ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Recognition of traffic jams using Hovering Data Clouds

presented by

Asha Nagendra

Under the guidance of Prof. Dr. Schindelhauer

Based on the paper by S´andor P. Fekete∗, Christiane Schmidt∗,
Axel Wegener†, and Stefan Fischer†

An overview

- A motivation for Hovering Data Clouds
- Definition of HDCs
- The role of HDCs
  1. in recognising Traffic jams
  2. in determining traffic density
- More potential applications.

Ants perform : A typical example For a complex system

Pheromones!

Asha Nagendra

Image at http://glutenfreekidsrock.blogspot.com/2009/09/high-hopes-even-ant-colonies-start.html

Complex system:

1. Self organisation

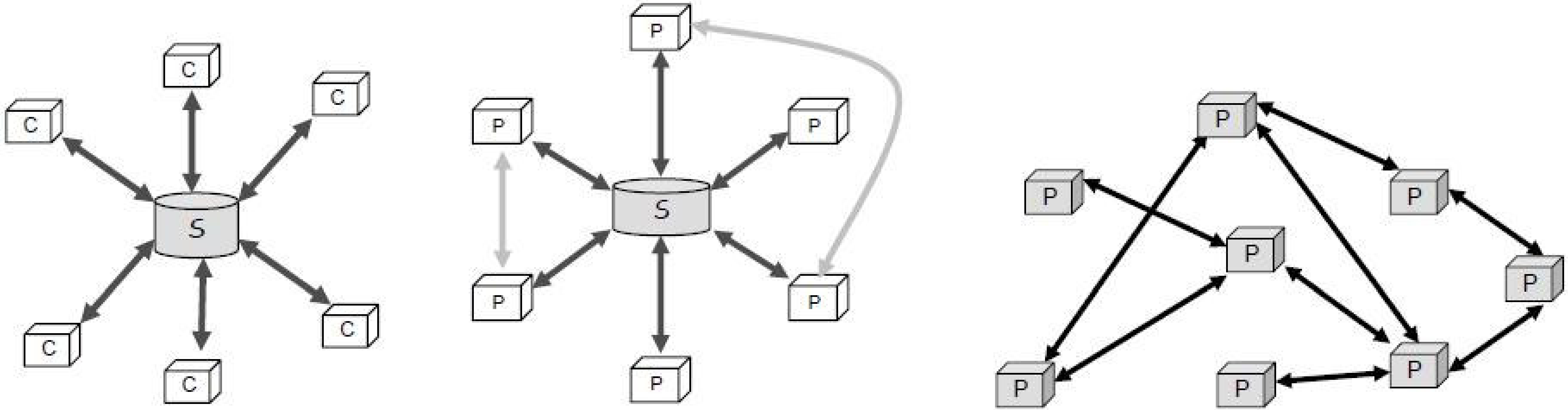
2. Self optimisation

3. Coordination

The means to self organise:

Communication

The paradigm shift



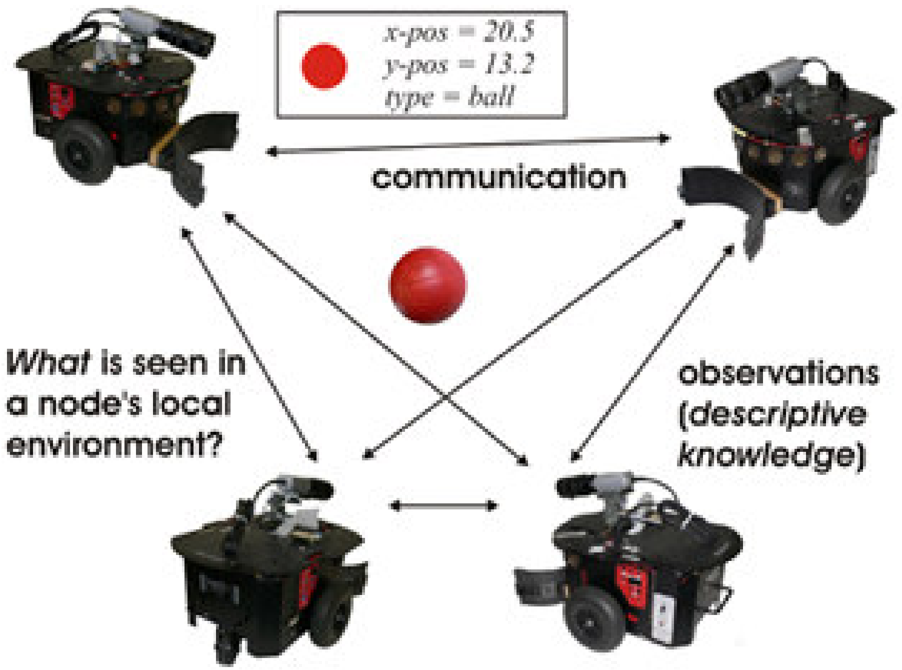
(a) Client/Server

(b) Hybrid

(c) Peer-to-Peer

Intelligent distributed systems:

