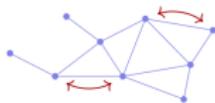


Distributed algorithm

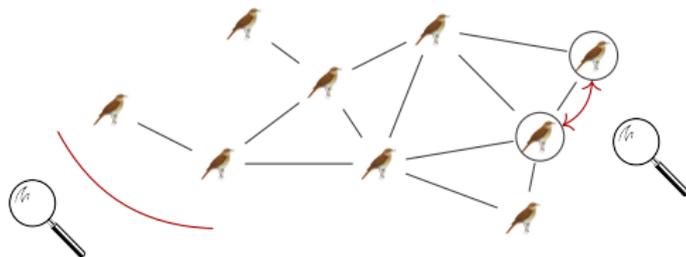
Arnaud Casteigts



Part of Graph Algorithms (14X061)
Masters of Computer Science

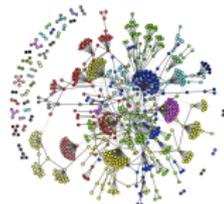
University of Geneva

Studying networks



Network as **input**

→ **centralized** algorithms...



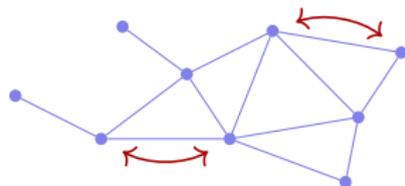
Network as **environment**

→ **decentralized** algorithms...
(a.k.a. distributed)



Distributed Algorithms

(Think globally, act locally)



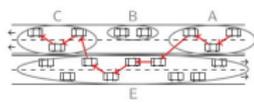
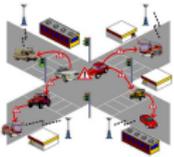
Collaboration of distinct entities to perform a common task.

No centralization available, interactions among neighbors.

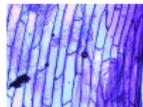
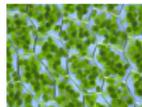
Theoretical aspects of collective intelligence.

The world is dynamic...

In technologies

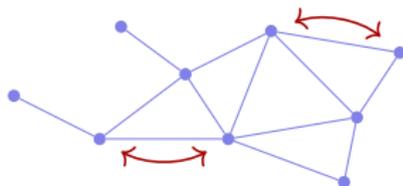


In nature



Distributed Algorithms

(Think globally, act locally)



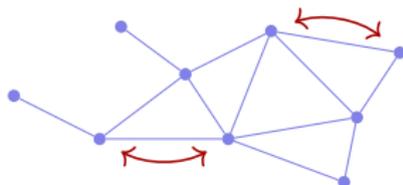
Collaboration of distinct entities to perform a common task.

No centralization available, interactions among neighbors.

Theoretical aspects of collective intelligence.

Distributed Algorithms

(Think globally, act locally)



Collaboration of distinct entities to perform a common task.

No centralization available, interactions among neighbors.

Theoretical aspects of collective intelligence.

Examples of problems:

Broadcast



Election



Spanning tree



Counting



Consensus, naming, routing, exploration, coloring, dominating sets, ...

Two basic examples

Broadcasting algorithm:

```
onSelection(): // selected node  
  informed  $\leftarrow$  true
```

```
onStart(): // all nodes  
  if (informed)  
    sendAll(new Message())
```

```
onMessage(message):  
  if ( $\neg$  informed)  
    informed  $\leftarrow$  true  
    sendAll(message)
```

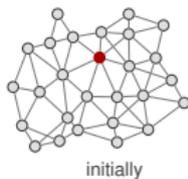
Two basic examples

Broadcasting algorithm:

```
onSelection(): // selected node
  informed  $\leftarrow$  true

onStart(): // all nodes
  if (informed)
    sendAll(new Message())

onMessage(message):
  if ( $\neg$  informed)
    informed  $\leftarrow$  true
    sendAll(message)
```



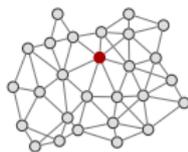
Two basic examples

Broadcasting algorithm:

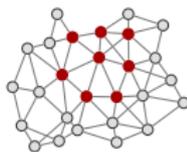
```
onSelection(): // selected node
  informed  $\leftarrow$  true

onStart(): // all nodes
  if (informed)
    sendAll(new Message())

onMessage(message):
  if ( $\neg$  informed)
    informed  $\leftarrow$  true
    sendAll(message)
```



initially

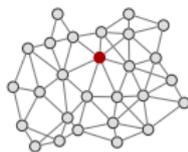


a bit later

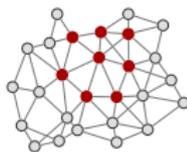
Two basic examples

Broadcasting algorithm:

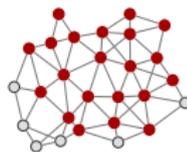
```
onSelection(): // selected node  
  informed  $\leftarrow$  true  
onStart(): // all nodes  
  if (informed)  
    sendAll(new Message())  
onMessage(message):  
  if ( $\neg$  informed)  
    informed  $\leftarrow$  true  
    sendAll(message)
```



initially



a bit later

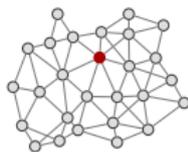


later

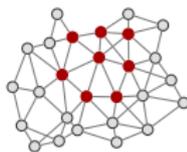
Two basic examples

Broadcasting algorithm:

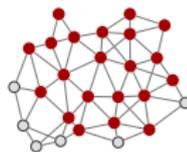
```
onSelection(): // selected node  
  informed  $\leftarrow$  true  
onStart(): // all nodes  
  if (informed)  
    sendAll(new Message())  
onMessage(message):  
  if ( $\neg$  informed)  
    informed  $\leftarrow$  true  
    sendAll(message)
```



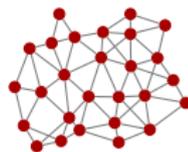
initially



a bit later



later



finally

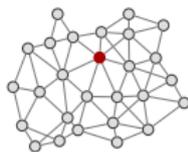
Two basic examples

Broadcasting algorithm:

