Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Lehrstuhl für Rechnernetze und Telematik

WS 2009

Seminar work

Recognizing Traffic Jams with Hovering Data Clouds

Sándor P. Fekete, Christiane Schmidt, Axel Wegener, and Stefan Fischer

Asha Nagendra

4. März 2010

Under the guidance of Prof. Dr. Christian. Schindelhauer

Abstract

A complex system is capable to self-organise when the entities it is composed of, are able to communicate amongst themselves. The need for hovering data clouds in the context of such complex systems is explained. The recognised applications of hovering data clouds are discussed. The definition of data clouds is made more generic and an attempt is made to come up with an algorithm to signal traffic using hovering data clouds. Furthermore, potential applications using HDCs are briefly introduced.

Inhaltsverzeichnis

1 Introduction

When a person is walking on the pedestrian path and he sees another person nearing him. i.e., when they are approaching each other, a kind of implicit communication takes place between them to let each other their own personal spaces when they cross one another. For a few seconds, this *data exchange* would take place and they would take care that they don't collide. After they cross each other, they would no longer remember about the previous position of the person or any such detail. The *data cloud* which once existed thus ceases to exist. If we think of all the people walking on a pedestrian lane, they would all be walking at different speeds, yet following some kind of protocol when we see them all as a group. They have the capability to organize themselves. Similar behavior would be when people wait in a queue. They implicitly number themselves. The data, i.e., the number on the person at the forefront would get transmitted to the one following him as the queue moves. The data would be there though the carriers change. The behavior exhibited in a group is rather different than how it would have been when each person is given a dedicated lane to walk on or when each person is given a service counter. There are many such examples as to how a system can self organize. But these are living beings and this is the natural quality exhibited by complex living systems. An example for a complex system not composed of living parts is traffic. They are now even empowered to communicate if they are equipped with wireless devices. So this can be exploited to mimic the *lifelike properties* and allow an entity to exchange data with its neighbors so that they can self organize. The authors suggest that a traffic jam can be recognized with the help of such “Hovering data clouds” [7].

1.1 Complex System

“A complex system is a system composed of interconnected parts that as a whole exhibit one or more properties not obvious from the properties of the individual parts.“[2]

There is no external control or a centralised control for a complex system. The entities composing a complex sytem are capable of interacting, contending and co-operating with each other, thus controlling the system implicitly with constant feedbacks between each other and in a decentralised, distributed way. The overall behavior of the system is the outcome of many decisions made by corresponding individual entities constituting it. The decision of one entity is always based on the decision made by some other entity with which it is competing or cooperating. This is also the essence of game theory. [1]

In the next section, the paradigms to control a complex system (esp.,for data storage and transmission) are discussed and the most suitable paradigm is opted for.

1.2 The Paradigm Shift

Let us consider the very popular internet as an example and the paradigm shift over the years.

One of the earliest form of data storage is the centralised one with one server and many clients it can correspond with. Client server model is not the most optimal one since the chances of single point of failure is more. It can also not be scaled well when there are too many clients. To overcome the defects of this architecture, hybrid models came into picture. Inspite of having a central server, communication amongst the clients without involving the server proved to be slightly more robust than a pure client server architecture. In cases of failure, the clients can elect a new server and may continue to function. Even then, loss of data may not be completely avoided. The peer to peer system is more applicable in a distributed system and there are equal sharing of responsibilities. The Peer-to-Peer filesystem opened a whole new world to a distributed approach. “It clearly reflects the paradigm shift from coordination to cooperation, from centralization to decentralization, and from control to incentives“ [8].

One more step ahead of distributed computing is the organic computing. Here, the systems can exhibit all the properties of a complex system effectively i.e., “self-organisation, self-healing, self configuration and self optimisation“ etc.[6] They can also adapt to the environments which change over time. The control of a complex system using organic computing does not rely on a separate intelligence but is inherent within the sytem. Clearly, this is custom made for ad hoc, highly decentralized systems. Now we have wireless networks, mobile participants forming an ad hoc network. MANET and VANET are the best examples. Following a centralized approach to enable communication would rather be a paradox in this case.

