CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

1
Introduction

Dependence is a common phenomenon, wherever one looks: ecological systems, astronomy, human history, stock markets. With global warming, the dependence of life on earth on the actions of mankind has become a burning issue. But what is the logic of dependence? In this book we set out to make a systematic logical study of this important concept.

Dependence manifests itself in the presence of multitude. A single event cannot manifest dependence, as it may have occurred as a matter of chance. Suppose one day it blows from the west and it rains. There need not be any connection between the wind and the rain, just as if one day it rains and it is Friday the 13th. But over a whole year we may observe that we can tell whether rain is expected by looking at the direction of the wind. Then we would be entitled to say that in the observed location and in the light of the given data, whether it rains depends on the direction of the wind. One would get a more accurate statement about dependence by also observing other factors, such as air pressure.

Dependence logic adds the concept of dependence to first order logic. In ordinary first order logic the meaning of the identity

$$x = y \tag{1.1}$$

is that the values of x and y are the same. This is a trivial form of dependence. The meaning of

$$fx = y \tag{1.2}$$

is that the interpretation of the function symbol f maps the value of x to the value of y. This is an important form of dependence, one where we actually know the mapping which creates the dependence. Note that the dependence

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

may be more subtle, as in

$$fxz = y.$$

Here y certainly depends on x but also on z. In this case we say that y depends on both x and on z, but is determined by the two together.

We introduce the new atomic formulas

$$=(x, y), \tag{1.3}$$

the meaning of which is that the values of x and y depend on each other in the particular way that values of x completely determine the values of y. Note the difference between Eqs. (1.1), (1.2) and (1.3). The first says that x determines y in the very strong sense of y being identical with x. The second says that x determines y via the mapping f. Finally, the third says there is *some* way in which x determines y, but we have no idea what that is.

The dependence in Eq. (1.3) is quite common in daily life. We have data that show that weather depends on various factors such as air pressure and air temperature, and we have a good picture of the mathematical equations that these data have to satisfy, but we do not know how to solve these equations, and therefore we do not know how to compute the weather when the critical parameters are given. We could say that the weather obeys dependence of the kind given in Eq. (1.3) rather than of the kind in Eq. (1.2). Historical events typically involve dependencies of the type in Eq. (1.3), as we do not have a perfect theory of history which would explain why events happen the way they do. Human genes undoubtedly determine much of the development of an individual, but we do not know how; we can just see the results.

In order to study the logic of dependence we need a framework involving multitude, such as multiple records of historical events, day to day observations of weather and stock transactions. This seems to lead us to study statistics or database theory. These are, however, wrong leads. If we observe that a lamp is lit up four times in a row when we turn a switch, but also that once the lamp does not light up even if we turned the switch (Fig. 1.1), we have to conclude that the light is not completely determined by the switch, as it is by the combined effect of the switch and the plug. From the point of view of dependence, statistical data or a database are relevant only to the extent that they record *change*.

In first order logic the order in which quantifiers are written determines the mutual dependence relations between the variables. For example, in

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

the variable x_1 depends on x_0, and the variable x_3 depends on both x_0 and x_2. In dependence logic we write down explicitly the dependence relations

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

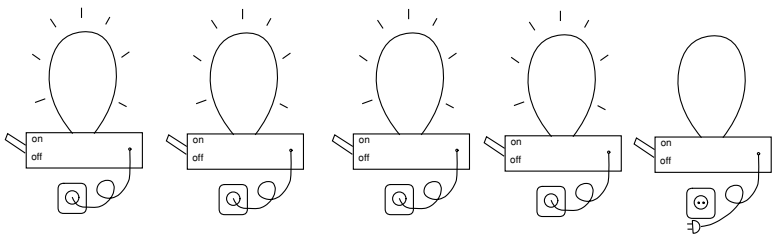

Fig. 1.1. Does the switch determine whether the lamp is lit?

between variables and by so doing make it possible to express dependencies not otherwise expressible in first order logic.

The first step in this direction was taken by Henkin with his partially ordered quantifiers, e.g.

