router mechanism

- Dependence of the lock-in time on the initial configuration of the rotor-router mechanism
  - Bampas, Gąsieniec, Hanusse, Ilcinkas, Klasing, and Kosowski, ***DISC'09***
- Min and max values of the lock-in time in considered cases

Scenario	Worst case	Best case
P-all	$\Theta(m)$	$\Theta(m)$
A(↻)P(℗)	$\Theta(m)$	$\Theta(m)$
P(℗)A(↻)	$\Theta(m \cdot \min\{\log m, D\})$	$\Theta(m)$
A(℗)P(↻)	$\Theta(m \cdot D)$	$\Theta(m)$
P(↻)A(℗)	$\Theta(m \cdot D)$	$\Theta(m)$ *for all* $D \leq n^{1/2}$
A-all	$\Theta(m \cdot D)$	$\Theta(m \cdot D)$

Traversal in rotor-router mechanism

- We show that after establishing an ***Eulerian cycle***

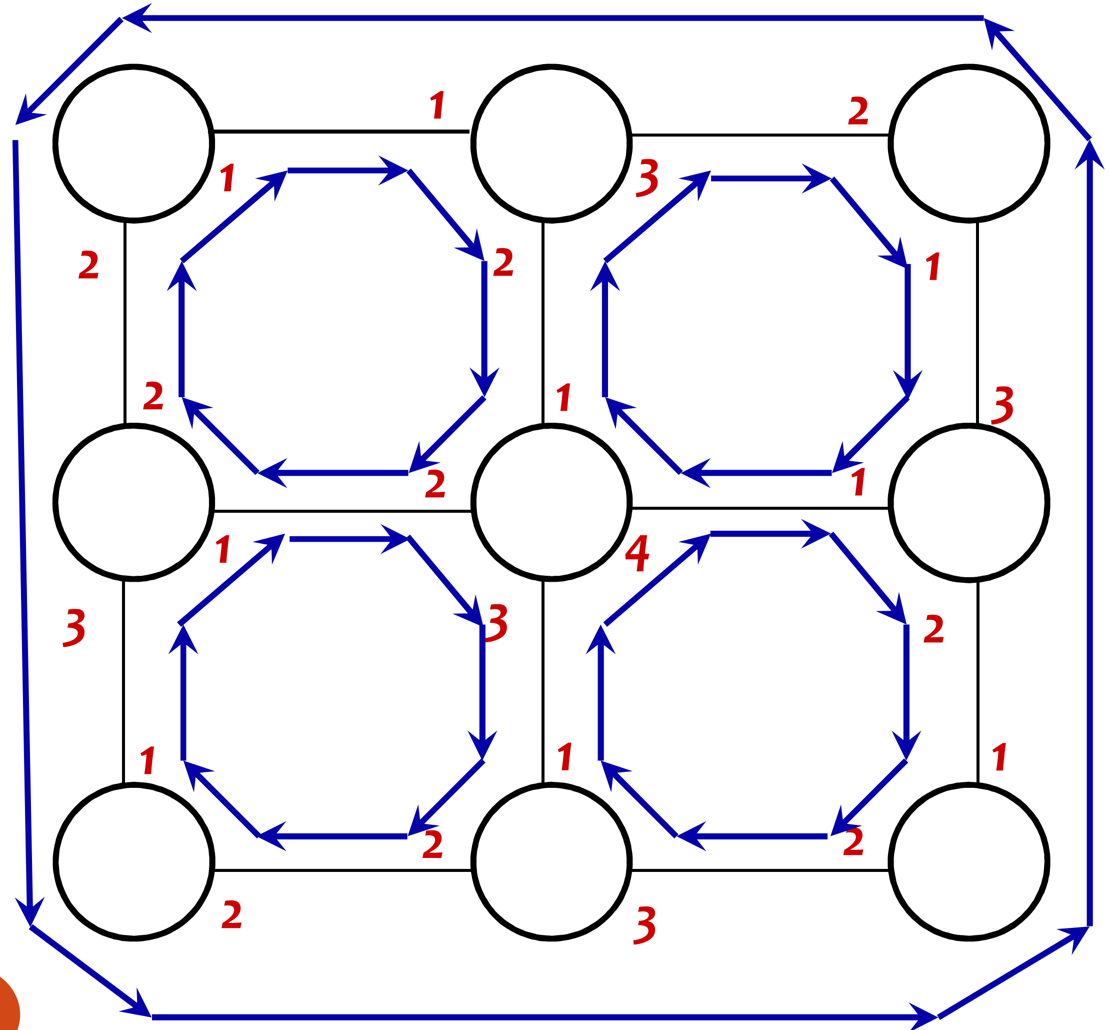
  Bampas, Gąsieniec, Klasing, Kosowski, and Radzik, ***OPODIS'09.***

