Dankwoord

Langs deze weg wens ik alle personen en instanties te danken, die direct of indirect hebben bijgedragen in het tot stand komen van deze thesis. Hoewel het vrij zeker is dat ik sommige personen dreig te vergeten wil ik toch enkele mensen en instellingen in het bijzonder vermelden.

In de eerste plaats gaat mijn dank uit naar mijn promotor, Jan Paredaens, voor zijn voortdurende steun en begeleiding van mijn onderzoek, en voor de talrijke raadgevingen, ook buiten het onderzoek.

Verder wens ik mijn collega's (en ex-collega's) op het Departement Wiskunde en Informatica van de UIA te danken voor de aangename werkomgeving, en voor de interessante al dan niet wetenschappelijke discussies, in het bijzonder Dirk Janssens, Raymond Verraedt, Marc Gyssens, Dirk De Baer, Chris Tuijn en Dirk Vermeir.

Ook wens ik langs deze weg alle buitenlandse collega's te bedanken die mij ter gelegenheid van congressen e. d. door hun opbouwende kritiek richtingen voor verder onderzoek hebben aangewezen.

Uiteraard ben ik zeer veel dank verschuldigd aan het Nationaal Fonds voor Wetenschappelijk Onderzoek en het Instituut tot Aanmoediging van het Wetenschappelijk Onderzoek in Nijverheid en Landbouw die dit onderzoek financiëel hebben mogelijk gemaakt.

Mijn familieleden en vrienden dank ik voor hun morele steun, en in het bijzonder mijn ouders die het mij mogelijk gemaakt hebben deze studies aan te vatten, en mijn vrouw, voor het vele geduld tijdens de soms lange werkdagen in de universiteit.

Tenslotte wil ik een woord van dank uitspreken voor D. Knuth en L. Lamport, voor de ontwikkeling van TEXen LATEX, die mij het schrijven van deze thesis en van verscheidene rapporten hebben vereenvoudigd.

Samenvatting

De decompositie van relaties is het belangrijkste onderwerp van de studie van het Relationele Gegevensbank Model geweest, gedurende de jongste vijftien jaar. De motivatie voor de decompositie is van tweeërlei aard:

1. Enkele technische argumenten zijn:
   - Decompositie vermindert redundantie: de informatie wordt niet verscheidene malen herhaald in dezelfde (of andere) relatie(s). Hierdoor verminderen ook de "update"-anomalieën: als de informatie gewijzigd wordt dient ze maar eenmaal gewijzigd te worden.
   - Decompositie verbetert de efficiëntie van het opvragen en wijzigen van informatie: doordat de relaties kleiner worden (door herhalingen te vermijden) verloopt het opzoeken sneller.
   - Kleinere relaties kunnen gemakkelijker verspreid worden over verscheidene computers dan één grote relatie.
2. Enkele gebruiker-georienteerde of semantische argumenten zijn:
   - Kleinere relaties zijn gemakkelijker te begrijpen (onder andere doordat ze betekenisvolle namen kunnen hebben).
   - Het formuleren van opvragingen en wijzigen van de informatie is eenvoudiger, doordat de overbodige informatie (voor een bepaalde opvraging of wijziging) in andere relaties zit en dus niet mee moet worden opgegeven.

Deze lijst is vanzelfsprekend niet volledig.

De decompositie-theorie heeft zich geconcentreerd op de *vertikale decompositie* van relaties. Het belangrijkste hulpmiddel voor deze decompositie vormt het bestaan van (semantische) beperkingen die moeten voldaan zijn in de databank (en dus ook in het deel van de "echte wereld" dat wordt voorgesteld door de databank). De studie van deze beperkingen

is aangevat door Codd [8], met de introductie van *functionele afhankelijkheden (fa's)*. In de vijftien jaar sinds dit eerste werk, zijn vele andere beperkingen bestudeerd, die (ruwweg) kunnen worden onderverdeeld in twee klassen: beperkingen die de structuur van de gegevensbank beschrijven, zoals *meerwaardige afhankelijkheden (mwa's)* [19] en *"join"-afhankelijkheden (ja's)* [25] (en vele andere), en beperkingen op de data zelf, zoals de *inclusie-afhankelijkheden* [4] en de *partitie afhankelijkheden* [9].

De fa's zitten eigenlijk in beide klassen: het zijn beperkingen op de data die gemakkelijk gebruikt kunnen worden om de structuur van de gegevensbank te beschrijven (en haar mogelijke decomposities).

De praktische waarde van de studie van beperkingen en decomposities hangt sterk af van de aanwezigheid van deze beperkingen in de "echte wereld". De functionele afhankelijkheid is de eenvoudigste beperking (met gemakkelijke en goede eigenschappen), maar het is ook de sterkste beperking, waardoor de kans dat ze in de echte wereld voorkomt erg klein is. De meerwaardige afhankelijkheid en de join-afhankelijkheid zijn zwakker (en dus komen ze waarschijnlijk vaker voor,) maar ze hebben meer ingewikkelde eigenschappen die moeilijker te begrijpen zijn. De vertikale decompositie, gebaseerd op fa's, mwa's en ja's, gebruikt een *join*-operator om de originele (grote) relatie te reconstrueren uit de kleine relaties die het resultaat zijn van de decompositie. Deze decompositie bevredigt technische en semantische behoeften: ze genereert kleinere relaties met minder redundantie, en ze scheidt onafhankelijke delen van de informatie.

In deze thesis behandelen we vooral functionele afhankelijkheden. Omdat dit de eenvoudigste beperkingen zijn worden ze het meest gebruikt om relaties te decomponeren. Maar omdat ze ook de sterkste beperkingen zijn is er nood aan een mechanisme om (een klein aantal) uitzonderingen toe te laten op deze beperkingen. De vertikale decompositie laat geen uitzonderingen toe. Daarom introduceren we een andere decompositiemethodologie: de *horizontale decompositie*. Deze decompositie voorziet in een behandeling van uitzonderingen op fa's die ingebouwd is in het databank-schema, dus zonder implementatie-afhankelijke technische truuks.

De horizontale decompositie voorziet meer in technische behoeften dan in semantische. De *uitzonderingen* op sommige functionele afhankelijkheden

worden verwijderd uit het "belangrijkste" deel van de relatie, en opgeslagen in andere (meestal kleine) relaties. In het "grotere" deel gelden de functionele afhankelijkheden (hoewel ze niet gelden in de gegevensbank als geheel, en ook niet in de echte wereld), en daardoor kan dit grote deel vertikaal gedecomponeerd worden. Dit betekent dat de technische voordelen van de vertikale decompositie kunnen bereikt worden, zelfs in gegevensbanken waar de vertikale decompositie (alleen) onmogelijk is (omdat de benodigde beperkingen niet gelden). De betekenins van de kleinere relaties is soms minder duidelijk, doordat verwante informatie wordt opgeslagen in verschillende relaties, terwijl sommige onafhankelijke informatie (over de uitzonderingen) in eenzelfde relatie zit.

Anderzijds biedt de horizontale decompositie een nieuwe semantische mogelijkheid: door access-permissies te geven of te ontzeggen aan sommige gebruikers, over de hele relatie of de deelrelaties, kan men hen eenvoudig de mogelijkheid geven of ontzeggen om uitzonderingen te kreëren en te verwijderen. Hiervoor is dus geen nieuwe privilege-structuur nodig.

Omdat het bestaan van fa's in de "echte wereld" dikwijls verband houdt met het bestaan van andere fa's, bestuderen we dit verband. Dit leidt tot een hierarchie van klassen van nieuwe beperkingen, die we *partiële implicaties tussen functionele afhankelijkheden* noemen. De horizontale decompositie kan gebaseerd worden op deze nieuwe beperkingen zonder de mogelijkheid te verliezen om uitzonderingen op willekeurige fa's te behandelen.

De horizontale decompositie introduceert een nieuwe soort update-anomalie: door informatie toe te voegen, te verwijderen of te veranderen kan men uitzonderingen kreëren of verwijderen. Doordat de uitzonderingen opgeslagen zijn in een afzonderlijke relatie (of relaties) kan een update de verplaatsing van data veroorzaken van een relatie naar een andere. Gelukkig kan dit update-probleem op efficiënte wijze opgelost worden. We geven een polynomiaal update algoritme, dat de "inter-relationele" datatraffiek minimaliseert.

Deze thesis is als volgt ingedeeld: in hoofdstuk 2 introduceren we de notaties en terminologie over het Relationele Gegevensbank Model, functionele afhankelijkheden en horizontale decomposities. We stellen ook

enkele theoretische hulpmiddelen voor die doorheen heel de thesis worden gebruikt, zoals de *Armstrong relaties* en het *conflict* concept.

In hoofdstuk 3 bestuderen we de horizontale decompositie, gebaseerd op uitzonderingen op functionele afhankelijkheden. We beschrijven de eigenschappen van functionele en *afunctionele* afhankelijkheden en we definiëren twee *Normaalvormen* waarvoor we decompositie-algoritmen geven [10, 11, 23].

In de 4 daaropvolgende hoofdstukken beschrijven we hoe de klassen van *partiële implicaties tussen fa's* kunnen gebruikt worden om horizontale decomposities te genereren. We volgen hierbij een historische aanpak: de 4 klassen van afhankelijkheden worden besproken in de volgorde waarin ze ontwikkeld werden. Elke klasse bevat de vorige klassen als bijzonder geval. De 4 klassen van beperkingen zijn: de *"voorwaardelijk-functionele afhankelijkheden"* [13], de *"opgelegd-functionele afhankelijkheden"* [14], de *"functionele-afhankelijkheid-implicaties"* [15] en de *"functionele-afhankelijkheid-verzameling-implicaties"* [16]. (De benamingen zijn vertaald uit het engels, vandaar dat ze misschien raar overkomen.)

In hoofdstuk 8 beschrijven we hoe een horizontaal gedecomponeerde databank "geupdate" kan worden [17]. Een efficiënt update algoritme wordt uitgewerkt. Een "Optimale" Normaalvorm wordt gedefiniëerd waarmee een nog efficiënter update algoritme mogelijk is. Het genereren van deze normaalvorm is niet veel moeilijker (i. e. polynomiaal) dan voor de normaalvormen van hoofdstuk 3.

Tenslotte, in hoofdstuk 9, tonen we enkele mogelijke uitbreidingen van de horizontale decompositie, voor de behandeling van uitzonderingen op andere beperkingen dan fa's.

Contents

Chapter 1

Introduction

The decomposition of relations has been a major topic in the study of the Relational Database Model during the last fifteen years. The motivations for the decomposition theory are twofold:

1. Some technical arguments are:
   - Decomposition reduces redundancy: the same information is not repeated several times in the same (or other) relation(s). This also reduces update anomalies: if the information is changed it has to be changed only once.
   - Decomposition improves query answering and updating: since the relations contain fewer information items (by removing repetitions) scanning relations goes faster.
   - Smaller relations are easier to distribute among different computers than one big relation.
2. Some user-oriented (or semantic) arguments are:
   - Smaller relations are easier to understand (and can have meaningful names).
   - Querying and updating smaller relations requires less effort, since irrelevant information (for some query or update) is contained in other relations.

This is not meant to be an exhaustive list.

The decomposition theory has focused on the *vertical decomposition* of relations. The main tool for this decomposition is the existence of some semantic constraints that must hold in the database (and hence also in the part of the "real world" represented by the database). The study of these

constraints was initiated by Codd [8], with the introduction of *functional dependencies (fd's)*. In the fifteen years after this first paper several other constraints were studied, which can be divided into two classes: constraints describing the structure of a database, such as *multivalued dependencies (mvd's)* [19] and the *join dependencies (jd's)* [25] (and many more), and constraints on the data, such as *inclusion dependencies (ind's)* [4] and *partition dependencies (pd's)* [9]. The fd's are contained in both classes: they are constraints on the data which can be easily used to describe the structure of the database (and its possible decompositions).

The practical importance of the study of constraints and decompositions highly depends on the presence of these constraints in the real world. The functional dependency is the easiest constraint (with relatively simple and nice properties), but it also is the strongest constraint, and therefore has a small chance of occurring in the real world. The multivalued dependency and join dependency are weaker, (so they probably occur more frequently,) but they have more complicated properties, which are more difficult to understand. The vertical decomposition based on fd's, mvd's or jd's uses a *join*-operator for reconstructing the original (big) relation from the small relations that are the result of the decomposition. This decomposition serves the technical and semantic needs: it generates smaller relations with less redundancy, and it separates independent pieces of information.

In this work, we are primarily concerned with functional dependencies. Since these constraints are the easiest, they are the most popular constraints for decomposing relations. But since they also are the strongest constraints, there is a need for a mechanism to allow (a small number of) exceptions to these constraints. With the vertical decomposition exceptions are absolutely forbidden. Therefore we introduce another decomposition methodology: the *horizontal decomposition*. This decomposition provides an exception handler for fd's, which is build into the database scheme, rather than some technical, implementation dependent gadgets.

The horizontal decomposition serves technical needs more than semantic needs. The *exceptions* to some functional dependencies are separated from the main part of the relation, and stored in other (probably small) relations. In the main part the fd's are satisfied (although in the database as

a whole, and in the real world they are not), and hence this main part can be decomposed vertically. This means that the technical advantages of the vertical decomposition can be obtained even in databases in which the vertical decomposition is impossible (because the necessary constraints do not hold). The meaning of the smaller relations is sometimes less obvious, since some related information is stored in separate relations (the exceptions), and some independent information (about the exceptions) is stored in one relation.

However, the horizontal decomposition also creates a new semantic aspect: by granting or denying access to the whole relation (or the subrelations), one can allow some users to create or remove exceptions, and deny this privilege to others.

Since the existence of some fd's in the real world is often related to the existence of some other fd's, we also study the relationships between fd's. This leads to a hierarchy of classes of new constraints, which we call *partial implications between functional dependencies* in general. The horizontal decomposition can be based entirely on these new constraints (without loosing the ability to consider exceptions to arbitrary fd's).

The horizontal decomposition introduces a new kind of update anomaly: by adding, deleting or modifying information one can create or remove exceptions. Since the exceptions are stored in a separate relation (or relations) an update may cause data to be moved from one relation to another. Fortunately this update problem can be solved efficiently. We present a polynomial update algorithm, which minimizes the "inter-relational" traffic of data.

This thesis is organized as follows. In Chapter 2, basic notations and terminology are introduced concerning the Relational Database Model, functional dependencies and horizontal decompositions, and some theoretical tools are provided, that will be used throughout the thesis, such as *Armstrong relations* and the *conflict* concept.

In Chapter 3 we study the horizontal decomposition based on exceptions to functional dependencies. We describe the properties of this decomposition, and we define *Normal Forms* for which we give decomposition algorithms [10, 11, 23].

In the next 4 chapters we describe how to use a class of *partial implications between fd's* for generating horizontal decompositions. We follow a historic approach: the (growing) classes of constraints are studied in the order in which they have been studied in the recent past. The 4 classes of constraints are called *conditional-functional dependencies (cfd's)* [13], *imposed-functional dependencies (ifd's)* [14], *functional dependency implications (fdi's)* [15] and *functional dependency set implications (fsi's)* [16].

In Chapter 8 we describe how to update a horizontally decomposed database [17]. An efficient update algorithm is given. An *Optimal Normal Form* is defined which leads to an even more efficient update algorithm, and it is proved that generating decompositions for this normal form is not much harder (i. e. polynomial) than for the normal forms given in Chapter 3.

Finally, in Chapter 9 we illustrate that the horizontal decomposition can also be used for handling exceptions to other constraints besides fd's.

Chapter 2

Notations and Terminology

In this chapter, we first give some basic notations and terminology about the relational database model. We then define the horizontal decomposition that will be used in Chapter 3. The horizontal decomposition will be redefined using different constraints in later chapters. We also define *Armstrong relations* and show some elementary properties.

2.1 The Relational Database Model

In the *relational database model* [6], a database is represented by a number of relations, which can be viewed as tables. We distinguish two aspects of a relation:

- the *relation scheme* (or simply *scheme*) is the structure of the relation (or the heading of the table). It is static (i. e. it does not change in time).
- the *relation instance* (or simply *instance*) represents the data stored in the relation at a given moment (i. e. the entries or rows of the table). It changes as information is being added, removed or modified regularly.

We now define these notions more formally, using the notations of [24] (with a few exceptions).

Definition 2.1 A *relation scheme* (or briefly a *scheme*) is a five-tuple $R = (\Omega, \Delta, dom, M, SC)$ where

- Ω is a finite set of *attributes*; the attributes are the headings of the columns of the table;

- Δ is a finite set of *domains*; each domain is a set of values which may be infinite;
- *dom* is a function that associates with each attribute a domain of Δ: $dom : \Omega \to \Delta$. For each attribute, only the values of the corresponding domain may appear in the column that is headed by that attribute;
- M is the *meaning* of the relation. This is an informal component of the definition, since it refers to the real world and since we will mostly describe the meaning in a human, natural language. In nearly all theoretical studies (except this one) the M component of a relation scheme has little importance. We include it in the definition of a relation scheme since it is a fundamental time independent property of the relation;
- SC is a set of *relation (scheme) constraints* or conditions. The significance of these conditions will be explained later on when we define the relation instances over a relation scheme in Definition 2.2.

□

Throughout this work, we shall use the notations of Definition 2.1 without specifying all aspects of a relation scheme. We shall usually omit the specification of the domains. We do not assume that each domain is infinitely denumerable, since this is not needed in this theory. We shall also omit the description of the meaning of a relation. To simplify this convention, we shall usually denote a relation scheme as a two-tuple $R = (\Omega, SC)$, which only contains a set of attributes and a set of constraints, or as a three-tuple $R = (\Omega, \mathcal{G}, SC)$, where $\mathcal{G}$ is a special part of the meaning, called the *set of goals* which we shall describe later.

Definition 2.2 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme.

- A *tuple* over the relation scheme R is a function t, $t : \Omega \to \bigcup_{\delta \in \Delta} \delta$ such that for every attribute A of Ω, $t(A) \in dom(A)$.
- A *relation constraint* of the relation scheme R is represented by a boolean function that associates with every set of tuples over R the value *true* or *false*. If that function associates the value *true* with a set of tuples of R, then we say that the set satisfies the relation constraint. As we indicated in Definition 2.1 the set SC contains the relation constraints of a relation scheme.

- A *relation instance* of the relation scheme $R = (\Omega, \Delta, dom, M, SC)$ is a set of tuples of R, that satisfies all the relation constraints of SC. We denote relation instances with small letters, relation schemes with capitals.

□

We represent a relation instance by a table with one column for each attribute, and one row for each tuple. It follows from Definition 2.2 that the order of the rows in the table is irrelevant and that all the rows of the table are different. Also the order of the columns (together with their heading attribute) does not influence the relation instance that is represented.

In most abstract examples we shall use capital letters from the beginning of the alphabet $(A, B, C \ldots)$ to denote single attributes, and capital letters from the end of the alphabet $(\ldots X, Y, Z)$ to denote sets of attributes. Whenever there is no ambiguity, we shall not distinguish the attribute A from the set $\{A\}$. For sets of attributes X and Y, XY denotes the union $X \cup Y$.
Lowercase letters from the beginning of the alphabet are used to indicate values of a domain $(a, b, c \ldots)$. Lowercase letters $t, u, v \ldots$ are used to denote tuples.
Capitals will also be used to denote relation schemes, and lowercase letters to denote relation instances. Usually we shall use R (resp. r) or S (resp. s) for this purpose. Whenever possible we avoid using these two letters to denote values or attributes.