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

where x_1 depends only on x_0 and x_3 depends only on x_2. The remarkable observation about the extension $L(H)$ of first order logic by this quantifier, made by Ehrenfeucht, was that $L(H)$ is not axiomatizable.

The second step was taken by Hintikka and Sandu, who introduced the slash-notation

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

where $\exists x_3 / \forall x_0$ means that x_3 is "independent" of x_0 in the sense that a choice for the value of x_3 should not depend on the value of x_0. The observation of Hintikka and Sandu was that we can add slashed quantifiers $\exists x_3 / \forall x_0$ coherently to first order logic if we give up some of the classical properties of negation, most notably the Law of Excluded Middle. They called their logic *independence friendly logic*.

We take the further step of writing down explicitly the mutual dependence relationships between variables. Thus we write

$$\forall x_0 \exists x_1 \forall x_2 \exists x_3 (=(x_2, x_3) \wedge \phi) \tag{1.4}$$

to indicate that x_3 depends on x_2 only. The new atomic formula $=(x_2, x_3)$ has the explicit meaning that x_3 depends on x_2 and on nothing else. This results in a logic which we call *dependence logic*. It is equivalent in expressive power to the logic of Hintikka and Sandu in the sense that there are truth-preserving translations from one to the other. In having the ability to express dependence on the atomic level it is more general.

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

Formulas of dependence logic are not like formulas of first order logic. Formulas of dependence logic declare dependencies while formulas of first order logic state relations. These two roles of formulas are incompatible in the following sense. It does not make sense to ask what relation a formula of dependence logic defines, just as it does not make sense to ask what dependence a formula of first order logic states. It seems to the author that the logic of such dependence declarations has not been systematically studied before.

At the end of this book we introduce a stronger logic called *team logic*, reminiscent of the extended independence friendly logic of Hintikka. Team logic is, unlike dependence logic and independence friendly logic, closed under the usual Boolean operations and it satisfies the Law of Excluded Middle.

Historical remarks

The possibility of extending first order logic by partially ordered quantifiers was presented by Henkin [14], where also Ehrenfeucht's result, referred to above, can be found. Independence friendly logic was introduced by Hintikka and Sandu [16] (see also ref. [17]) and advocated strongly by Hintikka in ref. [19]. Hodges [21, 22] gave a compositional semantics for independence friendly logic and we very much follow his approach. Further properties of this semantics are proved in refs. [4], [23] and [41]. Cameron and Hodges [5] showed that there are limitations to the extent to which the semantics can be simplified from the one given in ref. [21]. Connections between independence friendly logic, set theory and second order logic are discussed in ref. [40].

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

2
Preliminaries

2.1 Relations

An n-tuple is a sequence $(a_1, \ldots, a_n)$ with n components $a_1, \ldots, a_n$ in this order. A special case is the empty sequence $\emptyset$, which corresponds to the case $n = 0$. A *relation* on a set M is a set R of n-tuples of elements of M for some fixed n, where n is the *arity* of R. The simplest examples are the usual *identity* relations on a set M:

$$\{(x, x) : x \in M\},$$
$$\{(x, x, y) : x, y \in M\},$$
$$\{(x, y, x) : x, y \in M\},$$
$$\{(x, y, y) : x, y \in M\},$$
$$\{(x, x, x) : x \in M\}.$$

Two special relations are the empty relation $\emptyset$, which is the same in any arity, and the unique 0-ary relation $\{\emptyset\}$. We think of a function $f : M \to M$ as a relation $\{(x, f(x)) : x \in M\}$ on M.

2.2 Vocabularies and structures

A vocabulary is a set L of constant, relation and function symbols. We use c to denote constant symbols, R to denote relation symbols, and f to denote function symbols in a vocabulary, possibly with subindexes. Each symbol s in L has an *arity* $\#_L(s)$, which is a natural number. The arity of constant symbols is zero. The arity of a relation symbol may be zero. We use $x_0, x_1, \ldots$ to denote variables.

