Dribble: A learn-based timer scheme selector for mobility management in IoT

Authors:
Bruno P. Santos,
Paulo H. L. Rettore,
Luiz F. M. Vieira,
Antonio A. F. Loureiro

UFMG
UNIVERSIDADE FEDERAL DE MINAS GERAIS

UFOP
Universidade Federal de Ouro Preto

Agenda

1. Introduction
2. Background
3. Dribble
4. Evaluation
5. Conclusion and Future work

Contextualization

- Smart devices are part of our daily routine

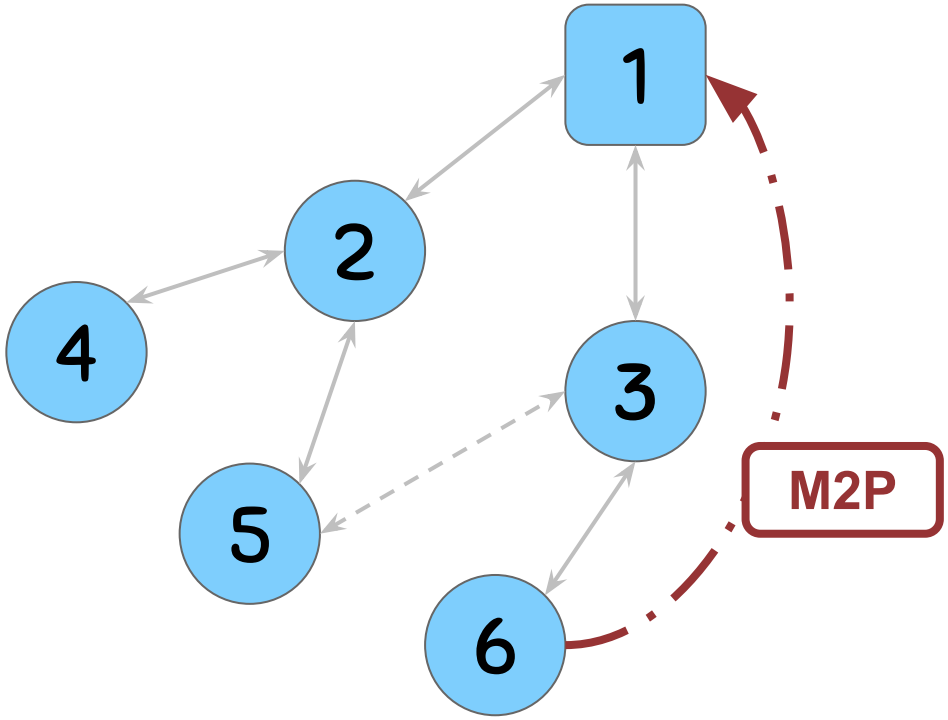
Mobility is a key challenge!

- IoT - Challenges
  - Internet adaptations
  - Heterogeneous devices
  - Constrained resources (Energy, CPU, Memory…)
  - Mobility

Mobility is a key challenge!

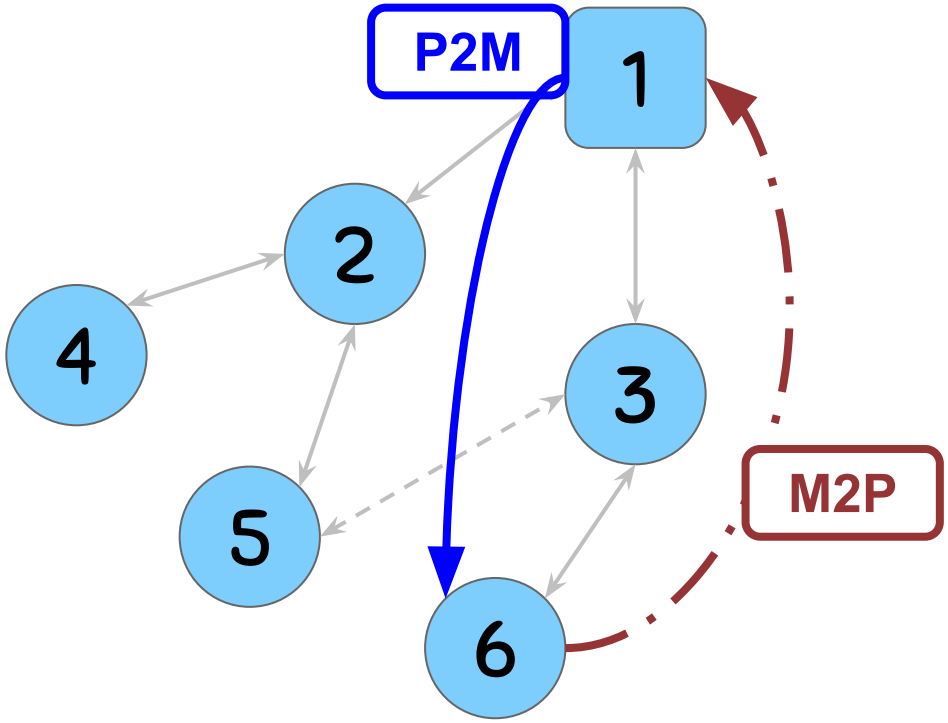
- IoT - Challenges
  - Internet adaptations
  - Heterogeneous devices
  - Constrained resources (Energy, CPU, Memory…)
  - **Mobility**
- We are interested in handle Mobility
  - Key aspect for mobile and wireless environment
  - Mobility from routing protocol lens

IoT routing in a nutshell



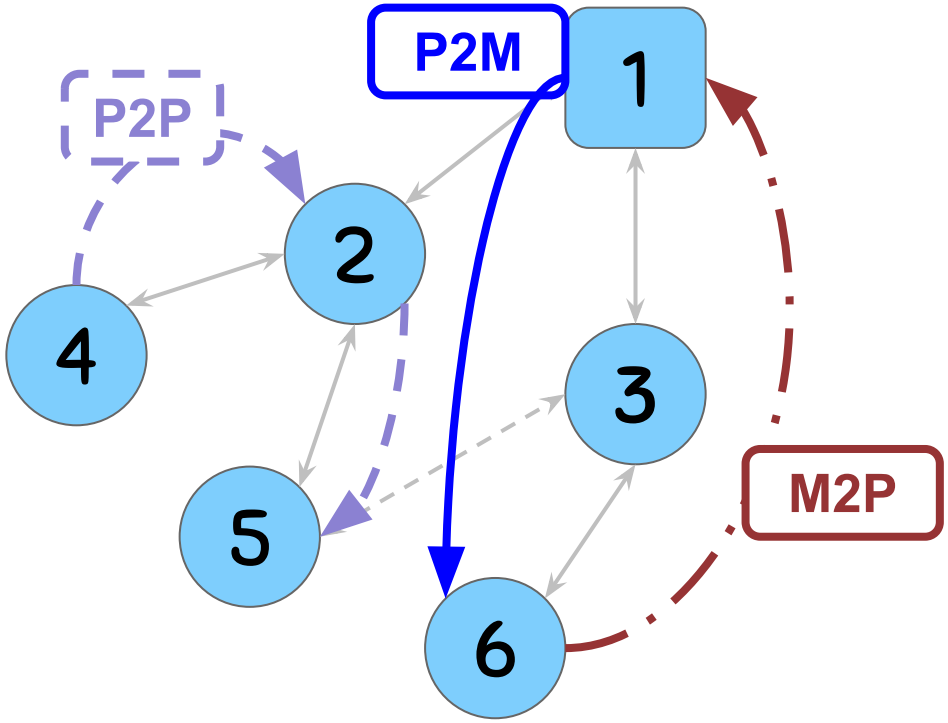
Data traffic patterns
over routing structures

IoT routing in a nutshell



Data traffic patterns
over routing structures

IoT routing in a nutshell



Data traffic patterns
over routing structures

IoT routing in a nutshell

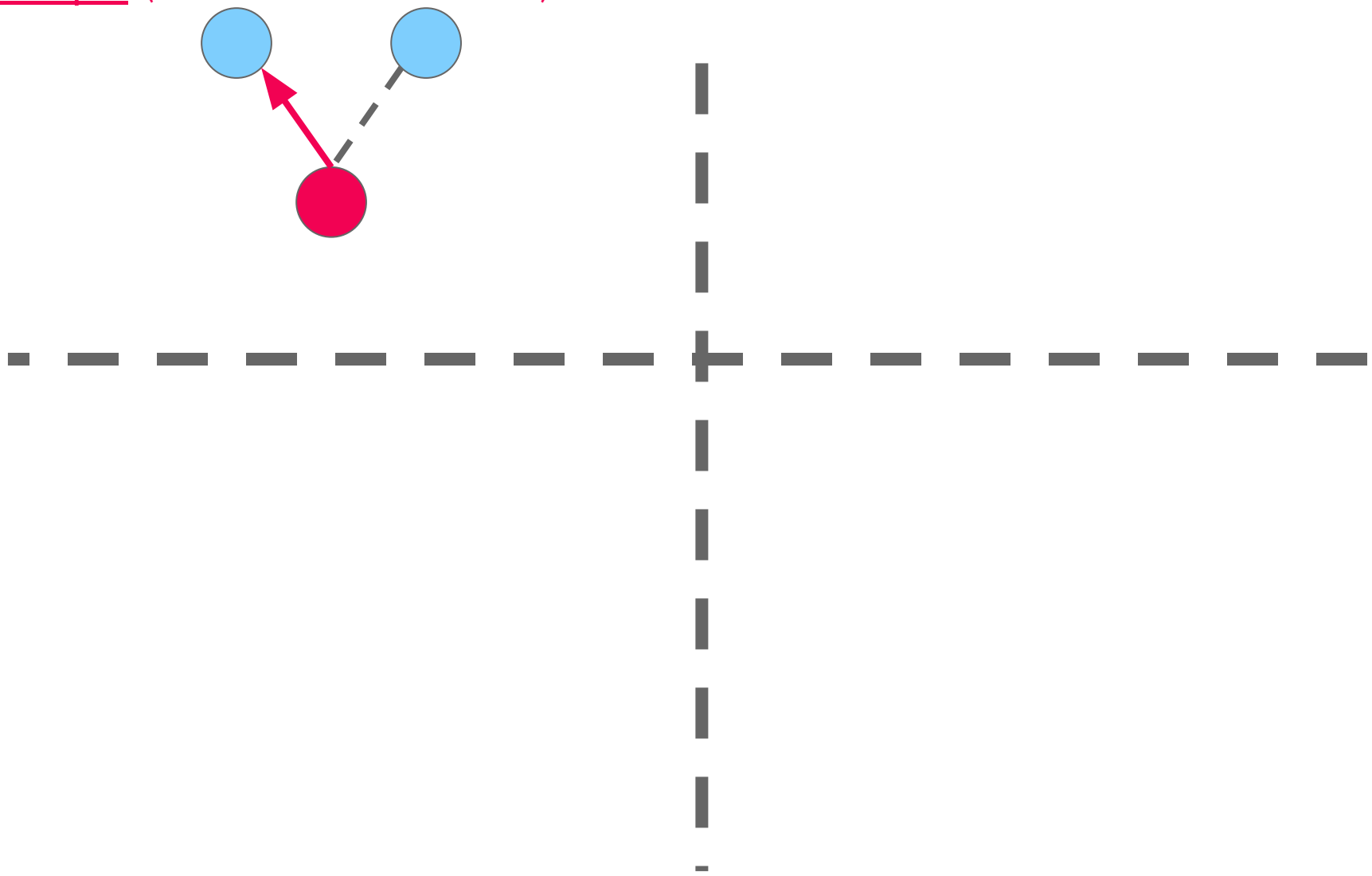
- Literature routing protocols
  - RPL (*de facto* the state-of-the-art)
    - Several RPL adaptations for mobile scenarios
      - Co-RPL, MRPL, MMRPL, ERPL...
  - Mobile Matrix
  - Hydro
  - XCTP
  - ...

Routing under mobility events

- Mostly of routing protocols for mobile IoT have one timer scheme
  - It governs the communication structure construction and maintenance

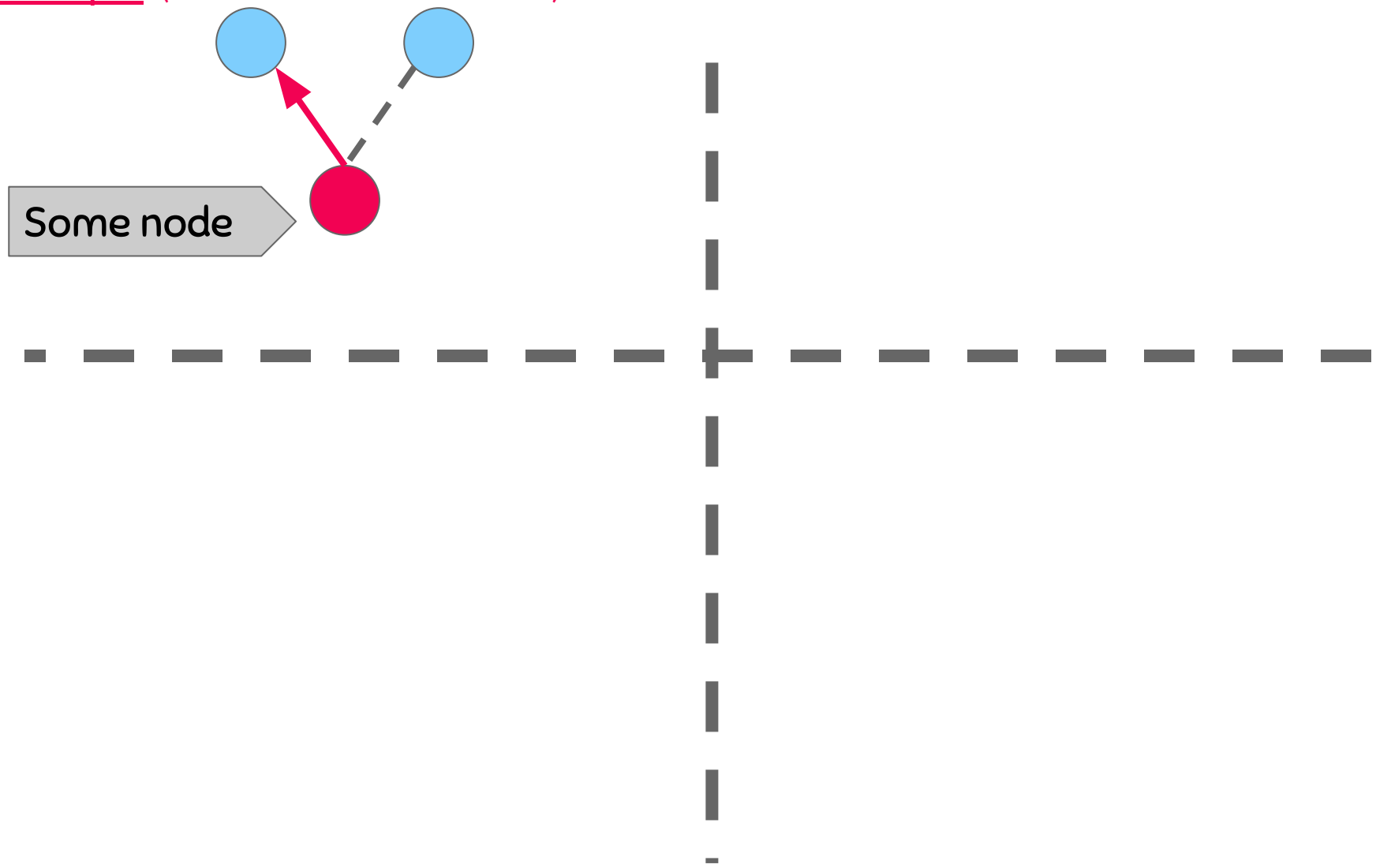
Routing under mobility events

Example. (note there are other solutions)



