Distributed Tasks for *Energy-Constrained* Mobile Robots

Shantanu Das

Aix-Marseille University, France

(*Joint work with*:
Jeremie Chalopin, Dariusz Dereniowski, Matus Mihalak, Christina Karousatou, Paolo Penna, Peter Widmayer)

MAC-GRASTA 2015 (Montreal)

Large Teams of Small Robots

Limitations of Robots ⟷ Distributed Tasks

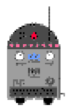
Small and inexpensive robots

- Limited Memory
- Limited Visibility
- No identifiers
- Inability to communicate
- Inability to measure (accurately)
- Not possible to leave marks

Are we forgetting something?

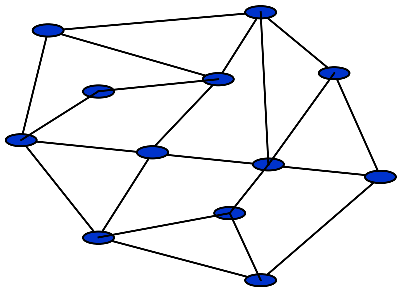

Battery low
Demotivation.us

Moving & Computing consumes Energy



Limited Energy

Distributed Tasks



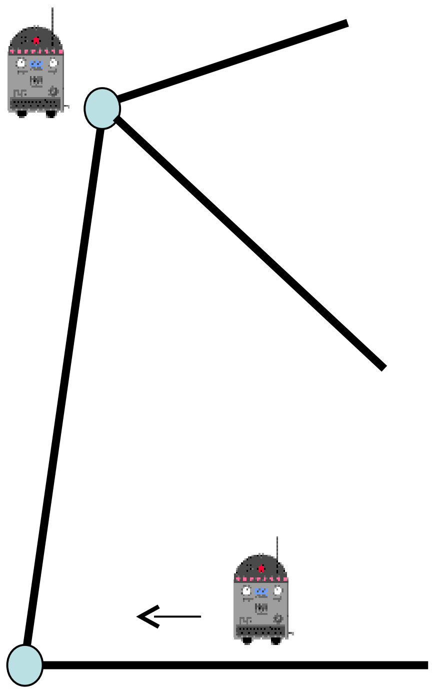
- Moving consumes more energy than computing!
- Small robots cannot have ***a large Fuel-Tank or Battery!***
- Robots cannot refuel or recharge while moving!

Our Assumption:
[Energy bound = B]   => At most B moves per robot.

When a robot runs out of battery it dies!

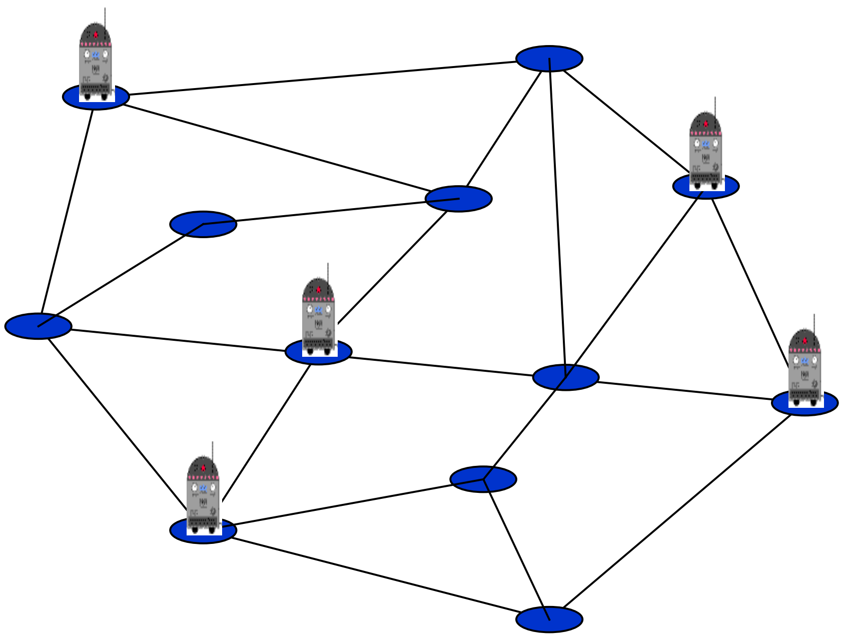
The Model

- Environment: Connected graph G.
- Nodes are identical, edges are locally ordered.
- The robots are numbered *1,2,3 ... k*
- Robots have internal memory.
- *Local* Visibility
- Communication:
  - ***Local*** **:** Face to face
  - ***Global*** : Wireless.
- *Each robot can traverse at most **B** edges.*



The Problems

- Data Transfer
  - One source to one target
  - Many to one (Convergecast)
  - One to Many (Broadcast)
- Exploration / Search
- *Map Construction*
- *Rendezvous*
- *Pattern Formation*



Optimization of Energy

- Total Energy Consumption

  (Total Movements / Time)

PREVIOUS RESULTS

- **B** : Maximum Energy used by a Robot

  (For fixed number of robots: k)

- **k** : Number of Robots used

  (For fixed energy bound B)

OUR OBJECTIVE

- Bi-criteria Optimization
- Time versus Energy tradeoff

FUTURE WORK

Prior Knowledge

OFFLINE

- With Global Knowledge

  (Global Communication between robots)

Optimize actual cost!