A database (scheme) in the relational model may consist of several relation schemes. However, we shall use only one relation scheme. This does not mean that this work relies on the "universal relation assumption" [20]. We only study relation constraints, not "inter-relation" constraints.

Definition 2.3 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme, let r be a relation instance of R, let t be a tuple of r, and let $X \subseteq \Omega$.

- The *projection of* t *onto* X, denoted $t[X]$, is obtained by restricting t to the attributes of X.
- The *projection of* r *onto* X, denoted $\pi_X(r)$ is defined as $\{t[X] \mid t \in r\}$.

- The *projection of R onto X*, denoted $\pi_X(R)$ is the scheme for which the instances are the projections of the instances of R onto X. The reader is invited to describe this scheme as a five-tuple.

□

We shall also call the projection of t onto X the *X-projection* of t, or also the *X-value* of t.

Definition 2.4 Let r be an instance of R. Let f be a (computable) function on instances of R, which maps tuples to the boolean values *true* and *false*.

- The *selection for f of r* is the set of all tuples of r which f maps to *true* for r. It is denoted $\sigma_f(r)$.
- The *selection for f of R*, $\sigma_f(R)$, is the scheme for which the instances are the selections for f of the instances of R. The reader is invited to define this scheme as a five-tuple.

□

This definition differs from the "classical" definitions of [22, 24, 26]. The most general definition appears in [24] and still requires f to be a function on tuples, not on instances.
The term *restriction* is often used in literature, instead of *selection*. It is sometimes also used to denote the projection. To avoid confusion we never use the term restriction in this work.

Definition 2.5 Let r and s be instances of R and S respectively, and let R and S have the same Ω, Δ and *dom*.

- The *union of r and s*, $r \cup s$, is the set-theoretic union of r and s.
- The *difference of r and s*, $r - s$, is the set-theoretic difference of r and s.
- The *union of R and S*, $R \cup S$, is a scheme for which the instances are the union of an instance of R and an instance of S.
- The *difference of R and S*, $R - S$, is a scheme for which the instances are the difference of an instance of R and an instance of S.

The reader is invited to define $R \cup S$ and $R - S$ as a five-tuple.

□

2.2 Constraints and Horizontal Decompositions

In literature [7, 19, 25] many different kinds of constraints have been defined, and have mainly been used to develop the decomposition theory which is based on the *projection* and the *join* operators. The first constraint that was studied is the functional dependency, and is the constraint that will be investigated in this thesis. The decomposition theory, which we call the "horizontal" decomposition, is based on the *selection* and the *union* operators.

Definition 2.6 Let R be a relation scheme, $X, Y \subseteq \Omega$.

- A relation instance r of R satisfies the *functional dependency (fd)* $X \to Y$ iff $\forall t_1, t_2 \in r : t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$.
- The scheme R satisfies $X \to Y$ iff all the instances of R satisfy $X \to Y$.

□

Instead of "r satisfies $X \to Y$" we often say $X \to Y$ holds in r.

The functional dependency is very easy to understand, and has (as we shall see) some very nice properties. However, it is a rather severe constraint, for which one usually must allow a few exceptions. At the end of this chapter we give a (nontrivial) example of a relation scheme which "almost" satisfies a number of fd's. This example will be used and extended throughout the thesis, to illustrate other constraints.

The main purpose of the horizontal decomposition is to "separate" the exceptions to an fd from the remaining part of a relation. Therefore we must first define an operator which performs this separation.

Definition 2.7 Let R be a relation scheme, r an instance of R, $X \subseteq \Omega$.

- A set of tuples, s, $s \subseteq r$, is called X-*complete* iff the tuples belonging to s all have other X-projections than those belonging to $r - s$. Formally: $\forall t_1 \in s, t_2 \in s - r : t_1[X] \neq t_2[X]$.
- s is said to be X-*unique* iff all tuples of s have the same X-value, i. e. $\forall t_1, t_2 \in s : t_1[X] = t_2[X]$.

□

One can easily see that the empty set of tuples is X-complete for every set X of attributes of R. In the sequel we shall often use the term X-*value* to denote an "X-complete X-unique set of tuples" as well as for the X-projection of a tuple, if no confusion is possible.

Definition 2.8 Let $R = (\Omega, \Delta, dom, M, SC)$ be a scheme. Let $X, Y \subseteq \Omega$.

- For every instance r of R, the *selection for* $X \to Y$ *of* r, $\sigma_{X \to Y}(r)$, is the largest X-complete subset (of tuples) of r in which $X \to Y$ holds.
- The *selection for* $X \to Y$ *of* R, $\sigma_{X \to Y}(R)$, is a scheme $R_1 = (\Omega, \Delta, dom, M_1, SC_1)$. The calculation of SC_1 will be described in Section 3.2. SC_1 contains $X \to Y$ of course. M_1 explains that all instances of R_1 must be the selection for $X \to Y$ of the instances of R.

□

When decomposing a relation horizontally for eliminating exceptions to a functional dependency, we want to keep as many tuples in the "good" subrelation as possible, without splitting up X-values. Therefore we select the largest X-complete set of tuples in which $X \to Y$ holds. Whether the fd $X \to Y$ holds for some X-value does not depend on the tuples having other X-values. So one can obtain the largest X-complete set of tuples satisfying $X \to Y$ by taking all tuples such that $X \to Y$ holds for their X-value. It is clear that this selection operator is computable, hence it satisfies Definition 2.4. Note however that one cannot decide whether a tuple should be included in the selection without considering the instance to which the tuple belongs. Hence our selection operator does not satisfy the definitions of [22, 24, 26].

Note that although the largest X-complete (sub)set of tuples satisfying $X \to Y$ is unique, the largest (sub)set of tuples satisfying $X \to Y$ is not necessarily unique. This ambiguity is the reason why we use X-complete sets of tuples for defining the horizontal decomposition.

Definition 2.9 A *goal* is an ordered pair of sets of attributes, denoted $X \not\to Y$ to indicate that this is an fd with possible exceptions. The *horizontal decomposition of a scheme* R*, according to the goal* $X \not\to Y$ is the ordered pair (R_1, R_2), where $R_1 = \sigma_{X \to Y}(R)$ and $R_2 = R - \sigma_{X \to Y}(R)$.

□

The goals, associated with a relation scheme, indicate that one would like to have the "corresponding" fd's in the scheme, but that there may be exceptions to these fd's. Goals are not constraints, since they do not put any restriction on the acceptance of sets of tuples as instances. The number of exceptions to the fd's may be very large (if the goals are not well chosen). However, the goals are part of the relation scheme, since

they indicate how to decompose the scheme horizontally. Therefore we include the set of goals $\mathcal{G}$ of a scheme R in the meaning M. Since this is the only part of the meaning that is of fundamental importance in this work, we often denote a scheme as a three-tuple $R = (\Omega, \mathcal{G}, SC)$.

By decomposing R horizontally we obtain a subscheme R_1 in which an additional fd $X \to Y$ holds. In the other subscheme R_2 we also have a new constraint: for every X-value (in every instance of R_2) there must be at least two different Y-values. This constraint is called an *afunctional dependency*.

Definition 2.10 Let R be a relation scheme, $X, Y \subseteq \Omega$.

- A relation instance r of R satisfies the *afunctional dependency (ad)* $X \not\Rightarrow Y$ iff $\forall t_1 \in r,\ \exists t_2 \in r : t_1[X] = t_2[X] \wedge t_1[Y] \neq t_2[Y]$.
- The scheme R satisfies $X \not\Rightarrow Y$ iff all the instances of R satisfy $X \not\Rightarrow Y$.

□

Note that $X \not\Rightarrow Y$ holds in the empty instance, for all $X, Y \subseteq \Omega$.

Remark 2.1 Using definitions 2.6, 2.7 and 2.10 one can easily prove that the following ways of defining fd's and ad's are all equivalent:

- An fd $X \to Y$ holds in r iff every X-unique set of tuples of r is also Y-unique.
- An fd $X \to Y$ holds in r iff every X-complete X-unique set of tuples of r is also Y-unique.
- An fd $X \to Y$ holds in r iff every Y-complete set of tuples of r is also X-complete.
- An fd $X \to Y$ holds in r iff $\forall t_1, t_2 \in r : t_1[Y] \neq t_2[Y] \Rightarrow t_1[X] \neq t_2[X]$.
- An ad $X \not\Rightarrow Y$ holds in r iff no X-complete set of tuples of r is Y-unique.
- An ad $X \not\Rightarrow Y$ holds in r iff the fd $X \to Y$ does not hold in any X-complete set of tuples of r.

□

Usually the set of constraints SC of a relation scheme R only consists of a set $\mathcal{F}$ of fd's and a set $\mathcal{A}$ of ad's. Hence we denote a relation scheme as $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$. (In Chapter 4, we define new classes of constraints, changing the notation of SC accordingly).

The subscheme for the exceptions can be redefined as follows, using the definition of ad's:

Definition 2.11 Let R be a relation scheme, $X, Y \subseteq \Omega$.

- For every instance r of R, the *selection for* $X \not\twoheadrightarrow Y$ *of* R, $\sigma_{X \not\twoheadrightarrow Y}(r)$, is the largest X-complete subset of r in which $X \not\twoheadrightarrow Y$ holds.
- The *selection for* $X \not\twoheadrightarrow Y$ *of* R, $\sigma_{X \not\twoheadrightarrow Y}(R)$, is a scheme $R_2 = (\Omega, \Delta, dom, M_2, SC_2)$. The calculation of SC_2 will be described in Section 3.2. SC_2 contains $X \not\twoheadrightarrow Y$. M_2 explains that all instances of R_2 must be the selection for $X \not\twoheadrightarrow Y$ of the instances of R.

□

The reader will note that the horizontal decomposition of R according to $X \overset{?}{\twoheadrightarrow} Y$ is the ordered pair (R_1, R_2), where $R_1 = \sigma_{X \to Y}(R)$ and $R_2 = \sigma_{X \not\twoheadrightarrow Y}(R)$.

2.3 Implications and Contradictions between Constraints

Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme. Although SC specifies which sets of tuples over Ω are to be called instances of R, it is possible that all instances of R satisfy some other constraints as well. This leads to the following definitions:

Definition 2.12 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme.

- If all (finite or infinite) instances r of R satisfy a constraint sc, we say that sc is a *consequence of* SC, or that SC *implies* sc. We denote this by $SC \models sc$.
- If SC implies all the constraints of a set SC' we also say that SC implies SC', and denote this by $SC \models SC'$.
- The set of all constraints (of the same "class" as those of SC) that are implied by SC is denoted by SC^*, and is called the *closure of* SC.

□

We do not distinguish between finite and unrestricted implication since they coincide for the classes of constraints we consider in this work.

Definition 2.13 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme. Let $\mathcal{IR}$ be a set of *inference rules* for constraints, i. e. a set of rules to generate new constraints from a given set of constraints.

- If a constraint sc can be generated from SC using the rules of $\mathcal{IR}$ we say that sc is *inferred from* SC *by* $\mathcal{IR}$, or that SC *infers* sc. We denote this by $SC \vdash sc$. (The set of rules $\mathcal{IR}$ will always be clear, therefore we do not include it in the notation.)
- The set of all constraints that can be inferred from SC is denoted by SC^+, and is called the *saturation* of SC.
- If all constraints that can be inferred from an arbitrary set SC of constraints by $\mathcal{IR}$ also are consequences of SC, we say that the set $\mathcal{IR}$ of rules is *sound*.
- If all constraints (of a certain "class") that are consequences of an arbitrary set SC of constraints can be inferred from SC by $\mathcal{IR}$, we say that the set $\mathcal{IR}$ of rules is *complete* for this class of constraints.

□

A major part of this thesis is devoted to proving that sets of inference rules for different classes of constraints are sound and complete for these classes. We have to restrict ourselves to certain classes of constraints, since it sometimes is possible that a set of constraints implies a constraint which has not been defined. For instance, the constraint "every employee has only one manager" implies "the number of managers is less than or equal to the number of employees". The first constraint is an fd, whereas the second one is not.
Proving the soundness of a set of inference rules means proving that $SC^+ \subseteq SC^*$ for every set SC, whereas proving the completeness means proving $SC^* \subseteq SC^+$.

The presence of several constraints (of different kinds) in a relation scheme involves the danger that the constraints may be in contradiction with each other. In such a case there are no possible database instances that satisfy all constraints. In general, i. e. when all kinds of constraints are allowed, this problem is undecidable. When only fd's and ad's are considered (and also with the other constraints which we shall define later) the constraints cannot be in contradiction with each other. However there is a weak kind of contradiction that turns out to be a useful tool in the next chapters.

Definition 2.14 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme. The set SC of constraints is said to be *in conflict* if the empty set of tuples

over Ω is the only instance of R, i. e. it is the only set of tuples in which all the constraints of SC hold.

□

An easy example of a set of fd's and ad's in conflict is $\{A \rightarrow B, A \not\twoheadrightarrow B\}$. Even only one ad can be in conflict: $A \not\twoheadrightarrow A$ only holds in an empty instance.

Defining a database with only one (empty) instance certainly is useless. Therefore we shall proceed as follows, for every class of constraints we define: we first show a way to detect conflict, and we then prove the soundness and completeness of a set of inference rules, only for sets of constraints which are not in conflict.

2.4 Armstrong Relations

In literature [1, 18] a special instance has been studied, which turns out to be very useful in the proofs of several theorems in the following chapters.

Definition 2.15 Let SC be a set of constraints (of a given class) over a set Ω of attributes. An *Armstrong relation for SC* is an instance in which only the constraints of SC hold, (together with their consequences), but no other constraints of the same class.

□

In literature the existence of Armstrong relations has been investigated for different classes of constraints. It has been proven (e. g. [21]) that for some classes (e. g. functional dependencies and *inclusion dependencies* together) there are no Armstrong relations. For the purpose of this thesis we only need Armstrong relations for fd's. However, for the theorems we need instances with a somewhat stronger property.

Definition 2.16 Let $\mathcal{F}$ be a set of fd's over a set Ω of attributes. A *strong Armstrong relation for* $\mathcal{F}$ is an instance in which all the fd's of $\mathcal{F}$ hold, and also their consequences, but in which for every other fd $X \rightarrow Y$ the "corresponding" ad $X \not\twoheadrightarrow Y$ holds.

□

Note that strong Armstrong relations (for fd's) are Armstrong relations for fd's.

Note also that strong Armstrong relations for fd's are quite different from Armstrong relations for fd's and ad's together (which also exist).

In the sequel we shall only use strong Armstrong relations, but we shall usually omit the word "strong". We describe how to generate Armstrong relations below, and if we refer to Armstrong relations in the sequel, we actually refer to this construction.

Theorem 2.1 *For every set $\mathcal{F}$ of fd's there exists a strong Armstrong relation.*

Proof We consider a construction similar to that of [18]. For every set of attributes $X \subseteq \Omega$ consider the instance $r(X) =$

$\overline{X}$	$\Omega - \overline{X}$
$0 \ldots 0$	$0 \ldots 0$
$0 \ldots 0$	$1 \ldots 1$

where $\overline{X} = \{A \mid \mathcal{F} \models X \rightarrow A\}$. (The problem of determining whether $\mathcal{F} \models X \rightarrow A$ is very well known [22, 24, 26]). One can easily see that $\mathcal{F}$ holds in $r(X)$ and if $\mathcal{F} \not\models X \rightarrow Y$ then $X \rightarrow Y$ does not hold in $r(X)$. Hence $X \not\twoheadrightarrow Y$ holds, since there is only one nonempty X-complete set of tuples in r.

Let r_1 and r_2 be sets of tuples over Ω. The *direct product* of r_1 and r_2, $r_1 \otimes r_2$, is constructed as follows: if $(a_1, \ldots, a_n) \in r_1$ and $(b_1, \ldots, b_n) \in r_2$ then $r_1 \otimes r_2$ contains $((a_1, b_1), \ldots, (a_n, b_n))$. One can easily see that an fd holds in $r_1 \otimes r_2$ if it holds in both r_1 and r_2, and that an ad holds in $r_1 \otimes r_2$ if it holds in r_1 or r_2 (or in both). Consider

$$Arm(\mathcal{F}) = \bigotimes_{X \subseteq \Omega} r(X)$$

In $Arm(\mathcal{F})$, $\mathcal{F}$ holds since it holds in all $r(X)$. For every fd $X \rightarrow Y$ that is not a consequence of $\mathcal{F}$ $X \not\twoheadrightarrow Y$ holds in $r(X)$, hence also in $Arm(\mathcal{F})$. □

To illustrate the use of Armstrong relations we show that ad's have no influence on the "implication problem" for fd's, and we indicate how to detect conflict for fd's and ad's.

Theorem 2.2 *If $\mathcal{F} \cup \mathcal{A}$ is not in conflict then $\mathcal{F} \cup \mathcal{A} \models X \rightarrow Y$ iff $\mathcal{F} \models X \rightarrow Y$.*

Proof The if-part is trivial, since $\mathcal{F} \subseteq \mathcal{F} \cup \mathcal{A}$.
For the only-if-part, suppose $\mathcal{F} \cup \mathcal{A} \models X \to Y$. In $Arm(\mathcal{F})$, $\mathcal{F}$ holds and if an ad $T \not\twoheadrightarrow U$ of $\mathcal{A}$ does not hold then $\mathcal{F} \models T \to U$ by Theorem 2.1, and $\mathcal{F} \cup \mathcal{A}$ would have been in conflict, a contradiction. Hence $\mathcal{F} \cup \mathcal{A}$ holds in $Arm(\mathcal{F})$. Hence $X \to Y$ holds in $Arm(\mathcal{F})$ and by Theorem 2.1 this implies that $\mathcal{F} \models X \to Y$.

□

Theorem 2.3 *$\mathcal{F} \cup \mathcal{A}$ is in conflict iff for some $X \not\twoheadrightarrow Y$ of $\mathcal{A}$, $\mathcal{F} \models X \to Y$ holds.*

Proof The if-part is trivial.
For the only-if-part, suppose that $\mathcal{F} \cup \mathcal{A}$ is in conflict and hence the empty instance is the only instance in which $\mathcal{F} \cup \mathcal{A}$ holds. Hence in $Arm(\mathcal{F})$, (which is not empty,) in which $\mathcal{F}$ holds, some ad $X \not\twoheadrightarrow Y$ of $\mathcal{A}$ does not hold. By Theorem 2.1 this implies that $X \to Y$ holds in $Arm(\mathcal{F})$ and that $\mathcal{F} \models X \to Y$.

□

This theorem suggests a very important property of ad's: ad's do not "work" together. For instance, if $A \to B$ and $B \to C$ hold then $A \to C$ holds by transitivity. For ad's no such rule can exist: if for every A-value there are at least two B-values, and for every B-value at least two C-values, then it is still possible that not for every A-value there are at least two C-values.

From Theorem 2.3 one can easily deduce the following algorithm to detect conflict:

Algorithm 2.1 *Conflict Detection*

Input: $\mathcal{F}, \mathcal{A}$, a set of fd's and a set of ad's.

Output: *true* or *false*.

Method:
for each $T \not\twoheadrightarrow U$ **in** $\mathcal{A}$ **do**
  if $\mathcal{F} \models T \to U$
    then return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

2.5 Example

We now present a nontrivial example of a relation scheme and instance, which we shall use throughout the thesis for illustrating the different classes of constraints, and the horizontal decompositions they lead to.