An L-structure $\mathcal{M}$ is a non-empty set M, the *domain* of $\mathcal{M}$, endowed with an element $c^{\mathcal{M}}$ of M for each $c \in L$, an $\#_L(R)$-ary relation $R^{\mathcal{M}}$ on M for

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

$R \in L$, and an $\#_L(f)$-ary function $f^{\mathcal{M}}$ on M for $f \in L$. The L–structures $\mathcal{M}$ and $\mathcal{M}'$ are *isomorphic* if there is a bijection $\pi : M \to M'$ such that $\pi(c^{\mathcal{M}}) = c^{\mathcal{M}'}$ and for all $a_1, \ldots, a_{\#_L(R)} \in M$ we have $(a_1, \ldots, a_{\#_L(R)}) \in R^{\mathcal{M}}$ if and only if $(\pi(a_1), \ldots, \pi(a_{\#_L(R)})) \in R^{\mathcal{M}'}$, and $f^{\mathcal{M}'}(\pi(a_1), \ldots, \pi(a_{\#_L(f)})) = \pi(f^{\mathcal{M}}(a_1, \ldots, a_{\#_L(f)}))$. In this case we say that π is an *isomorphism* from $\mathcal{M}$ to $\mathcal{M}'$, denoted $\pi : \mathcal{M} \cong \mathcal{M}'$.

If $\mathcal{M}$ is an L-structure and $\mathcal{M}'$ is an L'-structure such that $L' \subseteq L$, $c^{\mathcal{M}} = c^{\mathcal{M}'}$ for $c \in L'$, $R^{\mathcal{M}} = R^{\mathcal{M}'}$ for $c \in L'$, and $f^{\mathcal{M}} = f^{\mathcal{M}'}$ for $f \in L'$, then $\mathcal{M}'$ is said to be a *reduct* of $\mathcal{M}$ (to the vocabulary L'), denoted $\mathcal{M}' = \mathcal{M}{\restriction}L'$, and $\mathcal{M}$ is said to be an *expansion* of $\mathcal{M}'$ (to the vocabulary L). If $\mathcal{M}$ is an L-structure and $a \in M$, then the expansion $\mathcal{M}'$ of $\mathcal{M}$, denoted $(\mathcal{M}, a)$, to a vocabulary $L \cup \{c\}$, where $c \notin L$, is defined by $c^{\mathcal{M}'} = a$; $(\mathcal{M}, a_1, \ldots, a_n)$ is defined similarly.

2.3 Terms and formulas

Constant symbols of L and variables are *L-terms*; if $t_1, \ldots, t_n$ are L-terms, then $f t_1 \ldots t_n$ is an L-term for each f in L of arity n. The set $\mathrm{Var}(t)$ of *variables* of a term t is simply the set of variables that occur in t. If $\mathrm{Var}(t) = \emptyset$, then t is called a *constant term*. For example, $f c$ is a constant term. Every constant term t has a definite *value* $t^{\mathcal{M}}$ in any L-structure $\mathcal{M}$, defined inductively as follows: if t is a constant symbol, $t^{\mathcal{M}}$ is defined already. Otherwise, $(f t_1 \ldots t_n)^{\mathcal{M}} = f^{\mathcal{M}}(t_1^{\mathcal{M}}, \ldots, t_n^{\mathcal{M}})$.

Any function s from a finite set $\mathrm{dom}(s)$ of variables into the domain M of an L-structure $\mathcal{M}$ is called an *assignment* of $\mathcal{M}$. Set theoretically, $s = \{(a, s(a)) : a \in \mathrm{dom}(s)\}$. The restriction $s{\restriction}A$ of s to a set A is the function $\{(a, s(a)) : a \in \mathrm{dom}(s) \cap A\}$. An assignment s assigns a *value* $t^{\mathcal{M}}\langle s\rangle$ in M to any L-term t such that $\mathrm{Var}(t) \subseteq \mathrm{dom}(s)$ as follows: $c^{\mathcal{M}}\langle s\rangle = c^{\mathcal{M}}$, $x_n^{\mathcal{M}}\langle s\rangle = s(x_n)$, and $(f t_1 \ldots t_n)^{\mathcal{M}}\langle s\rangle = f^{\mathcal{M}}(t_1^{\mathcal{M}}\langle s\rangle, \ldots, t_n^{\mathcal{M}}\langle s\rangle)$.