ONLINE

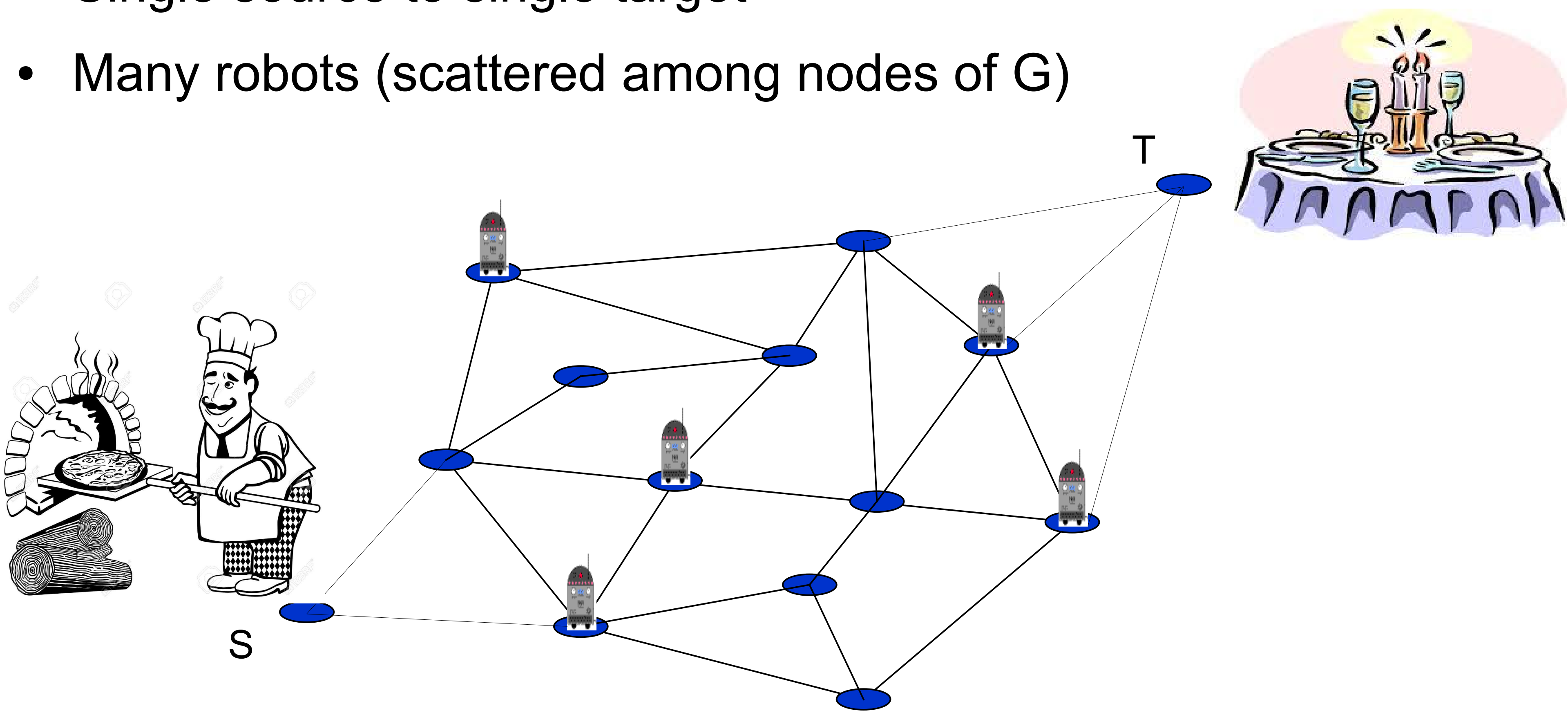
- Without Prior Knowledge

  (Local Communication between robots)

Optimize ***Competitive Ratio****!*

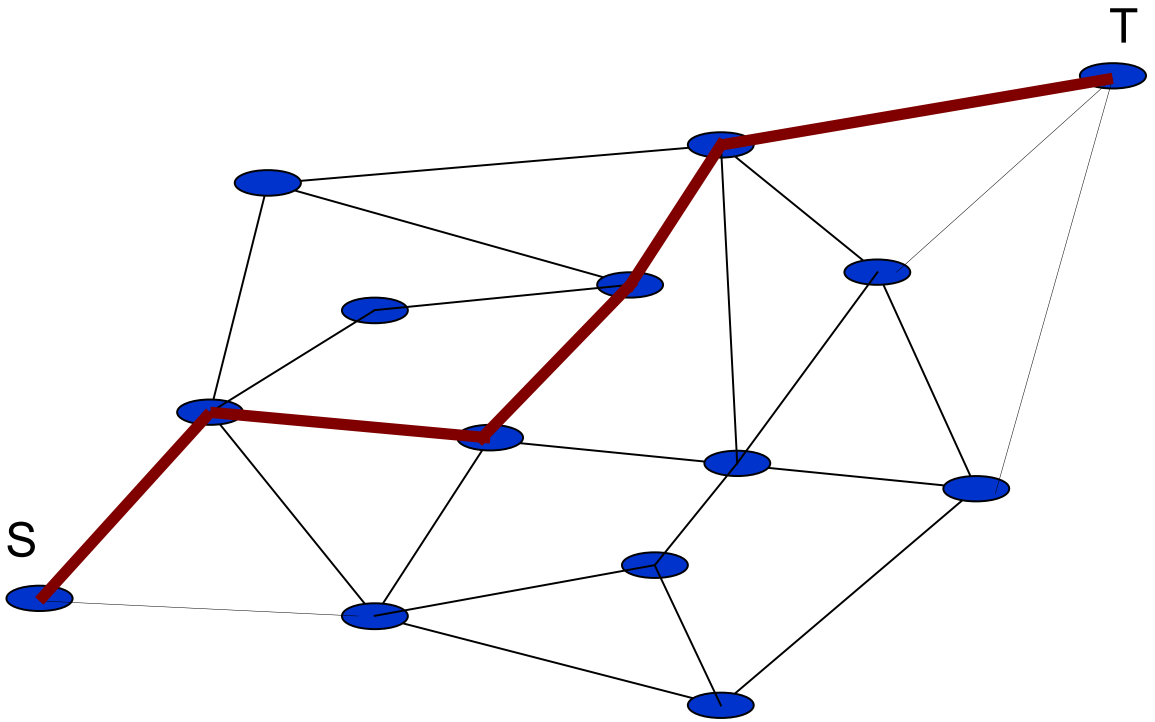
A simple Problem : Pizza Delivery

- Single source to single target
- Many robots (scattered among nodes of G)



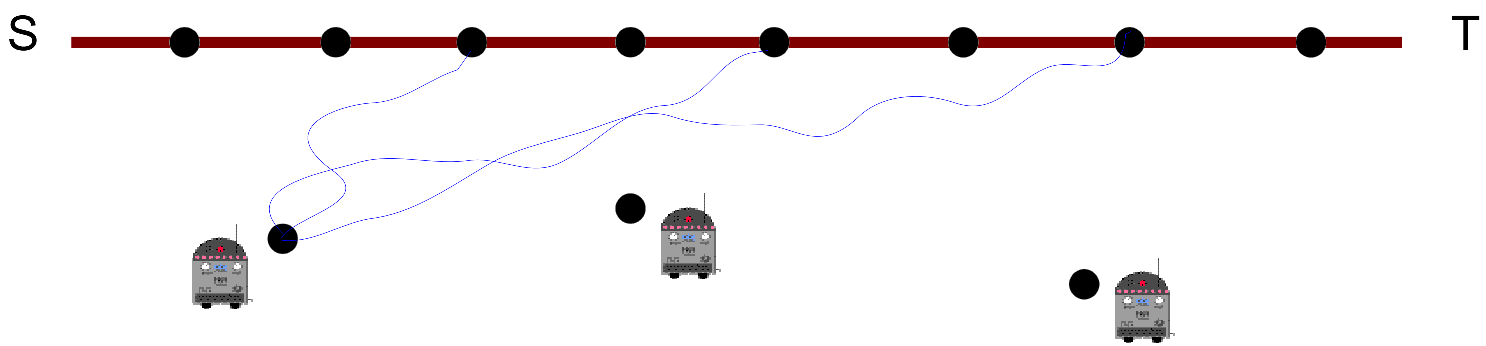
A simple Problem : Pizza Delivery

- Pizza must travel on some S-T path.
- Each robot pushes pizza on a continuous part of this path.