  - (i) if at some step the values of ***k*** pointers v are arbitrarily changed, then a ***new Eulerian cycle*** is obtained within $O(k \cdot m)$ steps;
  - (ii) if at some step ***k*** edges are added to the graph, then a ***new Eulerian cycle*** is established within $O(k \cdot m)$ steps;
  - (iii) if at some step an edge is deleted from the graph, then a ***new Eulerian cycle*** is established within $O(\gamma \cdot m)$ steps, where γ is the number of edges in a shortest cycle in graph G containing the deleted edge.
- The results are based on the relationship between Eulerian cycles and spanning trees known as the "BEST" Theorem (due to de Bruijn, van Aardenne-Ehrenfest, Smith and Tutte)

Basic walk

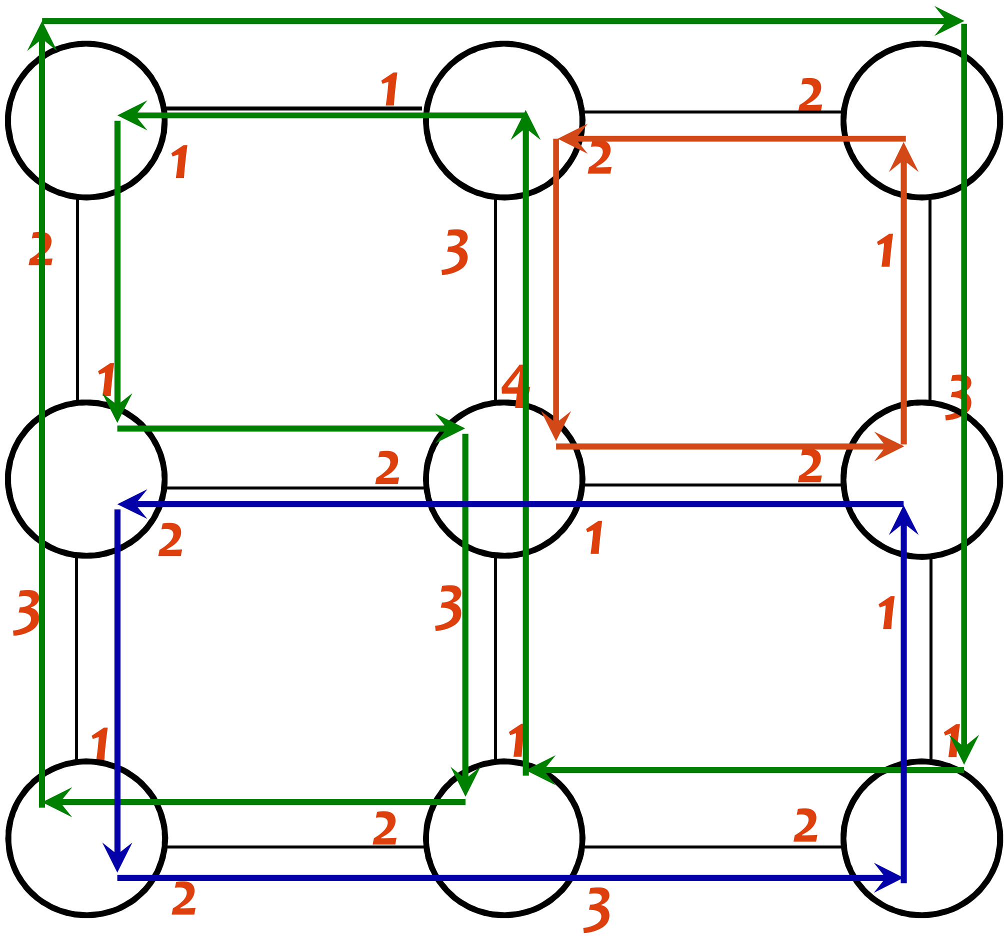
- This type of an algorithm can be used in case when the robot is barely equipped in the internal memory, i.e., the use of ***none*** or a ***constant number*** of memory bits is allowed.
- Simple actions of the robot are ***pre-programmed*** and could be seen as actions of a finite state machine, also ***the ports in the graph are pre-processed***.
- The task is to design a route based on port numbers and navigation abilities of the finite state machine that allows the robot to ***visit all graph nodes*** periodically.