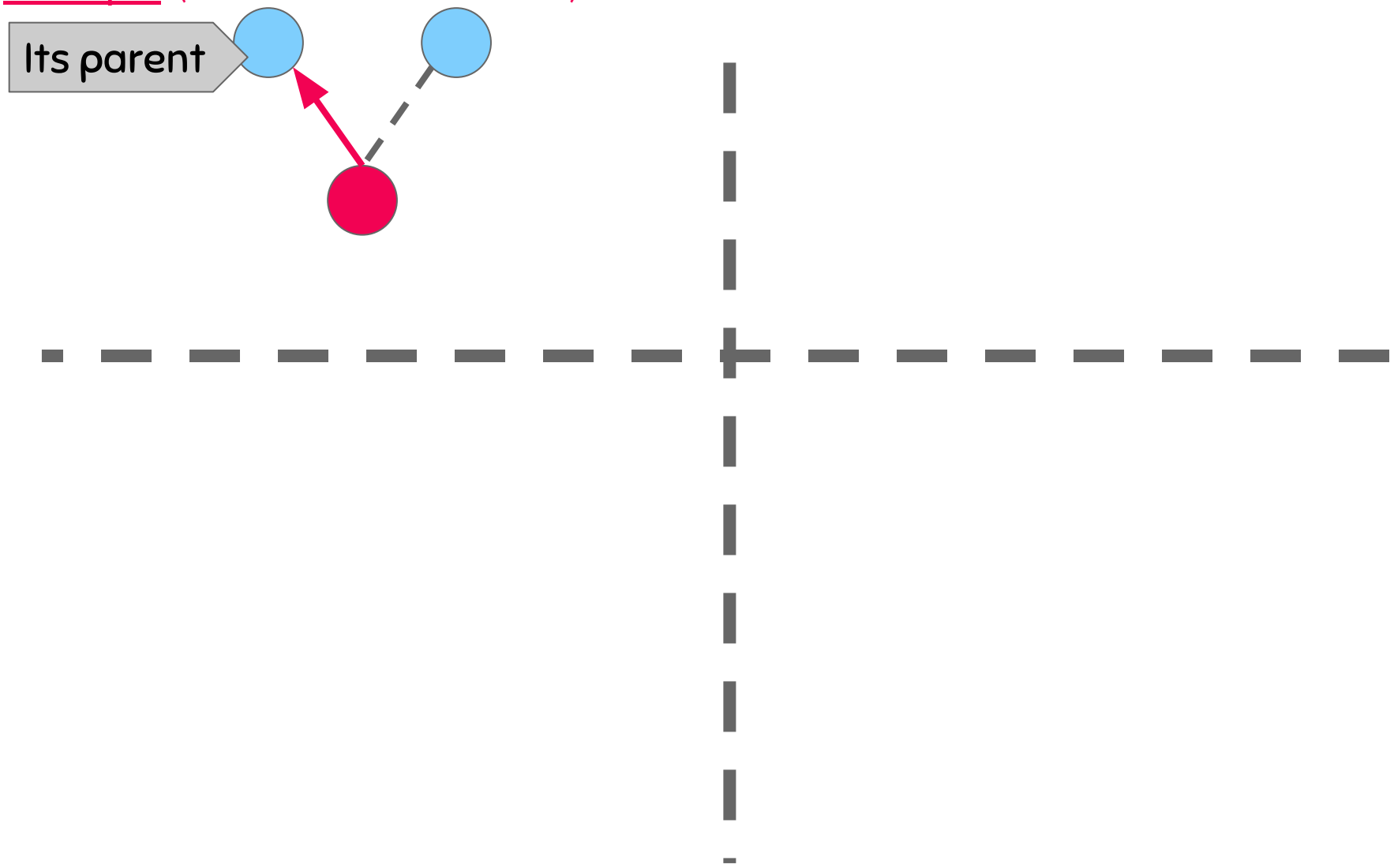
Routing under mobility events

Example. (note there are other solutions)



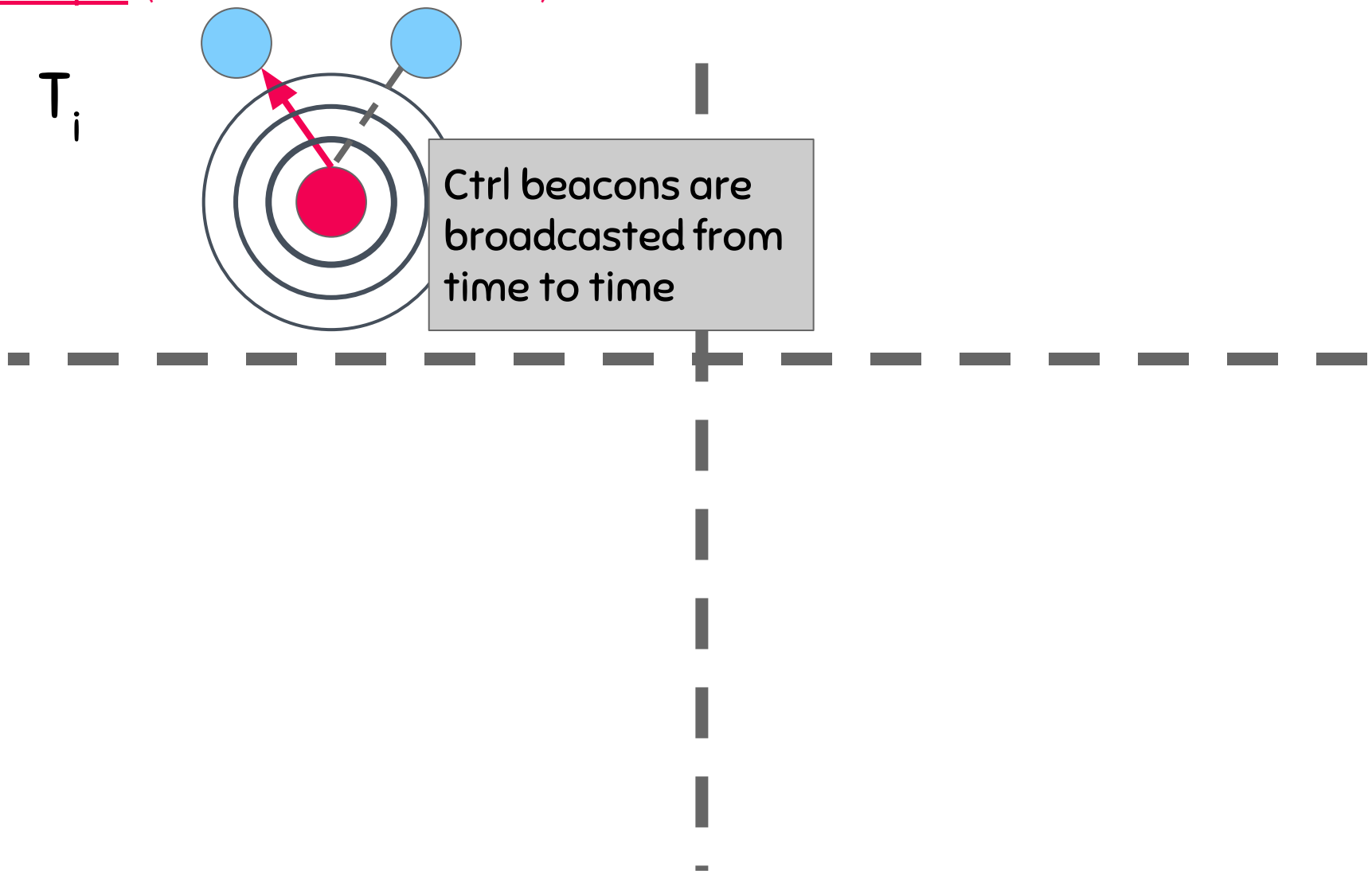
Routing under mobility events

Example. (note there are other solutions)



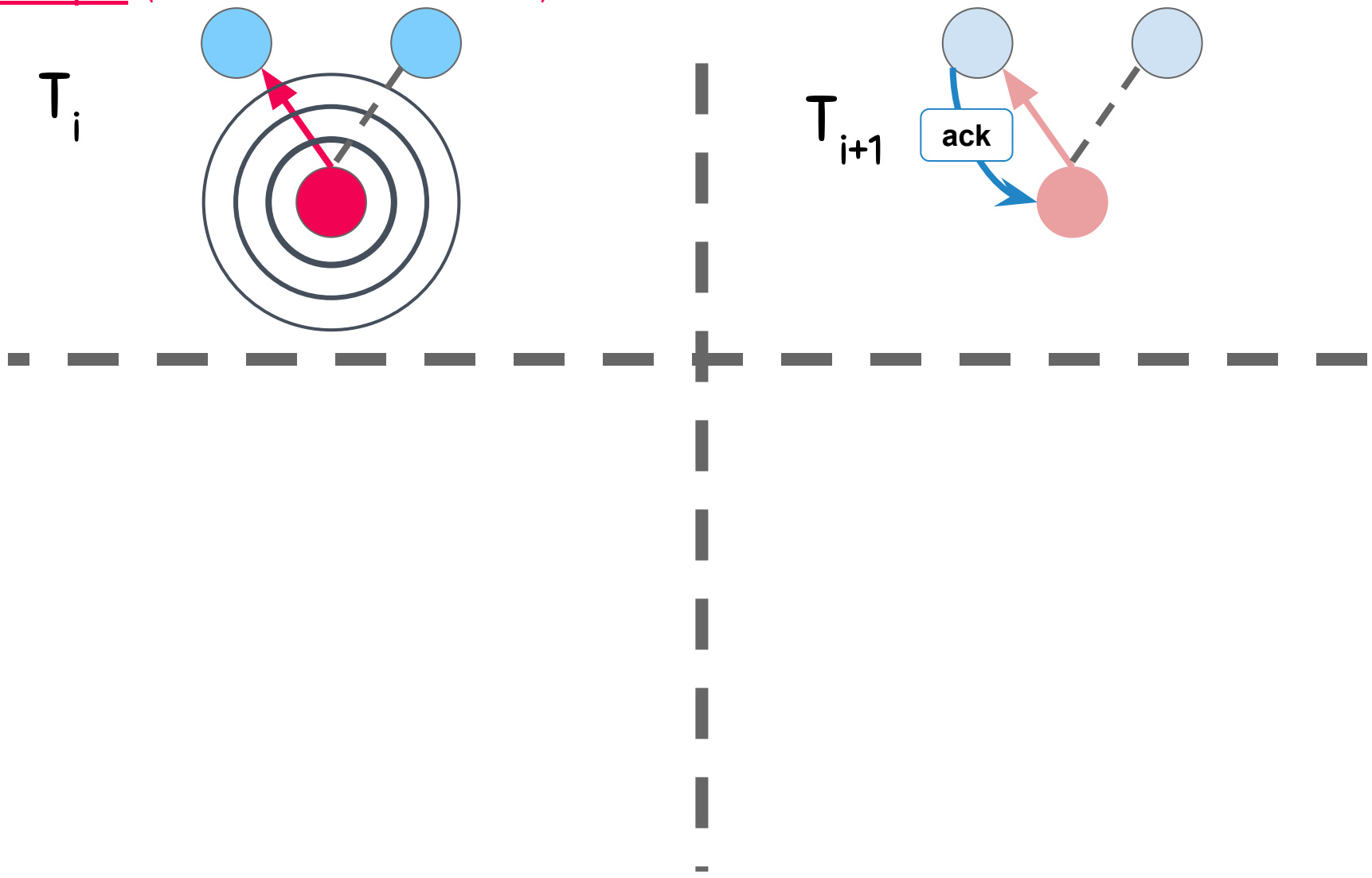
Routing under mobility events

Example. (note there are other solutions)



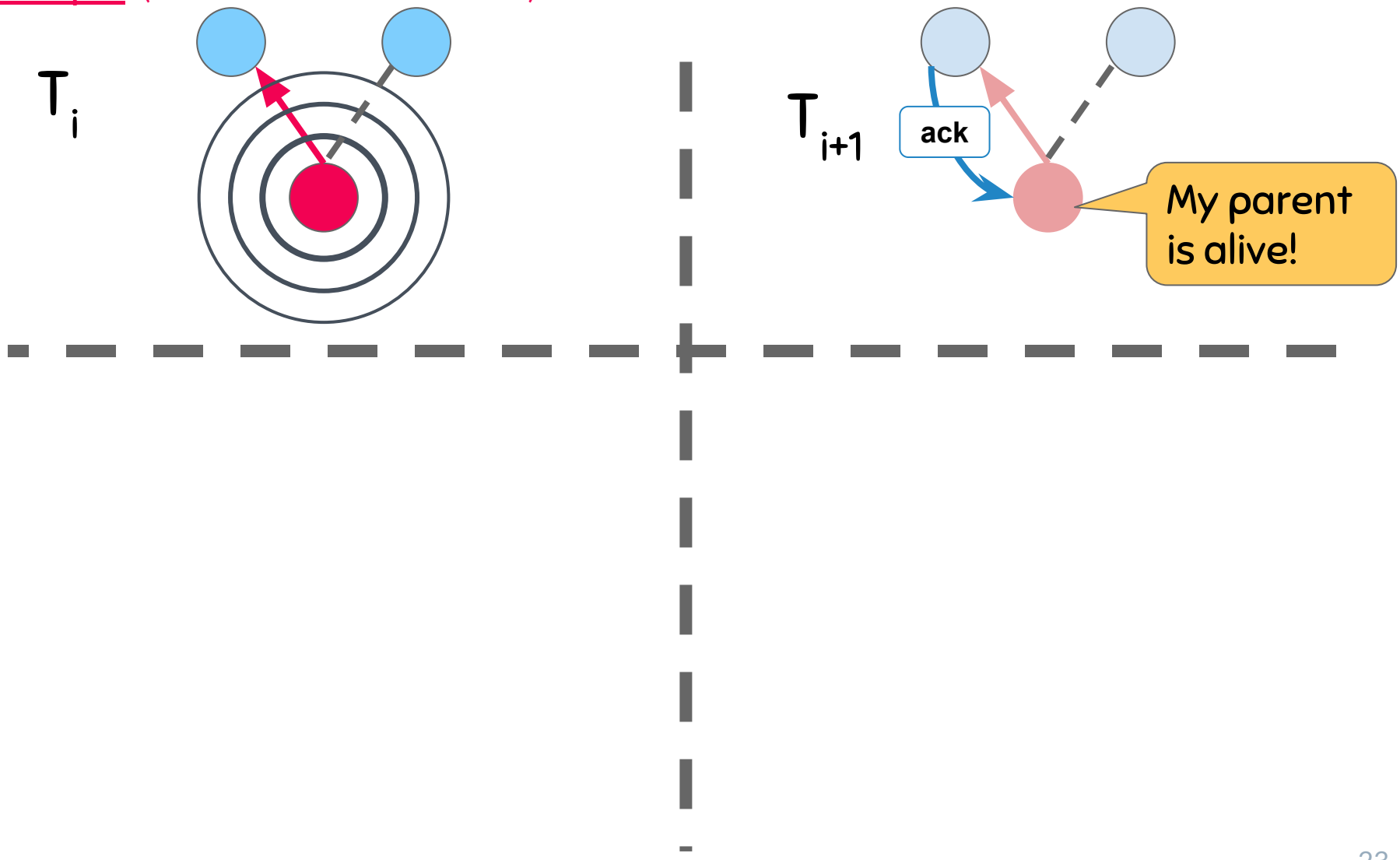
Routing under mobility events

Example. (note there are other solutions)