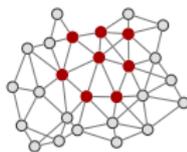
```
onSelection(): // selected node
  informed  $\leftarrow$  true

onStart(): // all nodes
  if (informed)
    sendAll(new Message())

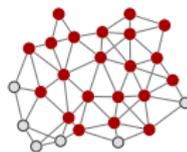
onMessage(message):
  if ( $\neg$  informed)
    informed  $\leftarrow$  true
    sendAll(message)
```



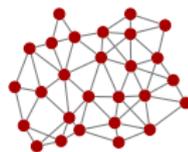
initially



a bit later



later



finally

Spanning tree construction:

```
onSelection(): // selected node
  root  $\leftarrow$  true

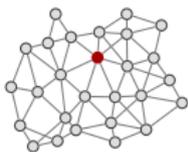
onStart(): // all nodes
  if (root)
    sendAll(message)

onMessage(message):
  if (parent == null)
    parent  $\leftarrow$  message.sender
    sendAll(message)
```

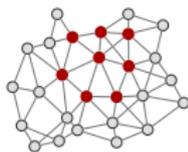
Two basic examples

Broadcasting algorithm:

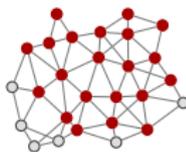
```
onSelection(): // selected node
  informed ← true
onStart(): // all nodes
  if (informed)
    sendAll(new Message())
onMessage(message):
  if (¬ informed)
    informed ← true
    sendAll(message)
```



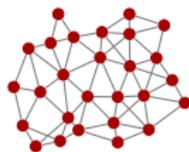
initially



a bit later



later



finally

Spanning tree construction:

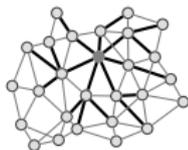
```
onSelection(): // selected node
  root ← true
onStart(): // all nodes
  if (root)
    sendAll(message)
onMessage(message):
  if (parent == null)
    parent ← message.sender
    sendAll(message)
```



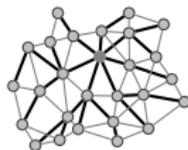
initially



a bit later



later



finally

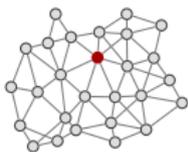
Two basic examples

Broadcasting algorithm:

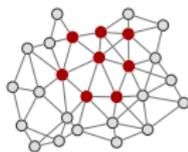
```
onSelection(): // selected node
  informed ← true

onStart(): // all nodes
  if (informed)
    sendAll(new Message())

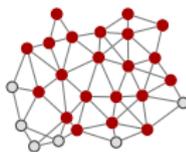
onMessage(message):
  if (¬ informed)
    informed ← true
    sendAll(message)
```



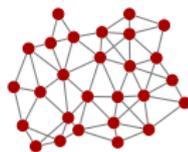
initially



a bit later



later



finally

Spanning tree construction:

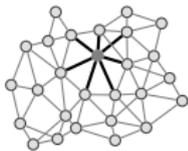
```
onSelection(): // selected node
  root ← true

onStart(): // all nodes
  if (root)
    sendAll(message)

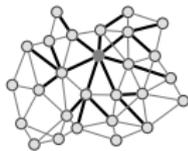
onMessage(message):
  if (parent == null)
    parent ← message.sender
    sendAll(message)
```



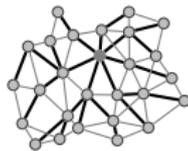
initially



a bit later



later



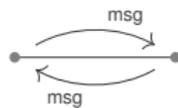
finally

Everybody has the same algorithm, but initial states may differ (**uniform** versus **non-uniform** initialization).

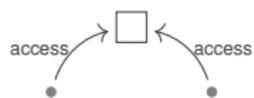
→ Various *modeling assumptions*

What assumptions?

Communication model:



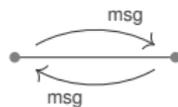
Message passing



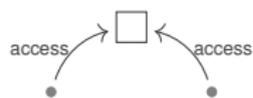
Shared memory

What assumptions?

Communication model:



Message passing

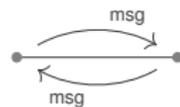


Shared memory

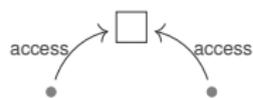
Other models : link registers, mobile agents, graph relabelings, (in nature) sounds, vision, smell, tactile, etc.

What assumptions?

Communication model:



Message passing



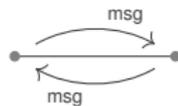
Shared memory

Other models : link registers, mobile agents, graph relabelings, (in nature) sounds, vision, smell, tactile, etc.

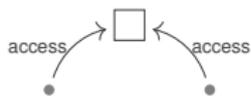
→ Most frequent in networks: **Message passing**.

What assumptions?

Communication model:



Message passing



Shared memory

Other models : link registers, mobile agents, graph relabelings, (in nature) sounds, vision, smell, tactile, etc.

→ Most frequent in networks: **Message passing**.

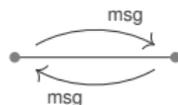
Modeling of the network:

→ Directed or undirected **graph** whose edges are communication links.

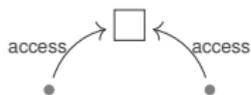


What assumptions?

Communication model:



Message passing



Shared memory

Other models : link registers, mobile agents, graph relabelings, (in nature) sounds, vision, smell, tactile, etc.

→ Most frequent in networks: **Message passing**.

Modeling of the network:

→ Directed or undirected **graph** whose edges are communication links.



Other assumptions:

- Unique **identifiers** versus **anonymous** networks.
- **Uniform** versus **non-uniform** initialization (e.g. distinguished node, like in the previous slide).
- **Synchronous** versus **asynchronous** (are the rounds of communication synchronized?).

