1 Introduction

Over the course of the last decade, wireless technology has taken the world by storm. It might at this point seem like that was an inevitability, but it was not always so clear. For example, Bob Metcalfe—who is rightly famous for his contributions to networking, particularly his efforts in the design of Ethernet—put it like this in 1993:

> *"Wireless mobile computers will eventually be as common as today's pipeless mobile bathrooms. Portapotties are found on planes and boats, at construction sites, rock concerts, and other places where it is very inconvenient to run pipes. But bathrooms are still predominantly plumbed. For more or less the same reasons, computers will stay wired. (...) So let's just wire up our homes and stay there."*
>
> — Bob Metcalfe [Met93].

And here we are, twenty years later. That may not be a particularly *short* time, but wireless computing has well and truly arrived. Many people carry a smartphone connected to mobile internet; many local area networks are now wireless. For many purposes it is no longer necessary to wire up our homes. Being required to stay there seems old-fashioned.

Wireless technology is of course older than its widespread consumer adoption — indeed, on a computer-science scale of time, it is *much* older. The ALOHAnet system [Abr70] from the early 1970s, for example, was an influential achievement, providing a radio-based network between the Hawaiian islands. In this case, the specifics of a situation demanded a non-standard solution; today, wireless communication might be considered the standard assumption. Such a wireless world poses new questions for computer science and brings new perspectives for existing concepts. In this introduction we shall touch upon several such aspects and how the content of this thesis addresses them.

Some of the work in this thesis is motivated by a specific application of wireless communication: sensor networks. As a general term, *sensor networks* encompass many things; specific examples will follow, but in generic terms the properties that come to mind are as follows. A sensor network consists of a large collection of small devices, each equipped with sensors and a wireless transceiver, carrying out some data-gathering or surveillance task. These devices are typically called sensor nodes.

What sets these sensor networks apart from wireless networks in general is mainly their hardware and the manner of their deployment. Firstly, the hardware in a sensor network is typically considered to be of low cost and low reliability. Secondly, the deployment of a sensor network might not be carefully designed or executed, possibly because the environment is hard to reach or hard to service. These two aspects put a focus on fault tolerance since we may be working with devices that were designed to be disposable: better to handle failures in the network than to make individual devices more reliable. Depending on the deployment and the hardware, it may be impossible or impractical to have replaceable or rechargeable batteries. This puts an additional focus on energy efficiency: a device might be permanently lost when it runs out of energy.

Much of networking theory is based on graphs: they are the *de facto* standard model for wired communication networks. Graphs have, in a sense, proven to be the 'right' model. The properties of wireless transmissions, on the other hand, have turned out harder to capture in graphical models. In wireless networking, there are certainly interesting and useful graph-based results to be had. (We provide some in Part II of this thesis.) Yet it is still very much up for debate what the 'right' model for wireless networking is, if there is such a thing. Common among many of the new models being proposed is a strong focus on the physical properties of radio transmission. In Part I of this thesis we focus on two such physical models: a new model for localisation and new results in the well-known *Signal to Interference and Noise Ratio* model.

Localisation. One of the areas that has seen a shift to more physically-motivated models is *localisation*, a problem that is particularly interesting in the context of sensor networks: where am I? The physical location of a sensor node is often important as this is where the measurements were taken. It is greatly advantageous if the nodes can figure out, by themselves, where they are: then there is no tedious initial configuration and the network is robust against misplaced or moving nodes.

A typical approach for the localisation problem comes from the field of graph realisation. The basis for this approach is node-to-node distance measurement. Such measurements can be based, for example, on the received signal strength of a known-power transmission. Another option is to measure the time difference between transmission and receipt of messages. This results in (approximate) distances between certain nodes being known. These data can be combined in a graph with specified edge lengths. The abstract problem of drawing a set of points such that a given set of point pairs has given distances is known as *graph realisation*. This problem is $\mathcal{NP}$-hard in many settings and variants [Sax79]. Still, the problem is well studied, for example in work on sensor network localisation [DKQW10].

