# CHAPTER 1

# Introduction

*Es ist nicht das Wissen, sondern das Lernen, nicht das Besitzen, sondern das Erwerben, nicht das Dasein, sondern das Hinkommen, was den größten Genuß gewährt.*

Carl Friedrich Gauß

## 1.1 Complexity Theory and Logic

Solving problems through an algorithm was already done by the Babylonians 1600 B.C. They described a formal method to solve the problems of factorizing and finding the square root of a natural number.

In order to study what kind of problems are theoretical decidable, Alan Turing introduced in 1936 [Tur36] a simple machine model, called *Turing machine*. The Church-Turing Thesis states that the class of intuitive computable problems coincides with the problems that a Turing machine can decide. Turing also shows in [Tur36] that there are problems which are not decidable by a Turing machine and thus it is very likely that there are problems that are not decidable by any machine model.

The modern complexity theory uses this machine model to answer questions concerning how efficient a problem can be solved. On the one hand it is interesting how much time is needed to answer a question of a problem and on the other hand how much memory is necessary to solve the problem. Often one considers the question whether a given problem can be solves efficiently. A problem is efficiently decidable if a corresponding decision algorithm exists, which

needs at most polynomial running time with respect to the length of the question. If a problem can be solved in polynomial time we also say that the problem is a member of the complexity class P. Usually a problem is classified to be intractable if the best known algorithm has an exponentially large runtime.

Another well known complexity class is the class of efficiently verifiable problems. This class is called NP. A problem is in the class NP if and only if a possible solution of the problem can be verified in polynomial time. It is not known if every efficient verifiable problem is also efficiently decidable. But since every efficiently solvable problem is also efficiently verifiable we know that $\mathsf{P} \subseteq \mathsf{NP}$. The question whether P is a strict subset of NP is the most prominent question in theoretical computer science. Often this problem is called the P-NP problem. A well known problem in NP is the *travelling salesman problem*. A salesman wants to visit $n$ cities, where each city is reachable from another in a specific time. Now the problem is to determine if it is possible that the salesman can find a tour through all cities within time at most $k$.

In order to solve the P-NP question the theory of NP-complete problems was studied. NP-complete problems are roughly spoken the hardest problems within NP. If one can find an efficient algorithm for an NP-complete problem this algorithm can also be used to solve all other NP-complete problems efficiently. The first problem that was shown to be NP-complete was the satisfiability problem for Boolean formulae (SAT). Cook and Levin have proven separately that that SAT is NP-complete in [Coo71, Lev73]. The satisfiability and the model checking problem are together the most important decision problems concerning the computational complexity of a logic. In the case of Boolean formulae the satisfiability problems decides whether for a given formula an assignments of the variables exists such that the formula evaluates to true. On the other hand the model checking problem for Boolean formulae decides whether a given formula without variables is equivalent to true or not.

Since Cook and Levin had shown the NP-completeness result for SAT many other problems were shown to be NP-complete. Since so much work is done regarding the P-NP question and many problems were classified to be NP-complete, but no one was able to prove a P algorithm for any NP-complete problem, a common conjecture is that P is a strict subset of NP.

In this thesis the focus is not on the complexity of Boolean formulae, but on an extension of such formulae. We are studying the computational complexity of modal logic (ML) variants in the context of functional dependencies between propositions. Modal logic adds to Boolean formulae the possibility to express sentences like "It is possible that..." and "It is necessary that...". During this thesis we will study several extensions of this logic and classify them with respect to their expressibility and computational complexity.

In the following sections we will give an introduction into the topics and logics that are studied in this thesis. We will also show for each topic how it fits into the current state of research.

## 1.2 Dependence Logic

In natural language it is common to express dependencies like:

"The occurrence of a rainbow is not only determined by the current weather."

From a scientific point of view expressing dependencies between values of variables is very important. For example dependencies are occurring in the data of physical experiments or in computer science during the execution of a discrete system.

An often used tool for formalizing systems, experiments or facts is first-order logic (FO). Consider the following FO-formula:

$$\exists x_0 \exists x_1 \forall x_2 \exists x_3 \phi,$$

where $\phi$ is quantifier free. In the usual interpretation the value of $x_3$ depends on the values of $x_0$, $x_1$ and $x_2$, but suppose in order to state the required property we have to encode that the value of $x_3$ does not depend on $x_1, x_2$ and that the values of $x_1, x_2$ are independent of $x_0$. Therefore Henkin introduced in [Hen61] the notion of partially-ordered quantifiers, where the desired property can be expressed through a parallel execution of the quantifiers:

$$\begin{pmatrix} \exists x_1 \forall x_2 \\ \exists x_0 \exists x_3 \end{pmatrix} \phi.$$