First algorithm in literature

Election in a **directed cycle**, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

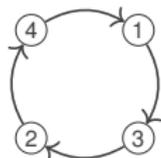
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){  
  send(otherID)  
}else{  
  if (otherID == myID){  
    ELECTED ← true  
  }  
}
```



First algorithm in literature

Election in a **directed cycle**, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

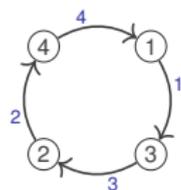
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){
  send(otherID)
}else{
  if (otherID == myID){
    ELECTED ← true
  }
}
```



round 1

First algorithm in literature

Election in a **directed cycle**, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

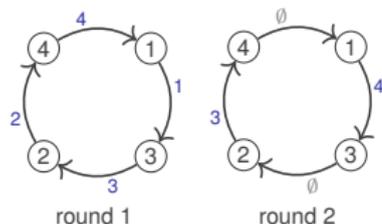
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){  
  send(otherID)  
}else{  
  if (otherID == myID){  
    ELECTED ← true  
  }  
}
```



First algorithm in literature

Election in a **directed cycle**, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

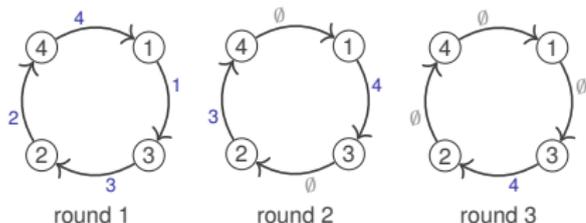
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){
  send(otherID)
}else{
  if (otherID == myID){
    ELECTED ← true
  }
}
```



First algorithm in literature

Election in a directed cycle, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

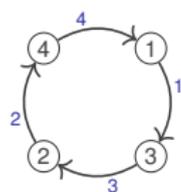
The node with the highest identifier gets elected.

onStart():

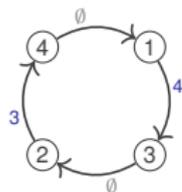
```
send(myID)
```

onMessage(otherID):

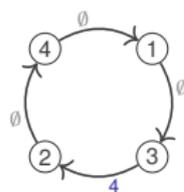
```
if (otherID > myID){  
  send(otherID)  
}else{  
  if (otherID == myID){  
    ELECTED ← true  
  }  
}
```



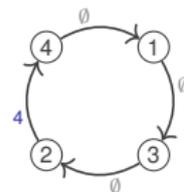
round 1



round 2



round 3



round 4

First algorithm in literature

Election in a directed cycle, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

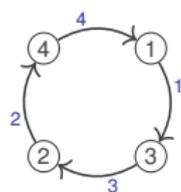
The node with the highest identifier gets elected.

onStart():

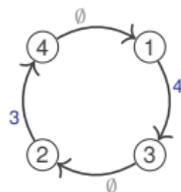
```
send(myID)
```

onMessage(otherID):

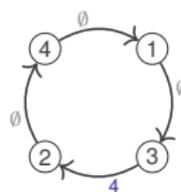
```
if (otherID > myID){
  send(otherID)
}else{
  if (otherID == myID){
    ELECTED ← true
  }
}
```



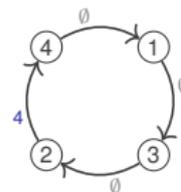
round 1



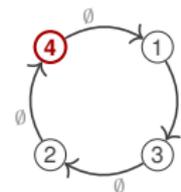
round 2



round 3



round 4



round 5

First algorithm in literature

Election in a directed cycle, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

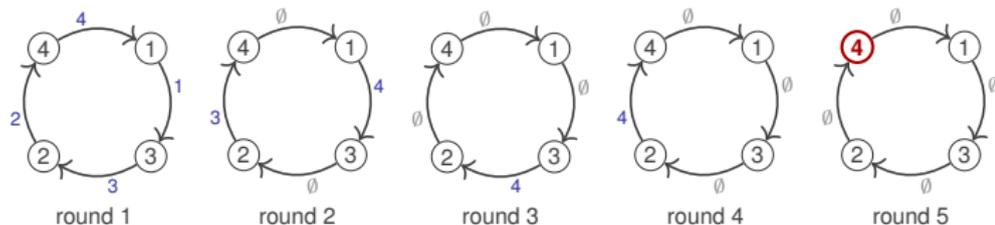
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){  
  send(otherID)  
}else{  
  if (otherID == myID){  
    ELECTED ← true  
  }  
}
```



Worst-case time complexity?

Worst-case message complexity?

First algorithm in literature

Election in a directed cycle, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

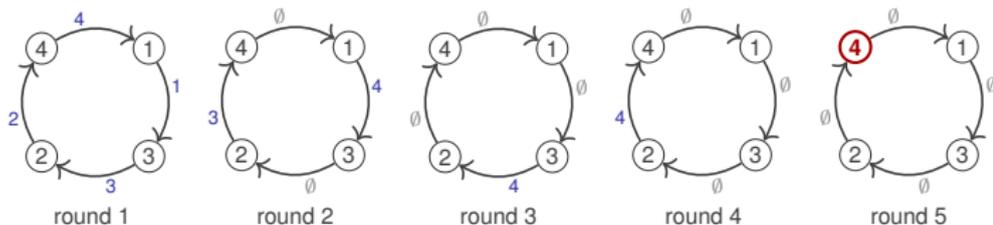
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){
  send(otherID)
}else{
  if (otherID == myID){
    ELECTED ← true
  }
}
```



Worst-case time complexity?

Worst-case message complexity?

$\Theta(n)$

First algorithm in literature

Election in a directed cycle, with **synchronous communication**, **unique identifiers**, and **uniform** initialization.
(Le Lann, Chang and Roberts, 1970's)

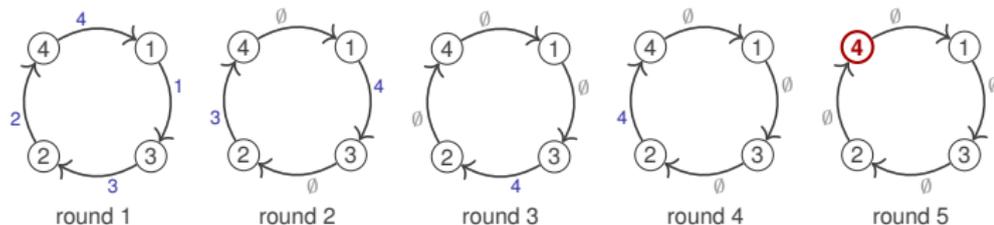
The node with the highest identifier gets elected.

onStart():

```
send(myID)
```

onMessage(otherID):

```
if (otherID > myID){
  send(otherID)
}else{
  if (otherID == myID){
    ELECTED ← true
  }
}
```



Worst-case time complexity?

