Reports of the Department of Mathematical Information Technology
Series C. Software and Computational Engineering
No. C 1/2014

Evolutionary Cloud for Cooperative UAV Coordination

Michael Cochez Jacques Periaux
Vagan Terziyan Kyryl Kamlyk
Tero Tuovinen

University of Jyväskylä
Department of Mathematical Information Technology
P.O. Box 35 (Agora)
FI–40014 University of Jyväskylä
FINLAND
fax +358 14 260 2731
http://www.mit.jyu.fi/

ISBN 978-951-39-5729-2
ISSN 1456-4378

Evolutionary Cloud for Cooperative UAV Coordination

Michael Cochez Jacques Periaux Vagan Terziyan
Kyryl Kamlyk Tero Tuovinen

Abstract

This report is dedicated to considerations and perspectives for the strategy of using unmanned aerial vehicles and Cloud Computing in emergency situations in a Smart City environment. The aim is to provide inspiring insights in this highly complex problem from a technological and a tactical point of view. This specific case can be seen as a member of a family of similar problems, and hence the solutions proposed in this chapter can be used to solve a large set of related cases. The efficient use of the presented set-up would require to solve a time-consuming multi-objective optimization problem, which one could - at least partially - solve using some form of Evolutionary Computing.

Emergency situations, by default, are unpredictable and often involves incomplete information which might be supplemented over time. The changing situation creates an unpredictable complexity for the computational problem and therefore we will need to cope with bursts in the need for computational power. To overcome this problem we propose to perform the computation using Cloud Computing. This solution also improves the robustness of the overall system because it provides fail-over and content replication. In order to integrate and analyze the various measurements performed by the robots we suggest the use of Semantic Web technologies. However, the size of the sensing data can be grow to enormous sizes and we note that this problem can be considered as a Big Semantic Data problem. Based on our earlier work, we propose to tackle this problem using an Evolving Knowledge Ecosystem.

1 Introduction

What is Smart City? From the time of first buildings made by man up to this day, we have considered buildings as being static and stable structures, which provide local functionality, like shelter and warmth. They were constructed with a given purpose which, of course, could change slowly over time. However, the buildings in a city did themselves not have any direct interactions with - or understanding of - each other.

The revolution of information technology has changed this set-up dramatically. Today we can discuss about environments which have a consciousness, i.e., building and structures with some level of understanding about themselves and others.

Obviously, this understanding is artificial and based on algorithms developed by programmers, but will anyway change our conception about the place we life today.

Smart cities can be identified on many levels. The identification can be close to daily life, like a smart environment, mobility, and living, or more abstract like a smart economy and governance or even smart people. The practical focus of recent studies are the growing amount of information and data and how to analyze and use it. We regard a smart city as a large multilevel construction where the buildings, streets, shops, bridges, traffic lights and so on, can discuss, collaborate, make decisions, and perform actions with or without human intervention. Using conventional technology, like e.g. the Internet, it allows to objects a possibility to share and gain information from their environment. This knowledge can be harvest by using multiple sensors, like, for instance, cameras and thermometers. Combining and analyzing this jointly collected information could potentially enrich the sense of the situation for each individual object and allow more accurate responding. All in all, this could provide more cost-effective solutions and real-time services for human kind.

When we also consider moving sensors and actuators, like UAVs, the amount of collected data and possible actions will increase dramatically. Earlier, discussions were focused on static objects gaining capabilities by having some form of consciousness. Next, we can think about what the opportunities are when these objects would gain the ability to move. It is known fact, that static sensors and cameras suffer from their limited degrees of freedom. Further, special measurement equipment is, even today, still quite expensive and there is no way to insert them everywhere where needed. For that reason, in large areas, there is no possibility to cover the area densely enough for reliable automatic decision making. Therefore, there is an evident benefit when employing devices which carry the sensors around, like UAVs.

Worldwide there is a growing interest in the use of mobile robots and different types of moving sensors. Popular examples of applications of that type of devices are found not only in homeland security, border protection, and vessel tracking but also in logistics and crisis management (fires, accidents, floods...). Perhaps the most versatile type of devices are unmanned aerial vehicles (UAV). These devices, sometimes called drones, have a high degree of mobility, can carry a wide range of sensors, and are able to perform a wide range of actions in the environment using actuators.

In this paper, we discuss aspects of using a fleet of collaborative unmanned aerial vehicles (UAVs) to complete missions. Especially, we consider a possible strategy at how these devices can be employed during emergency situations in smart cities and how Cloud Computing technologies can be used to aid their working. The idea is to inspire the audience to take actual steps for practical implementation of the propose automated services.

For illustrative purposes we will use the following scenario:

- There is a big fire in the forests close to a major city. Firemen are fighting the

fire using helicopters and specialized UAVs. Further, cameras placed on other UAVs are used to observe the area. The observations of the UAVs are collected in a server which is hosted in the Cloud.

- At some point it starts to rain heavily and thanks to the water the fire gets extinguished.
- There is ,however, so much rainfall that the city is threatened by it. However, based on the readings from sensors placed in the rivers and observations made by UAVs, the people in the city are warned in time.

In the next Section (Section 2), we will give a broader description of the problem. Then we describe how the actions needed in emergency situations can be regarded as an optimization problem in Section 3. In Section 4 we will discuss the potential of using Cloud Computing as a resource of computational power. Next, in Section 5 we discuss how we use agents, which use a semantic model for reasoning and data storage, to help the UAVs to perform their tasks. The amount of data collected by the UAVs is of such volume, arrives at such a velocity and is of such nature that we can consider it a Big Data problem for which we provipose a solution in Section 6. In the last Section (Section 7) we will conclude and describe future research directions.