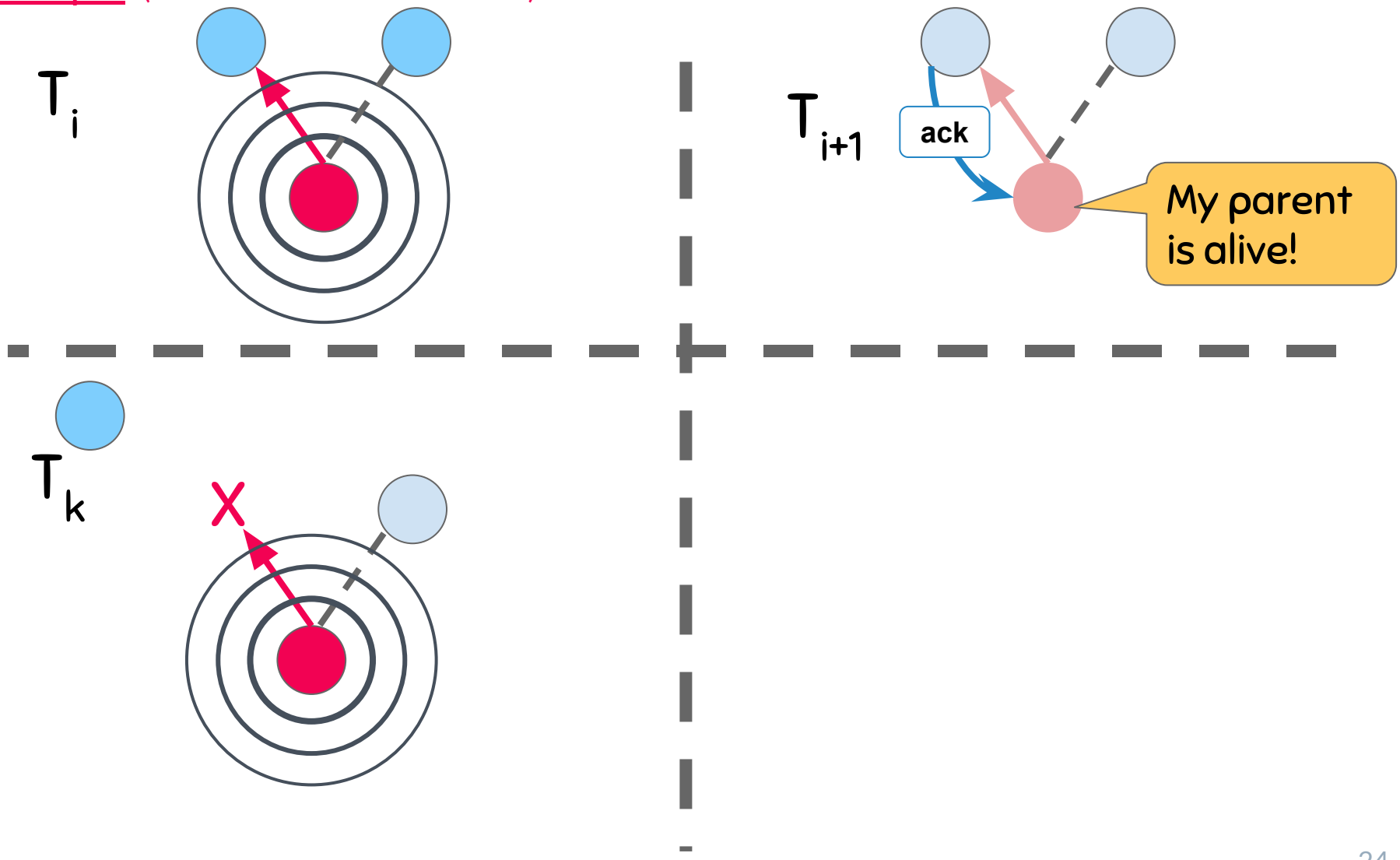
Routing under mobility events

Example. (note there are other solutions)



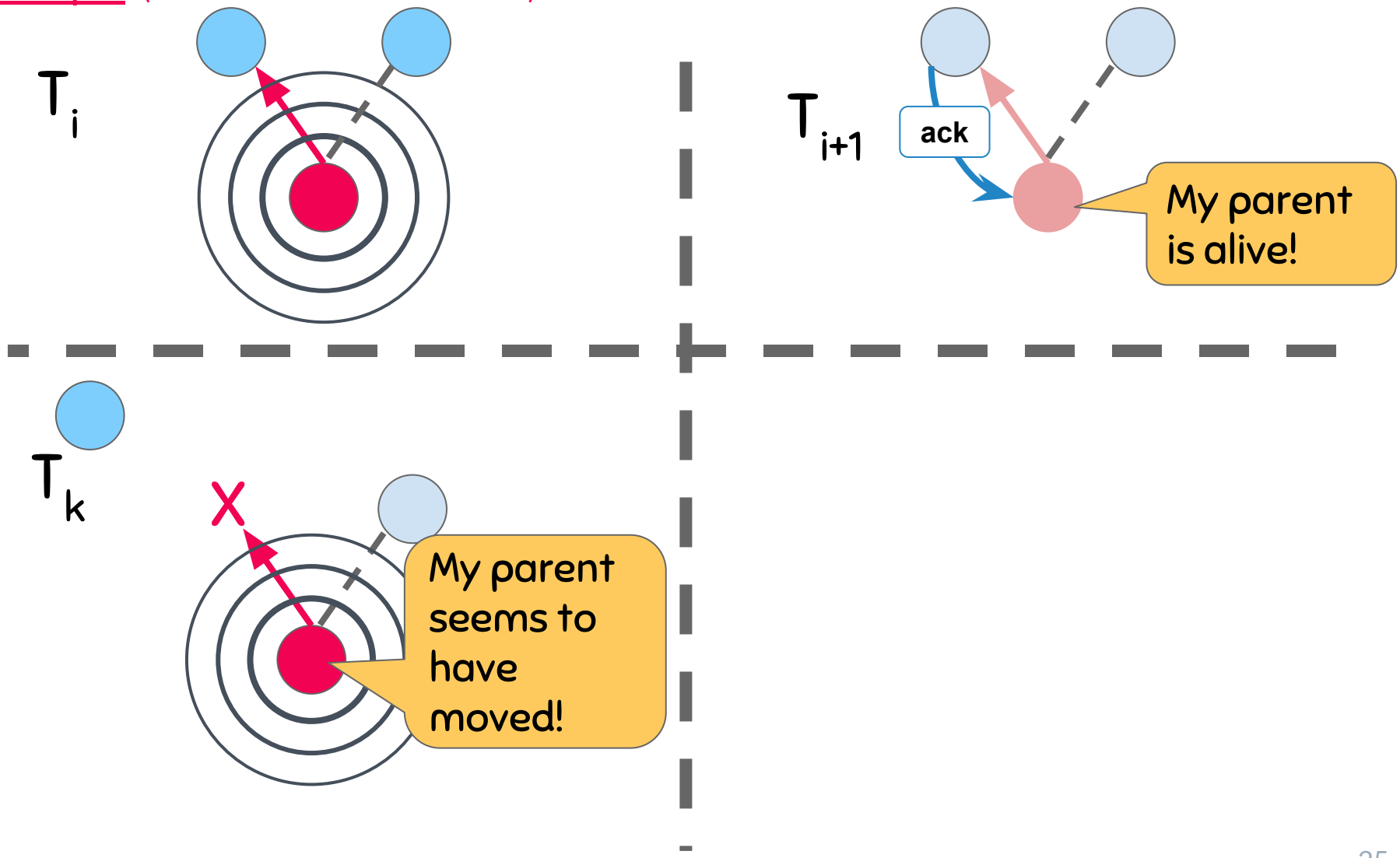
Routing under mobility events

Example. (note there are other solutions)



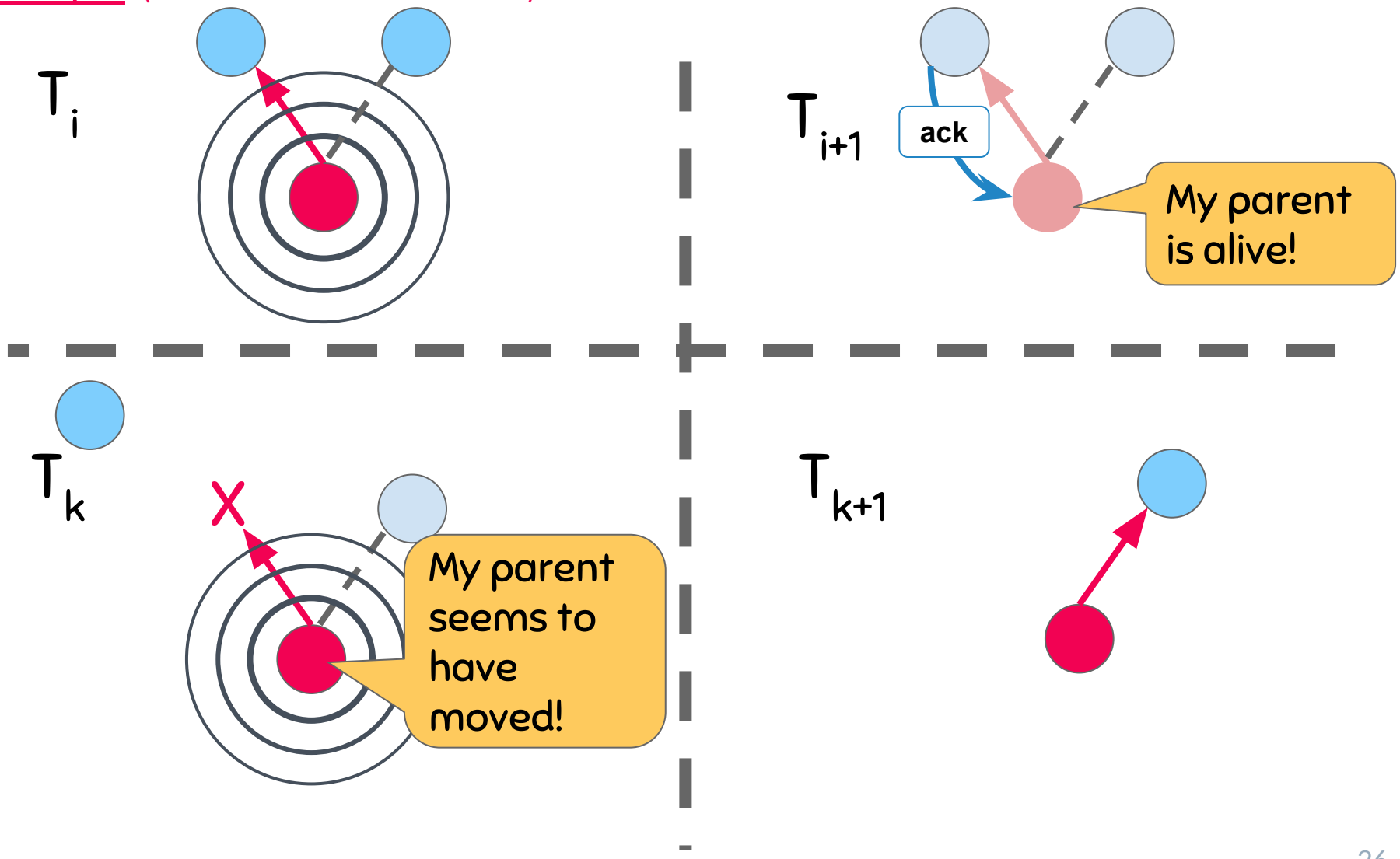
Routing under mobility events

Example. (note there are other solutions)



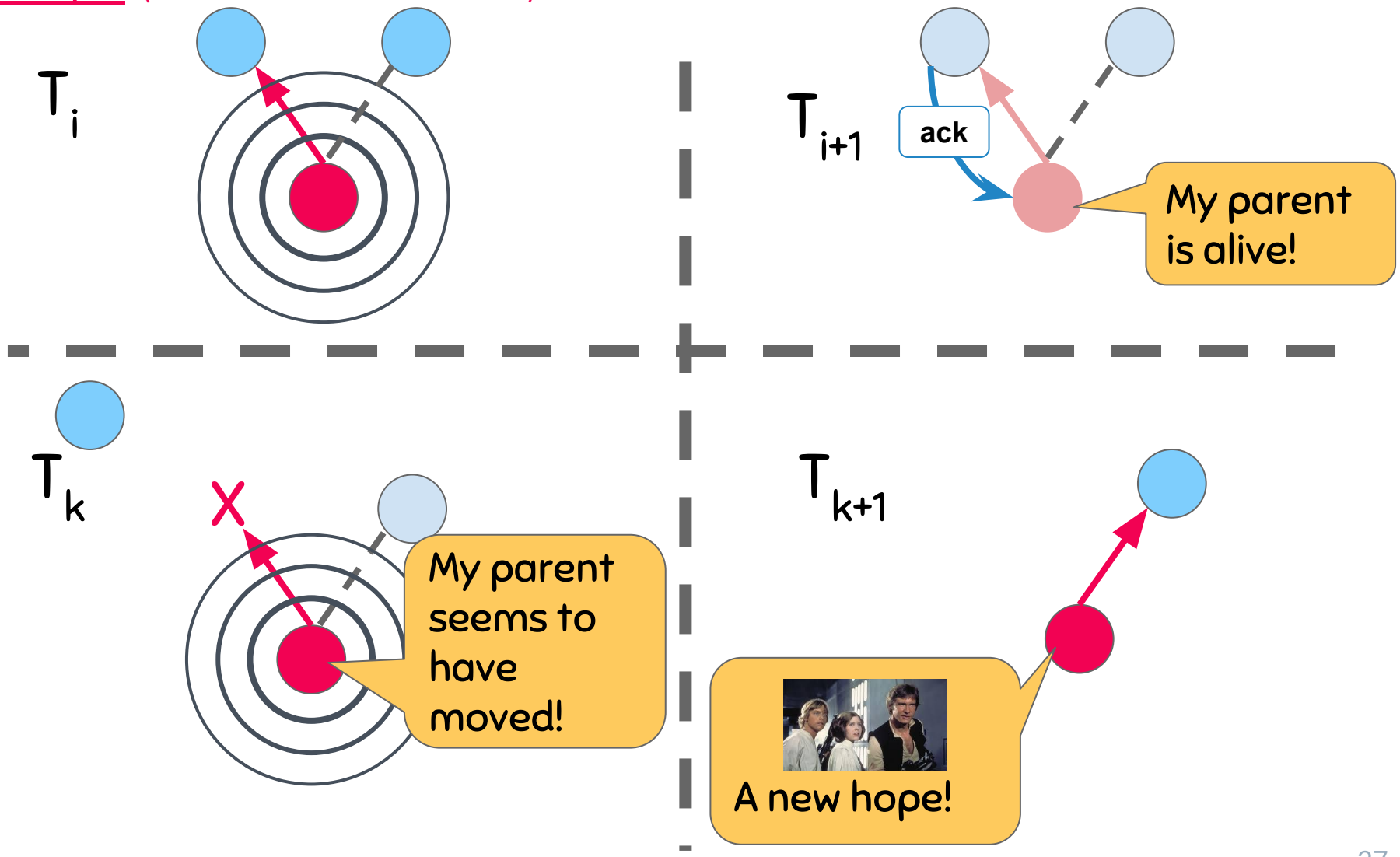
Routing under mobility events

Example. (note there are other solutions)