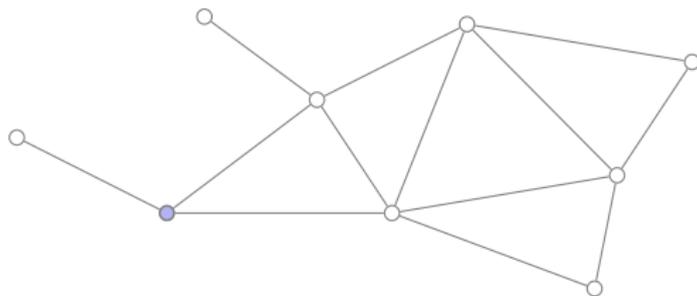
$\Theta(n)$

Worst-case message complexity?

$\Theta(n^2)$

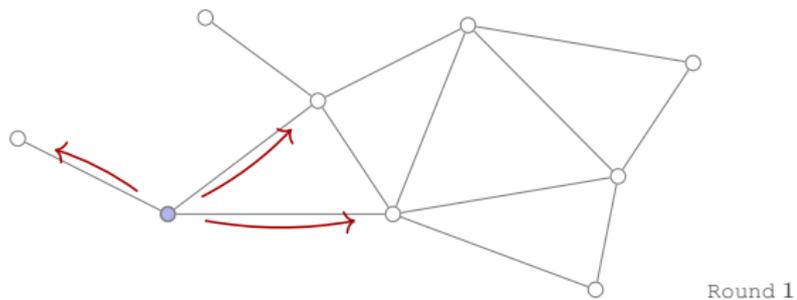
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



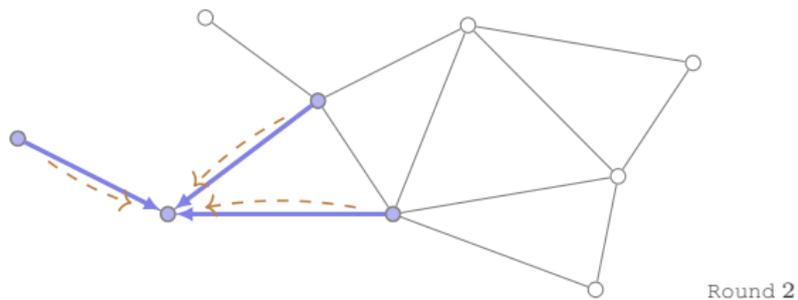
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



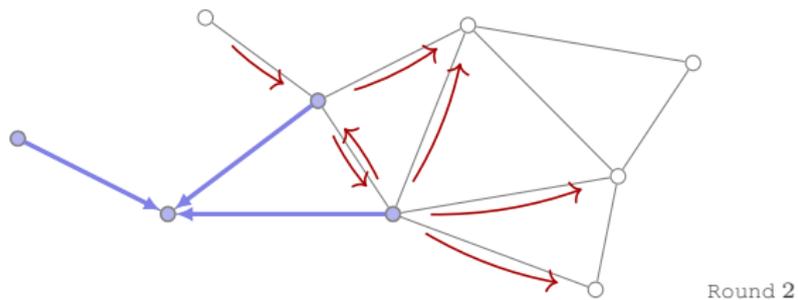
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



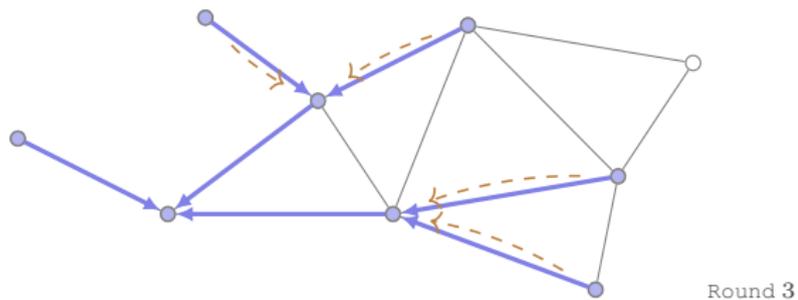
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



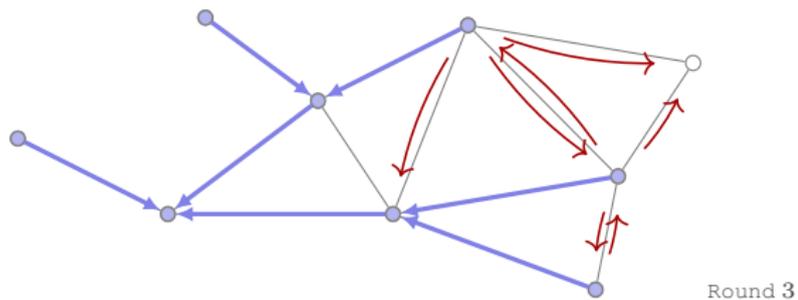
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



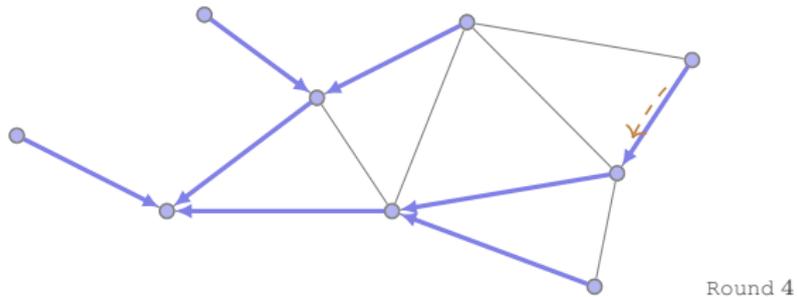
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



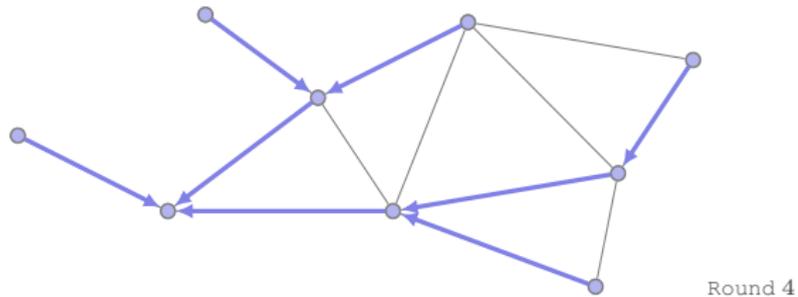
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