An example



Robots exchange observations

Image at http://cis-research.de/node/122

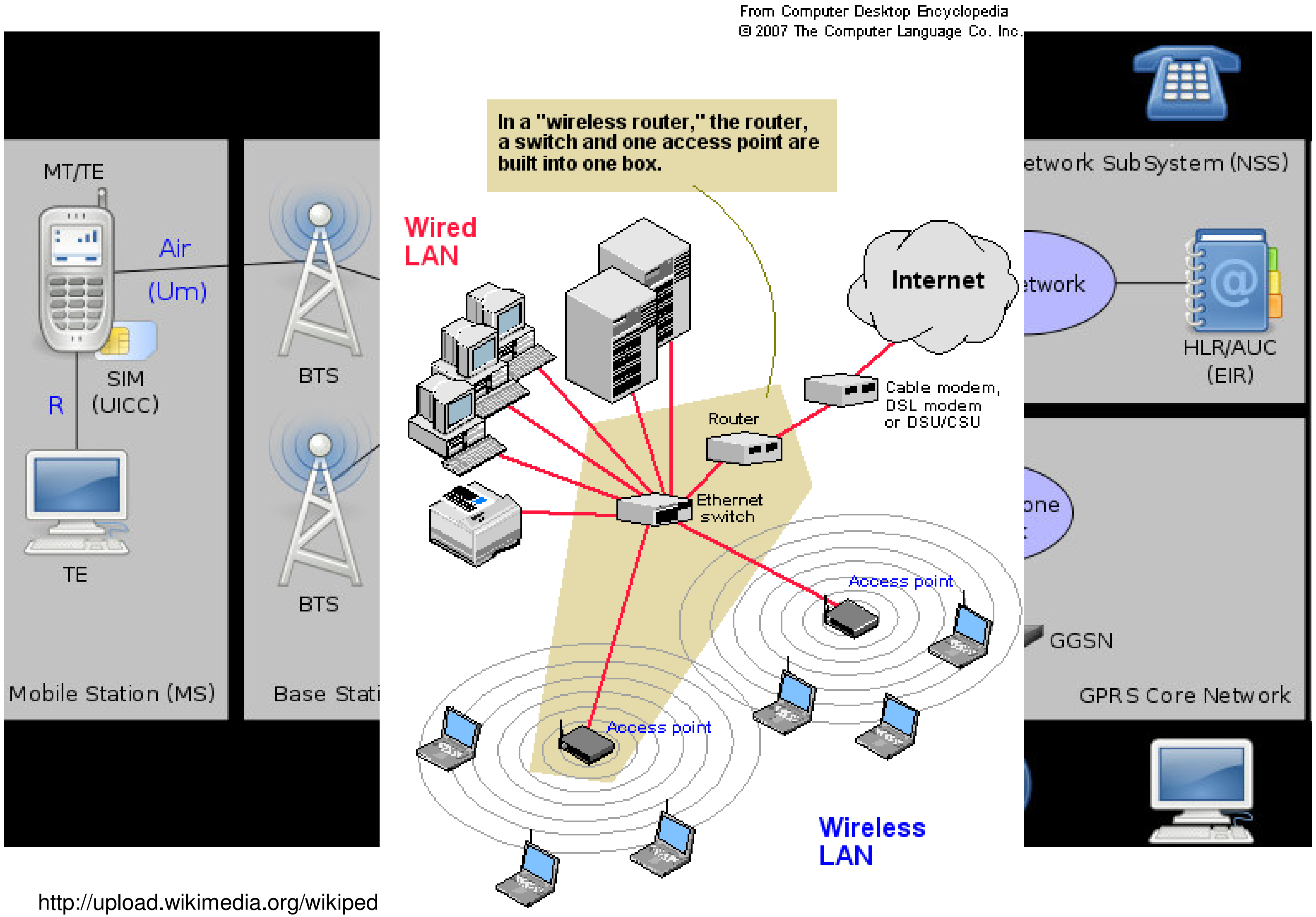
Scenario considered:

Computing paradigm:

