Feasibility of Asynchronous Rendezvous

Jurek Czyzowicz[1], David Ilcinkas[2],
Arnaud Labourel[2], Andrzej Pelc[1]

[1]Département d'informatique, Université du Québec en Outaouais,
Gatineau, Québec, Canada

[2]LaBRI, University Bordeaux, Talence, France

Carleton University, Distributed Computing by Mobile Robots
August 16th 2010

The rendezvous problem

The problem

Two mobile robots must meet in a geometric terrain.

Modeling the problem :

- Geometric terrain $\rightarrow$ A subset of plane with polygonal obstacles
- Mobiles robots $\rightarrow$ points moving in the terrain
- Agent's visibility region $\rightarrow$ A closed circle of radius $r \geq 0$ centered at robot's position
- Rendezvous $\rightarrow$ each robot belongs to the visibility region of the other robot
- Cost $\rightarrow$ sum of the lengths of the trajectories of the robots until rendezvous

Asynchronous rendezvous in CORDA

Rendezvous as a point formation problem

Previous work (some)

- Cieliebak, Flocchini, Prencipe, Santoro, ICALP 2003, gathering assuming multiplicity detection
- Prencipe, SIROCCO 2005, unsolvable without multiplicity detection
- Cohen, Peleg, ESA 2004 gravitational algorithm (also imprecise sensors)
- Ando, Oasa, Suzuki, Yamashita IEEE Trans. Robotics and Autom, 1999, limited visibility, instantaneous execution of the cycle
- Flocchini, Prencipe, Santoro, Widmayer, TCS 2005, limited visibility with sense of direction (the same coordinate system)

The model

- The terrain is unknown to the robot
- Robot is aware whether it is at a boundary point of the environment
- Robot stops while continuing the movement would bring it out of the terrain
- Robot is aware that another robot is within its visibility range
- Robot is aware of the distance traveled
- Robots are labeled (by integer numbers), though some results extend to anonymous robots
- In some cases robots have the same system of coordinates (i.e. North direction, unit distance and chirality)
- Robots have memory (bounded if the terrain is bounded)

Asynchronous movement

The robots

- Each robot chooses a sequence of steps forming its route (or the algorithm computing it)
- The robots try to choose their routes so they always meet

The omniscient adversary

- Tries to prevent the rendezvous
- Chooses the identities of the two robots and their starting positions
- Knows in advance the route chosen by the robot and determines the duration of each step of the route.

The route and the walk

The route

The route R chosen by the robot is a sequence of segments $(e_1, e_2, \dots)$. In stage i the robot traverses segment $e_i = [a_{i-1}, a_i]$, starting at a_{i-1} and ending in a_i. Stages are repeated indefinitely (until rendezvous).

The walk

Let $(t_1, t_2, \dots)$, where $t_1 = 0$, be an increasing sequence of reals, chosen by the adversary, that represent points in time. Let $f_i : [t_i, t_{i+1}] \to [a_i, a_{i+1}]$ be any continuous non-decreasing function, chosen by the adversary, such that $f_i(t_i) = a_i$ and $f_i(t_{i+1}) = a_{i+1}$. For any $t \in [t_i, t_{i+1}]$, we define $f(t) = f_i(t)$. The interpretation of the walk f is as follows: at time t the robot is at the point $f(t)$ of its route.

The asynchronous rendezvous problem

The feasibility

The asynchronous rendezvous problem has a solution if it is possible to choose a route for each agent $A_1, A_2, \ldots$ such that

- for any choice of agents A_i, A_j
- for any starting positions of A_i, A_j
- for any walks of the agents A_i, A_j

the agents A_i, A_j will eventually meet

The cost

The cost of the asynchronous rendezvous is the maximum sum of lengths of routes of the two agents (taken over all possible actions by the adversary)

Asynchronous rendezvous for the bounded terrain (Cz., Ilcinkas, Labourel, Pelc, SIROCCO 2010)

The same system of coordinates, anonymous robots

- $\Theta(P)$ cost ($\Theta(D)$ if map known, P, D are, respectively, the terrain perimeter and the original distance between the robots)
- Works also if not the same unit of length

The simple polygon (i.e. no obstacles), anonymous robots

- $\Theta(P)$ cost ($\Theta(D)$ if map known)
- Compute the medial axis and meet at the center

Labeled robots, obstacles, incoherent compasses

- $\Theta(P + x|L_{max}|)$ cost ($\Theta(D|L_{min}|)$ if map known), x is the perimeter of the largest obstacle, L_{min}, L_{max} are, respectively, lengths of the smaller and the larger label

The rendezvous problem in the plane

The problem

Two mobile robots must meet in the plane.