2 Emergency Management in Smart Cities

In this Section, we will focus on one challenging application area, namely, emergency management in smart cities. These situations, which usually evolve in an unpredictable manner, will require real-time optimization, and the deployment and coordination of a temporary, adaptive fleet of intelligent UAVs. There is also a need for a strategy which optimizes the Cloud-driven data collection and the distribution of data between populations, ground support and control centers.

Past studies have been mainly focused on the cases where there is only one vehicle without any dynamic cooperation (see for instance Gonzalez et al. (2009)). In some cases cooperation has been implemented using pre-planned routes and tasks. In a recent paper by Doherty et al. (2013), the authors acknowledge that in order to increase autonomy in heterogeneous UAV systems, there is a need for automatic generation, execution and partial verification of mission plans. We would like to augment this statement by saying that also validation of the plans is needed, i.e., it is not enough if the plan complies with a given specification, it must also be assured that the plan meet the needs of all stakeholders.

The robots or drones can use each others' observations to decide upon their own actions and hence disable some of their own sensors in order to save energy. Furthermore, a UAV squadron can be seen as a toolbox with several tools, where one can be used for the collection of information, the other will encrypt everything for secure communication, one can provide specific information for optimization calculations and yet another one can ensure communication with the Cloud. In our example scenario we have UAVs which are specifically designed for extinguishing

fire; this group of UAVs has specific physical properties which make them a perfect tool for the given task.

UAVs could also be used for preventive observations. Therefore, it is desired that a UAV can be instructed to limit how disturbing it is. This can be done by lowering its visibility and noise level, as opposed to the loud and visible activities which are likely during fast operation. In another situation, there could be a 'mother ship' with reloading capabilities and the ability to fly long distances. The number of options is enormous and the choice depends on the mission at hand.

The dynamic nature of this problem makes this system most interesting. There are plenty of possibilities for failure and a human programmer cannot take each and every aspect into account before the mission, especially because the mission will evolve over time. Actually, the kind of situations which we are talking about are very much like the 'Black Swan' as described by Taleb (2010), i.e., an event which lies outside the realm of regular expectations, has a high impact, and is retrospectively predictable. The predictability of the situation can be disputed, but at least the first two properties imply that it is not feasible to create static plans to cope with the situation. In our running example, a static plan could perhaps not have predicted that the rainfall would be so heavy that it would flood the city.

Missions should not fail if one UAV breaks or gets inhibited, nor if network communication does not complete as assumed. Therefore, we need a hierarchical model for different kinds of situations. This model depends on the additional information available during missions. At the low levels, we do not consider collaboration at all, the idea is that each UAV survives the mission as a 'lonely wolf'. At higher levels there will be an increase of the amount of information and collaboration available and more computational intelligence can be used in the mission control. At the highest level, the squadron works as a one big creature which limb are connected using information bonds. During the mission, we have to dynamically move up and down in this hierarchy and, in the same time, singular UAV have to learn how to survive and accomplish its mission. Clearly, one of the main questions is how to assign tasks and resources while controlling this complex system. From our example, one could imagine what happens when a UAV full of water looses the connection with other UAVs and the central control system. It should most likely take the autonomous decision to drop the water of at the location with most fire which it can observe and get back to get more water.

Operating multiple UAVs gives the possibility of fusing their combined capabilities, potentially leading to rich insights regarding the environment. However, combining the data to generate new knowledge and making the UAVs to react accordingly in a coordinated fashion leads to a high computational overhead. It is clear that the processing power which can be carried on board of the drone will not be adequate for making optimal coordinated decisions. Further, events registered by the UAVs or inferred from analysis of the collected data will lead to bursts of high near real-time computational need. The UAVs should collect as many data as needed about the situation which need to be analyzed to infer the actions which should be taken. Most likely in collaboration with human experts. In our example,

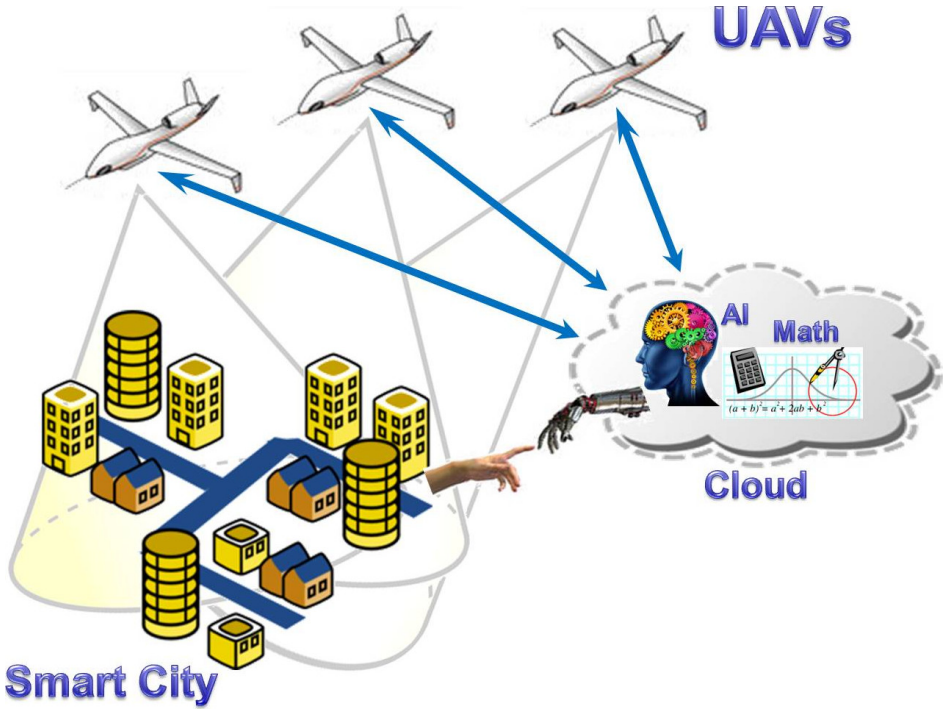

Figure 1: Abstract architecture of the system

UAVs should collect data from the fire situation using, for instance, infrared cameras. These measurements are then collected and a system should filter out the most interesting ones for human consideration. It is important to also keep a record of past observations, as they can, when combined with maps of the area and weather information, be used to make a prognosis on where the fire will move to. Calculating these prognoses is too computationally intensive for UAVs, already assuming that they would have all the other needed data available. It is also clear that these calculations are only needed in exceptional situation.