Organic computing

Asha Nagendra

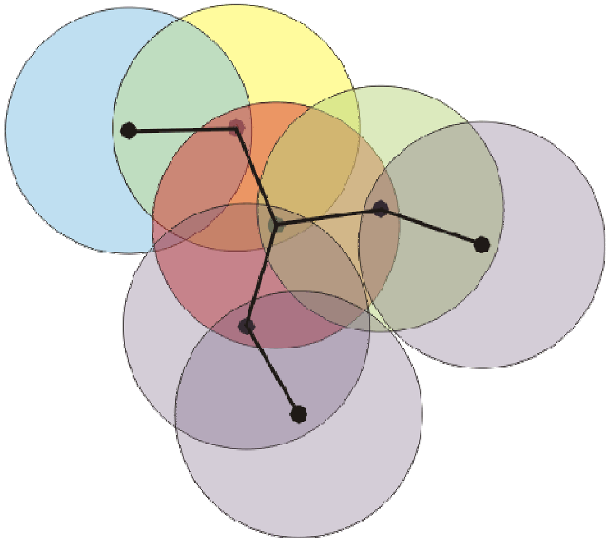
GSM and wireless LAN



http://upload.wikimedia.org/wikiped

Image at www.yourdictionary.com/computer/access-point

Ad hoc networks

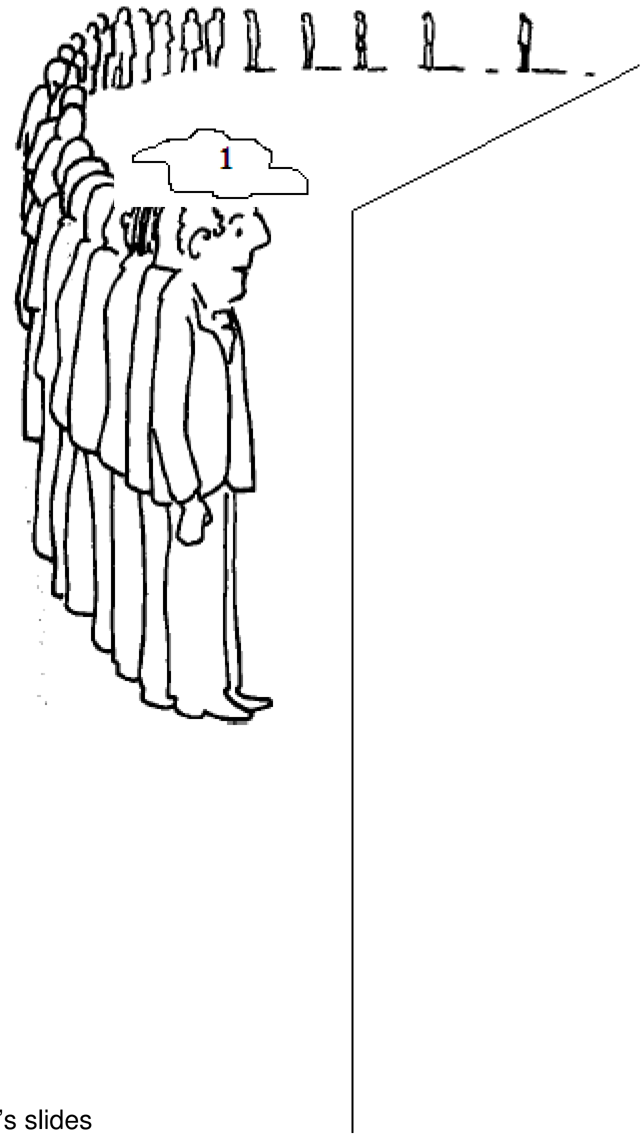


Asha Nagendra

The information system for such data exchange

Image source at www.themaclawyer.com

What is hovering data cloud?



A part of the image from one of Dr Malte Helmert's slides

HDCs in traffic jams

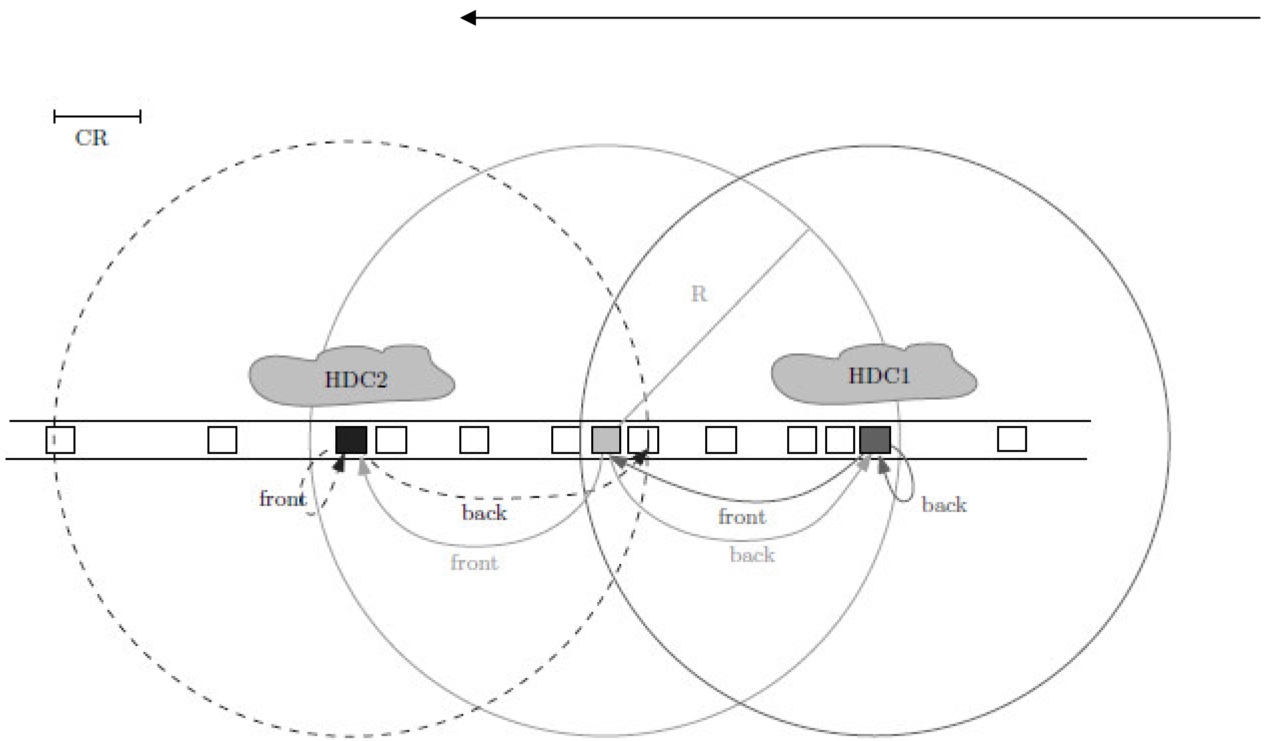
- The structure self-organizes with the onset of a traffic jam, and it ceases to exist when the jam disappears.

- It is located at a useful virtual location, which is defined by the traffic jam, e.g., its back.

- The structure continues to exist, even as their current carriers move or change their role.