Modeling the problem :

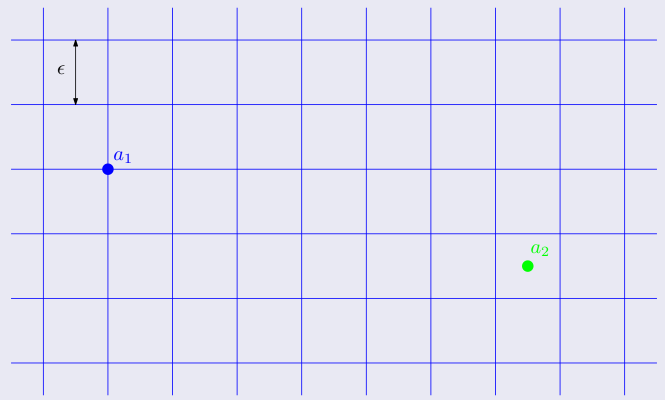
- Robots → points moving inside the plane along a polygonal trajectory. The robots have coherent compasses showing North and a common unit of length.
- Agent's visibility range → A circle of radius $\epsilon > 0$ centered at robot's current position
- Rendezvous → each robot belongs to the visibility range of the other robot
- Cost → sum of the lengths of the trajectories of the robots until rendezvous

Proof of the rendezvous algorithm in the plane without obstacles (the same coordinate system, unit visibility)

From plane to grid

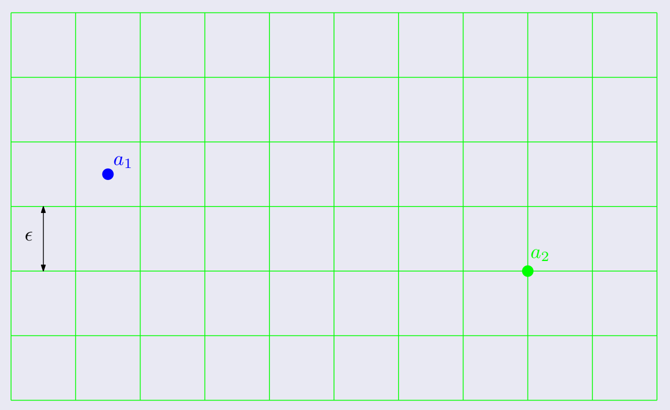
Proof of the rendezvous algorithm in the plane without obstacles (the same coordinate system, unit visibility)

From plane to grid



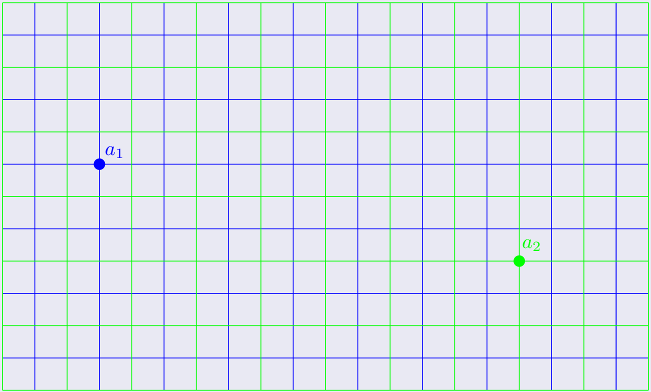
Proof of the rendezvous algorithm in the plane without obstacles (the same coordinate system, unit visibility)

From plane to grid



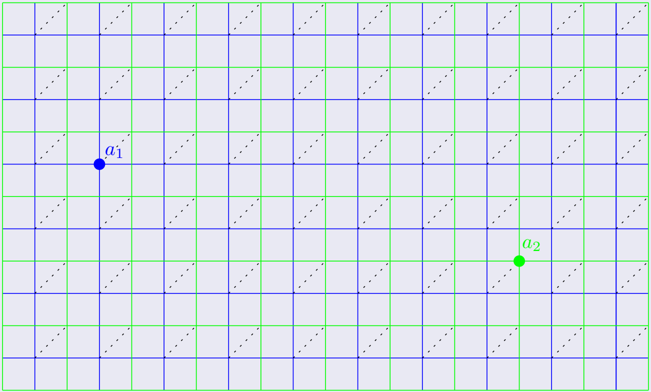
Proof of the rendezvous algorithm in the plane without obstacles (the same coordinate system, unit visibility)

From plane to grid



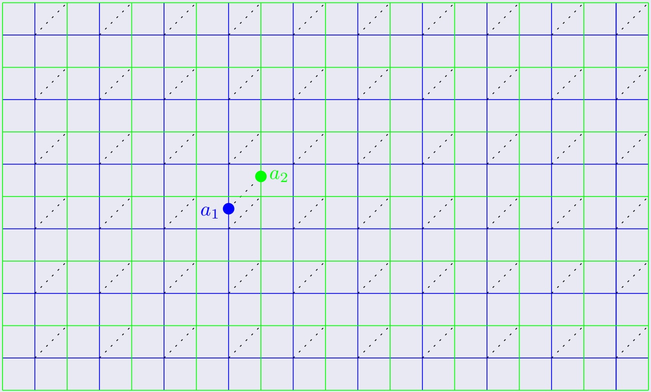
Proof of the rendezvous algorithm in the plane without obstacles (the same coordinate system, unit visibility)

From plane to grid

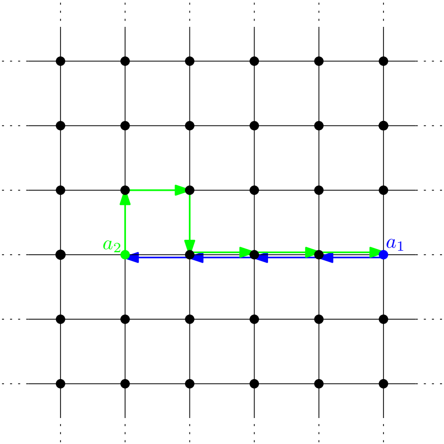


Proof of the rendezvous algorithm in the plane without obstacles (the same coordinate system, unit visibility)