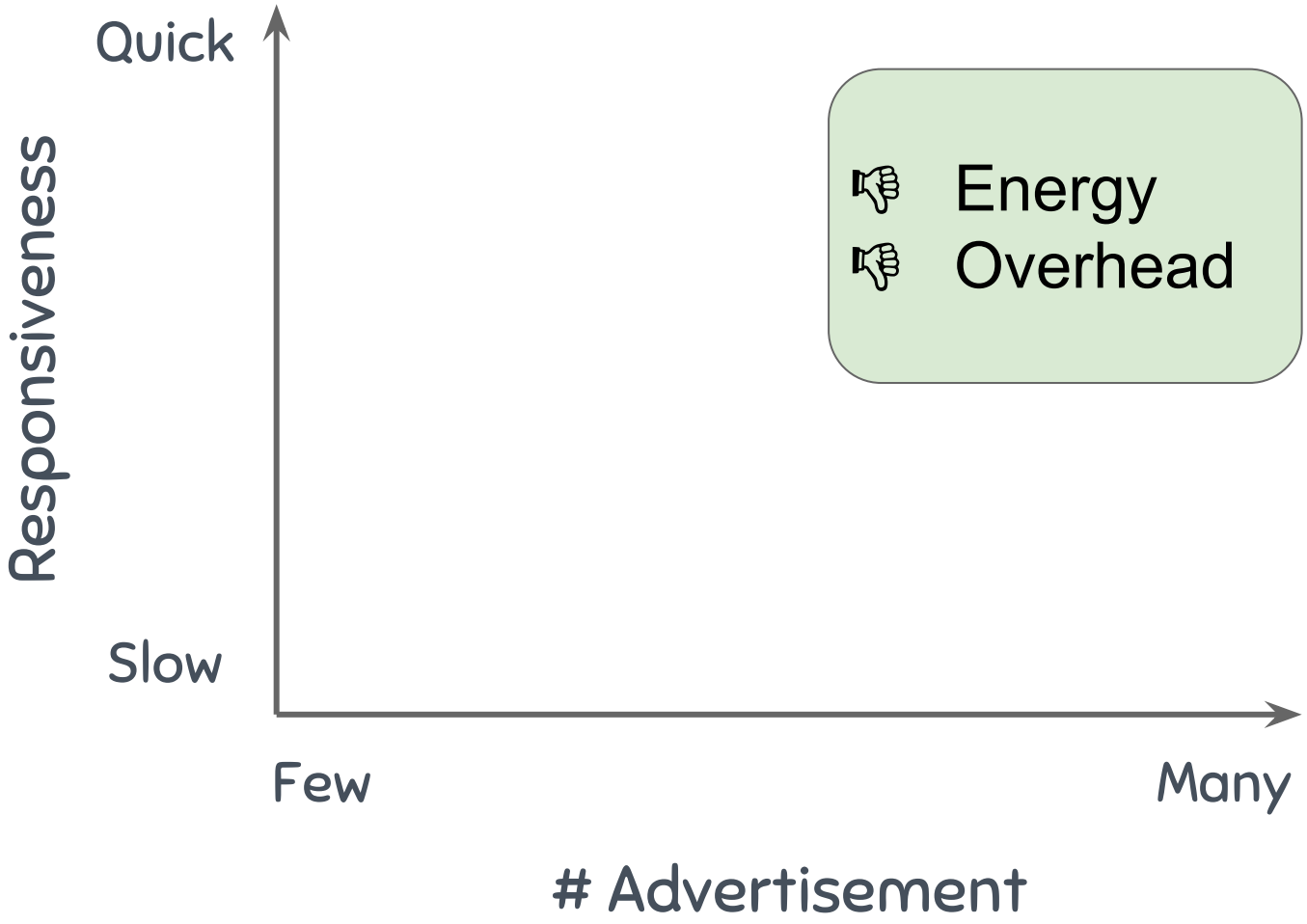


Routing under mobility events

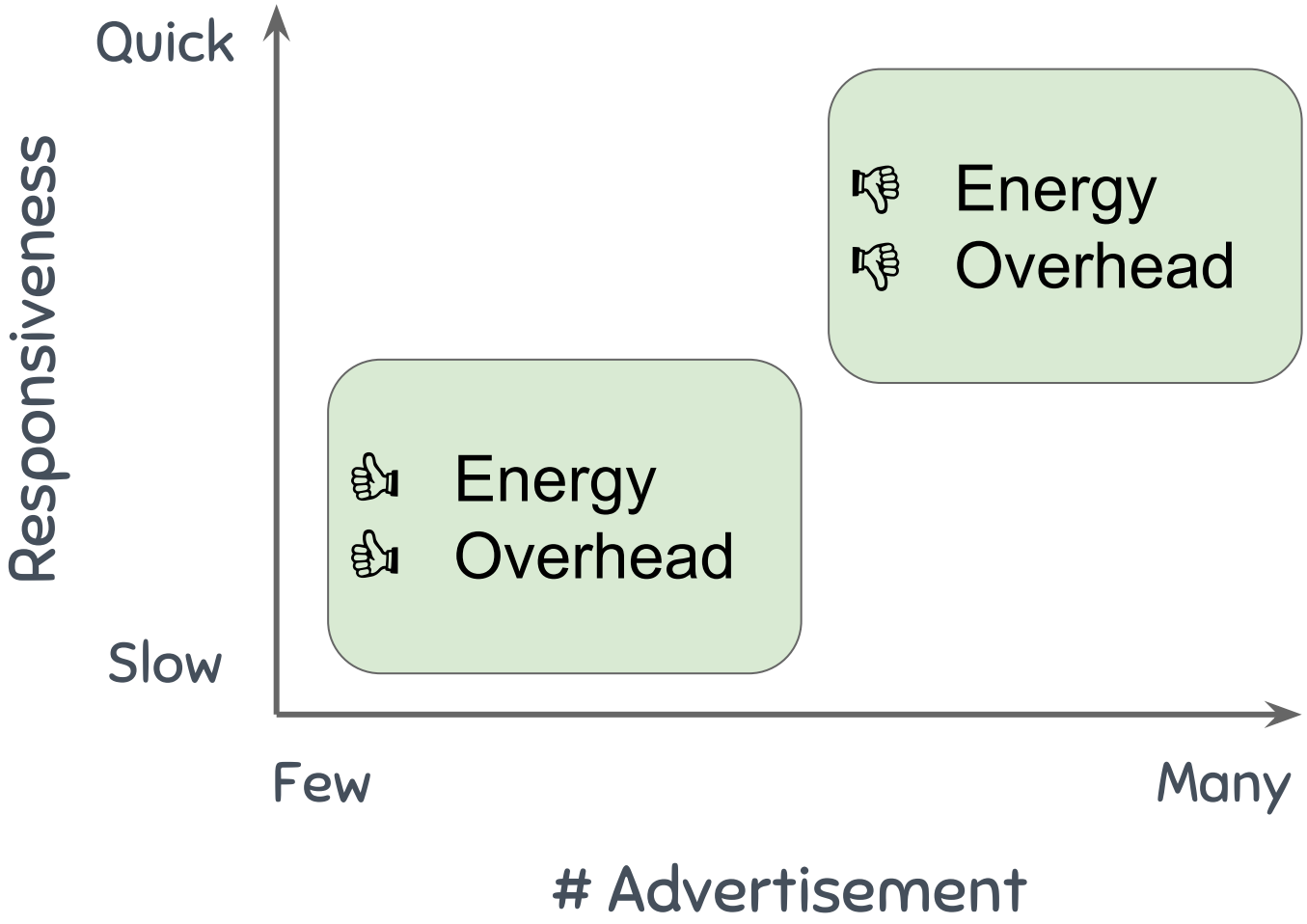
Example. (note there are other solutions)



Timer scheme trade-off



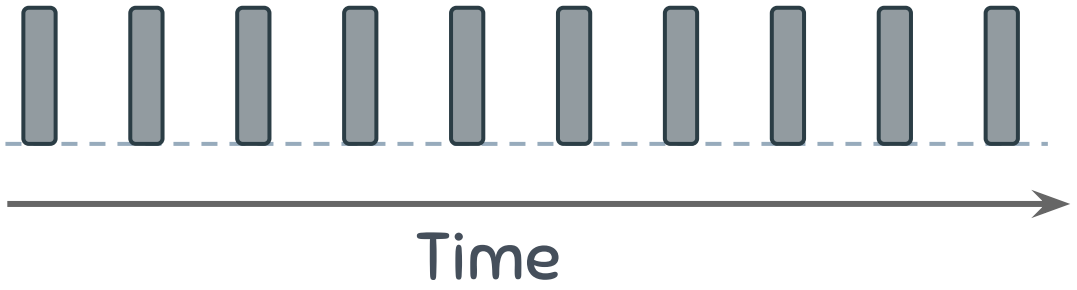
Timer scheme trade-off



Dealing with mobility and link dynamics

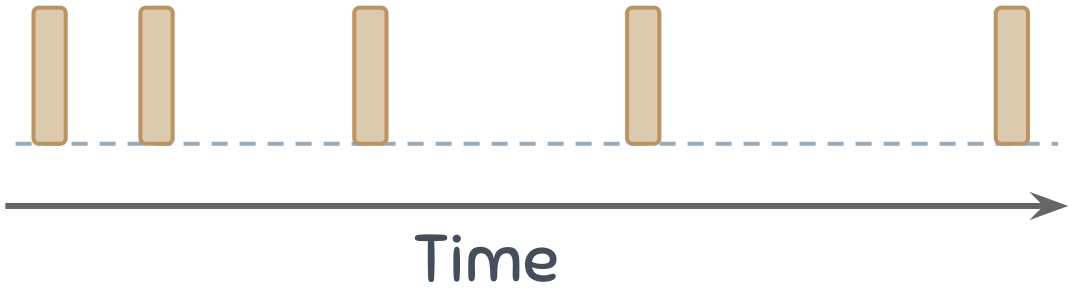
- Timer schemes
  - Control advertisements
  - Govern the communication structure construction and maintenance
- What timer schemes are most commonly used?

Dealing with mobility and link dynamics



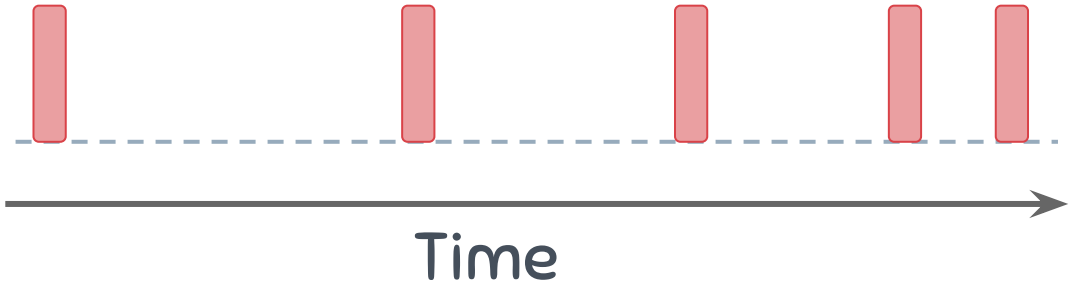
- **Periodic**
  - Large interval
    - 👍 Low channel and energy usage
    - 👎 Slow responsivity
  - Small interval
    - 👎 High channel and energy usage
    - 👍 Quick responsivity

Dealing with mobility and link dynamics



- **Trickle Timer**
  - Assumes that network will be stable (few link changes)
  - Fires bursts of advertisements when some inconsistency is detected
  - Decrease advertisement rate exponentially
  - Maximum interval ~2.3 h (RFC 6550) or ~20 min (ContikiOS)

Dealing with mobility and link dynamics



- **Reverse Trickle Timer**
  - The “opposite” of Trickle Timer
  - Assumes that as long as a node remains connected to a parent, it is likely that node will move away
  - Increase advertisement rate exponentially
  - Authors use ~20 min in their experiments

Dealing with mobility and link dynamics

1. **Reverse Trickle Timer,**
2. **Trickle Timer,**
3. **Periodic.**

- Such schemes assume:
  - 👎 Only one scheme governs the entire network
  - 👎 All devices follow the same mobility pattern

Dribble

A learn-based timer scheme selector for mobility management in IoT

- It learns the IoT device mobility pattern
- Automatically assign a proper timer scheme
  - Better balance the timer scheme trade-off

How it works...

Start with a default timer scheme

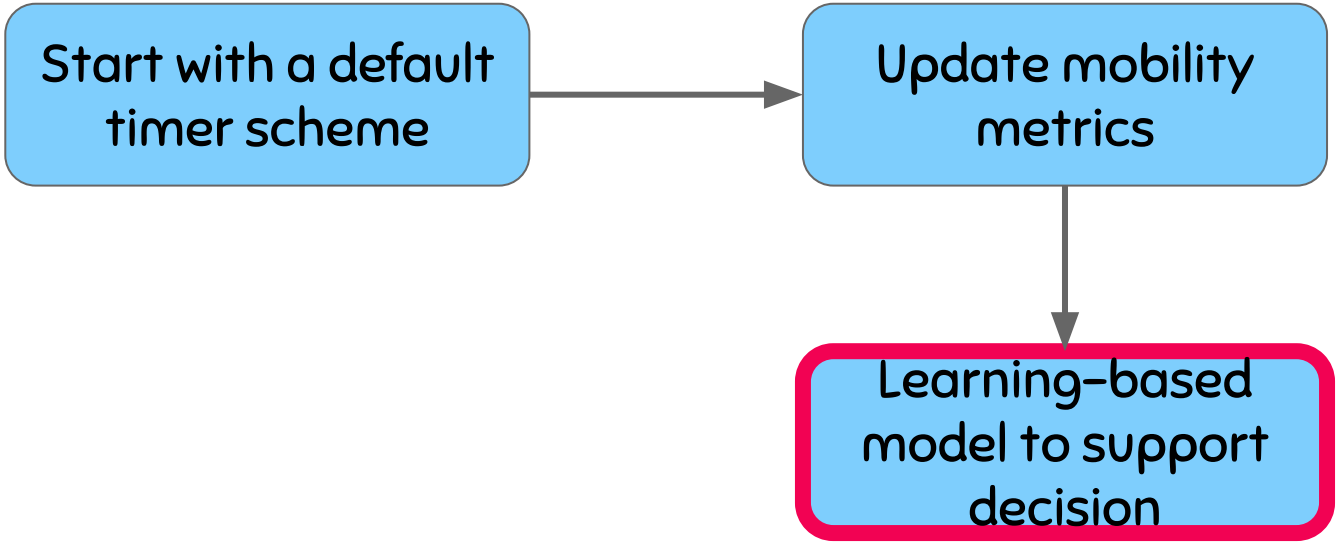
Ex:
Trickle Timer

How it works...

Ex:

- **Speed,**
- **GPS,**
- **Travel Distance,**
- **Visit Time,**
- **Interconnection Time**

How it works...