We propose that Cloud Computing is the most suitable option to solve the issue of bursts in processing power needs by providing adaptive computational power and storage capacity. Usage of Cloud Computing also improves fault and failure tolerance by providing the possibility to globally replicate large sets of data and distribute work load. Cloud Computing will, however, have an unacceptably large latency for certain tasks which will have to be solved on the UAVs themselves. As a consequence, there is a need for an adaptive hybrid solution with Cloud Computing and the real-time systems on board (see abstract architecture in Fig. 1). We will discuss this issue in more detail in Section 4.

3 Why to Optimize ?

The overall goal of the system is to keep itself in an ideal configuration for all missions it should perform simultaneously, given the evolving operational environ-

ment. A fleet of many UAVs and other autonomic devices can have numerous states. Factors which are part of the overall state include the environment, the amount, type, location and configuration of devices, the state of the mobile network used for communication, and the computational load and data stored on the devices. All these factors have an influence on the performance of the system. When the performance of the system is at its maximum, for that particular moment, the system has an ideal configuration. Clearly, the problem of finding an optimal configuration at each point of time leads to a computation so complex that it becomes unfeasible to perform, this can be seen easily since it is already impossible to model for instance a single UAV 100% accurately. The number of influencing factors is of such a magnitude that it is comparable to a situation which requires reasoning using the open world assumption (see Section 5) and we will have to be satisfied with models and approximations and hence quasi-optimal solutions for the problem.

The flying routes are one aspect which can be considered for optimization. During an emergency situation the best outcome is, in most cases, the minimum time needed for reaching the target location. However, we also need to make sure that the droid does not fly into areas which are dangerous like, for instance in our running example, a fire. Determining an optimal solution can be reduced to finding a minimum of a given cost function defined for the mission. However, in practice, the cost function will be too complex (e.g., not derivable), too expensive and too dynamic to solve analytically and we will have to consider the problem in a more pragmatic way. One promising approach is the use of some form of evolutionary computing since it allows for the function to be considered a black-box, i.e., an analytical description of the function is not required. Instead, it suffices that we are able to compute values from the function. Further, the parameters for some of these algorithms can be tuned such that solving a dynamically changing problem becomes possible. Lastly, most of the algorithms can be highly parallelized as was, for instance, done by Fok et al. (2007).

Another aspect which can be optimized, is the communication environment. The missions should, if possible, be planned such that the network remains reachable. Promising work has been performed in the field of Wireless Mesh Networks. This approach enlarges the coverage of the network and should help the devices to save power, which is obvious bottleneck of the system. A smart choice in routing paths and link scheduling algorithms can further lower interference, balance the load, and enhance the throughput and fairness as was studies by Pathak and Dutta (2011). It should also be observed that the situation is quite different from traditional wireless mesh networks because the nodes work cooperatively. We could, for instance, instruct one of the robots to move to a certain location where a higher transfer rate can be obtained, i.e., we can physically move the medium containing the data and even connect the device to a wired link in case of need. In the case of a fire we could for instance have one, perhaps physically specialized, droid which places itself between the fire zone and the central control station or closest radio mast. This droid could then relay the communication traffic from the others and hence reduce the transmission power needed, which also improves the transfer rates. Kopeikin

et al. (2013) used agents to simulate a more complicated setting where the choice and location of UAVs to be used as relays is depending on their utilization rate and network topology prediction. We select their approach for our setting since keeping the communication link working is, despite the fact that we plan for a system which can work without central control, a much desired property.

In the setting we are describing we will have to find optimal routes, communication, and other factors, which are contradicting objectives. Hence, we have to consider the case as a multi-objective optimization problem and will need to use appropriate methods for solving it. It also turns out that optimal becomes somewhat ill-defined since there will be multiple solutions which can be considered optimal, i.e., there will be a Pareto front of optimal solutions. Using Cloud Computing we might, as we elaborate further in the next Section, be able to consider all factors for the optimization of the multiple tasks which the system has simultaneously. On the practical level, a hybrid optimization environment has to be created. This environment should consider many algorithms for solving problems and simulate their outcomes before deploying the best ones to the UAV. Also the method used for finding out the best algorithm is likely to evolve over time, we develop this idea further in Section 6.

4 Computing Resources

In this Section we will look at how Cloud Computing can help us in emergency situations with the help of UAVs and other devices with a degree of mobility. We will discuss why a dynamic combination of Cloud Computing and the computational power of the components in the network should be used in an emergency situation, how the Cloud can help to handle bursts in the need for computation, how interaction with human gets facilitated, and discuss a model for data storage which we elaborate further on in the next sections.

Mell and Grance (2009) defined Cloud Computing as *'a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction.'*. Buyya et al. (2009) studied Cloud Computing as an emerging IT platform and compared it to other technologies like grid and cluster computing. They identified the differences in resource requirements of scientific, data analysis and Internet based applications. In emergency cases, the Cloud will be used from all three perspectives. Scientific computations and data analysis are needed for the analysis and the verification of models and strategies. Much of the interaction with the Cloud will happen trough Internet based applications, since it is the ubiquitous way of interacting with Cloud Services. The most important reasons why we select Cloud Computing as a solution for our scenario are rapid scalability (both up and down), high security and privacy, and strong support for fail-over and content replication. These features are not all at the same time available in either grid or cluster computing. We expect the services which are located in Cloud Computing infrastructure to assist the overall system by coordinating and

(re)configuring the other components.