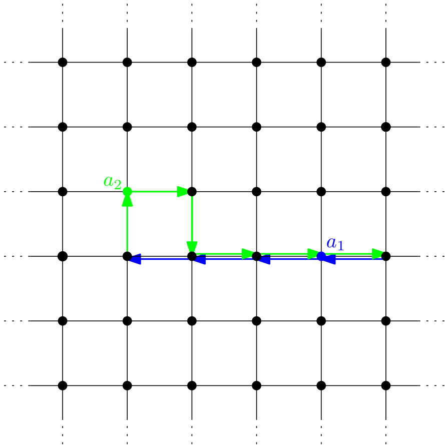
From plane to grid



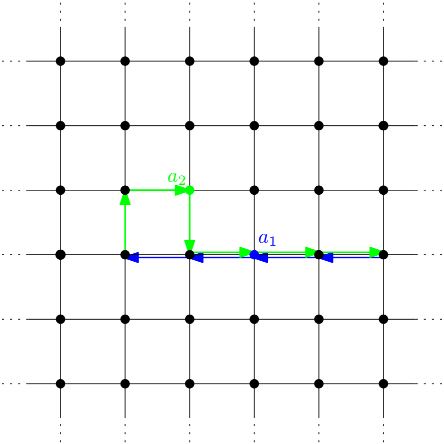
Example of routes that does not guarantee rendezvous



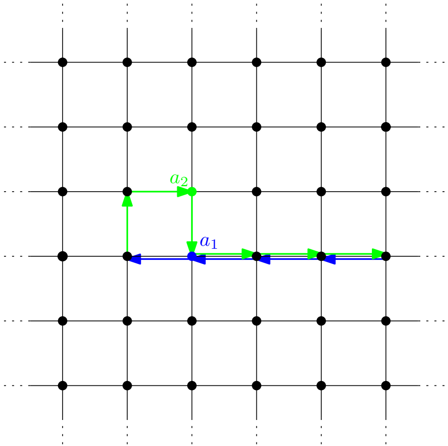
Example of routes that does not guarantee rendezvous



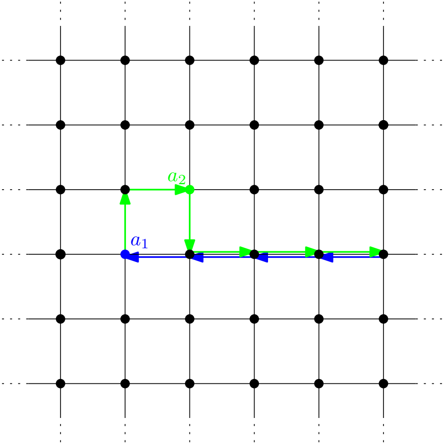
Example of routes that does not guarantee rendezvous



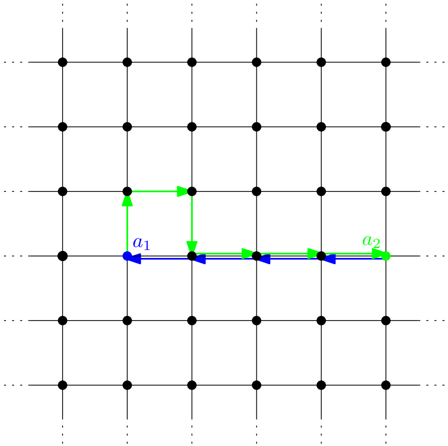
Example of routes that does not guarantee rendezvous



Example of routes that does not guarantee rendezvous

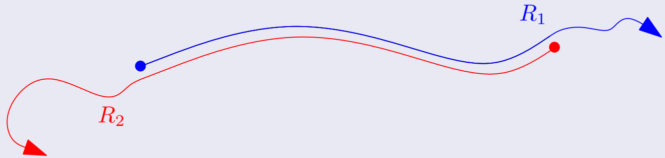


Example of routes that does not guarantee rendezvous



General idea of the algorithm

The tunnel



Tunnel : sequence of segments $s_1, s_2, \ldots, s_k$ such that :

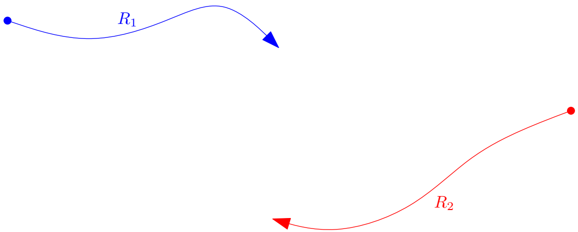
- route of robot 1 begins by $s_1, s_2, \ldots, s_k$
- route of robot 2 begins by $s_k, s_{k-1}, \ldots, s_1$

When the routes of the two robots form a tunnel the robots must meet

Forming tunnels

Fact

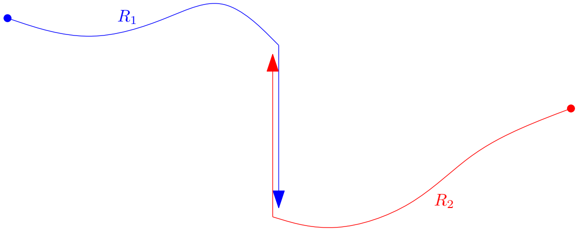
We can always extend two routes such that they form a tunnel.