An alternative approach would be to use an off-the-shelf *global positioning system* (GPS). This certainly works well on smartphones. But, as we will argue concretely in the main text, GPS is often not suitable for deployment in sensor network applications. Furthermore, edge-length measurements are, in practice, often not available at the precision required for effective graph realisation. For these reasons, there is increasing interest in *range-free* localisation. This is localisation without explicit measurement of point-to-point distances. After localisation has succeeded, point-to-point distances can of course be inferred, but a range-free system does not take direct distance measurements as input.

In Chapter 3 we propose a novel range-free localisation approach. Our localisation is performed in relation to some base stations. These are assumed to have access to a permanent power source and a powerful radio transmitter. The system is then designed to be as simple and cheap as possible for sensor nodes: they are after all the most constrained part of the system.

The base stations provide an ongoing stream of transmissions that the sensor nodes can tune into when they need to localise. We analyse how well our system performs with a probabilistic transmission schedule. Additionally we design deterministic schedules with worst-case performance bounds.

Scheduling wireless transmissions. A normal network cable connects two endpoints; an interesting aspect of wireless communication is that it features a medium shared by many devices. One of the consequences is that a single transmission may be detected and successfully interpreted by multiple receivers: broadcasting comes for free with the medium. On the other hand, simultaneous transmissions might interfere, to the effect that neither message can be understood. This greatly complicates the problem of coordinating communication. In this way, wireless technology forces us to revisit basic modelling assumptions.

This many-to-many nature of the medium notwithstanding, graphs are a powerful tool for the analysis of wireless networks. (A tool that we gladly use in Part II of this thesis, where we are not necessarily concerned with broadcasting or interference.) A broadcast from a node in a communication graph can be modeled by letting a messages arrive at all neighbouring nodes instead of just at a single recipient. Interference can then be modeled by saying a *collision* occurs if multiple messages arrive at a node simultaneously: then none of these messages are received. Many results have been proven in this model. Often one considers graphs with structural properties that are reasonable for wireless networks, such as disc graphs, bounded doubling dimension [GKL03] or bounded independence [KNMW05].

In addition, there is an increasing output of research concerning models that are not based on graphs. An important such model is the *signal-to-interference-plus-noise ratio* (*SINR*) model. It has also been called—perhaps a little presumptuously—*the* physical model. It is, in any case, *a* physical model: physically motivated, physically reasonable and explicitly nongraphical.

There are variants and parameters, but a typical version goes like this. Consider transceivers located on the 2D plane. The model then posits that the power of transmitted signal diminishes as $1/d^2$, where d is the distance from the transmitter. In particular, for a sender at point s and a receiver at point r the strength of the arriving signal equals $1/|r-s|^2$. This signal reaches all receivers: all transmissions are broadcast. To determine whether a message is correctly received, we look at the ratio of its signal strength versus the total strength all others signals (which interfere). A message is correctly received if and only if, *at the receiver*, it is at least as strong as the sum of all other signal strengths. We review the exact model in the preliminaries (Section 2.6).

In this *SINR* model, receivers are influenced most by nearby transmitters. On the other hand, the signals from many faraway transmitters can build up to cause significant interference over long distances. This global nature of interference is an interesting aspect of the model; one that is missing from graph-based models.

One of the reasons for the interest in the *SINR* model is the pioneering work of Gupta and Kumar [GK00] concerning the (stochastic) capacity of wireless networks in this model. Later it has become the dominant nongraphical model in algorithmic work following the investigation of transmission scheduling by Halldórsson et al. [GWHW09]. As opposed to earlier channel-capacity results, they look at worst-case networks. This way we are faced with a computational problem in the classical sense: given a network and a set of communication requests, devise a schedule that successfully transmits everything as quickly as possible.