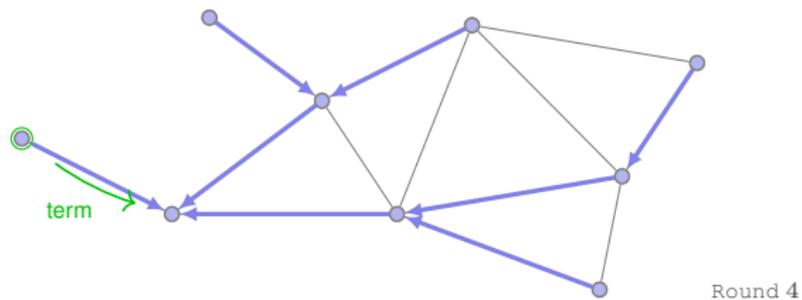
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



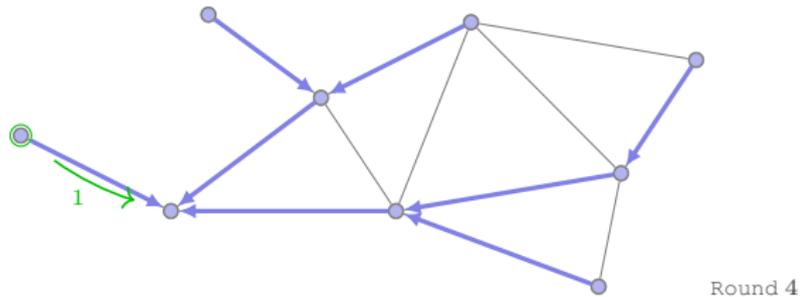
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



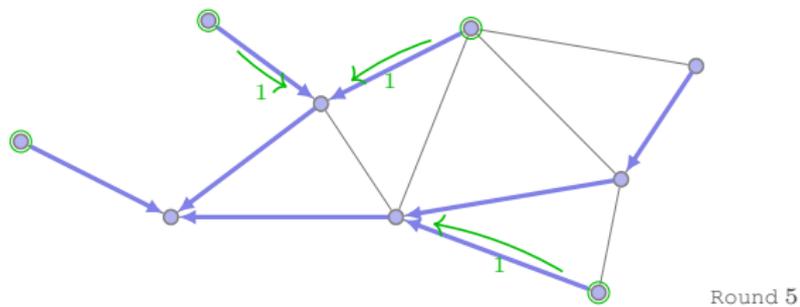
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



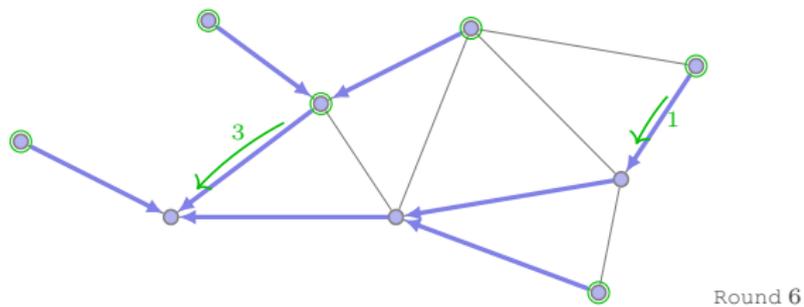
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



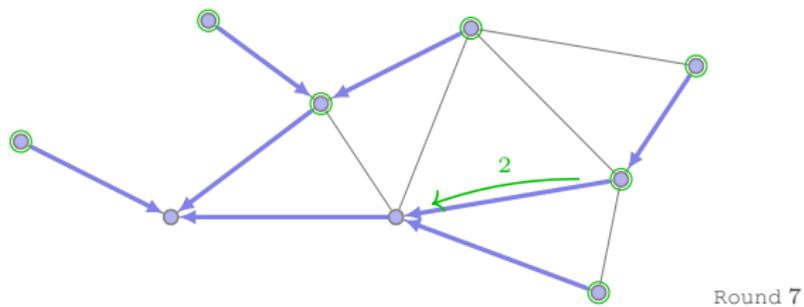
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



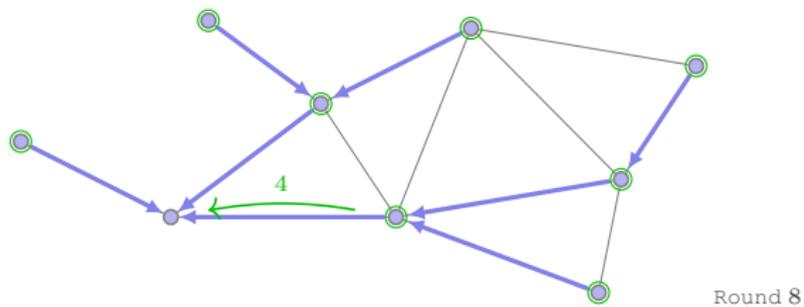
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



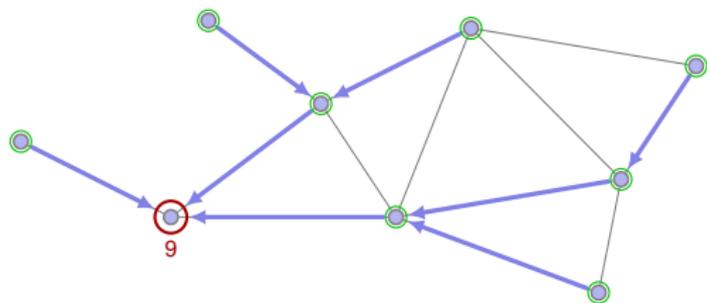
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



