

Theoretic Modelling of Routing in Selfish Ad Hoc Networks

Petteri Nurmi

Helsinki 2nd February 2006

UNIVERSITY OF HELSINKI

Department of Computer Science

Tiedekunta/Osasto — Fakultet/Sektion — Faculty		Laitos — Institution — Department	
Faculty of Science		Department of Computer Science	
Tekijä — Författare — Author			
Petteri Nurmi			
Työn nimi — Arbetets titel — Title			
Theoretic Modelling of Routing in Selfish Ad Hoc Networks			
Oppiaine — Läroämne — Subject			
Computer Science			
Työn laji — Arbetets art — Level		Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
M. Sc. Thesis		2nd February 2006	87 pages + 8 app. pages
Tiivistelmä — Referat — Abstract			
<p>In this Thesis we provide background information on ad hoc networks and game theory, after which we perform a literature review and critical analysis on existing cooperation mechanisms and theoretical models for routing in ad hoc networks. Next, we introduce a novel theoretical model for routing that attempts to correct the weaknesses in the earlier models; namely, taking into consideration the dynamic structure of routing decisions and a node's uncertainty about the resources of the other nodes. We subject the novel model under close scrutiny and analyse both the theoretical optimality and practical feasibility of the model. Furthermore, we use simulation studies to show that the resulting model is not too complex for practical purposes and to give insights about node performance under the proposed model.</p> <p>ACM Computing Classification System (CCS): C.2.0 Computer-Communication Networks - General G.3 Mathematics of Computing - Probability and Statistics I.2 Computing methodologies - Artificial Intelligence I.6 Computing methodologies - Simulation and modeling</p>			
Avainsanat — Nyckelord — Keywords			
ad hoc networks, cooperation mechanisms, game theory, theoretical modeling			
Säilytyspaikka — Förvaringsställe — Where deposited			
Muita tietoja — övriga uppgifter — Additional information			

Acknowledgements

First of all the authors wishes to acknowledge the help of Greger Lindén. His help has significantly improved the readability of the Thesis. Furthermore, the author wishes to thank the following colleagues for pushing me to finally finish the Thesis and for helping me to keep my insanity :-): Patrik Floréen, Michael Przybilski, Jukka Suomela, Miquel Martin, Stefan Gessler, Bernd Mhros, Stephan Steglich, Olaf Droegehorn, Michael Sutterer, SianLun Lau, Olivier Coutand, Johan Koolwaaij, Alfons Salden, Remco Poortinga, Adrian Flanagan and Agathe Battestini. Last, but not least, a big thanks to my parents for their support and encouragement.

Dedicated to my father, In memoriam.

List of Figures

1	An example of an ad hoc network with five devices. The circles and lines represent the connectivity graph the devices induce. . . .	2
2	Two examples of extensive form games.	23
3	The game tree of the prisoners' dilemma.	24
4	A signalling version of the prisoners' dilemma.	25
5	The repeated prisoners' dilemma.	27
6	The modified version of the prisoners' dilemma. The dotted circle is used to represent nature.	32
7	An example of a game of incomplete information in extensive form. For clarity the root of the game tree has been emphasized.	33
8	A game that illustrates the concept of consistency [KW82b].	34
9	The reputation sources in CONFIDANT.	47
10	The reputation sources in CORE.	47
11	An example game tree, which has a single intermediate node that has two possible types, and where the source (player I) sends either one or zero packets to the network.	57
12	Network topology for the path discovery example.	66
13	Overall number of packets sent as a function of iterations in the first experiment setting. In all plots, the three topmost represent the results in the three test runs, whereas the lowest plot is the average of the three test runs.	71
14	Fraction of packets that were actually sent and that successfully arrive at the destination node in the first experiment setting.	72
15	Fraction of packets forwarded in the first experiment setting for other nodes from all packets arriving at a node.	73
16	Overall number of packets sent as a function of iterations in the second experiment setting.	74
17	Fraction of packets that were actually sent and that successfully arrive at the destination node in the second experiment setting.	75

18	Fraction of packets forwarded in the second experiment setting for other nodes from all packets arriving at a node.	76
----	---	----

List of Tables

1	Matrix representation of the prisoners' dilemma [FT91]. The equilibrium strategy $(0,0)$ is denoted in bold.	19
2	Comparison of the models presented in Section 5.1.	52

Contents

1	Introduction	1
2	Ad Hoc Networks	4
2.1	Characteristics of ad hoc networks	5
2.2	Routing in Ad Hoc Networks	6
2.3	Dynamic Source Routing (DSR)	10
2.4	Watchdog	11
3	Game Theory	13
3.1	Basics	14
3.2	Example: Prisoners' Dilemma	19
3.3	Extensive form games	20
3.4	Repeated games	24
3.5	Evolutionary games	28
3.6	Bayesian Games	29
3.7	Mechanism Design	35
3.8	Adaptive learning in Bayesian games	37
4	Cooperation stimulation in Ad Hoc Networks	39
4.1	Selfishness and Models of Selfishness	39
4.2	Virtual Currency Systems	40
4.3	Reputation Mechanisms	43
5	Theoretical Modelling of Cooperation Stimulation	49
5.1	Previous Game Theoretic Models	49
5.2	Example Routing Scenario	53
5.3	A Theoretical Model for Routing	55
6	Experiments	64

6.1	Simulator overview	64
6.2	Experiment setting and results	69
7	Self critique and summary	75
	References	79
1	Proof of a perfect Bayesian equilibrium in the routing game	1
2	Bayesian probability theory	4

1 Introduction

In recent years, the number of wireless communication devices has increased rapidly and nowadays more and more people carry with them a mobile phone or a PDA everywhere they go. This has resulted in novel computing visions that attempt to leverage the availability of communication devices and to provide novel services to the end users. As an example, ubiquitous computing strives for making services available to users everywhere and at all times [Wei91]. However, at the moment realising these visions requires extensive investments in communication infrastructures such as base stations for mobile devices and routers for fixed networks. Unfortunately, such investments are not foreseen in the near future and, as a consequence, researchers have turned their attention to communication technologies that are able to work with limited or even without any infrastructure support. Within this Thesis we focus on one of these technologies, wireless ad hoc networks.

Wireless ad hoc networks are networks that are formed by a collection of devices, e.g. mobile hosts or PDAs, in a self-organizing fashion without any pre-established infrastructure support. The devices are usually called nodes and the network can be formally defined to be a graph, where the devices that form the network are the vertices of the graph and the edges are determined by the transmitters of the devices so that there is an edge between two vertices, if they are within the range of each others' transmitters. Alternatively, we say that the transmitters of the devices *induce a connectivity graph* that defines an ad hoc network and more specifically the *topology* or *structure* of that network. An example of a small ad hoc network formed by five devices is shown in Figure 1.

In communication networks, one of the most crucial tasks is that of routing, i.e. ensuring that packets arrive at the designated destination. But, as ad hoc networks lack infrastructure support, the nodes must implement all networking tasks by themselves and for routing this means that packets must be routed using other nodes as intermediate relays. However, the nodes (devices of the users) are often energy constrained by their battery level and thus it is not always in the best interest of a node to forward packets for other nodes. When nodes refuse to forward packets for other nodes, they are called selfish. Simulation studies have shown that in ad hoc networks the performance of many standard routing pro-

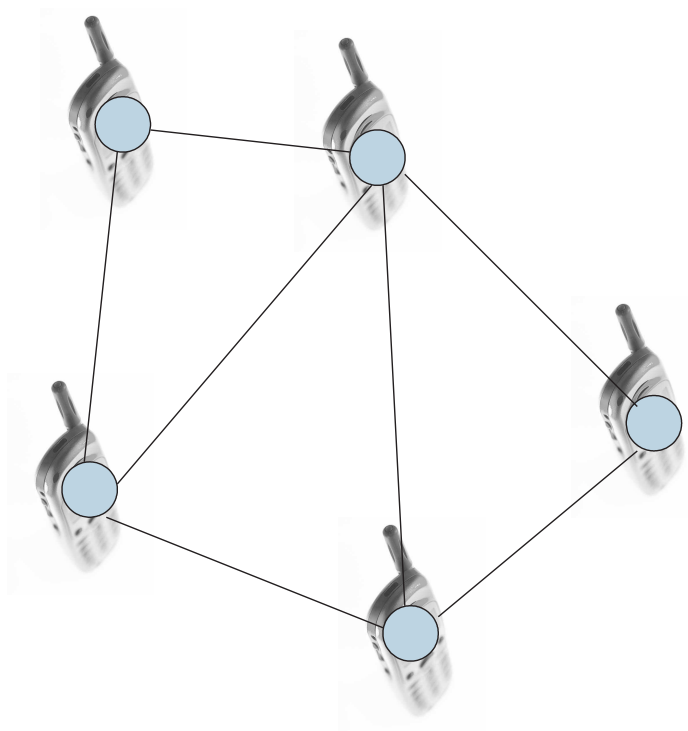


Figure 1: An example of an ad hoc network with five devices. The circles and lines represent the connectivity graph the devices induce.

protocols often degrades significantly when even a small portion of the nodes acts selfishly [MM02c]. For this reason, researchers have developed so-called cooperation (stimulation) mechanisms that attempt to motivate the nodes to forward packets for each other. Unfortunately, at the moment most of the employed cooperation mechanisms do not have a firm theoretical background, which would enable giving some guarantees about the network performance under varying conditions.

A routing decision is essentially a conflict situation that can be modelled using tools from game theory (e.g. [FT91]). The existing theoretical models for routing in ad hoc networks, however, are based on so-called static games, which means that routing decisions are made as if all the nodes on the routing path decide simultaneously whether to forward packets or not. Furthermore, the models usually ignore the uncertainty about the resources of other devices. To overcome these deficiencies, the main contribution of this Thesis is to present a novel theoretical model for routing in ad hoc networks. Furthermore, we perform simulation studies that illustrate what kind of network performance can be expected

under the presented model. Additionally, the Thesis covers background information about ad hoc networks, game theory and cooperation mechanisms, and conducts a literature review on existing theoretical models and compares and analyses them critically. Also the new model presented in this Thesis is analysed critically.

The rest of Thesis is divided into six sections. Sections 2, 3 and 4 cover background information about ad hoc networks, game theory and cooperation mechanisms. In Section 5, we first review critically existing theoretical models, after which we introduce a novel model for routing in ad hoc networks. The presented theoretical model has not been published in any scientific journals or conferences, but it is an extension of earlier work by the author [Nur04]. To validate the model, we have conducted experiments that are described in Section 6 alongside with the simulation results. Finally, in Section 7 we summarise the contents of the Thesis and discuss the flaws of the presented approach and the conducted simulations. The sections have been made as independent as possible and references to other sections are provided whenever we felt information from other sections to be relevant for better understanding the material under discussion.

2 Ad Hoc Networks

Roots of ad hoc networks stretch back to year 1968, when research on the *ALOHA* network was initiated [Abr70, FJL00]. The goal of the *ALOHA* project was to design a network for connecting educational facilities in Hawaii. Each facility could only communicate with the facilities that were within their transmitter's range. Thus no support for routing was provided and, furthermore, stations were assumed to be fixed. However, because the used communications protocol lent itself to distributed channel-access management, the *ALOHA* project can be seen as the mother of ad hoc networks.

Some surveys ignore *ALOHA* and consider instead packet radio networks [SG85] as the starting point of ad hoc networks. Research on packet radio networks was initiated by DARPA¹ in 1973 with the goal of designing a distributed network where the communication facilities cooperate to relay traffic on behalf of one another [FJL00]. Since then packet radio networks have matured considerably and they have been successfully used in various military and rescue applications. Unfortunately, packet radio networks were not able to arouse interest in the consumer segment, which is why IEEE² adopted the term *ad hoc* networks while designing the IEEE 802.11 wireless communications (WLAN) standard. Judging by the current level of interest it seems that the trick was rather fruitful as nowadays ad hoc networks are a wide research area, for which numerous conferences are held each year. Furthermore, applications that use ad hoc networking technologies are starting to emerge to the general market and, especially in novel mobile computing solutions, ad hoc networking is being used more and more as an enabling technology.

In the rest of this section we introduce some aspects of ad hoc networks. In Section 2.1 the focus is on characterizing ad hoc networks, whereas Section 2.2 discusses the most common routing techniques for ad hoc networks. In Sections 2.3 and 2.4 specific technologies, which serve as background information for later sections, are introduced.

¹Defence Advanced Projects Research Agency

²The Institute for Electrical and Electronic Engineering

2.1 Characteristics of ad hoc networks

According to [FJL00] the most widespread notion of mobile ad hoc networks is that they are

networks formed without central administration which consists of mobile nodes that use a wireless interface to send packet data.

From this definition it is possible to derive the typical characteristics of an ad hoc network. The characteristics we derive here are essentially the same as the ones derived by Giordano [Gio02].

First of all, the lack of central administration forces the nodes to implement all networking tasks by themselves. This property of ad hoc networks is called *self-organization* [BBC⁺01, HBGV01] and it is a major contemporary research topic. The most important issues within self-organization are distributed channel management and cooperation stimulation mechanisms. The former refers to the problem of selecting, which radio frequencies devices are supposed to use and, when to send data to the (radio) channel. Namely, if multiple radio signals are communicated using the same frequency spectrum, the signals collide and the transmitted messages distort. Cooperation stimulation, on the other hand, refers to the task of motivating the individual devices to cooperate for the common good of the network. As cooperation stimulation mechanisms are the focus of this Thesis, they are discussed extensively in subsequent sections. However, distributed channel management is not further discussed within the Thesis.

Because nodes are assumed to be mobile, the network topology can change dynamically. This imposes challenges to the used routing and data management schemes. Furthermore, mobility usually implies that the devices are energy constrained and thus optimizing resource usage is important in ad hoc networks. As is discussed in Section 4.1, energy constraints are also the main reason why cooperation stimulation mechanisms are needed.

The final key property of ad hoc networks is the wireless nature of communications, which causes, e.g., restricted physical security, lower transmission capabilities and higher loss, delay and jitter rates. These aspects are mainly ignored in

the Thesis.

2.2 Routing in Ad Hoc Networks

Routing protocols for fixed networks usually gather information about the network topology and select routing paths locally based on this information. In addition, in fixed networks the paths do not need to be optimal as there resource constraints do not play an important role. However, as discussed in the previous section, in ad hoc networks the situation differs significantly and this forces the routing protocols to have small and efficient data structures, to select paths so that resource usage is optimized and to take into account consistency problems arising from the dynamically changing topology. In order to meet the aforementioned challenges, novel routing paradigms, which are designed specifically for ad hoc network environments, have been developed. Nowadays the number of different routing protocols is exhaustive and in the Thesis our discussion concentrates around the most common *unicast* protocols, i.e. on protocols where the devices communicate with each other on a one to one basis. Most of the information in this subsection is not needed for understanding the main contents of the Thesis, but we have included the material for completeness.

The standard division for ad hoc routing protocols is to divide them into *proactive* and *reactive* protocols [TLW02, Raj02]. The former includes all table-driven approaches, i.e. approaches that store routing information, whereas the latter includes all approaches that discover the routing path *on-demand*, i.e. when packets need to be routed. Also variations and combinations of these two approaches exist, but they are omitted in the Thesis. In the rest of this subsection we first overview the most common proactive routing protocols, after which we introduce the most common reactive routing protocols.

Proactive routing protocols

The proactive protocols are essentially modifications of the *distance vector* (DV) and *link state* (LS) Internet routing paradigms [KR02]. In the distance vector approach each node i maintains for **each** destination x a table of distances $d_{ij}(x)$, where j is some neighbour of i . In the routing phase the next hop k is always chosen so that the distance is locally minimized, i.e. so that $d_{ik}(x) = \min_j d_{ij}(x)$.

Topology changes and router (node) failures lead to consistency problems and thus the distance vector information needs to be updated periodically. This can be accomplished simply by transmitting the distance information to neighbouring routers which can then perform the updates locally. However, in ad hoc networks retransmissions are performed also incrementally, when changes in network topology are observed.

An example of a proactive routing protocol for ad hoc networks is the *destination-sequenced distance-vector protocol* (DSDV) that modifies the standard distance vector scheme so that each node maintains a sequence number. Each time a node sends an update message the sequence number is increased [PB94, PB01]. When another node receives the message, it updates the values in the routing table only if the sequence number of the new message is larger than the sequence number stored in the routing table or if the sequence numbers are equal, but the new message has lower distance value. Another distance vector based approach is the *WSR-protocol* [MGLA96], in which each node maintains a minimum spanning tree [CSRL01] of the network structure. This information is maintained by storing in the distance table for each destination also the predecessor node on the shortest route. Because every predecessor node is again a possible destination the distance table contains the predecessor node for that node etc. Other differences in the WSR-protocol are related to the updating of the messages. For example, WSR considers the network topology as a weighted graph and assigns costs to individual edges. Whenever a link fails, the cost of that link is set immediately to infinity. In addition, WSR requires that update messages are explicitly acknowledged by the nodes to which the messages are sent.

Link state algorithms, on the other hand, maintain distance information only about the neighbouring nodes. The neighbourhood information is flooded to the network and eventually all nodes should have the same topology information. The best routing path can then be calculated locally using *Dijkstra's* shortest-path algorithm [Dij59, CSRL01]. The best known link-state algorithm for ad hoc networks is the *Optimized link state routing protocol* (OLSR), which modifies the standard link state algorithm so that control messages are only sent to a subset of all neighbouring nodes [JMC⁺01]. The nodes that belong to this subset are called *multipoint relays* (MPR) and they are selected locally from neighbourhood information so that the set of MPRs covers radio range-wise the entire two-hop neigh-

bourhood of a node. When the node floods information to the network, only the MPRs are assumed to broadcast (flood) further the routing information and thus communications are effectively reduced.

Reactive routing protocols

Contrast to the proactive protocols, reactive protocols construct the routing path on demand. As a consequence, the routing process consists of two phases: *route discovery*, where a node discovers the routing path to the destination, and *routing*, where packets (data) are sent to the network using the discovered route.

If the route discovery phase is performed anew each time a node wants to send packets to the network, approximately half of the packets contain only management information. In this case the communication overhead of the protocol is clearly significant, and, to reduce the overhead, reactive routing protocols employ a *route maintenance* framework where nodes store discovered paths locally. The stored path information is usually updated using different, protocol specific, heuristics.

A good example of a reactive protocol is the *ad hoc on demand distance vector* protocol (AODV) [PR01]. The AODV is based on the DSDV protocol, which was introduced above in the subsection about proactive routing protocols. As in the DSDV, also in AODV each node maintains a distance vector. However, the vector contains entries only for a subset of the nodes and paths for other destinations are discovered on demand. In the route discovery phase a node either replies the distance from its vector or, if the node does not have an existing entry, broadcasts a *route request* message to all of its neighbours. Because the protocol is based on the DSDV protocol, each request furthermore contains a sequence number that is initiated by the source node. If the destination sequence number in the request message is greater than the one stored in the distance table, also then the node broadcasts the route request message.

AODV routes packets on a single hop basis, which means that sent packets contain information only about the destination node and that the intermediate nodes determine the next hop using their own distance tables. The route maintenance

procedure adopted by AODV is based on the assumption that sent messages are observable by neighbouring nodes: when forwarding a message, a node can observe, if a link is broken and it can send an error message to the network. The nodes affected by the error message mark then the end node as unavailable (distance infinity). An additional maintenance step in the AODV is related to node mobility. Namely, once a node moves to another neighbourhood, it sends an incremental update message. However, opposed to DSDV, in AODV the update message results only in local updates.

Another important reactive protocol is the *dynamic source routing* protocol (DSR) [Joh94, JM96, JMB01, JMH04]. As the name suggests, the DSR is a source routing protocol which means that the complete routing path is contained in packet headers. However, while packets are routed in the network, intermediate nodes can modify the routing protocol if they have newer information about the relevant network properties. Most cooperation mechanisms (see Section 2.1) assume that the underlying protocol is based on a source routing scheme and, as the DSR is the most advanced source routing scheme, a separate section (Section 2.3) is reserved for introducing the protocol.

Both the proactive and reactive routing schemes have their advantages and disadvantages. First of all, proactive routing protocols are well suited for networks, where the network topology and link information changes slowly. This is because in such networks update messages are rarely needed. On the other hand, in highly dynamic networks the proactive protocols easily result in constant updates. Furthermore, proactive protocols are not efficient in terms of storage and processing requirements, especially when the number of updates grows. Reactive protocols, on the other hand, have been successfully used within research communities. However, the performance of the protocols is often sensitive to different parameters, such as cache expiration and packet delay timers, and thus the end performance of the protocols can vary radically depending on the network properties.

2.3 Dynamic Source Routing (DSR)

As mentioned above, the dynamic source routing (DSR) protocol is a reactive protocol that is based on the source routing scheme. In other words, routes are discovered when new packets need to be sent to the network and packet headers contain the full routing path. The protocol is also dynamic in the sense that, if the path described in the packet headers is no longer available, the node that discovers the error re-discovers the remaining part of the routing path. The DSR has been extensively tested both in laboratory and in real life situations, which is why the strengths and weaknesses of the protocol are well known. Furthermore, as most cooperation mechanisms are based on a source routing scheme, the DSR serves as a good candidate for testing the mechanisms in real life situations.

The basic version of DSR [Joh94] relies on *flooding*, which means that all messages are sent to all nodes in the network. When a node needs to find a route to some destination node, it sends a *route request* message to all neighbouring nodes. The request message carries with it the IP addresses of the source and destination nodes and a packet identification number. When the route request message arrives at an intermediate node, the node first checks if its IP address is contained in the message already. If this is the case, the message is discarded. Otherwise the node adds its IP address to the headers and sends the message to all neighbouring nodes. Also another pruning step is used in the protocol. Namely, nodes cache the *<source address, packet number>* pairs and, if a node has already sent a response to a particular pair, it is likely that the new request has arrived via a longer path and can thus be discarded. As a consequence, when the message arrives at the destination node, it is likely that it has arrived via the shortest path, which is readable from the message headers. The destination node then sends an acknowledgement (ACK) message to the source node using this path and, when the message arrives at the source node the routing path has been constructed.

After the route has been established it is possible that the topology of the network changes so that some relays are not accessible anymore. However, the protocol should be able to cope with this kind of situations with minimal loss and overhead. In DSR this is accomplished using route rediscovery, which can be initiated at various phases of the routing. For example, if a relay node notices that the next intermediate node on the path is not accessible, the relay node can reinitiate the

route request phase on behalf of the source node. Or if acknowledgements are used, the source node can notice unavailability of a path, if some packets are not acknowledged. The acknowledgement can be done either at the transport level by the relays or at the application level by the destination node. However, for our purposes it satisfies to know that some route rediscovery mechanism is used, if the path to the destination is not available.