Their initial paper proved $\mathcal{NP}$-hardness of this 'wireless scheduling' problem. Further results include ongoing research to give distributed approximation algorithms [HM11]. We argue that, in addition to that, it remains interesting to look at exact solutions—even though $\mathcal{NP}$-hardness dooms this to infeasibility for large instances. Being able to actually calculate optimal solutions provides valuable insight into their structural properties.

In Chapters 4 and 5 we study a problem that is intimately related to the scheduling problem. We are given the location of wireless transmitters and receivers. Along with this, we get a set of transmission requests: which transmitters have messages for which receivers. We then look to find a maximum *link independent set*, that is, a maximum-size set of transmissions that simultaneously succeed. This is the most requests we can immediately fulfill. The relation to scheduling all requests is that, at any time in a schedule, any simultaneous transmissions must be such a *link independent* set.

The problem of finding a link independent set of maximum cardinality is $\mathcal{NP}$-complete. We design a branching algorithm and analyse its moderately-exponential runtime. We implement the algorithm and demonstrate that it runs well. Using this implementation we then experimentally investigate the properties of random geometric instances. We additionally prove some of these properties. In particular, we prove that very large link-independent sets are unlikely—for a certain value of 'very large'—yet that rather large link-independent sets exist with high probability.

Robust routing. An issue that is important in wireless networking, and particularly in sensor networks, is fault tolerance: communication is not reliable. A particular wireless link might be less reliable than a cable. Also, mobile devices might go out of range, causing links to disappear—this can be a reasonable scenario where a cable becoming unplugged might be less so.

One reason that a wireless device might fail is that it runs out of battery power. This is particularly common in sensor networks, where devices are small and cheap. It can reasonably be assumed that not all nodes run out of battery power at the same time. Then we want the remaining part of the network to continue functioning. Deployment in harsh physical conditions might also lead to a significant hardware failure rate. Things like these make fault tolerance an important issue in wireless networks and sensor networks in particular.

Many of the problems studied in this thesis are computationally hard; likely too hard to solve within a real sensor network. As we argued for the link independent set problem before, it is still interesting to do the exact computations offline for analytical purposes. However, deploying the optimal solutions arrived at in this way seems like a dangerous move when failure tolerance is required. What use is a carefully crafted plan if it is no longer valid by the time it gets executed?

This leads us to look at *robust recoverability*, a concept from operations research that has recently seen development. In this framework one defines a 'simple' *recovery procedure* to deal with faults. Simple is here a relative concept, but we particularly want to run recovery within the network. The central idea of robust recoverability is then to take the behaviour of this recovery procedure into account during planning: we already know how the recovery procedure will behave and we can base our plan on this.

In Chapter 6 we study a very basic problem: finding a path between two given nodes in a graph. We will use this path to route packets in the network. The faults we consider are complete node failures. Perhaps a node has run out of battery power, perhaps it was fatally damaged; in any case, we can no longer use it. This is where the recovery procedure comes in. It consists of assigning, beforehand, a *backup node* for every node in the network. In case of failure, the backup node steps in as a replacement. (Perhaps the backup node was, up until that point, in sleep mode in order to save battery; perhaps the backup node will now serve double duty.) By assigning backups beforehand, this replacement can be handled within the network in an ad hoc fashion.

The computational problem is then to plan the path and its backup nodes. We formalise this problem and resolve the resulting complexity questions. Some variants are polynomial-time solvable and some are $\mathcal{NP}$-complete; we give algorithms for each. We also analyse the variation where we are given the path and have to select only the backup assignment. Again, some variants are hard and some are easy; again, we give algorithms for all of them.

Energy-efficient data gathering. The resources studied by computer science have primarily been time and space (memory). The exponentially-increasing transistor count provided by Moore's Law has given us increasingly fast processors, but also an increase in energy consumption [Max13]. On the scale of individual machines this has led to engineering problems in terms of heat dissipation. On an industrial scale, it has sparked interest in 'green computing' [Kur08]. But even without a concern for environmental footprint, wireless devices powered by a battery make energy consumption a very practical issue.