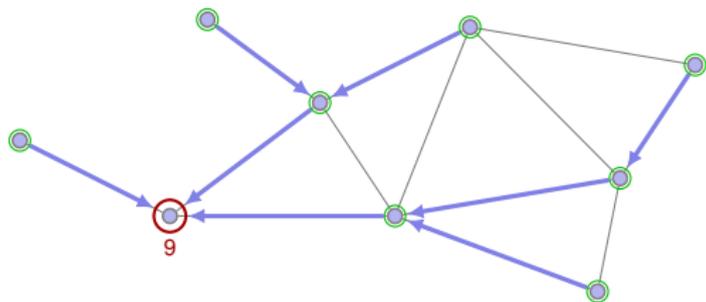
Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



Time complexity: $O(\text{diameter}(G)) = O(n)$

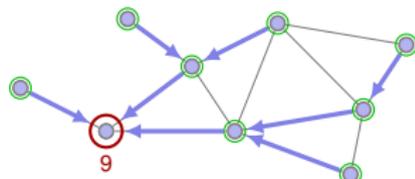
Message complexity: $O(m + n + n) = O(m)$

n : #nodes

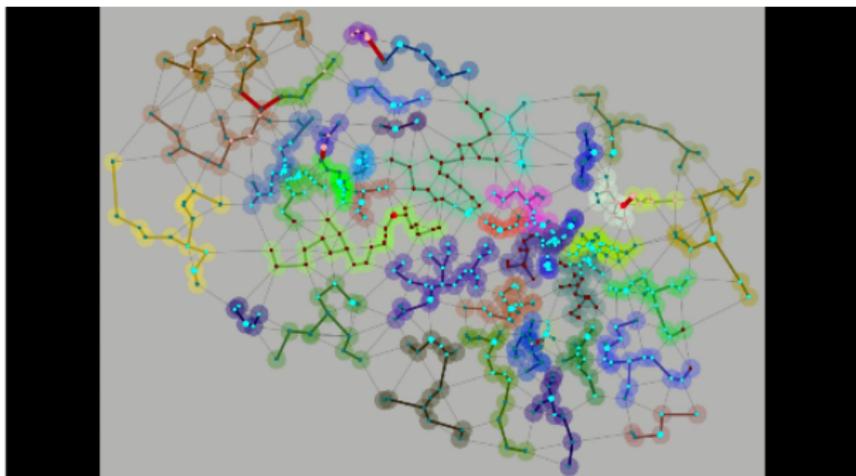
m : #edges

Complex example: Broadcasting | Spanning tree | Counting

Assumptions: synchronous communication / unique ids / distinguished node



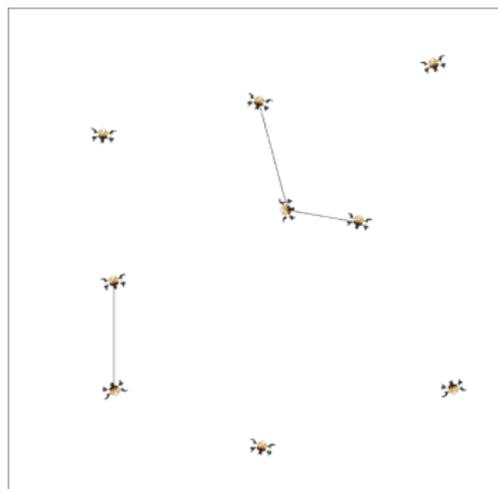
What if there are no distinguished nodes? (The GHS algorithm, 1983)



(Highly) dynamic networks?



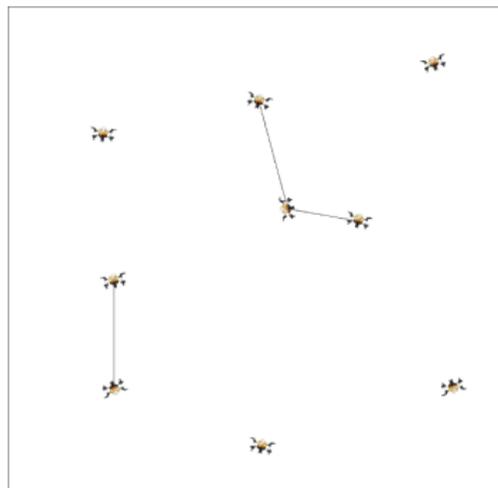
Example of scenario



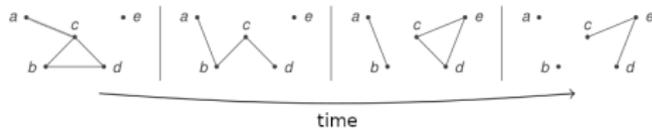
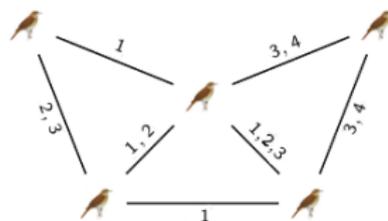
(Highly) dynamic networks?



Example of scenario



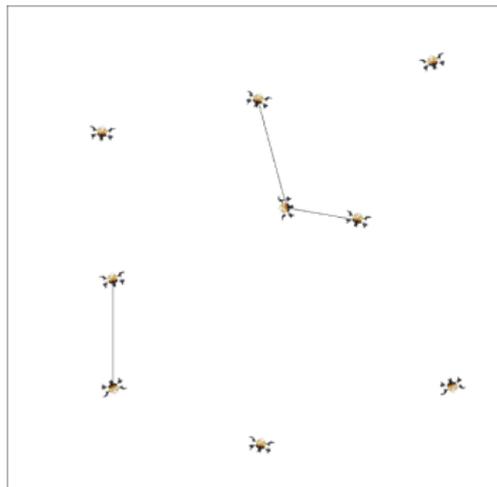
Modeling



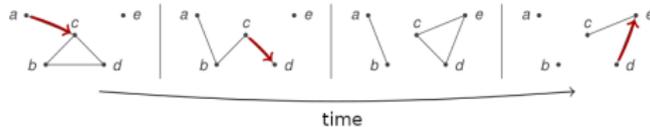
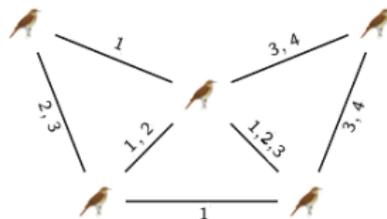
(Highly) dynamic networks?