The information system which helps in coordination between such self organizing entities is discussed in the following section based on [9].

2 Hovering Data Clouds: A Decentralized and Self-organizing Information System

As per the authors, a novel way to enable data exchange between ever changing carriers i.e, the cars in situations like traffic jams is by the Hovering Data Clouds (HDCs) [7]. They exist only when the need arises. We shall see the applications of HDCs as described in [7] and [9] in the following section.

2.1 Recognised applications of HDCs

2.1.1 Determining traffic densities

Hovering data clouds in the context of traffic jams are responsible for capturing the events which occur during the same and characteristics which pervade till the jam exists, and they arise with the onset of a traffic jam. At any point of time, a hovering data cloud has a distinct origin defined by a centre and a the propagation range. Both can change over time, accounting for the represented event. Static dataclouds are sufficient to describe traffic density. [9].

The car nearest to the centre of the HDC acts as the initiator and begins the forwarding process to determine the traffic density. The HDC in this case has a datastructure for storing the statistics of the traffic. Every passing car recieves this information (the location) from a prior carrier. Thus the cars forward HDC messages towards the cars which are approaching the HDC i.e, to the potential participants. Such forwarding of messages is bound within the range of the HDC and the origin, which in turn depends on the underlying traffic event which gave rise to the same.Common underlying events are aggregated at the HDCs to save some bandwidth and thus also helping with the computation. Only complex systems can achieve the desired behavior by exploiting the property to self organise since the set of hosting nodes change dynamically over time. [9].

Fig. 1 depicts the simulation results for stationary events that monitor car density on predefined locations as per [9]. The authors base the communication model on the standard IEEE 802.11. They set the communication range to 250m. The variations of the car density are produced and compared to the propagated and received HDC messages. The main purpose is to measure how close to reality the predicted car density is, the propagation delay and network overhead incurred by the HDCs. Since every message is broadcast, the overhead is typically high when there are too many participants. On increasing the update interval from 2.5s to 5s, the bandwidth consumption reduces from 19kb/s to 9kb/s, the propagation delay increases from 22s to 66s. At least one parameter needs to be compromised upon. The authors claim that the existing ad hoc approches with unicasting do not meet factors such as scalability though the overhead on the network would be much lesser. But multicasting is an open option.

Broadcasting is not the best approach to determine traffic density. A delay ranging from 22s to 66s is a bit high but works fine when the purpose is only to detect traffic density. The results are promising.

To recognise traffic jams and regulate them, a propagation delay to this extent is not acceptable. Hence the messages are broadcast only within the congestion range. This reduces the overhead on the network.

We shall discuss the algorithm for the recognising traffic jam using HDCs as per [7] briefly in the following section.