The *veritas* symbol $\top$ is an L-formula. Strings $t_i = t_j$ and $R t_1 \ldots t_n$ are *atomic L-formulas* whenever $t_1, \ldots, t_n$ are L-terms and R is a relation symbol in L with arity n. We sometimes write $(t_i = t_j)$ for clarity. Atomic L-formulas are L-formulas. If ϕ and ψ are L-formulas, then $(\phi \vee \psi)$ and $\neg\phi$ are L-formulas. If ϕ is an L-formula and $n \in \mathbb{N}$, then $\exists x_n \phi$ is an L-formula. We use $(\phi \wedge \psi)$ to denote $\neg(\neg\phi \vee \neg\psi)$, $(\phi \to \psi)$ to denote $(\neg\phi \vee \psi)$, $(\phi \leftrightarrow \psi)$ to denote $((\phi \to \psi) \wedge (\psi \to \phi))$, and $\forall x_n \phi$ to denote $\neg\exists x_n \neg\phi$. Formulas defined in this way are called *first order*. An L-formula is *quantifier free* if it has no quantifiers.

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

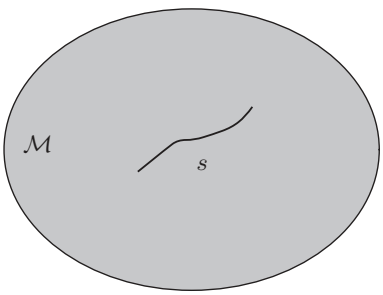

Fig. 2.1. A model and an assignment.

A formula, possibly containing occurrences of the shorthands $\wedge$ and $\forall$, is in *negation normal form* if it has negations in front of atomic formulas only.

The set $\mathrm{Fr}(\phi)$ of *free variables* of a formula ϕ is defined as follows:

$$\begin{aligned}
\mathrm{Fr}(t_1 = t_2) &= \mathrm{Var}(t_1) \cup \mathrm{Var}(t_2),\\
\mathrm{Fr}(Rt_1 \dots t_n) &= \mathrm{Var}(t_1) \cup \dots \cup \mathrm{Var}(t_n),\\
\mathrm{Fr}(\phi \vee \psi) &= \mathrm{Fr}(\phi) \cup \mathrm{Fr}(\psi),\\
\mathrm{Fr}(\neg\phi) &= \mathrm{Fr}(\phi),\\
\mathrm{Fr}(\exists x_n \phi) &= \mathrm{Fr}(\phi) \setminus \{x_n\}.
\end{aligned}$$

If $\mathrm{Fr}(\phi) = \emptyset$, we call ϕ an *L-sentence*.

2.4 Truth and satisfaction

Truth in first order logic can be defined in different equivalent ways. The most common approach is the following, based on the more general concept of *satisfaction* of L-formulas. There is an alternative game theoretic definition of truth, most relevant for this book, and we will introduce it in Chapter 5. In the definition below the concept of an assignment s satisfying an L-formula ϕ in an L-structure, denoted $\mathcal{M} \models_s \phi$, is defined by giving a sufficient condition for $\mathcal{M} \models_s \phi$ in terms of subformulas of ϕ.

For quantifiers we introduce the concept of a *modified* assignment. If s is an assignment and $n \in \mathbb{N}$, then $s(a/x_n)$ is the assignment which agrees with s everywhere except that it maps x_n to a. In other words, $\mathrm{dom}(s(a/x_n)) = \mathrm{dom}(s) \cup \{x_n\}$, $s(a/x_n)(x_i) = s(x_i)$ when $x_i \in \mathrm{dom}(s) \setminus \{x_n\}$, and $s(a/x_n)(x_n) = a$.

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

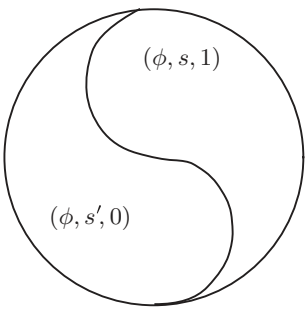

Fig. 2.2. Truth and falsity.