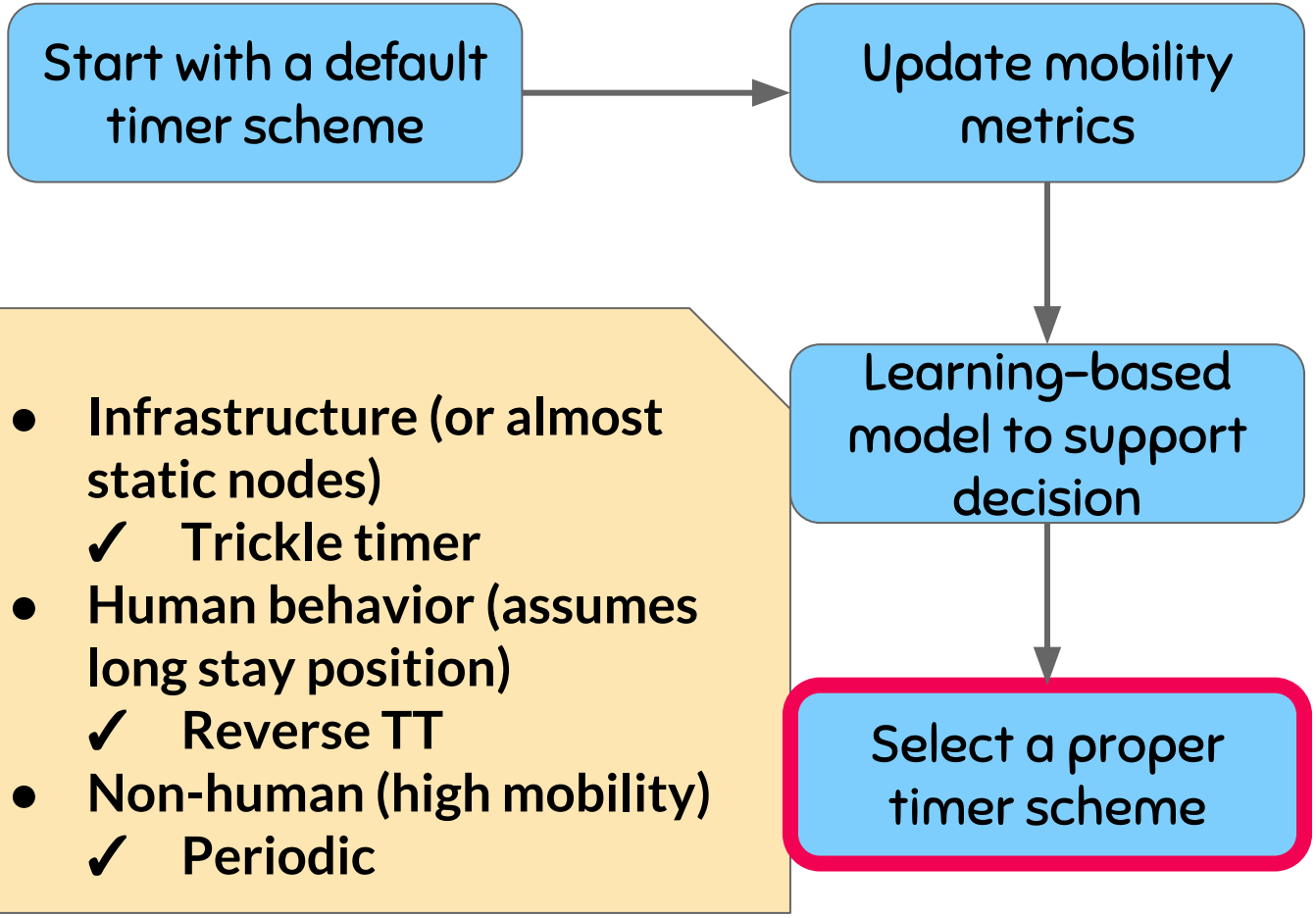


- We've tested
  - Supervised and unsupervised models
  - But we have labeled data
- Multi-Layer Perceptron classifier as learning algorithm

$$f : R^m \rightarrow R^p$$

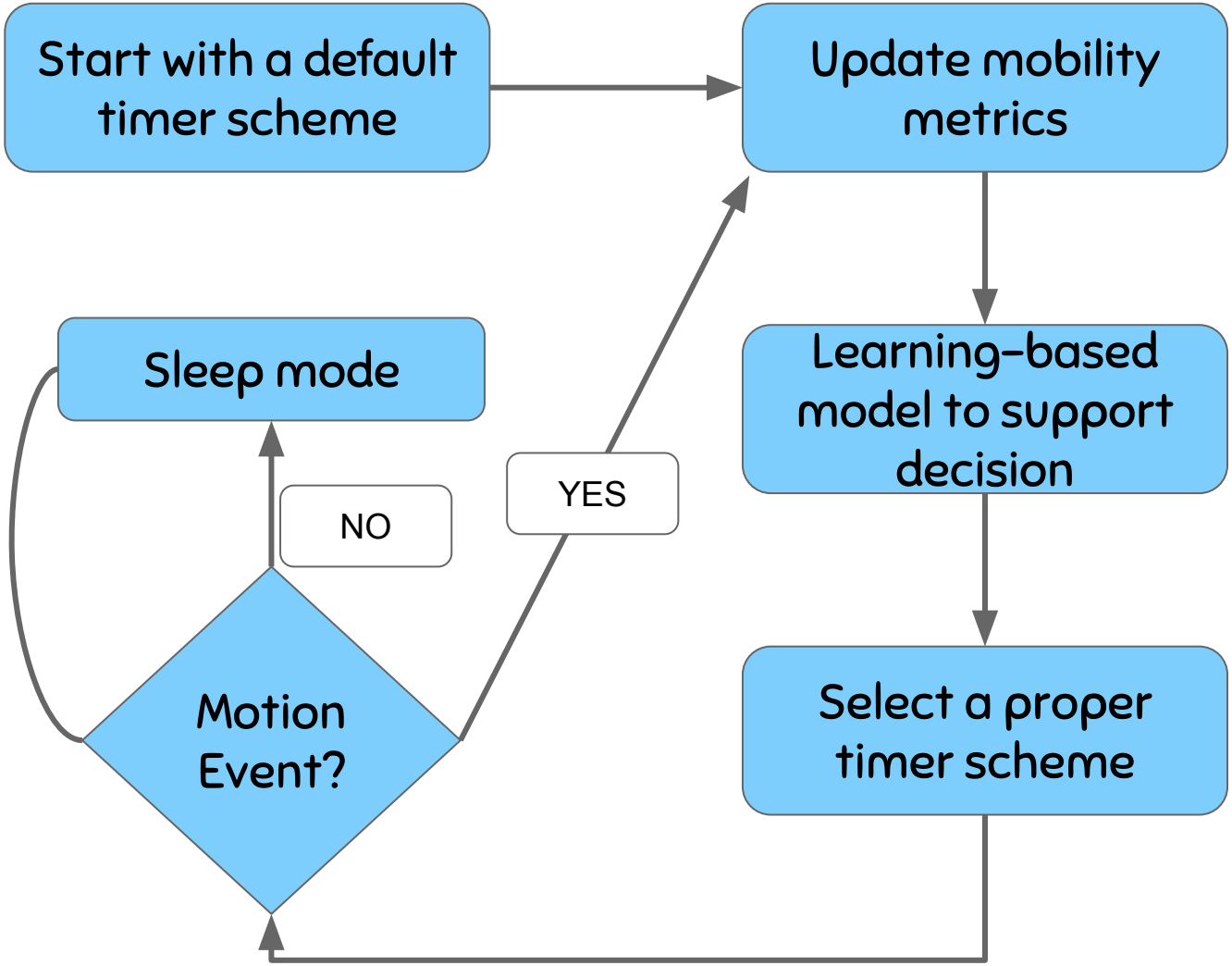
m is the mobility metrics
p is the mobility patterns

How it works...



- Infrastructure (or almost static nodes)
  - ✓ Trickle timer
- Human behavior (assumes long stay position)
  - ✓ Reverse TT
- Non-human (high mobility)
  - ✓ Periodic

How it works...


Start with a default timer scheme
Update mobility metrics
Learning-based model to support decision
Select a proper timer scheme
Motion Event?
YES
NO
Sleep mode

Experimental environment

- Sinalgo simulator
- RPL as routing protocol
  - Tree data traffic enabled: M2P, P2M, and P2P
  - Storing mode
  - ETX as Objective function

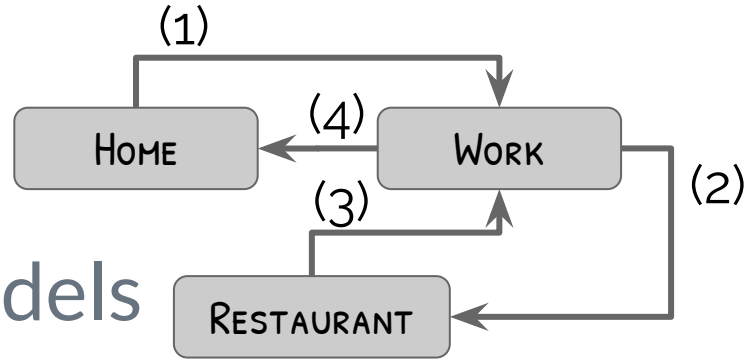
Simulation setup	
Duration	15 days

Simulation setup	
Duration	15 days
# nodes	200
Base station	1 (center)
Distribution	Random

Simulation setup	
Duration	15 days
# nodes	200
Base station	1 (center)
Distribution	Random
DIM	1500m x 1500m (campus)

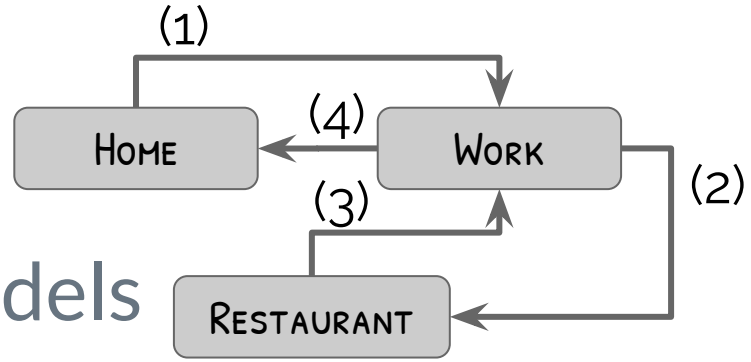
Simulation setup	
Duration	15 days
# nodes	200
Base station	1 (center)
Distribution	Random
DIM	1500m x 1500m (campus)
Radio Range	100 (m)
Transmission Model	CC2420-like
# random topologies	15
Timer schemes	
Trickle and Reverse Trickle timers	Min = 1s, Max = ~20 min
Periodic	60s

Device mobility modelling

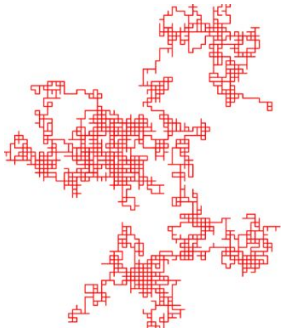


- We use two mobility models
  - Group Regularity Mobility model (GRM)
    - Human-like

Device mobility modelling

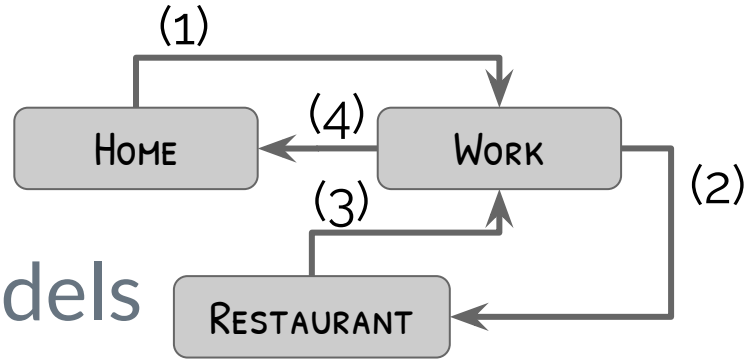


- We use two mobility models
  - Group Regularity Mobility model (GRM)
    - Human-like
  - Cyclical Random Waypoint Mobility Model (CRWP)
    - Non-human

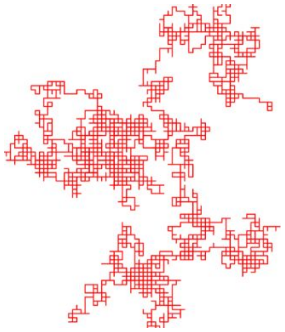


Evaluation

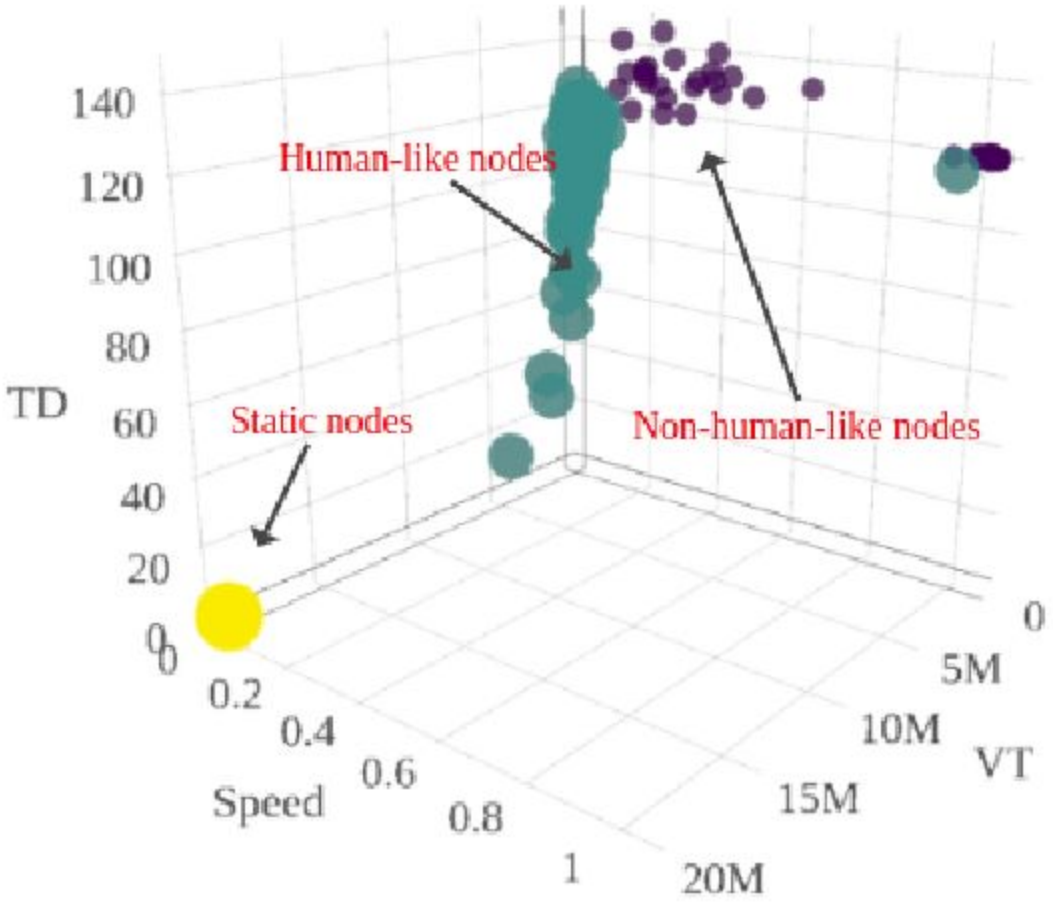
Device mobility modelling

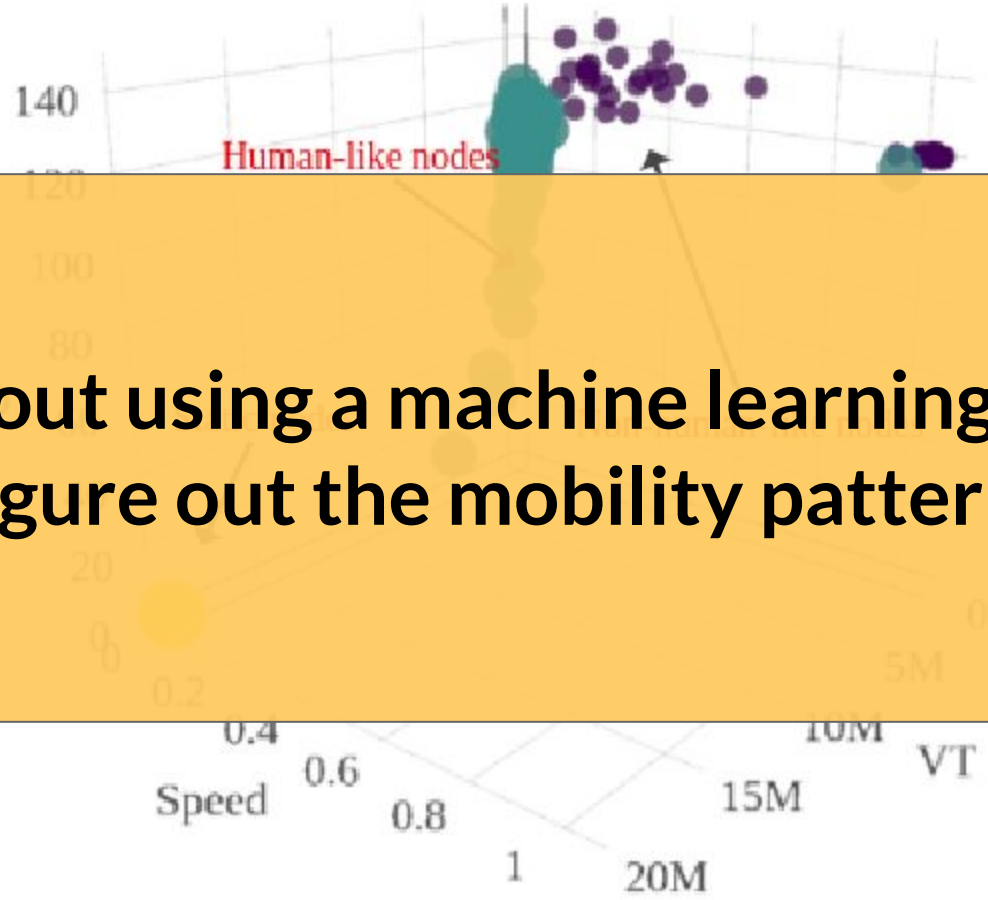


- We use two mobility models
  - Group Regularity Mobility model (GRM)
    - Human-like
  - Cyclical Random Waypoint Mobility Model (CRWP)
    - Non-human



- Some static nodes to represent the infrastructure


Human-like nodes
Static nodes
Non-human-like nodes
TD
140
120
100
80
60
40
20
0
0.2
0.4
0.6
0.8
1
Speed
0
5M
10M
15M
20M
VT



What about using a machine learning model to figure out the mobility pattern?