Forming tunnels

Fact

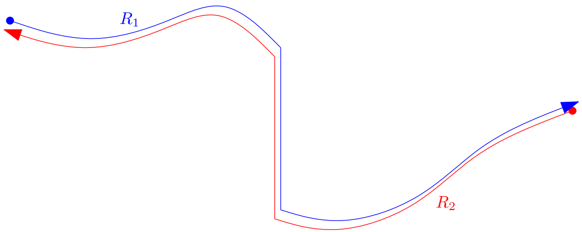
We can always extend two routes such that they form a tunnel.



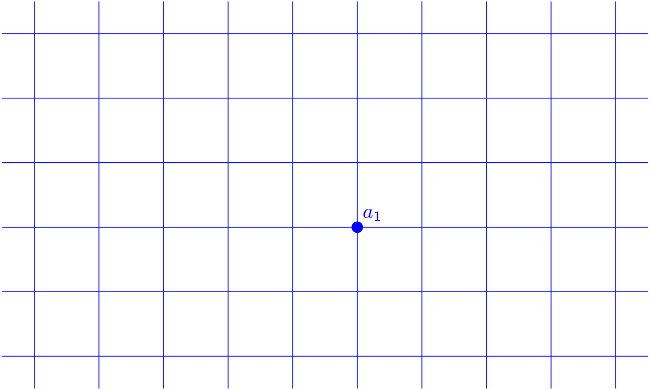
Forming tunnels

Fact

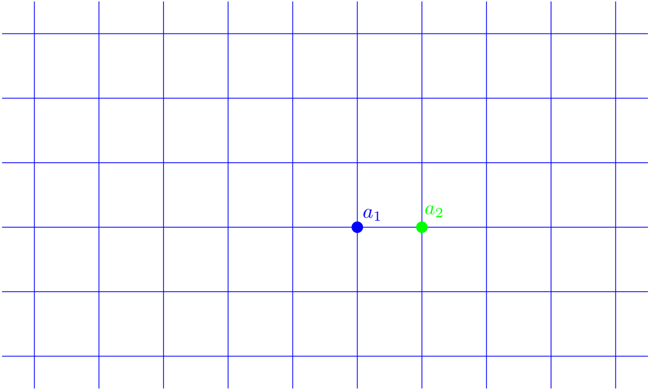
We can always extend two routes such that they form a tunnel.



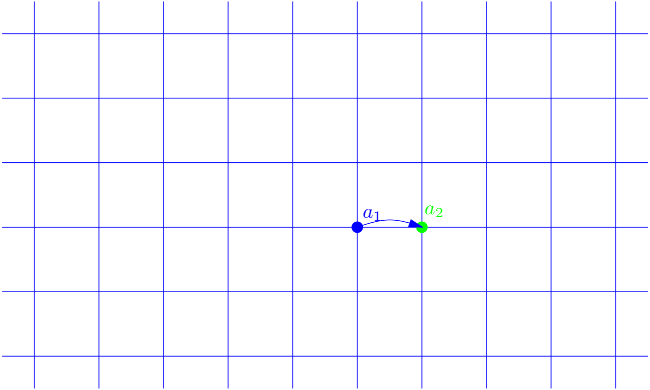
How to meet asynchronously two robots in the grid?



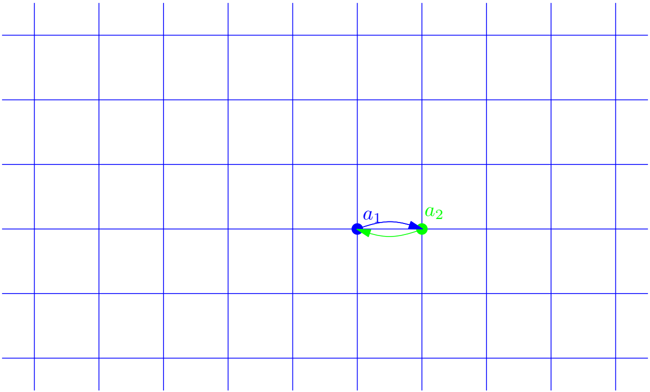
How to meet asynchronously two robots in the grid?



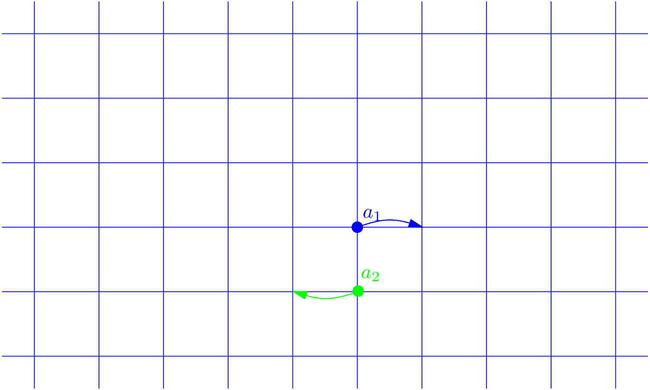
How to meet asynchronously two robots in the grid?