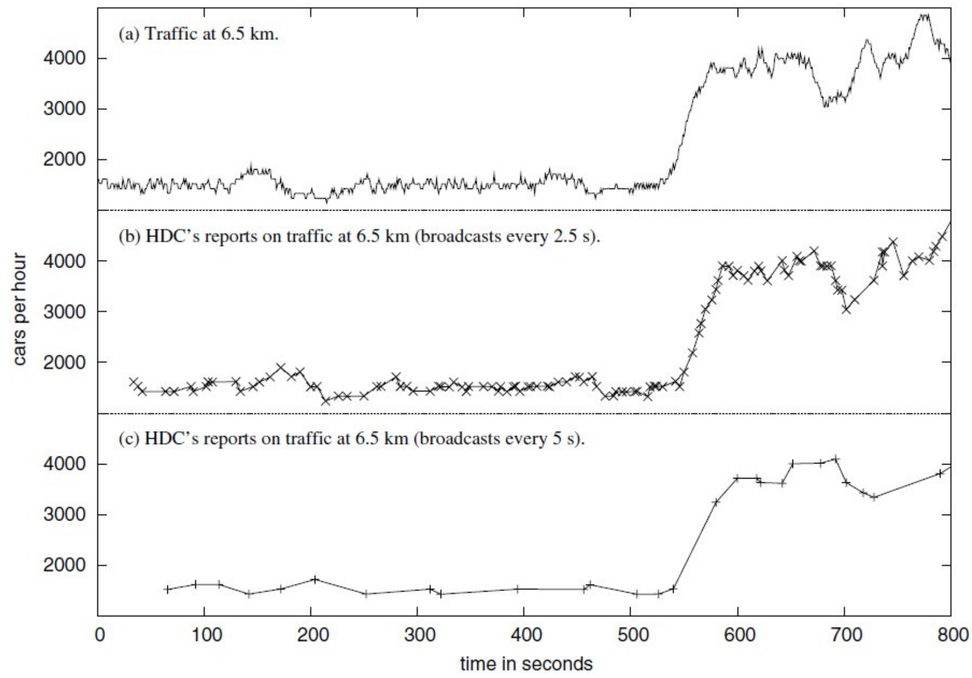

Abbildung 1: Car density over time. Comparison of real car density and that reported by HDC messages.

2.1.2 Key concept of the algorithm to recognize traffic jams

The authors consider a single lane highway to keep it simple. The algorithm deals the recognition of a traffic jam which can serve as a stepping stone towards regulation of the same.

Hovering data clouds are described to have the following properties in [7].

1. The structure organizes itself when the traffic jam emerges, and exists only until the jam exists, disintegrates as soon as the jam disappears.
2. It is located at the front or the back of the traffic jam since these are the obvious useful locations when we consider a queue in general.
3. The structure prevails, even as their current carriers move or dynamically change their role over time.(the carriers here refer to the GPS enabled cars.)
4. The information within the structure is updated regularly.(The new back and front of the queue are recomputed regularly.)

The data clouds at the front and the back of the queue are HDCs. See fig. 2. The complete algorithm is in [7]. A discrete sequence of timeslots is considered. Each interval between two consecutive time slots is divided into two subintervals:

1. A small interval for broadcasting the velocity and position of the current processor only within the congestion range. (This range is not specified by the authors. It is just described as a function of the current velocity of the vehicles.)
2. A larger interval for receiving the messages broadcast in the small interval and also to process these messages.

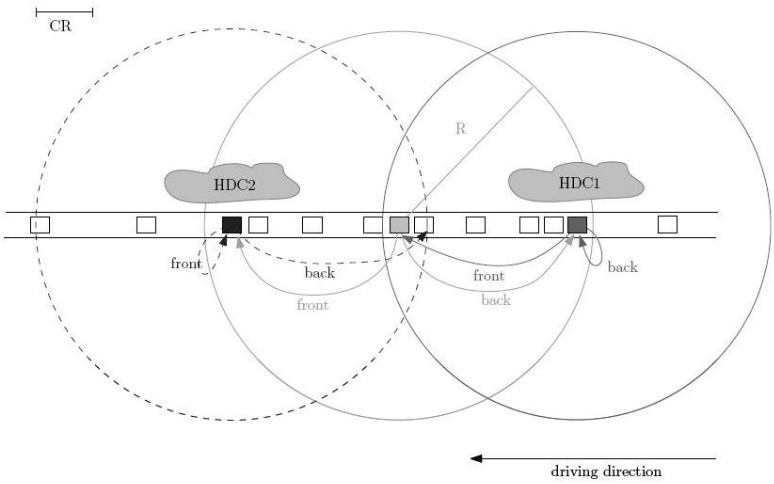

Abbildung 2: Example for the two HDCs as in [7] : different cars (rectangles) determine front and back

Upon reciept of such a message by a processor, it does not immediately process it, but after some delay. Until all the processors in the range 2R start the larger time interval, it is not processed. The respective variable is set when it recieves such a message from one of its immediate neighbors (the car in the immediate front or the immediate back). A sender within the congestion range and velocity lower than a threshold becomes a congestion participant. A counter is incremented to account for the number of participants. Also the new back is computed taking into account the new participant (analogously the new front is computed). If the new participant is the real back of the queue (at the extreme rear end), then a method called congestion is invoked.