Example of scenario



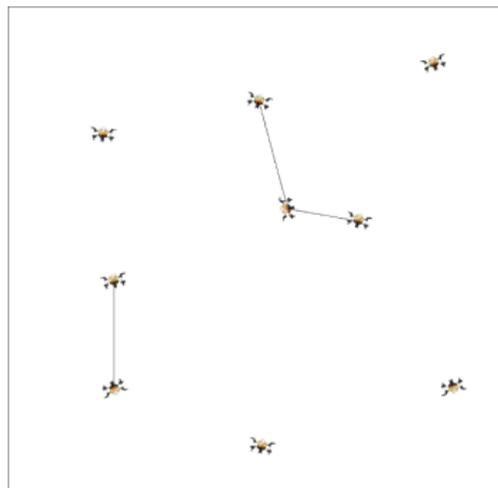
Modeling



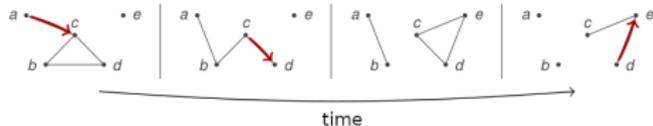
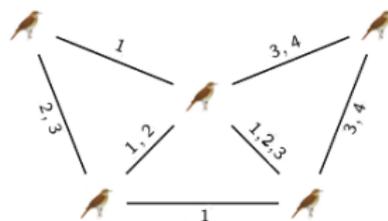
(Highly) dynamic networks?



Example of scenario



Modeling



Properties:

- ▶ Temporal connectivity?
- ▶ Repeatedly?
- ▶ Recurrent links?
- ▶ In bounded time?
- ▶ ...