A simple Problem : Pizza Delivery

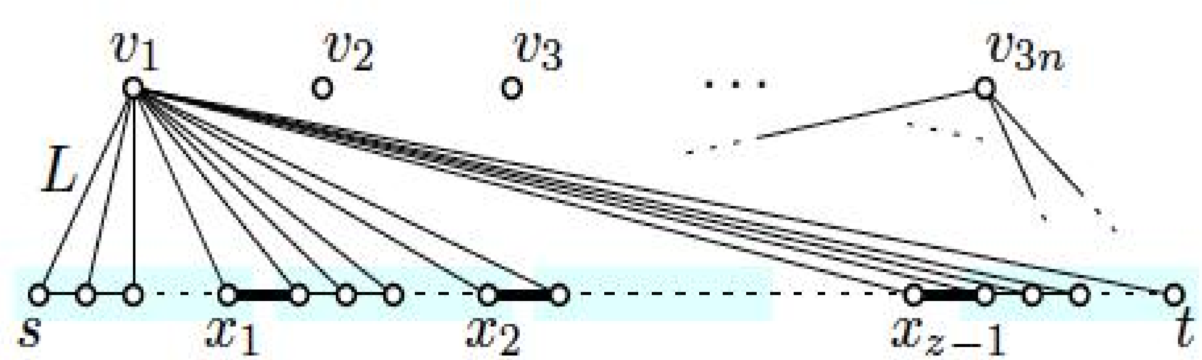
- Pizza must travel on some S-T path.
- Each robot pushes pizza on a continuous part of this path.



Order on Robots   => Strategy for Delivery

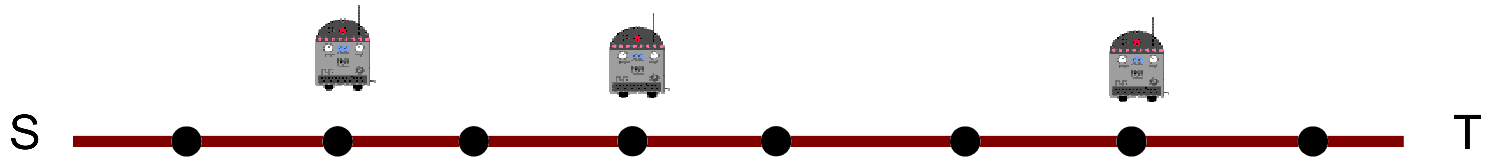
Pizza Delivery is NP-complete

- By a reduction from 3-PARTITION Problem [ALGOSENSORS 2013]



Pizza Delivery on a Line

- Pizza Delivery on a line is poly-time solvable.
- If each robot is already on the line and has same energy B.

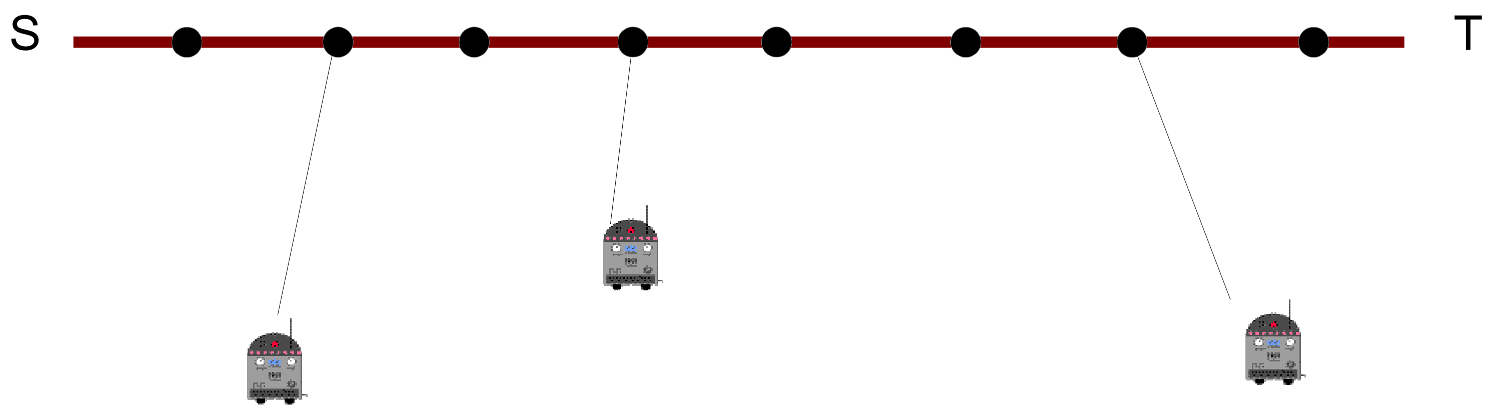


If robots have arbitrary energy levels (B1,B2,B3,B4 ...)

- Pizza-Delivery on a line is (weakly) NP-hard !
- Reduction from Weighted-4-partition problem.
  [Chalopin et al. ICALP 2014]