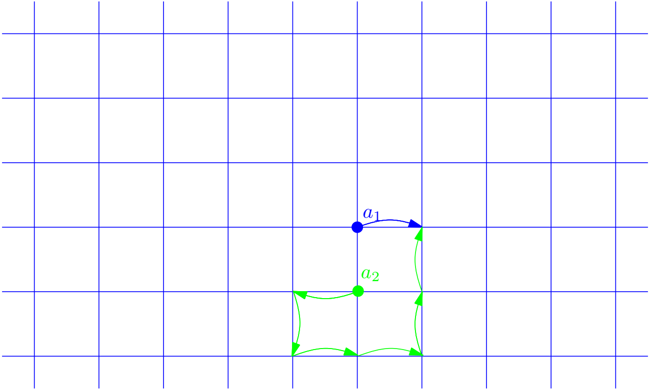
How to meet asynchronously two robots in the grid?



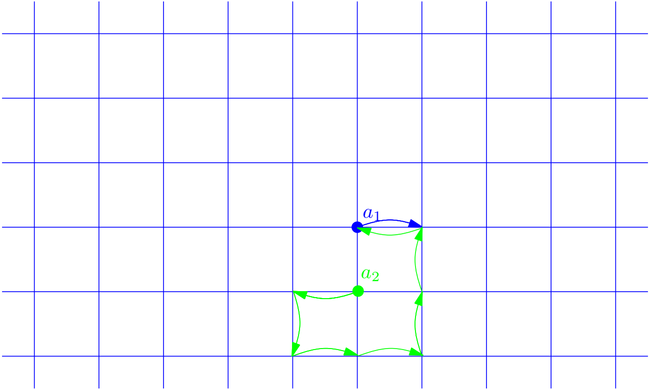
How to meet asynchronously two robots in the grid?



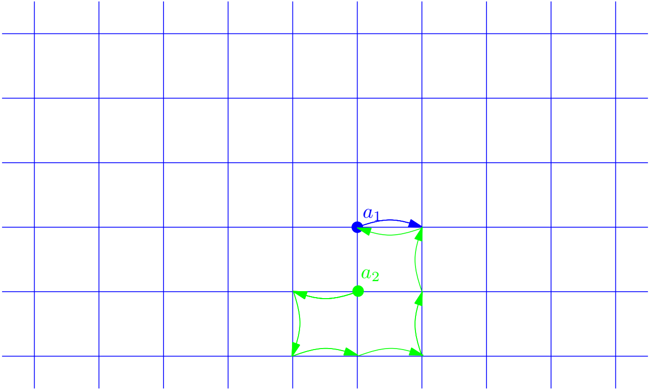
How to meet asynchronously two robots in the grid?



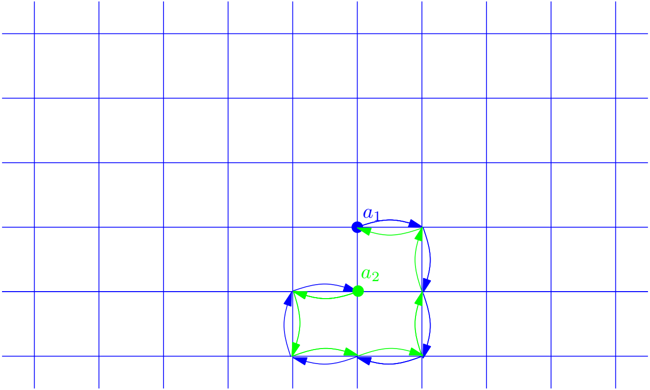
How to meet asynchronously two robots in the grid?



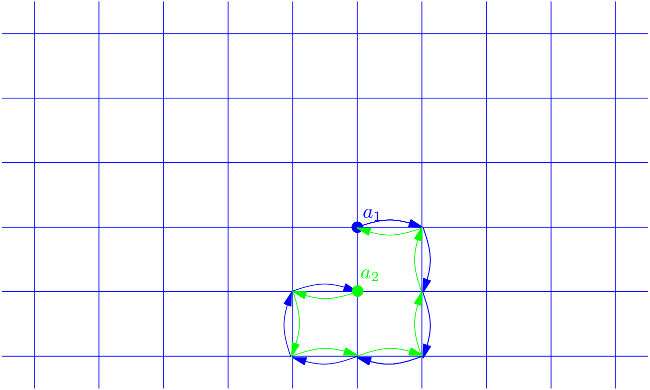
How to meet asynchronously two robots in the grid?



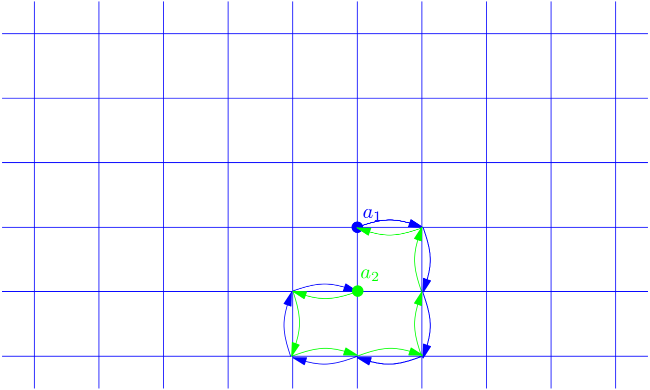
How to meet asynchronously two robots in the grid?