\mathcal{TC}

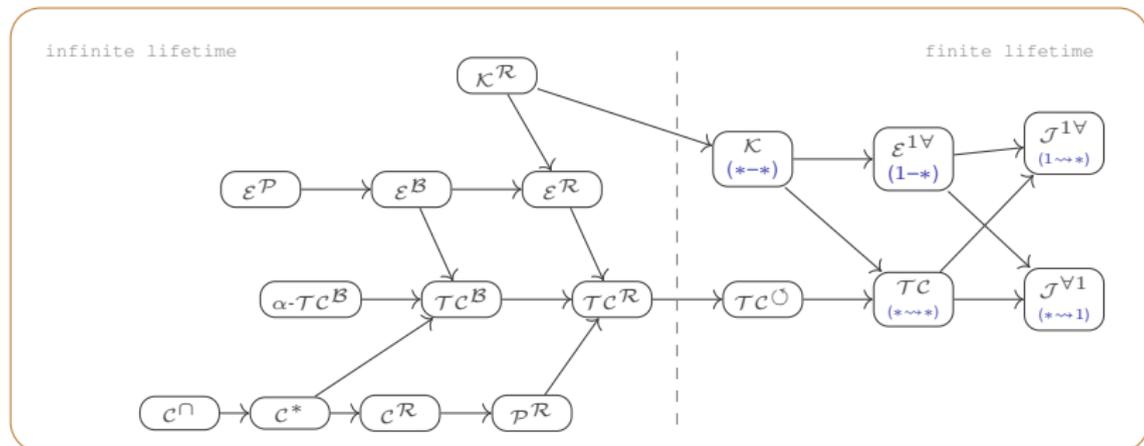
$\mathcal{TC}^{\mathcal{R}}$

$\mathcal{E}^{\mathcal{R}}$

$\mathcal{E}^{\mathcal{B}}$

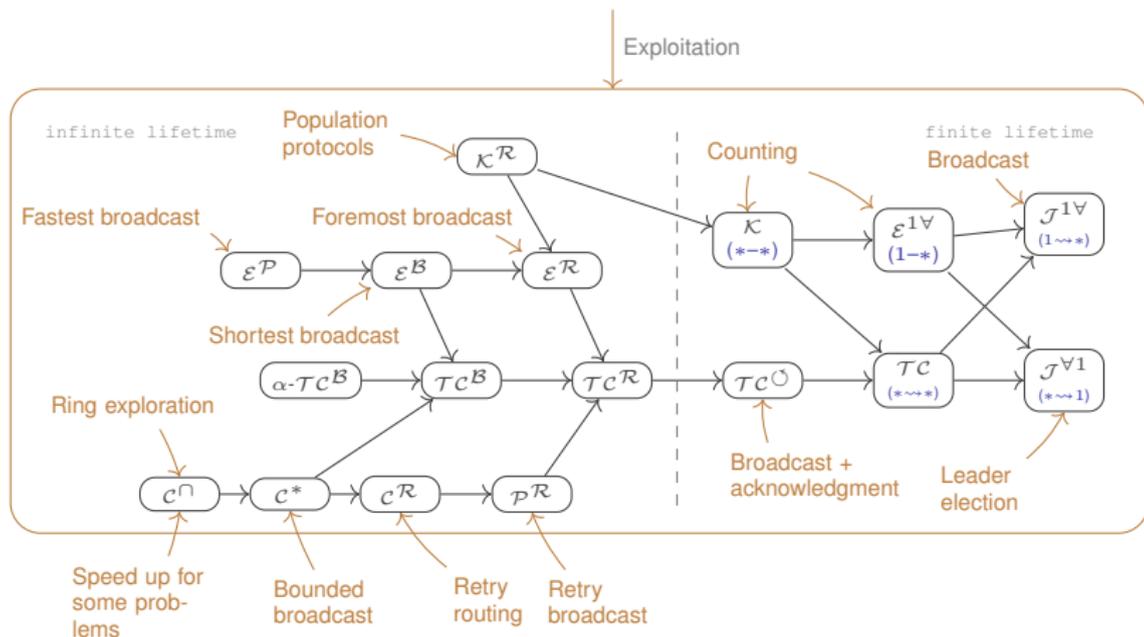
→ Classes of temporal graphs

Some classes of temporal graphs



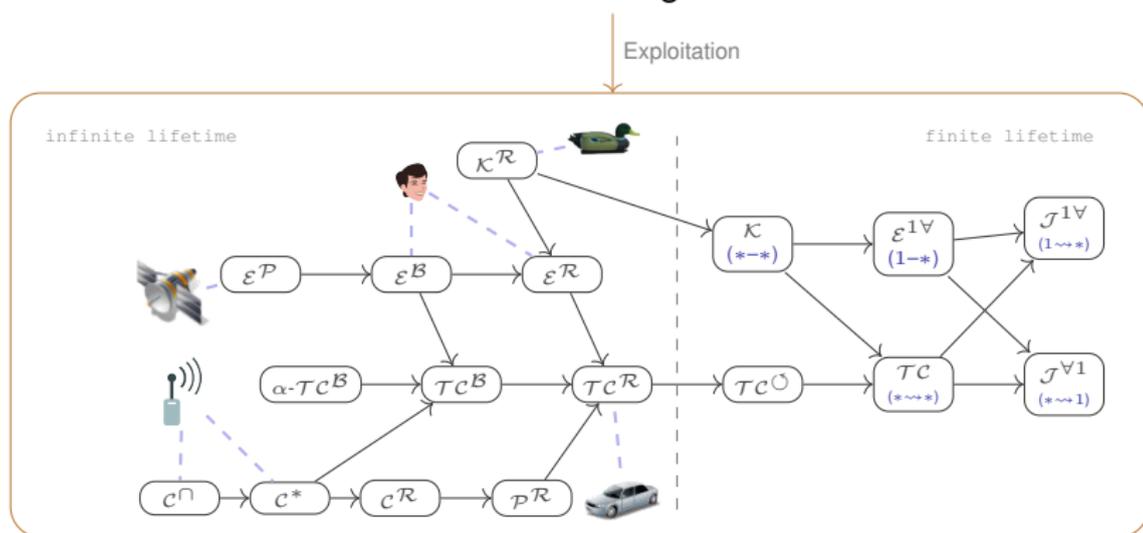
Some classes of temporal graphs

Distributed algorithm



Some classes of temporal graphs

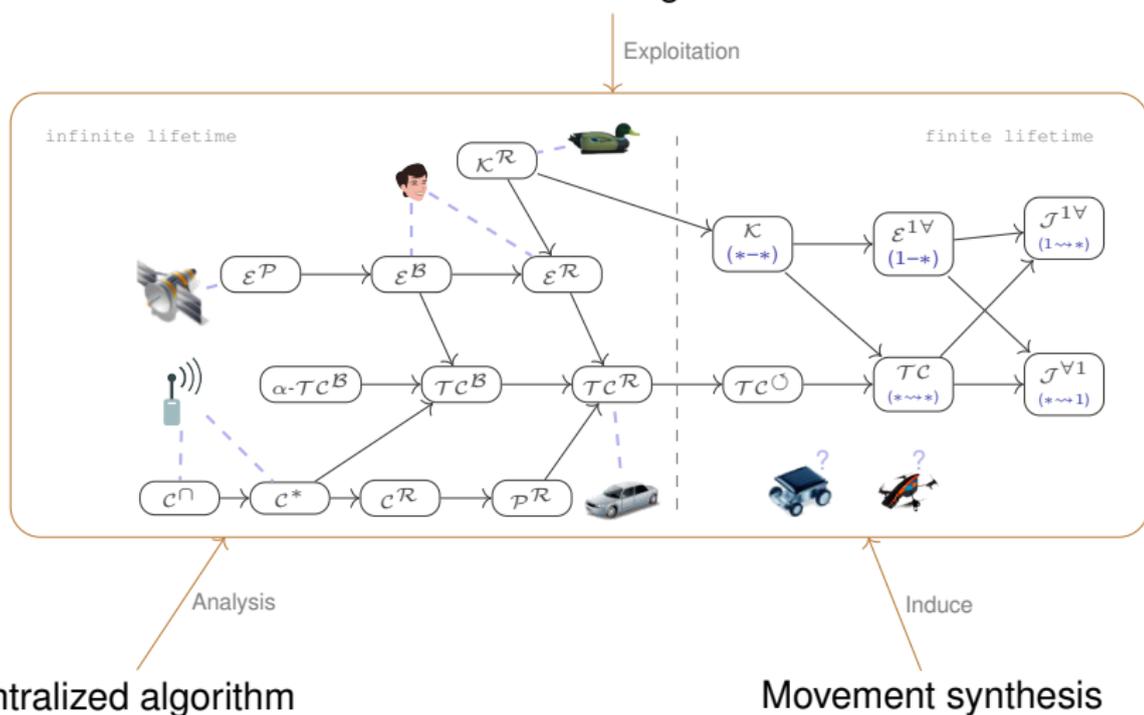
Distributed algorithm



Centralized algorithm

Some classes of temporal graphs

Distributed algorithm



Tool for the exercise session

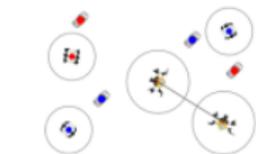
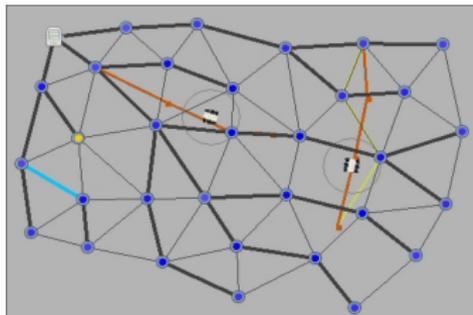
JBOTSIM: Collective intelligence / Network algorithms / Motion planning / Robotics



(h) Acceleration constraints



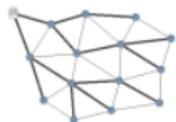
(k) Obstacle management



(i) Heterogeneous park cleaning



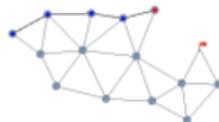
(o) Navigation (embedding)



(a) Data aggregation



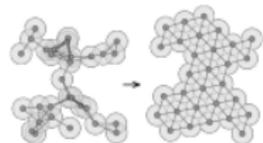
(b) Vehicular networks



(c) Geographical routing



(g) Territory sharing



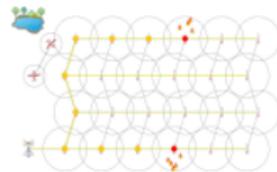
(d) Deployment by virtual forces



(e) Travelling Salesman Problem



(f) Toroidal space



(j) Fire-fighting aircrafts