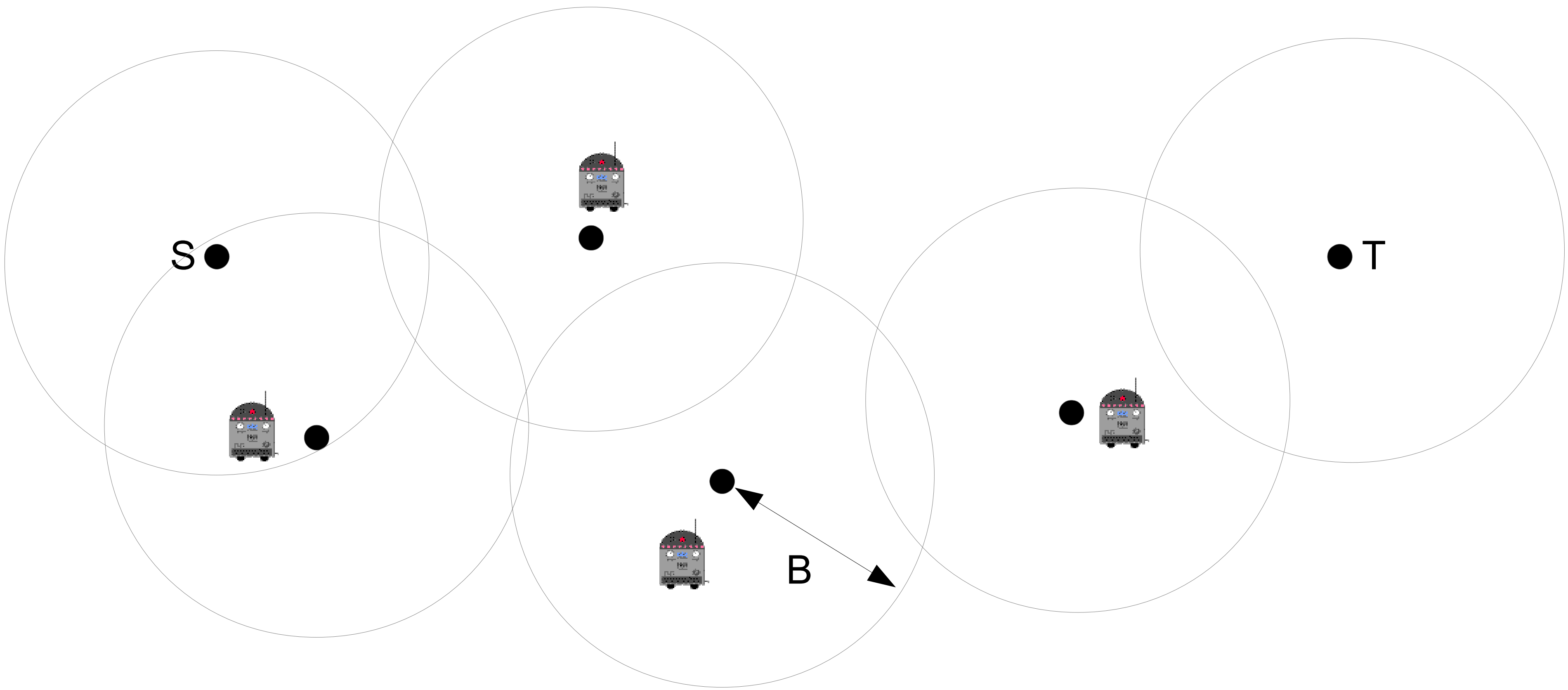
Pizza Delivery on a Tree

- Pizza Delivery on a **tree** is **NP-hard**.
- Even if each robot start with same energy B.



Algorithms for Pizza Delivery

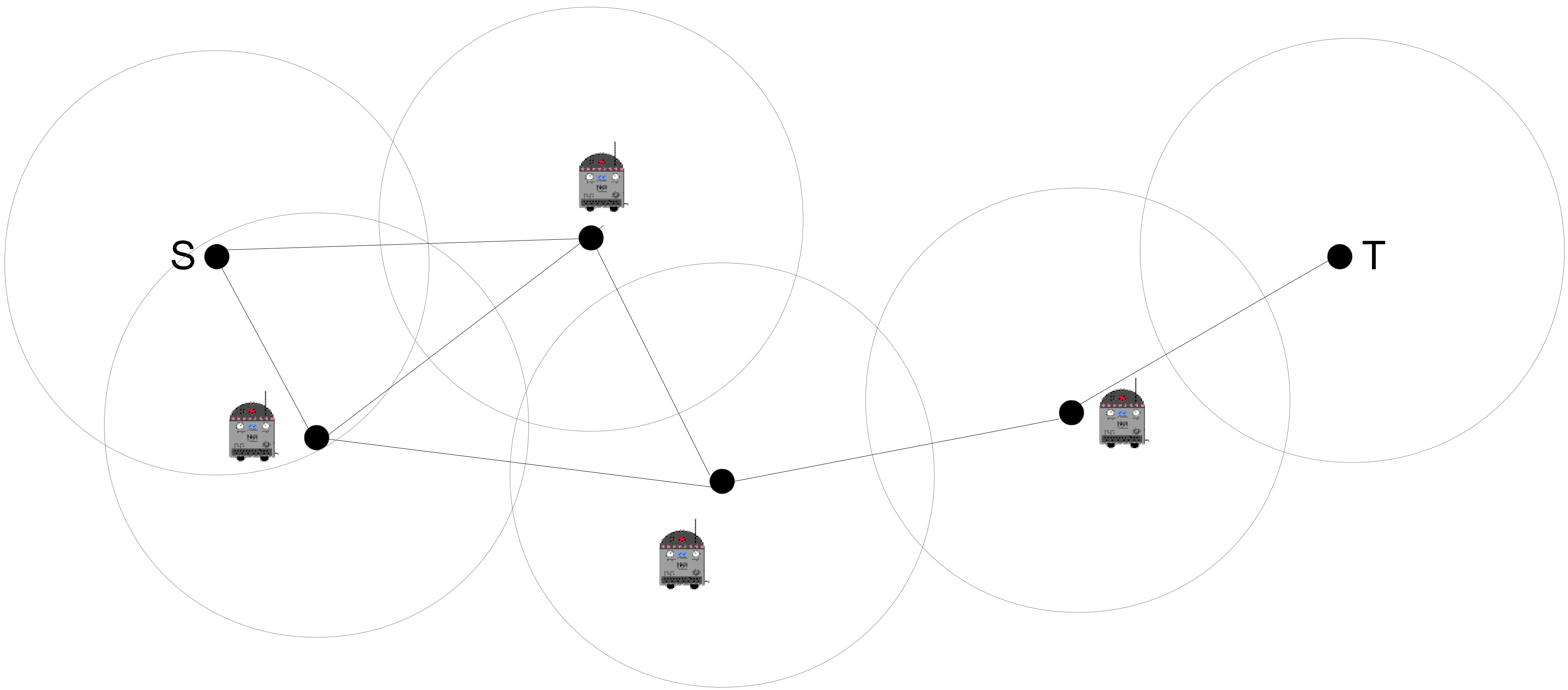
Necessary Condition:



Algorithms for Pizza Delivery

Necessary Condition:

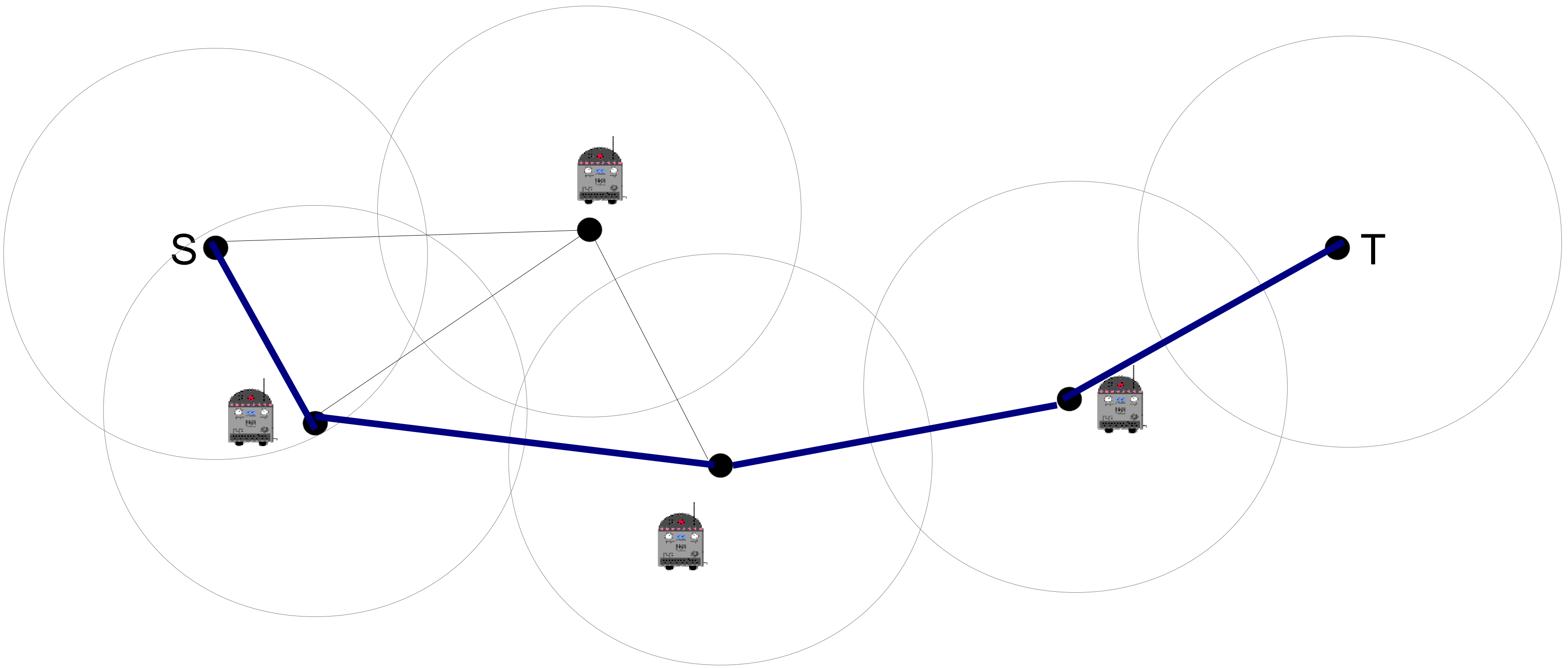
- There exists a S-T path in the intersection graph.