Example 2.1 Consider a relation scheme $STAFF = (\Omega, \Delta, dom, M, SC)$, where

- $\Omega = \{emp, job, man, sal, dep, div\}$.
- Δ and dom are left to the reader.
- M is the meaning of $STAFF$. This says that $STAFF$ describes the staff of a large company. This company has several *div*isions, each divided into a number of *dep*artments. The work is done by *emp*loyees, who have one or more *job*s, *man*agers and *sal*aries. Also part of the meaning are the goals, which we shall describe later.
- $SC = \{emp\ job \rightarrow sal\}$. There is only one constraint, which says that every *emp*loyee earns only one *sal*ary for each of his *job*s.

The classical idea of a relation scheme would certainly include the fd $emp \rightarrow job\ man\ sal\ dep\ div$, which makes emp a key. However, one can easily see that in any large company there **will** be *emp*loyees with more than one job, or who have two *man*agers, or work in more than one *dep*artment, or even in more than one *div*ision.

Having functional dependencies is very practical for decomposing a relation vertically, but our example shows that the fd's just do not hold in the real world. Therefore we introduce a set of goals $\mathcal{G}$, which represents fd's for which we expect the number of exceptions to be small (although this assumption has no influence on the horizontal decomposition):

$$\mathcal{G} = \{emp \stackrel{?}{\twoheadrightarrow} job\ man\ sal\ dep\ div,\ man \stackrel{?}{\twoheadrightarrow} div\}.$$

The choice of these goals is arbitrary. One can find other plausible goals.

Instead of the goal $emp \stackrel{?}{\twoheadrightarrow} job\ man\ sal\ dep\ div$ one can also consider the goals $emp \stackrel{?}{\twoheadrightarrow} job$, $emp \stackrel{?}{\twoheadrightarrow} man\ dep\ div$ and $emp \stackrel{?}{\twoheadrightarrow} sal$ (several other choices of splitting up the five attributes are possible). As fd's the one goal would be equivalent to the other fd's together. As goals this is not true: with the different goals we can for instance distinguish *emp*loyees having only

one *job* but two *man*agers from *emp*loyees having two *job*s but only one *man*ager. Both employees are different kinds of exceptions, and the decomposition can generate separate subrelations for both kinds of exceptions. If we put everything into one goal, we will have only one subrelation for all kinds of exceptions. In the next chapter we shall describe the decomposition that is generated in both cases.

Table 2.1 shows a possible instance for *STAFF*. It illustrates that there are exceptions to the given goals. To keep the table reasonably small the number of exceptions is much larger than one would expect in any "real" company.

□

emp	*job*	*man*	*sal*	*dep*	*div*
Wallace	accountant	Brown	2000	sales	Los Angeles
Diamond	accountant	Brown	1500	sales	Los Angeles
Murrel	secretary	Wallace	1000	sales	Los Angeles
Murrel	secretary	Diamond	1000	sales	Los Angeles
Murrel	secretary	Brown	1000	sales	Los Angeles
Brown	sales manager	Goldstein	3000	sales	Los Angeles
Goldstein	gen. manager	Goldstein	4000	sales	Los Angeles
Eltman	carrier	Kedesdy	1000	stock	Los Angeles
Kedesdy	accountant	Brown	2000	stock	Los Angeles
Carlson	chauffeur	Kedesdy	1500	stock	Los Angeles
Pike	secretary	Kedesdy	1300	stock	Los Angeles
Goldstein	gen. manager	Goldstein	4000	stock	Los Angeles
Pierce	accountant	Shapiro	1500	sales	Santa Barbara
Goodwin	secretary	Pierce	1000	sales	Santa Barbara
Goodwin	secretary	Shapiro	1000	sales	Santa Barbara
Jones	chauffeur	Shapiro	1000	sales	Santa Barbara
Shapiro	accountant	Shapiro	1000	sales	Santa Barbara
Shapiro	sales manager	Goldstein	2000	sales	Santa Barbara
Goldstein	gen. manager	Goldstein	4000	sales	Santa Barbara
Matthews	accountant	Shapiro	1600	sales	Bakersfield
Tyrrell	chauffeur	Shapiro	1000	sales	Bakersfield
Tyrrell	carrier	Shapiro	800	sales	Bakersfield
Shapiro	sales manager	Goldstein	2000	sales	Bakersfield
Goldstein	gen. manager	Goldstein	4000	sales	Bakersfield

Table 2.1: Instance for *STAFF*.

Chapter 3

Decomposition with Goals

In this chapter we describe how to use goals for generating a horizontal decomposition of a relation scheme. This chapter covers [10, 11, 12, 23].

Consider a relation scheme $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$, and suppose that $\mathcal{F} \cup \mathcal{A}$ is not in conflict. (We can verify that using Algorithm 2.1). Suppose $\mathcal{F}$ contains $X \to Y$ and $\mathcal{G}$ contains $X \stackrel{?}{\to} Y$. If we decompose R according to $X \stackrel{?}{\to} Y$, we obtain a subrelation in which $X \to Y$ holds, and another subrelation for the exceptions, in which $X \not\to Y$ holds. But since $X \to Y$ holds in R, this second subrelation will have only an empty instance. This means that the constraints that must hold in this subrelation are in conflict.

To avoid generating conflict in the subrelations, we must know which constraints (usually called *dependencies* since they are functional or afunctional dependencies) hold in the subrelations that are the result of a horizontal decomposition. This is called the *inheritance problem* (since the dependencies of the subrelations are "inherited" from the main relation). To know which dependencies hold in the subrelations we must also be able to find out if a dependency is a consequence of a given set of dependencies.

This chapter is divided into 3 sections: First we solve the implication problem for fd's and ad's. We also provide a sound and complete set of inference rules for fd's and ad's. Then we show which dependencies hold in the subschemes of a horizontally decomposed relation. Finally, we provide two normal forms for horizontal decompositions. We give a decomposition algorithm for both normal forms, and illustrate them using Example 2.1.

3.1 The Implication Problem for fd's and ad's

For fd's only, the implication problem is very well known [22, 24, 26]. Several algorithms have been developed, of which the best ones run in $O(\sharp\mathcal{F} \times \sharp\Omega)$ time (worst case), i. e. the number of fd's multiplied by the number of attributes[1]. In this section we focus on the implication problem for mixed fd's and ad's. From Theorem 2.2 we already know that the presence of ad's does not influence the implication problem for fd's, as long as there is no conflict. So we only have to solve the implication problem for ad's (considering the presence of fd's and ad's).

We first propose the following set of inference rules for (mixed) fd's and ad's:

$(F1)$: if $Y \subseteq X$ then $X \to Y$.
$(F2)$: if $X \to Y$ and $W \subseteq V$ then $XV \to YW$.
$(F3)$: if $X \to Y$ and $Y \to Z$ then $X \to Z$.
$(A1)$: if $XV \not\twoheadrightarrow YW$ and $W \subseteq V$ then $X \not\twoheadrightarrow Y$.
$(FA1)$: if $X \to Y$ and $X \not\twoheadrightarrow Z$ then $Y \not\twoheadrightarrow Z$.
$(FA2)$: if $Y \to Z$ and $X \not\twoheadrightarrow Z$ then $X \not\twoheadrightarrow Y$.

□

The rules $F1, F2, F3$ are the "classical" inference rules for fd's, *reflexivity*, *augmentation* and *transitivity*, which are known to be sound and complete for fd's [26, 22, 24]. This means that "$\models$" and "$\vdash$" are equivalent for fd's. In this case we prefer to write $\models$ instead of $\vdash$.

Lemma 3.1 *The rules A1, FA1 and FA2 are sound.*

Proof We show that the rules cannot be false:

$A1$: Suppose $XV \not\twoheadrightarrow YW$ holds (and $W \subseteq V$) and for some X-value (in some instance) $X \not\twoheadrightarrow Y$ does not hold (hence $X \to Y$ holds for that X-value). Since $XV \to X$ by $F1$, this X-value is an XV-complete set of tuples, in which (by $F2$) $XV \to YW$ holds. This contradicts with the assumption that $XV \not\twoheadrightarrow YW$ holds.

$FA1$: Suppose that $X \to Y$ holds, and for some Y-value $Y \not\twoheadrightarrow Z$ does not hold (hence $Y \to Z$ holds for that Y-value). Consider an

[1] In [3, 26] such algorithms are claimed to run in linear time, but this depends on the representation of fd's

(arbitrary) X-value, that corresponds to that Y-value (meaning that there is a tuple in that X-value having that Y-projection). Since $X \rightarrow Y$ all tuples in that X-complete set of tuples have that same Y-value. And since we assumed that $Y \rightarrow Z$ holds for that Y-value these tuples all have the same Z-value. This means that $X \rightarrow Z$ holds for that X-value, hence $X \not\twoheadrightarrow Z$ cannot hold.

*FA*2 : Suppose $Y \rightarrow Z$ and $X \not\twoheadrightarrow Z$ hold, but for some X-value $X \not\twoheadrightarrow Y$ does not hold (hence $X \rightarrow Y$ holds for that Y-value). By $F3$ we infer that $X \rightarrow Z$ holds in this X-complete set of tuples, a contradiction with $X \not\twoheadrightarrow Z$.

□

As suggested earlier, there are no rules for deducing an ad from two (or more) other ad's. Rule $A1$ is the "opposite" of the augmentation rule for fd's, and $FA1$ and $FA2$ "reverse" the transitivity rule for fd's.

The following theorem shows the link between the implication (or "membership") problem for mixed fd's and ad's and the conflict concept.

Theorem 3.1 *Let $\mathcal{F} \cup \mathcal{A}$ be not in conflict, $X \not\twoheadrightarrow Y$ an ad. Then $\mathcal{F} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$ iff $\mathcal{F} \cup \mathcal{A} \cup \{X \rightarrow Y\}$ is in conflict.*

Proof The only-if-part is trivial.
For the if-part, suppose that $\mathcal{F} \cup \mathcal{A} \cup \{X \rightarrow Y\}$ is in conflict. Then, by Theorem 2.3 $\mathcal{F} \cup \{X \rightarrow Y\} \models T \rightarrow U$ for some $T \not\twoheadrightarrow U \in \mathcal{A}$. We prove that this implies $\mathcal{F} \cup \{T \not\twoheadrightarrow U\} \models X \not\twoheadrightarrow Y$.
Let $\overline{P} = \{A \mid \mathcal{F} \models P \rightarrow A\}$. There are two possibilities:

1. $X \not\subset \overline{T}$. Consider the following instance r:

$\overline{T}$	$\Omega - \overline{T}$
$0 \ldots 0$	$0 \ldots 0$
$0 \ldots 0$	$1 \ldots 1$

In r, $\mathcal{F}$ holds, because of the definition of $\overline{T}$; $X \rightarrow Y$ holds since $X \not\subset \overline{T}$; and $T \not\twoheadrightarrow U$ holds since $U \not\subset \overline{T}$ (otherwise $\mathcal{F} \cup \mathcal{A}$ would have been in conflict). Hence $\mathcal{F} \cup \{X \rightarrow Y\} \not\models T \rightarrow U$, since r is a counterexample, in which $\mathcal{F} \cup \{X \rightarrow Y\} \cup \{T \not\twoheadrightarrow U\}$ holds. So this case ($X \not\subset \overline{T}$) is impossible if $\mathcal{F} \cup \{X \rightarrow Y\} \models T \rightarrow U$. This means that the following case must hold:

2. $X \subseteq \overline{T}$. There are (again) two cases:
   (a) $U \not\subset \overline{YT}$. Consider the following instance r:

$\overline{YT}$	$\Omega - \overline{YT}$
$0 \ldots 0$	$0 \ldots 0$
$0 \ldots 0$	$1 \ldots 1$

   In r, $\mathcal{F}$ holds, because of the definition of $\overline{YT}$, $X \to Y$ holds since $Y \subseteq \overline{YT}$, and $T \not\Rightarrow U$ holds, since $U \not\subset \overline{YT}$. Hence $\mathcal{F} \cup \{X \to Y\} \not\models T \to U$, since $\mathcal{F} \cup \{X \to Y\} \cup \{T \not\Rightarrow U\}$ holds in r. So this case is also impossible if $\mathcal{F} \cup \{X \to Y\} \models T \to U$. This means that the following (final) case must be true:
   (b) $U \subset \overline{YT}$ (and $X \subset \overline{T}$). We show that not only $\mathcal{F} \cup \{T \not\Rightarrow U\} \models X \not\Rightarrow Y$ but even $\mathcal{F} \cup \{T \not\Rightarrow U\} \vdash X \not\Rightarrow Y$, i. e. we shall show how to infer $X \not\Rightarrow Y$ from $\mathcal{F} \cup \{T \not\Rightarrow U\}$ using the inference rules for fd's and ad's.
   - $T \not\Rightarrow U$ and $TY \to U$ induce $T \not\Rightarrow TY$ by $FA2$,
   - $T \not\Rightarrow TY$ induces $T \not\Rightarrow Y$ by $A1$ and
   - $T \to X$ and $T \not\Rightarrow Y$ induce $X \not\Rightarrow Y$ by $FA1$.

The proof is completed by remarking that $\mathcal{F} \cup \{T \not\Rightarrow U\} \subseteq \mathcal{F} \cup \mathcal{A}$. □

By using the inference rules in the above proof, this also proves the following corollary:

Corollary 3.1 *The inference rules $F1$, $F2$, $F3$, $A1$, $FA1$ and $FA2$ are complete for mixed fd's and ad's.* □

From the proof of Theorem 3.1 one can also easily deduce the following remark concerning the inference problem for fd's:

Remark 3.1 Let $\mathcal{F}$ set of fd's, $X \to Y$ an fd such that $\mathcal{F} \not\models X \to Y$ and $T \to U$ an fd such that $\mathcal{F} \not\models T \to U$ but $\mathcal{F} \cup \{X \to Y\} \models T \to U$. Then we have:

- $\mathcal{F} \models T \to X$.
- $\mathcal{F} \models TY \to U$.

□

Using Theorem 3.1 and Lemma 2.3 one can easily prove the correctness of the following membership algorithm for ad's (in the presence of fd's).

Algorithm 3.1 *Membership Detection*

Input: $\mathcal{F}, \mathcal{A}$, a set of fd's and a set of ad's, not in conflict; $X \not\twoheadrightarrow Y$ an ad.

Output: *true* or *false*.

Method:

for each $T \not\twoheadrightarrow U$ **in** $\mathcal{A}$ **do**
  if $\mathcal{F} \cup \{X \to Y\} \models T \to U$
    then return(*true*) {and exit}
od
return(*false*) {only reached if for-loop is done}

□

This algorithms confirms our previous note, that ad's do not "work" together: the implication problem for ad's always depends on only one ad.

3.2 The Inheritance of fd's and ad's

Let $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$ be a relation scheme. We already know that we should not decompose R according to $X \stackrel{?}{\twoheadrightarrow} Y \in \mathcal{G}$ if $\mathcal{F} \models X \to Y$ or $\mathcal{F} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$. (Otherwise we generate conflict in one of the subschemes.) The same restriction applies to the (further) decomposition of the subschemes (using other goals).

Determining whether for a goal $X \stackrel{?}{\twoheadrightarrow} Y$ either $X \to Y$ or $X \not\twoheadrightarrow Y$ holds in R is described in the previous section. After one decomposition step, this decision depends on the sets of dependencies that hold in the subschemes which result from that decomposition step. In the present section we show how these sets of dependencies can be calculated. Since they are derived from the "parent" scheme, we call them *inherited* dependencies.

Notation 3.1 In the sequel we treat the horizontal decomposition of a scheme $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$, according to $X \stackrel{?}{\twoheadrightarrow} Y \in \mathcal{G}$, into the (sub)schemes $R_1 = (\Omega, \mathcal{G}_1, \mathcal{F}_1 \cup \mathcal{A}_1)$ and $R_2 = (\Omega, \mathcal{G}_2, \mathcal{F}_2 \cup \mathcal{A}_2)$. We assume that $\mathcal{F} \cup \mathcal{A}$ is not in conflict, $\mathcal{F} \not\models X \to Y$ and $\mathcal{F} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$.

The sets $\mathcal{G}_1$ and $\mathcal{G}_2$ can be chosen arbitrarily. (We are free to give any meaning we like to the subrelations.) In the next section we shall show that different choices of $\mathcal{G}_1$ and $\mathcal{G}_2$ are possible, which lead to useful normal forms.

□

In this section we show how to calculate the sets $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{A}_1$ and $\mathcal{A}_2$. Since there are many sets of fd's and ad's that are equivalent (that have the same closure), we shall only show how to find a *generating* set for the dependencies.

For the fd's the "inheritance problem" is easy to formulate and to prove.

Theorem 3.2 *Using Notation 3.1,* $\mathcal{F}_1 = \mathcal{F} \cup \{X \to Y\}$ *and* $\mathcal{F}_2 = \mathcal{F}$. □

This theorem only says that fd's are always inherited. This is fairly obvious: If for some X-value there is only one Y-value in R, then there cannot be two Y-values for that X-value in a subrelation of R.

For ad's the inheritance problem is more complicated. Some ad's can be violated by decomposing a relation horizontally. This is illustrated by the following example:

Example 3.1 Let $\Omega = \{A, B, C\}$, with integer domains. Let $\mathcal{F} = \emptyset$ and $\mathcal{A} = \{A \not\twoheadrightarrow B\}$. Consider the following instance r:

A	B	C
0	0	0
0	1	0
0	1	1

Let R be decomposed according to $B \overset{?}{\twoheadrightarrow} C$. This yields

r_1

A	B	C
0	0	0

r_2

A	B	C
0	1	0
0	1	1

The ad $A \not\twoheadrightarrow B$ is violated in both r_1 and r_2. This is caused by the fact that the A-complete sets in r_1 and r_2 are not A-complete in r. □

The example suggests that for an ad $T \not\twoheadrightarrow U$ to be inherited by R_1 or R_2 the T-values of r_1 (resp. r_2) must be T-complete in r, i. e. the T-values that occur in r_1 must not occur in r_2 and vice versa. Hence, by dividing the tuples in two X-complete sets (that is what the horizontal decomposition does) the T-unique sets of tuples must not be split up. A sufficient condition is to require that all T-unique sets of tuples are also X-unique

(i. e. every T-value corresponds to only one X-value). We shall prove that this condition is also necessary.

The above observation is formalized in the following lemma:

Lemma 3.2 *Using notation 3.1, an ad $T \not\twoheadrightarrow U$ holds in both R_1 and R_2 if $\mathcal{F} \cup \mathcal{A} \models T \not\twoheadrightarrow U$ and $\mathcal{F} \models T \rightarrow X$.*

□

The "only if variant" of the above lemma is not true: Let $\mathcal{F} = \{X \rightarrow T\}$ and $\mathcal{A} = \{X \not\twoheadrightarrow U,\ T \not\twoheadrightarrow U\}$. Let R be decomposed according to $X \overset{?}{\twoheadrightarrow} Y$. Theorem 3.2 says that $X \rightarrow T$ holds in R_1 and R_2. Lemma 3.2 says that $X \not\twoheadrightarrow U$ holds in R_1 and R_2, since $X \rightarrow X$ holds. The ad $T \not\twoheadrightarrow U$ does not satisfy the condition of Lemma 3.2, since $T \rightarrow X$ does not hold. However $T \not\twoheadrightarrow U$ holds in both R_1 and R_2 since it can be inferred from $X \rightarrow T$ and $X \not\twoheadrightarrow U$ by rule $FA1$.