Since the introduction of the protocol, various optimizations have been proposed (see, e.g., [JM96], [JMB01] and [JMH04]). These optimizations are usually based on different caching schemes. For example, once discovered routes can be cached and used as the reply in the route discovery phase [JM96]. Another extension is to use blacklists to restrict communications with malicious and non-cooperative nodes [JMH04].

2.4 Watchdog

The reason why most cooperation mechanisms are based on a source routing scheme is that source routing allows gathering path information, which can then be evaluated using knowledge about the behaviour of the nodes on the path. In order to gather the knowledge, a mechanism that monitors constantly the behaviour of nodes in the network needs to be employed. One possible mechanism is the *watchdog* [MGLB00], which gathers behaviour information about neighbouring nodes. Furthermore, the watchdog allows re-evaluating discovered paths so that paths containing malicious or non-cooperative nodes are avoided. Because a similar mechanism is incorporated into many existing cooperation mechanisms, understanding the basic principles of the watchdog is essential for later sections.

In the watchdog, the monitoring is based on the assumption of *omnidirectional antennas*, i.e. that the transmission signals cover a circular area. Under this assumption, the sender can observe whether neighbouring devices forward packets for it or not. However, due to, e.g., obstacles, node mobility and limited transmission energy this assumption sometimes fails. This causes the watchdog to assume that a node is misbehaving although in reality it cooperates. To overcome this problem, a threshold value is typically used to ensure that a node is deemed misbehaving only, when there is enough evidence supporting node misbehaviour.

Implementing the monitoring is relatively simple. Each time a node sends a packet it stores some header information, e.g., packet id and the IP address of the first relay node in a buffer. When the watchdog observes traffic from a neighbouring node, it checks if the IP address and packet id match an entry in the buffer. If there is a match, a positive update is carried out and the entry is removed from the buffer. On the other hand, if, for some neighbouring node, the number of entries in the buffer exceeds the predefined threshold, the node is deemed misbehaving.

In the watchdog, a module called *pathrater* is responsible for re-evaluating discovered paths. In general, the pathrater maintains ratings for nodes and calculates score values for different paths using the rating information. The path with the highest score is then selected for routing. Cooperation mechanisms that use this kind of an approach typically employ their own scoring functions and thus details about how the pathrater calculates the scores are not relevant for our purposes.

The watchdog suffers from several weaknesses. First of all, the generic mechanism actually encourages misbehaviour as paths with misbehaving nodes are excluded, i.e. packets are not routed through them, but packets of the misbehaving nodes are still routed. Secondly, the assumption of omnidirectional antennas is seldom valid in real networks. Furthermore, other aspects, such as collusion formation, have been ignored in the design and the mechanism used by the pathrater is not based on a theoretically justifiable model. Nevertheless, the generic mechanism has proven to be very useful, at least for simulation purposes. Furthermore, the simplicity of the mechanism has ensured that several modifications, which try to correct the abovementioned weaknesses, have been introduced.

3 Game Theory

First attempts to analyse formally decision making in conflict situations were made by such mathematicians as Hugo Steinhaus³, Ernst Zermelo⁴ and Emile Borel⁵ [Kuh04]. However, the foundation for a proper formal treatment was laid by the Hungarian mathematician John von Neumann⁶, when he in 1928 published a paper that discusses decision making in such two person conflict situations where the goals and preferences of the persons are assumed to be exactly opposite, i.e. the gain of one is the loss of the other. Wider interest of research communities was achieved after von Neumann published together with economist Oskar Morgenstern⁷ the book *Theory of Games and Economic Behaviour* [vNM04]. The book formalises a general theory for decision making in conflict situations with arbitrary many opponents and since the publication of the book the theory has been known as *game theory*. From that point onwards, game theory has been an active research area especially in (applied) mathematics and economics.

General interest towards game theory arose in 1994 when John Nash⁸, John C. Harsanyi⁹ and Reinhard Selten¹⁰ were awarded the Nobel Prize in Economics¹¹ for their achievements in game theory. As a consequence, in the last decade applications of game theory have started steadily to emerge in various research fields such as biology, philosophy and computer science. The diversity of these fields serves to emphasise that game theory can be seen as a generic tool for modelling arbitrary decision making situations where the goals of the participants conflict.

In the rest of this section we introduce various game theoretic concepts. The first three sections contain background information that is needed for understanding the other sections. The next four sections, on the other hand, contain material that is useful for obtaining a better understanding of the existing game theoretic

³Hugo Dyonizy Steinhaus, Hungarian mathematician, 14.01.1887 - 25.02.1972.

⁴Ernst Friedrich Ferdinand Zermelo, German mathematician, 27.07.1871 - 21.05.1953.

⁵Félix Edouard Justin Emile Borel, French mathematician, 07.01.1871 - 03.02.1956.

⁶John (János) von Neumann, Hungarian-American mathematician 28.12.1903 - 08.02.1957.

⁷Oskar Morgenstern, German-born American economist, 24.01.1902 - 26.07.1977.

⁸John Forbes Nash, American mathematician, 13.06.1928 -

⁹John Charles Harsanyi, Hungarian-American economist, 29.05.1920 - 09.08.2000.

¹⁰Reinhard Selten, German economist, 10.10.1930 -

¹¹The exact name of the prize is the *Bank of Sweden Prize in Economic Sciences in Memory of Alfred Nobel*, but as the Nobel Prize is better recognizable, we use that instead.

models that are introduced in Section 5.1. With the exception of Section 3.6, these sections are not necessary for understanding the main contents of the Thesis, but we have included the material for completeness. Finally, the last section contains material that is used in Section 5.3 to derive a new theoretical model for routing and thus that is relevant for understanding the Thesis.

We begin in Section 3.1 by covering basic terminology, such as strategies and the concept of equilibrium. Practical insights about the basic concepts are given in Section 3.2, whereas Section 3.3 presents a generic formulation for a game, the *extensive form of a game*. Section 3.4 introduces repeated games and Section 3.5 discusses evolutionary game theory. In Section 3.6 another family of games, Bayesian games, is introduced, and in Section 3.7 these concepts are used to introduce the field of mechanism design. Finally, Section 3.8 discusses adaptive learning in Bayesian games.

3.1 Basics

In order to have a conflict situation, multiple actors with at least partially conflicting goals are needed. In game theory the actors are called *players*. The collection of players is a finite set, which is denoted by N . Individual players are indexed using variable i and thus for all i we have $i \in N$. The number of players in an arbitrary conflict situation is denoted by n . In applications of game theory the nature of the players can be quite arbitrary, but usually the players are individual persons, groups, companies or other similar entities. However, also, e.g., the conflicting goals of an individual person can be modelled as players. In ad hoc networks, the players are the devices that participate in the functions of the network.

In order to achieve her¹² goal(s) (satisfy her interests) each player needs to act in the conflict situation. The nature of the actions depends on the application at hand; for example, in routing situations in ad hoc networks the actions are related to sending packets to the network. The action of player i is denoted by a_i and the set of all possible actions of player i is denoted by A_i . The set A_i is called the *action space* of player i . The Cartesian product of the action spaces of

¹²The female form is conventionally used in game theory to refer to arbitrary players.

the individual players is the action space of the conflict situation and it is denoted by \mathcal{A} , i.e.

$$\mathcal{A} = \times_{i=1}^n A_i. \quad (1)$$

An element of the overall action space is a vector whose i^{th} component is the action performed by player i . This vector is called an *action profile* and it is denoted by \mathbf{a} . Formally an action profile is defined as

$$\mathbf{a} = (a_1, \dots, a_n). \quad (2)$$

Another important concept is the *deleted action profile* of player i which contains the actions of all other players except i , i.e.

$$\mathbf{a}_{-i} = (a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_n). \quad (3)$$

An action profile determines, how the conflict situation is resolved and thus determines the end of the conflict. From individual player's point of view, some endings are preferable to others¹³. The mathematical way to model the preferences of the players is to define for each player i a preference relation \succeq_i over the possible endings. However, unfortunately this leads to cumbersome set theoretic analysis. Luckily, von Neumann and Morgenstern show, how the preference relation can be replaced with a numerical quantity called the *utility* of a player [vNM04, Chap.3]. Not all preference relations can be expressed as utility functions and to ensure that the utilities capture correctly the preferences of the players, the preference relation needs to satisfy a set of axioms [vNM04, Sections 3.5 - 3.6]. The most crucial axiom is that the utilities need to preserve the ordering of the preferences over the different endings. Thus, if v and v' are two different endings and v is preferred to v' , the utility assigned to v should be greater than the utility assigned to v' . The other axioms have been later replaced with other mathematical properties and as a consequence they are not relevant for our purposes.

¹³To clarify the distinction between endings and action profiles: an ending is related to the original set theoretic definition of a game, according to which the game ends in a state whose goodness is expressed by a preference relation and nothing is said about what actions the players have performed. The action profile, on the other hand, contains the actions that the players have performed during the game play and hence the action profile clearly defines the ending of the game, but not vice versa.

As different action profiles determine potentially a different ending for the conflict situation, the *utility function* of player i is defined to be a function that assigns a real-valued number to each possible action profile. This function is denoted by $u_i(\cdot)$ and it is a mapping from the action space of the conflict situation to a subset of the one dimensional Euclidean space \mathbb{R} , i.e.

$$u_i : \times_{i=1}^n A_i \rightarrow \mathbb{R}. \quad (4)$$

As an action profile completely determines the ending of the conflict situation, the axiom of von Neumann and Morgenstern can be restated and formalised as follows:

Axiom 1 *If ending v , associated with action profile a , is preferred to ending v' , associated with action profile a' , by player i , the utility of player i is greater for action profile a than for action profile a' , i.e.*

$$v \succeq_i v' \Rightarrow u_i(a) > u_i(a'). \quad (5)$$

The vector function \mathbf{u} , which is defined as

$$\mathbf{u}(\mathbf{a}) = (u_1(\mathbf{a}), \dots, u_n(\mathbf{a})), \quad (6)$$

determines the utility of each player in the conflict situation and the resulting vector $\mathbf{u}(\mathbf{a})$ is called the outcome of the game. Thus formally the outcome is determined by the mapping

$$\mathbf{u}(\mathbf{a}) : \times_{i=1}^n A_i \rightarrow \mathbb{R}^n. \quad (7)$$

According to rationality arguments, in a conflict situation the players should strive for the outcome that maximizes their personal utility as, by definition, it represents the most preferable end result. This assumption is called the *rationality assumption* and it is central in game theoretic analysis. The process of selecting, which actions to perform so that the optimal outcome is achieved, is called the *strategy* of a player.

Strategies are further divided into *pure* and *mixed* strategies. Pure strategies are strict choices of actions, whereas mixed strategies are randomizations over the

pure strategies. For example, in a routing situation in ad hoc networks, a pure strategy corresponds to forwarding always the same share of packets in a specific situation. Respectively, a mixed strategy corresponds to a probability distribution over the possible packet shares. Pure strategies of player i are denoted using the variable s_i and mixed strategies using the variable π_i . Analogously to earlier definitions, the *pure* and *mixed strategy spaces* of a player are defined as the collections of all possible strategies of the respective kind. These collections are denoted by the variables S_i and Π_i . The overall pure and mixed strategy spaces of the conflict situation are defined to be the Cartesian products of the individual pure and mixed strategy spaces. These spaces are denoted by S and by Π . Finally, *pure* and *mixed strategy profiles* of a conflict situation are defined as

$$\mathbf{s} = (s_1, \dots, s_n) \text{ and } \pi = (\pi_1, \dots, \pi_n). \quad (8)$$

Pure strategies are easier to analyse as our definition of utility is well defined for pure strategies. However, for mixed strategies the earlier definition needs to be replaced with an expectation over the utilities of the possible actions. Let $p_{i,\pi_i}(a_i)$ denote the probability that player i chooses action a_i when she is selecting her actions according to mixed strategy π_i . Accordingly, the expected utility of player i , with respect to the mixed strategy π_i , is defined as

$$u_i(\pi_i, \mathbf{a}_{-i}) = \sum_{a_i \in A_i} p_{i,\pi_i}(a_i) u_i(a_i, \mathbf{a}_{-i}), \quad (9)$$

where $u_i(a_i, \mathbf{a}_{-i})$ is the utility of an action profile \mathbf{a}' , whose components $1, \dots, i-1$ and $i+1, \dots, n$ are given by \mathbf{a}_{-i} and whose i^{th} component is a_i .

Another important concept related to mixed strategies is the *support* of a mixed strategy, which is the set of actions to which the strategy assigns a positive probability, i.e.

$$\text{Supp}(\pi_i) = \{a_i | p_{i,\pi_i}(a_i) > 0\}. \quad (10)$$

The concepts that we have defined thus far allow renaming the conflict situation and the decision making process related to the conflict situation as a *game* that the players are *playing*. Formally a game is defined as follows:

Definition 1 A game (in normal form) is a three-tuple $\langle N, \mathcal{S}, \mathbf{u} \rangle$, where N is the set of players, \mathcal{S} is the pure strategy space of the game and \mathbf{u} is the utility function of the game as given by Equation 6.

As mixed strategies are linear combinations of the pure strategies, they do not have to be explicitly mentioned in the definition.

According to the rationality assumption each player attempts to maximize her personal utility and thus an optimal outcome of the game is such that it maximizes the utility of each player. In game theory this kind of outcome is called an *equilibrium* and the main goal of game theoretic analysis is to determine, whether particular games have equilibriums and, if so, how can the players reach an equilibrium. The most famous theoretical result is by American mathematician John Forbes Nash, who proved that, if the utility functions are continuous functions, every finite (normal-form) game has a mixed strategy equilibrium [Nas50]. In honour of this proof, the equilibrium of a game is called the *Nash equilibrium* of that game. The proof of this theorem is central to game theory and it can be found in most game theory books (e.g. [FT91]).

The strategies of the individual players that constitute the strategy that leads to an equilibrium are called equilibrium strategies and they are denoted by s_i^* and by π_i^* . Formally an equilibrium is determined as a strategy profile that satisfies for each player i the relation

$$s_i^* \in \arg \max_{s_i} u(s_i, \mathbf{s}_{-i}). \quad (11)$$

Thus an equilibrium \mathbf{s}^* (or π^*) is defined as a strategy profile whose each component is an equilibrium strategy, i.e.

$$\mathbf{s}^* = (s_1^*, \dots, s_n^*) \quad \text{and} \quad \pi^* = (\pi_1^*, \dots, \pi_n^*). \quad (12)$$

As a summary of the contents of the section we conclude that, in order to use game theory for modelling conflict situations, the players of the game, the actions of the players and the utility functions of the players need to be defined. Furthermore, the resulting game is analyzed to see if it has equilibriums and, if so, how can the players reach one. Equilibrium analysis is usually done analytically, but some numerical methods have been developed for specific situations. An overview of this topic is given in [].

	C	D
C	1, 1	-1, 2
D	2, -1	0, 0

Table 1: Matrix representation of the prisoners' dilemma [FT91]. The equilibrium strategy (0,0) is denoted in bold.

3.2 Example: Prisoners' Dilemma

To give better insights to the concepts of the previous Section, we consider a famous game theoretic problem called the *prisoners' dilemma*. Fudenberg and Tirole [FT91] describe the problem as follows:

Two prisoners are arrested for a crime. The police lack sufficient evidence to convict either suspect and consequently need them to give testimony against each other. The police put each suspect in a different cell to prevent the two suspects from communicating with each other. The police tell each suspect that if he testifies against the other (doesn't cooperate with), he will be released and will receive a reward for testifying, provided the other does not testify against him. If neither suspect testifies, both will be released on account of insufficient evidence, and no rewards will be paid. If one testifies, the other will go to prison; if both testify both will go to prison, but they will still collect the rewards for testifying.

In the prisoners' dilemma the set of players consists of two players, who have two possible actions: to *defect* (D), i.e. testify, or to *cooperate* (C). The utility values of the different outcomes are not uniquely determined, but according to Axiom 1, the relationships between the utility values are always the same. A possible configuration for the values of the different outcomes is given in Table 1.

The equilibrium analysis of the prisoners' dilemma can be performed using a procedure called *elimination of dominated strategies* [FT91]. The procedure looks for strategies which are never played and removes them from further analysis. For example, in the prisoners' dilemma, player I should never cooperate because her utility is always greater if she defects. Thus the outcomes on row C of Table

1 are excluded from further analysis. The same line of reasoning applies also to player II and thus all other outcomes except $(defect, defect)$ are excluded. As this choice of strategies maximizes the utilities of the individual players, $(defect, defect)$ is the unique Nash equilibrium of the game.

The importance of the prisoners' dilemma was identified by Robert Axelrod, according to whom the prisoners' dilemma is an abstract formulation of situations, in which what is best for each for each person leads to mutual defection, whereas everyone would have been better off with mutual cooperation [Axe84]. Thus, the prisoners' dilemma can be seen as a generic model that allows to study under which conditions cooperation emerges. As this issue has close connections to evolutionary game theory, we return to the topic in Section 3.5.

3.3 Extensive form games

The definition of a game in normal form, given in Definition 1, states nothing about the order in which the players make their decisions, nor does it state anything about what the players know before making their decision. Thus the normal form of a game is not suitable for modelling games with dynamic structure or games, where players have different amounts of information available. However, these aspects are important for many practical applications and an alternative formalisation for a game is required. A suitable formalisation for this purpose is the *extensive form* of a game [Kuh53], which, among other things, makes the order of moves and the information of the players explicit. Furthermore, the extensive form has a natural geometric interpretation, which makes it ideal for visualising games.

The basic component of the extensive form is the *game tree* which is a directed graph. The variable \mathcal{T} is used to denote the game tree and the variables \mathcal{V} and \mathcal{E} are used to denote the collections of vertices and edges of the graph, i.e. $\mathcal{T} = (\mathcal{V}, \mathcal{E})$. As before, the set of players is denoted by N and individual players are indexed using the variable i . The vertices are further divided into sets of terminal and non-terminal vertices. Terminal vertices have no successors, i.e. child vertices, and they define an outcome for the game. Each non-terminal vertex $v \in \mathcal{V}$ is labelled using one of the players of the game and V_i is defined to be the collec-

tion of vertices that are labeled with i . At each non-terminal vertex $v \in V_i$ player i needs to perform an action. The set of actions available to player i at vertex v is denoted by A_v . The action space of player i is clearly the union of the sets A_v , i.e. $A_i = \bigcup_{v \in V_i} A_v$. The actions are also used as labels for the edges so that there are $\#(A_v)$ edges from vertex v to other nodes, where $\#$ is the set cardinality operator, i.e., number of elements in a set, and each edge is associated with an action $a \in A_v$.

The game tree is assumed to have a single initial vertex from which the game play starts. At each vertex v the player whose label is assigned with the vertex v performs an action a from the set A_v and the game play continues from the end vertex of the edge associated with action a . When a terminal vertex is reached, the path from the root vertex to the terminal node constitutes a sequence of actions, which can be associated with a payoff. Thus the path fully defines the outcome of the game.

The players seldom have complete information about the actions the other players have made earlier in the game. The information about previous actions that are known is called the *history* of game play and it is denoted by h . In order to model the uncertainty about previous actions, the concept of *information sets* [Kuh53] is required.

Whereas the sets V_i partition the set of vertices V according to whose turn it is to move, information sets partition sets V_i into subsets of vertices of which the player i is indifferent of. Thus, if v and v' are two vertices of player i and player i does not know the complete history of game play, i.e. whether she is at vertex v or at vertex v' , the vertices v and v' belong to the same information set. The partitioning is required to be such that the player always knows the information set at which she is located, but not at which vertex within the information set. The information sets of player i are denoted by I_{i_j} and by definition $V_i = \bigcup_j I_{i_j}$. The variable \mathcal{I} is used to denote the overall partitioning of the set of (non-terminal) vertices \mathcal{V} into information sets and, analogously, the variable I_i is used to denote the partitioning of V_i into information sets. When extensive form games are visualised, vertices belonging to the same information set are connected using a dashed line.

Although a player might not know the vertex she is currently located at, the player is assumed to have (subjective) beliefs about the possible vertices. These beliefs are modelled using probabilities, which are, for now, assumed to be fixed. The probability distribution associated with a single information set I_{i_j} is denoted by $\mu_{I_{i_j}}$. Furthermore, the variable μ_i is used to denote the collection of probability distributions for player i and the overall collection of probability distributions of the game is denoted by μ .

The concepts introduced thus far allow us to define the extensive form of a game as follows:

Definition 2 *A game in extensive form is a six tuple $\langle N, \mathcal{T}, \mathbf{u}, \mathcal{A}, \mathcal{I}, \mu \rangle$, where N is the set of players, \mathcal{T} is the game tree, \mathbf{u} is the utility function of the game, \mathcal{A} is the action space of the game, \mathcal{I} is the partitioning of the non-terminal vertices of the game tree into information sets and μ is the probability distribution associated with \mathcal{I} .*

Strategies in extensive form games are called *behaviour strategies*. Let I_{i_j} be an information set of player i and $\hat{\sigma}_i(I_{i_j})$ a probability distribution that assigns a probability over the actions that are available at information set I_{i_j} ¹⁴. The distribution $\hat{\sigma}_i(I_{i_j})$ is called a *local strategy* [Sel75] and a mapping σ_i that assigns a local strategy for each information set I_{i_j} of player i is called a *behaviour strategy*. Let σ be a behaviour strategy profile of the game, i.e. $\sigma = (\sigma_1(I_1), \dots, \sigma_n(I_n))$. The vertices v , to which σ assigns a positive probability are called *on-the-path* vertices and the other vertices are called *off-the-path* vertices¹⁵.

The basic equilibrium concept in the theory of extensive games is the Nash equilibrium. However, as extensive games can often have multiple Nash equilibrium, various refinements have been developed. The goal of these refinements is to,

¹⁴The proper definition of behaviour strategies is that they assign a probability distribution over the actions available at an information set conditional on the event the information set is reached [VR03]. However, we will consider only games of perfect recall (see next section), in which case a Theorem by Kuhn [Kuh53] states that behaviour strategies are equivalent to mixed strategies.

¹⁵A further clarification: μ_{I_i} is the probability distribution that assigns probabilities over vertices in information set I_i and $\sigma(I_i)$ is a probability distribution over the actions that are available at information set I_i . Because an action causes the play to advance to some vertex v , σ is used in the definition of off-the-path and on-the-path vertices.

first of all, rule out outcomes that are unlikely, and, secondly, to handle actions that cause the game play to progress to an off-the-path information set. The most important refinement is *subgame perfection* [Sel65, Sel75], but before discussing it in detail, a definition for a subgame is needed.