Basic walk – cover by directed cycles



- The basic walk idea and an ***arbitrary arrangements of port numbers*** partitions all unidirectional edges (obtained from replacing each undirected edge by a pair of arcs with the opposite directions) into a ***number of directed cycles***

Basic walk - a tour



- In this model periodic graph exploration refers to ***arrangement of ports***, s.t., at least ***one tour containing all nodes*** in the graph is formed.

Comment: what about random ordering of ports? *It seems that the expected length of a cycle is ≈ 79.*

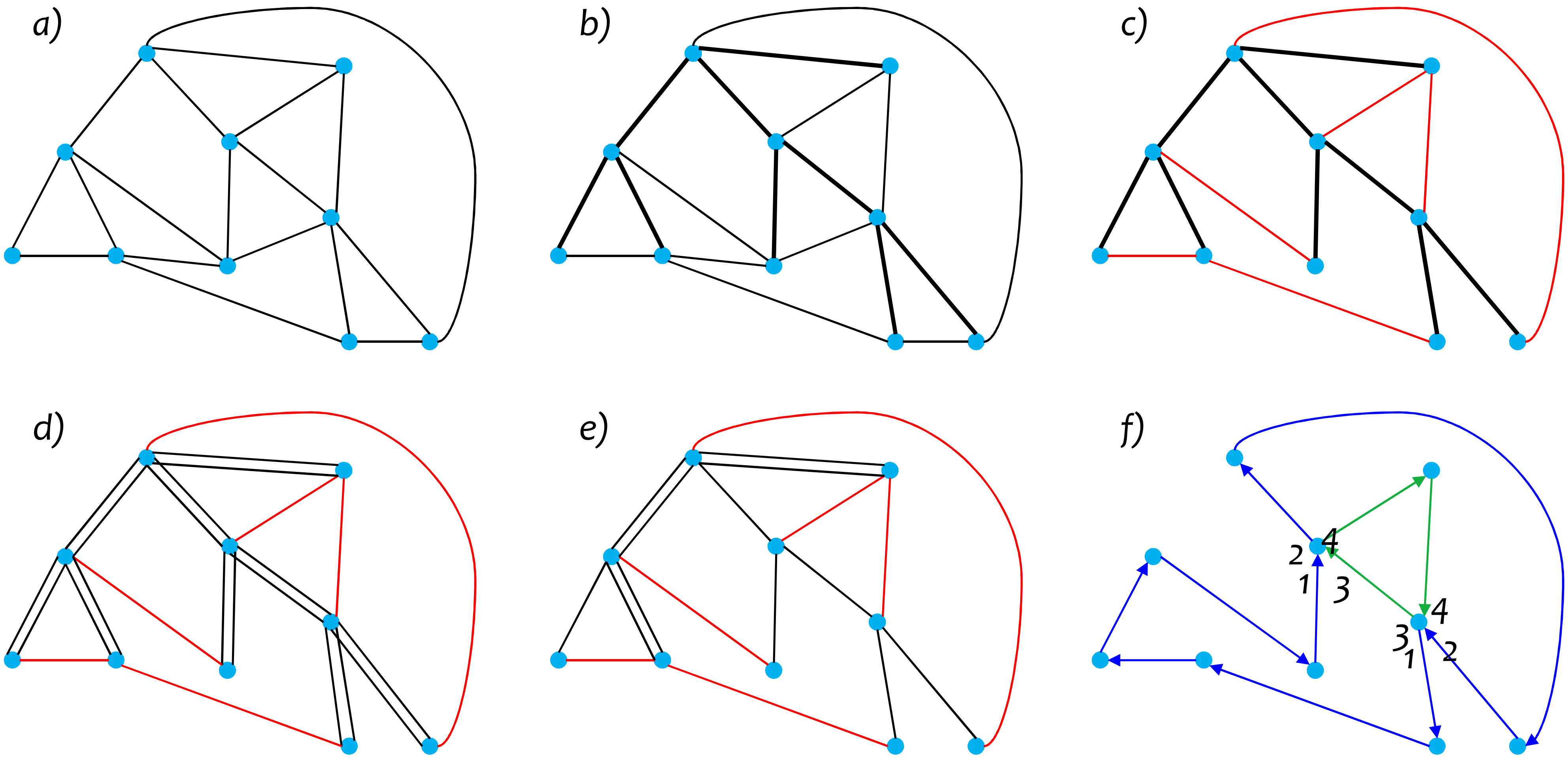
Oblivious robots, tour < 2n

In graphs having a spanning tree with non-saturated nodes