Research on changing transport situations has some similarities with coping with disasters: in both scenarios there is an amount of slowly evolving infrastructure, e.g., the roads and the cities' shape, faster and repetitive events, e.g., the behavior of road users and city inhabitants, and unexpected situations with a severe impact, e.g., traffic accidents and wide scale disasters. Earlier research by Li et al. (2011) proposes the use of Cloud Computing for solving simulation and decision making problems for urban transportation systems. In their research the authors indicate that for performing traffic simulations both great amounts of computing power and storage capacity are needed. These simulations happen both on-line and many of them can be ran in parallel. They conclude that Cloud Computing is a reasonable approach since a supercomputer would imply a risk for insufficient capacity in the future. A supercomputer would also suffer from a lot of wasted capacity while in the Cloud Computing setting multiple systems can share the Cloud's 'infinite' capability. In a similar realm we want to tap into this capacity to solve hard problems in a short time.

A system can use several strategies to support the missions. The basic options can be found in Trigui et al. (2012) which distinguishes a centralized strategy, a distributed strategy, and a market-based approach which is a trade-off between these two.

The centralized strategy, where one central component has control over the whole system, sounds like a reasonable solution when combining the multiple components with the help of a central Cloud Computing based solution. This strategy was also most time efficient in the simulations by Trigui et al. (2012) where a combination of mobile robots and a Wireless Sensor Network were used to optimize the capturing of intruders. However, it is noted in the paper that *'[the simulation] did not take into consideration the communication delays and we provide insights only on the pure behavior of the coordination strategies independently from the communication.'* In the scenario which we consider these delays in communication can have an unacceptably large latency for certain tasks. For instance, a robot which detects an obstacle using an on-board camera cannot send the data to a central place and wait for a rerouting. The navigation decisions should be performed autonomously as, for instance, demonstrated by Kang and Prasad (2013) for an uncertain environment Other tasks, such as moving object recognition (Nimmagadda et al. (2010)), can be solved partially on the UAV or completely on a centralized server, dependent on factors like wireless bandwidth, image complexity, and the size of the reference image database. In their particular experiment, it was always more favorable to offload the task partially or completely to the server, when the image database contained 5000 images or more. The authors also use a speed-up factor to indicate the computation speed difference between the centralized system and the UAVs, and note that this factor can be treated infinity in the case of Cloud Computing.

Another argument against the use of a centralized system is that it becomes a single-point-of-failure. This last drawback is mitigated by the use of Cloud Computing since, as discussed by Buyya et al. (2009) , it has a strong support for fail-over

and content replication as compared to grid or cluster computing which usually only support restarting failed computations.

In the case of an emergency it is very well possible that the central control station becomes unreachable. When in this scenario the centralized strategy is implemented, the robots would become unusable since they are totally dependent on orders from the central control. The distributed strategy seems to offer a solution; each robot or agent is making a decision based on information which is available locally. However, as shown in the experiments by Trigui et al. (2012), the distributed strategy suffers from an increased mission time. This is as expected since the individual components will rarely be able to compute a global optimal solution by only using the locally available information. A clear benefit is, however, that if the volume of data needed to make the centralized decision becomes huge or the communication is unreliable or slow, the mission time for a distributed system would actually become lower.

The final strategy considered is the marked-based approach which does not require all information to be sent to the central control station. Instead, as argued by Dias et al. (2006), to perform a given task the robots should be chosen based on their own estimation of resource usage. In other words, the robots make their own estimation of the cost and based on this figure a central decision making entity decides which robot will perform the job. The measurements in Trigui et al. (2012) show that in terms of mission time, this strategy performs somewhere in between the centralized and the distributed strategy. The distance which each robot travels, i.e., the resource use, is not significantly different from the centralized strategy.

As argued above, both a purely centralized or distributed setting have undesired properties when using multiple UAVs for emergency situations. These factors are quite specific to the setting. When using a centralized setting, like a purely Cloud based solution, the system will have a latency which is unacceptable for flying robots. A purely distributed setting would then again have the drawback that a global optimum is most likely unattainable. The consequence of these two extremes, complete autonomy of the UAVs and complete offloading to the Cloud, shows the need for an adaptive hybrid solution with Cloud Computing and the real-time systems on board (see abstract architecture in Fig. 1). In other words, we will need something close to the market-based approach which combines the centralized and the distributed strategy. This dynamic strategy would continuously evaluate which of the strategies is most beneficial at a given time-point and would also have to take the survivability of the system into account. The robots will then in case when the central control station cannot be reached for a certain amount of time continue to work autonomously, perhaps with a lower performance. To attain this, the functions of the Cloud which are essential for the working of the overall system have to be be replicated in the UAV network. At any point, the Cloud should simulate and try to predict what would happen in case of different kinds of failures and pro-actively adapt the algorithms on the UAVs.

It is expected that certain tasks which we would like the Cloud to solve are not computable within reasonable time, or not at all, due to an inherent complexity of

the problem at hand. Hence, the Cloud needs to hand certain tasks over to humans in order to get them solved. Further, interactions with humans are also needed to assist rescue teams in case of disasters. We expect that a certain subset of the tasks will be of such nature that they can be solved using a technique similar to Amazon's mechanical Turk [1]. This crowd-sourcing platform enables its customers to create *Human Intelligence Tasks* and decide upon the price which human workers will get paid upon completion of the task. This kind of tasks can be, for instance, the annotation of imagery as was studies by Sorokin and Forsyth (2008). The authors concluded that *'the annotation happens quickly, cheaply and with minimum participation of the researchers, we can allow for multiple runs of annotation to iteratively refine the precise definition of annotation protocols.* One important concern with the use of crowd-sourcing is that in case of a disaster related to a conflict between humans, the opposite party might try to counterfeit the answers to the system. In the case of our illustrative example, we could use a Human Intelligence Task to check preselected frames, recorded by the UAVs, to decide whether there are humans visible.