A subgame is a game that is part of another game. In order to be properly defined as a game, the subgame must have a unique root and the information sets must be well defined, i.e. information sets of the original game need to be preserved. A subgame is said to be *trivial*, if it contains either the whole game tree or only a single vertex of the game tree. A subgame that is not trivial is said to be *proper* (or sometimes non-trivial). For example, the game on the left hand side of Figure 2 has seven subgames: $\{\{v_1, v_3, v_4\}, \{v_2, v_5, v_6\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_6\}\}$ and the complete game. Of these the subgames $\{v_1, v_3, v_4\}$ and $\{v_2, v_5, v_6\}$ are proper and the others are trivial. On the other hand, the game on the right hand side has five subgames that are all trivial.

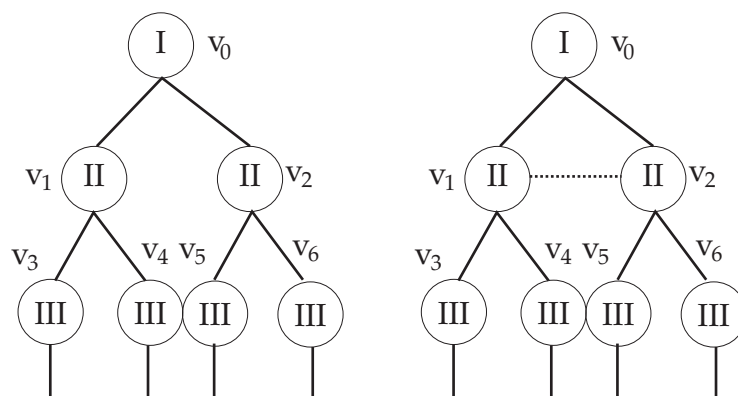


Figure 2: Two examples of extensive form games.

The definition of subgame perfection states that a strategy is subgame perfect, if the restriction of the strategy to an arbitrary subgame of the original game is Nash equilibrium of that subgame. Thus a strategy is subgame perfect, if it is optimal everywhere in the game. Subgame perfection can be identified using a procedure called *backward induction*. Backward induction starts from a terminal vertex and works upwards towards the root. At each intermediate information set the procedure selects the optimal action and when the procedure finds a path from a terminal vertex to the root vertex it has found an equilibrium strategy profile that is subgame perfect.

In order to give better insights to the concepts, we consider again the prisoners' dilemma, whose game tree is depicted in Figure 3. As the game has two players I and II we have $N = \{I, II\}$. The game tree has tree vertices, denoted by v_0, v_1 and v_2 . The set V_I is a singleton set containing vertex v_0 , whereas V_{II} contains vertices v_1 and v_2 . However, both players have only a single information set: $I_I = \{v_0\}$ and $I_{II} = \{v_1, v_2\}$. At each node, the players have available to them two actions: cooperate (C) and defect (D). Thus $A_v = \{C, D\}$ for each vertex $v \in \mathcal{V}$. The only subgame of the game is the game itself and thus subgame perfection does not apply for this game. However, from earlier analysis (Section 3.2) we know that the game has a unique equilibrium point (D, D) .

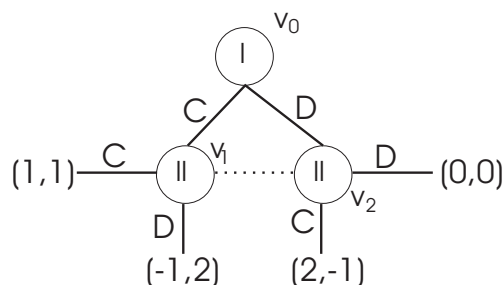


Figure 3: The game tree of the prisoners' dilemma.

As another example we consider a signalling version of the prisoners' dilemma, in which the action of the first player is told to player II before she is required to act. The modified game tree is shown in Figure 4. In this case the game has three subgames and subgame perfection can be applied. Namely, if player I cooperates, the best strategy of player II is to defect. Similarly, if player I defects, the best strategy of player II is to defect. However, the alternative (C, D) can be ruled out with backward induction. If we start from vertex v_1 , the best strategy of player II is to defect. However, at vertex v_0 the best choice of player I is to defect and thus (C, D) is never played and (D, D) is the unique equilibrium of the game.

3.4 Repeated games

In many practical applications of game theory the conflict situation reoccurs at different instances of time and, when this is the case, the actions of the players

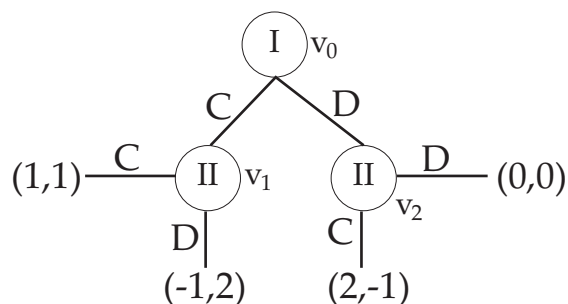


Figure 4: A signalling version of the prisoners' dilemma.

may affect their future utilities. For example, in ad hoc networks the decision of how many packets to forward for a particular node affects the future willingness of the other node to forward packets in return. These kind of games are called *repeated games* and they are further categorized into *infinitely* and *finitely* repeated games. In infinitely repeated games, the end of the game is not foreseen by any of the players, whereas in finitely repeated games all the players know when the game will end. Although this distinction might seem subtle, it affects significantly the equilibrium analysis of the games as will be shown later in this Section.

The basic building block of a repeated game is the *stage game*, which is the game that is repeated over time. The sequence of stage games is called the *repeated game*. The stage game is assumed to be finite and the players are assumed to have no information about the actions the others have made in the current repetition of the stage game. Furthermore, it is assumed that the environment is stationary, i.e. that the payoffs and actions are the same in every stage game.

Although players are assumed to have no information about the actions the others have made during the current stage game, they may have information about actions performed in earlier repetitions. For example, in games with observable actions, which are our main focus, the stage game is played repeatedly so that after each repetition, the actions of the previous play of game are revealed to all players. Thus, at the beginning of each new stage, the players have full information about what has happened during the previous stages. We also assume that the players have *perfect recall*, which means they do not forget information. The repetitions are indexed with a superscript t and the information about actions performed during the previous repetitions constitutes the history of the repeated

game. The history information available in the t^{th} repetition is denoted by h^t and by definition the history of the repeated game is the collection of stage game action profiles, i.e. $h^t = \{a^0, \dots, a^{t-1}\}$ ¹⁶. We note at this point that observable actions and perfect recall imply that each player knows at the beginning of stage t the entire history h^t .

The concept of history allows the players to condition their actions on past game play. Thus strategies in the t^{th} repetition of the stage game assign actions or probability distributions over the actions to possible histories h^t . For example, $\sigma_i^t = (\sigma_i, h^t)$ defines a behaviour strategy for player i in the repeated game. As the repeated game is merely a sequence of stage games, also the strategies in the overall game are sequences of stage game strategies so that $\sigma_i = \{\sigma_i^0, \dots, \sigma_i^T\}$, where T is the last repetition of the game, which is allowed to be infinity.

The actual selection of actions should naturally be dictated by the utilities assigned to individual actions. However, as actions now may affect future income (utility), the selection should look at a longer period of time. In finitely repeated games the overall utility can often be calculated as a sum of the utilities of the individual stage games. Alternatively, the average payoff over all future repetitions can be used. In the infinite case this is not feasible and thus a discounted sum, given by Equation 13, is used instead [Abr88].

$$u_i(s_i) = \sum_{t=0}^T \gamma^t u_i(\sigma_i^t, \sigma_{-i}^t), \quad 0 < \gamma \leq 1. \quad (13)$$

The variable γ is a discount factor which states that the further we go from the current situation, the less valuable the utility is to us now.

The theory of repeated games does not introduce new equilibrium concepts, but, due to discounting, in infinitely repeated games the structure of Nash equilibriums may differ from those of the individual stage games. To illustrate this, we consider the repeated prisoners' dilemma, which is illustrated in Figure 5.

¹⁶For clarification: h denotes the information available within a single stage, which by definition of a repeated game equals \emptyset . The variable h^t on the other hand denotes all information about previous repetitions. If the stage games were allowed to be extensive games with a dynamic structure, the information available would be given by $h^t \cup h$.

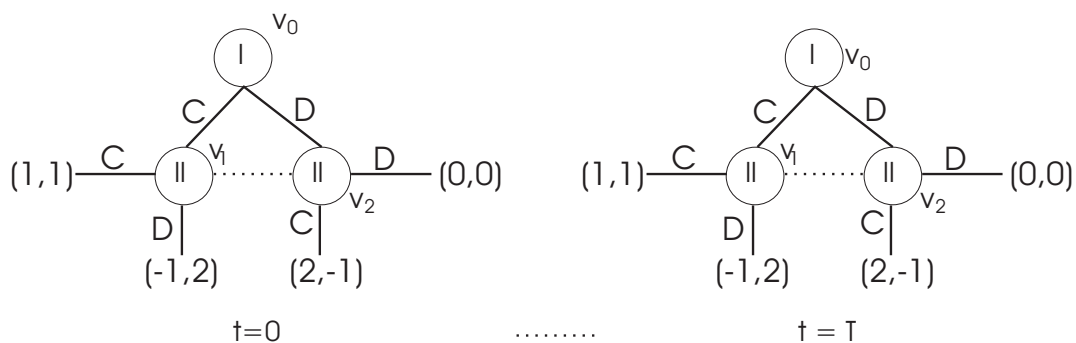


Figure 5: The repeated prisoners' dilemma.

We consider first the case where T is finite. In this case the optimal action of a player is to always defect in the last repetition. Namely, if one of the players cooperated, the other could increase her overall payoff from the game by defecting in the last period. However, as the other player expects this behaviour, she also defects. The same argument holds for period $t - 1$ and, furthermore, for period $t - 2$ etc. Thus, according to backward induction, the unique Nash equilibrium of the finite case is for both players to always defect. This equilibrium point is also subgame perfect.

In the infinitely repeated case the optimal strategy of a player is more complicated. Namely, if the other player has defected (at least) once, the equilibrium strategy is to always defect. However, if the other player has cooperated thus far, the optimal strategy is to cooperate, if the discounted payoff exceeds the value obtained from defection. Consider first the case where the player defects. In this case her future payoff is $2 + 0 + 0 + \dots = 2$. On the other hand, if she decides to cooperate and assuming that $0 < \gamma < 1$, her payoff is

$$\sum_t \gamma^t \cdot 1 = \frac{\gamma}{1 - \gamma}. \quad (14)$$

Thus, the optimal strategy of the player is to cooperate if $\gamma/(1 - \gamma) \geq 2$. The equality holds when γ equals 0.5 and when γ exceeds this, the player should cooperate. It can furthermore be proven that this is a subgame perfect (Nash) equilibrium of the infinitely repeated prisoners' dilemma. Analysis of the case $\gamma = 1$ requires to use a so-called *folk theorem* [FT91], but discussion about folk theorems is out of scope for the Thesis.

3.5 Evolutionary games

An alternative approach to the theory of repeated games is *evolutionary game theory* [Wei97], which was initially introduced as a way to explain why certain kind of behaviour survives Darwinian selection in evolutionary processes [Smi82]. However, it was soon discovered that the theory is also suitable for explaining, why cooperation emerges in situations, where selfish individuals interact with each other without central authority [Axe84]. Since then the theory has found applications in numerous fields, such as foreign relations theory [Axe84, BS97] as well as ad hoc networks [SNCR03]. We return to the latter case in Section 5.1.

In general, evolutionary game theory studies repeated pair wise interactions of individuals in populations, where each individual is assumed to be genetically programmed to play according to a specific pure strategy. Let N be the overall population and $\mathbf{x} = (x_1, \dots, x_k)$ the share of individuals (percentage of players in N) playing according to pure strategy s_j ($j = 1, \dots, k; \sum_{j=1}^k x_j = 1, x_j \geq 0$). Assume furthermore that at arbitrary instances of time a pair of individuals is randomly drawn from the population to play a specific game, such as the prisoners' dilemma. The expected payoff to strategy s_m in this game, given the population share \mathbf{x} , is

$$u(s_m)_{\mathbf{x}} = \sum_{j=1}^k x_j u(s_m, s_j), \quad (15)$$

where $u(s_m, s_j)$ is the payoff to strategy s_m , when the opponent plays according to strategy s_j ¹⁷.

In biological terms the utility $u(s_m)$ reflects the fitness of the strategy and, according to evolutionary principles, the fitness of the strategy should affect how the share of strategy s_m increases or decreases in the population over time. The model that governs the changes in the shares of the strategies is called the *population dynamics* [Smi82]. The biologically motivated choice for the population dynamics is to use the *replicator dynamics* [Fis30, Wei97], according to which the change rate of a strategy's population share depends on the difference between the expected payoff of the strategy and the average expected payoff of all strate-

¹⁷For simplicity, we consider only two-player games. For a more thorough treatment, we refer to [Wei97].

gies. In other words, the more the expected payoff exceeds the average payoff, the more the share increases. Respectively, if the payoff is less than the average payoff, the share decreases in a similar fashion.

Equilibrium analysis of evolutionary games differs significantly from the models discussed in the previous subsections. Instead of trying to find an optimal strategy profile for the players, evolutionary game theory attempts to identify strategies that survive evolutionary selection. To this end, assume that at an arbitrary instance of time the population undergoes a mutation so that a small subset of the population is reprogrammed to play s_n instead of s_m . Let ϵ be the amount with which the population share of s_m is changed and let \mathbf{x}_ϵ be the (perturbed) population share vector, which is derived from \mathbf{x} by decreasing s_m by the amount of ϵ and increasing s_n by ϵ . The strategy s_m is said to be *evolutionary stable* [Smi82], if

$$u(s_m)_{\mathbf{x}} > u(s_m)_{\mathbf{x}_\epsilon} \quad (16)$$

holds for all alternative strategies s_n ($n \neq m$) with a positive value of ϵ ¹⁸. If the inequality is not strict, i.e. $u(s_m)_{\mathbf{x}} \geq u(s_m)_{\mathbf{x}_\epsilon}$, strategy s_m is said to be *weakly evolutionary stable*. When s_m is evolutionary stable, the maximum value of ϵ , for which Equation 16 holds, is called the *invasion barrier* of strategy s_m .

3.6 Bayesian Games

In many practical applications of game theory, such as in ad hoc networks, a player can have private information that is only known to the player herself, but that affects her decisions. When this is the case, the game is called a game of *incomplete information*. The classes of games that we have discussed in the previous sections are not suited for modelling the uncertainty in the information of others and thus an alternative class of games, *Bayesian games* [Har67, Har68a, Har68b], needs to be introduced (for background information on Bayesian probability theory see Appendix 2). We first consider the case, where all players move simultaneously, i.e. the game does not have a dynamic structure.

¹⁸As an additional note we mention that the evolutionary stable strategies correspond to attractors [Spr03] of the population dynamics of the evolution process.

In Bayesian games the private information of the players is modelled by defining a set of *types* for each player. The set of types is assumed to be discrete and each type corresponds to a particular set of private information. The currently active type of a player determines, how she is assumed to behave in the game. The types of a single player are denoted by θ_i and the set of all possible types, the *type space*, of a player, is denoted by Θ_i . Accordingly, a type profile of the game is

$$\theta = (\theta_1, \dots, \theta_n), \quad (17)$$

and the type space of the game is

$$\Theta = \times_{i=1}^n \Theta_i. \quad (18)$$

As an example of the concept of type, we consider ad hoc networks, in which the type typically corresponds to the energy class of a node, i.e. a discretized representation of the remaining energy¹⁹.

The currently active type of each player is assumed to be selected by an additional player, *nature*, who is assumed to have no interest in the outcome of the game. In the (nowadays) standard approach, the types of the players are selected and communicated to the players before any actions are performed²⁰. The selection of types is assumed to be done by drawing the types of the players from a common *objective* distribution p , defined over the type space of the game. The objective distribution is assumed to assign a strictly positive value to each possible type profile of the game, i.e. $p(\theta') > 0$ holds for all $\theta' \in \Theta$. Each player only knows the distribution of her own types, i.e. player i only knows the distribution $p(\theta_i)$. However, the players are assumed to have beliefs about the unknown variables and these beliefs are represented as *subjective probability distributions*. Thus, each player has a belief distribution defined over the possible deleted type profiles, θ_{-i} , given her own type θ_i . This distribution is denoted $\mu(\theta_{-i}|\theta_i)$ ²¹.

¹⁹To date in ad hoc networks the concept of type has only been used for the energy class of a node. However, alternative uses may also be possible.

²⁰This is the so-called *prior-lottery model*, introduced by John C. Harsanyi. In the alternative formulation the types of the players are selected *after* they have performed an action. This, so-called *posterior-lottery model*, approach was proposed by Reinhard Selten. Both approaches were initially introduced in [Har67] as the latter model was communicated to Harsanyi by Selten and it was never published by Selten himself.

²¹The beliefs over the types of others correspond to probabilities over vertices in an information set and thus both concepts are denoted by the variable μ .

In Bayesian games a strategy for a single player needs to assign an action for each type of the player. Let $\theta_i = \{\theta_{i_1}, \dots, \theta_{i_k}\}$ be the set of types of player i and $\sigma_i(\theta_i)$ a behaviour strategy vector that assigns a probability distribution over the possible actions available at each information set and type of player i . The expected utility of a strategy for a given player is then defined as the expectation of the belief distribution over the possible deleted type profiles θ_{-i} given the players own type θ_i , i.e.

$$\sum_{\theta_{-i} \in \Theta_{-i}} \mu(\theta_{-i} | \theta_i) u(\sigma_i(\theta_i), \sigma_{-i}(\theta_{-i})). \quad (19)$$

The set of optimal strategies for player i is thus defined as the set of strategies σ_i^* that satisfy the following condition:

$$\sigma_i^*(\theta_i) \in \arg \max_{\sigma_i' \in \Sigma_i} \sum_{\theta_{-i} \in \Theta_{-i}} \mu(\theta_{-i} | \theta_i) u(\sigma_i'(\theta_i), \sigma_{-i}(\theta_{-i})). \quad (20)$$

Respectively, a strategy profile $\sigma(\theta) = (\sigma_1(\theta_1), \dots, \sigma_n(\theta_n))$ is said to be a *Bayes-Nash equilibrium* [Har68b], if for each player $\sigma_i(\theta_i)$ belongs to the set of optimal strategies $\{\sigma_i^*(\theta_i)\}$.

As an example of a Bayesian game, we consider a modified prisoners' dilemma. In this version, the first player has two types: cooperative and non-cooperative. The cooperative type is assumed to always cooperate and, respectively, the non-cooperative type is assumed to always defect. The second player is assumed to have only one type, for which both the actions (defect and cooperate) are available. The game tree is illustrated in Figure 6.

The strategy of the first player is trivial: as she has only one possible action per type, her optimal strategy is to perform the only available action when the corresponding type is selected. Regarding the second player, she must have beliefs over the possible types of player I. Let p_1 be her belief that player I is cooperating and $1 - p_1$ her belief that player II is defecting. Player II has two possible strategies: to cooperate or to defect. According to Equation 19, the expected utility of cooperating is,

$$u_{II}(C) = p_1 - 1(1 - p_1) = 2p_1 - 1. \quad (21)$$

Respectively, the expected utility of defecting is

$$u_{II}(D) = 2p_1 + 0(1 - p_1) = 2p_1. \quad (22)$$

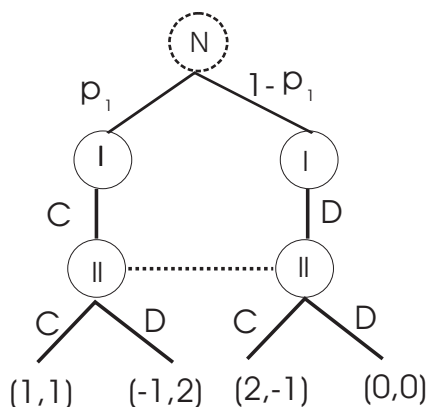


Figure 6: The modified version of the prisoners' dilemma. The dotted circle is used to represent nature.

Thus, irrespective of the beliefs of player II, the Bayes-Nash equilibrium of the game is $\{(C, D), (D, D)\}$, i.e. player II to always defect.

If the game has a dynamic structure, some of the Bayes-Nash equilibriums may be unwanted. The concept of *subgame perfection* was used in Section 3.3 to rule out the unwanted equilibriums. However, in Bayesian games the players do not know each others types and the game has no proper subgames. As an example, we consider the game in Figure 7 ("Selten's horse"). The only subgame of the game is the game itself. The game has two pure strategy equilibriums: (D, a, l) and (A, a, r) . The first of these is not sensible as, if the information set of player II is reached, her optimal action is d instead of a .

In the literature several refinements have been proposed. The most fundamentals of these are the *perfect Bayesian equilibrium* (PBE) [Sel75, FT91] and *sequential equilibrium* [KW82b]. Both of these equilibrium concepts add a consistency check on the set of equilibrium points and equilibrium points failing the check are ruled out as unwanted. In this Thesis we concentrate on sequential equilibrium as it plays an important role in later sections. For information about the PBE, we refer to [Rat93].

In games with incomplete information, the beliefs should affect the decision making and thus we define an *assessment* to be a strategy-belief pair (σ, μ) , which is

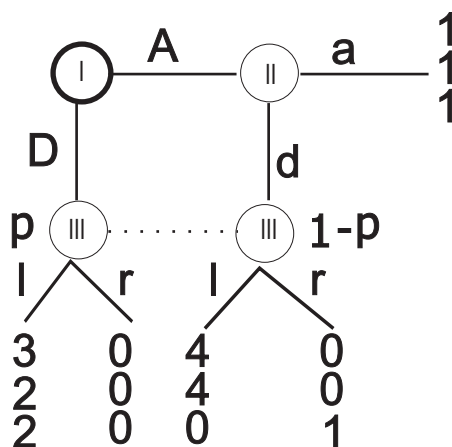


Figure 7: An example of a game of incomplete information in extensive form. For clarity the root of the game tree has been emphasized.

defined over the entire game tree. Assessments can be restricted to particular information sets and we define $(\sigma, \mu)_{I'}$ to be the restriction of (σ, μ) to information set I' . For example, in the game of Figure 7, $(A, a, r; p)$ is an assessment, whenever $p \in [0, 1]$. Respectively, $(r; p)$ is a restriction of this assessment to the information set of player III. An assessment (σ, μ) is said to be *sequentially rational*, if, starting from any reachable information set I' , the strategy σ maximizes the expected utility of each player in the rest of the game. In the game of Figure 7, sequential rationality rules out the unwanted equilibrium, $(D, a, l; p)$. Namely, if player III were to choose l , player II should choose d and player I A . Thus, $(D, a, l; p)$ is not sequentially rational and (A, a, r) is the only sensible equilibrium of the game.