We define $\mathcal{T}$ as the smallest set such that:

(P1) if $t_1^{\mathcal{M}}\langle s\rangle = t_2^{\mathcal{M}}\langle s\rangle$, then $(t_1 = t_2, s, 1) \in \mathcal{T}$;
(P2) if $t_1^{\mathcal{M}}\langle s\rangle \neq t_2^{\mathcal{M}}\langle s\rangle$, then $(t_1 = t_2, s, 0) \in \mathcal{T}$;
(P3) if $(t_1^{\mathcal{M}}\langle s\rangle, \ldots, t_n^{\mathcal{M}}\langle s\rangle) \in R^{\mathcal{M}}$, then $(Rt_1 \ldots t_n, s, 1) \in \mathcal{T}$;
(P4) if $(t_1^{\mathcal{M}}\langle s\rangle, \ldots, t_n^{\mathcal{M}}\langle s\rangle) \notin R^{\mathcal{M}}$, then $(Rt_1 \ldots t_n, s, 0) \in \mathcal{T}$;
(P5) if $(\phi, s, 1) \in \mathcal{T}$ or $(\psi, s, 1) \in \mathcal{T}$, then $(\phi \vee \psi, s, 1) \in \mathcal{T}$;
(P6) if $(\phi, s, 0) \in \mathcal{T}$ and $(\psi, s, 0) \in \mathcal{T}$, then $(\phi \vee \psi, s, 0) \in \mathcal{T}$;
(P7) if $(\phi, s, 1) \in \mathcal{T}$, then $(\neg\phi, s, 0) \in \mathcal{T}$;
(P8) if $(\phi, s, 0) \in \mathcal{T}$, then $(\neg\phi, s, 1) \in \mathcal{T}$;
(P9) if $(\phi, s(a/x_n), 1) \in \mathcal{T}$ for some a in M, then $(\exists x_n\phi, s, 1) \in \mathcal{T}$;
(P10) if $(\phi, s(a/x_n), 0) \in \mathcal{T}$ for all a in M, then $(\exists x_n\phi, s, 0) \in \mathcal{T}$.

Finally we define $\mathcal{M} \models_s \phi$ if $(\phi, s, 1) \in \mathcal{T}$. A formula ψ is said to be a *logical consequence* of another formula ϕ, in symbols $\phi \Rightarrow \psi$, if for all $\mathcal{M}$ and s such that $\mathcal{M} \models_s \phi$ we have $\mathcal{M} \models_s \psi$. A formula ψ is said to be *logically equivalent* to another formula ϕ, in symbols $\phi \equiv \psi$, if $\phi \Rightarrow \psi$ and $\psi \Rightarrow \phi$.

Exercise 2.1 *Prove for all first order ϕ: $(\phi, s, 1) \in \mathcal{T}$ or $(\phi, s, 0) \in \mathcal{T}$.*

Exercise 2.2 *Prove that for no first order ϕ and for no s we have $(\phi, s, 1) \in \mathcal{T}$ and $(\phi, s, 0) \in \mathcal{T}$.*

Exercise 2.3 *Prove for all first order ϕ: $(\neg\phi, s, 1) \in \mathcal{T}$ if and only if $(\phi, s, 1) \notin \mathcal{T}$.*

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

We define two operations $\phi \mapsto \phi^{\mathrm{p}}$ and $\phi \mapsto \phi^{\mathrm{d}}$ by simultaneous induction, using the shorthands $\phi \wedge \psi$ and $\forall x_n \phi$, as follows:

$$\begin{aligned}
\phi^{\mathrm{d}} &= \neg\phi \text{ if } \phi \text{ atomic},\\
\phi^{\mathrm{p}} &= \phi \text{ if } \phi \text{ atomic},\\
(\neg\phi)^{\mathrm{d}} &= \phi^{\mathrm{p}},\\
(\neg\phi)^{\mathrm{p}} &= \phi^{\mathrm{d}},\\
(\phi \vee \psi)^{\mathrm{d}} &= \phi^{\mathrm{d}} \wedge \psi^{\mathrm{d}},\\
(\phi \vee \psi)^{\mathrm{p}} &= \phi^{\mathrm{p}} \vee \psi^{\mathrm{p}},\\
(\exists x_n \phi)^{\mathrm{d}} &= \forall x_n \phi^{\mathrm{d}},\\
(\exists x_n \phi)^{\mathrm{p}} &= \exists x_n \phi^{\mathrm{p}}.
\end{aligned}$$