The Cloud and modern web infrastructure is also known for the introduction of NoSQL databases. This term refers vaguely to several types of databases which do not support SQL with traditional ACID guarantees. This abbreviation stands for Atomicity, Consistency, Isolation, and Durability, which means that any transaction in the database happen completely or not at all, will not violate database rules, happens without interference from other transactions, and is stored in the database even in case of failure immediately after the transaction, respectively. (See also Haerder and Reuter (1983))

Many of these NoSQL databases have been designed to handle specific types of data or huge amounts of data. We will discuss more about a certain type of graph data, which we choose to use for storing data, and the handling of huge amounts of this kind of data in the following sections.

5 Use of Semantic Agents

From an abstract perspective, a Smart City can be seen as a collection of entities with some kind of identity and relations between them. When we consider this idea, we can see that we could represent the Smart City as a graph where vertices model entities and edges are used to describe their relations. This observation suggests that the Semantic Web, as originally suggested by Berners-Lee et al. (2001), would provide suitable concepts for describing and reasoning about Smart Cities. The Semantic Web encompasses the idea that entities and the relations between them in the real world can be described in a similar way as web documents and links between them. The entities of interest, in a Smart City, have some kind of functionality, and many have sensors, actuators or some sort of, perhaps artificial, intelligence. This is to say that the abilities of the entities vary greatly. Examples of entities do not only include wireless sensors for temperature and pollution, traffic and street light control sys-

[1] `https://www.mturk.com/`

tems, and pavement heating, but also digital services and the people living in the city. Also the UAVs are just examples of entities. The integration for these types of entities was studies before in Terziyan (2008) and each of them was regarded as a Smart Resource. In the same article the author also suggest that the Semantic Web does not reach its aims if it can only be used to describe passive, non-functional entities. Instead, it should be possible to describe the behavior of entities, which includes communication, coordination, and proactivity, i.e., the ability to predict the upcoming situation and act accordingly. The suggested approach was also further enhanced in Katasonov et al. (2008), which adds semantic agents to the picture. These agents are a representation of the Smart Resource in an agent environment. By encapsulating each Smart Resource with an agent, the entities gain the ability to communicate with each other. When this communication happens in a Semantic language which allows for dynamics, like, for instance, S-APL formalized in Cochez (2012), we can reach the goal of describing the behavior of entities.

When looking at disasters in Smart Cities, it appear reasonable, or perhaps even natural, to use the Semantic Web concepts to describe and exchange information. The reason is that disasters tend to happen unexpectedly, and when they happen they have a great local impact. Also the time-scale of disasters varies a lot, the flow of information is raising over time, while available resources, like batteries, decrease. This unexpected nature implies that it is hard, if not impossible to give a framework or fixed set of terms to describe a situation. Further, we are not able to identify beforehand all entities which would play a role during the development of events. Therefore, we need a way of describing any entity and a way of reasoning about them. This flexibility is offered by the Open World assumption, which is one of the cornerstones of the Semantic Web. This assumption encompasses the idea that if we do not know something about an entity we cannot assume that it is not true, i.e., we always assume that our graph of data is incomplete. If we, from our illustrative scenario, have not received any information about the flood yet, we can still not make the assumption that everything is fine. It could namely be that certain UAVs are saving their energy by not communicating immediately. This view and way of reasoning is opposed to a Closed World assumption, where the fact that something is not known would immediately imply that it is not true, i.e., the world exists only of the part which we are able to observe. The closed world way of reasoning is typically used in relational databases.

The main drawback of using Semantic technologies could be the reduced processing speed. Effort related to making semantic data storages better able to cope with big amount of data can, for instance, be found in Abadi et al. (2007). We, however, expect that not the storage, but the processing might become the bottleneck in a concrete implementation, partially caused by a need for alignment and negotiations. This extra processing power could be provided by the Cloud infrastructure at the cost of latency as described in Section 4.

The use of an Agent System is a natural choice since it reflects the actual environment to a great extent. The UAVs can be represented by agents because they are able to act, to perceive their environment, and to communicate with each other. Fur-

ther, we are looking into UAVs which can function autonomously and which have the ability to achieve their goals. These requirements, needed for the UAV to be classified as an agent, and further research on distribution of intelligence and coordination among agents, have been collected in Ferber (1999). It should be noted that the same work also handles the concept of organizations and sub-organizations. An organization groups agents together based on their functions and roles. One of these organizations can itself be considered as an agent in a bigger organization. Similarly, when we are combining Cloud Computing into our agent system, the Cloud itself is regarded as an organization and hence also as an agent inside the whole.

Joining the capabilities of Agents and Cloud Computing has been proposed earlier in the context of Smart Cities, like, for instance, by Ito et al. (2012) who proposed their combined use for solving questions about smart housing, senior people care and road congestion management. Further research, related to the use of agents and Cloud Computing in traffic situations, can be found from Li et al. (2011). Their main argument in favor of using a multi-agent setting is that mobile agents only require a runtime environment and can run their computations near the data which increases performance, reduces latency and communication costs. Moreover, a multi-agent system is said to perform better as a static agent system when faced with requirements of dynamic traffic scenes. The agents are able to travel trough the component involved and autonomously implement strategies. It is also stated that thanks to the use of Mobile Agents new strategies can be distributed while the system is running and hence it is possible to keep up with scientific development. Based on these ideas we add the requirement for the agents to be mobile to our system. The UAVs will be mere shells in which the mobile agents reside and which offers them services.