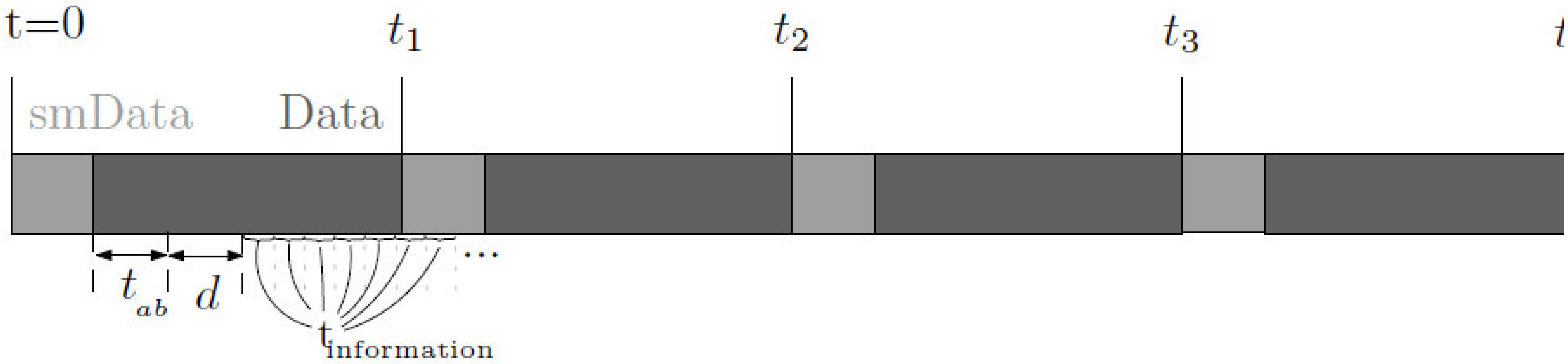
- It contains up-to-date information that describes the traffic jam.

Traffic jam in a single lane and data clouds



Example for the two HDCs

Time slots and timer



A discrete sequence of time slots, ti, ti+1, ... is considered.

The interval between to consecutive time slots is divided into two subintervals:

- A small interval (smData)
- A bigger one (Data),

In the end of smData (on-Timer(smData)), the timer for Data is initiated and vice versa.

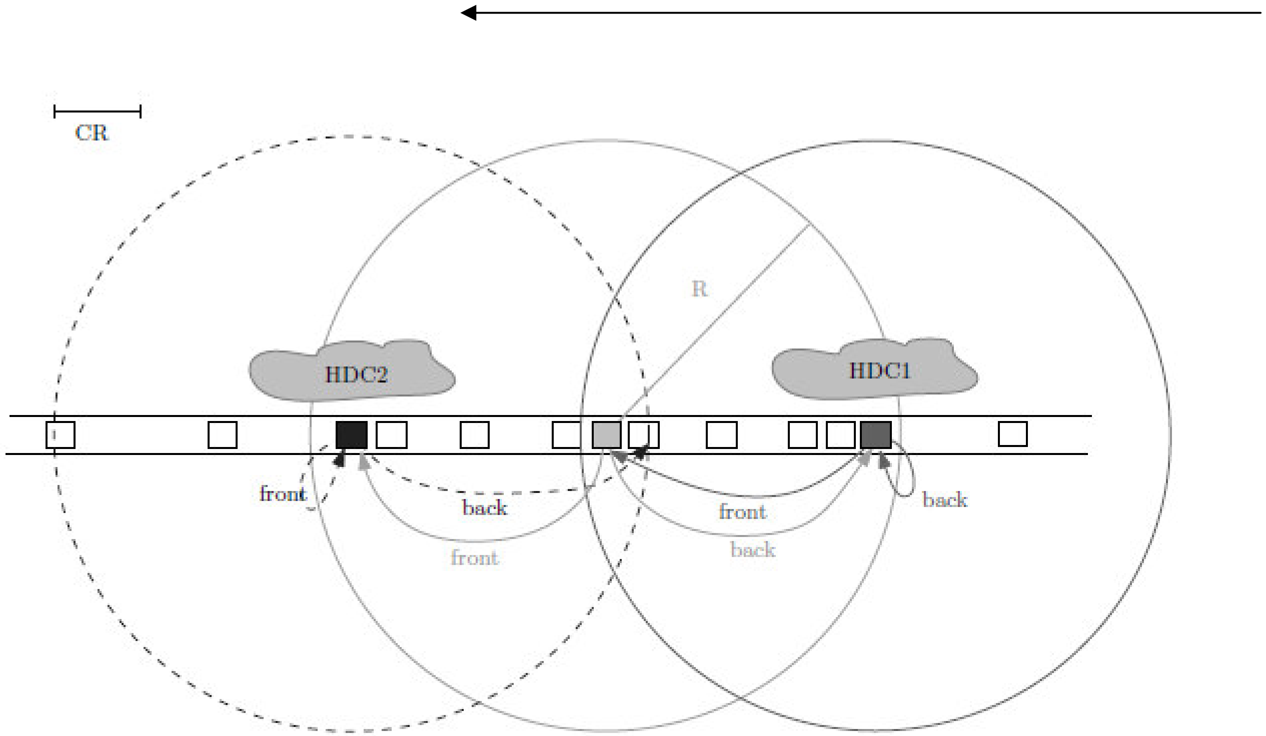
onTimer(smData)

The current processor position and its velocity is only broadcast within CR.

onTimer(NewMessage)

- If a processor receives such a message which is broadcast in onTimer(smData), it is processed here after an additional delay of d.
- In case a processor receives such a message from ahead, a variable representing the neighbor at the front, p is set.
- Analogously, a message from behind results in setting q equal to 1.

Traffic jam in a single lane and data clouds



Example for the two HDCs

Asha Nagendra

onTimer(Data)

- For sending more data (position, velocity, p, q) in a wider range, several cases are distinguished.