If we want to express a more complex dependence, this technique has its limits. Assume that $x_3$ should not depend on the value of $x_1$

and $x_2$. Because this dependence structure can not be easily transformed into partially ordered quantifiers, Hintakka and Sandu introduced in [Hin98] the notion of independence friendly logic, where independencies are written as:

$$\exists x_0 \exists x_1 \forall x_2 \exists x_{3 \setminus \{x_1, x_2\}} \phi.$$

Recently in 2007 Väänänen introduced dependence logic (D), where dependencies are modelled by a special first-order dependence atom. The following atom expresses that the term $t_n$ is functionally determined by the terms $t_1, \ldots, t_{n-1}$:

$$=(t_1, \ldots, t_{n-1}; t_n)$$

Functional dependence in this context means that the value of $t_n$ is given by $f(t_1, \ldots, t_{n-1})$ for an arbitrary function $f\colon A^{n-1} \to A$, where $A$ is the universe. Now the sentence that $x_3$ does not depend on $x_1, x_2$ and $x_1, x_2$ are independent from $x_0$ is modelled in D as

$$\exists x_0 \exists x_1 \forall x_2 \exists x_3 =(x_0; x_3) \wedge =(x_1) \wedge =(x_1; x_2) \wedge \phi.$$

Whereas the other sentence that $x_3$ only depends on $x_0$ is modelled by

$$\exists x_0 \exists x_1 \forall x_2 \exists x_3 =(x_0; x_3) \wedge \phi.$$

Clearly the notion of dependence does not make sense in the context of a single world or a single experiment. Therefore we have to consider multiple worlds or experiments in order to observe dependencies. We call such sets of worlds, or experiments, teams.

For example consider the team described by Table 1.1 which corresponds to the multiple execution of a physical experiment, where in each experiment the position of a particle and its velocity is measured over $k$ time steps. An element of the considered universe is a tuple of two natural numbers.

Now we want to verify the physical law of classical mechanics that the position of a particle is determined by its previous position and its prior velocity.

$$\bigwedge_{i=1}^{k-1} =(p_i, v_i; p_{i+1})$$

Obviously for the team given in Table 1.1 this is true, because the following function $f$ exists:

$$f(p, v) = v + p.$$

| | $p_1$ | $v_1$ | $p_2$ | $v_2$ | $\cdots$ | $p_k$ | $v_k$ |
|---|---|---|---|---|---|---|---|
| $t_1$ | (5,4) | (1,0) | (6,4) | (3,2) | $\cdots$ | (2,1) | (1,0) |
| $t_2$ | (7,2) | (1,1) | (8,3) | (3,0) | $\cdots$ | (8,5) | (0,1) |
| $t_3$ | (7,4) | (2,1) | (9,5) | (1,2) | $\cdots$ | (2,7) | (3,2) |
| $t_4$ | (8,0) | (2,2) | (10,2) | (2,1) | $\cdots$ | (4,8) | (0,1) |
| $t_5$ | (9,0) | (3,2) | (12,2) | (1,0) | $\cdots$ | (0,0) | (2,0) |
| $t_6$ | (9,6) | (3,3) | (12,9) | (2,1) | $\cdots$ | (1,4) | (4,3) |
| $t_7$ | (1,4) | (4,3) | (5,7) | (1,0) | $\cdots$ | (0,7) | (1,4) |

Table 1.1: Experimental data

## 1.3 Modal Dependence Logic

Modal Dependence Logic (MDL) was introduced by Väänänen in 2008 [Vää08] and transfers the concept of functional dependence into the context of modal logic.

In contrast to first-order dependence logic in MDL dependencies occur between propositions and not between terms. By considering the standard embedding of modal logic into first-order logic, dependencies are expressed between unary relations and not between terms.

Because the phenomena of dependence cannot be observed in a single world of a Kripke model, we have to consider a set of worlds. We call such sets of worlds teams. A team could for example be a set of evaluations of a physical experiment or multiple states of a discrete system.

In the following we want to give an example on how dependencies can be modelled in a discrete system. Therefore consider the following safety condition of an automotive software system.

> "The activation of the anti-lock breaking system (ABS) is determined by the relational speed of the four wheels."

In order to express this sentence we encode the relational speed for each wheel by 8 propositions, which corresponds to an 8-bit sampling of the relational speed,

$$s_1^1, \ldots s_8^1, \ldots, s_1^4, \ldots, s_8^4.$$

If the ABS is activated is encoded with 1-bit by $a_{\mathrm{ABS}}$. Now the following MDL-formula expresses the sentence:

$$=(s_1^1, \ldots, s_8^1, \ldots s_1^4, \ldots, s_8^4; a_{\mathrm{ABS}}).$$

In order to study the computational complexity of MDL, usually two decision problems are considered. The first is the satisfiability problem, which decides for a given MDL-formula whether there exists a Kripke model and a team such that the formula is satisfied. The second is the model checking problem which decides for a given MDL-formula, a given Kripke model and a given team whether the formula is fulfilled in the model and the team or not. These decision problems are well studied for MDL. In [Sev09] Sevenester showed that satisfiability for MDL-formulae is NEXPTIME-complete, which means it is intractable. Therefore Lohmann and Vollmer classified in [LV10] the satisfiability complexity of MDL operator fragments. They have shown that there are tractable operator fragments where the corresponding satisfiability problem is P-complete.