Consider the game of Figure 8. In the game, player I has two types. The probability for being located at vertex x is $1/3$ and the probability for being at vertex x' is $2/3$. Furthermore, the probability that player I selects action U is one and the probability that she selects action D is zero. If, for some reason, player I deviates from the expected behaviour and plays D , which has probability zero, the question is what beliefs should player II assign over the vertices y and y' . The solution to this question is answered by the concept of *consistency* [KW82b, KR87], according to which the beliefs about vertices that belong to off the equilibrium path information sets (that are reached with zero probability) should be such that they are the limit of a sequence of small errors (or trembles). To this end, let ϵ^t be a sequence of probabilities that converges to zero as time t passes. Furthermore,

assume that the game of Figure 8 is perturbed so that player I chooses U with probability $1 - \epsilon^t$ and D with probability ϵ^t . Now, the Bayes' rule can be used to derive the beliefs and

$$\mu(y) = \lim_{t \rightarrow \infty} \mu(y^t) = \lim_{t \rightarrow \infty} \frac{\mu^t(x)\epsilon^t}{\mu^t(x)\epsilon^t + \mu^t(x')\epsilon^t} = \frac{1}{3}. \quad (23)$$

Respectively, the beliefs assigned to vertex $\mu(y')$ are $2/3$.

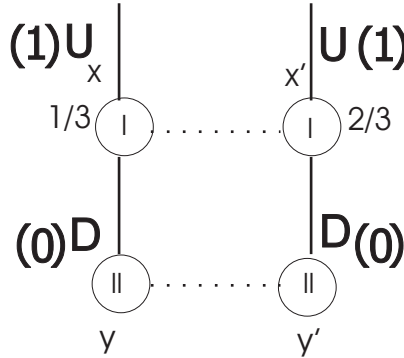


Figure 8: A game that illustrates the concept of consistency [KW82b].

An assessment (σ, μ) is said to be a *sequential equilibrium* [KW82b], if the strategies are sequentially rational given the beliefs, if Bayes' rule is used to derive the probabilities at all on the path information sets and, if beliefs at off the path information sets are limit beliefs of a sequence of perturbed games.

To give an example of a sequential equilibrium, we return to the game of Figure 7. For the strategy profile (A, a, r) the limit probability of reaching the vertex on the left hand side of player III's information set is zero. Because of this, player III knows that player II played a and that player I played A . Considers the case where player III chooses l with a probability greater than $1/4$. In this case player II would play d instead and the strategy profile would not be an equilibrium strategy. We use σ_3 to denote a probability distribution for player III over the actions l and r . When the probability of reaching the information set is zero, σ_3 is an equilibrium behaviour strategy whenever $\sigma_3(r) \geq 3/4$. As a consequence, the set of sequential equilibrium points for the game is $\{(A, a, \sigma'_3) | \sigma'_3(r) \geq 3/4\}$.

3.7 Mechanism Design

Mechanism design is a large research area and an important branch of microeconomics. As a consequence covering mechanism design thoroughly is out of scope for this thesis. Instead, we consider the basic concepts of mechanism design and present the Vickrey-Clarke-Groves (VCG)²²-mechanism [Vic61, Cla71, Gro73].

The problem setting that we consider is called the *incentive problem in a conglomerate* [Gro73]. A conglomerate is an organization model that consists of a central administration and a set of partially independent subunits that are linked together only through a central administration. In the conglomerate the functions of a subunit are independent of the other subunits and the actions performed by the subunits are driven by self-interest. The *utility* or *profit* of the subunit is determined by the actions the subunit performs and by the coordinating actions made by the central agency. The overall profit of the organization is determined by the sum of the profits of the subunits and the goal of the organization's head is to maximize the overall profit²³.

As mentioned above, the coordination actions performed by the central agency affect the outcomes of the subunits. However, the subunits often have private information that affects their decision making and, in order to act optimally, the central agency requires knowledge about the private information of the subunits. Because the subunits are striving for optimal performance, they do not necessarily report their private information truthfully to the central agency as this can reduce their profit. The goal of mechanism design in this kind of setting is to devise a payment scheme, also known as the *incentive structure*, so that the subunits report their private information truthfully. Accordingly the central agency should compensate for the loss of profit for the subunits, if they report their private information truthfully.

In order for the VCG-mechanism to be applicable to the conglomerate model,

²²In the literature other permutations of letters are used. We use the chronological order while some use the alphabetical (CGV) or reverse alphabetical (VGC) ordering. The actual mechanism was designed by Clarke and generalised by Groves. Hence Vickrey's name is sometimes omitted and the name Clarke-Groves (CG) -mechanism is used instead.

²³Also called a utilitarian mechanism [NR01].

it is required that the utilities of the subunits are *quasi-linear*. The general form of quasilinear functions is given in Equation 24, when q_i is the compensation paid to agent i and $v_i(a, \theta_i)$ is the valuation (utility) of agent i for action profile a (consisting of actions performed by all agents and the coordinating action) and type θ_i .

$$u_i = q_i + v_i(a, \theta_i). \quad (24)$$

The payment scheme of VCG-mechanism is based on the *Vickrey sealed bid auction model* [Vic61], in which agents are permitted a single bid for the item(s) of interest and they do not observe each others bids. In Vickrey's payment scheme, the highest bidder wins, but pays the second highest price. It can be proven that this kind of payment scheme maximizes the overall utility of the game and that bidding the amount corresponding to the true valuation is a (weakly) dominant strategy, i.e., it yields an equilibrium for the game the agents are playing. The first property, that the overall utility is maximized, is also known as *Pareto-optimality*.

Clarke [Cla71] and Groves [Gro73] generalized Vickrey's idea to the centrally coordinated model and thus also to the conglomerate model. The idea is to define the compensations²⁴ q_i so that they are independent of the report of agent i . Typically this is achieved by combining the reports of the other players and subtracting this from the overall utility. Thus if ϕ is the overall utility of the organization, the side-payment q_i is given by

$$q_i = \phi - \sum_{j \neq i} w_j, \quad (25)$$

where w_j is the reported utility of agent j . It can be proven that by using this kind of scheme, truth-telling is a weakly dominating strategy and thus an equilibrium for the "incentive" game the agents are playing in the model.

The idea of how to apply mechanism design into ad hoc networks is fairly intuitive. A node that wants to send packets to another node is the central agency and the other nodes of the network represent subunits. Each unit is assumed to be purely driven by self-interest and thus the nodes route packets only if they profit from the process. In the first phase, the sending node asks other nodes what their cost to route a packet is and based on the information provided by others, the

²⁴Also called tariffs, incentives, side-payments etc. depending on the problem that is modelled.

node selects the optimal route and routes the packet(s) along the route. Clearly, without an incentive structure this kind of a model leads to non-optimal behaviour. Thus, when the VCG mechanism is applied to ad hoc networks, the nodes that are on the optimal path are compensated, but the compensation is independent of their report. Applying this model to ad hoc networks is discussed more thoroughly in Section 4.2.

3.8 Adaptive learning in Bayesian games

In this section we consider a framework, where a Bayesian game in extensive form is repeated over time, and where the players learn the behaviour strategies and beliefs of the game over time. Each repetition is considered to be a single-shot game, i.e. when the players decide their actions, they are assumed to ignore possible future gain [GJS99, DFL04]. As before, the variable t is used to index the repetitions. At the beginning of each repetition, the players have a *theory* $(\hat{\sigma}_i, \hat{\mu}_i)^t$ about the probability distributions over the possible actions (behaviour strategy profiles of the game) and about the probability distributions over the vertices at each of their information set. After each instance of game the players refine their theories using the Bayes' rule. A sequence of repetitions and the corresponding theory refinements constitute a *learning process*, which is said to be *adaptive*, if the theories of the players eventually come close to the empirical frequencies of actions in past play and if the beliefs derived from past play converge [GJS99].

In contrast to earlier sections, players are not assumed to act optimally at each period of time. Instead it is assumed that, in an instance of the game, at each reached information set the players either take an action that is optimal or select a suboptimal action with a probability that decreases as a function of time. The suboptimal actions represent experiments (and errors) and they are necessary for the players to learn the probabilities overall in the game.

In game theory details about the learning process are usually left unspecific, but instead it is assumed that the learning process satisfies certain generic conditions. Firstly, it is required that, in a learning sequence of infinite length, all information sets are reached infinitely many times. For most games this assumption is trivially satisfied. Secondly, it is assumed that suboptimal play vanishes in the long

run, i.e. that the degree of experimentation (and errors) decreases over time.

In Section 3.6, the concept of sequential equilibrium was introduced as a suitable solution concept for Bayesian games in extensive form. According to [GJS99], if the learning process is adaptive and satisfies the set of constraints introduced above, the strategies and beliefs converge to a sequential equilibrium. Thus, if it can be proven that the game has at least one Nash equilibrium point, that the learning process is adaptive and that the constraints are satisfied, the setting admits a sequential equilibrium. Thus the model is optimal in the sense that most of the time the players act optimally given their beliefs and that the probability of erroneous play decreases as a function of time.

4 Cooperation stimulation in Ad Hoc Networks

As the first applications of ad hoc networking were designed for military (reconnaissance) and rescue operations, the network designers could assume that the nodes are willing to cooperate in the routing functionalities of the network, i.e. forward packets for other nodes. However, when ad hoc network technologies are applied to personal communication environments, the assumption is not valid anymore. Namely, the devices of the users are resource constrained and thus users might refuse to forward packets in order to save energy or to minimize communication costs. Moreover, if all nodes of the network act according to this rationale, no packets are forwarded for anyone and the network ceases to function. To cope with the situation, a mechanism for motivating the individual nodes to cooperate is needed. Such a mechanism is called a *cooperation (stimulation) mechanism* and in this section we present common mechanisms that are used in ad hoc networks. We begin in Section 4.1 by explaining more carefully why these mechanisms are needed. Next, in Section 4.2 we introduce so-called virtual currency systems, whereas Section 4.3 introduces reputation systems.

4.1 Selfishness and Models of Selfishness

The exact reason why non-cooperative behaviour emerges in applications of ad hoc networking in personal communication environments is that the devices are resource constrained, i.e. they have limited energy, connectivity, memory and computational capabilities, **and** that they are lacking a common goal. In military and rescue operations the devices are energy constrained, but because they have a common goal, they are willing to cooperate. On the other hand, if the nodes have no common goal, but they have unlimited resources, cooperation still emerges. Namely, in the case of unlimited resources, the only thing that matters to a node is that it gets its packets routed to the destination. But if it refuses to forward packets for other nodes, the other nodes might start also to refuse forwarding packets. Hence the nodes have no motivation for refusing to forward packets.

In personal communication environments nodes have no motivation to share their resources with other devices (as there is no common goal) and thus because the nodes want to minimize unnecessary resource consumption and to maximize

throughput of their own messages, they refuse to forward packets of other nodes and non-cooperative behaviour arises. This kind of behaviour is called *selfishness*. Another kind of non-cooperativeness emerges, when a node wants to harm the functionality of the network by, e.g. intentionally refusing to forward packets for a subset of all the nodes. As this behaviour is not driven by self-interest, but by the intention of harming others, the behaviour is called *maliciousness*. The main goal of the underlying cooperation stimulation mechanism should first and foremost to be discouraging selfishness, but if the mechanism can also handle maliciousness, it is naturally advantageous.

In order to determine whether a node is selfish (or malicious) or not, a node needs to observe the routing behaviour of other nodes. However, reasons such as interference, link failures and congestion cause failures in the network routing functions although the behaviour is not noncooperative (as no decisions are made whether to forward the packets or not). Thus there is some "noise" in the observations. The existing mechanisms attempt to cope with this kind of random events by using a noise threshold which tells how much non-cooperative behaviour is tolerated in a particular period. However, this can also cause another kind of selfishness. Namely, a node can attempt to learn the threshold and the period and optimize its behaviour so that it refuses to cooperate exactly the amount the threshold allows for. This kind of behaviour has not been studied in the literature, but if the cooperation stimulation mechanisms are ignorant of this kind of possibility, it might be possible to circumvent the behaviour of the underlying mechanism so that if majority of the networks nodes behave like this, the overall performance of the routing function degrades significantly.

4.2 Virtual Currency Systems

The first class of mechanisms that we discuss is that of virtual currency systems. These mechanisms model the routing situation as an economic market, where the source node needs to buy the routing services from intermediate nodes using virtual monetary units. The reason why virtual money is preferred to real money is that the wireless nature of communications does not offer enough security support for monetary transactions.

The NUGLET approach is the first mechanism based on this scheme that has been proposed in the literature [BH00, BH01, BBC⁺01, BH03]. In the approach the virtual monetary units are called NUGLETs and each device is assumed to be equipped with a tamper resistant hardware module that acts as an automatic cashier for the NUGLETs of a device. When a node sends packets to the network, the hardware module of the sender attaches a set of NUGLETs to the packet headers and when an intermediate node forwards the message, the hardware module of that node extracts the share of NUGLETs needed to forward the packets from the packet headers. Because the cashier resides in the tamper resistant module, the intermediate nodes can not extract too large a share. However, a problem with this approach is that the sender must determine how many NUGLETs to attach to the packets and to this problem the authors propose two different payment models. First of all, in the *packet purse model* the source estimates the required number of NUGLETs and attaches to the packet headers a suitable number of them based on the estimate. If the attached amount is too small, the packets are discarded before reaching the destination. On the other hand, excess NUGLETs are kept by the destination. Thus, if a node wants to get its packets delivered, it should overestimate the number of NUGLETs required.

The problems with the packet purse model are evident. Namely, the estimation problem is hard to solve efficiently, especially, when the network topology is changing constantly. Moreover, if the overestimate is too crude, the node quickly runs out of NUGLETs. To overcome these problems, the authors propose another payment model called the *packet trade model*, in which the packets do not contain any NUGLETs, but each hop buys the packets from the previous node. Thus in the model the source node pays nothing for the packets. Also this approach has serious drawbacks. Namely, the network is vulnerable to flooding attacks as sending packets is free of cost. Secondly, the forwarder does not necessarily want to buy the packets sent to it, e.g. due to maliciousness. In addition, the requirement of equipping each node with a tamper resistant hardware module is very questionable.

Another approach, *Sprite* [ZCY03], uses ideas from mechanism design (see Section 3.7) to overcome the requirement of having a separate tamper resistant module. In the Sprite approach nodes store receipts about messages they have helped to forward and, when a node needs more (virtual) money, it sends the receipts to

a bank-like entity, *credit clearance service (CCS)*, which determines the compensation a node receives. To prevent the nodes to send false receipts, the contents of a receipt are derived from the original message. Furthermore, the size of a receipt is assumed to be much smaller than the size of other packets so that the nodes have a motivation to report them.

The payment scheme of Sprite is such that the CCS determines the last node ever receiving a particular message (however, ending always if the destination is on the path) and asks the source to pay some amount, say β , to the last node and a larger amount, say α , to the intermediate nodes. The last node does not have to be the true destination, but as the last node is compensated less than the intermediate nodes, the nodes are motivated to forward messages for others. In addition, to ensure that coalitions do not emerge, the source is required to pay the CCS a small additional fee, say ϵ .

The authors prove the correctness of the Sprite-mechanism, but unfortunately the proof holds only in very specific scenarios. First of all, the intermediate nodes are compensated by a constant amount although their true cost depends on the distance of the next hop and the level of remaining energy. If the true personal cost is greater than the compensation, according to rationality arguments the nodes do not cooperate. Thus the actual behaviour with this scheme is very sensitive to the values of the compensation parameters α , β and ϵ . An additional flaw in the article is that the authors do not give any motivation why the source should pay anything in the first place. Namely, the model has no punishment scheme and, as the nodes are not equipped with additional hardware, there is no entity that could force the source to pay anything.

From theoretical point of view, the best approach in this category is the *ad hoc VCG* [AE03] which uses the infamous Vickrey-Clarke-Groves mechanism (see Section 3.7) to derive a decentralized system that requires no additional hardware support. The approach is designed to work on top of a source routing scheme (see Sections 2.2 and 2.3) so that in the route discovery phase each node is additionally asked its cost to forward a particular set of packets. When the discovery request reaches the destination node, it calculates the minimum cost path and sends the source an acknowledgement message along that path.

Clearly, in the basic setting the nodes have no motivation for reporting their true forwarding costs. However, reporting the true cost becomes the best choice, if nodes on the minimum cost path are paid additional compensations. The compensations need to be independent of the reporting of an individual node and the authors suggest paying node i as a compensation the difference between the minimum cost path and the minimum cost path that does not have node i on it. The authors prove that by using this payment scheme, rational nodes will cooperate. Namely, a rational node wants to be on the minimum cost path if it receives additional compensation for its costs and, on the other hand, a node does not underestimate the cost because that would yield negative personal utility for it.

The main problem with the ad hoc VCG is more practical related. Namely, the authors admit that their scheme requires significant overhead in the discovery phase, which makes the model less attractive in practice. From theoretical point of view, the weakness is that the VCG mechanism works only, if the personal utility of an agent can be represented using a linear function. However, as we have no evidence about the true form of the personal utility function of a single node, it might be the case that in reality the scheme does not work.

4.3 Reputation Mechanisms

The second class of mechanisms that we consider is that of reputation mechanisms. The goal of these mechanisms is to facilitate trust between interacting agents by providing summaries about the past behaviour of other agents [RZFK00]. Typically these summaries are represented as numerical values that attempt to characterize some aspects of interest about the observed behaviour. For example, the values could represent the proportion of messages that a node has forwarded from all the messages sent through that node.

In practical implementations of reputation mechanisms the core component is the *reputation table* which is a data structure that maintains reputation values for other agents (nodes). Other important elements are the *calculation* and *update functions* which determine how the reputations are calculated in the first place and how the values should be updated when new information is obtained. In

ad hoc networks the implementations must be globally distributed and thus each node maintains locally a reputation rating for other nodes. The main source of information for these ratings are the node's own observations about the behaviour of others. However, in order to obtain this kind of information, nodes must have support for monitoring the behaviour of others. One possible way to realise this support is to use, e.g., the Watchdog (see Section 2.4) [MGLB00].

Another possible source of information are observations made by other nodes. However, a problem with third-party information is that the reliability of the information is often questionable. For this reason, reputation mechanisms usually put more weight to own observations. However, the way in which different information sources are combined usually varies from one approach to another. Other differences arise, e.g., from the way the reputation ratings are calculated.

With the virtual currency systems, the routing path was always selected by the underlying protocol. However, with reputation mechanisms the situation is slightly different as first the protocol discovers a set of paths after which a score value is calculated for them. The path with the highest score is then used for routing the packets. An example of a suitable scoring scheme is the overall reputation of the nodes on the routing path divided by the length of the path.

The first practical implementation of reputation mechanisms that we describe is the *CONFIDANT* [BB02, BLB02, BB03, BLB03, BLB04]. The *CONFIDANT* system consists of four main components: the *monitor*, the *trust manager*, the *reputation system* and the *path manager*. The monitoring component is essentially a watchdog-like device that monitors all nodes in the neighbourhood of a device and makes measurements about the performance of the underlying routing protocol. If misbehaviour is detected, the trust manager sends an *ALARM* message to the network. When another node receives the *ALARM* message, the trust manager of that node decides whether the contents of the message are reliable and whether any actions need to be performed. In addition to these tasks, the trust manager is responsible for maintaining auxiliary data structures such as the trust ratings of other nodes and a list of received *ALARM* messages.

The reputation system of *CONFIDANT* is based on a so-called commitment model

[Ake70, KW82a, MR82], which, when applied to ad hoc networks, assumes that node j forwards messages with a rate equal to the value of some parameter θ_j ($\in [0, 1]$). Furthermore, in the CONFIDANT system it is also assumed that another parameter γ_j determines the rate with which node j sends false ALARM messages to the network. Thus the goal of each node i is to estimate the parameters θ_j and γ_j over time. At all instances of time the most recent estimates are used to select the optimal routing path and to determine the trustworthiness of other nodes. In order to perform the actual parameter estimation, two assumptions are needed. First of all, it is assumed that the observations about the behaviour of another node are binary, i.e. at a particular instance of time a node either cooperates or not. Secondly, it is assumed that the observations are statistically independent and identically distributed (see Appendix 2). Under these assumptions the parameters θ_j and γ_j can be seen as parameters of distinct binomial distributions and the estimation can be performed using the Bayesian framework (see Appendix 2).

The Bayesian way to estimate the (bias) parameter θ of a binomial distribution is to use two counters. One of the counters measures the number of successful outcomes and it is denoted by the variable α . Respectively, the other counter measures non-successful outcomes and it is by with the variable β . It can be shown (e.g. [Nea04]) that the best parameter estimate, denoted by $\hat{\theta}$, for θ is given by²⁵

$$\hat{\theta} = \frac{\alpha}{\alpha + \beta}. \quad (26)$$

When new information is obtained, the estimate is updated by increasing the counters α and β and then, if necessary, using Equation 26 to calculate an updated estimate.

When the framework is applied to ad hoc networks, each node i maintains locally two counters $\alpha_{i,j}$ and $\beta_{i,j}$ for each node j . An outcome in this context is determined as whether node j cooperated or not. Thus $\alpha_{i,j}$ measures the number of times node j has cooperated and $\beta_{i,j}$ the number of times it has refused to forward packets for node i .

Instead of using the standard Bayesian approach, CONFIDANT uses a modi-

²⁵This is equal to the expectation of the posterior distribution of θ

fied scheme for updating the counters $\alpha_{i,j}$ and $\beta_{i,j}$. In the modified scheme the updates are calculated as a weighted sum of past observations [BB03]. Before presenting the update equations of the modified scheme we define two auxiliary variables. First of all, let $s_{i,j}$ be a binary indicator that indicates whether node j cooperated the last time node i sent something through it or not. Secondly, let δ be a real number for which the relation $0 < \delta < 1$ holds. Using the auxiliary variables, the update equations of CONFIDANT can be written as

$$\begin{aligned}\alpha_{i,j}(\theta) &= \delta\alpha_{i,j}(\theta) + s_{i,j} \\ \beta_{i,j}(\theta) &= \delta\beta_{i,j}(\theta) + (1 - s_{i,j}),\end{aligned}\tag{27}$$

where $\alpha_{i,j}(\theta)$ and $\beta_{i,j}(\theta)$ denote node i 's counters for the estimate of node j 's parameter θ (either θ_j or γ_j). Using recursive substitution it can be shown that these Equations correspond to *exponential smoothing* of time series (see e.g. [Cha03]).