- Only if the processor is participating in a HDC or if it is neither caught up in a traffic jam (t_i) was so before but (t_{i-1}) below a certain velocity, it will send such a message.

- This helps in reducing the amount of transmitted messages.

How does a processor become a congestion participant?

- A processor receives data messages from its surrounding processors within R.

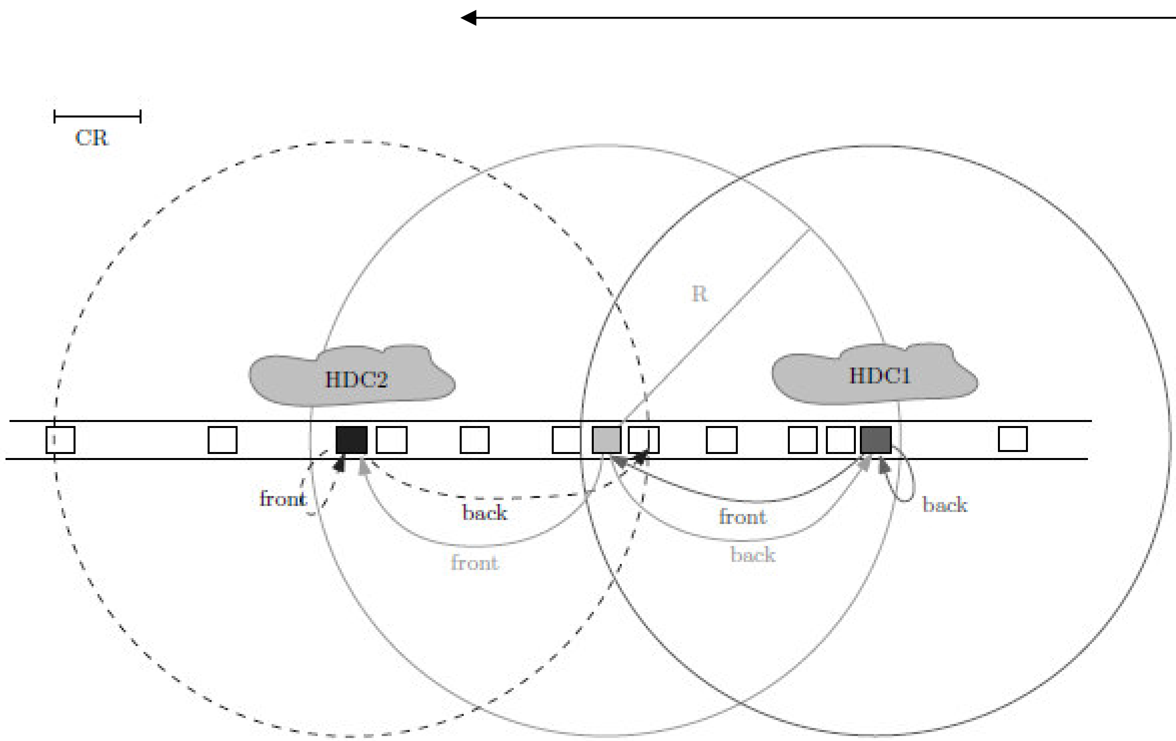
- The data of each such processor is stored in a matrix, env.

- Is the sender close (less than CR) to the receiver?
  Is the velocity < 60 km/h?
  If so, the back is computed from the position of the two processors and the previous back.

- Furthermore, the processor becomes a participant of the traffic jam.

- Similarly, for all processors in env, it is checked whether they are located close to the sending processor and fall below a velocity of CV. In both cases a counter for the congestion is incremented.

Traffic jam in a single lane and data clouds



Example for the two HDCs

Asha Nagendra

When is congestion invoked?

- If the counter was incremented.
- the processor lies close (less than rHDC) to the back,
- the back has no following neighbor (thus,it is really the back)
- all messages from processor within the range of R were received,

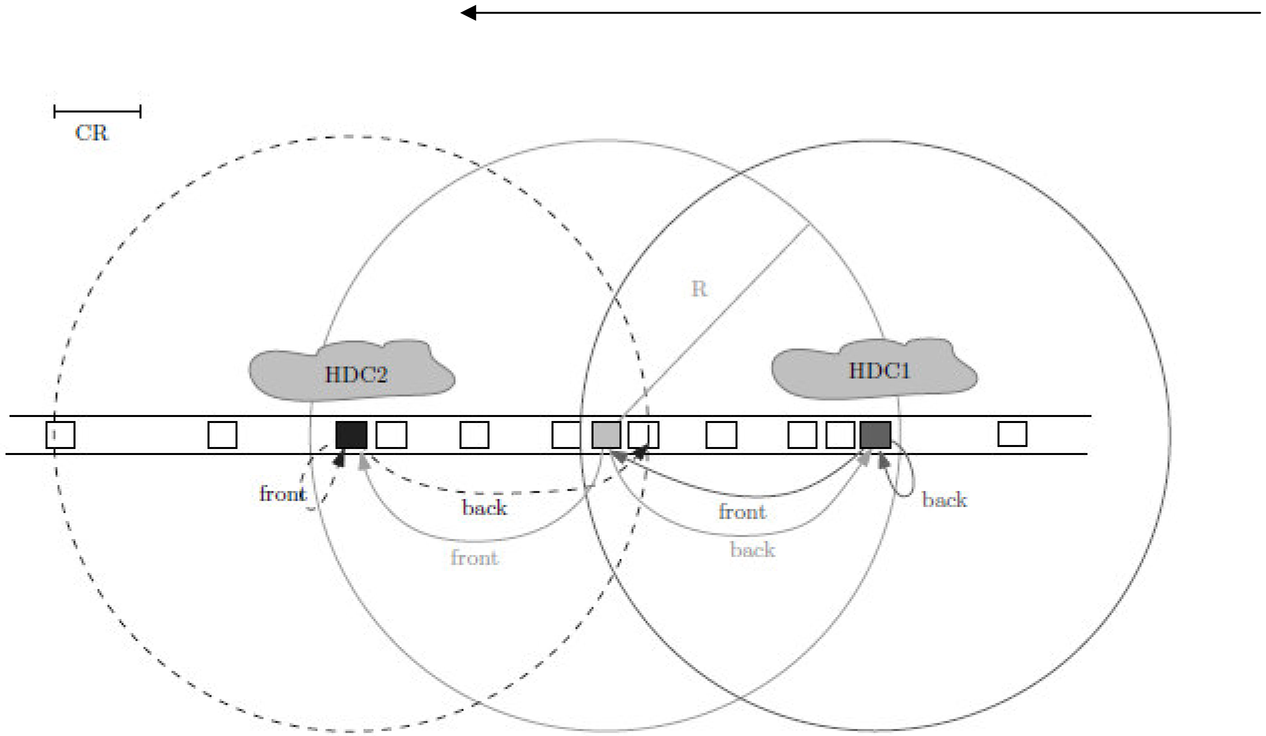
The front of the jam is treated analogously; **CongestionAhead** is invoked here.