Ebbing and Lohmann investigated in [EL12] the model checking complexity of MDL and they showed that model checking for MDL-formulae is NP-complete. In order to find tractable fragments of MDL they also studied the complexity for operator fragments and they found subsets of MDL were the corresponding model checking problem is decidable in P.

### 1.3.1 Modal Team Logic

Modal team logic (MTL) extends MDL by Boolean negation. Thus MTL can make use of all Boolean functions and many other team and dependence operators. Since all Boolean functions are available, MTL can express a very important kind of sentences:

> "If in a system a property $q$ is determined by the properties $p_1, \ldots, p_n$ than the system has to be in the state $\phi$."

This kind of implication is not expressible in MDL, but it is a very useful in the description of automated systems.

Another kind of implication was investigated by Ebbing, Lohmann and Yang in [ELY11]. They studied modal intuitionistic dependence logic (MIDL), which adds to MDL an intuitionistic implication. They have shown that MIDL-MC is PSPACE-complete. We will see in Chapter 4 that MTL can simulate MIDL as well.

Actually we will show that MTL is the most expressive modal dependence logic that had been considered so far.

## 1.4 Independence Logic

Functional dependencies were widely studied since Vänäänen introduced dependence logic, for example in [Kon10, Loh12, Ebb12, Yan14]. Somehow functional dependence is the strongest notion of dependence because a variables value is absolutely determined by other variables values. Väänänen introduced together with Grädel in 2013 [GV13] independence logic (I). The following sentence in independence logic

$$x_1, \ldots, x_n \perp y_1, \ldots y_n$$

denotes that the variables $x_1, \ldots, x_n$ are independent of the variables $y_1, \ldots, y_n$. But what does this mean exactly? In order to establish such a strong independence notion, as Väänänen did for dependence logic, it means that if the values of $x_1, \ldots, x_n$ are given we can say nothing about the values of $y_1, \ldots, y_n$ and vice versa.

This kind of independence is used in many scientific disciplines like physics, cryptography or statistical mathematics. For example consider the following informal example which corresponds to a law of physics:

"The size of a falling particle is independent of its mass."

Also consider the following example, which relates to the notion of independence in statistical mathematics, which is compatible to the independence notion given above:

"The result of rolling a dice is independent from the result of rolling another dice."

In [GV13] Grädel and Väänänen also studied the expressive power of I. They have shown that independence logic is equally expressive as existential second-order logic which is again equally expressive as dependence logic.

In this thesis we want to investigate a modal version of independence logic, called modal independence logic (MIL). We will show that from a complexity point of view MDL and MIL are equivalent for the decision problems satisfiability and model checking, because the decision problems for the two problems are decidable in the same complexity classes. But we will also show that they differ in their expressive power. In contrast to first-order logic, MIL is strictly more expressive than MDL.

### 1.4.1 Generalized Dependence Atoms

We have seen so far dependence and independence logic which are both extensions of first-order logic with teams semantics by new atoms. Galliani introduced in [Gal12] two other atoms, namely the inclusion atom ($\subseteq$) and the exclusion atom ($|$). He proved that $\mathsf{FO}(\subseteq, |)$ is also equally expressive as existential second order logic and thus $\mathsf{I} \equiv \mathsf{D} \equiv \mathsf{FO}(\subseteq, |)$. We do not want to get deeper into the definition of these newly added atoms, but since all of this logics are equivalent in terms of expressive power, the question arises if their is a more general notion of team atoms. Therefore Kuuisto introduced in [Kuu13] the notion of generalised dependence atoms. He introduces generalised atoms trough first-order formulae which are defining the atoms. Afterwards it was shown by Galliani in [Gal13] that with the independence atom all generalised atoms can be expressed, that are defined trough a first-order formula and are fulfilled by the empty team.

In this thesis we are introducing the concept of generalised dependence atoms for modal logic. We will give a notion of generalised atoms that are defined trough sets of Boolean matrices and a more restricted characterisation trough FO-formulae over Kripke structures. We will show that the computational complexity of satisfiability and model checking for modal logic with FO-definable atoms is at most as complex as for MDL.