Further research on similar ideas, but focused on transportation, by Wang and Shen (2011) points out that contemporary Cloud offerings also include infrastructure beyond general purpose computers. Especially when combined with agents which share similar behaviors it is possible to harness the power of parallel execution. The authors proposed and experimented the use of Graphics Processing Units (GPUs) as a host for traffic signal timing optimization problems. This type of hardware has also been used earlier for running evolutionary computations as, for instance, by Fok et al. (2007).

The fact that we allow the UAVs to communicate with each other trough agents and to the Cloud creates many opportunities. First, the computational power of the Cloud can be used to calculate better cooperative mission paths compared to what would be possible with the limited computational power of available on the drones. Even the combined resources on the UAVs would be dwarfed by the virtually unlimited power available in the Cloud. However, as discussed before, the result of doing the computation in the Cloud can be better, but will always involve a certain latency. And when the connection to the Cloud is lost, the distributed intelligence of the UAVs, which can be managed by the agents, should be used instead.

Second, the robots could use each others' observations to decide upon their own actions and hence disable some of their own sensors in order to save energy. This also applies to other actions like physically altering the environment and even fly-

ing. A UAV which is low on energy could forward a task to another UAV, which has plenty of resources left or there could be a carrier UAV which carries other smaller UAVs on-board to save the energy needed while flying.

Finally, loosing an individual UAV is not such a problem since the other UAVs can try to take over the task by migrating the mobile agents. This is facilitated by the Smart Resource framework because the mobile agents are build as semantic agents which are able to communicate with each other about their behavior and capabilities.

6 Toward Evolving Knowledge Ecosystems for Big Data Understanding

We expect the total amount of data which the all sensors in the Smart City collect jointly to be enormous and varying over time. It is possible to filter the data, but discarding parts which appear to be needed later could lower the potential performance of the system dramatically. The data also needs to be processed in a timely fashion, because if the system realizes earlier that a reconfiguration is needed it can save essential resources. Hence, we will need to store and process this voluminous data at a high velocity. As we argued in the previous Section, we would benefit from the use of a semantic representation of the data since it allows for an unlimited extension and the use of the open world assumption. In conclusion, we could say that we have a Big Semantic Data problem since we have the three aspects which this type of problems consists of according to Zikopoulos et al. (2012), namely the need to work with a big volume of varying data with a high velocity.

In previous work (Ermolayev et al. (2013)) we looked at how to solve a Big Semantic Data problem which consisted of handling streams of tokens arriving at high rate and altogether representing a huge amount of data. That work stayed on a very abstract level. The scenarios described in this chapter are concrete example of situations where such a system could be used. In the following subsections we will, based on the previous work, describe the balance between volume and velocity, how big data can lead to big knowledge, and how a system inspired by the mechanisms of natural evolution could provide a solution for managing this knowledge. This type of system should be implemented and deployed on Cloud infrastructure to provide scalability. We will also offer suggestions on how the system could work, based on our illustrative example.

6.1 Volume versus Velocity

Volume, the size of the data, and velocity, the amount of time allowed for its processing are clearly the main factors when talking about our Big Semantic Data problem. Both of them manifest themselves when a high number of measurements, like for instance camera recordings, temperature, and humidity, have to be handled within a reasonable time. When the system extracts information or knowledge from the

data it does this *effectively* if no important facts are overlooked, i.e, the analysis is complete. We say that the *utility* of the extraction is high, when the results are useful. The *effort* is a measurement of how much time and resources the system uses to perform this extraction. The ratio of the utility to its associated effort is interpreted as the *efficiency* of the system. To illustrate these ideas, imagine that the system receives an image of a fire for processing. If the system is able to extract all possible information from it, including perhaps data not related to the fire, it is effective. In case these results are useful, we achieved a high utility. When the system is able to do this fast or with a low amount of resources it has a low effort measure. If all of these are true, we achieve a high efficiency, i.e., a lot of useful information got extracted with little effort.

Note that when the effectiveness of the system increases, the computational complexity will increase and hence there is a risk that the efficiency of the system drops. This does not necessarily happen in case the information found appears to be useful. Hence, if we would like to make a deeper analysis of the collected data, we would have a less efficient system. To cover up for this lower efficiency, we would need a higher computational power, which Cloud Computing can provide. The reason is that the Cloud infrastructure can reduce the effort (at least the time component, which is most relevant) to a minimum by using more resources. Obviously, even when using Cloud Computing the system will be bound by theoretical and practical limits. Moreover, as mentioned above, the computations which are performed in the Cloud come with an inherent communication latency.

6.2 Big Knowledge

Since we are trying to extract all relevant information from all sensors which we have available in the Smart City, we start from a vast amount of data. Also, when we try to extract all knowledge from the data, we might end up with an unmanageable amount of knowledge. From that observation we identified some aspects which should be taken into account while working with Big Data. We called this approach *3F+3Co* which stands for Focusing, Filtering, and Forgetting + Contextualizing, Compressing and Connecting. It should be noted here that not all terms are novel in the sense that they have been used in different domains and interpretations, see, for example, Dean and Caroline (2011) for a slightly different view on 3F. We gave an occasionally overlapping meaning to each of these terms in the context of Big Data analysis as follows:

Focusing is mainly concerned with the order in which the data is processed. An optimal focus will only scan the data which is absolutely needed to come to an answer for the question which is at hand and will hence lead to a higher efficiency. This facet will play a very significant role in the emergency system since the data is arriving continuously and hence the focus will need to go to the data which is related to the event which is currently going on. For instance, in our illustrative example, the focus should move from the fire fighting to the

flood damage prevention as soon as there is an indication that the flood might cause more damage as the fire.