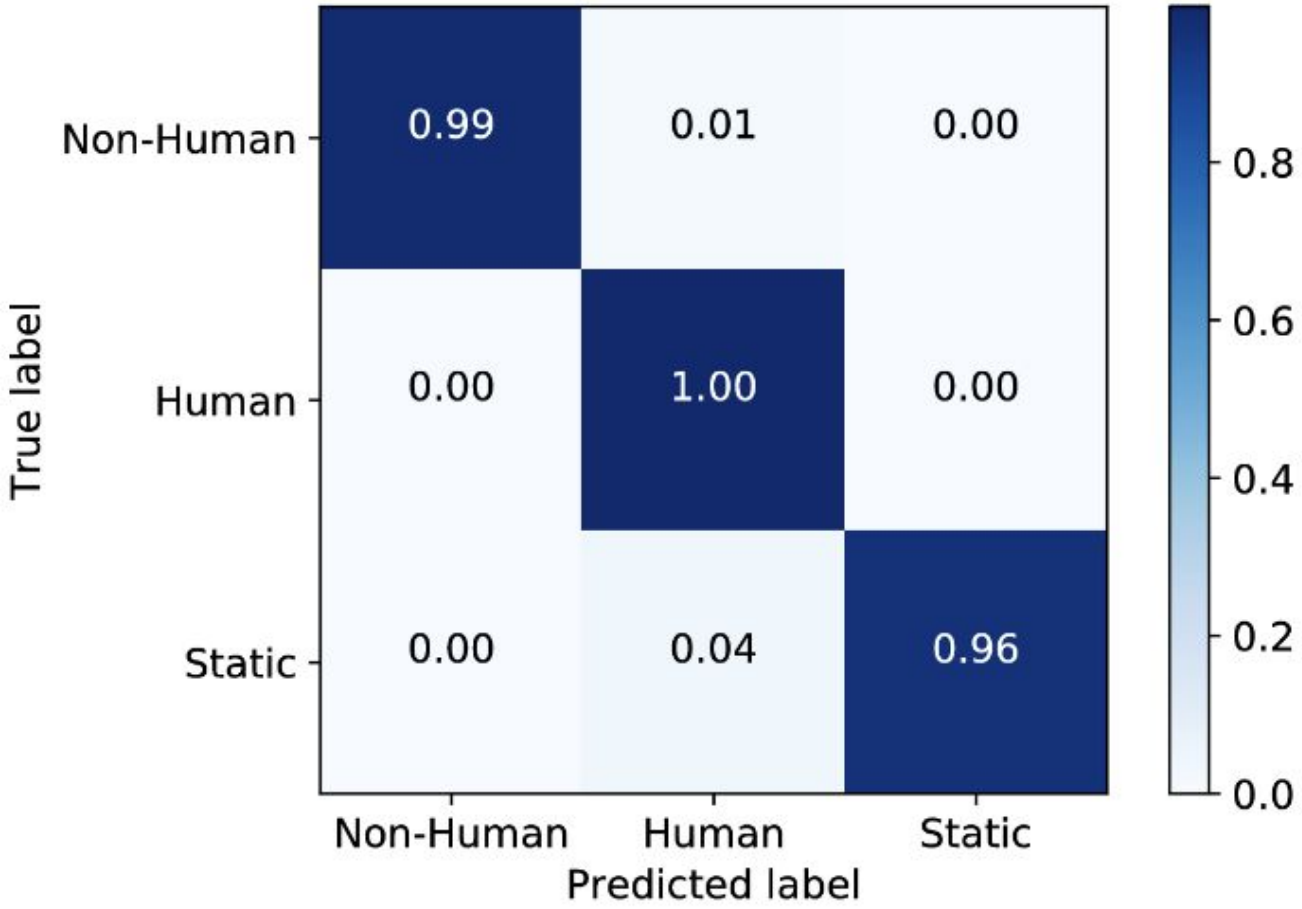
Neural Network (Multi-Layer Perceptron) Architecture and parameters

Architecture	1 Hidden layer with 100 neurons

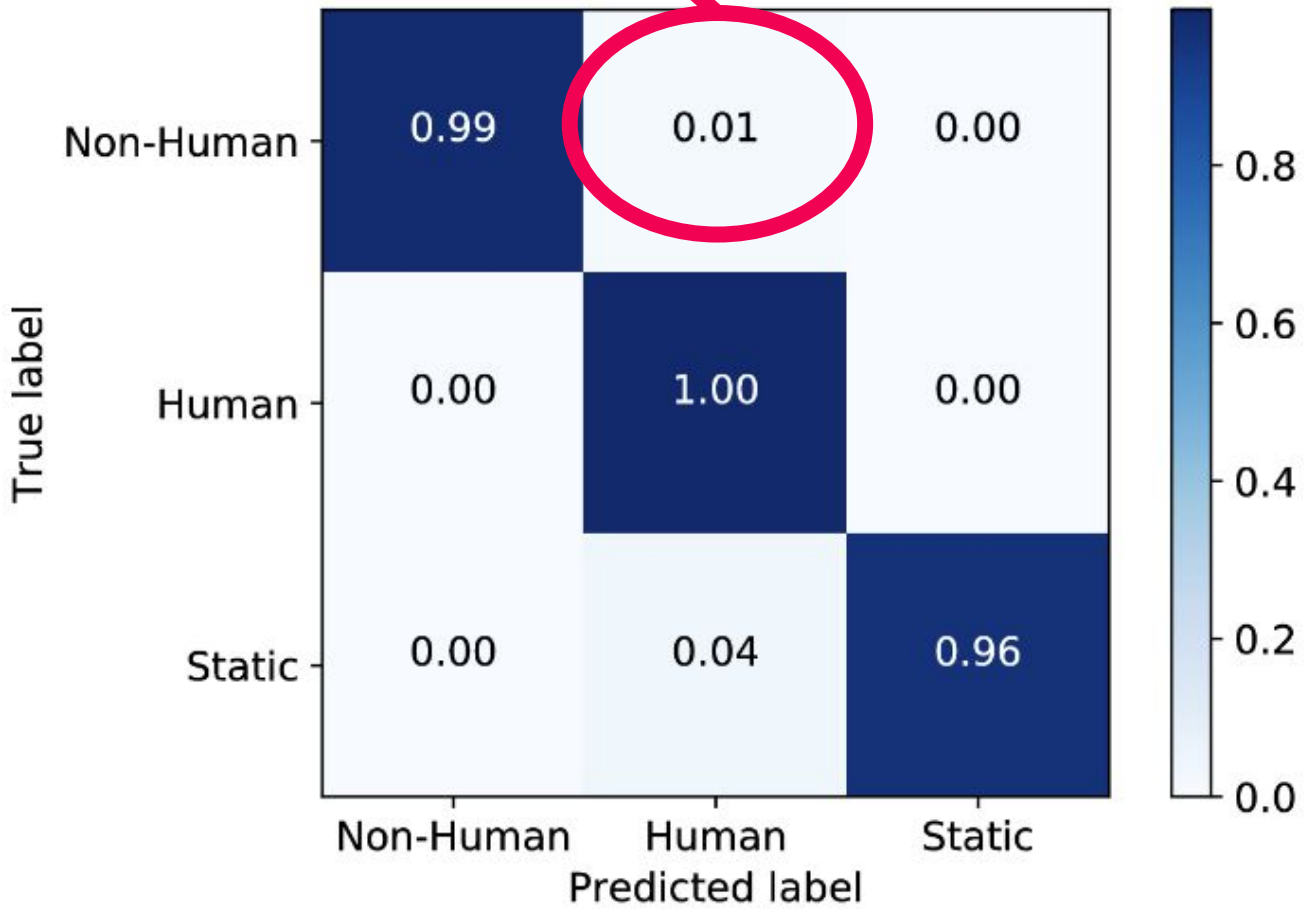
Neural Network (Multi-Layer Perceptron) Architecture and parameters	
Architecture	1 Hidden layer with 100 neurons
Activation	Rectified linear unit function
Learning rate	Constant
# epochs	500
Weight optimization	Adam
Train dataset	10 random topologies
Validation model	10-fold cross-validation

Neural Network (Multi-Layer Perceptron) Architecture and parameters	
Architecture	1 Hidden layer with 100 neurons
Activation	Rectified linear unit function
Learning rate	Constant
# epochs	500
Weight optimization	Adam
Train dataset	10 random topologies
Validation model	10-fold cross-validation

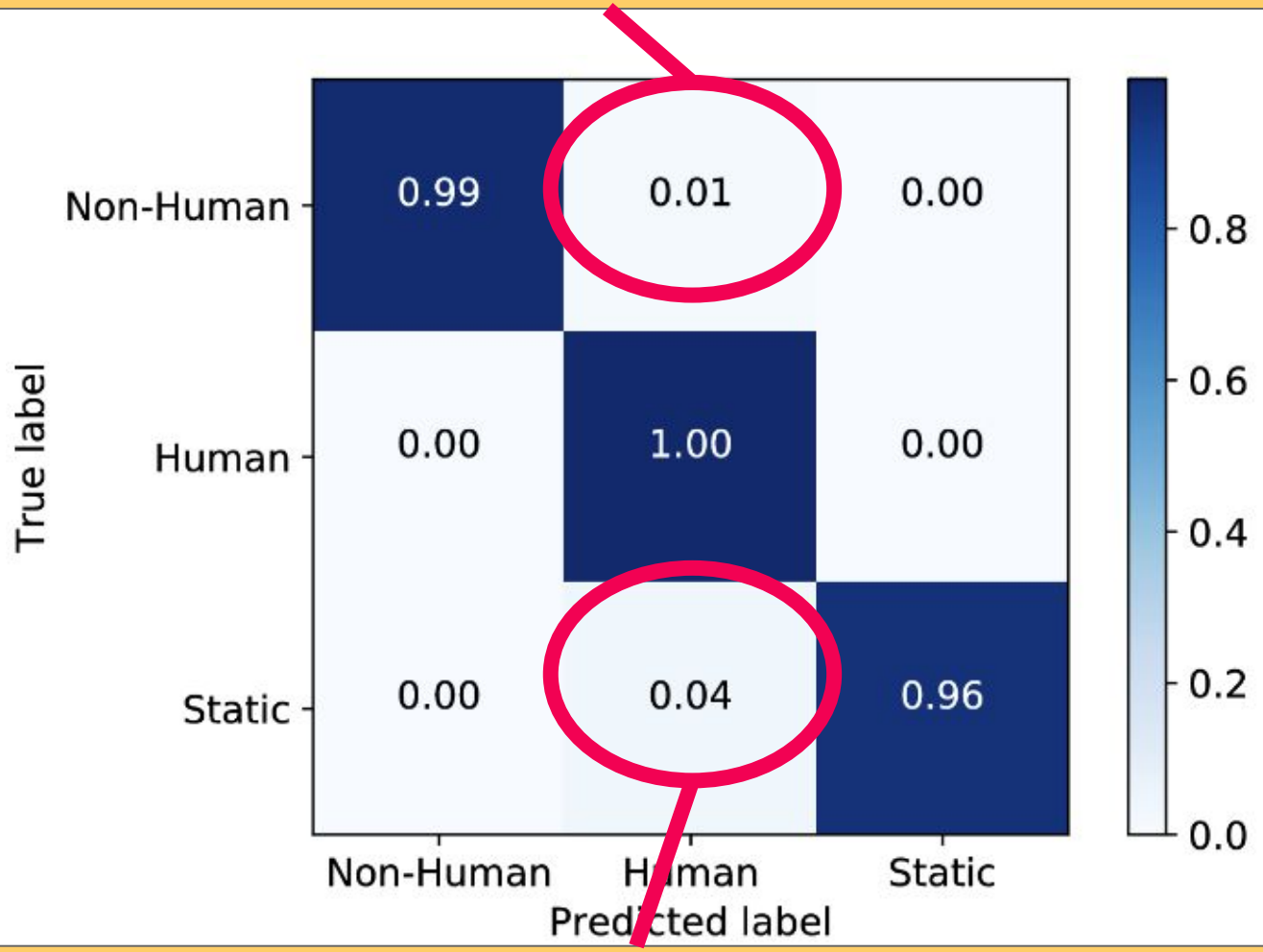
	Precision	Recall
Non-Human	1	0.99
Human	0.98	1
Static	1	0.96
Avg/Total	**0.99**	**0.99**


Non-Human
Human
Static
True label
0.99
0.01
0.00
0.00
1.00
0.00
0.00
0.04
0.96
Non-Human
Human
Static
Predicted label
0.8
0.6
0.4
0.2
0.0