Algorithms for Pizza Delivery

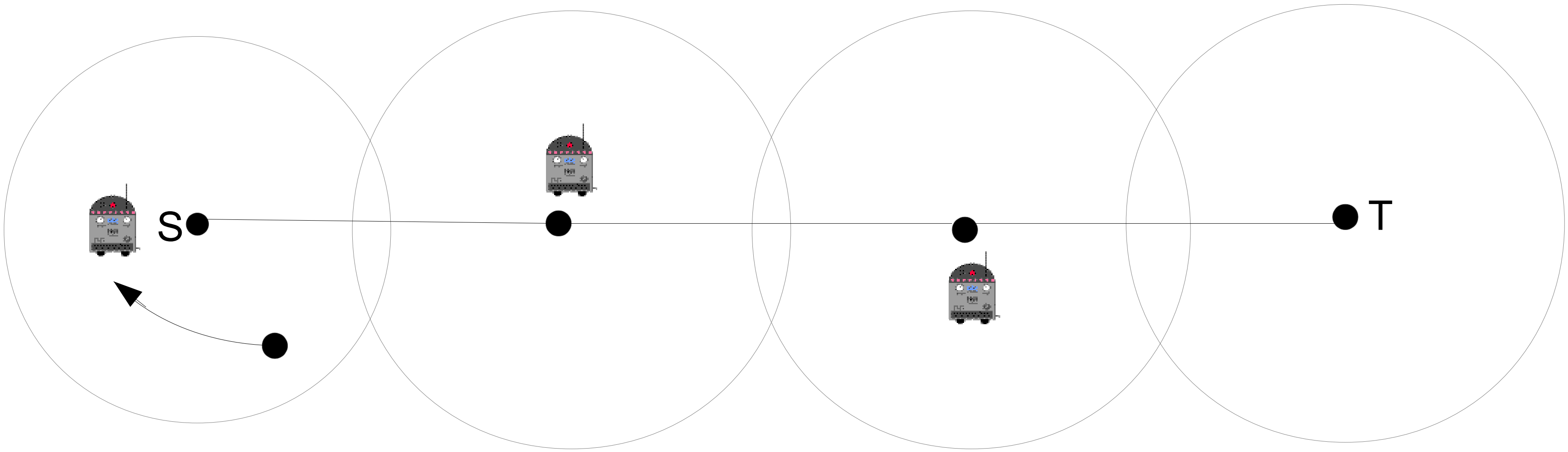
If there exists a S-T path in the intersection graph,

=> there is poly-time algorithm using 3B energy per robot.



2-Approx. Algorithm

- Suppose there is a robot at S.
- Each robot can carry to neighboring robot using 2B energy.
- Guess the first robot r(i) in the optimal strategy.
- Place r(i) at S with reduced energy (smaller ball).

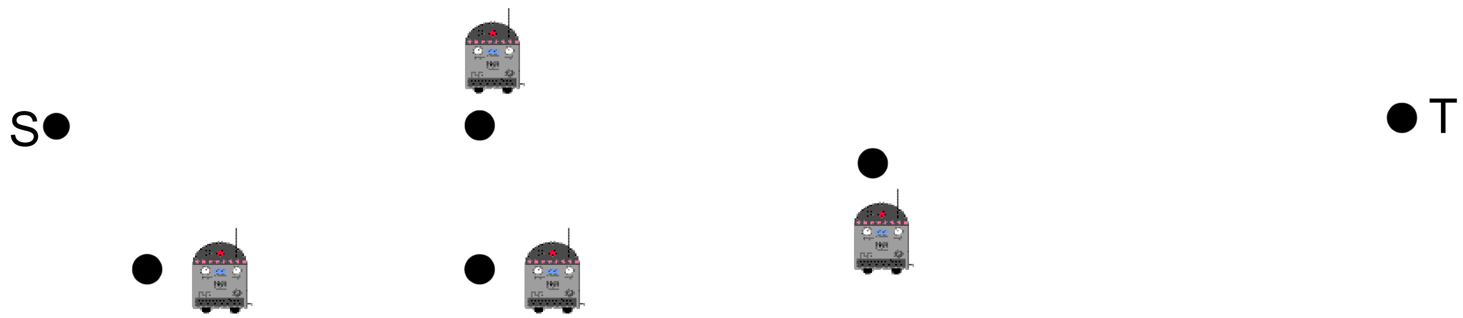


Robots in Continuous Space

<u>Open Question</u>:

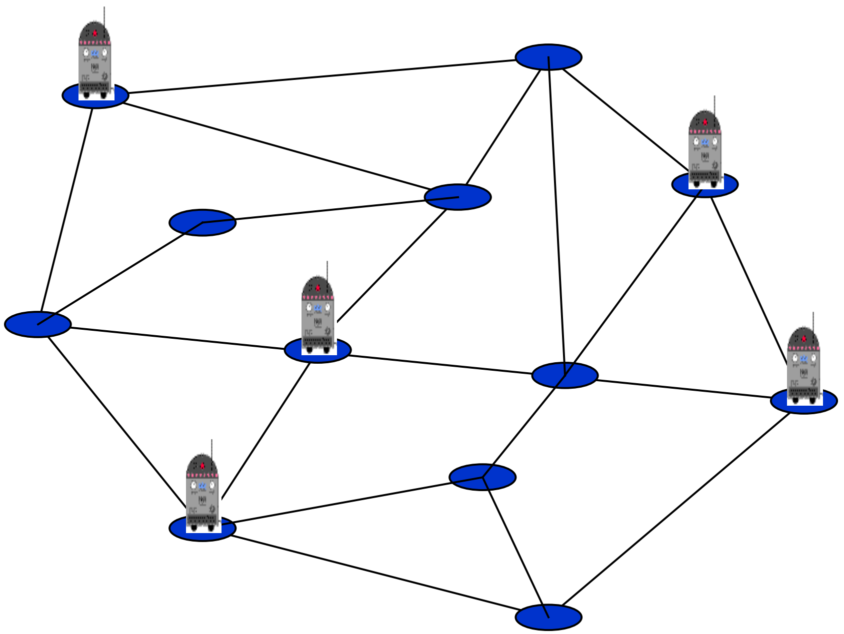
- How to solve Pizza-delivery in 2D plane?