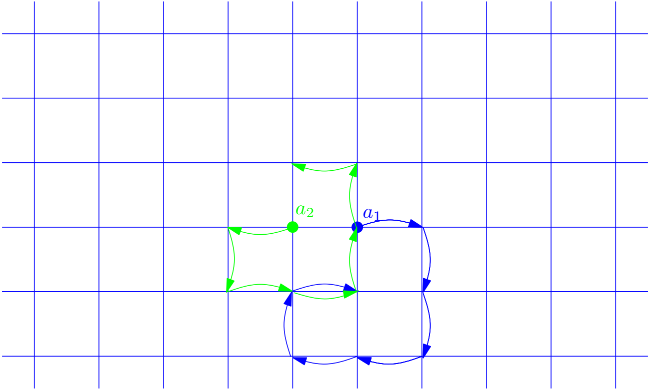
How to meet asynchronously two robots in the grid?



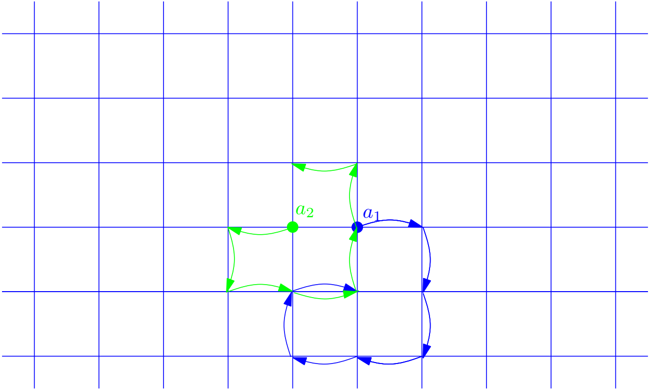
How to meet asynchronously two robots in the grid?



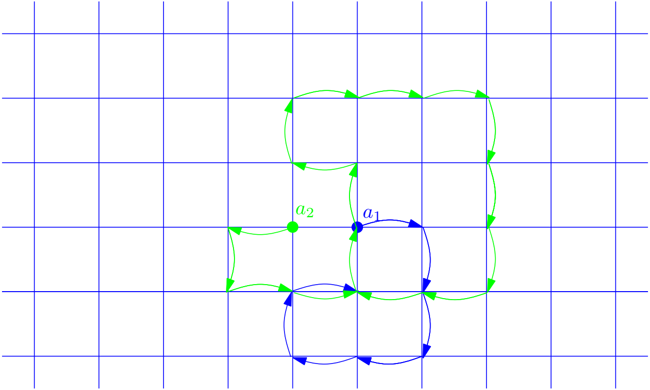
How to meet asynchronously two robots in the grid?



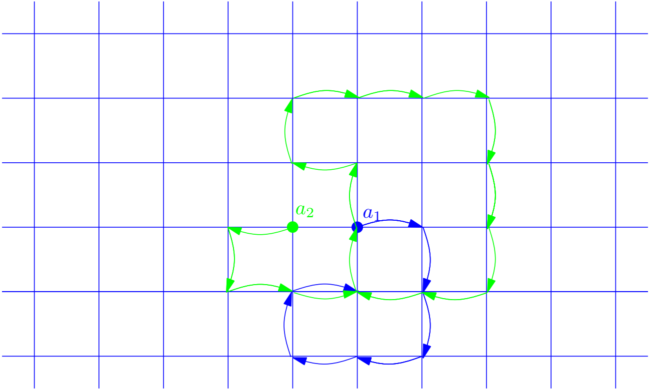
How to meet asynchronously two robots in the grid?



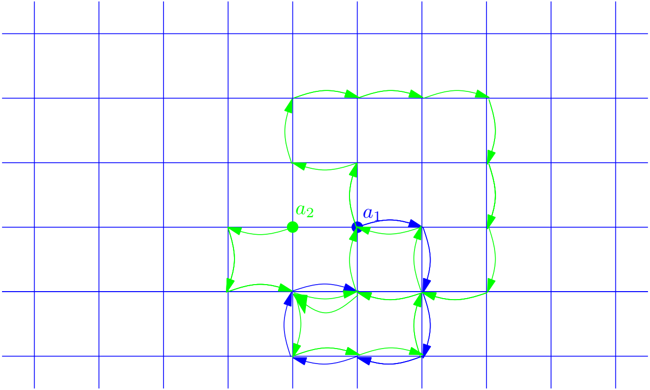
How to meet asynchronously two robots in the grid?



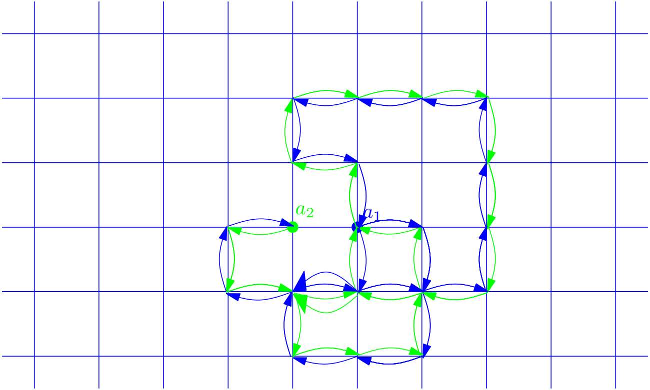
How to meet asynchronously two robots in the grid?