Now that we have a sufficient condition for ad's to be inherited, we shall develop a necessary condition. First we prove a partial result, which shows that although it is possible to lose some ad's, it is not possible to create new ones:

Lemma 3.3 *If $T \not\twoheadrightarrow U$ holds in R_1 (resp. R_2) then $\mathcal{F} \cup \mathcal{A} \cup \{X \rightarrow Y\} \models T \not\twoheadrightarrow U$ (resp. $\mathcal{F} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \models T \not\twoheadrightarrow U$).*

Proof Consider an ad $T \not\twoheadrightarrow U$ such that $\mathcal{F} \cup \mathcal{A} \cup \{X \rightarrow Y\} \not\models T \not\twoheadrightarrow U$. Then $\mathcal{F} \cup \mathcal{A} \cup \{X \rightarrow Y\} \cup \{T \rightarrow U\}$ is not in conflict, by Theorem 3.1. We shall prove that there exists an r in which $\mathcal{F} \cup \mathcal{A}$ holds, and for which $T \not\twoheadrightarrow U$ does not hold in r_1.

Let $r = Arm(\mathcal{F} \cup \{X \rightarrow Y\} \cup \{T \rightarrow U\})$. In r, $\mathcal{F} \cup \mathcal{A}$ holds since (by Theorem 2.1). After decomposing r according to $X \overset{?}{\twoheadrightarrow} Y$ $r_1 = Arm(\mathcal{F} \cup \{X \rightarrow Y\} \cup \{T \rightarrow U\})$ and $r_2 = \emptyset$. In r_1 $T \not\twoheadrightarrow U$ does not hold.

The proof for R_2 is somewhat similar. Consider an ad $T \not\twoheadrightarrow U$ such that $\mathcal{F} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$. Then $\mathcal{F} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \cup \{T \rightarrow U\}$ is not in conflict, by Theorem 3.1.

Let $r = Arm(\mathcal{F} \cup \{T \rightarrow U\})$. In r, $\mathcal{F} \cup \mathcal{A}$ holds and after decomposing r according to $X \overset{?}{\twoheadrightarrow} Y$ $r_1 = \emptyset$ and $r_2 = Arm(\mathcal{F} \cup \{T \rightarrow U\})$. In r_2 $T \not\twoheadrightarrow U$ does not hold.

□

Lemma 3.3 states (in other terms) that $(\mathcal{F}_1 \cup \mathcal{A}_1)^* \subseteq (\mathcal{F} \cup \mathcal{A} \cup \{X \to Y\})^*$ and $(\mathcal{F}_2 \cup \mathcal{A}_2)^* \subseteq (\mathcal{F} \cup \mathcal{A} \cup \{X \not\to Y\})^*$.

The result of Lemma 3.3 does not take into account the ad's of $\mathcal{A}$ that may not be inherited. Therefore we first define a smaller set of ad's:

Notation 3.2 Using Notation 3.1 let $\hat{\mathcal{A}} = \{T \not\to U \in \mathcal{A} \mid \mathcal{F} \models T \to X\}$. □

Note from Lemma 3.2 that the ad's of $\hat{\mathcal{A}}$ are inherited by both R_1 and R_2.

If an ad $T \not\to U$ does not hold in an instance then for some T-value $T \to U$ must hold in that instance. Such a T-value (i. e. T-unique and T-complete set of tuples) will be called a *violation* of $T \not\to U$.

In the proof of the inheritance of ad's a special instance is needed, of which the construction is described below. This construction starts with a set of tuples in which some ad's of $\mathcal{A}$ do not hold, and by adding some more tuples the violations of the ad's of $\mathcal{A}$ are eliminated.

Lemma 3.4 *Let $\mathcal{A}' \subseteq \mathcal{A}$, $\mathcal{F} \subseteq \mathcal{F}'$, and let $\mathcal{F}' \cup \mathcal{A}$ be not in conflict. Let sot be a set of tuples in which $\mathcal{F} \cup \mathcal{A}'$ holds, but in which some ad $P \not\to Q \in \mathcal{A} - \mathcal{A}'$ does not hold. Then one can construct a set sot$'$ of tuples, containing sot as a subset, in which $\mathcal{F} \cup \mathcal{A}'$ still holds, and in which the number of violations of $P \not\to Q$ is less than in sot.*

Proof Suppose that the domains of $Arm(\mathcal{F}')$ and *sot* are disjoint.[2] Suppose also that in $Arm(\mathcal{F}')$ the domains of the attributes all are disjoint. (This can be easily achieved by replacing $Arm(\mathcal{F}')$ by an equivalent instance with other domains.)

Let $\overline{P} = \{A \in \Omega \mid \mathcal{F} \models P \to A\}$. Let t be a tuple in an arbitrary violation of $P \not\to Q$ (in *sot*). Let s be an arbitrary tuple of $Arm(\mathcal{F}')$. The domain of $Arm(\mathcal{F}')$ (and $Arm(\mathcal{F}')$ too) is changed such that $s[\overline{P}] := t[\overline{P}]$. Let r' be the union of this adapted Armstrong relation and *sot*.

- In r', $\mathcal{F}$ still holds, since $\mathcal{F}$ holds in *sot* and $Arm(\mathcal{F}')$, and since if $V \not\subset \overline{P}$ then $V \to W \in \mathcal{F}$ still holds because no tuple of *sot* has the same V-projection as any tuple of $Arm(\mathcal{F})$, and if $V \subseteq \overline{P}$ then $V \to W \in \mathcal{F}$ still holds because also $W \subseteq \overline{P}$ and if the V-projection of

[2]Strictly speaking we must assume that the domains are the same, but that the instances use no common elements of the domains.

a tuple of *sot* and of a tuple of $Arm(\mathcal{F}')$ are equal then this projection is $t[V]$, and hence for the tuple of *sot* and the tuple of $Arm(\mathcal{F}')$ the W-projection is $t[W]$ by the construction of r'.

- In r', $\mathcal{A}'$ holds since $\mathcal{A}'$ holds in *sot* and $\mathcal{A} \supseteq \mathcal{A}'$ holds in $Arm(\mathcal{F}')$ and since an ad cannot be violated by taking a union.

- In $Arm(\mathcal{F}')$, $P \not\twoheadrightarrow Q \in \mathcal{A}$ holds, hence $Arm(\mathcal{F}')$ does not contain any violation of $P \not\twoheadrightarrow Q$. In r' the violation of $P \not\twoheadrightarrow Q$ in *sot* that contains t is no longer a violation of $P \not\twoheadrightarrow Q$. Hence in r' the number of violations of $P \not\twoheadrightarrow Q$ is (strictly) less than in *sot*.

□

The construction, used in the proof of the above lemma is illustrated by the following, (simplified) example.

Example 3.2 Consider a scheme R with attributes $\{A, B, C, D\}$, fd's $\mathcal{F} = \mathcal{F}' = \{A \rightarrow B, B \rightarrow A, C \rightarrow D, D \rightarrow A\}$ and ad's $\mathcal{A} = \{A \not\twoheadrightarrow C, A \not\twoheadrightarrow D\}$. Let $\mathcal{A}' = \{A \not\twoheadrightarrow C\}$.

Consider the following set of tuples *sot*:

A	B	C	D
a	a	a	a
a	a	b	a
b	b	c	b
b	b	d	b
c	c	e	c
c	c	f	d

In *sot* $\mathcal{F} \cup \mathcal{A}'$ holds, but $A \not\twoheadrightarrow D$ does not hold. There are two violations of $A \not\twoheadrightarrow D$: {tuples 1 and 2} and {tuples 3 and 4}.

The following instance is equivalent to $Arm(\mathcal{F})$:

A	B	C	D
0	0	0	0
1	1	1	1
0	0	2	2
0	0	3	0
1	1	4	4
1	1	5	1
0	0	6	2
1	1	7	4

With the notation of Lemma 3.4 $\overline{P} = AB$. Let $t = \{a, a, a, a\}$ and $s = \{0, 0, 0, 0\}$. After the renaming of $Arm(\mathcal{F})$, the union of *sot* and $Arm(\mathcal{F})$ is the instance:
$r' =$

A	B	C	D
a	a	a	a
a	a	b	a
b	b	c	b
b	b	d	b
c	c	e	c
c	c	f	d
a	a	0	0
1	1	1	1
a	a	2	2
a	a	3	0
1	1	4	4
1	1	4	4
1	1	5	1
a	a	6	2
1	1	7	4

In r', $\mathcal{F} \cup \mathcal{A}'$ still holds, and the number of violations of $A \not\twoheadrightarrow D$ in r' is less than in *sot*: only {tuples 3 and 4} is still a violation of $A \not\twoheadrightarrow D$.

□

The effect of repeatedly performing the above construction is described by the following corollary:

Corollary 3.2 *Let $\mathcal{A}' \subseteq \mathcal{A}$ and let $\mathcal{F} \cup \mathcal{A}$ be not in conflict. Let sot be a set of tuples in which $\mathcal{F} \cup \mathcal{A}'$ holds. Then there exists a set of tuples sot', containing sot as a subset, in which $\mathcal{F} \cup \mathcal{A}$ holds.*

Proof *sot'* is obtained by repeating the above construction for every violation of every ad of $\mathcal{A} - \mathcal{A}'$ which does not hold in *sot*. □

The construction, described in Lemma 3.4 and Corollary 3.2 is the key to the solution of the inheritance problem, not only for ad's, but also for the constraints that will be defined in the following chapters. It will also help solve the implication problem of some constraints.

The solution of the inheritance problem is not hard to prove now:

Theorem 3.3 *Using Notation 3.1 and 3.2, $\mathcal{F}_1 = \mathcal{F} \cup \{X \to Y\}$, $\mathcal{A}_1 = \hat{\mathcal{A}}$, $\mathcal{F}_2 = \mathcal{F}$ and $\mathcal{A}_2 = \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\}$.*

Proof We already know that $(\mathcal{F} \cup \{X \to Y\} \cup \hat{\mathcal{A}})^* \subseteq (\mathcal{F}_1 \cup \mathcal{A}_1)^* \subseteq (\mathcal{F} \cup \{X \to Y\} \cup \mathcal{A})^*$ and $(\mathcal{F} \cup \{X \not\twoheadrightarrow Y\} \cup \hat{\mathcal{A}})^* \subseteq (\mathcal{F}_2 \cup \mathcal{A}_2)^* \subseteq (\mathcal{F} \cup \{X \not\twoheadrightarrow Y\} \cup \mathcal{A})^*$.

Consider an ad $T \not\twoheadrightarrow U$ such that $\mathcal{F} \cup \{X \to Y\} \cup \mathcal{A} \models T \not\twoheadrightarrow U$ but $\mathcal{F} \cup \{X \to Y\} \cup \hat{\mathcal{A}} \not\models T \not\twoheadrightarrow U$. We prove that $T \not\twoheadrightarrow U$ does not hold in R_1.

Since $\mathcal{F} \cup \{X \to Y\} \cup \hat{\mathcal{A}} \not\models T \not\twoheadrightarrow U$ we know that $\mathcal{F} \cup \{X \to Y\} \cup \hat{\mathcal{A}} \cup \{T \to U\}$ is not in conflict, by Theorem 3.1. Hence there exists a set of tuples *sot* in which $\mathcal{F} \cup \{X \to Y\} \cup \hat{\mathcal{A}} \cup \{T \to U\}$ holds.

Let $r' = sot$. When r' is decomposed according to $X \overset{?}{\twoheadrightarrow} Y$, $r'_1 = sot$ and $r'_2 = \emptyset$. In *sot* $\mathcal{F} \cup \hat{\mathcal{A}}$ holds, but a number of ad's of $\mathcal{A} - \hat{\mathcal{A}}$ may not hold. By the construction of Lemma 3.4 and Corollary 3.2 one can build an instance r which contains *sot* and in which $\mathcal{F} \cup \mathcal{A}$ is satisfied. In this construction a number of copies of $Arm(\mathcal{F})$ are added to *sot*. Since $X \not\twoheadrightarrow Y$ holds in $Arm(\mathcal{F})$, and since $\mathcal{F} \cup \mathcal{A} \not\models T \to X$ (hence $X \not\subset \overline{T}$), these copies are added to r_2. Hence $r_1 = sot$ remains unchanged, implying that $T \to U$ holds in r_1. Hence $T \not\twoheadrightarrow U$ does not hold in r_1, which means that $\mathcal{F}_1 \cup \mathcal{A}_1 \not\models T \not\twoheadrightarrow U$.

For R_2 the proof is similar: Consider an ad $T \not\twoheadrightarrow U$ such that $\mathcal{F} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \models T \not\twoheadrightarrow U$, but $\mathcal{F} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$. We prove that $T \not\twoheadrightarrow U$ does not hold in R_2.

Since $\mathcal{F} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$ we know that $\mathcal{F} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\} \cup \{T \rightarrow U\}$ is not in conflict. Hence there exists a set of tuples *sot* in which $\mathcal{F} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\} \cup \{T \rightarrow U\}$ holds.

Let $r' = sot$. When r' is decomposed according to $X \stackrel{?}{\twoheadrightarrow} Y$ $r'_1 = \emptyset$ and $r'_2 = sot$. In *sot* $\mathcal{F} \cup \hat{\mathcal{A}}$ holds, but a number of ad's of $\mathcal{A} - \hat{\mathcal{A}}$ may not hold. By the construction of Lemma 3.4 and Corollary 3.2 one can build an instance r which contains *sot* and in which $\mathcal{F} \cup \mathcal{A}$ is satisfied. We perform this construction, not by adding copies of $Arm(\mathcal{F})$ but of $Arm(\mathcal{F} \cup \{X \rightarrow Y\})$ this time, (which is allowed since $\mathcal{F} \cup \{X \rightarrow Y\} \cup \mathcal{A}$ is not in conflict). Since $X \rightarrow Y$ holds in $Arm(\mathcal{F} \cup \{X \rightarrow Y\})$, and since $\mathcal{F} \not\models T \rightarrow X$ (hence $X \not\subset \overline{T}$), these copies are added to r_1. Hence r_2 remains unchanged, implying that $T \rightarrow U$ holds in r_2. Hence $T \not\twoheadrightarrow U$ does not hold in r_2, which means that $\mathcal{F}_1 \cup \mathcal{A}_2 \not\models X \rightarrow Y$.

□

3.3 Normal Forms for Horizontal Decompositions

When decomposing a relation scheme both horizontally (according to goals) and vertically (according to fd's) one should decide which decomposition has to be performed first. Our approach is to perform the horizontal decomposition steps first. This is based on Theorem 3.2, which states that the horizontal decomposition, according to a goal, preserves fd's. Also, performing the vertical decomposition first would cause problems: only the "real fd's" (without exceptions) can be used for this decomposition, and after it there may be no more subrelation with all attributes of a goal (since the subrelations that result from vertical decompositions have fewer attributes), so the horizontal decomposition may not be possible after all, and last but not least, if the horizontal decomposition is still possible, it will generate new fd's, which can be used for vertical decomposition again. So we perform the horizontal decomposition first. This means that we need not consider the normal forms for the vertical decomposition when designing normal forms for the horizontal decomposition.

There is one gap in the theory of the previous sections: we have proved the inheritance for fd's and ad's, but not for goals. The reason is that, since goals are not constraints, they are not necessarily inherited; it is an

arbitrary choice whether we want a certain goal to be inherited by the subrelations or not. The goals are part of the meaning of the relation, but the meaning of the subrelations has not yet been defined.
In this section we show two possible ways of defining the inheritance of goals, and the normal forms they lead to.

Definition 3.1 Let $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$ be a relation scheme.

- Decomposing R according to $X \twoheadrightarrow Y \in \mathcal{G}$, for which $\mathcal{F} \not\models X \to Y$ and $\mathcal{F} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$, into $R_1 = (\Omega, \mathcal{G}_1, \mathcal{F}_1 \cup \mathcal{A}_1)$ and $R_2 = (\Omega, \mathcal{G}_2, \mathcal{F}_2 \cup \mathcal{A}_2)$ is called a *horizontal decomposition step* (or *decomposition step* for short).
- The decomposition steps and the (sub)schemes together are called a *decomposition tree (for R).*
- The "final" subschemes of a decomposition tree together are called a *decomposition (of R).*

□

First we give a "trivial" way to decompose relation schemes (with goals) and we show that it is too simple to be useful.

Definition 3.2 $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$ is called an *atomic (relation) scheme* if for all $X \twoheadrightarrow Y \in \mathcal{G}$, $\mathcal{F} \models X \to Y$ or $\mathcal{F} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$.

□

Definition 3.3 Let $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$ be a relation scheme.

- A decomposition step in which the whole set of goals $\mathcal{G}$ is (defined to be) inherited by both subschemes (i. e. $\mathcal{G}_1 = \mathcal{G}_2 = \mathcal{G}$) is called a *trivial decomposition step.*
- A decomposition $(R_1, \ldots, R_n)$ of R is said to be a *trivial decomposition* of R if it is obtained by means of trivial decomposition steps. (This implies that $\mathcal{G}_1 = \ldots = \mathcal{G}_n = \mathcal{G}$.)
- A decomposition $(R_1, \ldots, R_n)$ of R is said to be in *Horizontal Normal Form (HNF)* iff all the $R_i, i = 1 \ldots n$, are atomic schemes.

□

Figure 3.1 shows a trivial decomposition step for an arbitrary scheme R (the calculation of $\mathcal{A}_1$ and $\mathcal{A}_2$ is described in Section 3.2).
We now show an example which illustrates how a "bad" choice of goals can make it impossible to obtain a trivial decomposition which is in *HNF* because the decomposition steps generate "non-atomic" schemes.

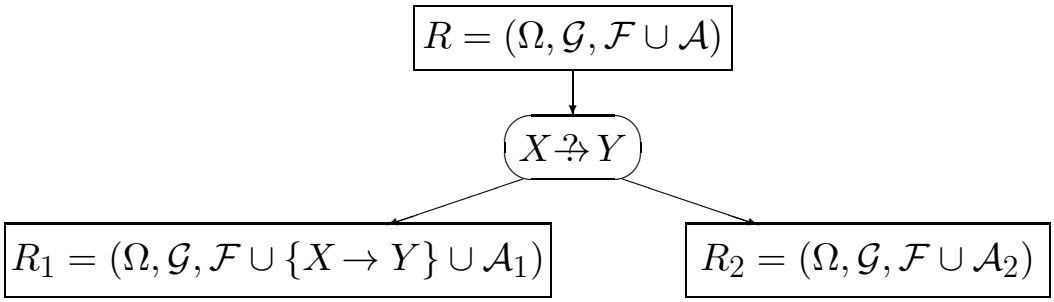


Figure 3.1: A trivial decomposition step.

Example 3.3 Consider a scheme $R = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \emptyset)$ Since R has no constraints, both goals can be used for horizontal decomposition.

Decomposing R according to $X \dot{\twoheadrightarrow} Y$ produces two subschemes: $R_1 = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \{X \to Y\})$ and $R_2 = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \{X \not\twoheadrightarrow Y\})$. Both R_1 and R_2 (still having the same set of goals) can be decomposed further on using $Y \dot{\twoheadrightarrow} X$. For R_1 this produces two atomic subschemes: $R_{11} = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \{X \to Y\ ,\ Y \to X\})$, and $R_{12} = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \{X \to Y\ ,\ Y \not\twoheadrightarrow X\})$. For R_2 the situation is different: if we decompose R_2 according to $Y \dot{\twoheadrightarrow} X$ the ad $X \not\twoheadrightarrow Y$ is lost. So the resulting subschemes $R_{21} = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \{Y \to X\})$ and $R_{22} = (\{X, Y\}, \{X \dot{\twoheadrightarrow} Y\ ,\ Y \dot{\twoheadrightarrow} X\}, \{Y \not\twoheadrightarrow X\})$ are not atomic: they can be decomposed (again) using $X \dot{\twoheadrightarrow} Y$. One can easily see that this decomposition process can go on for ever: the "rightmost" subrelation $R_{2\ldots2}$ is never atomic. Figure 3.2 shows a decomposition tree from which the effect of losing ad's but keeping goals is clear. □

There are two ways of solving the problem with trivial decomposition steps: since the infinite decomposition is caused by the fact that ad's may be lost, but goals are preserved, one can either decide to restrict the decompositions to avoid the loss of ad's, or one can define the goals not always to be inherited. Both solutions can be used, and lead to useful decomposition steps and normal forms, described below.