When each robot can move an Euclidean distance of at most B.



Robot to Robot Data-Transfer

- Each robot carries some data.
- Robots can exchange information on meeting at a node.
- Problems studied:
  - **Convergecast** (many to one)
  - **Broadcast** (one to many)



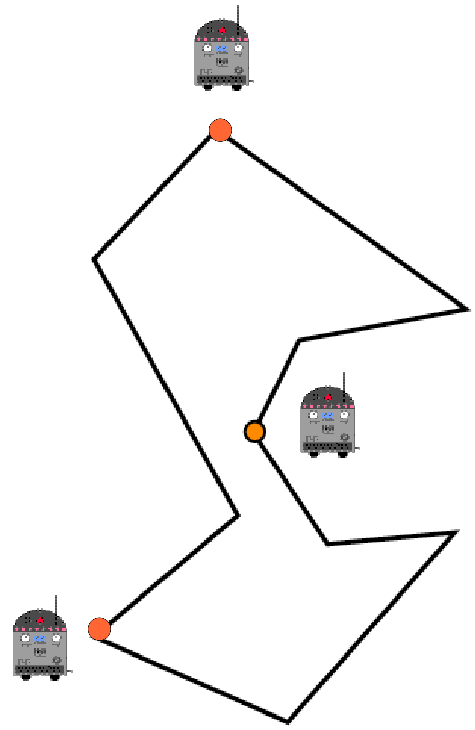
Robot to Robot Data-Transfer

Results: [Anaya et al. 2012]

- OFFLINE
  - **Convergecast and Broadcast** are NP-hard in Trees
  - 2-approximation algorithm for any graph (Convergecast)
  - 4-approximation algorithm for any graph (Broadcast)
- ONLINE
  - 2-competitive algorithm (Convergecast)
  - 4-competitive algorithm (Broadcast)
  - No (2-ε) competitive algorithm is possible.

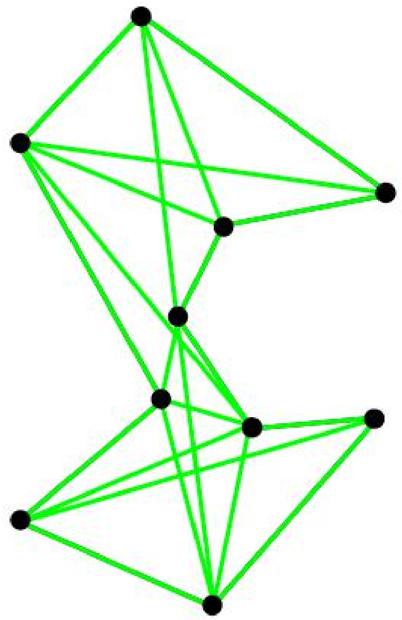
Robots moving on Polygon

- Robots occupy vertices of polygon
- Can move to any visible vertex
- At most B moves per robot



Robots moving on Polygon

- Robots occupy vertices of polygon
- Can move to any visible vertex
- At most B moves per robot



<u>Problems studied</u>:

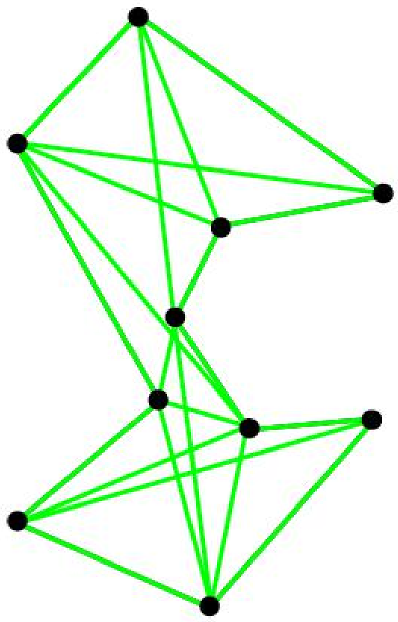
- **Rendezvous**
  - Gather in one vertex
- **CONNECTED**
  - Form a connected configuration
- **CLIQUE**
  - Place robots on a k-clique

Robots moving on Polygon

Results: [Bilo et al. 2013]

OFFLINE Optimization

- Rendezvous
  - O(mn) time to compute
- CONNECTED
  - NP hard to compute optimal strategy
  - APX-hard (for Euclidean distance)
- CLIQUE
  - NP hard to compute optimal strategy
  - No (1.5 – ε) approximation algorithm



Global Knowledge

OFFLINE

- With Global Knowledge

  (Global Communication between robots)

Optimize actual cost!