If the HDC at the back has yet to receive information from the HDC at the front, then the position of front is set to the position of the most advanced processor within R.

Congestion:

- The status of a processor is maintained in Congestion: if it is active, a timer for Information is set, i.e., as long as the processor is active,
- the position of the HDC at the back of the jam,
- the position of the HDC at the front, and
- the current speed of the back are broadcast.
- Only if the position of the back HDC > current processor location, the processor continues to broadcast, and processors approaching this position become *joining*.

Traffic jam in a single lane and data clouds



Example for the two HDCs

CongestionAhead:

- $status_2$ is updated.
- If the processor is *active,* the timer for AheadInfo is set i.e.,
- The position of the front HDC is broadcast regularly, as long as the processor is active.
- When such a message hdc_distance is processed, the back HDC variable *front* indicating the position of the front HDC is updated for an active processor: inactive processors between front and back HDC pass on the message towards the back.

Thus, the messages are broadcast to:

- relate the positions of the processors (messages broadcast in onTimer(smData), onTimer(Data), processed in onTimer(NewMessage)),
- transmit the information of the HDC at the front of the traffic jam to the one at the back (messages in on-Timer(AheadInfo)),
- transmit the information of the back HDC to following cars (messages in onTimer(Information)).

Determining traffic density using data clouds

An HDC,

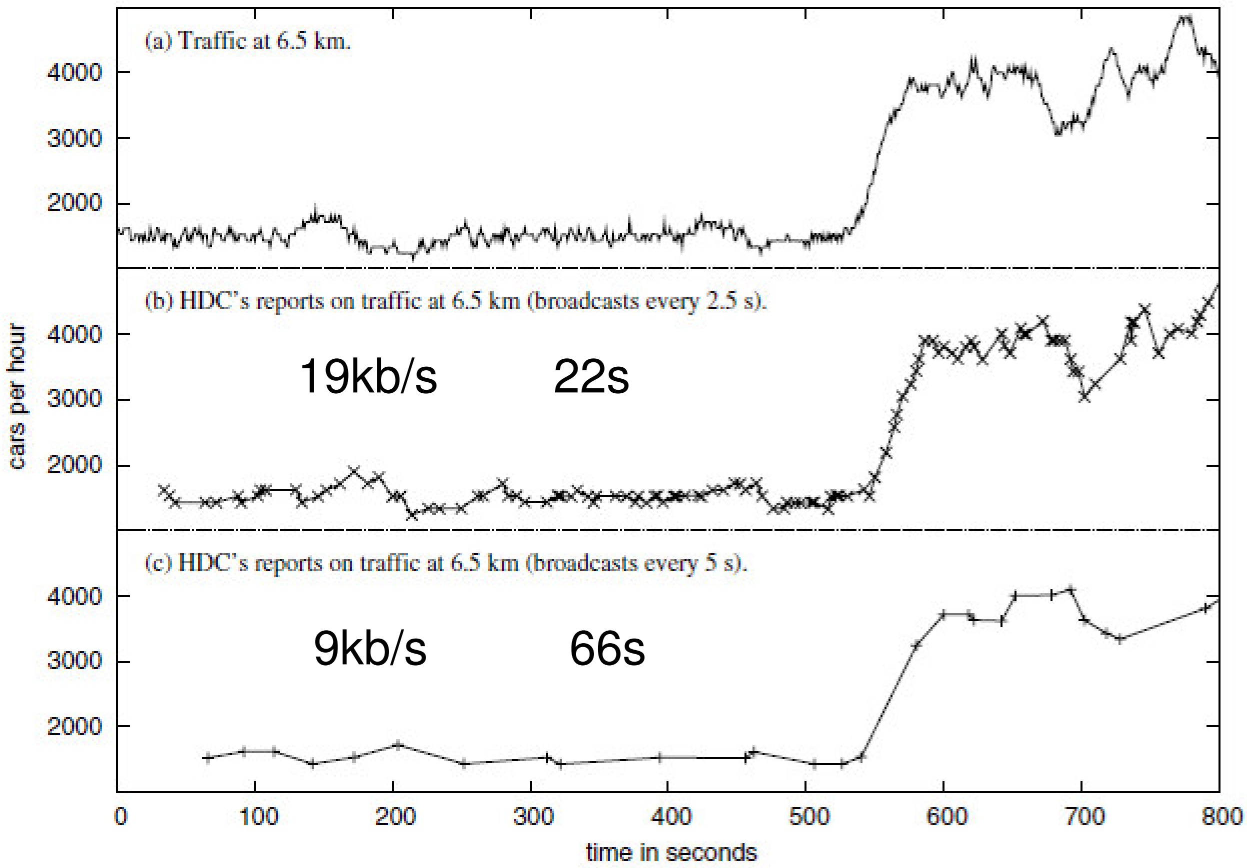
1. captures the events and characteristics, which arise with the onset of the traffic jam.