Definition 3.4 Let $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$ be a relation scheme.

- A goal $X \dot{\twoheadrightarrow} Y$ is said to be *clean* iff
  1. neither $X \to Y$ nor $X \not\twoheadrightarrow Y$ holds in R and

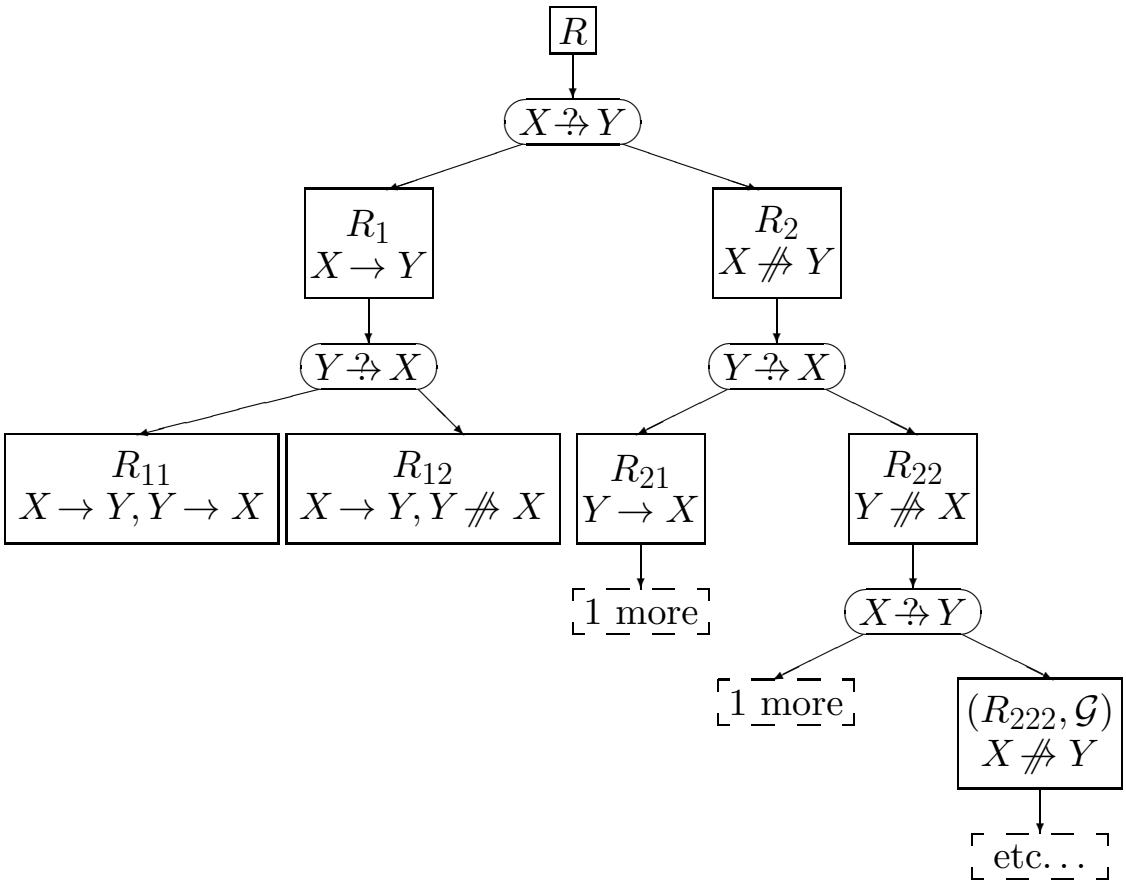


Figure 3.2: Infinite trivial decomposition tree.

2. all ad's of $\mathcal{A}$ are inherited by the decomposition of R according to $X \dot{\twoheadrightarrow} Y \in \mathcal{G}$.

- Decomposing R according to a clean goal, into R_1, R_2, is called a *clean decomposition step.*
- A decomposition $R_1, \ldots, R_n$ is said to be a *clean decomposition* of R if it is obtained by clean decomposition steps.
- A scheme R is called a *clean scheme* iff $\mathcal{G}$ does not contain any clean goal.
- A decomposition $R_1, \ldots, R_n$ is in *Clean Normal Form (CNF)* iff all the R_i, $i = 1 \ldots n$, are clean (sub)schemes.
- The decomposition steps and the subschemes together are called a *clean decomposition tree.*

□

Figure 3.3 shows a clean decomposition step for a scheme R.
Note that both $\mathcal{F}$ and $\mathcal{A}$ are inherited by the subschemes. This means that clean decomposition steps are *dependency preserving.*

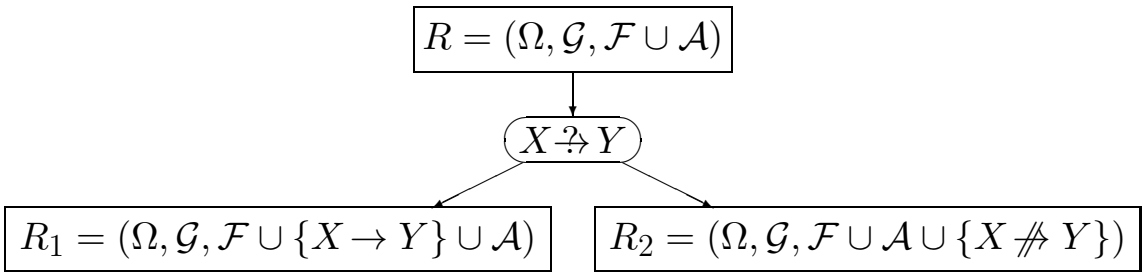


Figure 3.3: A clean decomposition step.

Since no fd or ad can be lost by clean decomposition steps, it is obvious that the clean decomposition into *CNF* cannot produce an infinite number of subschemes. In fact the maximal number of subschemes is $2^{\sharp\mathcal{G}}$. The (sub)schemes of a clean decomposition into *CNF* are not necessarily atomic schemes: there may still be goals which can be used in trivial decomposition steps, but which are not clean.

Using Theorem 3.3 one can easily show the correctness of the following algorithm for generating a clean decomposition which is in *CNF*:

Algorithm 3.2 *Clean Decomposition into CNF*

Input: $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$.

Output: A clean decomposition $(R_1, \ldots, R_n)$ of R

Method:

return($decompose(R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A}))$)

function $decompose(R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A}))$
begin
  for each $X \dot{\twoheadrightarrow} Y$ **in** $\mathcal{G}$ **do**
    if $isclean(X \dot{\twoheadrightarrow} Y, \mathcal{F} \cup \mathcal{A})$
      then
        return($decompose(R_1 = (\Omega, \mathcal{G}, \mathcal{F} \cup \{X \to Y\} \cup \mathcal{A}))$,
              $decompose(R_2 = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\}))$)
        { and exit function }
  od
  return($R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$) { if no clean goal in R }
end

function $isclean(X \dot{\twoheadrightarrow} Y, \mathcal{F} \cup \mathcal{A})$
begin

var $\hat{\mathcal{A}}$: set of ad's := $\emptyset$
if $\mathcal{F} \models X \rightarrow Y$
  then
    return(*false*)
if $\mathcal{F} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$
  then
    return(*false*)
for each $T \not\twoheadrightarrow U$ **in** $\mathcal{A}$ **do**
  if $\mathcal{F} \models T \rightarrow X$
    then
      $\hat{\mathcal{A}} := \hat{\mathcal{A}} \cup \{T \not\twoheadrightarrow U\}$
od
for each $T \not\twoheadrightarrow U$ **in** $\mathcal{A}$ do
  if $\mathcal{F} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\} \not\models T \not\twoheadrightarrow U$
    then
      return(*false*)
od
return(*true*) { only reached if goal is clean }
end

□

We now present an example to illustrate the clean decomposition into *CNF*.

Example 3.4 Recall Example 2.1. It has one fd: *emp job* $\rightarrow$ *sal*. The first proposal for $\mathcal{G}$ is $\{emp \overset{?}{\twoheadrightarrow} job\ man\ sal\ dep\ div, man \overset{?}{\twoheadrightarrow} div\}$. Figure 3.4 shows a clean decomposition tree for *STAFF*, with this set of goals.
The second set of goals, of which we claimed that it would produce more subrelations, is $\{emp \overset{?}{\twoheadrightarrow} job, emp \overset{?}{\twoheadrightarrow} man\ dep\ div, emp \overset{?}{\twoheadrightarrow} sal, man \overset{?}{\twoheadrightarrow} div\}$. Figure 3.5 shows a clean decomposition tree for *STAFF*, with this set of goals. The number of subrelations becomes rather large. Remember that the largest possible number of subrelations grows exponentially with the number of goals. This explains why we did not split up the goal with the 5 attributes on its right side into 5 different goals. The largest possible number of subschemes would become 64!
In the figures only a generating set for the constraints is given. Note that after the decomposition according to $emp \overset{?}{\twoheadrightarrow} job$, the fd $emp \rightarrow sal$ already

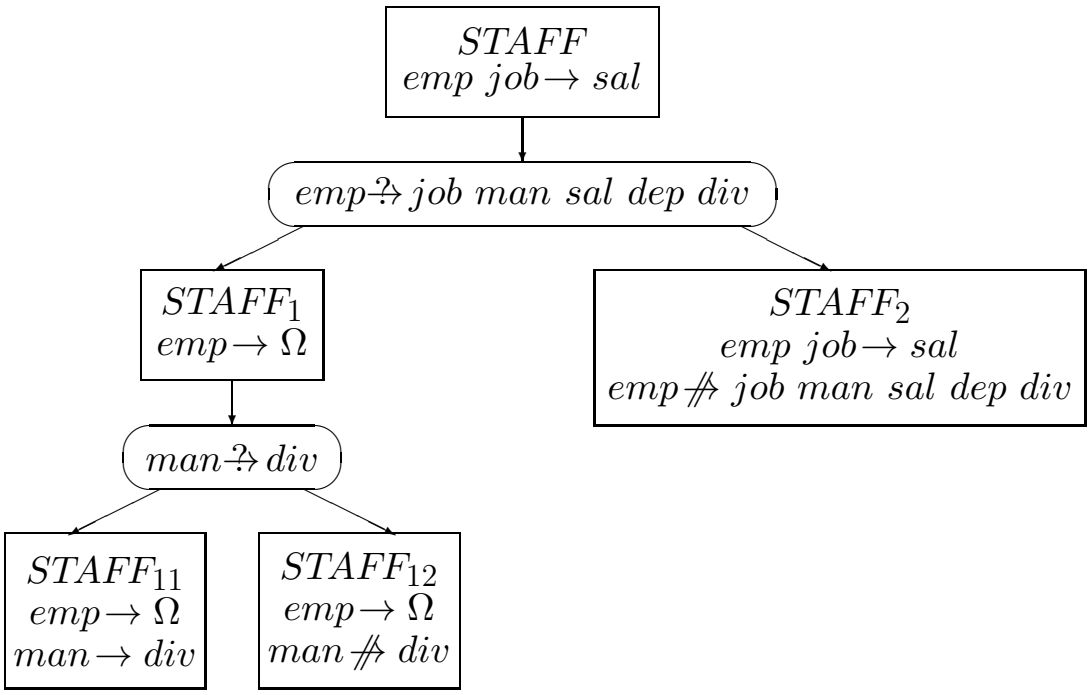


Figure 3.4: A clean decomposition tree for *STAFF* (with 3 goals).

holds in $STAFF_1$. This implies that the goal $emp \twoheadrightarrow^{?} sal$ cannot be used in the subtree below $STAFF_1$. Note also that the maximum number of subrelations is not reached (in any of the two figures). The decomposition tree, and the number of subrelations are dependent on the order in which the goals are used. But it is always possible that no choice of the order will lead to the maximal number $2^{\sharp\mathcal{G}}$.

Recall Table 2.1. When *staff* is decomposed according to the set of (4) goals, as in Figure 3.5, the following subinstances are obtained:

$staff_{111}$ contains the *emp*loyees having only one *job*, one *man*ager, one *dep*artment and one *div*ision (and hence also only one *sal*ary), and whose *man*ager has only one *div*ision for these *emp*loyees. (He may have other *div*isions for other *emp*loyees, as is the case for Brown: his *man*ager Goldstein has other *div*isions, but not for these *emp*loyees.)

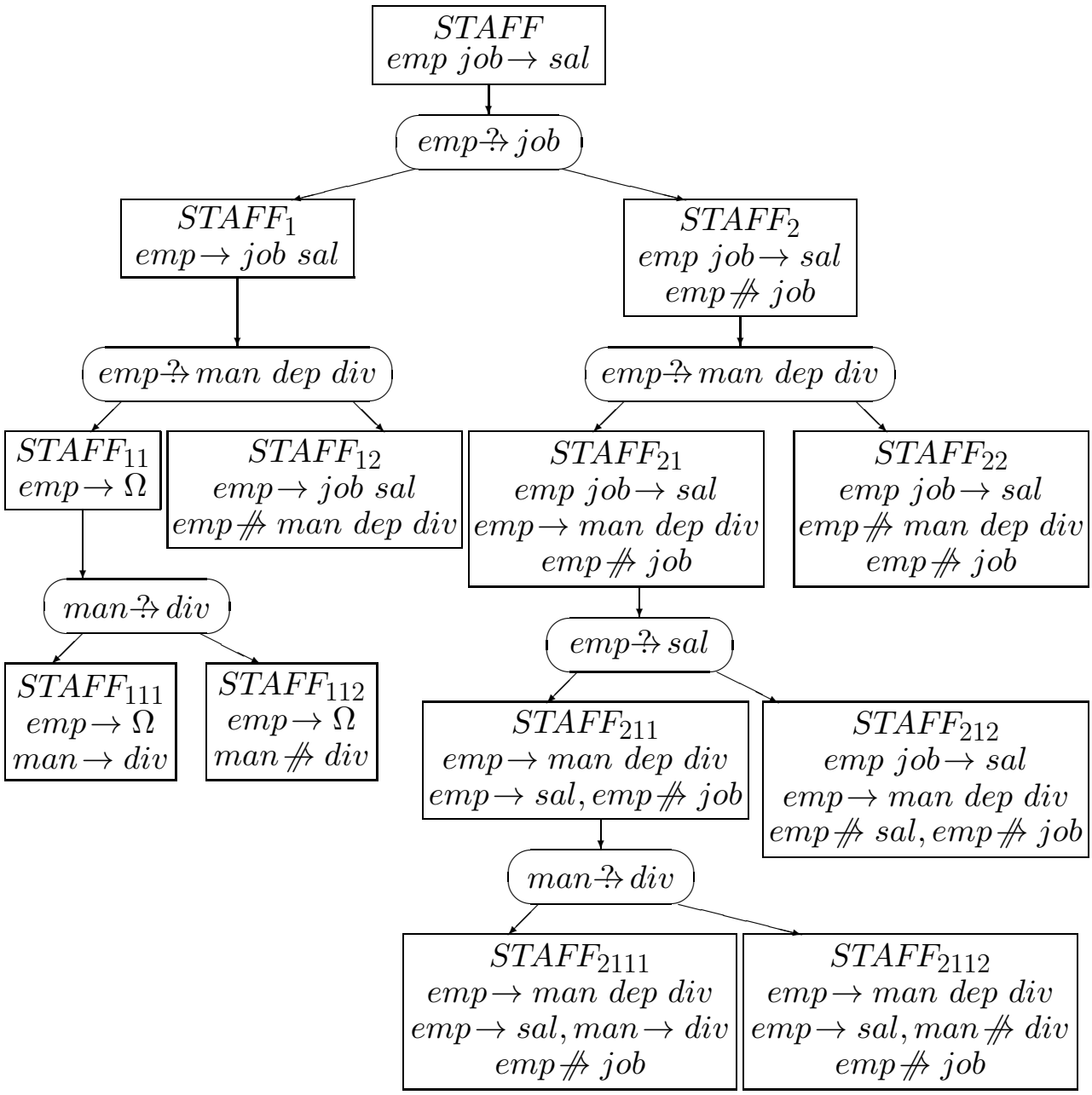


Figure 3.5: A clean decomposition tree for *STAFF* (with 5 goals).

$staff_{111}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Wallace	accountant	Brown	2000	sales	Los Angeles
Diamond	accountant	Brown	1500	sales	Los Angeles
Brown	sales manager	Goldstein	3000	sales	Los Angeles
Eltman	carrier	Kedesdy	1000	stock	Los Angeles
Kedesdy	accountant	Brown	2000	stock	Los Angeles
Carlson	chauffeur	Kedesdy	1500	stock	Los Angeles
Pike	secretary	Kedesdy	1300	stock	Los Angeles

$staff_{112}$ contains the same kind of *emp*loyees as $staff_{111}$, but whose *man*ager has more than one *div*ision for these *emp*loyees.

$staff_{112}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Pierce	accountant	Shapiro	1500	sales	Santa Barbara
Jones	chauffeur	Shapiro	1000	sales	Santa Barbara
Matthews	accountant	Shapiro	1600	sales	Bakersfield

$staff_{12}$ contains the employees with only one *job*, but with more than one *man*ager, *dep*artment or *div*ision. There are different kinds of exceptions which we cannot distinguish: Murrell has different *man*agers, but only one *dep*artment and *div*ision, whereas *Goldstein* has only one *man*ager (himself) and more than one *dep*artment and *div*ision. We could distinguish these by splitting up the goals, but we have not done so to keep the number of subrelations reasonably small.

$staff_{12}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Murrel	secretary	Wallace	1000	sales	Los Angeles
Murrel	secretary	Diamond	1000	sales	Los Angeles
Murrel	secretary	Brown	1000	sales	Los Angeles
Goldstein	gen. manager	Goldstein	4000	sales	Los Angeles
Goldstein	gen. manager	Goldstein	4000	stock	Los Angeles
Goldstein	gen. manager	Goldstein	4000	sales	Santa Barbara
Goldstein	gen. manager	Goldstein	4000	sales	Bakersfield
Goodwin	secretary	Pierce	1000	sales	Santa Barbara
Goodwin	secretary	Shapiro	1000	sales	Santa Barbara

$staff_{2111} = \emptyset$ and $staff_{2112} = \emptyset$, because $staff_{211} = \emptyset$. $staff_{21}$ contains the *emp*loyees with more than one *job*, but only one *man*ager, *dep*artment and *div*ision. There are two ways of storing the *sal*ary for these *emp*loyees: either one stores the part of the *sal*ary that belongs to a certain *job* in the tuples with that *job*, or else one stores the total *sal*ary for the *emp*loyee in all tuples with this *emp*loyee. In our instance we have taken the first possibility, since otherwise the fd $emp \rightarrow sal$ would hold, which does not belong to SC. $staff_{211}$ contains the *emp*loyees who earn exactly the same *sal*ary for each of their *job*s. In our example there are no such *emp*loyees (but there may be in general).

$staff_{212}$ contains the *emp*loyees with more than one *job*, only one *man*ager, *dep*artment and *div*ision, and more than one *sal*ary.

$staff_{212} =$

emp	*job*	*man*	*sal*	*dep*	*div*
Tyrrell	chauffeur	Shapiro	1000	sales	Bakersfield
Tyrrell	carrier	Shapiro	800	sales	Bakersfield

$staff_{22}$ contains the *emp*loyees having more than one *job* and more than one *man*ager, *dep*artment or *div*ision.