1% of Non-human were misclassified as human



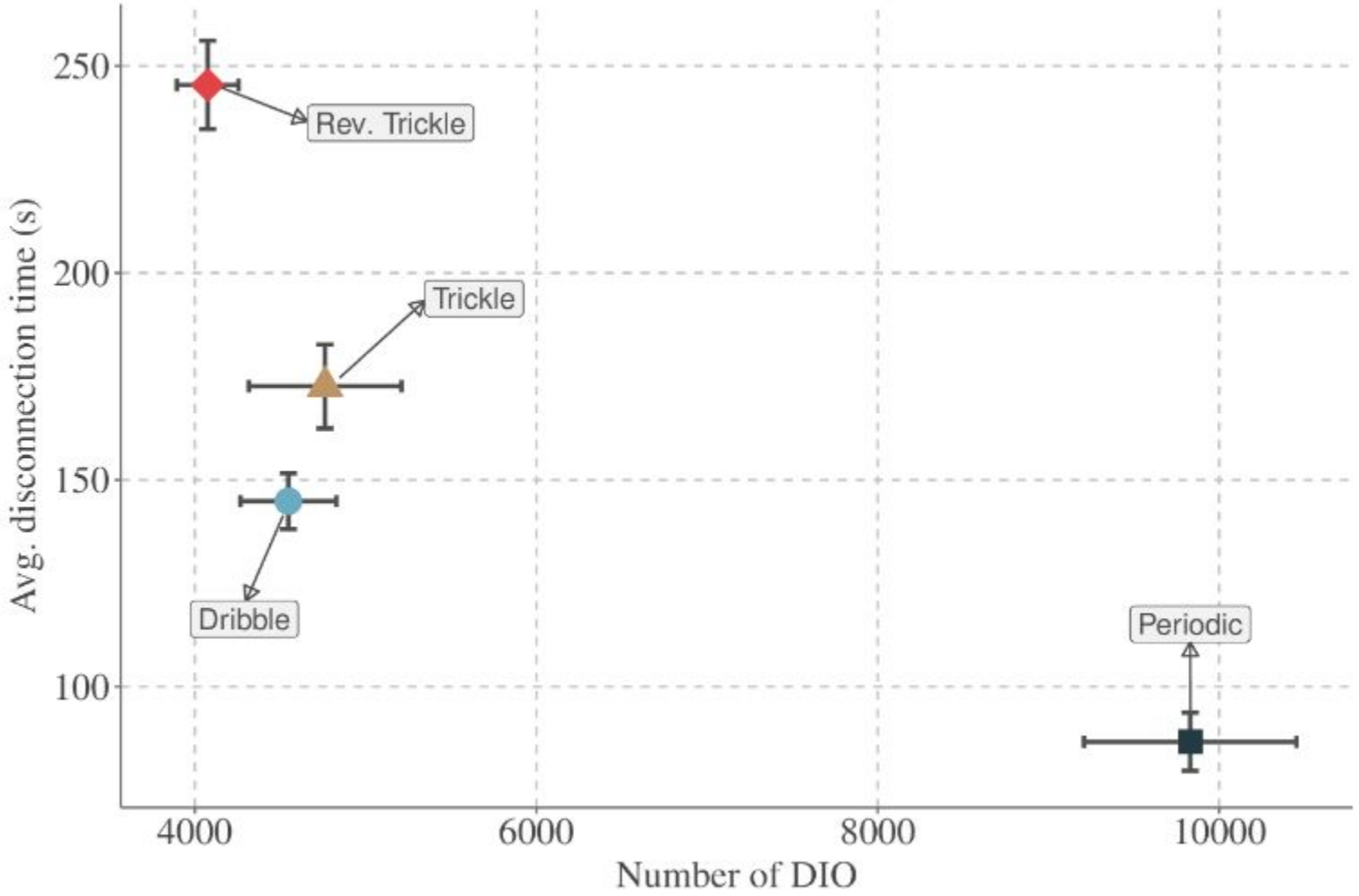
1% of Non-human were misclassified as human



4% of static were misclassified as human

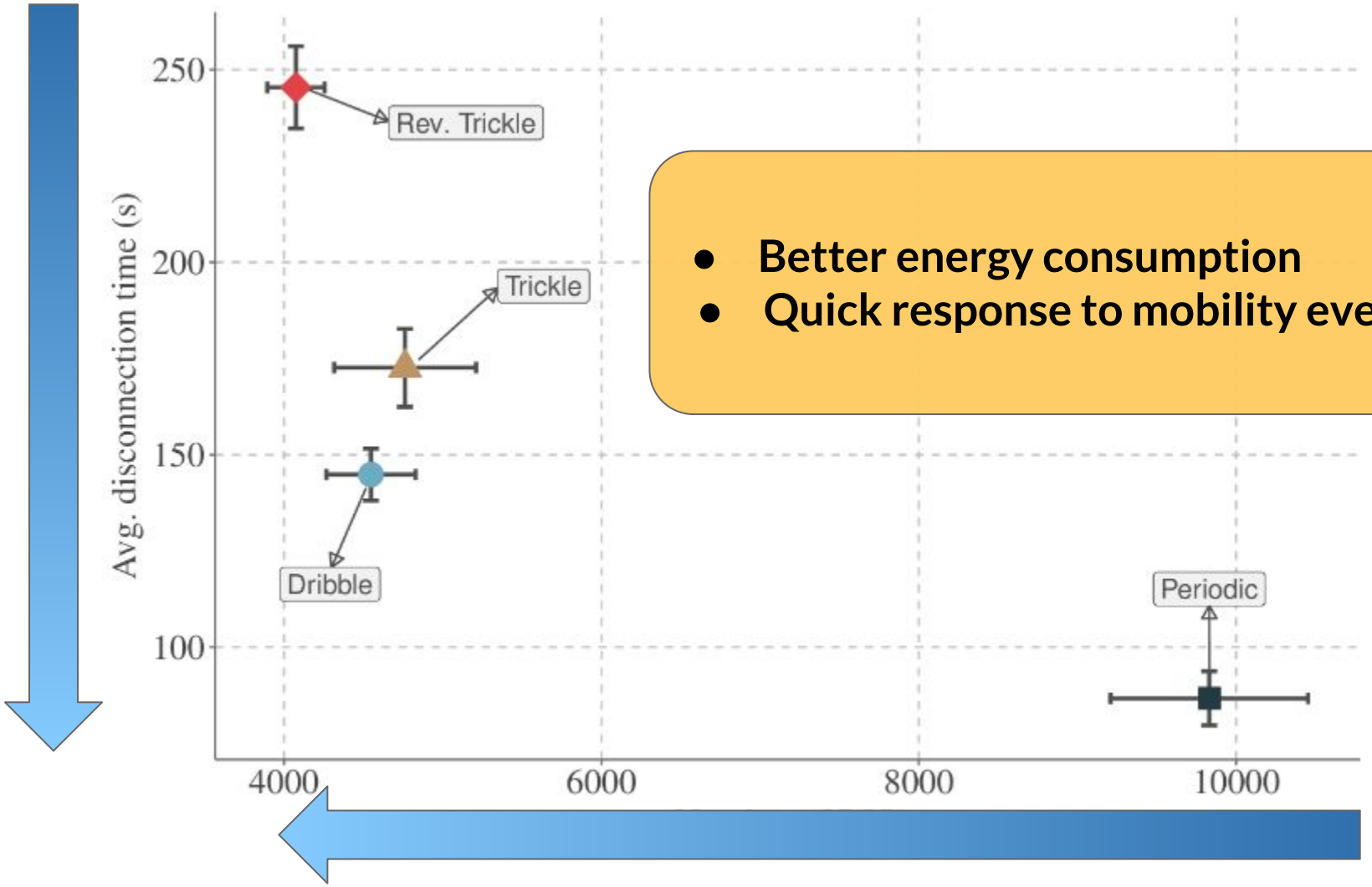
Evaluation

Trade-off balance



Evaluation

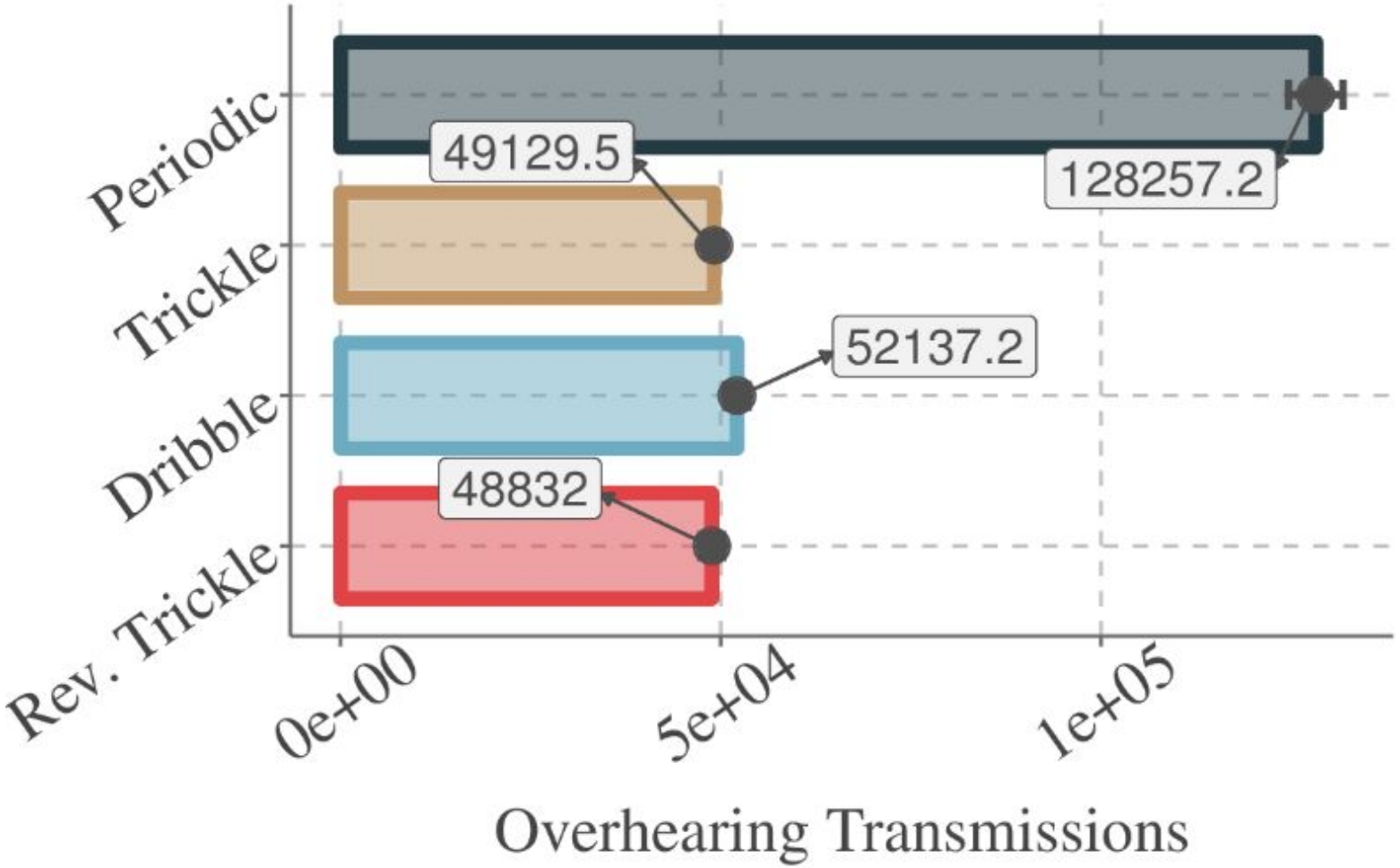
Trade-off balance



- **Better energy consumption**
- **Quick response to mobility events**

Evaluation

Overhead



Conclusion and future work

- We have proposed Dribble
  - A learn-based time scheme selector
  - It sets a custom timer scheme given the mobility pattern of a IoT device
  - Also, Dribble presented a better timer scheme trade-off balance

Conclusion and future work

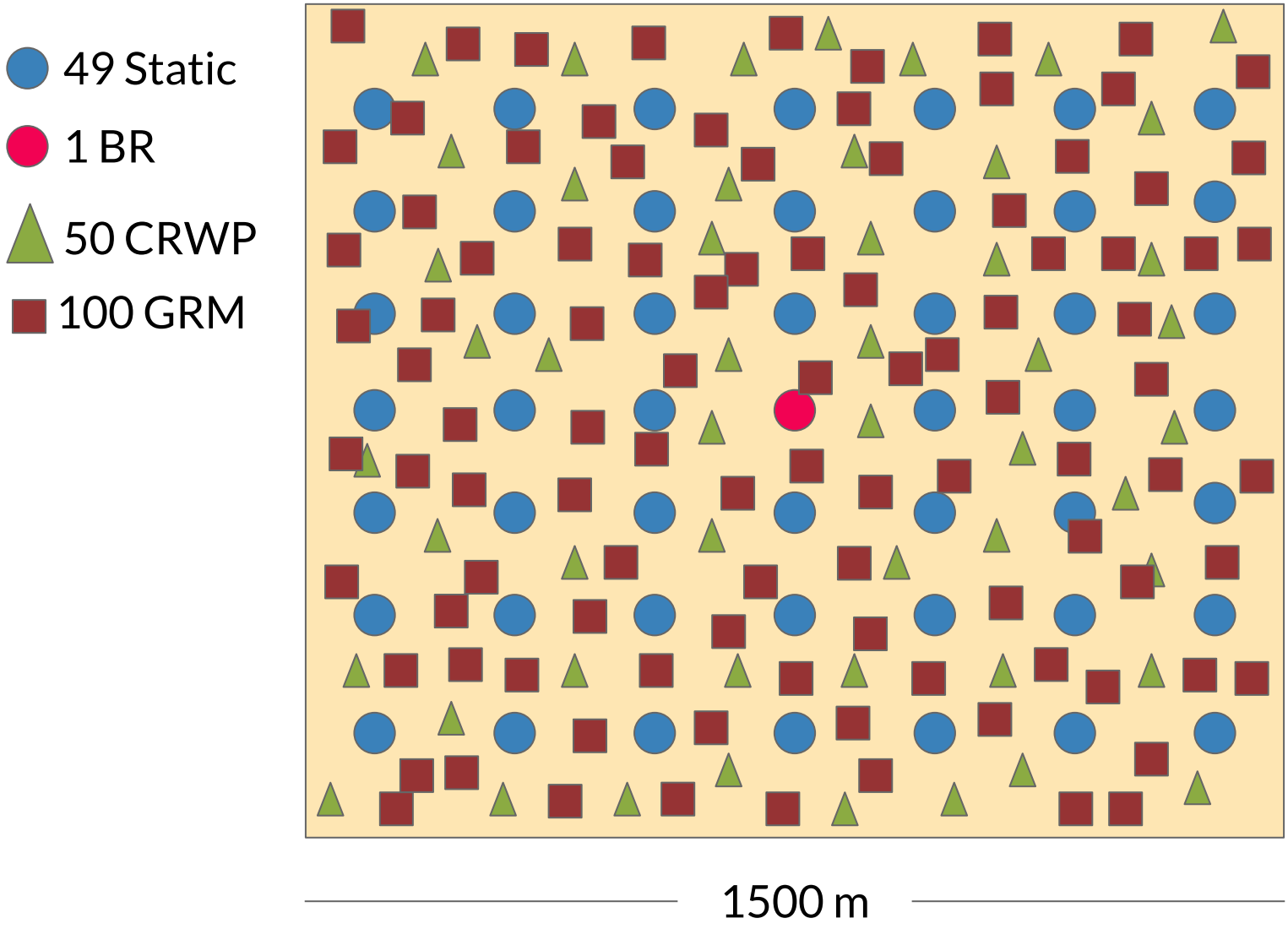
- We intent to extend Dribble to support:
  - Automatic parametrization of timer schemes
  - Automatic way to associate mobility patterns to timer schemes.

Thanks!