2. has a distinct origin defined by a center and an expanse.

(Both can change over time, accounting for the represented event.)

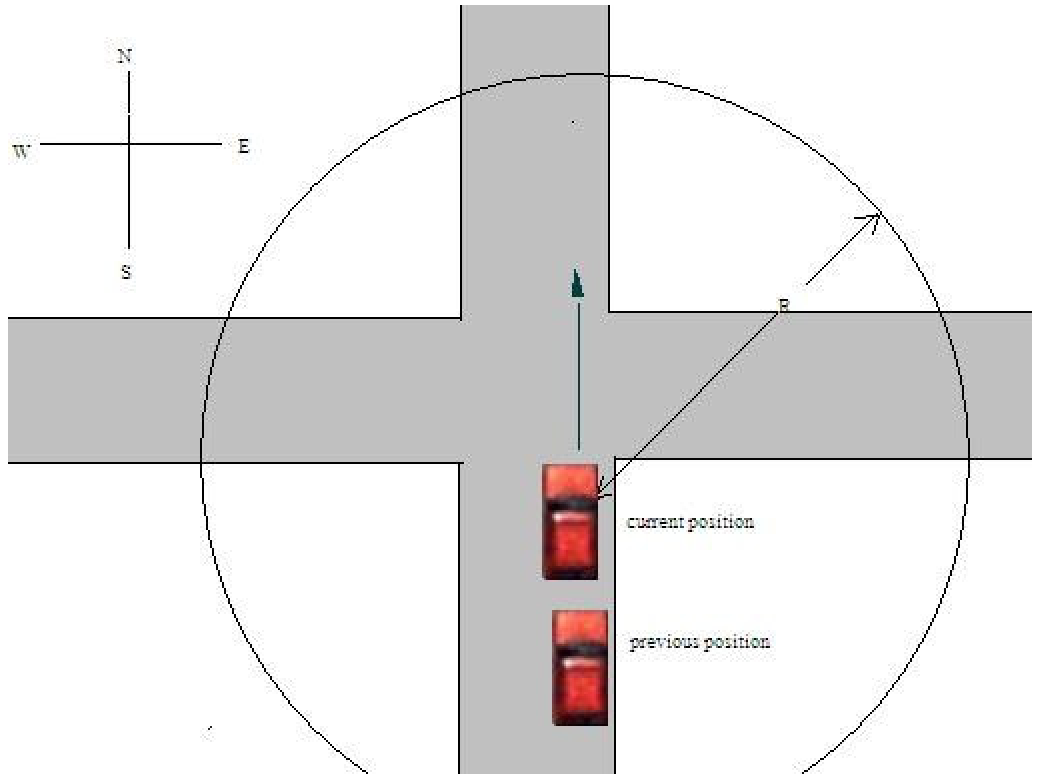
Traffic density can be described with motionless HDCs.