$staff_{22} =$

emp	*job*	*man*	*sal*	*dep*	*div*
Shapiro	accountant	Shapiro	1000	sales	Santa Barbara
Shapiro	sales manager	Goldstein	2000	sales	Santa Barbara
Shapiro	sales manager	Goldstein	2000	sales	Bakersfield

□

The second alternative for solving the problem with trivial decomposition steps is to define the inheritance of goals differently.

Definition 3.5 Let the scheme $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$ be decomposed according to $X \twoheadrightarrow Y \in \mathcal{G}$. A goal $T \twoheadrightarrow U \in \mathcal{G}$ is *inherited* by R_1 (resp. R_2) (i. e. is in $\mathcal{G}_1$, resp. $\mathcal{G}_2$) if $\mathcal{F}_1 \not\models T \rightarrow U$ (resp. $\mathcal{F}_2 \not\models T \rightarrow U$) and $\mathcal{F}_1 \cup \mathcal{A}_1 \not\models T \not\twoheadrightarrow U$ (resp. $\mathcal{F}_2 \cup \mathcal{A}_2 \not\models T \not\twoheadrightarrow U$).
In particular, the goal $X \twoheadrightarrow Y$ is not inherited by R_1, nor by R_2.
When considering this definition of the inheritance of goals, we speak about *inherited decomposition steps*, *inherited decompositions* and *inher-*

ited decomposition trees. When decomposing subschemes only inherited goals can be used.

□

Figure 3.6 shows an inherited decomposition step for a scheme R.

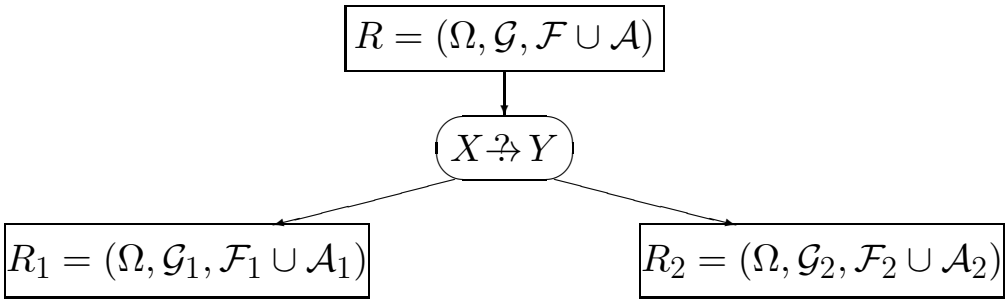


Figure 3.6: An inherited decomposition step.

Using Theorem 3.3 one can easily show the correctness of the following algorithm for the inherited decomposition of a relation, into *HNF*:

Algorithm 3.3 *Inherited Decomposition into HNF*

Input: $R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A})$.

Output: An inherited decomposition $(R_1, \ldots, R_n)$ of R

Method:

return($decompose(R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A}))$).

function $decompose(R = (\Omega, \mathcal{G}, \mathcal{F} \cup \mathcal{A}))$
begin
  for each $X \stackrel{?}{\twoheadrightarrow} Y$ **in** $\mathcal{G}$ **do**
    if $usable(X \stackrel{?}{\twoheadrightarrow} Y, \mathcal{F} \cup \mathcal{A})$
      then
        begin
          $\hat{\mathcal{A}} := \emptyset$
          for each $T \not\twoheadrightarrow U$ **in** $\mathcal{A}$ **do**
            if $\mathcal{F} \models T \rightarrow X$
              then
                $\hat{\mathcal{A}} := \hat{\mathcal{A}} \cup \{T \not\twoheadrightarrow U\}$
          $\mathcal{G}_1 := \emptyset$
          for each $P \stackrel{?}{\twoheadrightarrow} Q$ **in** $\mathcal{G}$ **do**

```
            if F ∪ {X → Y} ⊭ P → Q and
               F ∪ {X → Y} ∪ Â ⊭ P ↛ Q
               then
                  G1 := G1 ∪ {P ↛? Q}
         od
         G2 := ∅
         for each P ↛? Q in G do
            if F ⊭ P → Q and
               F ∪ Â ∪ {X ↛ Y} ⊭ P ↛ Q
               then
                  G2 := G2 ∪ {P ↛? Q}
         od
         return(decompose(R1 = (Ω, G1, F ∪ {X → Y} ∪ Â)),
                  decompose(R2 = (Ω, G2, F ∪ Â ∪ {X ↛ Y})))
         { and exit function }
      end
   od
   return(R = (Ω, G, F ∪ A))   { if no usable goal in R }
end

function usable(X ↛? Y, F ∪ A)
begin
   if F ⊨ X → Y
      then
         return(false)
   if F ∪ A ⊨ X ↛ Y
      then
         return(false)
   return(true)    { only reached if goal is usable }
end
```

□

Example 3.5 Recall the scheme R of Example 3.3. Figure 3.7 shows a clean decomposition of R into *CNF*, whereas Figure 3.8 shows an inherited decomposition of R into *HNF*. In the inherited decomposition tree the set of goals is indicated for each subrelation. (With the clean decomposition the set of goals remains the same in all subrelations). Note that in this

example the maximal number of subrelations is reached with the inherited decomposition (but this will not always be the case). □

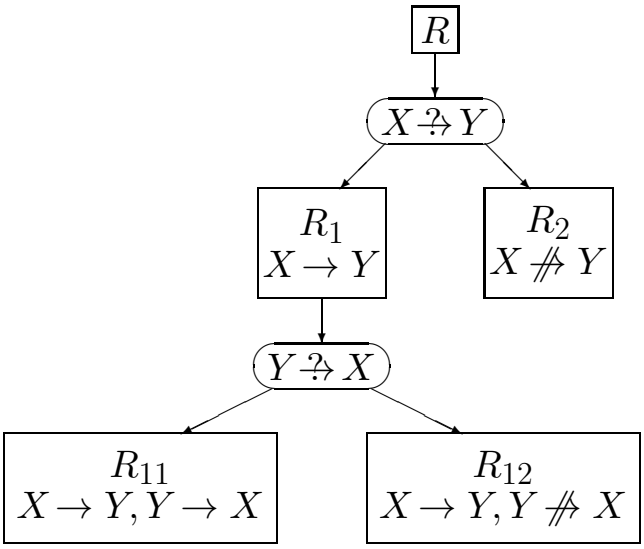


Figure 3.7: Clean decomposition for $(R, \{X \dot{\twoheadrightarrow} Y, Y \dot{\twoheadrightarrow} X\})$.

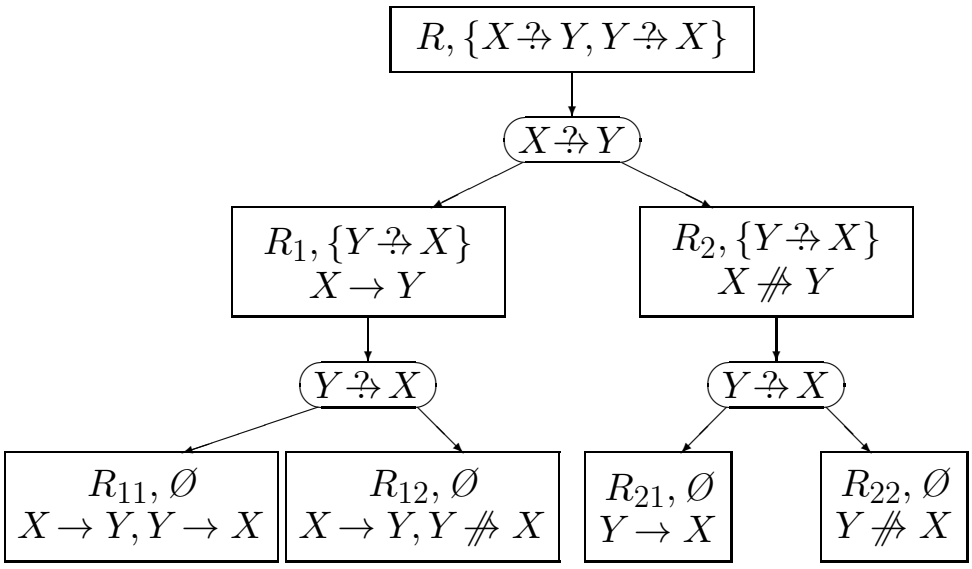


Figure 3.8: Inherited decomposition tree for $(R, \{X \dot{\twoheadrightarrow} Y, Y \dot{\twoheadrightarrow} X\})$.

Example 3.6 Recall Example 2.1 and 3.4. The decomposition trees (for the case with 2 and the case with 4 goals) are not given. The case with 2 goals produces a decomposition tree, similar to Figure 3.8, with the maximal number of 4 "final" subrelations. The decomposition tree for the 4 goals does not contain the maximal number of 16 final subrelations, but much more than the 7 subrelations of the clean decomposition into *CNF*.

The subrelations $STAFF_{12}$, $STAFF_{22}$ and $STAFF_{212}$ of Figure 3.5 are clean, but not atomic. Hence they still can be decomposed further on, if inherited decomposition steps are used instead of clean ones.

□

A third possibility for avoiding infinite decompositions is to combine the clean and inherited decomposition steps. One can easily define a *clean inherited decomposition step* (an inherited decomposition step according to a clean goal) and a *Clean Horizontal Normal Form (CHNF)*. The reader is invited to prove that this *CHNF* is equivalent to the *CNF*, or in other words, that goals which are not inherited by the schemes that result from a clean inherited decomposition step can never become clean in the remaining part of the decomposition process.

The relationship between *HNF*, *CNF*, trivial-, clean- and inherited decompositions is as follows:

- A trivial decomposition into *HNF* is also in *CNF* (if we disregard differences in the sets of goals of the final subrelations). However it may not be possible to generate it using clean decomposition steps.
- An inherited decomposition in *HNF* is also in *CNF* (disregarding goals again). It may not be possible to generate it using clean decomposition steps.
- The schemes obtained by clean decomposition steps can also be obtained using inherited decomposition steps, (disregarding goals again), but the resulting decomposition (in *CNF*) may not (yet) be in *HNF*.

Chapter 4

Decomposition with cfd's

The horizontal decomposition, based on goals, is (as far as we know) the only theoretical study on the relational database model, which uses a part of the meaning of a relation. In this chapter we describe how to generate horizontal decompositions using a new class of constraints, the *conditional-functional dependencies* (cfd's). The cfd's contain the functional dependencies as a subclass. The "goals" of Chapters 2 and 3 can be expressed as "trivial" cfd's.

This chapter covers [13]. It is divided into four parts: in Section 4.1 we define the horizontal decomposition, based on cfd's and we redefine the *Armstrong relation* and the *conflict concept* for cfd's. In Section 4.2 the implication problem is solved, for mixed cfd's and ad's. In Section 4.3 we treat the inheritance problem for cfd's and ad's. In Section 4.4 finally, a new normal form for horizontal decompositions is proposed: the *Conditional Normal Form*, and a nontrivial example illustrates the algorithm for decomposing a relation scheme into this normal form.

4.1 Conditional-Functional Dependencies

In Example 2.1 a number of goals were given, to describe that "most *emp*loyees have only one *job*, one *man*ager, one *sal*ary, one *dep*artment and one *div*ision. However, not all kinds of exceptions are possible. Figure 3.5 shows a subrelation $STAFF_{21}$ (which is still decomposed further on) for the *emp*loyees with only one *man*ager, *dep*artment and *div*ision, but with several *job*s. One can easily imagine that this situation cannot

occur in some companies. Also, if the fd $emp\ job \rightarrow sal$ is not present, one can generate a subrelation for *emp*loyees having only one *job*, *man*ager, *dep*artment and *div*ision, but several *sal*aries. This clearly is impossible in every reasonable company. The decomposition based on goals does not provide a way for avoiding the generation of such "ridiculous" subrelations, which will always be empty, although their set of constraints is not in conflict.

The reason why the horizontal decomposition sometimes generates subrelations that are always empty is that there is a relationship between the fd's, represented by the goals. The fact that *emp*loyees having only one *job*, *manager*, *dep*artment and *div*ision can have only one *sal*ary is a constraint, which cannot be expressed using fd's and ad's only (and is weaker than $emp\ job \rightarrow sal$). Its formal definition is:

Definition 4.1 Let R be a relation scheme, $X, Y, Z \subseteq \Omega$.

- A relation instance r of R satisfies the *conditional-functional dependency (cfd)* $X \rightarrow Y \supset\!\!- X \rightarrow Z$ iff in every X-complete set of tuples in r, in which the fd $X \rightarrow Y$ holds, the fd $X \rightarrow Z$ holds too.
- The scheme R satisfies $X \rightarrow Y \supset\!\!- X \rightarrow Z$ iff all the instances of R satisfy $X \rightarrow Y \supset\!\!- X \rightarrow Z$.

□

The constraint which prevents the "salary" problem is the cfd

$$emp \rightarrow job\ man\ dep\ div \supset\!\!- emp \rightarrow sal.$$

It is what we call in general a "partial implication between fd's": if the first fd holds in a (well defined) part of a relation instance, then the second fd also holds in that part of the relation instance.

The cfd's contain the fd's as a subclass. Indeed a cfd $X \rightarrow Y \supset\!\!- X \rightarrow Z$ is equivalent to the fd $X \rightarrow Z$ iff $Y \subseteq X$. In the sequel we shall usually denote fd's as fd's and not as these special cfd's, although this notation is in fact ambiguous, since there are many cfd's that are equivalent to the same fd.

Note that the cfd $X \rightarrow Y \supset\!\!- X \rightarrow Z$ implies the constraint expressed by "if $X \rightarrow Y$ holds in an instance r then $X \rightarrow Z$ also holds in r", but is stronger. (In Chapter 6 we shall see how to express the weaker constraint.)

The horizontal decomposition, based on cfd's, is defined in a similar way than for goals:

Definition 4.2 The *horizontal decomposition of a scheme R, according to the cfd $X \to Y \supset\!\!- X \to Z$*, is the ordered pair (R_1, R_2), where $R_1 = \sigma_{X \to Y}(R)$ and $R_2 = R - R_1$.

□

Note that the horizontal decomposition of a scheme, according to $X \to Y \supset\!\!- X \to Z$ does not depend on the set Z. However, since for every instance r of R, $\sigma_{X \to Y}(r)$ is an X-complete set of tuples of r in which $X \to Y$ holds, $X \to Z$ also holds in R_1. Note also that $R_2 = \sigma_{X \not\twoheadrightarrow Y}(R)$.

From now on we shall assume that the set SC of constraints of a relation scheme R consists of a set $\mathcal{C}$ of cfd's and a set $\mathcal{A}$ of ad's.

When $Z \subseteq XY$ the cfd $X \to Y \supset\!\!- X \to Z$ holds in every instance (since the "right" fd is a consequence of the "left" fd). However, including trivial cfd's in $\mathcal{C}$ is not useless: a trivial cfd can lead to nontrivial horizontal decompositions. Using a trivial cfd $X \to Y \supset\!\!- X \to Z$ is equivalent to the decomposition according to the goal $X \not\twoheadrightarrow Y$. Hence the horizontal decompositions of Chapters 2 and 3 can be "simulated" by the decompositions described in the present chapter.

The calculation of the constraints that hold in the selections of R is described in Section 4.3. The main difference with the inheritance problem of Section 3.2 is that both cfd's and ad's (which hold in R) may not hold in some selections of R.

Since fd's are "special" cfd's the conflict-problem of fd's and ad's also occurs with cfd's and ad's. We give a conflict-detection algorithm for cfd's and ad's below, but to prove its correctness we first need the definition (and existence) of Armstrong relations for fd's, considering the presence of cfd's.

Definition 4.3 Let $\mathcal{C}$ be a set of cfd's over a set Ω of attributes. A *strong Armstrong relation for $\mathcal{C}$* is a strong Armstrong relation for the set of all fd's that are a consequence of $\mathcal{C}$.

□

To show a construction of (strong) Armstrong relations for fd's, considering the presence of cfd's, we first define a special set of fd's:

Definition 4.4 $FSAT_{\mathcal{C}}(X, Y)$ is the smallest possible set of fd's, such that:

1. $X \to Y \in FSAT_{\mathcal{C}}(X, Y)$.
2. If $T \to U \in FSAT_{\mathcal{C}}(X, Y)$ and $T \to U \supset\!- T \to V \in \mathcal{C}$ then $T \to V \in FSAT_{\mathcal{C}}(X, Y)$.
3. If $FSAT_{\mathcal{C}}(X, Y) \models T \to V$ then $T \to V \in FSAT_{\mathcal{C}}(X, Y)$.

□

$FSAT_{\mathcal{C}}(X, Y)$ can be constructed starting from $\{X \to Y\}$ by repeatedly trying to satisfy 2 and 3 of the definition.

Lemma 4.1 *Let $\mathcal{C}$ be a set of cfd's over Ω, $T, V, X, Y \subseteq \Omega$. $T \to V \in FSAT_{\mathcal{C}}(X, Y)$ iff $\mathcal{C} \cup \{X \to Y\} \models T \to V$.*

Proof For $X \to Y$ it is obvious that $\mathcal{C} \cup \{X \to Y\} \models X \to Y$. For other fd's we prove the lemma by induction.

- Suppose $T \to V$ is added by trying to satisfy part 2 of the definition. Then (by induction) there is a cfd $T \to U \supset\!- T \to V \in \mathcal{C}$ for which $\mathcal{C} \cup \{X \to Y\} \models T \to U$. As remarked earlier, $T \to U$ and $T \to U \supset\!- T \to V$ imply $T \to V$.
- Suppose $T \to V$ is added by trying to satisfy part 3 of the definition. Then it is a consequence of a number of fd's for which the lemma already holds (by induction hypothesis). Hence $T \to V$ also is a consequence of $\mathcal{C} \cup \{X \to Y\}$.

This proves that all the fd's of $FSAT_{\mathcal{C}}(X, Y)$ are consequences of $\mathcal{C} \cup \{X \to Y\}$. To prove that the opposite inclusion also holds, consider $Arm(FSAT_{\mathcal{C}}(X, Y))$. In $Arm(FSAT_{\mathcal{C}}(X, Y))$ all the fd's of $FSAT_{\mathcal{C}}(X, Y)$ hold, and (since $FSAT_{\mathcal{C}}(X, Y)^* = FSAT_{\mathcal{C}}(X, Y)$ by part 3 of the definition) no other fd's hold. Suppose some cfd $T \to U \supset\!- T \to V$ of $\mathcal{C}$ does not hold in $Arm(FSAT_{\mathcal{C}}(X, Y))$. Then there exists a T-complete set of tuples in $Arm(FSAT_{\mathcal{C}}(X, Y))$ in which $T \to U$ holds and in which $T \to V$ does not hold. By Theorem 2.1 this means that $FSAT_{\mathcal{C}}(X, Y) \models T \to U$, and hence $T \to U \in FSAT_{\mathcal{C}}(X, Y)$ (by part 3 of the definition). By part 2 of the definition this implies that $T \to V \in FSAT_{\mathcal{C}}(X, Y)$ and hence $T \to V$ must hold in $Arm(FSAT_{\mathcal{C}}(X, Y))$, a contradiction.

□

From Theorem 2.1 and the above lemma one can easily deduce that

Theorem 4.1 *Let $\mathcal{C}$ be a set of cfd's. $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$ is a strong Armstrong relation for $\mathcal{C}$. In other words, for all $T, U \subseteq \Omega$ holds that $\mathcal{C} \models T \rightarrow U$ iff $T \rightarrow U \in FSAT_{\mathcal{C}}(\emptyset, \emptyset)$.*

□

Note that the Armstrong relation for $\mathcal{C}$ is not an Armstrong relation for cfd's, only for fd's.