Congestion If active, a timer is set for Information, i.e., until the processor is active, current velocity of the back, the position of the front HDC and the back HDC, are broadcast. The processor continues to broadcast only if the position of the back HDC is more than the position of the current processor. The cars approaching this position reset their status variable to joining.(The authors have three status variables namely idle, active and joining for marking the status of the processors)

Analogous to congestion, the processors which are not active help in passing messages between front and back HDCs in the method **CongestionAhead**.

Hence the messages are broadcast in order to:

1. locate the relative positions of the processors.
2. pass the information from the front HDC to the back HDC.
3. pass the information of the back HDC to following cars.(very relevant in case of traffic jams.)

3 Other potential applications of hovering data clouds

The definition of Hovering Data Clouds is tightly coupled to traffic jams in the above two cases in [7] and [9] since it is a part of the project Autonomos [4].

Ideally, this should not be the case. Hovering data cloud can be given a more general definition and then this can be extended to each application instead of making it application specific at the first place. In any Mobile ad hoc Network or vehicular ad hoc network or any system which is complex, adaptive and composed of entities which are capable to communicate and self organize, an information system which can be used to transmit data for the coordination can be called as Hovering Data Cloud. “Data Cloud” since it exists only during the situation which needs to be controlled for the proper functioning of the system and “hovering” since it lingers only till the situation is controlled. It is also not mapped to a specific set of entities. The carriers of the data keep changing continuously.

3.1 To signal traffic

Traffic signal is a part and parcel of traffic control. At one point of time, we had policemen standing in the middle of an intersection manually signaling the traffic. We now have traffic lights and every day millions and billions of vehicles wait on them. Is the centralized signal the best way to gain coordination? Many a time, the wait time seems unnecessary when there is no traffic on the other roads which meet at the intersection. This wait time can be avoided if things are more decentralized. A hovering data cloud persists only if there are contenders approaching an intersection.

Before thinking of regulating traffic at an intersection, to know that there is an intersection ahead is an important aspect. One way to detect an intersection is through the electronic map in GPS enabled cars. We shall stick to the assumption that the vehicles know that there is an intersection ahead before we tackle the problem of regulating the traffic at the intersection.

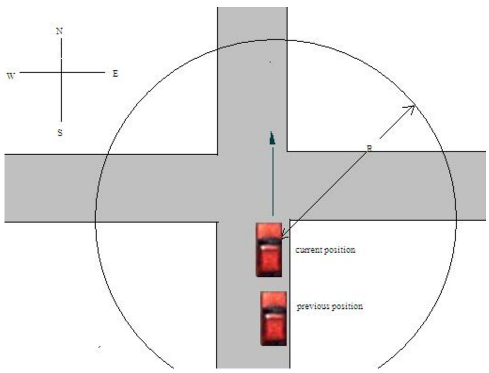

Abbildung 3: No contention scenario

In fig. 3, an intersection of road towards which a single entity is approaching can be seen. Each car approaching an intersection tries to detect another car within its range by broadcasting its id, velocity, position. If it receives no message or if it recieves messages only from cars driving in the same lane, it means that there are no contenders, and it is free to cross the intersection with no waiting time.

A HDC is formed at the intersection only if there are contenders to the traffic moving in one lane.