An input graph G

Find a spanning tree

Pick single edges



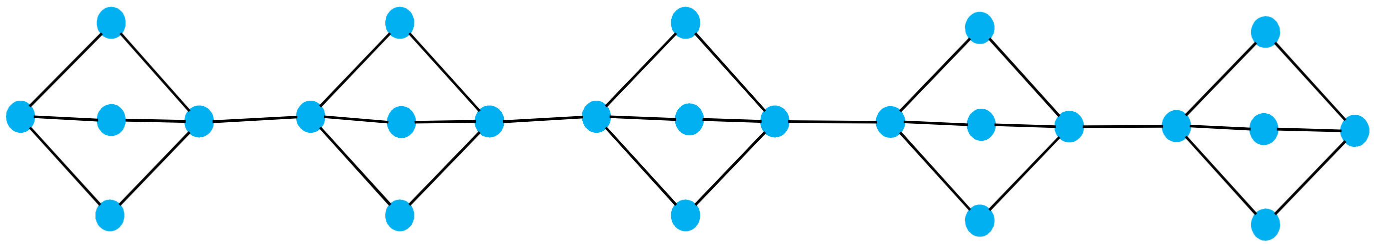
Double tree edges

Restore parity at nodes and remove double edges

One cycle of length < 2n

Oblivious robots, summary

- ***Searching for spanning trees*** with external graph edges at each node of the tree is ***NP-hard***. This problem is equivalent to finding a ***Hamiltonian cycle*** in ***cubic graphs*** (known to be NP-hard).
- ***Not every graph*** have a spanning tree with the ***desired property***, thus in general a different approach is needed.
- The best currently known ***bounds*** on the length of the periodic route used by oblivious robots are:
  - Upper ***4n***
  - Lower ***2.8n***