Definition 4.4 and Lemma 4.1 have been given for $FSAT_{\mathcal{C}}(X, Y)$ with arbitrary X and Y, since this notation will be useful in the following sections. It is not necessary to do so, since $FSAT_{\mathcal{C}}(X, Y)$ and $FSAT_{\mathcal{C} \cup \{X \rightarrow Y\}}(\emptyset, \emptyset)$ are equal. However, the second notation does not indicate that in the set $\mathcal{C} \cup \{X \rightarrow Y\}$ the fd $X \rightarrow Y$ is "special".

Note also that $FSAT_{\mathcal{C}}(X, Y)$ is equal to $FSAT_{\mathcal{C}}(\emptyset, \emptyset)$ if $Y \subseteq X$.

With the Armstrong relation for $\mathcal{C}$, we can prove the conflict detection algorithm which is based on the following theorem:

Theorem 4.2 *$\mathcal{C} \cup \mathcal{A}$ is in conflict iff for some ad $X \not\Rightarrow Y$ of $\mathcal{A}$, $\mathcal{C} \models X \rightarrow Y$ holds.*

Proof The if-part is trivial.
For the only-if-part, suppose that $\mathcal{C} \cup \mathcal{A}$ is in conflict. Hence in the instance $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$, in which $\mathcal{C}$ holds (cfr. the proof of Lemma 4.1), some ad $X \not\Rightarrow Y$ of $\mathcal{A}$ does not hold. By Theorem 2.1 this implies that $X \rightarrow Y$ holds in $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$. Hence $X \rightarrow Y \in Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$. Theorem 4.1 then implies that $\mathcal{C} \models X \rightarrow Y$.

□

Algorithm 4.1 *Conflict Detection*

Input: $\mathcal{C}, \mathcal{A}$, a set of cfd's and a set of ad's.

Output: *true* or *false*.

Method:
for each $T \not\Rightarrow U$ **in** $\mathcal{A}$ **do**
  if $\mathcal{C} \models T \rightarrow U$
    then
      return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

In Algorithm 4.1 we did not explain how to verify whether $\mathcal{C} \models T \to U$. This means verifying whether $T \to U \in FSAT_{\mathcal{C}}(\emptyset, \emptyset)$, and we know how to construct $FSAT_{\mathcal{C}}(\emptyset, \emptyset)$. However, this procedure is very inefficient, since calculating $FSAT_{\mathcal{C}}(\emptyset, \emptyset)$ implies calculating the closure of sets of fd's, which takes much time because the closure of a set of fd's has exponentially more elements than the original set. However, when we explain the membership algorithms for mixed cfd's and ad's in the next section, we shall give an efficient algorithm for verifying $T \to U \in FSAT_{\mathcal{C}}(\emptyset, \emptyset)$, without actually calculating closures of sets of fd's.

4.2 The Implication Problem for cfd's and ad's

The implication problem for fd's has been studied exhaustively in literature [2, 3]. In the previous chapter we have reduced the implication problem for fd's and ad's to the implication problem for fd's only. In this section we shall do the same for cfd's and ad's.

We propose the following set of inference rules for cfd's:

$(C1)$: if $Z \subseteq XY$ or $XY \to Z$ then $X \to Y \supset\!\!- X \to Z$.

$(C2)$: if $X \to Y \supset\!\!- X \to Z$ and $X \to Y \supset\!\!- X \to T$ then $X \to Y \supset\!\!- X \to ZT$.

$(C3)$: if $X \to Y \supset\!\!- X \to Z$ and $Z \to T$ then $X \to Y \supset\!\!- X \to T$.

$(C4)$: if $X \to Y \supset\!\!- X \to Z$ and $X \to Z \supset\!\!- X \to T$ then $X \to Y \supset\!\!- X \to T$.

$(C5)$: if $X \to Y \supset\!\!- X \to Z$ and $W \to Y \supset\!\!- W \to X$ and $X \to W$ then $W \to Y \supset\!\!- W \to Z$.

□

As fd's are special cfd's the use of fd's in these rules is allowed. The classical inference rules for fd's (reflexivity, augmentation and transitivity) can be deduced from $C1 \ldots C4$ as follows:

Lemma 4.2 *The rules $C1 \ldots C4$ are complete for fd's.*

Proof We first prove that all representations of the same fd are equivalent: let $Y \subseteq X$ and $Y' \subseteq X$. $X \to Y' \supset\!\!- X \to Y$ holds by $C1$. With $X \to Y \supset\!\!- X \to Z$ it induces $X \to Y' \supset\!\!- X \to Z$ by $C4$. So all $X \to Y \supset\!\!- X \to Z$ with $Y \subseteq X$ are equivalent.

We now show how to prove the inference rules for fd's from $C1 \ldots C5$.

$F1:$ Let $Y \subseteq X$. Then $X \to X \supset\!\!- X \to Y$ holds by $C1$.

$F2:$ Let $X \to Y$ and $W \subseteq V$. $XV \to XV \supset\!\!- XV \to X$ holds by $C1$. With $X \to Y$ it induces $XV \to XV \supset\!\!- XV \to Y$ by $C3$. $XV \to Y \supset\!\!- XV \to YW$ also holds by $C1$. The latter two cfd's together induce $XV \to XV \supset\!\!- XV \to YW$ by $C4$.

$F3:$ Let $X \to Y$ and $Y \to Z$. $X \to X \supset\!\!- X \to Y$ and $Y \to Z$ induce $X \to X \supset\!\!- X \to Z$ by $C3$.

□

Note that for the inference of fd's (represented as special cfd's) we do not need rule $C5$.

Theorem 4.3 *The rules $C1 \ldots C5$ are sound.*

Proof For $C1 \ldots C4$ this is very easy, and left to the reader. We only prove $C5$:

Let S be an arbitrary W-complete set of tuples. Since $X \to W$ holds, S is also X-complete (see Remark 2.1). If $W \to Y$ holds in S then so does $X \to Y$ by transitivity on $X \to W$ and $W \to Y$. $X \to Y$ in S induces $X \to Z$ in S and $W \to Y$ in S induces $W \to X$ in S. By transitivity on $W \to X$ and $X \to Z$, $W \to Z$ holds in S.

□

The proof of the completeness of $C1 \ldots C5$ for cfd's relies on some special properties of $FSAT_{\mathcal{C}}(X, Y)$, which we describe first.

Lemma 4.3 *If $T \to V \in FSAT_{\mathcal{C}}(X, Y)$ then $\mathcal{C} \vdash T \to V$ or $\mathcal{C} \vdash T \to X$.*

Proof For $T \to V = X \to Y$ this property is trivial. We prove that the property remains valid during the construction of $FSAT_{\mathcal{C}}(X, Y)$ by repeatedly trying to satisfy parts 2 and 3 of Definition 4.4.

Let $\mathcal{C} = \{X_i \to Y_i \supset\!\!- X_i \to Z_i : i = 1 \ldots n\}$.

- Let part 2 be the reason that $T \to V \in FSAT_{\mathcal{C}}(X, Y)$. Then for some i, $T = X_i$ and $V = Z_i$ and by induction $\mathcal{C} \vdash X_i \to Y_i$ or $\mathcal{C} \vdash X_i \to X$. If $\mathcal{C} \vdash X_i \to Y_i$ (the other case being trivial) we have $\mathcal{C} \vdash X_i \to X_i \supset\!\!- X_i \to Y_i$ (equivalent to $\mathcal{C} \vdash X_i \to Y_i$), $\mathcal{C} \vdash X_i \to Y_i \supset\!\!- X_i \to Z_i$ (since $X_i \to Y_i \supset\!\!- X_i \to Z_i \in \mathcal{C}$) and hence by $C4$ $\mathcal{C} \vdash X_i \to X_i \supset\!\!- X_i \to Z_i$, i. e. $\mathcal{C} \vdash X_i \to Z_i$.

- Let part 3 be the reason that $T \to V \in FSAT_{\mathcal{C}}(X, Y)$. Then there are three possibilities:
  1. If $V \subseteq T$, then, by $C1$ $\mathcal{C} \vdash T \to T \supset\!\!- T \to V$ i. e. $\mathcal{C} \vdash T \to V$.
  2. If $T = T_1T_2$ and $V = V_1V_2$ with $T_1 \to V_1 \in FSAT_{\mathcal{C}}(X, Y)$ and $V_2 \subseteq T_2$ we have that $\mathcal{C} \vdash T_1 \to V_1$ or $\mathcal{C} \vdash T_1 \to X$.
     - If $\mathcal{C} \vdash T_1 \to V_1$ then $\mathcal{C} \vdash T \to V$ by augmentation.
     - If $\mathcal{C} \vdash T_1 \to X$ then $\mathcal{C} \vdash T \to X$, also by augmentation.
  3. If $T \to U$ and $U \to V \in FSAT_{\mathcal{C}}(X, Y)$ we have ($\mathcal{C} \vdash T \to U$ or $\mathcal{C} \vdash T \to X$) and ($\mathcal{C} \vdash U \to V$ or $\mathcal{C} \vdash U \to X$). Using transitivity we obtain $\mathcal{C} \vdash T \to V$ or $\mathcal{C} \vdash T \to X$.

The proof is completed by remarking that reflexivity, augmentation and transitivity are complete for fd's.

□

Lemma 4.4 *If* $T \to V \in FSAT_{\mathcal{C}}(X, Y)$ *then* $\mathcal{C} \cup \{X \to Y \supset\!\!- X \to T\} \vdash X \to Y \supset\!\!- X \to V$.

Proof For $T \to V = X \to Y$ this property is trivial (by $C1$). As in Lemma 4.3 we prove that the property remains valid during the construction of $FSAT_{\mathcal{C}}(X, Y)$ by repeatedly trying to satisfy parts 2 and 3 of Definition 4.4.

Let $\mathcal{C} = \{X_i \to Y_i \supset\!\!- X_i \to Z_i : i = 1 \ldots n\}$.

- Let part 2 be the reason that $T \to V \in FSAT_{\mathcal{C}}(X, Y)$. Then for some i, $T = X_i$ and $V = Y_i$ and by induction $\mathcal{C} \cup \{X \to Y \supset\!\!- X \to X_i\} \vdash X \to Y \supset\!\!- X \to Y_i$. By Lemma 4.3 we have that $\mathcal{C} \vdash X_i \to Z_i$ or $\mathcal{C} \vdash X_i \to X$.
  1. If $\mathcal{C} \vdash X_i \to Z_i$ then $X \to Y \supset\!\!- X \to X_i$ and $X_i \to Z_i$ induce $X \to Y \supset\!\!- X \to Z_i$ by $C3$.
  2. If $\mathcal{C} \vdash X_i \to X$ then we have that $X \to Y \supset\!\!- X \to Y$ holds by $C1$, $X \to Y \supset\!\!- X \to X_i$ induces $X \to Y \supset\!\!- X \to Y_i$ by induction, hence $X \to Y \supset\!\!- X \to YY_i$ holds by $C2$.
     $X \to YY_i \supset\!\!- X \to Y$ (holding by $C1$) and $X \to Y \supset\!\!- X \to X_i$ induce $X \to YY_i \supset\!\!- X \to X_i$ by $C4$, and $X_i \to YY_i \supset\!\!- X_i \to Y_i$ (holding by $C1$) and $X_i \to Y_i \supset\!\!- X_i \to Z_i$ induce $X_i \to YY_i \supset\!\!- X_i \to Z_i$ by $C4$.

$X_i \to YY_i \supset\!\!- X_i \to Z_i$ and $X \to YY_i \supset\!\!- X \to X_i$ and $X_i \to X$ induce $X \to YY_i \supset\!\!- X \to Z_i$ by $C5$, and finally $X \to Y \supset\!\!- X \to YY_i$ and $X \to YY_i \supset\!\!- X \to Z_i$ induce $X \to Y \supset\!\!- X \to Z_i$ by $C4$.

- Let part 3 be the reason that $T \to V \in FSAT_{\mathcal{C}}(X, Y)$. Then there are three possibilities:
  1. If $V \subseteq T$, then $X \to Y \supset\!\!- X \to T$ and $T \to V$ (reflexivity) induce $X \to Y \supset\!\!- X \to V$ by $C3$.
  2. Suppose $T = T_1T_2$ and $V = V_1V_2$ with $T_1 \to V_1 \in FSAT_{\mathcal{C}}(X, Y)$ and $V_2 \subseteq T_2$.
     $X \to Y \supset\!\!- X \to T$ and $T \to T_1$ (reflexivity) induce $X \to Y \supset\!\!- X \to T_1$ by $C3$, and $\mathcal{C} \cup \{X \to Y \supset\!\!- X \to T_1\} \vdash X \to Y \supset\!\!- X \to V_1$ by induction.
     $X \to Y \supset\!\!- X \to T$ and $T \to T_2$ (reflexivity) induce $X \to Y \supset\!\!- X \to T_2$ by $C3$, and $X \to Y \supset\!\!- X \to T_2$ and $T_2 \to V_2$ (reflexivity) induce $X \to Y \supset\!\!- X \to V_2$ by $C3$.
     Finally $X \to Y \supset\!\!- X \to V_1$ and $X \to Y \supset\!\!- X \to V_2$ induce $X \to Y \supset\!\!- X \to V$ by $C2$.
  3. If $T \to U$ and $U \to V \in FSAT_{\mathcal{C}}(X, Y)$ we have that $\mathcal{C} \cup \{X \to Y \supset\!\!- X \to T\} \vdash X \to Y \supset\!\!- X \to U$ and $\mathcal{C} \cup \{X \to Y \supset\!\!- X \to U\} \vdash X \to Y \supset\!\!- X \to V$ by induction, hence $\mathcal{C} \cup \{X \to Y \supset\!\!- X \to T\} \vdash X \to Y \supset\!\!- X \to V$ by combining both "derivations".

□

Using these two lemmas the link between the membership problem for cfd's and the set $FSAT_{\mathcal{C}}(X, Y)$ can be easily established:

Theorem 4.4 *$\mathcal{C} \models X \to Y \supset\!\!- X \to Z$ iff $X \to Z \in FSAT_{\mathcal{C}}(X, Y)$.*

Proof Let $\mathcal{C} = \{X_i \to Y_i \supset\!\!- X_i \to Z_i \mid i = 1 \ldots n\}$.
If $X \to Z \in FSAT_{\mathcal{C}}(X, Y)$ then $\mathcal{C} \vdash X \to Y \supset\!\!- X \to Z$ by Lemma 4.4 (since $X \to Y \supset\!\!- X \to X$ is trivial). Hence $\mathcal{C} \models X \to Y \supset\!\!- X \to Z$ because $C1 \ldots C5$ are sound.

Conversely, if $X \to Z \notin FSAT_{\mathcal{C}}(X, Y)$, consider $Arm(FSAT_{\mathcal{C}}(X, Y))$. In $Arm(FSAT_{\mathcal{C}}(X, Y))$, $X \to Z$ does not hold, $X \to Y$ holds and $\mathcal{C}$ holds too because for every i we have:

- if $X_i \to Y_i \in FSAT_{\mathcal{C}}(X, Y)$ then $X_i \to Z_i \in FSAT_{\mathcal{C}}(X, Y)$, and hence both fd's hold in $Arm(FSAT_{\mathcal{C}}(X, Y))$, and

- if $X_i \to Y_i \notin FSAT_{\mathcal{C}}(X,Y)$ then $X_i \not\twoheadrightarrow Y_i$ holds, so $X_i \to Y_i \supset\!\!- X_i \to Z_i$ holds trivially.

Since $X \to Y \supset\!\!- X \to Z$ does not hold in $Arm(FSAT_{\mathcal{C}}(X,Y))$ $\mathcal{C} \not\models X \to Y \supset\!\!- X \to Z$.

□

From the proof of the above theorem immediately follows:

Corollary 4.1 *$C1 \ldots C5$ are complete for cfd's.*

□

Theorem 4.4 shows that the membership problem for cfd's is very closely related to that of fd's. In fact one can imagine a cfd as an ad-hoc implication between two fd's. The construction of $FSAT_{\mathcal{C}}(X,Y)$ indeed consists of generating the "closure" of a set of fd's, using the normal inference rules (in part 3 of Definition 4.4) and some "additional" rules (the cfd's in part 2). It is not obvious that cfd's should behave this way, since the cfd $X \to Y \supset\!\!- X \to Z$ is a stronger constraint than the expression that "if the fd $X \to Y$ holds (in the entire relation) then also $X \to Z$ holds".

From Theorem 4.4 one can deduce a membership algorithm which simply calculates $FSAT_{\mathcal{C}}(X,Y)$ and verifies whether $X \to Z$ is an element of this set of fd's. However, such an algorithm would take exponential time, since it has to calculate the closure of a set of fd's.

We propose a polynomial time membership algorithm for cfd's.

Algorithm 4.2 *Membership detection for cfd's.*

Input: $\mathcal{C} = \{X_i \to Y_i \supset\!\!- X_i \to Z_i \mid i = 1 \ldots n\}$; $X_0 \to Y_0 \supset\!\!- X_0 \to Z_0$.
Output: *true* or *false*. (meaning $\mathcal{C} \models X_0 \to Y_0 \supset\!\!- X_0 \to Z_0$ or not.)
Method:

var $P_0, P_1, \ldots, P_n$: set of attributes
change : boolean
i, j : integer

begin
$P_0 := X_0 Y_0$
for $i := 1$ **to** n **do**
$P_i := X_i \quad \{A\}$
od

```
    repeat
        change := false
        for j := 0 to n do
          for i := 0 to n do
              if X_j ⊆ P_i
                then
                  if P_j ⊄ P_i    {B}
                    then
                      begin
                        P_i := P_i ∪ P_j
                        change := true
                      end
          od
        od
        for i := 1 to n do
          if Y_i ⊆ P_i
              then
                if Z_i ⊄ P_i    {C}
                  then
                    begin
                      P_i := P_i ∪ Z_i
                      change := true
                    end
        od
    until change = false
    if Z_0 ⊆ P_0
        then
          return(true)
        else
          return(false)
end
```

□

Theorem 4.5 *Algorithm 4.2 correctly detects whether* $\mathcal{C} \models X_0 \rightarrow Y_0 \supset\!\!- X_0 \rightarrow Z_0$ *or not.*

Proof The algorithm always stops since only a finite number of times at least one attribute can be added to any of the P_i.

1. From test {B} it is obvious that for $0 \leq i, j \leq n$ holds that if $X_j \subseteq P_i$ then $P_j \subseteq P_i$ too.
2. From test {C} in the algorithm it is obvious that for $1 \leq i \leq n$ holds that if $Y_i \subseteq P_i$ then $Z_i \subseteq P_i$ too.

We prove the algorithm, using Theorem 4.4. We first show that $X_i \rightarrow P_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$ for $i = 0 \ldots n$.