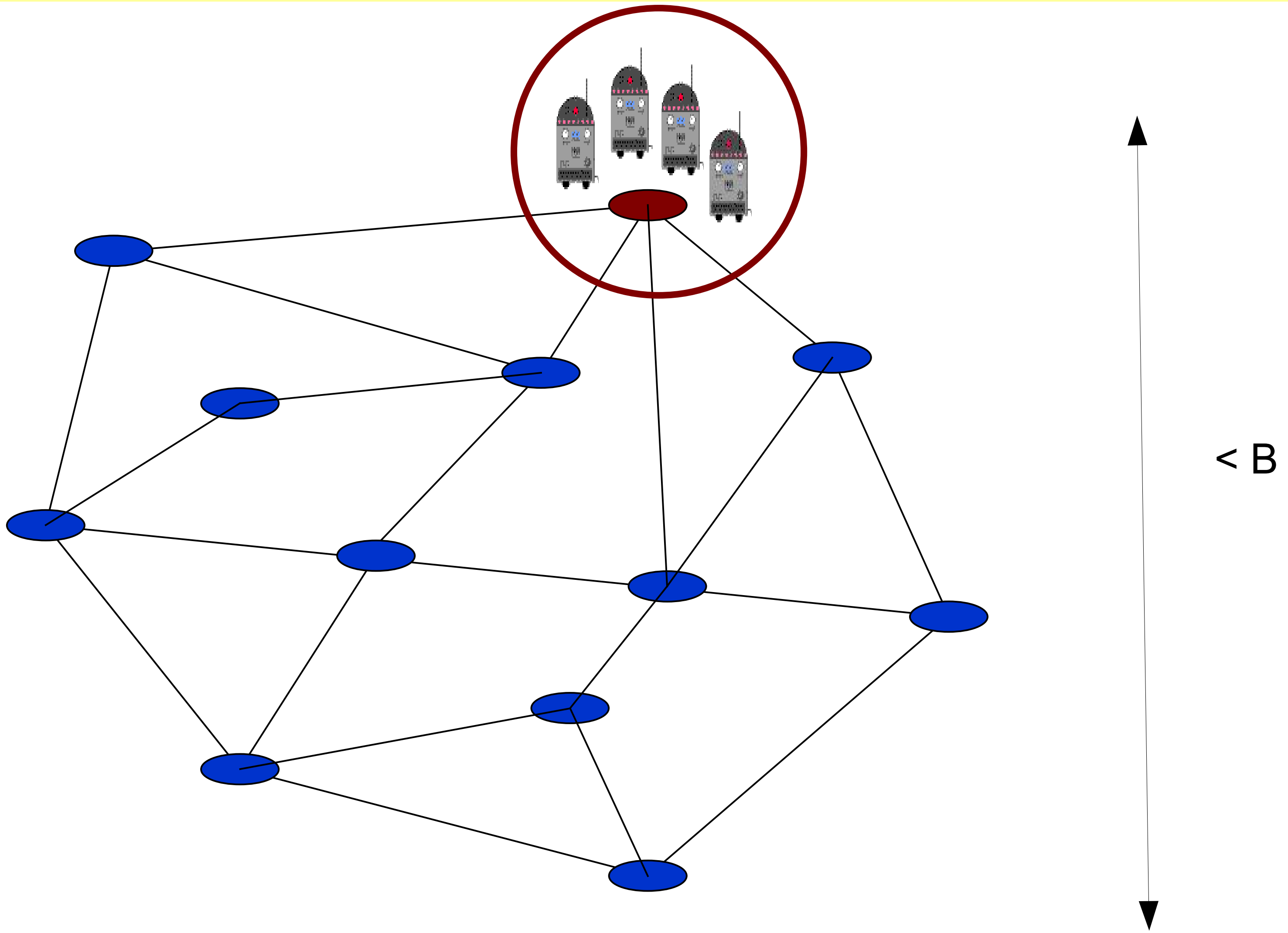
ONLINE

- Without Prior Knowledge

  (Local Communication between robots)

Optimize Competitive Ratio!

Exploration Problem



Exploration of Known Trees

Instance: An undirected tree $T = (V,E)$, $|V| = n$, a fixed node $r \in V$, an integer $k > 0$

Solution: tours $C\_1, C\_2, \ldots C\_k$, where $\cup C\_i = E$ and each tour contains the node r.

Goal: Minimize $B = \max\{|Ci| : i = 1, \ldots k\}$

Computing Optimal offline exploration is NP-hard!

[Fraigniaud et al. 2006]

Reduction from
3-PARTITION Problem

Online Exploration

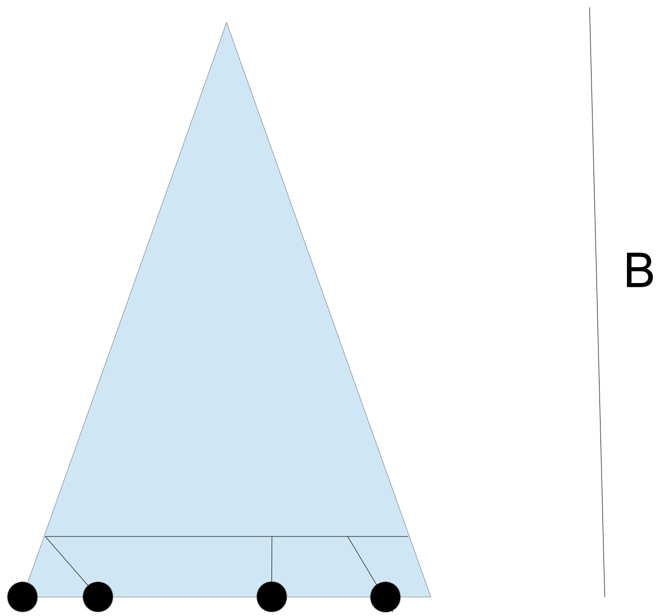
- The offline version of the problem is NP-hard, even for trees.
- We consider the online exploration problem for Trees.
- For any tree T and starting vertex r,
  - Let Cost(T,r) be cost of our online exploration algorithm
  - Let OPT(T,r) be cost of optimal offline algorithm

- Competitive Ratio = $\max_{(\text{all } T,r)} \left(\text{Cost}(T,r) / \text{OPT}(T,r) \right)$

Online Tree Exploration

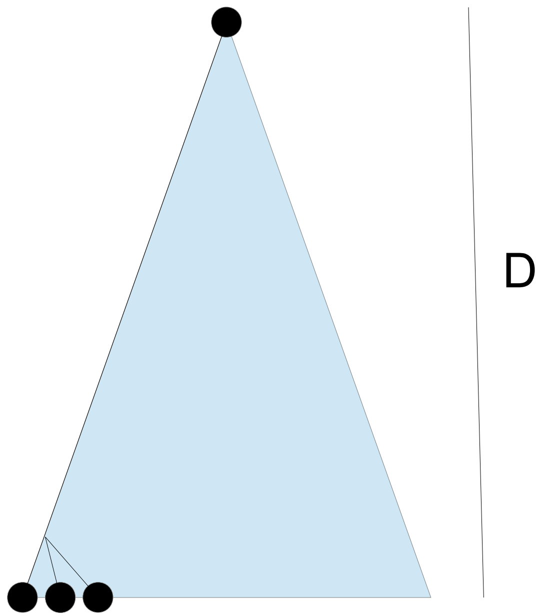
- The tree T is unknown, except for starting vertex r.
- For a team of **k agents**, minimize B [Dynia et al. 06]
  - 2-approximation algorithm (Offline version)
  - Competitive ratio of 8 (Online version)
  - Lower bound of 1.5
- For robots of **fixed energy B**, minimize team-size k [ThisTalk]
  - Algorithm using O(log B).OPT agents (Local communication)
  - Lower bound of Ω(log B).OPT agents

Height of the Tree



- If the height of the tree (from r) is more than B it cannot be fully explored!
- We assume that the height of the tree is at most B-1.

Lower Bound



(1) If there is no communication between r and depth D-1

- Algorithm sends x agents.
- Algorithm fails if x+1 leaves

(2) If there is communication between r and depth D-1

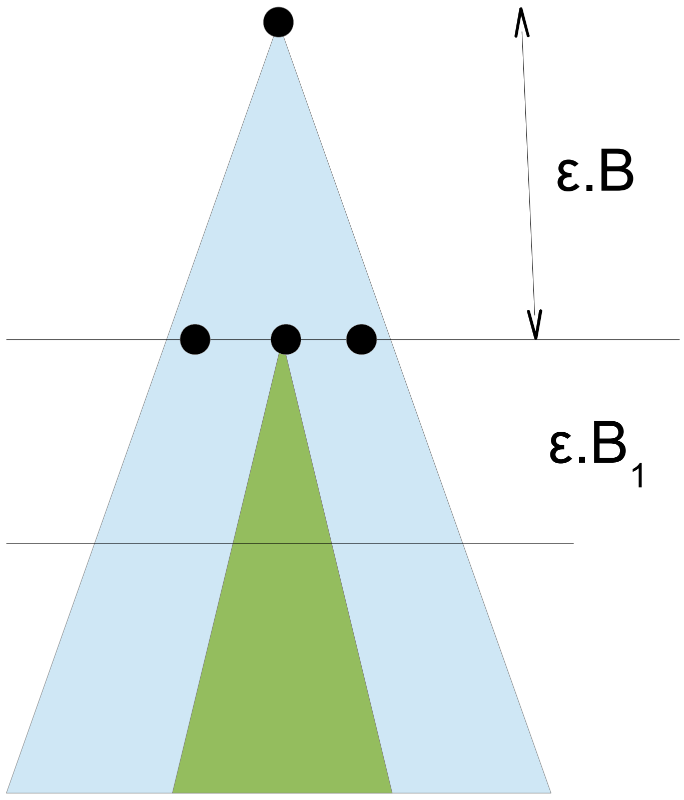
- If D=B-1, at least (log B) agents needed for communication
- If only one leaf, then competitive ratio = log B