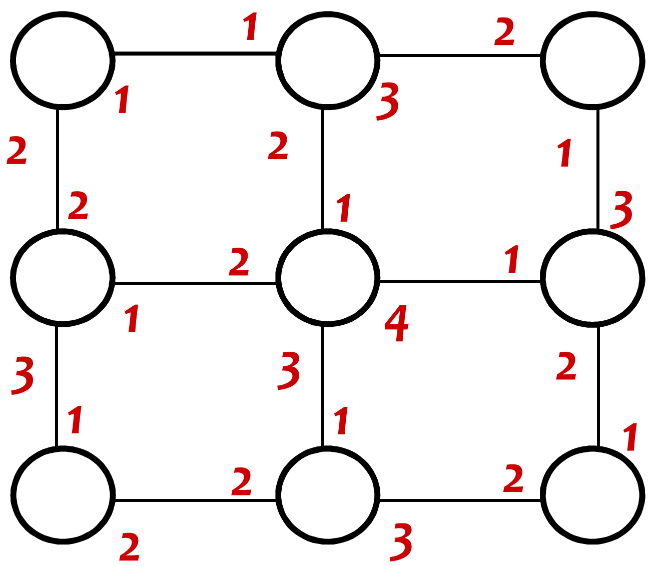


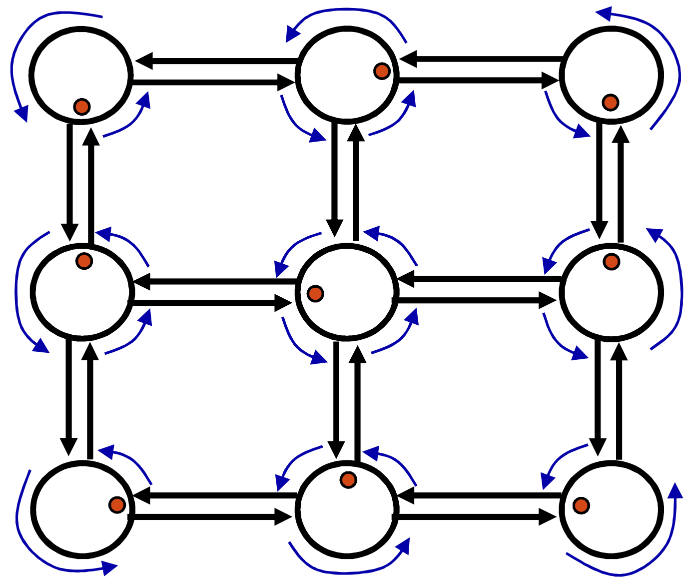
In this graph all edges must be traversed in two directions

Does extra memory help?

- In the model with implicit port numbers one needs to insert a ***fixed marker*** at one port of each node of the network.
- This breaks symmetry at the node and allows to use efficiently the memory provided to a robot