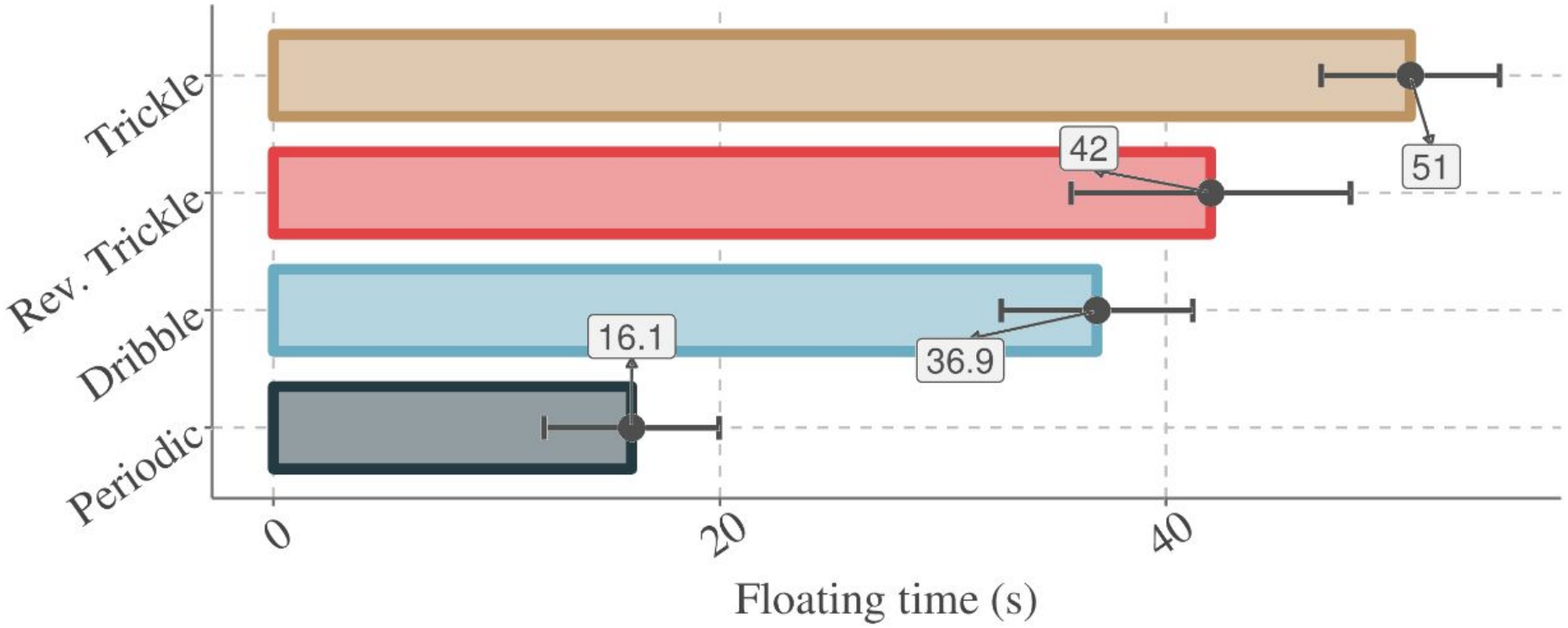
Any questions?

You can find us at:

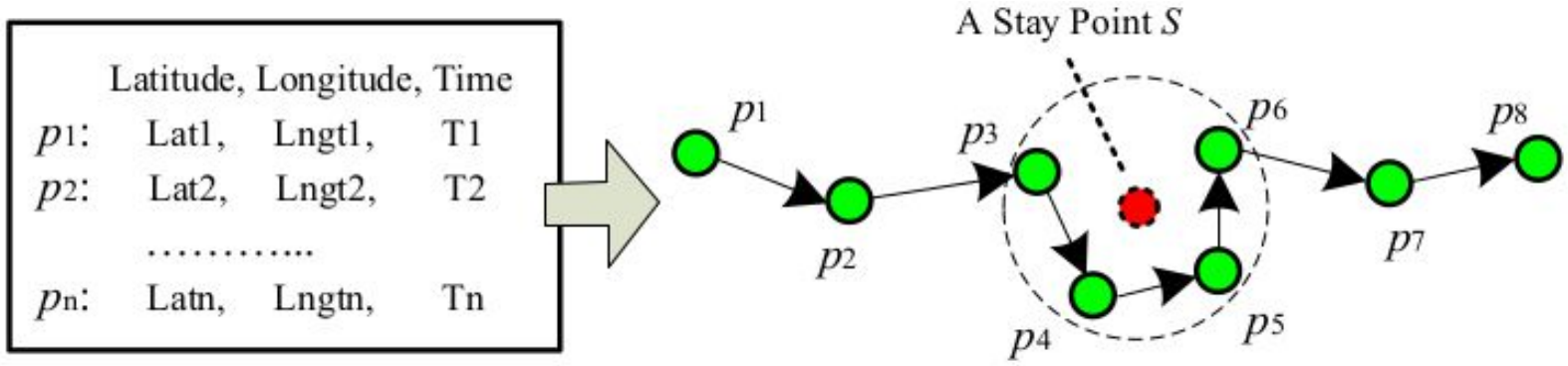
- bruno.ps@dcc.ufmg.br
- rettore@dcc.ufmg.br
- lfvieira@dcc.ufmg.br
- loureiro@dcc.ufmg.br


49 Static
1 BR
50 CRWP
100 GRM
1500 m

Time of disconnection



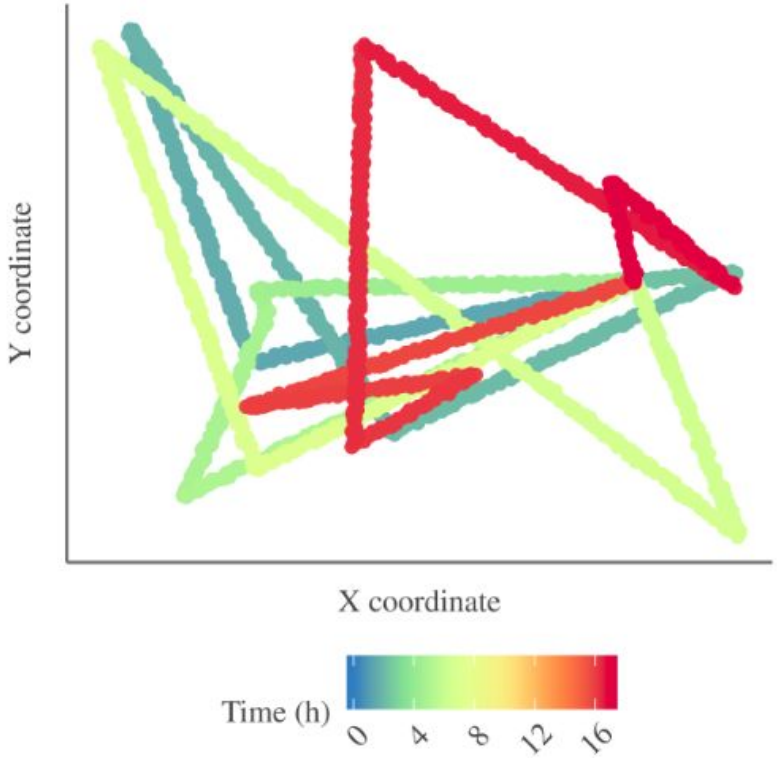
When do updates?



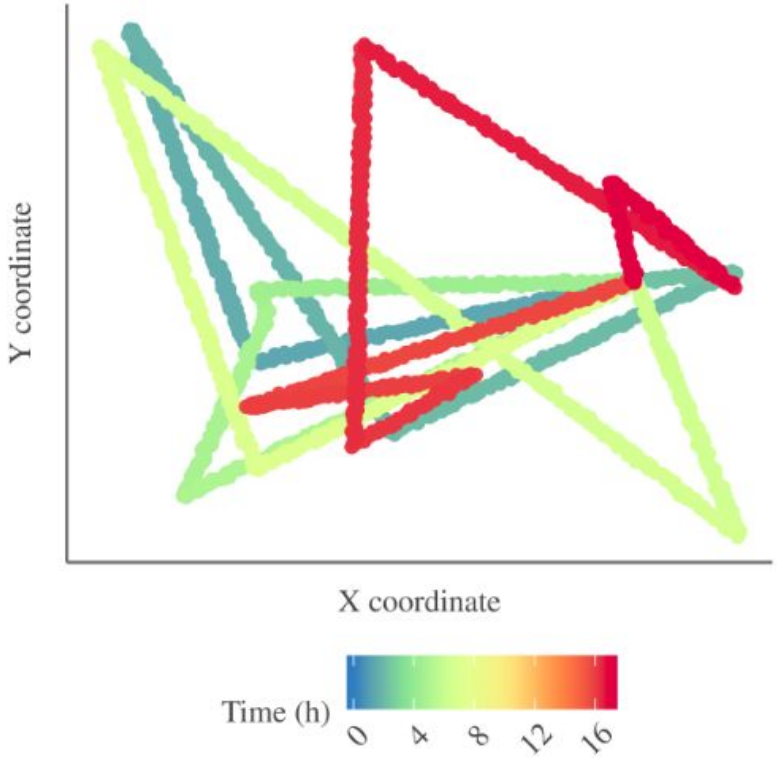
GPS points: $P = \{p_m, p_{m+1}, \dots, p_n\}$ $\forall$ $m < i \leq n$

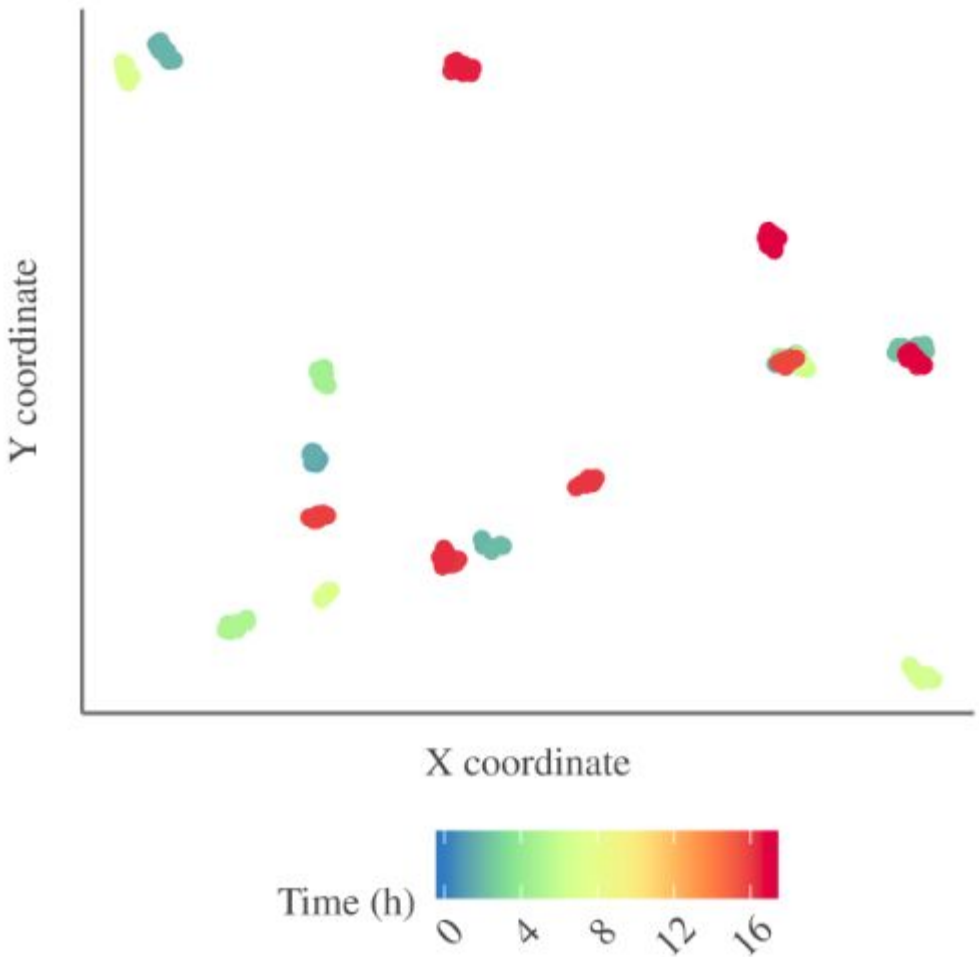
- $Distance(p_m, p_n) \leq D_{threshold}$ and
- $|p_n.T - p_m.T| \geq T_{threshold}$

Zheng, Y., Zhang, L., Xie, X., & Ma, W. (2009). Mining Interesting Locations and Travel Sequences from GPS Trajectories.

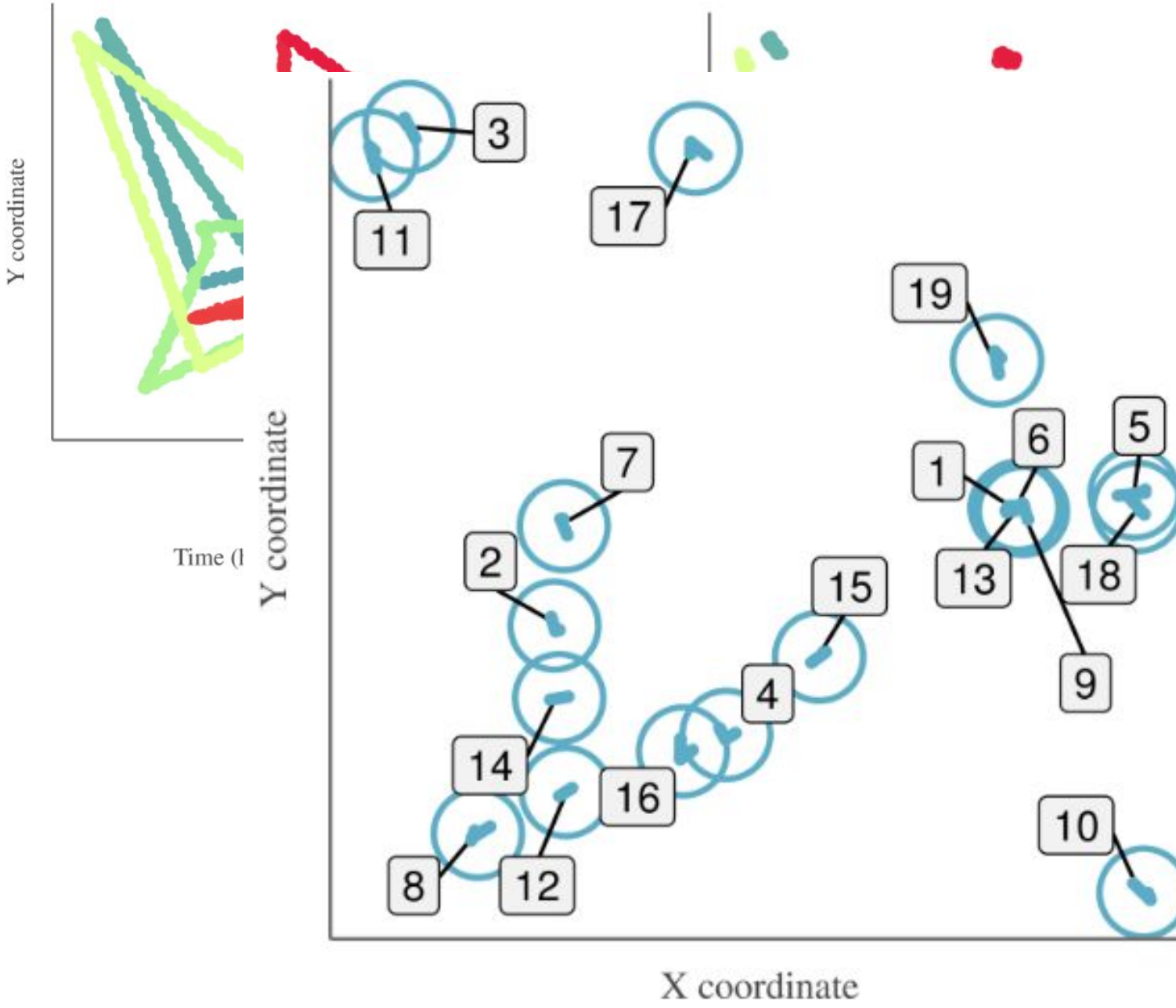


Raw data





Stay locations


Y coordinate
Time (
Y coordinate
3
11
17
19
7
2
15
1
6
5
13
18
9
4
14
16
10
8
12
16
X coordinate