The energy limitations of battery-powered operation are especially tangible in the field of *sensor networks*. Here we consider small, simple, sometimes even disposable devices equipped with some computing power and a wireless transceiver. Coupled with sensors for physical quantities such a device is called a *sensor node*.

An example application would be sensor nodes scattered among the crops on a farm, measuring the amount of rainfall, sunlight, soil moisture et cetera. This can support control decisions for such things as irrigation and the application of fertiliser [WZW06]. As a different, more specific example, a network including sensors for strain, vibration and temperature has been embedded in the concrete of the Hollandse Brug, a bridge at Muiderberg in the Netherlands. The network is being used to monitor the structural health of the bridge [KBK+10, VKV+11].

The battery of sensor nodes is typically considered nonreplaceable. This is most clear in the example where the nodes are embedded in concrete.[1] Once deployed, a sensor node performs its task until it breaks down. This provides a direct link between between energy consumption and the operational lifetime of a sensor network.

As a final topic in this thesis we consider energy consumption. This has already influenced design decisions in Chapter 3, but in Chapters 7 and 8 we look explicitly at energy budgets. We study a task that is often central in sensor networks: gathering the data from the sensors in a base station in the network. How much data can we gather before exhausting the network? We model this as a graph problem and, for its relation to classical network flow, call it *energy-constrained flow*.

[1] In the case of the bridge, one might consider wired power and networking. The advantage of a wireless approach is in the ease of deployment and the low impact on the overall engineering of the bridge.

We show that the problem of maximising the amount of data gathered is strongly $\mathcal{NP}$-complete and even $\mathcal{APX}$-hard. When restricted to geometric networks, with physically reasonable energy costs, the problem remains hard. For graphs of bounded treewidth, we give pseudopolynomial-time algorithms.

Then we look to find good solutions despite this hardness. We develop heuristic algorithms based on linear programming and column generation. Experiments with an implementation of these algorithms demonstrate their effectiveness. Rounding of the linear program also gives efficient approximation algorithms, which works particularly well for st-planar graphs.

In dealing with these topics, we will have touched on many aspects of wireless sensor networks, their structure and the related algorithms. Before the thesis proper begins, let us briefly draw attention to two divisions that it contains. The first is readily apparent from the table of contents: a Part I about physical models and a Part II about graphical models. This is a technical divide and merely a result of organising the material.

The second divide is methodological in nature and a divide that, where possible, we have tried to bridge. On the one hand, we have theoretical, worst-case results about the runtime of algorithms and the structural properties of wireless networks. On the other hand, we have experimental results based on implementations of our algorithms and on simulations. These two approaches are sometimes seen as being at odds with each other—indeed, as the saying goes: *"In theory, theory and practice are the same. In practice, they are not."*[2] In recognising this difference, we find that in fact theoretical and experimental research complement each other to give a more comprehensive view of the subject.

As an example in this thesis, we can point to Chapters 4 and 5 about the Link independent set problem. Even though Chapter 4 is mainly concerned with the formal correctness and worst-case runtime of an algorithm, the design of the algorithm was very much informed by experiments in the early stages of the research. Then, in Chapter 5 we complement the worst-case result with runtime measurements on an actual implementation of the algorithm. With this implementation we observe some structural properties of wireless networks and, moving to theory again, continue to prove some of the observed behaviour. As another example, the concept of a *regular* schedule (Chapter 3) and the corresponding theorems were inspired by the experimental results in Table 3.2. In these ways, there is a back and forth between theoretical and experimental results.

It is important to not be satisfied too easily with experimental appearances: models and theories are what computer science is built on, for good reason. Yet a theorem might not tell the whole story, and pure theory might not be how we arrive at a result (cf. [Tic98]).

[2] Variously attributed to Albert Einstein, Jan van de Snepscheut, Yogi Berra, and others.