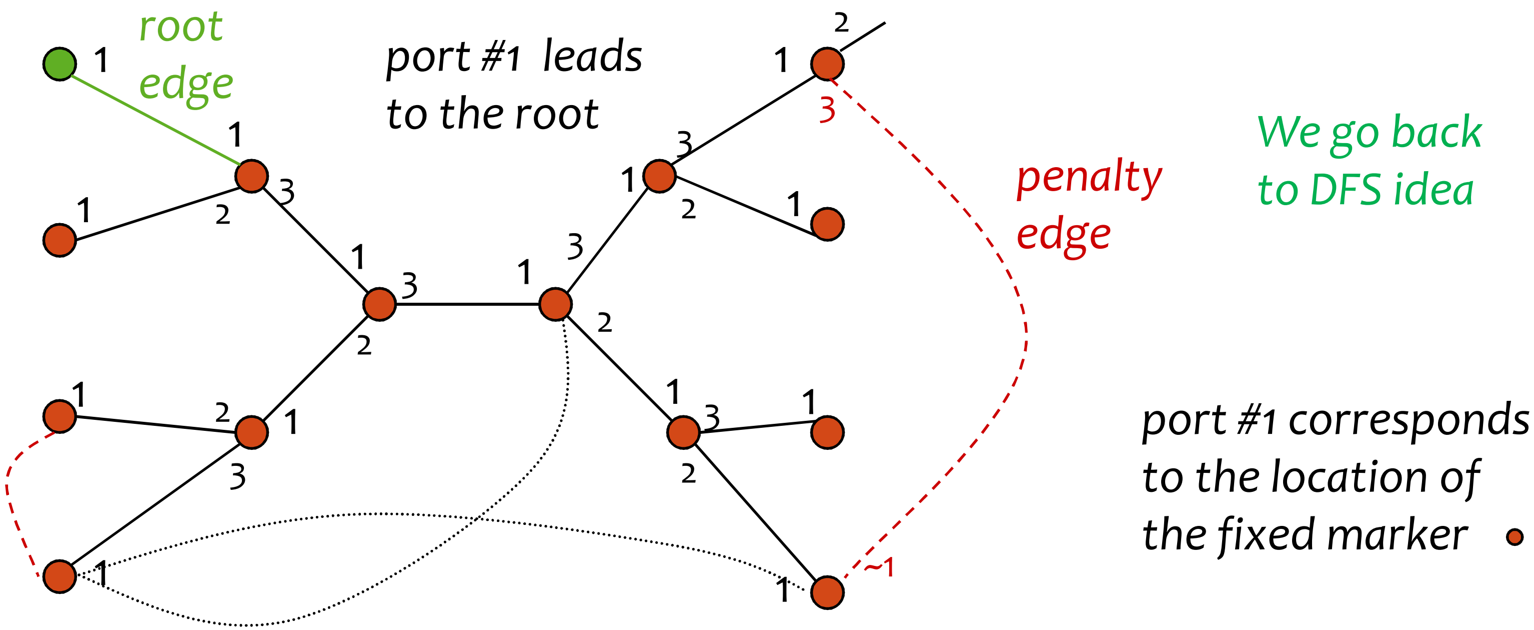
• *fixed marker*





Memory utilisation

- The exploration is performed along edges of a spanning tree encoded by port numbers.



Every node potentially carries ***a penalty edge***, thus the length of the tour is ≤ **4n-2**, where **2n-2** comes the spanning tree and **2n** from penalty edges. We know how to avoid at least n/4 penalty edges. This gives a tour of length at most ***3½n***.

Results in the basic walk model

- ***state-less*** graph exploration with the tour of length ***10n***
  - S. Dobrev, J. Jansson, K. Sadakane, W.-K. Sung, ***SIROCCO'05***
- ***2 bit-state*** exploration with the tour of length ***4n-2***; also *conjectured* lower bound of ***4n-O(1).***
  - D. Ilcinkas, ***SIROCCO'06***
- ***constant bit-state*** exploration with the tour of length ***3.75n-2.***
  - L. Gąsieniec, R. Klasing, R. Martin, A. Navarra, X. Zhang, ***SIROCCO'07***
- ***state-less*** exploration with the tour of length ***4.3(3)n*** and

  constant bit-state exploration with the tour of length ***3.50n-2.***
  - J. Czyzowicz, S. Dobrev, L. Gąsieniec, D. Ilcinkas, J. Jansson, R. Klasing, Y. Lignos, R. Martin, K. Sadakane, W.-K. Sung, ***SIROCCO'09***
- ***state-less*** exploration with the tour of length ***4n*** and
  - A. Kosowski and A. Navarra, ***MFCS'09***.

Other related problems

- Rendezvous problems
- Asynchronous computation/communication

Summary and further work

- *Model with preprocessed port numbers*
  - *(basic walk) oblivious robots* **2.8n … 4n**
  - *(basic walk) robots with constant memory* **2n … 3.5n**
- *(Model with the worst case port numbers*
  - *(rotor router) exact bounds on stabilization in various graph classes*
  - *(random walk vs. rotor router) exploration similarities/differences*
- *Model with random port numbers*
  - *(rotor router) performance in different classes of graphs*
  - *(random walk) performance in different classes of graphs*
  - *(basic walk) distribution of cycles, how many possible tours?*
  - *study of hybrid models, e.g., random rotor-router*
- *Multi-robot problems*
  - *Graph exploration, rendezvous and gathering, asynchronous agents, etc*