In CONFIDANT reputation and trust ratings are represented as pairs of the form $(\alpha_{i,j}; \beta_{i,j})$. The primary source of information for these ratings are the reputation estimates θ_j , which count for node's own observations. The secondary source of information are the observations of others and incorporating this information into the model is done using a two phase process. In the first phase node i calculates the expectation of the estimate of trust parameter γ_j . If this value exceeds a predefined threshold, the new information can be directly incorporated into the model. However, if node j is deemed untrustworthy, the trust manager checks the reliability of all information from node j using a *deviation test*. The deviation test simply checks whether the ratings reported by node j differ significantly from the own current estimates. If not, the information can be incorporated into the model using a weighted sum of the form

$$r_{i,j} = r_{i,j} + \mu r_{k,j}.\tag{28}$$

Here $r_{i,j}$ is node i 's reputation rating for node j , k is some other node and μ is a small positive constant.

The CONFIDANT system has many positive aspects. Namely, in contrast to many other approaches, the authors attempt to build their mechanism on top of a proper theoretical framework. However, unfortunately the modification of the Bayesian parameter estimation scheme makes the resulting mechanism very

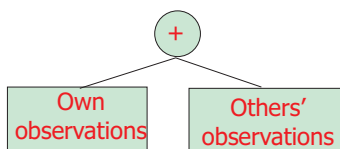


Figure 9: The reputation sources in CONFIDANT.

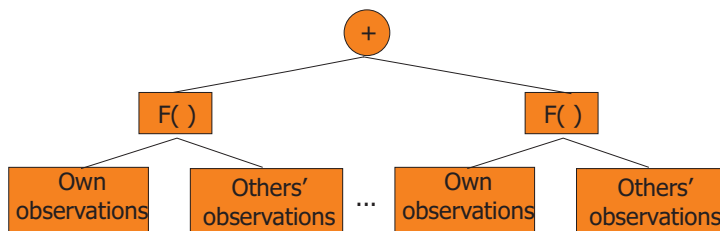


Figure 10: The reputation sources in CORE.

questionable. Furthermore, the way in which the different information sources are combined is very ad hoc by nature. Finally, the system has numerous free parameters whose values are set based on subjective assumptions and heuristics. As a result the end performance of the system might vary a lot depending on the parameter values. However, the sensitivity of the system to the parameter values has not been empirically evaluated.

The second and last reputation mechanism that we introduce is CORE [MM02a, MM02b, MM04]. The CORE approach is very similar to the CONFIDANT and thus it is discussed in less detail. The main difference between the approaches is that, whereas CONFIDANT uses plain observations as the information sources, CORE uses observations made with respect to different network functionalities. For example, nodes can observe, e.g., packet routing and route discovery behaviour of another node and estimate another node's willingness to cooperate based on the various sources of information. Similarly, other nodes can propagate to the network observations made with respect to different functionalities. The reputation model of CORE is illustrated in Figure 10 and, for comparison, the reputation model of CONFIDANT is illustrated in Figure 9.

The observations made with respect to a particular functionality $F(\cdot)$ are called the *functional reputation* with respect to function $F(\cdot)$. Furthermore, these observations are divided into own observations and information gathered by others. The various information sources are then integrated using a two-fold weighting scheme which first weights own observations more than observations of others. Secondly, the functionalities are assigned different weights before the final reputation value is calculated.

The main weaknesses of CORE are that it is lacking a theoretical justification and that the number of free parameters becomes quickly large (two weights per function). However, overall the idea of using different functionalities as the information sources is an interesting idea.

5 Theoretical Modelling of Cooperation Stimulation

In this section we combine the concepts of the previous sections by using game theory for building theoretical models of routing in ad hoc networks. Section 5.1, overviews and analyses critically existing theoretical models, whereas Section 5.2 presents an example routing scenario that serves as an introduction to our novel model, introduced in Section 5.3.

5.1 Previous Game Theoretic Models

Although the idea to model routing in ad hoc networks with game theory is relatively new, other types of communication networks have been analyzed with tools from game theory. Especially, game theoretic analysis of Internet congestion control has been an active research area (see e.g. [She95, Pap01, GKK02, Rou02]). However, due to node mobility and resource constraints, ad hoc networks have more inherent sources for uncertainty and, as a consequence, models designed specifically for ad hoc networks are needed. In this section we introduce earlier work that has applied either evolutionary (Section 3.5) game theory or static Bayesian games, i.e. Bayesian games (Section 3.6) with a non-dynamic structure.

Chronologically the first approach is by Srinivasan, Nuggehalli, Chiasserini and Rao who model the routing situation using the repeated prisoners' dilemma (see Section 3.4) [SNCR03]. Thus when a node is making a routing decision, it has two possible modes of behaviour: to cooperate or to defect. According to evolutionary game theory, an effective strategy in this kind of setting is the so-called *TIT-FOR-TAT* strategy [Sel78, Axe84], in which the player cooperates on the first move and thereafter imitates the other player [Wei05]. The authors argue that this kind of behaviour is too severe for ad hoc networks and propose a modification, called the *generous TIT-FOR-TAT* (GTFT), in which the behaviour of the nodes is stochastic so that node i forwards packets for node j with a probability μ_i^j . The probabilities are estimated over time using

$$\hat{\mu}_i^j = \frac{C_i^j}{D_i^j}, \quad (29)$$

where C_i^j is the number of packets node j has forwarded for node i and D_i^j is the total number of messages node i has sent to node j to be forwarded. The authors

prove that with GTFT the total throughput rate of the network is *Pareto-optimal* [PR04], i.e. all resources of the network are utilized optimally.

Although the model achieves theoretical optimality it is not realistic as the authors make non-justifiable assumptions about how nodes communicate. For example, the authors assume that all nodes always transmit either a positive or a negative acknowledgement to the source node, that network topology remains fixed during the transmission of a single packet and that transmitting a single packet costs a constant amount of energy. The restrictive nature of these assumptions makes it hard to generalize the model to more realistic environments. Furthermore, the model does not take into account the dynamic structure of routing decisions or the uncertainty about the resources of other devices.

The next approach that we consider is by Altman, Kherani, Michiardi and Molva [AKMM04a, AKMM04b]. Also in their model each node j is assumed to forward packets with some probability μ_j . The probabilities are assumed to be independent of the source. When a packet is sent to the network, each node computes the current equilibrium strategy and uses the probability indicated by the equilibrium strategy for forwarding. However, as has been discussed earlier, without a punishment or a reward mechanism, the equilibrium strategy is $\mu_j = 0$ for each node j . To solve this, the authors propose a punishment mechanism where other nodes decrease their forwarding probabilities, if someone deviates from the equilibrium strategy. The authors prove that, under this framework, the equilibrium strategy of the nodes increases as the number of intermediate nodes increases. Furthermore, the authors give an algorithm for calculating the equilibrium strategies (in this setting) in a distributed manner.

The main problem in this approach is that the routing model is not generic. First of all, it is unlikely that the forwarding probabilities of the nodes are independent of the source – at least in the long run. Secondly, the complexity of calculating the equilibrium strategy is not feasible in terms of storage and computational effort. Finally, also this approach ignores the dynamic structure of routing decisions and the uncertainty about the resources of other devices.

Closest to our work is the work of Urpi, Bonucelli and Giordano [UBG03] who

use static Bayesian games and infinitely repeated games to model the routing situation. In their model, the nodes are divided into classes so that nodes in the same class generate traffic according to the same stochastic process. Once assigned, the energy class of a node stays fixed for the entire lifetime of the node²⁶. A node can only have information about nodes that belong to its neighbourhood Γ_i and nodes that belong to Γ_i are assumed to have beliefs about the class of node i and vice versa. In the model, time is assumed to be discrete and during each time step a Bayesian game is played. The actions available to the players depend on the amount of energy the players have available and as a consequence the action space shrinks over time. In each instance of the game a node has to decide how many of its own packets to send to the network and how many packets to forward for a neighbouring node. Because the overall game is a repeated game, the utility of a player depends on the discount factor, which is used to weight future utilities. In the model each node has a discount factor γ_j for each node j in Γ_i , which is interpreted as the probability that node j belongs to the neighbourhood of node i in the next time step.

The model has the same problems as the others we have discussed. That is, routing decisions are static and the uncertainty about resources of others is ignored. Furthermore, the model utilises only local information and thus no guarantees can be given that packets are actually routed to the destination. The model has also a game theoretic flaw. Namely, the players' information increases over time and thus the beliefs of the players should be updated. However, in the proposed model the beliefs stay the same.

Although the earlier models have flaws, they are better suited for ad hoc networks than the models used for fixed networks. Due to the various sources of uncertainty (e.g. mobility, resources etc.) in ad hoc networks, especially the work of Urpi et al. ([UBG03]) seems promising. Nevertheless, the existing models still are overly simplistic. Especially the resource constraints of the devices are something that should be, at least in our opinion, taken into account. Furthermore, routing decisions do not take place immediately after packets have been sent to

²⁶In the traditional approach to Bayesian games, Nature draws the type of each player from a fixed probability distribution before each repetition / stage game and thus the types are not fixed. However, the model can be seen to belong to a more general class of games, stochastic Bayesian games, where the probability distribution is allowed to change over time [DFL04].

Model	+	-
Srinivasan et al.	<ul style="list-style-type: none"> - Based on a valid mathematical theory. - Theoretically optimal. 	<ul style="list-style-type: none"> - Makes non justifiable assumptions about how nodes communicate. - Ignores dynamic structure of routing decisions. - Ignores uncertainty about others' resources.
Altman et al.	<ul style="list-style-type: none"> - Less severe than other models \Rightarrow more believable as a real model. 	<ul style="list-style-type: none"> - Forwarding decisions are assumed to be independent of the source. - Requires calculating the equilibrium strategy. - Ignores dynamic structure of routing decisions. - Ignores uncertainty about others' resources.
Urpi et al.	<ul style="list-style-type: none"> - Takes uncertainty into account 	<ul style="list-style-type: none"> - Ignores dynamic structure of routing decisions. - The uncertainty model is not realistic. - Ignores updates of beliefs over time.
Our model	<ul style="list-style-type: none"> - Takes uncertainty into account - Beliefs updated over time. - Dynamic structure of routing decisions. 	<ul style="list-style-type: none"> - Forwarders' beliefs ignored - Dynamics of energy classes ignored - Energy estimation based on heuristics

Table 2: Comparison of the models presented in Section 5.1.

the network and during the delay periods the resources of the devices can change so that it affects the optimal strategies. A summary of the good and bad aspects

of the different models is presented in Table 2.

5.2 Example Routing Scenario

Before discussing the novel routing model, we present an example scenario that describes the routing situation in more detail. To this end, we assume that an arbitrary node i wants to send some packets to another node j . We assume that i knows already the path to j . If this is not the case, a source routing protocol such as the DSR protocol (see Section 2.3) can be used to discover the path.

At some point of time, node i decides to send either all packets or a subset of them to the network. The decision of how many packets to send to the network depends on node i 's beliefs about the packets reaching the destination node j and on the available resources of node i .

When the first intermediate node receives the packets, it stores them in a buffer and processes them when it has resources to spare. Typically the total latency, which consists of the delay between receiving and processing and of the signal propagation delay, is very short. However, in some cases, e.g. due to heavy network congestion or to large distances, the latency period is long enough to affect the forwarding decision of the intermediate node. Namely, the routing situation can be seen as an iterated prisoners' dilemma (see Sections 3.2 and 3.4) and, according to Axelrod [Axe84], the willingness of a player to cooperate depends on past experiences and on the probability that the players will meet again in the future. In ad hoc networks the probability to meet again in the future depends on the intermediate node's energy level²⁷ and thus during the latency period the energy level of the intermediate node can decrease so that it is not beneficial for the node to cooperate anymore.

It is unlikely that the source node could know exactly the energy level of the intermediate node as the intermediate nodes would not have any incentives to report it truthfully to the source node. However, based on device capabilities, history information and own observations, it is possible to derive a crude estimate of the

²⁷Note: we are talking about the probability of meeting at all in the future, not about the probability of meeting at the next stage.

remaining energy. Thus, node i can have beliefs about the energy levels of the intermediate nodes and it can use these beliefs to guide its forwarding and sending decisions.

Assuming that the first intermediate node has forwarded some packets from node i , the packets arrive at the second intermediate node, which again decides whether to forward the received packets or not. Eventually the packets arrive at the destination or they are discarded at an intermediate node. This ends the lifetime of a single set of packets and in our model a stage game is defined so that it starts when a source node generates packets to be sent to the network and ends when the packets either arrive at the destination node or are discarded. We assume that the underlying protocol uses an acknowledgement scheme that allows the source node to know, when the packets arrive at the destination. To cope with the possibility that packets are discarded, it is furthermore assumed that the protocol uses a packet timer mechanism for sent packets, i.e. if the packets are not acknowledged within a certain period, they are considered to be discarded.

To conclude the example, the phases of the routing process are summarised in the list below.

1. Source node i generates a set of packets and performs the route discovery phase.
2. The source node estimates the probability that the packets arrive to the destination and decides how many packets (if any) to send to the network. If only some of the packets are sent to the network, the remaining packets are postponed to the next time step, i.e. we interpret the remaining packets as if source node had generated packets.
3. An intermediate node receives the packets and stores them in a buffer. After some time, the intermediate node processes the packets and decides, how many packets to send to the next node based on its own energy level and past experiences with the source node i .
4. While some packets are remaining and the current node is not the destination node, Step 3 is repeated.
5. The destination node receives the packets (or a subset of thereof) and sends

an acknowledgement message to the source node i .

6. The source node receives the acknowledgements or packet timers go off. Based on this information node i updates its beliefs. The mechanism that sends the acknowledgements must be designed with care to ensure that the source node has the relevant information available to it.

It should be noted here that the acknowledgement messages are typically much smaller than standard packets. Furthermore, if the source does not receive the acknowledgement, it decides that the packets were discarded. Thus, if some intermediate node decides to forward packets, its personal utility becomes negative if it does not forward the acknowledgement later on. Hence we can see this (small) cost as being included in the decision making process of the intermediate node.

5.3 A Theoretical Model for Routing

In this section, we introduce a mathematical model for energy constrained routing in selfish ad hoc networks. The model uses adaptive learning in dynamic games, i.e. the players can have information about the actions of others before choosing their action, with incomplete information (see Sections 3.6 and 3.8). In the model, each node defines a unique game where that particular node acts as the source of all traffic. However, the same node participates in other games, where the source of traffic is some other node, as a forwarder. We consider a single such game and fix node i to be the source of all traffic.

Each time node i generates packets to the network, a new stage begins in the game she is playing with the other nodes in the network. For ease of presentation, we assume that the destination of packets is the same in a single stage. In the model the stage games are Bayesian games and at the beginning of each stage, the nodes hold a theory about the parameters of interest [DFL04]. Whenever new information becomes available, latest at the end of the stage, the nodes update their theories using the Bayes' rule. Before going into details about the model we define some notation.

Notation

Let \mathcal{G} be an ad hoc network and $G \subset \mathcal{G}$ a subset of nodes so that there is at least one route from every node $i \in G$ to every other node $j \in N$. Thus, in graph theoretic terms, G is a *connected component* of the network graph \mathcal{N} . Each node i is assumed to have a discrete representation for time so that a new time step for node i begins when the node generates new traffic to the network. The time steps of node i are denoted t_k^i . As we are considering only a single game, we drop the indices i from all variables. According to the definition of a time step, node i generates some traffic at the beginning of time step t_k . We define a *packet* to be the smallest unit of communication and $g(t_k)$ to be the number of packets node i generates at time step t_k . By definition, the variable $g(t_k)$ is an integer value that is greater than zero, i.e. $g(t_k) \in \mathbb{Z}_+$. The number of packets node i actually sends to the network is denoted $s(t_k)$. The number of generated packets clearly serves as an upper bound for the number of packets sent and thus the relation $0 \leq s(t_k) \leq g(t_k)$ holds.

Each node in the network is assumed to have a discrete representation for energy. We call the discretized energy the *energy class* of a node and use the variable θ_i to denote it. The discretization is assumed to be global so that the set of possible energy class values is the same for all nodes in the network. The energy class of a node is only known by the node itself and thus the energy class corresponds to the concept of type in Bayesian games (see Section 3.6). However, nodes are assumed to have beliefs about the energy classes of the other nodes. Node i 's beliefs about the energy class of a forwarding node j are denoted $\mu_{t_k}^j$. We assume that the energy class of the source does not affect the decisions of the forwarders and thus the forwarding nodes do not have beliefs²⁸. The main motivation for this assumption is that the overall model remains practical. However, the assumption can be also justified so that, as nodes can replenish their energy reserves, estimating the probabilities, with which the nodes meet again in the future, is infeasible for the nodes. Furthermore, a reasonable assumption is that throughput of own packets is more important than energy, and, as a consequence, the nodes should overestimate the probability as otherwise the throughput rate of the network would decrease.

²⁸Note that we are considering only a single game here. All nodes have beliefs about the energy classes of other nodes, but these beliefs are only used when the node acts as the source of traffic.

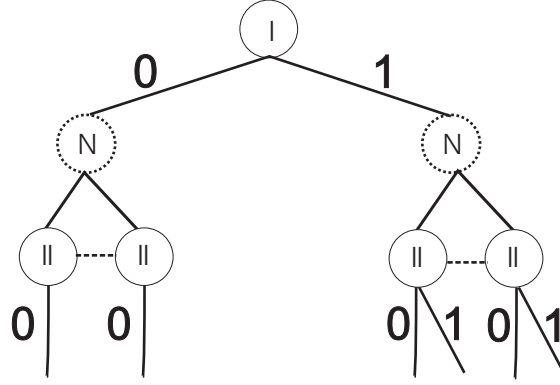


Figure 11: An example game tree, which has a single intermediate node that has two possible types, and where the source (player I) sends either one or zero packets to the network.

Stage games

We assume that the path discovery phase of the routing has been performed and that $J = \{j | j \in N \setminus \{i\}\}$ is the collection of intermediate nodes j through which the packets can be routed to the destination. Under the mentioned assumptions, the (active) set of players N in the t_k :th stage game is $N = \{i\} \cup J$. Respectively, the type space of the stage game is $\Theta = \{\Theta_i\} \times_{j \in J} \{\Theta_j\}$, where Θ_j is the set of possible energy class values θ_j and Θ_i is a singleton set (the current discretized energy of node i). As we assumed that the discretization is global, the type space Θ_j is the same for all possible intermediate nodes j . The belief system of the stage game is $\mathcal{B} = \{\mu_{t_k}^j\}$. The action space of source i is the set $A(t_k) = \{s(t_k) | s(t_k) \leq g(t_k)\}$, i.e. the possible amounts $(0, \dots, g(t_k))$ of packets node i can send to node j at time step t_k . Let $f_j(t_k)$ denote the number of packets node j forwards for node i . The action space of a forwarding node j (in the game where node i is the source of all traffic) is the set $A_j(t_k) = \{f_j(t_k) | 0 \leq f_j(t_k) \leq g(t_k)\}$. Note that at the beginning of the stage the intermediate nodes do not know how many packets node i sends to the network and thus the action space of node j consists of all alternatives between zero and $g(t_k)$. However, the actions of node i clearly restrict the actions of node j within the game tree. As an example of this kind of situation, Figure 11 contains a game tree with a single intermediate node j that has two possible types and where node i has sent a single packet to the network.

From Figure 11, we can deduce that a forwarder has $\#A(t_k)$ information sets²⁹, where $\#$ is the cardinality (number of elements in a set) operator and the size of each information set is $s(t_k)$, i.e. the number of packets on the route to that particular information set. The probabilities for the different vertices of the forwarder are determined by the objective distribution (see Section 3.6) and thus are known to the forwarder. The information sets of the source are singleton sets and thus the probabilities assigned over the vertices of the source are trivial (one everywhere).

In order to finalise the definition of a stage game, according to Definition 2, we still need to define the utility functions of the nodes. The utility functions are discussed later in more detail and, at this point, we only assume that the utility functions of the players are defined in the field \mathbb{Z}^2 , i.e., for all k , $u_k : \mathbb{Z}^2 \rightarrow \mathbb{R}$. The variables u_i and u_j are used to denote the utility functions of the source and a forwarder. Using these definitions, we define the t_k :th stage game of node i formally as follows:

Definition 3 *The t_k :th stage game of source i is the 8-tuple $\langle N, \mathcal{T}, u, \mathcal{A}, \mathcal{I}, p, \Theta, \mathcal{B} \rangle$, where N is the set $\{i\} \cup J$, u is the vector function $u = (u_i, u_{j_1}, \dots, u_{j_n})$, \mathcal{A} is the collection $\{A(t_k)\} \times_{j \in J} \{A_j(t_k)\}$, Θ is the collection $\{\Theta_i\} \times_{j \in J} \{\Theta_j\}$, \mathcal{B} is the set $\{\mu_{t_k}^j\}$ and \mathcal{T} is the game tree, which is constructed in a similar fashion as in Figure 11. The information space \mathcal{I} is derived from the game tree in a trivial fashion and the probabilities for the vertices within information sets are given by the objective distribution p , i.e. the probability distribution governing the random selection of types.*

Behaviour of the intermediate nodes

In order to define optimality in the stage games, we must define the utility functions of the nodes more precisely. We start by considering the intermediate nodes and fix j to be an intermediate node and θ_j its current energy class. Let $\gamma(\theta_j)$ be a function that assigns a probability to energy class θ_j . We assume that the function $\gamma(\cdot)$ maintains ordering of the energy classes, i.e. if θ_{j1} represents less energy than θ_{j2} then $\gamma(\theta_{j1}) < \gamma(\theta_{j2})$.

²⁹As the intermediate nodes do not have beliefs, the only thing that matters to node j is its own energy class and the action of node i . Thus the game tree resembles that of Figure 11 everywhere along the routing path.

Next we assume that a cooperation mechanism that gives probabilistic estimates about the cooperation of another node is employed. Such mechanisms are presented, e.g., in [BB03, BLB04] and [Nur05, Nur06]. Let $\hat{\gamma}_{j,k}$ denote node j 's estimate about the willingness of node k to cooperate as given by the employed cooperation mechanism. The probability $\sigma_{j,i}$, with which node j forwards packets for node i , is assumed to be a combination of the energy dependent probabilities and the probabilities given by the cooperation mechanism, i.e.

$$\sigma_{j,i} = \alpha_j \hat{\gamma}_{j,i} + \beta_j \gamma(\theta_j), \quad (30)$$

where α_j and β_j are importance parameters that indicate how important energy and cooperation rate are for a particular node. We require that the weights sum up to one, i.e. $\alpha_j + \beta_j = 1$, in which case the Equation 30 can be seen as an instance of Bayesian model averaging [HMRV99, OF04]. Thus, given the action s_i of node i , on average the node j forwards $\sigma_{j,i} \cdot s_i$ packets, where the probability $\sigma_{j,i}$ changes as a function of available energy of node j and actions of node i .