## 1.5 Temporal Logic

Temporal logics are a well studied technique for verifying and modelling automated systems. Temporal logics extending modal logics by introducing new modalities, called temporal operators. In [BdRV01, Chapter 1.2] the basic version of temporal logic TL was discussed. TL is a multi-modal logic, where each modality operates on a different relation. In TL the past operator $\langle P \rangle$ and the future operator $\langle F \rangle$ were introduced, where the past operator is defined on a backwards and the future operator on a forwards relation. Usually these relations are defined in dual to each other.

Since this approach is not expressive enough to formulate complex temporal properties of a system, the computational tree logic CTL was introduced by Emerson and Clark in [CE81]. Also linear temporal logic LTL is a widely used temporal logic, which was intro-

duced by Sistla and Clark [SC85]. In both logics the newly added 'until', 'future' and 'globally' modalities are used to express temporal phenomena. The difference between CTL and LTL is that in LTL properties are always expressed over a single path in time. For example properties of a single run of an automated system. Whereas CTL expresses temporal properties on the possible branching structure of a system. Therefore consider the following LTL-formula

$$E(F\phi \to G\psi),$$

which informally means that there exists a path such that if there exists a point in the future where $\phi$ holds, $\psi$ has to hold on the current and all future points. This property is not expressible in CTL. In Chapter 6 we transfer the concepts of the basic temporal logic to modal dependence logic. We introduce extended modal dependence logic (EMDL) as an extension of MDL by multi-modalities and allowing to state dependencies between multi-modal ML-formulae. Now in contrast to MDL, EMDL can express dependencies between events in time. Therefore consider the following possible physical behaviour in classical mechanics:

"The state of a particle is determined by its $k$ previous states."

We can not express this phenomena with MDL, but with the following EMDL formula

$$=(\langle P\rangle\phi, \langle P\rangle^2\phi, \ldots, \langle P\rangle^k\phi; \phi),$$

where $\phi$ encodes the state of the particle.

Again as mentioned for basic temporal logic with EMDL it is difficult to express complex temporal properties of an automated system. Thus in the conclusion of Chapter 6 we give an outlook towards linear temporal dependence logic (LTDL). With LTDL it is possible to express dependencies in points of time during the evaluation of one specific time line.

## 1.6 Publications

Chapter 3 is based on [EKMV12]. In Chapter 4 Theorems 31, 33 and Section 4.3 base on [MV13]. The other results in Chapter 4 are new. The results of Chapter 5 base on [KMSV14]. Finally Chapter 6 base on [EHM+13], but the part about linear temporal dependence logic in the conclusion is new.

## 1.7 Results

In Chapter 3 we will introduce two Horn fragments of dependence logic. We will show that the less restricted version, where at most one occurrence of a determined existentially quantified variable is allowed in each clause, can express NP-complete problems. Also in that chapter we will prove that the more restricted version, where additionally existentially quantified variables $y, y'$ can only occur in atoms like $y = y'$ and $y = 0$, is equivalent to second-order Horn logic.

In Chapter 4 we turn to the study of variants of modal logic with team semantics. We will show that MTL-MC is in general PSPACE complete, but by restricting the set of allowed operators we obtain $\mathsf{P}^{\mathsf{NP}[1]}$, P, NL and $\mathrm{NC}^1$-complete fragments. We will also study the influence of the Boolean negation. We will prove that by restricting the nesting of the Boolean negation to at most $k$, MTL-MC is $\Sigma^p_{k+1}$-complete for even $k$. Finally we consider Post's technique to classify $\mathsf{MTL}_B(\Diamond)$-MC for any arbitrary set $B$ of Boolean functions. Furthermore we will show that $\mathsf{MTL}_{\mathrm{L}}(\Diamond)$-MC is PSPACE-complete and thus the exclusive disjunction is sufficient for a PSPACE-completeness result.

A modal variant of independence logic will be studied in Chapter 5. We consider the expressive power of MIL and we will show that on teams MIL is more expressive than MDL and on singletons MIL is equally expressive as ML. We also classify the computational complexity of the decision problems satisfiability and model checking. It will be obtained that MIL-SAT is NEXPTIME-complete and MIL-MC is NP-complete. Afterwards we generalise some of these results to a whole set of team based modal logics. We consider generalised dependence atoms and we will show that for FO-definable atoms the upper bound, for satisfiability and model checking, are the same as for MIL. The expressive power of these atoms on singletons will be proven to be equal to the expressive power of ML.

Finally in Chapter 6 we define a modal dependence logic variant which can express temporal phenomena. We will show that EMDL-MC is NP-complete and EMDL-SAT is NEXPTIME-complete. By comparing the expressive power of EMDL and MDL we will show that EMDL is strictly more expressive. We will also give a short outlook towards the definition of a linear temporal dependence logic.