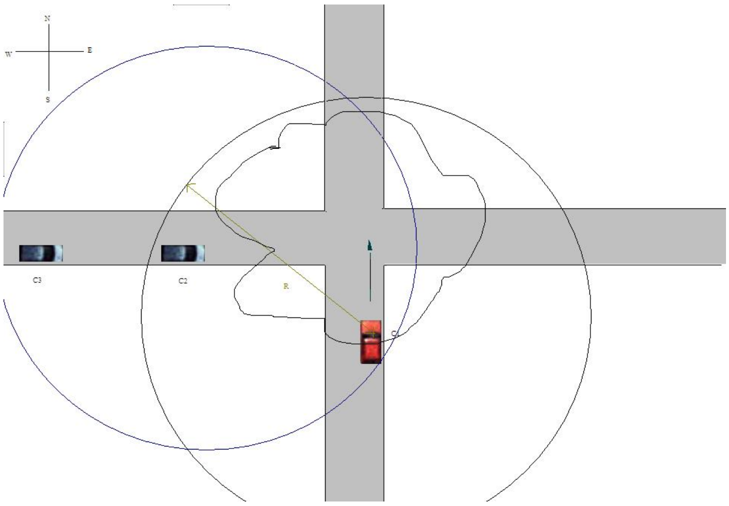

Abbildung 4: Initial contention Scenario

In fig. 4,$lane_{WE}$ indicates the lane where the driving direction is from west to east. If the car C_2 receives a message from C_1, it broadcasts its velocity, position and driving direction. If the (count of received messages at C_1 from South to North lane + 1) is lower than $lane_{WE}$), fix the priority for the cars moving in $lane_{SN}$ till a timeout interval, T. T_{stop} is passed on from every car moving at the intersection to the one following it. At T_{stop} the cars in $lane_{SN}$ SN halt and T_{start} is reset. If the cars at $lane_{WE}$ stop detecting cars within range R from $lane_{SN}$ before T, they get to move on before the timeout.This avoids unnecessary wait. The priority is shifted to the cars moving in $lane_{WE}$. This contention resolving mechanism will not work if the number of participants in both the directions is the same. So we have another parameter, the average of the all the received velocities of one lane. The same concept can be extended when there are cars moving in all four lanes. After fixing up the priority once, it just gets shifted in a round robin fashion depending on whether it is left side or right driving. Here right side driving is considered. The arriving contenders should only see if the recieved velocities are zero which accounts for the cars in waiting lane. In that case,they have to wait for their turn.

3.2 For decentralized coordination in airborne ad hoc networks

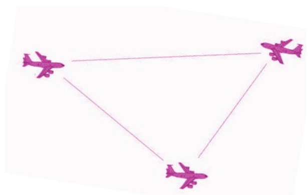

Abbildung 5: Airborne Ad hoc Network

Data clouds can further be extended for air borne ad hoc networks. Air borne ad hoc network is an active research area and the complete shift of paradigm towards decentralised approach is really challenging in this regard . However if adhoc networks are formed between air borne entities even at those high velocities, then HDCs will undoubtedly play a major role for co ordination and self organisation, instead of every node contacting the ground station [5]. This can help in collision avoidance, avoidance of adverse weather conditions and in military applications as in [3]

4 Conclusion

A data cloud is just a new way of data storage or an information system which does not depend on specific carriers. This is a generic concept and they need not be restricted to traffic jams only. Traffic signal can be implemented in an ad hoc manner using data clouds using the algorithm which we have come up with in this work. It is decentralised and does not depend on a specific set of carriers and is easy enough to be implemented with a single HDC. It has its advantages over the traditional traffic lights. Many more such application areas can be identified as in the last section of this work. This shall pave new methods of bottom up, adhoc appraoch to common problems which have been solved so far.

Literatur

[1] http://en.wikipedia.org/wiki/complex_adaptive_system.

[2] http://en.wikipedia.org/wiki/complex_system.

[3] http://www.ausairpower.net/te-gps-guided-weps.html.

[4] http://www.auto-nomos.de/.

[5] by HQ ESC/NII for the USAF Airborne Network Special Interest Group. Airborne network architecture. Version 1.1.

[6] Emre Cakar, Moez Mnif, Christian Müller-Schloer, Urban Richter, and Hartmut Schmeck. Towards a quantitative notion of self-organization. 2007.

[7] 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.

[8] R. Steinmetz and K. Wehrle. P2P Systems and Applications. In *LNCS 3485*, pages 9–16, 2005.

[9] 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.