How to meet asynchronously two robots in the grid?



How to meet asynchronously two robots in the grid?



The rendezvous algorithm for a grid

General idea of the algorithm

- We want to create a tunnel for every possible configuration of the algorithm (for each pair of robots and their possible initial positions in the graph)
- Each configuration is a triple consisting of two robots' identifiers and their relative position in the grid (i.e. the pair of differences between their coordinates)
- Enumerate all the configurations of the algorithm
- Iteratively, for any subsequent configuration in the enumeration, extend the routes of both involved robots to form a tunnel
- Since each triple $(r_i, r_j, (x, y))$ exists somewhere in the enumeration, any placement in the grid of any pair of robots will result in their rendezvous

Rendezvous in graphs

Graphs considered

- Nodes are anonymous (do not have identifiers)
- Edges incident to a node are locally numbered (by port numbers)
- Graphs are connected, finite or infinite (with countable node set and edge set)

Movement of the agent

At each step, an agent :

- chooses a port number of the current node (outgoing port)
- moves along the corresponding edge
- accesses the target node of the traversed edge via the port number in the new node (incoming port)

Deterministic rendezvous

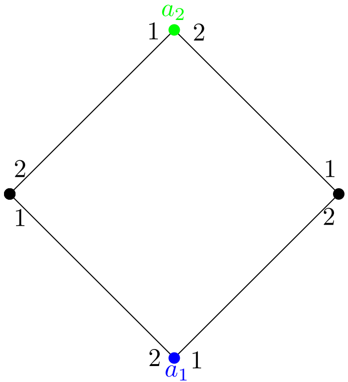
Deterministic Algorithm

The route (sequence of port numbers) followed by an agent only depends on :

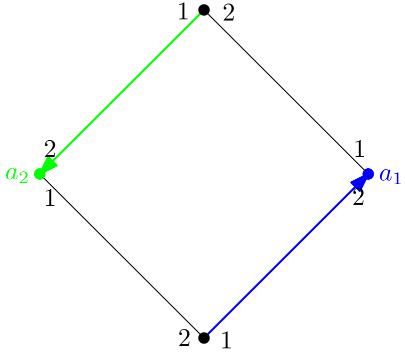
- the environment : the starting position of the agent and the graph (more precisely the part of the graph that the agent learned up to date)
- the identifier of the agent

Agent's identifier is required for deterministic model
$\Rightarrow$ Without identifiers, deterministic agents never meet in a ring because of symmetry.

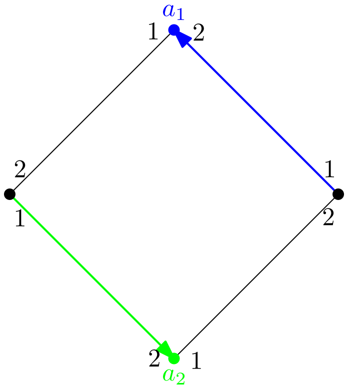
Deterministic rendezvous impossible without identifier



Deterministic rendezvous impossible without identifier



Deterministic rendezvous impossible without identifier



Total knowledge of the agents

Rendezvous in finite graph known to the agents [1]

Rendezvous algorithm at cost $\Theta(D|L_{\min}|)$ if each of the agents knows the graph, its starting position and the starting position of the other agent.

$|L_{\min}|$: size in bits of the smaller of the two identifiers of the agents
D: distance between the two starting positions of the agents

[1] G. De Marco, L. Gargano, E. Kranakis, D. Krizanc, A. Pelc, U. Vaccaro, Asynchronous deterministic rendezvous in graphs, Theoretical Computer Science 355 (2006), 315-326.

Finite graphs partially known by the agents

Rendezvous in a graph partially known by the agents [1]

Rendezvous algorithm (cost exponential in the size of the graph) if the size of the graph is known by the agent.

[1] G. De Marco, L. Gargano, E. Kranakis, D. Krizanc, A. Pelc, U. Vaccaro, Asynchronous deterministic rendezvous in graphs, Theoretical Computer Science 355 (2006), 315-326.

The rendezvous algorithm for an unknown graph

General idea of the algorithm

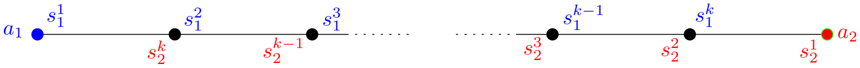
- Each configuration is a quadruple consisting of two agents' identifiers and two sequences of ports potentially traversed by the routes of both agents
- Enumerate all the configurations of the algorithm
- Iteratively, for any subsequent configuration in the enumeration, **if**
    - the quadruple contains your identifier, and
    - your route corresponds to a valid path in the graph, say from node v to w
    - the second route corresponds to the reverse path from w to v
- **then** extend the route to form a tunnel with the other route

Rendezvous algorithm

Initial configuration : quadruple (L_1, S_1, L_2, S_2)

- S_1, S_2 : two sequences of integers of same length.
- L_1, L_2 : identifiers of the two robots.