Simulation results



Comparison of real car densities reported by HDCs over time

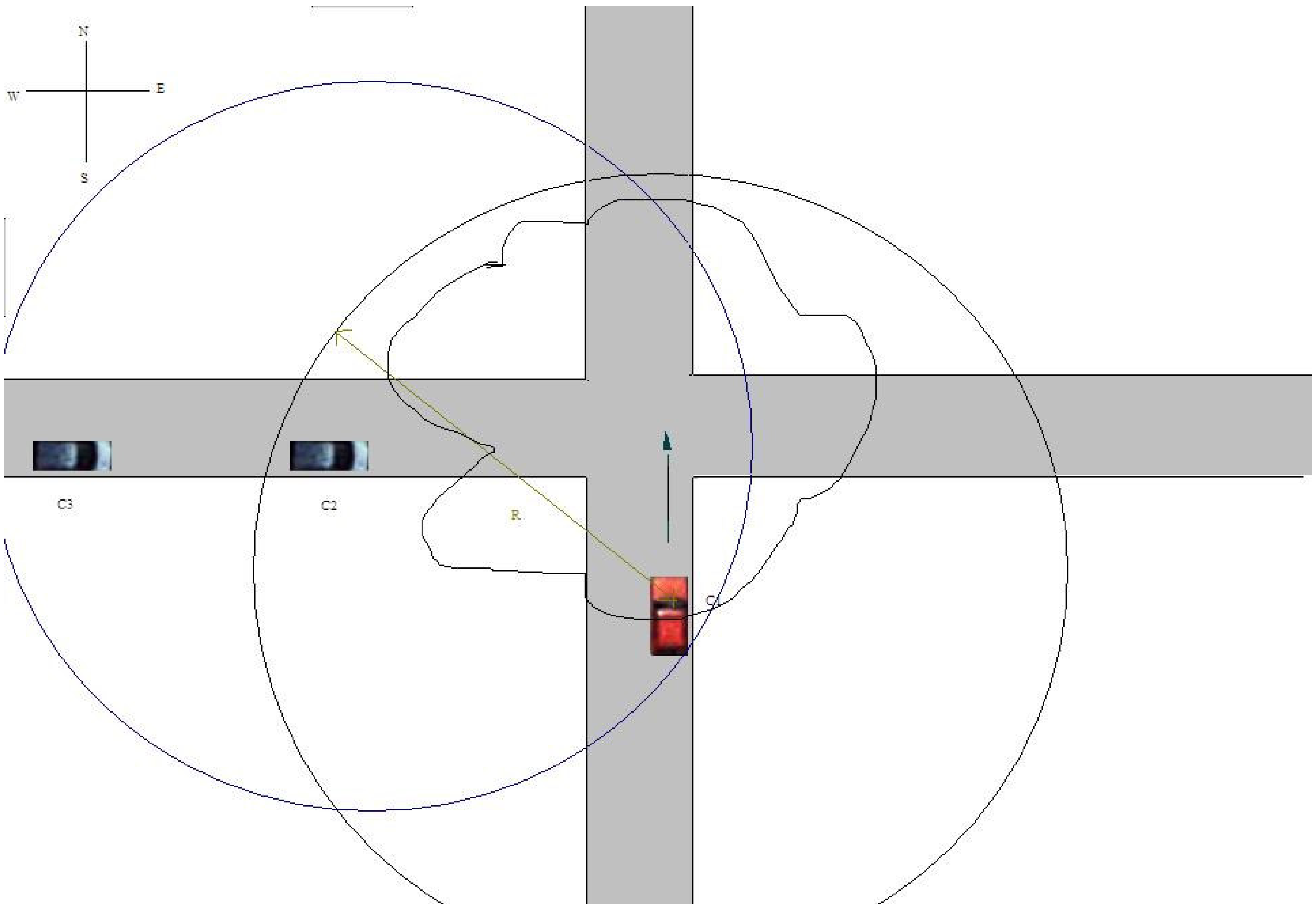
Traffic signals with HDCs



No contention scenario

Asha Nagendra

Traffic signals with HDCs



Resolving contention

Asha Nagendra

Traffic signals with HDCs

What if the count is equal?

Just compare the avg_vel

if($\text{count}\_lane_{SN} < \text{count}\_lane_{WE}$)

Priority $\_lane_{SN}$ = high

Pass on T_Stop to the following neighbor

$lane_{WE}$ waits till timeout, T | no_message_recpt

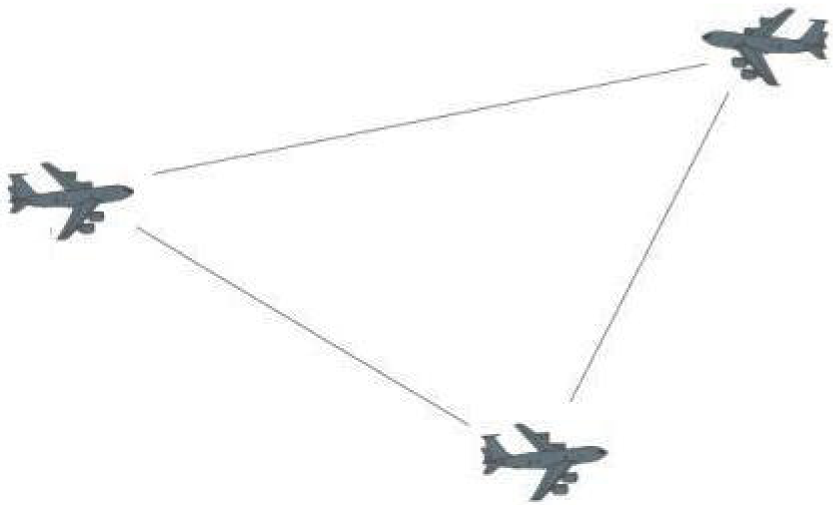
This avoids unnecessary wait time

Information exchanged:

1. Driving direction
2. Position
3. Velocity
4 T_Stop

Future applications of HDCs in

Airborne ad hoc networks



Asha Nagendra

Conclusions:

- The definition of HDCs need not be restricted to traffic jams.
- They can take their place in any complex system when there is an effective way to communicate and compute.
- This new approach works out better in solving problems encountered in daily life in an adhoc, bottom up manner.

References:

1. http://cis-research.de/
2. http://www.auto-nomos.de/.
3. Emre Cakar, Moez Mnif, Christian Müller-Schloer, Urban Richter, and Hartmut Schmeck. Towards a quantitative notion of self-organization. 2007.
4. Sándor P. Fekete, Christiane Schmidt, Axel Wegener, and Stefan Fischer. Recognizing Traffic Jams with Hovering Data Clouds.
In Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, ISoLA 2006., pages 198–203, 2006.
5. R. Steinmetz and K. Wehrle. P2P Systems and Applications. In LNCS 3485, pages 9–16, 2005.
6. Axel Wegener, Elad M. Schiller, Horst Hellbrück1, Sándor P. Fekete, and Stefan Fischer.Hovering Data Clouds: A Decentralized and Self-organizing Information System. In H. de. Meer and J. P. G. Sterbenz (Eds.): IWSOS 2006, LNCS 4124, pages 243–247, 2006.
7 by HQ ESC/NII for the USAF Airborne Network Special Interest Group. Airborne network architecture. Version 1.1.