Any online algorithm has competitive ratio of $\Omega(\log B)$

Our Algorithm

- Recursive Algorithm
- Explore up to depth (ε.B)
- For each node at next level, recursively call the algorithm
- Number of levels = $\log_{(1/1-\varepsilon)} B$

(We try to use no more than OPT agents for each level)



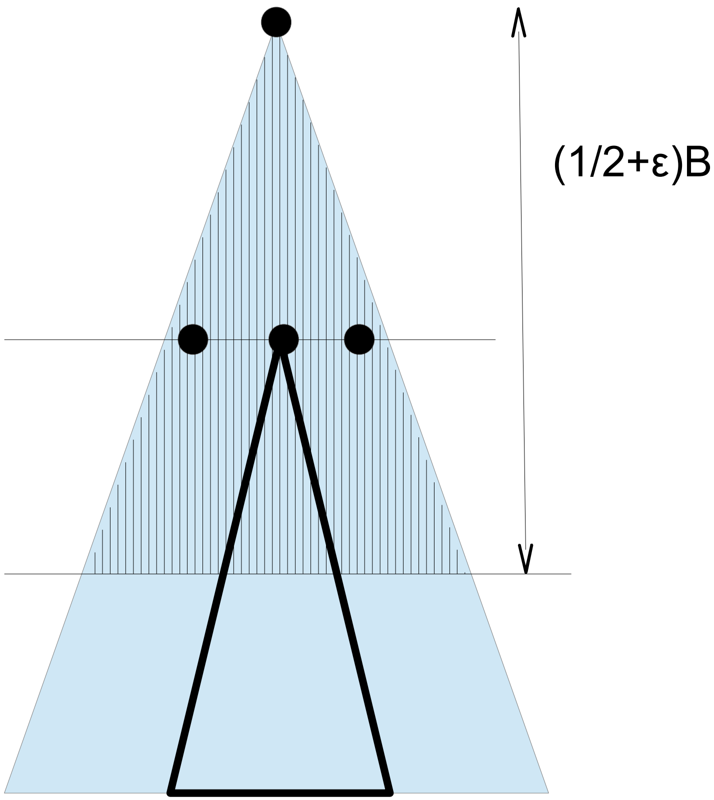
$0 < \varepsilon < 1/4$

The Look-ahead

- For each level i, explore beyond the next level (i+1)
- Overlap of depth = 1/2 B_i
- For each node at level (i+1), the algorithm is called only if there are unexplored nodes in the sub-tree.

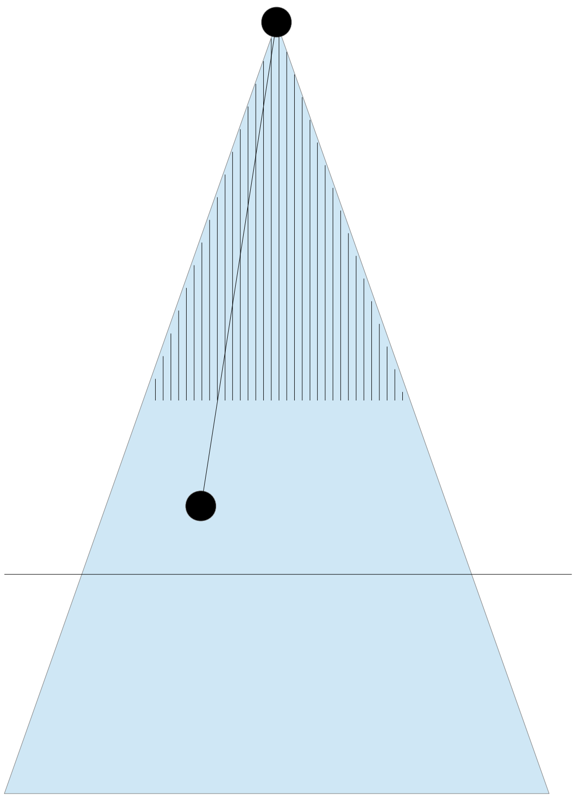
Two sub-trees at the same level are ***independent***!

(No agent can go from unexplored part of one subtree to unexplored part of the other subtree)



Exploring a sub-tree

- Perform DFS restricted to depth d_i
- If an agent runs out of energy, the next agent from the root, arrives to continue with the exploration.
- Each agent saves $x(b)= (1/2-\varepsilon)b/2$ units of energy for later use.



<u>Note</u>:
We assume Global communication.
We will later remove this assumption.

Cost of the Algorithm

- Each agent uses at least (1/2-ε)b/2 units of energy for exploring new nodes.
- For **k** agents, we have

k . (1/2-)b/2 **>** 2.|T| **>** 2 . OPT . b

- If the subtrees at a level are independent, we can add the costs.
- Thus, the total number of agent used at each level is a constant times the optimal number of agents for the whole tree.
- Cost of the algorithm = O(log B) . OPT

From Global to Local Communication

- Each agent A needs a constant number (m<4) of helper agents.
- The first helper A1 goes halfway with agent A and waits, the second helper A2 goes half of the remaining depth and waits, and so on.
- When agent A runs out of energy, it uses the saved energy to move towards to the last helper agent A_m.
- The information is propagated to the root of the subtree.
- So we have a competitive ratio of O(log B) even for the local communication model.

Conclusions

- We presented an algorithm for exploring an unknown tree with multiple agents, each having limited energy B.
- The number of agents used by the algorithm is O(log B) times the optimal offline algorithm. This result is asymptotically tight.
- The competitive ratio is *independent of the size of the tree* (and depends only on the height or the energy limit).
- Our algorithm can explore trees of height at most B, while the algorithm for single agent with refuelling can only explore trees of depth B/2.

Open Problems

- How to explore general graphs with energy-constrained robots? What is the competitive ratio in that case?
- What if the robots are allowed to exchange energy (i.e. If a robot can give its remaining energy to recharge another robot)?
- What is the competitive ratio of exploration with global communication?
- If the graph/tree is large, how many nodes can be explored by an online algorithm compared to the optimal offline algorithm?

THANK YOU!

MERCI