S_1 and S_2 correspond to sequence of ports number of a path linking the two starting positions of the robots.



$$S_1 = (s_1^1, s_1^2, s_1^3, \ldots, s_1^{k-1}, s_1^k)$$
$$S_2 = (s_2^1, s_2^2, s_2^3, \ldots, s_2^{k-1}, s_2^k)$$

Pseudo-code of the algorithm

Rendezvous algorithm

For each quadruple $\varphi_k = (i, S_1, j, S_2)$ **Do**

  If *identifier*(Agent) = i **Then**

    follow port s_1^1 then s_1^2 ... then s_1^k

    If $(\forall i \leq k, s_2^{k+1-i}$ is the outgoing port of $s_1^i)$**Then**

      execute route of robot j until quadruple φ_{k-1}

      extend the route of the robot i such that the routes of robots i et j form a tunnel

    return to the starting point

  If Identifier(Agent) = j **Then**

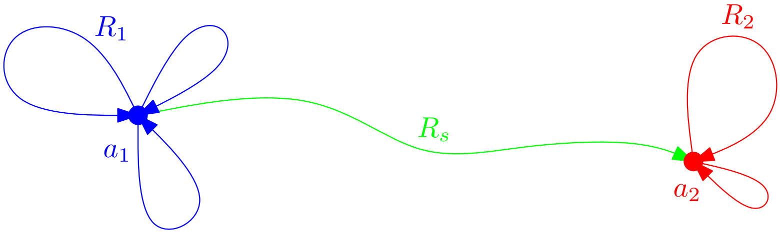
    Do the same with S_2 instead of S_1 and j instead of i.

Execution of the step of the main loop corresponding to the initial configuration of the robots

Route of agent a_1: R_1

Route of agent a_2: R_2

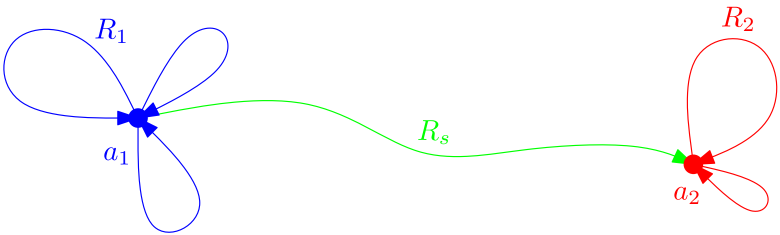
Execution of the step of the main loop corresponding to the initial configuration of the robots



Route of agent a_1: $R_1 + R_s$

Route of agent a_2: $R_2 + R_s^{-1}$

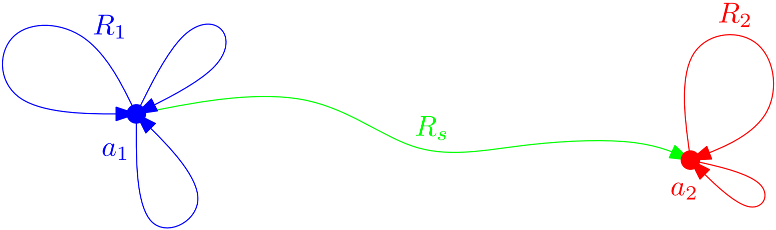
Execution of the step of the main loop corresponding to the initial configuration of the robots



Route of agent a_1: $R_1 + R_s + R_2$

Route of agent a_2: $R_2 + R_s^{-1} + R_1$

Execution of the step of the main loop corresponding to the initial configuration of the robots



Route of agent a_1: $R_1 + R_s + R_2 + R_s^{-1} + R_1^{-1} + R_s + R_2^{-1}$

Route of agent a_2: $R_2 + R_s^{-1} + R_1 + R_s + R_2^{-1} + R_s^{-1} + R_1^{-1}$

From polygonal terrains to graphs

Robots with rational coordinates in any closed terrain

- Consider an (infinite) graph G having as nodes all the points of the given terrain T having rational coordinates.
- The edges of G are segments belonging to T
- Solving the rendezvous problem for agents in G implies solving the rendezvous problems for robots in T (at rational initial coordinates)

Our results: Cz., Labourel, Pelc, SODA2010

- **Thm:** There is an algorithm which guarantees asynchronous rendezvous for arbitrary two agents starting from arbitrary nodes of any connected graph (also infinite).
- **Thm:** There is an algorithm which guarantees asynchronous rendezvous for arbitrary two robots starting from arbitrary rational interior points of any closed terrain T with path-connected interior.
- **Thm:** There is an algorithm which guarantees ϵ-approximate rendezvous for any $\epsilon > 0$, for arbitrary robots starting from arbitrary interior points of any closed terrain T with path-connected interior.
- **Prop:** There is no algorithm that guarantees asynchronous rendezvous of arbitrary robots with point visibility, starting from arbitrary points in the plane.

Cost of the rendezvous exponential or polynomial?

Open problems

- Does there exist an asynchronous deterministic algorithm for the class of finite graphs such that the cost of rendezvous is polynomial in the size of the graph?
- Does there exist an asynchronous deterministic rendezvous algorithm such that the cost of rendezvous is polynomial in the original distance between the initial positions of the agents?