Behaviour of the source node

As mentioned earlier in this section, the source node is assumed to have theories that guide its decisions of how many packets to send to the network. These theories are about the types of the other players and about their behaviour strategies. We define $\phi_j(t_k)$ to be the theory of the source node about the quantities of interest of a forwarding node j , i.e.

$$\phi_j(t_k) = \{\hat{\sigma}_{i,j}, \hat{\theta}_j\}, \quad (31)$$

where $\hat{\sigma}_{i,j}$ is node i 's theory about the overall forwarding probability $\sigma_{j,i}$ (behaviour strategy) of node i and $\hat{\theta}_j$ is node i 's theory about the energy class (type) of node j .

The actions of the source node should balance throughput of own messages and energy usage. Let \mathcal{P} be a collection of routing paths from the source node i to the destination. Similarly to the Pathrater in the Watchdog framework (see Section 2.4), we assume that the source node evaluates the different paths in \mathcal{P} so that each path is calculated a probability with which the packets are routed to the

destination and the path with the highest probability is used for routing. To guarantee theoretical optimality, we need to assume that a suboptimal path is selected with a small error probability.

The probabilities assigned to the different paths in \mathcal{P} are assumed to utilise the theories so that the assigned probability is a combination of the observed behaviour history of a node and the estimated energy class. Thus, the probability of a routing path $P \in \mathcal{P}$ is

$$\mu(P) = \prod_{j \in P} \left[\alpha_i \hat{\gamma}_{i,j} + \beta_i \gamma(\hat{\theta}_j) \right], \quad (32)$$

where α_i and β_i are importance parameters of node i (as with the intermediate nodes), $\hat{\gamma}_{i,j}$ and $\hat{\theta}_j$ are the current theory and $\gamma(\cdot)$ is a function that assigns a probability to an energy class. Thus the evaluation scheme of the source node is assumed to be similar as the decision mechanism of a forwarding node (see Equation 30).

After the source node has decided which path to use for the packets, it has still to decide how many packets it should send to the network. To this end, we define the expected utility of an action $s(t_k)$ and a path P as

$$u(s(t_k), P) = \sum_{s'(t_k) \leq s(t_k)} \mu(P) [h(s'(t_k)) - h_\gamma(s'(t_k))], \quad (33)$$

where $h(\cdot)$ is a function that tell how much the packets are worth to the source node, whereas $h_\gamma(\cdot)$ tells how much the energy needed to forward the packets is worth to the source node. In accordance with the adaptive learning framework (see Section 3.8 and Appendix 1), node i selects the action $\hat{s}(t_k)$ that maximizes Equation 33 with a probability $1 - \epsilon_t$, where ϵ_t is a sequence of small errors that decreases as a function of time, i.e. $\lim_{t \rightarrow \infty} \epsilon_t = 0$.

In order to do this optimization, we can use brute force and simply evaluate all the alternatives (additionally some pruning may be used) and select the best path. Thus the complexity of the brute force algorithm for Equation 32 is upper bounded by $\mathcal{O}(M|P|)$, where M is the length of the longest path $|P|$ is the number of paths. The overall complexity is then bounded by $\mathcal{O}(M|P| + g)$ as Equation 33 requires to go through all possible amounts of packets. Typically the number of paths is reduced already by the used routing protocol and this makes the

brute force approach a feasible alternative for most cases. On the other hand, the complexity of the forwarder depends on how the probability in Equation 30 is derived. For most reputation mechanisms the complexity is $\mathcal{O}(1)$ and this is also the case for the mechanism we use. However, naturally nothing prevents from using more complex mechanisms.

The memory requirements on the other hand depend on the used reputation scheme. Typically the memory requirements of a reputation scheme are $\mathcal{O}(c|N|)$, where c is some constant and N is the number of nodes. The exact memory requirements of the mechanism we use is $2|N| + 1$, which is clearly feasible. On the other hand, in the decision making phase we need information about the paths and hence the upper bound $\mathcal{O}(M|P|)$ holds for the memory requirements of the decision phase.

The overall game

From the definition of the overall game, we are still lacking the definition of how and when the updates are carried out. In the single hop case, it suffices that the source node is equipped with a watchdog-like mechanism (see Section 2.4) and with a packet timer mechanism. Namely, these assumption allow the source to observe the actions of the forwarders and thus the source can update its theories when it observes that node j has forwarded packets (positive update) or when a packet counter goes out (negative update). Thus we can define the *routing game* between a source node i and the network $N \setminus \{i\}$ to be a repeated game where each stage game is as in Definition 3 and where the theories are as discussed above.

When the routing path contains more intermediate nodes, additional technical solutions are needed for the gathering of feedback information. For our purposes it suffices to assume that some mechanism exists with which the actions of the individual nodes can be tracked, but for concreteness, in the following we present one such scheme.

The binary propagation scheme

At the end of a stage, the source needs to update its beliefs about all the intermediate nodes. The way this can be implemented is to consider each packet separately

in which case the actions are binary – either forward all or forward none. At this point we need to make some assumptions about the underlying protocol and require that the protocol implements packet numbering, timers and acknowledgements. Thus each packet can be uniquely identified by the combination of the source address and the packet number. Moreover, each node is assumed to maintain a timer for the packets it sends or forwards. The timer mechanism should work so that once the packet is sent, the node starts the timer. Once an acknowledgement message is received with the same packet number and source address, the timer is stopped. Because the nodes are assumed to be rational, the mechanism works if the size of the acknowledgement messages is much smaller than the size of packets. If a node has already forwarded a packet for the source node, it is reasonable to assume that, because it already has spent energy for forwarding, it wants to maintain a good status in the eyes of the source. To achieve this, the source needs to know that the node actually forwarded (some) packets, which is achieved via the acknowledgement scheme just discussed. A problem with this scheme is that nodes may be able to manipulate the packet header in which case the information is not reliable anymore. However, the encryption schemes that are needed to avoid this problem are out of scope for the Thesis.

The theoretical model that we introduced assumed that each new time step considers a single set of packets instead of a single packet. Thus we still need to modify the binary propagation scheme discussed above. The way to modify the scheme is to consider a continuous probability distribution defined over the interval $[0, 1]$ and where the value $x \in [0, 1]$ is the fraction of messages that were forwarded by some intermediate node. By adding a time frame number to each packet, we can separate between packets belonging to a particular time frame. Assuming that at time step t_k node i sent $s_i(t_k)$ packets, the unbiased estimator of the forwarding rate of the first relay node is $\sum_b f_j(b)/s_i(t_k)$, where each $f_j(b)$ is a binary indicator that tells whether packet b was forwarded by node j or not. Now the source node can update the probability distributions of the different relay nodes by considering the value of the estimator as the new evidence in the calculation of the posterior beliefs. The exact calculations depend on the used probability distribution. In our case we use a multinomial distribution and information on how the updates can be carried out can be found from Appendix 2.

Optimality

Theoretical optimality of the model follows easily from the results in Dekel et al. [DFL04], Kreps and Wilson [KW82b], Ratliff [Rat93], Fudenberg and Tirole [FT91] and Nurmi [Nur04]. First of all, from Dekel et al. [DFL04] it follows that if the stage games have a sequential equilibrium, this equilibrium is reached eventually (in the limit). Secondly, from Kreps and Wilson [KW82b] it follows that the game has an equilibrium and, because all vertices are reached with a strictly positive probability, the perfect Bayesian equilibrium points coincide with the sequential equilibrium points. That the game admits a perfect Bayesian equilibrium point, on the other hand, can be proven, according to Fudenberg and Tirole [FT91] and Ratliff [Rat93], as is proven by Nurmi [Nur04]. For completeness, this proof is also included in Appendix 1.

6 Experiments

Validating the theoretical model proposed in the previous section is impossible without access to real data. Unfortunately, such data is not publicly available and gathering the data would require large technological investments. Nevertheless, to give an idea of how nodes behave in the proposed model, we have conducted simulation studies on the model. Our simulations serve to show how nodes behave in the model and how their performance varies over time and under different conditions. As this is, to our best knowledge, the first attempt to model energy constraints in routing, there is no comparison to existing work. In this section we first describe the implementation of a simulator that was used to conduct the studies. Next we detail the tests that were run and present the results of the experiments.

6.1 Simulator overview

Although many simulators for ad hoc networks, e.g. SWANS [BHvR05], GloMoSim [BTA⁺99] and NS-2 [NS2], exist already, they are not well suited for testing the proposed model as the model requires modifications to the processing of packets. To this end, we implemented a simple simulator using J2SE 1.502 [JAV]. The implemented simulator lacks support for several aspects of ad hoc networks that are implemented in other simulators, such as signal fading models and obstacles in the environment, but these are not relevant for our purposes.

In the simulator, the nodes reside on a closed rectangular area. Thus, once a node reaches a border, it can not move beyond it. The locations within the rectangular area are defined by coordinates that are expressed using floating point values.

When the simulator is started, first a set of nodes is deployed to the area. The used deployment algorithm generates a random location and, if there are no nodes in the generated location, a node is deployed to that location. Otherwise a new location is generated and the process is repeated. Because floating point numbers are used, the locations hardly ever overlap. Although in the worst case this results in a deployment where the nodes are cluttered into small clusters that have no access to other clusters, most random number generators are able to generate

values that are approximately uniformly distributed and thus the resulting network topology is usually suitable for simulations.

In addition to a location, each node has a *traffic model* and a *movement model*. As the traffic model we use a *Poisson* distribution and as the movement model we use the so-called *random waypoint model* [JM96]. These models are the current standard choices for simulations and thus we felt them to be appropriate.

When the Poisson distribution is used as the traffic model, traffic is simply generated by drawing random samples from a Poisson distribution. The probability density function of a Poisson distribution is

$$\frac{\exp(-\lambda) \cdot \lambda^n}{n!}, \quad (34)$$

where n is an integer value and λ is the mean of the distribution. Sampling from a Poisson distribution is usually accomplished using *rejection sampling*, where first a random value u is drawn from a uniform distribution that is defined over the interval $[0, 1]$. Next an integer value n is generated and Equation 34 is used to calculate the probability of n . If the probability of n is greater than the value of u , the sampled value (n) is accepted and otherwise the process is repeated.

In the random waypoint model, on the other hand, a random destination is sampled from a uniform distribution that is defined over the possible coordinates. The sampled location is then used as the next destination of a node. Typically nodes are also assigned a speed, but this was not relevant for our purposes and thus speed is not supported by the simulator. When a node arrives at the destination, it either stays there for a while (typically a random delay) or a new destination is assigned for the node. Because of the movement model, the network topology changes dynamically over time which also reduces the effect of cluttering.

Once a node has generated packets that are to be sent to the network, the packets are assigned a destination. For simplicity, all packets generated within a single iteration (time step) of a node are assigned the same destination. The destination is selected randomly under the constraints that the destination is not allowed to be in the neighbourhood of the source and, naturally, that the destination is not the

same as the source. If no nodes that satisfy the constraints are found, the source does not send any packets to the network.

After the packets have been assigned a destination, the next step is to find all paths between the source and the destination. The discovery phase works in a breadth-first manner so that the simulator first generates all candidate paths that contain a single intermediate node, after which all candidate paths containing two intermediate nodes are generated etc. As the potential number of paths is huge and as we want to keep the (memory) footprint of the simulator reasonable, some pruning steps are required. First of all, a new candidate path is generated only if the next node does not belong to the current path yet. Thus, if the neighbours of node 8 are $\{9, 14, 16\}$, and the neighbours of node 9 are $\{10, 4, 28\}$, a candidate path of length four that contains node 9 would not be created. As the second pruning step, a new candidate path is generated only if the distance to the destination decreases. The distance is measured using the Euclidean distance from a node to another in the simulated area. Finally, the maximum length of a path is five. To give better insights into the path discovery we consider a simple example using the network topology in Figure 12.

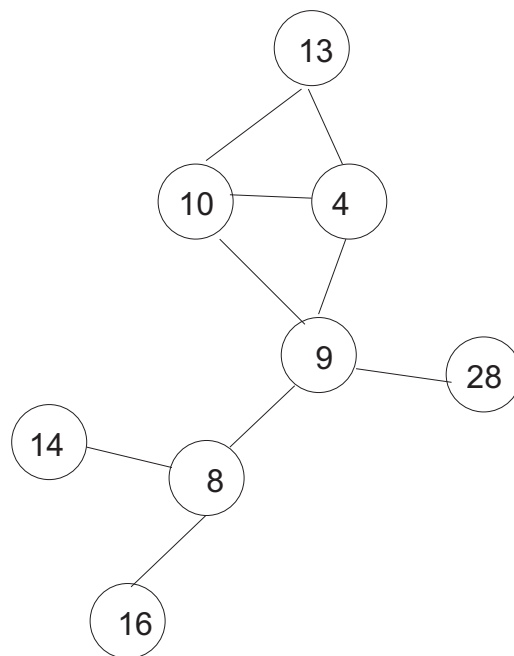


Figure 12: Network topology for the path discovery example.

We assume that node 8 is the source of traffic and that node 13 is the destination. First, candidate paths of length two are generated. Because nodes 16 and 14 are further from node 13 than node 8 they are pruned out and the only candidate path of length two is $8 \rightarrow 9$. Next, the candidate paths of length three are generated from the paths of length two. Because node 28 is further from the destination than node 9, only the neighbours 4 and 10 are considered. Thus, the candidate paths of length three are $8 \rightarrow 9 \rightarrow 4$ and $8 \rightarrow 9 \rightarrow 10$. The final set of paths is

$$\begin{aligned} &8 \rightarrow 9 \rightarrow 10 \rightarrow 13 \\ &8 \rightarrow 9 \rightarrow 4 \rightarrow 13 \\ &8 \rightarrow 9 \rightarrow 10 \rightarrow 4 \rightarrow 13. \end{aligned}$$

The reason for the last path is that node 4 is closer to node 13 than node 10. Thus the path $8 \rightarrow 9 \rightarrow 10 \rightarrow 4 \rightarrow 13$ satisfies the constraints, whereas the path $8 \rightarrow 9 \rightarrow 4 \rightarrow 10 \rightarrow 13$ is pruned out.

For practical reasons the simulator ignores time delay and energy consumption resulting from the route discovery phase. Furthermore, all found paths are simply given to the source nodes, whereas in reality the destination node needs to send them back to the source route via a suitable path. A potential solution for real ad hoc networks is that when the destination receives the first path, it starts a timer and when the timer goes off, it sends all paths, of which it has information, to the source node using, e.g., the first (and usually also shortest) path.

Once the source has information about the paths, the next step is to evaluate the paths and select the best one for routing. As we want to test the model proposed in this Thesis, we assume that the evaluation follows Equation 32, i.e. that each path is assigned a probability that is a product of the forwarding probabilities of the nodes along the path and where the forwarding probability of a node is a combination of the current cooperation and energy estimates. The weights α_i and β_i in Equation 32 are assigned when the node is created so that α_i is randomly assigned a value from the interval $[0.6, 0.8]$ and β_i is set to $1 - \alpha_i$. The path with the highest probability is then selected for routing.

After the best path has been found, the node needs to decide how many packets to send to the network. In the model, this is done using Equation 33. As the

packet "worth" function of Equation 33 we use $h(s'(t_k)) = 1000 * s'(t_k)$, i.e. the utility of each sent packet is considered to be 1000. As the energy function we use $h_\gamma(s'(t_k)) = \lambda d^2$, where d is the distance to the next node and λ is a predefined constant. The value of λ is set so that the maximum cost to send a packet is 10 units of energy. Thus, when d_i is the transmission range of node i , λ is set to $10/d_i^2$. In order to guarantee the experiment condition of the adaptive learning framework, i.e. that all information sets are reached infinitely many times and that suboptimal play vanishes in the long run, each node is assumed to select the optimal action with a probability $1 - \frac{1}{k}$. The proof that this sequence satisfies the necessary conditions is also part of Appendix 1.

After a node has sent packets to the network, it processes packets from other nodes. Each node has a buffer for incoming messages and each received message contains the source node's id and the path that is supposed to be used in the routing. If the current node is not the destination, the node needs to decide how many packets to forward for the source. This decision is assumed to be based on Equation 30 so that node j forwards packets for node i with a probability $\sigma_{j,i}$ that is a combination of node j 's current cooperation estimate for node i and the remaining energy of node j . The importance weights α_j and β_j of Equation 30 are assumed to have the same values as the weights that are used to decide how many packets to send to the network. Thus α_j is a value from the interval $[0.6, 0.8]$ and β_j is set to $1 - \alpha_j$. The resulting probability $\sigma_{j,i}$ is then used to decide how many packets node j forwards in a randomized fashion. Namely, for a given set of packets $s(t_k)$, node j generates $s(t_k)$ random values and forwards the number of packets for which the random value is smaller than the value of $\sigma_{j,i}$. Although this behaviour is not very realistic, it guarantees that the node forwards the "correct" number of packets on average. In reality it is more likely that for some nodes all packets are forwarded and for others only a subset of thereof.

If node j is the destination, it sends an acknowledgement to the source, which then updates its probability estimates. To introduce some uncertainty to the estimation phase, the source does not get full information about the packets, but instead the source is informed only about the id of the last node processing the packets and about the number of packets the last node received. Thus, if the last node processing the packets is the destination, the source is informed how many packets arrived at the destination. On the other hand, if the last hop is not the

destination, the source is informed about the number of packets the last node received and the id of the node that discarded the packets.

For the estimation of energy and cooperation rate ($\hat{\gamma}_{i,j}$ in Equation 33) we employ two multinomial distributions (see Appendix 2) that have a memory of ten iterations. Thus, when the estimates need to be calculated, only information from the last ten iterations is considered. Let $f(t_k)$ be the number of packets the last node received and $s(t_k)$ the original number of packets sent to the network. The cooperation rate of a node j is simply estimated by considering $f(t_k)$ to be the number of successful outcomes and $s(t_k) - f(t_k)$ the number of failures. If the last node is not the destination, the estimate of the last node is updated by considering all packets $s(t_k)$ as failures.

The energy estimation is more complicated and we resort to a (simple) heuristic for deriving the updates. First of all, let f be the fraction of packets the last node receives. Because the discretization of energy was assumed to be global, the fraction f can be mapped to match an energy class e . In our setting, we assume that energy classes are uniformly distributed, in which case the mapping can be done using a simple interpolation. Let E be the number of energy classes. The number of outcomes of class e is set to E , whereas the number of outcomes for another class e' is set to $E - d(e, e')$, where $d(e, e')$ is the distance between the indices of class e and e' . Thus, if there are ten energy classes ($E = 10$) and f is mapped to class with index 5, the vector (6, 7, 8, 9, 10, 9, 8, 7, 6, 5) is used for updating the energy estimates of the intermediate nodes and, if the last node is not the destination node, the vector (10, 9, 8, 7, 6, 5, 4, 3, 2, 1) is used to update the energy estimate of the last node. Due to the limited memory of the distributions this scheme is sufficient, but otherwise the estimation would need to consider also the dynamics governing the energy consumption, i.e. that certain energy classes become more probable as a function of time.

6.2 Experiment setting and results

In order to give insights about node performance in our setting, we performed two tests, both of which were repeated three times. In all test runs 30 nodes were deployed in a rectangular area of 100×100 (virtual) distance units. The transmis-

sion range of each node was set to 35 units and thus all nodes should be accessible with paths less than three hops. The nodes were initially set to have energy between 800 and 1000 (virtual) units. Because the maximum cost of forwarding a packet was set to 10 units of energy, each node was then able to send/forward at least 80 packets. The maximum number of packets a node could generate within a single time frame was set to ten and thus the maximum number of packets in a single iteration of the simulator was 300.

In the first experiment, we assumed that the energy reserves of the nodes are not replenished. Thus, when a node ran out of energy, it died and did not forward any packets nor did it generate new traffic to the network. In this kind of setting, the system performance should decrease as a function of time and, as can be seen from Figure 13, this also is the case.

In Figure 13, the three topmost plots represent the number of packets sent in the three test runs, whereas the lowest plot is an average of the test runs. As the Figure indicates, the packet throughput decreases rapidly and eventually the whole network dies out, which seems like a reasonable assumption in ad hoc networks. In reality, the devices often have more energy than in our simulations and, as a consequence, the resulting curve would be initially smoother and decrease more slowly. In addition, the rather strong fluctuations in the first ten iterations, which are the caused by the optimistic initialization scheme and the learning period of the employed cooperation scheme, would be less severe.

A more interesting question is how node performance varies as a function of iterations (time). To answer this question, the fraction of packets successfully arriving at the correct destination is illustrated in Figure 14.

As can be seen from the Figure, initially the performance increases rapidly, after which it decays slightly before stabilizing to an approximately constant level. This happens because all nodes start assuming that the energy of the other nodes is low and, as a consequence, the optimization step of the sources causes the nodes to send only very little (if any) traffic to the network.

Another interesting question is how much packets nodes forward for other nodes.

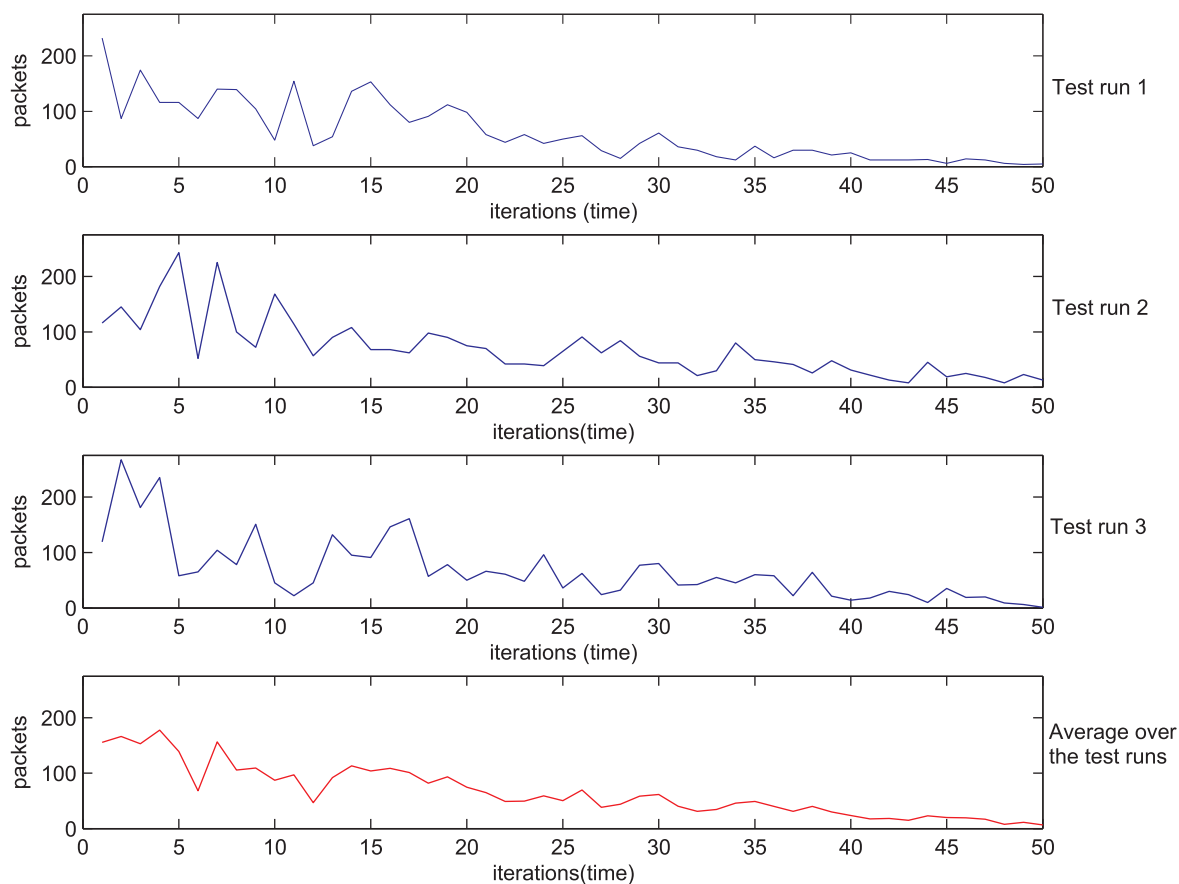


Figure 13: Overall number of packets sent as a function of iterations in the first experiment setting. In all plots, the three topmost represent the results in the three test runs, whereas the lowest plot is the average of the three test runs.