Filtering is ignoring anything which is, hopefully, not of importance for immediate analysis. We use *hopefully* since deciding whether information is relevant or not can in most cases not be done with a hundred percent certainty. One way to filter is to only focus on specific features of the data, which also reduces its variety and complexity. The Smart City emergency system could for instance ignore information from sensors in the center of the city when there is a fire on its outskirts.

Forgetting is a further step from filtering where data, or knowledge derived from it, is completely removed from the system. This trashing can remove potentially valuable information. In the work which we did around Evolutionary Knowledge Systems (see Section 6.3), we use the technique of "forgetting before storing ". This means that there has to be reason before anything is stored at all in the knowledge base. Using this technique is sensible in case the amount of information which would need to be stored would be of such size that even nowadays' Cloud Computing solutions and data centers would be overwhelmed, if this is not the case it might be reasonable to keep all the data stored somewhere, at least for a certain amount of time.

Contextualizing comprises not only the change of sense of statements in different contexts, but also judgments, assessments, attitudes, and sentiments. There are various facets which contribute to the context of data. Examples include the origin of the data, the tools used, and the place in which the result will be used. Conceptualization happens for instance when one measurement is given more importance as another one because it was made with a better measurement device.

Compressing stands for both lossy and lossless compression. Where lossy compression is similar to Forgetting which was discussed above. The lossless compression might be very effective because the high amount of data leads to a high probability that repetitive or periodical patterns are present.

Connecting can be done if information is added to an already existing body of data. The whole body is build incrementally. The benefit of linking the data before processing it further is that data and knowledge mining, knowledge discovery, pattern recognition, etc can be performed more effectively and efficiently. A good example of this connecting used for building an index of the world wide web can be found in Peng and Dabek (2010).

6.3 Evolving Knowledge Ecosystems

The Cloud component of the emergency management system receives a lot of information. The content is likely to evolve over time because emergency situations are

subject to rapid and unpredictable changes. To anticipate these changes the system should be able to change its inner working. In the chapter Ermolayev et al. (2013) we proposed an Evolving Knowledge Ecosystem which is able to adapt to external change.

The core idea behind the Ecosystem is that

> The mechanisms of knowledge evolution are very similar to the mechanisms of biological evolution. Hence, the methods and mechanisms for the evolution of knowledge could be spotted from the ones enabling the evolution of living beings.

Starting from this idea, we derived that we could model the knowledge evolution inside the system using ideas from natural evolution. One of the core ideas behind the Evolving Knowledge Ecosystem is that, similar to the idea of natural selection proposed by Darwin Darwin (1859), knowledge which is more fit for its environment, has a higher chance to survive as less fit knowledge. The environment here is formed by the incoming information to the system.

The Knowledge Ecosystem assumes that the information which it is delivered is in the form of knowledge tokens. These tokens are self-contained fractions of the data, with a structure which allows integration into what we call knowledge organisms. In order to create these tokens, the streams of data should be transformed. Natural language text can be converted into knowledge tokens using natural language processing (NLP) techniques. The Evolving Knowledge Ecosystem is, however, not limited to that type of data. Any data stream can, in principle, be converted to a stream of knowledge tokens. In the case of our UAV scenario, the extraction of data from measurements might be even much more simple as the NLP example. Much more complicated is the conversion or analysis of images and video. The research concerning these is, due to its complexity, not as mature as for text analysis. (See for instance Atmosukarto et al. (2012) for an approach to action recognition.) There is also effort in integrating capabilities of text and media analysis by Damjanovic et al. (2011).

The benefit of using a Knowledge Ecosystem for managing the data is that it evolves over time and can hence adapt itself to new trends in the data. Also a lot of meta-data is kept inside the system and using these it is possible to propose beneficial algorithms for future actions. It could, for instance, remember that a certain area of forest burned down a year ago and realize that the fire does not go there this time. From these two fact, it could derive a rule about likelihood of spreading of a fire in a given location.

The Knowledge Ecosystem is also able to provide signals to the outside world. As a concrete example, it could instruct the UAV fleet to collect more information about the temperature in a given area. The Knowledge Ecosystem can further be queried for information and trends.

7 Conclusions and Future Work

In this chapter we proposed approaches related to several aspects of the use of a fleet of collaborative UAVs and Cloud Computing in Smart Cities. We started by describing what a Smart City is and what we would need from this system in emergency situations. These situations have specific characteristics like an unpredictable development and a need for immediate action. Then, we gave several examples on how the collaboration between UAVs might be used to improve the overall performance of the system. We looked at this from an optimization perspective and concluded that in order to keep the system in an optimal state, we have to continuously evaluate a multi-objective function. It was also indicated that this function is not available in analytic form and we hence proposed the use of some form of evolutionary computing which would also be very suitable for execution in a Cloud Computing environment.

Cloud Computing was selected as the main computing resource for the system since it has virtually unlimited capacity and also allows for fail-over and content replication, even geographically. Then we stated that it has to remain possible for the fleet of UAVs to work autonomously from the central command center which is hosted in the Cloud.

Next, we proposed the use of semantic agents. Agents are a natural choice to represent the drones, and adding the semantic components allows us to integrate and make analysis from data with any structure. Further, we noted that instead of having a single agent for each drone, we use mobile agents which also hae the possibility to migrate from one drone to the other in case of need.

Then, because the data which is being processed has such a volume, velocity, and structure, it is not realistic to use standard tools. Hence, we end up with a Big Semantic Data problem. In order to solve this problem, we proposed the use of an Evolving Knowledge Ecosystem which allows the integration of information from different sources in a manner similar to the workings of natural organisms.

The problem areas which were touched in this chapter were described from a high level perspective. There is still a very broad scope of topics which can be researched from both a theoretical and experimental perspective. Examples include, but are not limited to, what kind of robots are most useful, which factors influence the optimality of the system and up to which extend, how the communication between the robots and the Cloud should be designed, how the decision whether to perform a computation in the Cloud or on the devices locally should be taken, what type of agent system should be used, whether the evolving knowledge ecosystem has a performance high enough to accommodate the needs, and so on.