We call ϕ^{d} the *dual* of ϕ. The basic result concerning duality in first order logic is that $\phi \equiv \phi^{\mathrm{p}}$ and $\neg\phi \equiv \phi^{\mathrm{d}}$. Thus the dual operation is a mechanical way for translating a formula ϕ to one which is logically equivalent to the negation of ϕ, without actually adding negation anywhere except in front of atomic formulas. Note that the dual of a formula in negation normal form is again in negation normal form. This is important because negation does not, *a priori*, preserve the negation normal form, unlike the other logical operations $\wedge, \vee, \exists, \forall$.

Exercise 2.4 *Show that ϕ^{p} and ϕ^{d} are always in negation normal form.*

Exercise 2.5 *Prove $(\phi^{\mathrm{d}})^{\mathrm{d}} = \phi^{\mathrm{p}}$ and $(\phi^{\mathrm{p}})^{\mathrm{p}} = \phi^{\mathrm{p}}$.*

Exercise 2.6 *Compute $(\phi^{\mathrm{p}})^{\mathrm{d}}$ and $(\phi^{\mathrm{d}})^{\mathrm{p}}$.*

Exercise 2.7 *Prove $\phi \equiv \phi^{\mathrm{p}}$ and $\neg\phi \equiv \phi^{\mathrm{d}}$ for any ϕ in first order logic.*

Both $\phi \mapsto \phi^{\mathrm{d}}$ and $\phi \mapsto \phi^{\mathrm{p}}$ preserve logical equivalence. Thus if we define the formula ϕ^*, for any first order formula ϕ written in negation normal form, to be the result of replacing each logical operation in ϕ by its dual (i.e. $\wedge$ by $\vee$ and vice versa, $\forall$ by $\exists$ and vice versa), then any logical equivalence $\phi \equiv \psi$ gives rise to another logical equivalence $\phi^* \equiv \psi^*$. This is the *Principle of Duality*.

Exercise 2.8 *Prove that $\phi \equiv \psi$ implies $\phi^* \equiv \psi^*$.*

In terms of game theoretic semantics, which we discuss in Chapter 5, the dual of a sentence, in being logically equivalent to its negation, corresponds to permuting the roles of the players.

© in this web service Cambridge University Press & Assessment www.cambridge.org

CAMBRIDGE
Cambridge University Press & Assessment
978-0-521-70015-3 — Dependence Logic
Jouko Väänänen
Excerpt
More Information

3
Dependence logic

Dependence logic introduces the concept of *dependence* into first order logic by adding a new kind of atomic formula. We call these new atomic formulas *atomic dependence formulas*. The definition of the semantics for dependence logic is reminiscent of the definition of the semantics for first order logic, presented in Chapter 2. But instead of defining satisfaction for assignments, we follow ref. [21] and jump one level up considering *sets* of assignments. This leads us to formulate the semantics of dependence logic in terms of the concept of the *type* of a set of assignments.

The reason for the transition to a higher level is, roughly speaking, that one cannot manifest dependence, or independence for that matter, in a single assignment. To see a pattern of dependence, one needs a whole set of assignments.

This is because dependence notions can be best investigated in a context involving repeated actions by agents presumably governed by some possibly hidden rules. In such a context dependence is manifested by recurrence, and independence by lack of it.

Our framework consists of three components:

teams, agents, and features.

Teams are sets of agents. Agents are objects with features. Features are like variables which can have any value in a given fixed set.

If we have n features and m possible values for each feature, we have altogether m^n different agents. Teams are simply subsets of this space of all possible agents.

Although our treatment of dependence logic is entirely mathematical, our intuition of dependence phenomena comes from real life examples, thinking of different ways dependence manifests itself in the real world. Statisticians certainly have much to say about this, but when we go deeper into the logic of dependence we see that the crucial concept is determination, not mere

© in this web service Cambridge University Press & Assessment www.cambridge.org