To answer this question, Figure 15 illustrates the fraction of packets the nodes forwarded for other nodes in the first experiment setting.

As can be seen from the Figure, initially the fraction of packets the nodes forward for other nodes increases rapidly. The rate with which the fraction grows initially depends on the initialization of the used estimation schemes. As we used a very optimistic initialization scheme, in our setting the nodes have initially a very optimistic view about the other nodes. However, after about ten iterations the memory of the used estimation schemes is full and the estimation schemes start to give more reliable estimates.

In the second experiment, we replenished the nodes so that, if a node ran out of energy, it was fully "recharged" (energy was set to full) after a single iteration.

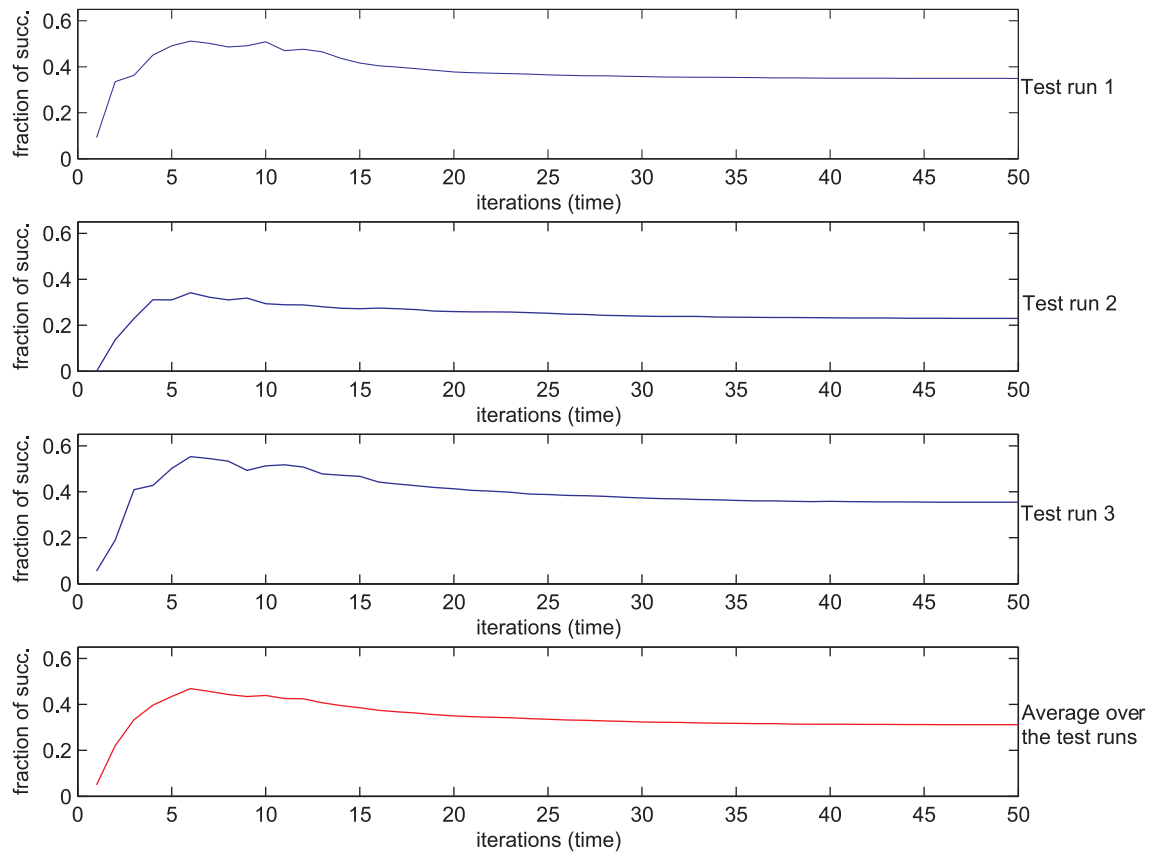


Figure 14: Fraction of packets that were actually sent and that successfully arrive at the destination node in the first experiment setting.

In this kind of setting, the throughput rate of the network should converge to an approximately constant rate and also the number of packets the nodes send to the network should be somewhat stable. However, the large variations in the energies of the nodes should cause rather heavy fluctuations in the throughput and the performance should be constant only on average. This effect is illustrated in Figure 16. In the first two test runs this effect is clearly visible, but in the third test run, a large number of nodes ran out of energy simultaneously, which caused a major decrease in the overall performance. Furthermore, because there were only few iterations remaining, the employed mechanisms did not have enough time to recover.

In the second experiment setting, the network performance (fraction of successfully sent packets) should stabilise steadily. Furthermore, the performance should be higher than in the first setting. As can be seen from Figure 17, this also is the

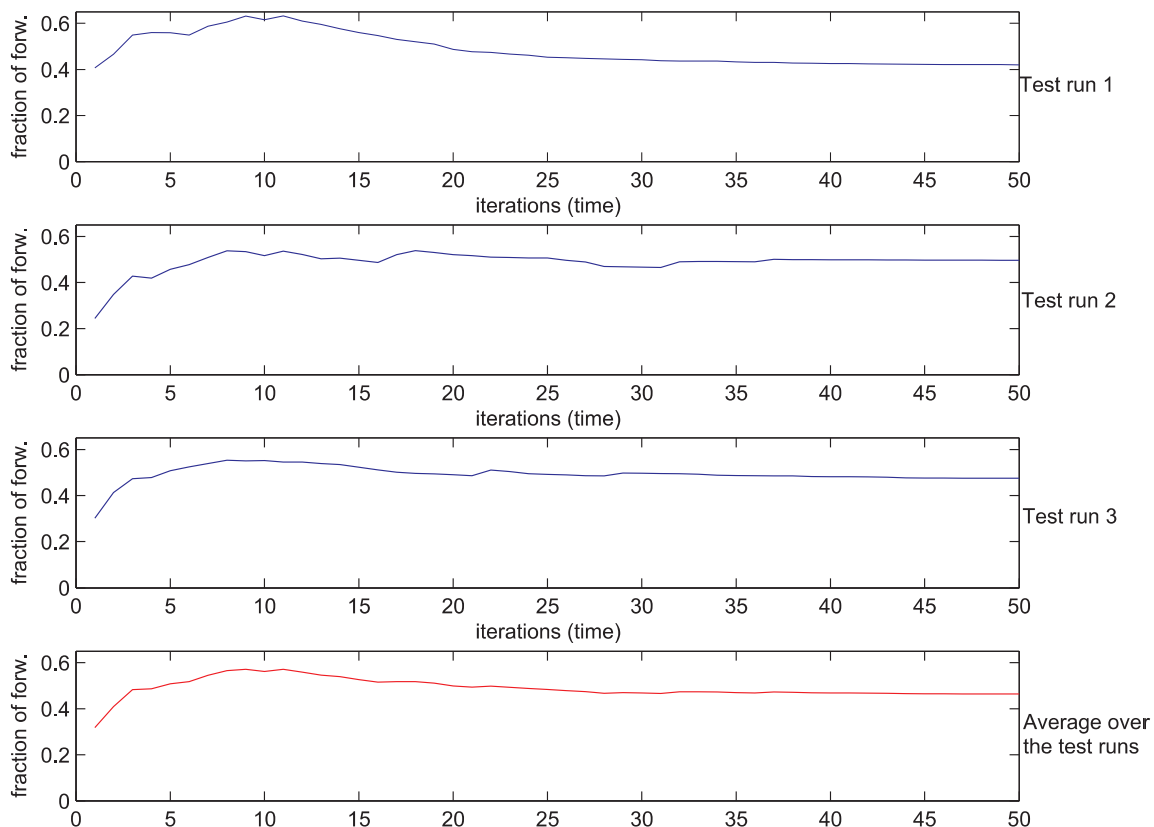


Figure 15: Fraction of packets forwarded in the first experiment setting for other nodes from all packets arriving at a node.

case as the fraction of successfully sent packets is around 0.5 in all test runs. We want to note that before the presented optimization and estimation schemes were employed, the performance of the nodes was below 0.1 and thus the increase in performance is significant.

Finally, Figure 18 illustrates the forwarding rate of nodes in the second experiment setting. As can be seen from the Figure, also the rate, with which nodes forward packets for other nodes, changes smoother than in the first experiment setting and is on average around 0.6. From a rationality perspective, a theoretically justifiable model is such that the forwarding rate of a node is approximately the same as the rate of successfully sent packets. Although our model does not achieve this exactly, it comes close and it is likely that the difference could be further reduced by optimizing the utility functions and estimation schemes. However, as our main intention was to illustrate how a system that is implemented according to the principles of the proposed model behaves, discussions about

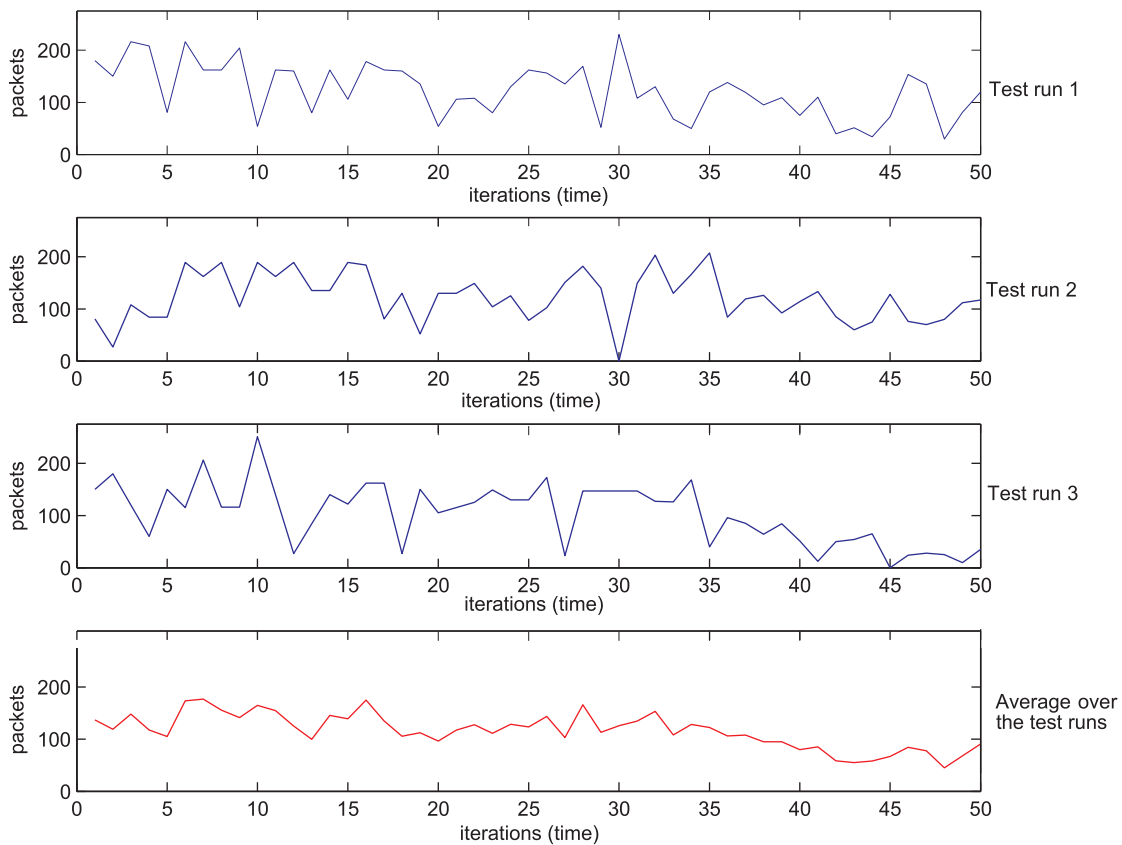


Figure 16: Overall number of packets sent as a function of iterations in the second experiment setting.

optimizing estimation schemes etc. are out of scope for the Thesis.

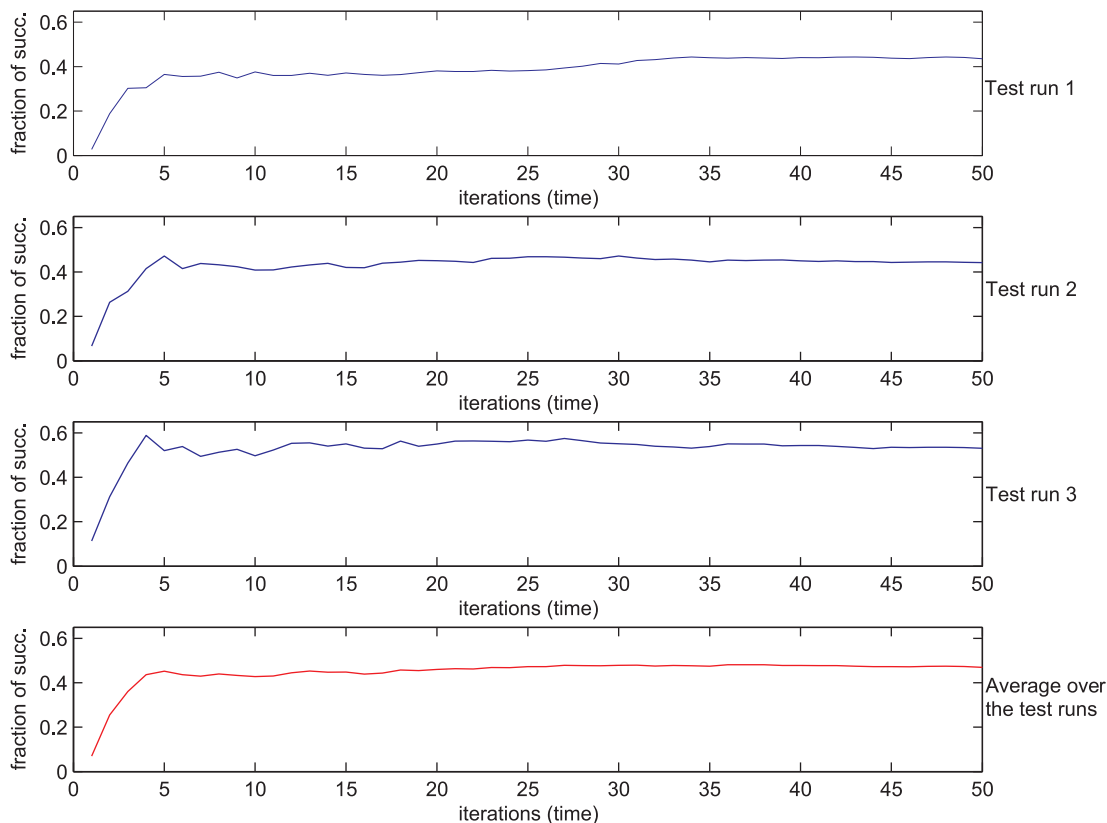


Figure 17: Fraction of packets that were actually sent and that successfully arrive at the destination node in the second experiment setting.

7 Self critique and summary

Self critique

As we have analyzed critically the approaches of others in this Thesis, it is important to subject also our proposed model under close scrutiny. In this section we shortly discuss some flaws in the theoretical model, in the simulations and in the experiments.

From game theoretic perspective, our model has one flaw as it assumes that the source's energy class does not affect the routing decisions of the forwarders. However, in practice the nodes can not reliably estimate the energy class of the source and thus the uncertainty in the estimates would be so high that it would not affect the routing decisions.

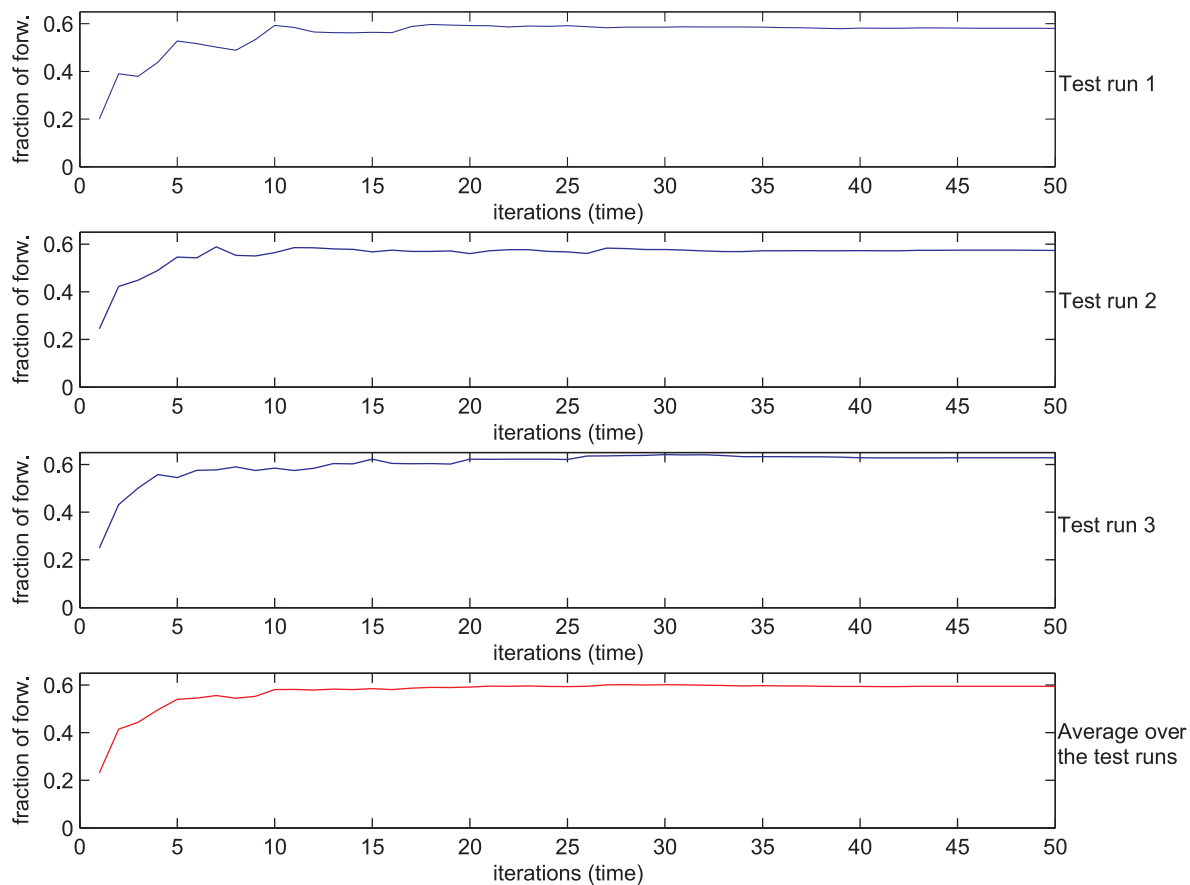


Figure 18: Fraction of packets forwarded in the second experiment setting for other nodes from all packets arriving at a node.

From a simulation perspective, a weakness in the model is that the number of free parameters is large. Namely, each node can have different weight parameters and different utility functions for energy and for throughput. The main problem with this is that the proper evaluation of the model in simulations is rather difficult because of the huge number of possible variations in the parameters. On the other hand, from practical perspective the estimation uses only two parameters which is feasible and thus if access to real life data is provided, the model can be properly evaluated.

A minor flaw in the model is that it assumes that the used discretization scheme is global. However, the assumption is made only for simplifying the optimality proof and in practice it does not matter whether the discretization is global or not. Another minor flaw is that we assume that the graph induced by the nodes is undirected. In reality the graph would be directed due to different energy reserves, transmission strengths and non-omni-directional antennas. This restriction however does not affect any of the game theoretic results and the only modification is that, in the binary propagation scheme, some intelligence is needed for routing the acknowledgement messages.

Also the performed simulations have many flaws. First of all, because the memory footprint of the simulator easily explodes, the overall number of nodes that could be deployed was somewhat small (50). Secondly, because the path discovery phase requires extensive amounts of memory, the network parameters, such as node transmission power and simulator area, needed to be set so that the length of paths remained short enough (max 5 hops). Furthermore, although the used movement and traffic models are the current standards in the research literature they are not necessarily the most realistic ones. However, discussion about whether this is the case or not is out of scope for the Thesis.

In addition to the above mentioned problems, the size of the memory footprint required us to keep the number of iterations rather small (max 50). This, on the other hand, forced us to set the energy of the nodes so that it decreases much more rapidly than in reality. Thus the experiments can be seen as worst case analysis of the model and typically the performance curves and fluctuations would be much smoother. Finally, the number of test runs in the different settings was rather

small (3) and thus the experiments can not be seen as fully conclusive. For this reason further experiments need to be performed. However, before this, we plan to implement some improvements on the used simulation tool.

Summary and Conclusions

Although many theoretical models for routing in ad hoc networks have been presented, they feel somewhat inadequate because they ignore several aspects such as the uncertainty about the resources of others and the dynamic structure of routing decisions. In this Thesis we have shown that it is possible to construct a theoretically valid model that takes also these aspects into account. Furthermore, we have shown that the resulting model is not too complex, but cooperation mechanisms and simulations can be performed on the model. The experiments we conducted showed the kind of performance one would expect from a realistic ad hoc network. However, because we had no access to real life data, the experiments are not fully conclusive and further work is still needed to verify the model and to develop suitable cooperation mechanisms and utility functions.