Acknowledgements : The authors would like to thank the department of Mathematical Information Technology of the University of Jyväskylä for financially supporting this research. Further we would like to thank the members of the Industrial Ontologies Group (IOG) and the Scientific Computing and Optimization in Multidisciplinary Applications (SCOMA) group of the university of Jyväskylä for their support in the research.

References

Abadi, D. J., Marcus, A., Madden, S. R., and Hollenbach, K. (2007). Scalable semantic web data management using vertical partitioning. In *Proceedings of the 33rd international conference on Very large data bases*, VLDB '07, pages 411–422. VLDB Endowment.

Atmosukarto, I., Ghanem, B., and Ahuja, N. (2012). Trajectory-based fisher kernel representation for action recognition in videos. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3333–3336.

Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The semantic web. *Scientific american*, 284(5):28–37.

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., and Brandic, I. (2009). Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616.

Cochez, M. (2012). Semantic agent programming language: use and formalization. *Master's thesis, University of Jyvskyl, Finland (2012)*.

Damjanovic, V., Kurz, T., Westenthaler, R., Behrendt, W., Gruber, A., and Schaffert, S. (2011). Semantic enhancement: the key to massive and heterogeneous data pools. In *Proceeding of the 20th international IEEE ERK (electrotechnical and computer science) conference*.

Darwin, C. (1859). On the origins of species by means of natural selection. *London: Murray*.

Dean, D. and Caroline, W. (2011). Recovering from information overload. *McKinsey Quarterly*.

Dias, M., Zlot, R., Kalra, N., and Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270.

Doherty, P., Heintz, F., and Kvarnstrm, J. (2013). High-level mission specification and planning for collaborative unmanned aircraft systems using delegation. *Unmanned Systems*, 01(01):75–119.

Ermolayev, V., Akerkar, R., Terziyan, V., and Cochez, M. (2013). *Big Data Computing*, chapter Towards Evolving Knowledge Ecosystems for Big Data Understanding. Taylor & Francis group - Chapman and Hall/CRC.

Ferber, J. (1999). *Multi-agent systems: an introduction to distributed artificial intelligence*, volume 1. Addison-Wesley Reading.

Fok, K.-L., Wong, T.-T., and Wong, M.-L. (2007). Evolutionary computing on consumer graphics hardware. *Intelligent systems, IEEE*, 22(2):69–78.

Gonzalez, L. F., Lee, D.-S., Walker, R. A., and Periaux, J. (2009). Optimal mission path planning (mpp) for an air sampling unmanned aerial system. In Scheding, S., editor, *2009 Australasian Conference on Robotics & Automation*, pages 1–9, Sydney. Australian Robotics & Automation Association.

Haerder, T. and Reuter, A. (1983). Principles of transaction-oriented database recovery. *ACM Comput. Surv.*, 15(4):287–317.

Ito, T., Chakraborty, S., Kanamori, R., and Otsuka, T. (2012). Innovating multiagent algorithms for smart city: An overview. In *Service-Oriented Computing and Applications (SOCA), 2012 5th IEEE International Conference on*, pages 1–8.

Kang, K. and Prasad, J. V. R. (2013). Development and flight test evaluations of an autonomous obstacle avoidance system for a rotary-wing uav. *Unmanned Systems*, 01(01):3–19.

Katasonov, A., Kaykova, O., Khriyenko, O., Nikitin, S., and Terziyan, V. (2008). Smart semantic middleware for the internet of things. In *Proceedings of the 5-th International Conference on Informatics in Control, Automation and Robotics*, pages 11–15.

Kopeikin, A. N., Ponda, S. S., Johnson, L. B., and How, J. P. (2013). Dynamic mission planning for communication control in multiple unmanned aircraft teams. *Unmanned Systems*, 01(01):41–58.

Li, Z., Chen, C., and Wang, K. (2011). Cloud computing for agent-based urban transportation systems. *IEEE Intelligent Systems*, 26(1):73–79.

Mell, P. and Grance, T. (2009). The nist definition of cloud computing, version 15. national institute of standards and technology (nist). *Information Technology Laboratory. www. csrc. nist. gov*.

Nimmagadda, Y., Kumar, K., Lu, Y.-H., and Lee, C. S. G. (2010). Real-time moving object recognition and tracking using computation offloading. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2449–2455.

Pathak, P. and Dutta, R. (2011). A survey of network design problems and joint design approaches in wireless mesh networks. *Communications Surveys Tutorials, IEEE*, 13(3):396 –428.

Peng, D. and Dabek, F. (2010). Large-scale incremental processing using distributed transactions and notifications. In *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation*.

Sorokin, A. and Forsyth, D. (2008). Utility data annotation with amazon mechanical turk. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, pages 1–8.

Taleb, N. N. (2010). *The Black Swan:: The Impact of the Highly Improbable Fragility*. Random House Digital, Inc.

Terziyan, V. (2008). Smartresource–proactive self-maintained resources in semantic web: Lessons learned. *International Journal of Smart Home*, 2(2):33–57.

Trigui, S., Koubaa, A., Ben Jamaa, M., Chaari, I., and Al-Shalfan, K. (2012). Coordination in a multi-robot surveillance application using wireless sensor networks. In *Electrotechnical Conference (MELECON), 2012 16th IEEE Mediterranean*, pages 989 –992.

Wang, K. and Shen, Z. (2011). Artificial societies and gpu-based cloud computing for intelligent transportation management. *IEEE Intelligent Systems*, 26(4):22–28.

Zikopoulos, P. C., Eaton, C., deRoos, D., Deutsch, T., and Lapis, G. (2012). *Understanding Big Data*. Mc Graw Hill.