References

- Abr70 Abramson, N., The ALOHA system - another alternative for computer communications. *Proceedings of the AFIPS Fall Joint Conference*, volume 37, 1970, pages 281 – 285.
- Abr88 Abreu, D., On the theory of infinitely repeated games with discounting. *Econometrica*, 56,2(1988), pages 383 – 396.
- AE03 Anderegg, L. and Eidenbenz, S., Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom)*. ACM Press, 2003, pages 245 – 259.
- Ake70 Akerlof, G. A., The market for lemons: Quality, uncertainty and the market mechanism. *The Quarterly Journal of Economics*, 84,3(1970), pages 488 – 500.
- AKMM04a Altman, E., Kherani, A. A., Michiardi, P. and Molva, R., Non-cooperative forwarding in ad hoc networks. Technical Report RR-5116, INRIA, Sophia Antipolis, France, February 2004.
- AKMM04b Altman, E., Kherani, A. A., Michiardi, P. and Molva, R., Non-cooperative forwarding in ad-hoc networks. *Proceedings of the 15th IEEE International Symposium On Personal, Indoor and Mobile Radio Communications*, 2004.
- Axe84 Axelrod, R., *The Evolution of Cooperation*. Basic Books, 1984.
- BB02 Buchegger, S. and Boudec, J.-Y. L., Performance analysis of the CONFIDANT protocol: Cooperation of nodes – fairness in dynamic ad hoc networks. *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2002, pages 226 – 236.
- BB03 Buchegger, S. and Boudec, J.-Y. L., A robust reputation system for mobile ad hoc networks. Technical Report IC/2003/50, EPFL-IC, CH-1015 Lausanne, July 2003.
- BBC⁺01 Blazevic, L., Buttyán, L., Capkun, S., Giordano, S., Hubaux, J.-P. and Boudec, J.-Y. L., Self-organization in mobile ad hoc networks: The

- approach of terminodes. *Communications Magazine*, 39,6(2001), pages 166 – 174.
- BH00 Buttyán, L. and Hubaux, J.-P., Enforcing service availability in mobile ad-hoc wans. Technical Report DSC/2000/025, EPFL-DSC-ICA, CH-1015 Lausanne, Switzerland, May 2000.
- BH01 Buttyán, L. and Hubaux, J.-P., Nuglets: a virtual currency to stimulate cooperation in self-organized mobile ad hoc networks. Technical Report 200146, EPFL-IC, CH-1015 Lausanne, January 2001.
- BH03 Buttyán, L. and Hubaux, J.-P., Stimulating cooperation in self-organizing mobile ad hoc networks. *Mobile Networks and Applications*, 8,5(2003), pages 579 – 592.
- BHvR05 Barr, R., Haas, Z. J. and van Renesse, R. *Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, chapter Scalable Wireless Ad Hoc Network Simulation, pages 297 – 311. CRC Press, 2005.
- BLB02 Buchegger, S. and Le Boudec, J. Y., Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks. *Proceedings of Tenth Euromicro PDP (Parallel, Distributed and Network-based Processing)*, Gran Canaria, January 2002, pages 403 – 410.
- BLB03 Buchegger, S. and Le Boudec, J. Y., The effect of rumor spreading in reputation systems for mobile ad-hoc networks. *Proceedings of WiOpt '03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Sophia-Antipolis, France, March 2003.
- BLB04 Buchegger, S. and Le Boudec, J. Y., A robust reputation system for peer-to-peer and mobile ad-hoc networks. *Proceedings of P2PEcon 2004*, Harvard University, Cambridge MA, U.S.A., June 2004.
- BS97 Bendor, J. and Swistak, P., The evolutionary stability of cooperation. *The American Political Science Review*, 91,2(1997), pages 290–307.
- BTA+99 Bajaj, L., Takai, M., Ahuja, R., Tang, K., Bagrodia, R. and Gerla, M., Glomosim: A scalable network simulation environment. 990027, University of California, Los Angeles (UCLA), May 1999.

- Cha03 Chatfield, C., *The Analysis of Time Series: An Introduction*. Chapman & Hall, 6th edition, 2003.
- Cla71 Clarke, E. H., Multipart pricing of public goods. *Public Choice*, 8, pages 19 – 33.
- CSRL01 Cormen, T. H., Stein, C., Rivest, R. L. and Leiserson, C. E., *Introduction to Algorithms*. McGraw-Hill Higher Education, second edition, 2001.
- DFL04 Dekel, E., Fudenberg, D. and Levine, D. K., Learning to play bayesian games. *Games and Economic Behaviour*, 46, pages 282 – 303.
- Dij59 Dijkstra, E. W., A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, pages 269–271.
- Fis30 Fisher, R., *The Genetical Theory of Natural Selection*. Oxford: Clarendon, 1930.
- FJL00 Frodigh, M., Johansson, P. and Larsson, P., Wireless ad hoc networking - the art of networking without a network. *Ericsson Review*, 4.
- FT91 Fudenberg, D. and Tirole, J., *Game Theory*. MIT Press, Cambridge, Massachusetts, 1991.
- GCSR04 Gelman, A., Carlin, J., Stern, H. and Rubin, D., *Bayesian Data Analysis*. Chapman & Hall / CRC, 2004.
- Gio02 Giordano, S., Mobile ad hoc networks. In *In [Sto02]*, 2002, chapter 15, pages 325 – 346.
- GJS99 Groes, E., Jacobsen, H. J. and Sloth, B., Adaptive learning in extensive form games and sequential equilibrium. *Economic Theory*, 13, pages 125 – 142.
- GKK02 Garg, R., Kamra, A. and Khurana, V., A game-theoretic approach towards congestion control in communication networks. *ACM SIGCOMM Comput. Commun. Rev.*, 32,3(2002), pages 47–61.
- Gro73 Groves, T., Incentive in teams. *Econometrica*, 41,4(1973), pages 617–631.

- Har67 Harsanyi, J. C., Games with incomplete information played by bayesian players i-iii. part i: The basic model. *Management Science*, 14,3(1967), pages 159–182.
- Har68a Harsanyi, J. C., Games with incomplete information played by bayesian players i-iii. part ii: Bayesian equilibrium points. *Management Science*, 14,5(1968), pages 320 – 334.
- Har68b Harsanyi, J. C., Games with incomplete information played by bayesian players i-iii. part iii: The basic probability distribution of the game. *Management Science*, 14,7(1968), pages 486 – 502.
- HBGV01 Hubaux, J.-P., Boudec, J.-Y. L., Gross, T. and Vetterli, M., Towards self-organizing mobile ad-hoc networks: the terminodes project. *IEEE Communications Magazine*, 39,1(2001), pages 118 –124.
- HMRV99 Hoeting, J. A., Madigan, D., Raftery, A. E. and Volinsky, C. T., Bayesian model averaging: A tutorial. *Statistical Science*, 14,4(1999), pages 382 – 417.
- JAV Java standard edition 1.5, <http://java.sun.com/j2se/1.5.0/download.jsp>.
- JM96 Johnson, D. B. and Maltz, D. A., Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, Imielinski, T. and Korth, H., editors, Kluwer Academic Publishers, 1996, chapter 5, pages 153–181.
- JMB01 Johnson, D. B., Maltz, D. A. and Broch, J., DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks. In *[Per01]*, 2001, chapter 5, pages 139 – 172.
- JMC⁺01 Jacquet, P., Mühlethaler, P., Clausen, T., Laouiti, A., Qayyum, A. and Viennot, L., Optimized link state routing protocol for ad hoc networks. *Proceedings of the International Multitopic Conference (INCMIC)*. IEEE, 2001, pages 62–68.
- JMH04 Johnson, D. B., Maltz, D. A. and Hu, Y.-C., The dynamic source routing protocol, IETF-MANET Draft, July 2004.

- Joh94 Johnson, D. B., Routing in ad hoc networks of mobile hosts. *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*. IEEE, December 1994, pages 158–163.
- KR87 Kreps, D. M. and Ramey, G., Structural consistency, consistency, and sequential rationality. *Econometrica*, 55,6(1987), pages 1331 – 1348.
- KR02 Kurose, J. F. and Ross, K. W., *Computer Networking A Top-Down Approach Featuring the Internet*. Pearson Education, second edition, 2002.
- Kuh53 Kuhn, H. W., Extensive games and the problem of information. In *[Kuh97]*, 1953, pages 46 – 68.
- Kuh97 Kuhn, H. W., editor, *Classics in game theory*. Princeton University Press, Princeton, New Jersey, 1997.
- Kuh04 Kuhn, H. W., Foreword. In *[vNM04]*, 2004.
- KW82a Kreps, D. and Wilson, R., Reputation and imperfect information. *Journal of Economic Theory*, 27, pages 253 – 279.
- KW82b Kreps, D. M. and Wilson, R., Sequential equilibria. *Econometrica*, 50,4(1982), pages 863 – 894.
- MGLA96 Murthy, S. and Garcia-Luna-Aceves, J. J., An efficient routing protocol for wireless networks. *Mobile Networks and Applications*, 1,2(1996), pages 183–197.
- MGLB00 Marti, S., Giuli, T. J., Lai, K. and Baker, M., Mitigating routing misbehavior in mobile ad hoc networks. *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, August 2000, pages 255 – 265.
- MM02a Michiardi, P. and Molva, R., Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security*. Kluwer, 2002, pages 107 – 121.
- MM02b Michiardi, P. and Molva, R., Game theoretic analysis of security in mobile ad hoc networks. Technical Report RR-02-070, Eurécom, Sophia-Antipolis, April 2002.

- MM02c Michiardi, P. and Molva, R., Simulation-based analysis of security exposures in mobile ad hoc networks. *European Wireless Conference*, 2002.
- MM04 Michiardi, P. and Molva, R., Analysis of coalition formation and co-operation strategies in mobile ad hoc networks. Technical Report RR-04-099, INRIA, Sophia Antipolis, France, February 2004.
- MR82 Milgrom, P. and Roberts, J., Predation, reputation and entry deterrence. *Journal of Economic Theory*, 27, pages 280 – 312.
- Nas50 Nash, J. F., Equilibrium points in n-person games. In [Kuh97], 1950, pages 3 – 4.
- Nea04 Neapolitan, R., *Learning Bayesian Networks*. Prentice Hall, Upper Saddle River, NJ, 2004.
- NR01 Nisan, N. and Ronen, A., Algorithmic mechanism design. *Games and Economic Behaviour*, 35, pages 166 – 196.
- NS2 The network simulator - NS-2, <http://www.isi.edu/nsnam/ns/>.
- Nur04 Nurmi, P., Modelling routing in ad hoc networks with dynamic Bayesian games. *Proceedings of the 1st Annual Conference on Sensor and Communication Networks IEEE SECON*, October 2004.
- Nur05 Nurmi, P., Bayesian game theory in practice: A framework for on-line reputation systems. C 2005-03-10, University of Helsinki, Department of Computer Science, March 2005.
- Nur06 Nurmi, P., A bayesian framework for online reputation systems. *Proceedings of the International Conference on Internet and Web Services (ICIW)*. IEEE Computer Society, 2006. to appear.
- OF04 O’Hagan, A. and Forster, J., *Bayesian Inference*, volume 2B of *Kendall’s Advanced Theory of Statistics*. Arnold Publishing, second edition, 2004.
- Pap01 Papadimitriou, C., Algorithms, games, and the internet. *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. ACM Press, 2001, pages 749–753.

- PB94 Perkins, C. E. and Bhagwat, P., Highly dynamic destination-sequenced distance-vector routing (dsv) for mobile computers. *Proceedings of the Conference on Communication Architectures, Protocols and Applications*. ACM, 1994, pages 234 – 244.
- PB01 Perkins, C. E. and Bhagwat, P., Dsv routing over a multihop wireless network of mobile computers. In [Per01], 2001, chapter 3, pages 53 – 74.
- Per01 Perkins, C. E., editor, *Ad Hoc Networking*. Addison-Wesley, New York, 2001.
- PR01 Perkins, C. E. and Royer, E. M., The ad hoc on-demand distance-vector protocol. In [Per01], 2001, chapter 6, pages 173 – 219.
- PR04 Pindyck, R. S. and Rubinfeld, D. L., *Microeconomics*. Prentice Hall, 6th edition, 2004.
- Raj02 Rajaraman, R., Topology control and routing in ad hoc networks: a survey. *SIGACT News*, 33,2(2002), pages 60–73.
- Rat93 Ratliff, J., Perfect bayesian equilibrium in extensive-form games (lecture notes), <http://www.virtualperfection.com/gametheory/6.3.PerfectBayesExtensiveForm.1.0.pdf>, 1993. 04.04.2005.
- Rou02 Roughgarden, T., *Selfish routing*. Ph.D. thesis, Cornell University, May 2002.
- RZFK00 Resnick, P., Zeckhauser, R., Friedman, E. and Kuwabara, K., Reputation systems: Facilitating trust in internet interactions. *Communications of the ACM*, 43,12(2000), pages 45 – 48.
- Sel65 Selten, R., Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit. *Zeitschrift für die gesamte Staatswissenschaft*, 121, pages 310–324 and 667 – 689.
- Sel75 Selten, R., Reexamination of the perfectness concept for equilibrium points in extensive games. In [Kuh97], 1975, pages 317–354.
- Sel78 Selten, R., The chain-store paradox. *Theory and Decision*, 9, pages 127 – 159.

- SG85 Sinha, R. and Gupta, S., Mobile packet radio networks: State-of-the-art. *IEEE Communications Magazine*, 23,3(1985), pages 53 – 61.
- She95 Shenker, S., Making greed work in networks: a game-theoretic analysis of switch service disciplines. *ACM/IEEE Transactions on Networking*, 3,6(1995), pages 819–831.
- Smi82 Smith, J. M., *Evolution and the Theory of Games*. Cambridge University Press, 1982.
- SNCR03 Srinivasan, V., Nuggehalli, P., Chiasserini, C. F. and Rao, R. R., Cooperation in wireless ad hoc networks. *Proceedings of the 22nd INFOCOM*, volume 2. IEEE, March 2003, pages 808 – 817.
- Spr03 Sprott, J. C., *Chaos and Time-Series Analysis*. Oxford University Press, Oxford, UK, 2003.
- Sto02 Stojmenović, I., editor, *Handbook of Wireless Networks and Mobile Computing*. John Wiley & Sons, New York, 2002.
- TLW02 Tseng, Y.-C., Liao, W.-H. and Wu, S.-L., Mobile ad hoc networks and routing protocols. In [Sto02], 2002, chapter 17, pages 371 – 392.
- UBG03 Urpi, A., Bonuccelli, M. and Giordano, S., Modelling cooperation in mobile ad hoc networks: a formal description of selfishness. *Proceedings of Modelling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, March 2003.
- Vic61 Vickrey, W., Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16,1(1961), pages 8 – 37.
- vNM04 von Neumann, J. and Morgenstern, O., *Theory of Games and Economic Behaviour, 60th anniversary edition*. Princeton University Press, Princeton, NJ, 2004.
- VR03 Vega-Redondo, F., *Economics and the Theory of Games*. Cambridge University Press, 2003.
- Wei91 Weiser, M., Computer of the 21st century. *Scientific American*, 265,3(1991), pages 94 – 104.
- Wei97 Weibull, J. W., *Evolutionary Game Theory*. MIT Press, 1997.

- Wei05 Weisstein, E. W., Tit-for-tat, From MathWorld–A Wolfram Web Resource. <http://mathworld.wolfram.com/Tit-for-Tat.html>, April 2005.
- ZCY03 Zhong, S., Chen, J. and Yang, Y. R., Sprite: A simple, cheat-proof credit-based system for mobile ad-hoc networks. *Proceedings of the 22nd INFOCOM*, volume 3. IEEE, March 2003, pages 1987 – 1997.

1 Proof of a perfect Bayesian equilibrium in the routing game

In this section we, first of all, prove that the theoretical model we introduced in Section 5, has at least one perfect equilibrium point. Namely, a result by Kreps and Wilson states that perfect equilibrium points are sequential equilibrium points. Secondly, we prove that the mechanism we used in Section 6 satisfies the conditions set on an adaptive learning process and thus the actions suggested by the mechanism converge to a sequential equilibrium. We begin by stating some equilibrium results and convergence conditions that are relevant for the proof.

Theorem 1 *All extensive games admit at least on sequential equilibrium.*

Proof of Theorem 1 [KW82b, p.15]. ■

Theorem 2 *Every perfect equilibrium point is a sequential equilibrium*

Proof of Theorem 2 [KW82b, p.21]. ■

In order to use the adaptive learning framework, we must specify the conditions that a learning process needs to satisfy. These are given in Condition 1 - 4.

Condition 1 *Every information set is reached infinitely many times. More formally, let h be some information set and $\mathbf{a}^t(h)$ the set of information sets that are reached at repetition t . We require that for each information set h_0 the set $\{n \in N | h_0 \in \mathbf{a}^n(h)\}$ is infinite.*

Condition 2 *Suboptimal play vanishes in the long run. More precisely, let h_0 be an information set which is reached infinitely many times. The empirical frequency of reaching a suboptimal information set h_1 from information set h_0 approaches zero as the number of iterations grows, i.e.*

$$\frac{\#\{n \in N | h_0, h_1 \in \mathbf{a}^t(h)\}}{\#\{n \in N | h_0 \in \mathbf{a}^t(h)\}} \rightarrow 0, \text{ when } n \rightarrow \infty \quad (35)$$

Condition 3 *If the empirical frequencies converge to some point, the theory of each player j must converge to that point, i.e. if the frequencies of the actions over the information sets σ converge to σ^* and if the frequencies of the probabilities defined of the information sets μ converge to μ^* , the theory (σ^t, μ^t) converges to (σ^*, μ^*) .*

Condition 4 *The frequencies of nature's moves converge to the objective distribution p .*

Groes et al. prove the following Theorem, which is important for our proof:

Theorem 3 *A learning process satisfying conditions 1 - 4 converges to a sequential equilibrium.*

Proof of Theorem 3 See [GJS99, p.12]. ■

A perfect Bayesian equilibrium must satisfy the subgame perfection criterion and in addition the model must satisfy four Bayesian postulates, which are given below.

- B1** *For each information set, the players must have beliefs about the node the game play has reached.*
- B2** *Whenever it is a player's turn to move, her actions must be optimal from that point onwards given her beliefs.*
- B3** *The player's beliefs about reachable (on-the-path) nodes, must be determined using the Bayes' rule.*
- B4** *The player's beliefs about unreachable (off-the-path) nodes must be determined using the Bayes' rule, whenever possible (whenever probabilities are positive).*

Lemma 1 *The model satisfies the Bayesian postulates **B1-B4**:*

Proof of Lemma 1 *The condition **B1** is trivially satisfied as all information sets are singleton sets and we can assign probability one to each node. The condition **B2** follows directly from the definition of the game. We assumed that the support of the action space of an arbitrary node equals the common action space which implies that there are no off-the-path nodes and thus condition **B4** is satisfied. The last condition, **B3**, follows directly from the way the belief system is constructed. ■*

Lemma 2 *The learning sequence $\epsilon_k = \frac{1}{k}$, satisfies Condition 1, i.e. all information sets are visited infinitely many times.*

Proof of Lemma 2 With the learning sequence, a suboptimal action is selected with probability $\frac{1}{k}$. We define a random variable $\chi_I(k)$ such that $\chi_I(k) = 1$, if information set I is visited on the k^{th} iteration and $\chi_I(k) = 0$ otherwise. The sum over the iterations for which $\chi_I(k) = 1$ form a harmonic series, and, as a consequence, the sequence $\sum \epsilon_k \chi_I(k)$ diverges as it has a divergent subsequence ■.

Lemma 3 The model satisfies Condition 2, i.e. suboptimal play vanishes in the long run.

Proof of Lemma 3 As $\lim \epsilon_k = 0$, suboptimal play clearly vanishes in the long run ■.

Lemma 4 The proposed model satisfies Conditions 3 and 4, i.e. if the empirical frequencies converge at some point, the theory of each player converges to that point and the frequencies of nature's moves converge to the objective distribution.

Proof of Lemma 4 These are immediate consequences of using the Bayes rule as the Bayesian updates ensure asymptotic convergence to the true distribution. ■.

Theorem 4 The proposed model has at least one sequential equilibrium to which the learning sequence converges.

Proof of Theorem 4 Existence of a sequential equilibrium follows from Theorems 1 and 2 and the fact that a sequential equilibrium is reached follows from Lemmas 2 - 4 and Theorem 3 ■.

2 Bayesian probability theory

Bayesian probability theory is a statistical theory of making statements about uncertain events θ . Initially events of interest are assigned a prior belief $p(\theta)$ which reflects existing knowledge about the event and the problem area. Later, as new information \mathcal{D} becomes available, the subjective beliefs are updated using the Bayes' rule

$$\text{posterior} = p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} = \frac{\text{likelihood} \times \text{prior}}{\text{normalizer}}. \quad (36)$$

The likelihood term $p(\mathcal{D}|\theta)$ measures the probability of seeing particular realizations of the event θ whereas the normalizer $p(\mathcal{D})$ is used to ensure that the values of $p(\theta|\mathcal{D})$ sum up to one and thus define a proper probability distribution. After the update, the values of the posterior $p(\theta|\mathcal{D})$ are used as the new priors $p(\theta)$.

Consider the case where the likelihood $p(\mathcal{D}|\theta)$ is given by a binomial distribution, e.g. the observations are a sequence of coin throws. If we assign the prior probabilities using a *Beta*-distribution, it can be proven that the resulting posterior beliefs are also Beta-distributed (see e.g. [GCSR04, Nea04]). In this case we say that the Beta-distribution is the conjugate prior of the binomial distribution and thus we can model the beliefs about events that follow a binomial distribution using a Beta-distribution. In addition, this results in an efficient update rule for the posterior beliefs.

More precisely, let D be data from a experiment where the observed random variable follows a binomial distribution. For example, let us assume that the data represents n coin tosses with an identical coin whose bias is θ . Moreover, let s be the number of "successes" or heads in the data and f the number of "failures" or tails in the data. In this case, the likelihood of the data is given by

$$p(D|\theta) = \binom{s+f}{s} \theta^s (1-\theta)^f. \quad (37)$$

Next, assume that the parameter θ is also a random variable and that it is Beta-distributed with parameters α and β that are initialized to some values α_0 and β_0 . Using the Bayes rule, it can be easily shown that the posterior can be updated

sequentially and that the posterior distribution is also a Beta distribution, which has

$$\begin{aligned}\alpha_n &= \alpha_{n-1} + s \\ \beta_n &= \beta_{n-1} + f\end{aligned}\tag{38}$$

as its parameter values. A generalization of the binomial - Beta example is to use a multinomial distribution with a Dirichlet prior. We have use this approach also in the implemented system and we refer to [Nea04] for details about the Multinomial distribution.