- At point {A} this is certainly true since at that point $P_0 = X_0Y_0$ and $P_i = X_i$, $i = 1 \ldots n$. Only when the test at {B} or {C} turns out true some P_i may change.
- Suppose $X_i \rightarrow P_i$ and $X_j \rightarrow P_j$ both belong to $FSAT_{\mathcal{C}}(X_0, Y_0)$. If $X_j \subseteq P_i$ then $P_i \rightarrow X_j$ is trivial and hence belongs to $FSAT_{\mathcal{C}}(X_0, Y_0)$. Hence $X_i \rightarrow P_i$, $P_i \rightarrow X_j$, $X_j \rightarrow P_j$ and by transitivity $X_i \rightarrow P_j$ is in $FSAT_{\mathcal{C}}(X_0, Y_0)$.
  $X_i \rightarrow P_i$ and $X_i \rightarrow P_j$ induce that $X_i \rightarrow P_i \cup P_j \in FSAT_{\mathcal{C}}(X_0, Y_0)$ too, which proves that the modification to P_i at {B} does not violate the property that $X_i \rightarrow P_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$.
- Suppose $X_i \rightarrow P_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$ and $Y_i \subseteq P_i$ for $i = 1 \ldots n$. Then also $X_i \rightarrow Y_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$. Because of the construction of $FSAT_{\mathcal{C}}(X_0, Y_0)$ we have $X_i \rightarrow Z_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$. Hence by augmentation $X_i \rightarrow P_i \cup Z_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$ too.
  This proves that the modification to P_i at point {C} does not violate the property that $X_i \rightarrow P_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$.

The converse of this property must hold too: we prove that if $X_i \rightarrow T \in FSAT_{\mathcal{C}}(X_0, Y_0)$ then $T \subseteq P_i$, for the final P_i, $i = 1 \ldots n$, by showing that this property remains valid during the construction of $FSAT_{\mathcal{C}}(X_0, Y_0)$. We denote the final value of P_i by P_i^e.

- Initially only $X_0 \rightarrow Y_0 \in FSAT_{\mathcal{C}}(X_0, Y_0)$, and $Y_0 \subseteq X_0Y_0 = P_0 \subseteq P_0^e$.
- If $X_i \rightarrow Y_i \in FSAT_{\mathcal{C}}(X_0, Y_0)$ and $Y_i \subseteq P_i$, for $i \neq 0$, then $X_i \rightarrow Z_i$ is added to $FSAT_{\mathcal{C}}(X_0, Y_0)$ and Z_i is added to P_i at point {C}. Hence $Z_i \subseteq P_i^e$ and hence the second point of the definition of $FSAT_{\mathcal{C}}(X_0, Y_0)$ does not cause a violation of the property.

- The third point of the definition of $FSAT_{\mathcal{C}}(X_0, Y_0)$ means calculating the closure of $FSAT_{\mathcal{C}}(X_0, Y_0)$. Let us first consider sets of attributes that are an X_i for some i. Afterwards we show that it is no restriction to neglect the other sets of attributes.
  - If $X_i \rightarrow T$ is trivial, then $T \subseteq X_i \subseteq P_i^e$.
  - Suppose $X_i \rightarrow T$ is obtained by augmenting an fd $X_j \rightarrow S \in FSAT_{\mathcal{C}}(X_0, Y_0)$ for which $S \subseteq P_j^e$. Then $X_i = X_j U$ and $T = SV$ with $V \subseteq U$, hence $P_j^e \subseteq P_i^e$. We have $S \subseteq P_j^e \subseteq P_i^e$ and $V \subseteq X_i \subseteq P_i^e$, hence $T = SV \subseteq P_i^e$.
  - Suppose $X_i \rightarrow T$ is obtained by transitivity on $X_i \rightarrow X_j$ and $X_j \rightarrow T$ with $X_j \subseteq P_i^e$ and $T \subseteq P_j^e$. Then $T \subseteq P_j^e \subseteq P_i^e$.

  If there were no other sets of attributes than the X_i's, the proof would be complete. We prove that it makes no difference for the P_i^e whether all set of attributes are X_i's or not.

  Suppose X_{n+1} is a set or attributes, different from all X_i. Let $\mathcal{C}' = \mathcal{C} \cup \{X_{n+1} \rightarrow X_{n+1} \supset\!\!- X_{n+1} \rightarrow X_{n+1}\}$. We claim that Algorithm 4.2 produces the same sets $P_0 \ldots P_n$ for $\mathcal{C}'$ as for $\mathcal{C}$.

  The cfd $X_{n+1} \rightarrow X_{n+1} \supset\!\!- X_{n+1} \rightarrow X_{n+1}$ does not change P_{n+1} at point {C}. But if there is an X_j with $X_j \subseteq P_{n+1}$ then P_j is added to P_{n+1}. If later on $X_{n+1} \subseteq P_i$ for some i then P_{n+1} is added to P_i at point {B}. However, this change is exactly the same as the change caused at {B} for $X_j \subseteq P_i$ since the attributes added to P_i (at {B} for $X_{n+1} \subseteq P_i$) all are in P_j. Hence introducing X_{n+1} and P_{n+1} does not affect $P_0^e \ldots P_n^e$. Therefore not all sets of attributes have to be an X_i for some i.

  □

Let n be the number of cfd's of $\mathcal{C}$ and r the number of attributes of Ω. The reader is invited to prove that the time-complexity of Algorithm 4.2 is $O(n^3 r^2)$. The time-complexity of Algorithm 4.1 then becomes $O(n^3 r^2 m)$ where m is the number of ad's in $\mathcal{A}$. This probably is not the best time-complexity one can achieve, since better algorithms exist for the "fd-part" of Algorithm 4.2 already [2].

We now present a set of inference rules for mixed cfd's and ad's. The rules are not complete for the inference of cfd's and ad's, but are only given to show some implications between cfd's and ad's:

$(CA1)$: if $X \to Y$, $X \to Z \supset\!\!- X \to T$ and $X \not\Rightarrow T$ then $Y \not\Rightarrow Z$.
$(CA2)$: if $X \not\Rightarrow Y$ then $X \to Y \supset\!\!- X \to Z$ for all $Z \subseteq \Omega$.

□

Lemma 4.5 *The rules $CA1$ and $CA2$ are sound.*

Proof We only prove $CA1$. (The other rule is rather trivial.)
Consider an arbitrary Y-complete, Y-unique set s of tuples (in an instance, satisfying $X \to Y$, $X \to Z \supset\!\!- X \to T$ and $X \not\Rightarrow T$). Because of $X \to Y$ s is also X-complete. Since $X \not\Rightarrow T$ holds in this X-complete set of tuples, $X \not\Rightarrow Z$ must hold too (otherwise $X \to Z \supset\!\!- X \to T$ would be violated). Hence (by $X \not\Rightarrow Z$) in s there are several Z-values, for the only one Y-value, which means that $Y \not\Rightarrow Z$ holds in s.

□

Since fd's are special cfd's prove the rules for fd's and ad's ($A1$, $FA1$ and $FA2$) from $C1, \ldots, C5$ and $CA1$ as an exercise:

Remark 4.1 *$C1, \ldots, C4$ and $CA1$ are complete for fd's and ad's.*

Proof Since we already know that $C1 \ldots C4$ are complete for fd's, we only have to prove $A1$, $FA1$ and $FA2$.

$A1$: Let $XV \not\Rightarrow YW$ and $W \subseteq V$. $XV \to X$ (reflexivity), $XV \to Y \supset\!\!- XV \to YW$ ($C1$) and $XV \not\Rightarrow YW$ induce $X \not\Rightarrow Y$ by $CA1$.

$FA1$: Let $X \to Y$ and $X \not\Rightarrow Z$. $X \to Y$, $X \to Z \supset\!\!- X \to Z$ ($C1$) and $X \not\Rightarrow Z$ induce $Y \not\Rightarrow Z$ by $CA1$.

$FA2$: Let $Y \to Z$ and $X \not\Rightarrow Z$. $X \to Y \supset\!\!- X \to Y$ ($C1$) and $Y \to Z$ induce $X \to Y \supset\!\!- X \to Z$ by $C3$. $X \to X$ (reflexivity), $X \to Y \supset\!\!- X \to Z$ and $X \not\Rightarrow Z$ induce $X \not\Rightarrow Y$ by $CA1$.

□

Note that rules $C5$ and $CA2$ are not needed for the inference of fd's and ad's only.

We now show a solution to the implication problem for cfd's and ad's, and then we give an example which explains that $C1, \ldots, C5, CA1$ and $CA2$ are not complete.

Theorem 4.6 *Let $\mathcal{C} \cup \mathcal{A}$ be not in conflict, $X, Y \subseteq \Omega$. Then $\mathcal{C} \cup \mathcal{A} \models X \not\Rightarrow Y$ iff $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\}$ is in conflict.*

Proof The only-if-part is trivial.

For the if-part, suppose that $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\}$ is in conflict. Then, by Theorem 4.2 $\mathcal{C} \cup \{X \to Y\} \models T \to U$ for some $T \not\to U \in \mathcal{A}$. We prove that $\mathcal{C} \cup \{T \not\to U\} \not\models X \not\to Y$ is impossible.

Without loss of generality we may assume that $\mathcal{C}$ is minimal for the property $\mathcal{C} \cup \{X \to Y\} \models T \to U$, i. e. all cfd's of $\mathcal{C}$ are used during the construction of $FSAT_{\mathcal{C} \cup \{X \to Y\}}(T, T) = FSAT_{\mathcal{C}}(X, Y)$. "Being used" means that the cfd $X_i \to Y_i \supset\!\!- X_i \to Z_i$ satisfies the condition of part 2 of Definition 4.4. Hence both $X_i \to Y_i$ and $X_i \to Z_i$ are in $FSAT_{\mathcal{C}}(X, Y)$ for all $X_i \to Y_i \supset\!\!- X_i \to Z_i \in \mathcal{C}$. By Lemma 4.3 this implies that $\mathcal{C} \models X_i \to Y_i$ or $\mathcal{C} \models X_i \to X$.

If $\mathcal{C} \cup \{T \not\to U\} \not\models X \not\to Y$ then there exists an instance r in which $\mathcal{C} \cup \{T \not\to U\}$ holds, but in which $X \not\to Y$ does not hold.
Consider $r' = \sigma_{X \to Y}(r)$.

- If for $X_i \to Y_i \supset\!\!- X_i \to Z_i \in \mathcal{C}$, $X_i \to Y_i$ (and hence also $X_i \to Z_i$) holds in r then $X_i \to Y_i$ and $X_i \to Z_i$ still hold in r' since fd's cannot be violated by taking a selection of r.
- If for $X_i \to Y_i \supset\!\!- X_i \to Z_i \in \mathcal{C}$, $X_i \to X$ holds in r, then all X_i-values correspond to one X-value (each). Hence, in r' all the tuples of r with some X_i-projection are included, or none are. Hence $X_i \to Y_i \supset\!\!- X_i \to Z_i$ still holds in r'. (This is in fact part of the "inheritance" problem, and explained more carefully in Section 4.3). Hence $\mathcal{C}$ holds in r'.

Since $\mathcal{C}$ and $X \to Y$ hold in r', (the latter by the definition of $\sigma_{X \to Y}(r)$,) $T \to U$ must hold in r' to, since $\mathcal{C} \cup \{X \to Y\} \models T \to U$. However, since $\mathcal{C} \not\models T \to U$ and $\mathcal{C} \cup \{X \to Y\} \models T \to U$ we have that $\mathcal{C} \models T \to X$ by Lemma 4.3. Since $T \to X$ the argument given above for the X_i also holds for T: all tuples with some T-value are included in r' or none are. So if $T \to U$ holds in r', $T \not\to U$ cannot hold in r, a contradiction. □

Using Theorem 4.2 and 4.6 one can easily prove the following algorithm:

Algorithm 4.3 *Membership Detection for ad's with cfd's*

Input: $\mathcal{C}, \mathcal{A}$, a set of cfd's and a set of ad's, not in conflict; $X \not\to Y$ an ad.

Output: *true* or *false*

Method:

for each $T \not\to U$ **in** $\mathcal{A}$ **do**

if $\mathcal{C} \cup \{X \to Y\} \models T \to U$
then return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

The time needed to perform a membership test for ad's is the same as for the conflict detection: $O(n^3r^2m)$ where $n = \sharp\mathcal{C}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$.

In Chapter 2 we have seen that the implication problem for fd's is not affected by the presence of ad's in the relation scheme. Rule $CA2$ shows that for cfd's this is not the case.

Theorem 4.7 *Let $\mathcal{C} \cup \mathcal{A}$ be not in conflict. Then $\mathcal{C} \cup \mathcal{A} \models X \to Y \supset\!\!- X \to Z$ iff $\mathcal{C} \models X \to Y \supset\!\!- X \to Z$ or $\mathcal{C} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$.*

Proof The if-part is trivial.
For the only-if-part, assume that $\mathcal{C} \not\models X \to Y \supset\!\!- X \to Z$ and $\mathcal{C} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$. By Theorem 4.6 $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\}$ is not in conflict. Consider $Arm(FSAT_{\mathcal{C}}(X,Y))$. In $Arm(FSAT_{\mathcal{C}}(X,Y))$ $\mathcal{C} \cup \{X \to Y\}$ holds by definition (of Armstrong relations and of $FSAT_{\mathcal{C}}(X,Y)$). If some ad $T \not\twoheadrightarrow U$ of $\mathcal{A}$ does not hold then $\mathcal{C} \cup \{X \to Y\} \models T \to U$, a contradiction with $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\}$ not being in conflict. In $Arm(FSAT_{\mathcal{C}}(X,Y))$, $X \to Z$ does not hold by Theorem 4.4. Hence $Arm(FSAT_{\mathcal{C}}(X,Y))$ is an instance in which $\mathcal{C} \cup \mathcal{A}$ holds and in which $X \to Y \supset\!\!- X \to Z$ does not hold. Hence $\mathcal{C} \cup \mathcal{A} \not\models X \to Y \supset\!\!- X \to Z$.

□

A membership algorithm for cfd's (considering the presence of ad's) now really becomes trivial:

Algorithm 4.4 *Membership Detection for cfd's with ad's*

Input: $\mathcal{C}, \mathcal{A}$, a set of cfd's and a set of ad's, not in conflict; $X \to Y \supset\!\!- X \to Z$ a cfd.

Output: *true* or *false*

Method: return($(\mathcal{C} \models X \to Y \supset\!\!- X \to Z)$ or $(\mathcal{C} \cup \mathcal{A} \models X \not\twoheadrightarrow Y)$)

□

Because of Theorem 4.7 this membership algorithm for cfd's takes as much time as a membership algorithm for ad's: $O(n^3r^2m)$ with $n = \sharp\mathcal{C}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$.

We now show why our inference rules are not complete for cfd's and ad's. We also show that there exists no complete k-ary set of inference rules. In the next chapter we can explain why it is unlikely that a complete set of inference rules for cfd's and ad's exists.

Example 4.1 Let $\mathcal{C} = \{AB \to C \supset\!\!- AB \to E\ ,\ A \to D \supset\!\!- A \to F\}$ and $\mathcal{A} = \{AB \not\Rightarrow EF\}$. We show that $\mathcal{C} \cup \{A \to CD\} \models AB \to EF$, hence also $\mathcal{C} \cup \{AB \not\Rightarrow EF\} \models A \not\Rightarrow CD$, but $\mathcal{C} \cup \{AB \not\Rightarrow EF\} \not\vdash A \not\Rightarrow CD$.

If $A \to CD$ holds, then also $AB \to C$ and $A \to D$), by simple applications of the rules $F1 \ldots F3$. The two cfd's then generate $AB \to E$ and $A \to F$, from which one easily deduces $AB \to FE$.

Although Theorem 4.2 and 4.4 already show that $\mathcal{C} \cup \{AB \not\Rightarrow EF\} \models A \not\Rightarrow CD$ we show this explicitly:

Consider an instance in which $\mathcal{C} \cup \{AB \not\Rightarrow EF\}$ holds. Consider an arbitrary A-value in this instance. $AB \not\Rightarrow EF$ induces $A \not\Rightarrow EF$ by $FA1$ (with the trivial fd $AB \to A$). So in this A-complete set there are at least two EF-values. There are two possibilities: There are two E-values or there are two F-values (or both). If there are two F-values then $A \to D \supset\!\!- A \to F$ induces $A \not\Rightarrow D$ (for this A-value) by $CA1$. If there are two E-values (and only one F-value) then for every AB-value (with the same A-value) we have two E-values (since $AB \not\Rightarrow EF$ holds). Hence $AB \not\Rightarrow E$ holds in these AB-complete sets. The first cfd induces $AB \not\Rightarrow C$ by $CA1$, which induces $A \not\Rightarrow C$. This proves that in every A-complete set of tuples $A \not\Rightarrow C$ or $A \not\Rightarrow D$ holds, hence $A \not\Rightarrow CD$ holds in the entire instance.

The reader is invited to show that $\mathcal{C} \cup \{AB \not\Rightarrow EF\} \not\vdash A \not\Rightarrow CD$. This can be done by generating all possible ad's that can be inferred from $\mathcal{C} \cup \mathcal{A}$ and verifying that $A \not\Rightarrow CD$ is not among them. However the main reason for not being able to generate $A \not\Rightarrow CD$ is that Lemma 4.4 cannot be applied since $A \to CD \supset\!\!- A \to AB$ does not hold. Also, the fd $ABD \to EF$ does not hold (this fd is suggested in Remark 3.1).

□

The real reason why the rules are not complete is explained in the next chapter. However, we can show that no k-ary set of inference rules exists for cfd's and ad's. The example is a special case of the following rule:

CA_k : if $X_1 \to Y_1 \supset\!\!- X_1 \to Y_2$
$X_2 \to Y_3 \supset\!\!- X_2 \to Y_4$
$\ldots$
$X_k \to Y_{2k-1} \supset\!\!- X_k \to Y_{2k}$
$X_1 X_2 \ldots X_k \not\twoheadrightarrow Y_2 Y_4 \ldots Y_{2k}$
then $X_1 \cap X_2 \cap X_k \not\twoheadrightarrow Y_1 Y_3 \ldots Y_{2k-1}$

The reader is invited to show that the rules CA_k are correct (following the argument of the example) and that the rule CA_k cannot be simulated by $CA1, CA2, CA_1 \ldots CA_{k-1}$. Also, there exists no set of $k-1$ or less cfd's, equivalent to the cfd's of CA_k. This leads to:

Remark 4.2 *There exists no complete k-ary set of inference rules for mixed cfd's and ad's.*

□

4.3 The Inheritance of cfd's and ad's

The membership problem has been studied to decide whether a decomposition according to a cfd is trivial or not (i. e. whether for the cfd $X \to Y \supset\!\!- X \to Z$ the fd $X \to Y$ or the ad $X \not\twoheadrightarrow Y$ holds). When performing several decomposition steps (i. e. decomposing the subschemes further on) it is necessary to know which dependencies hold in the subschemes that are the result of a decomposition step. This *inheritance problem* is slightly more complicated than for fd's and ad's, since both cfd's and ad's can be lost by a horizontal decomposition step, whereas fd's cannot.

Notation 4.1 In the sequel we treat the horizontal decomposition of a scheme $R = (\Omega, \mathcal{C} \cup \mathcal{A})$, according to $X \to Y \supset\!\!- X \to Z \in \mathcal{C}$, into the schemes $R_1 = \sigma_{X \to Y}(R) = (\Omega, \mathcal{C}_1 \cup \mathcal{A}_1)$, and $R_2 = \sigma_{X \not\twoheadrightarrow Y}(R) = (\Omega, \mathcal{C}_2 \cup \mathcal{A}_2)$. We assume that $\mathcal{C} \cup \mathcal{A}$ is not in conflict, $\mathcal{C} \not\models X \to Y$ and $\mathcal{C} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$. (Otherwise we obtain "trivial" decompositions.) We again do not consider "complete" sets of dependencies. The sets of dependencies we shall generate for the subschemes are only generating for the set of all dependencies, holding in the subschemes.

□

Since fd's cannot be violated by taking a selection of a relation we have:

Remark 4.3 *All the fd's that hold in R also hold in both R_1 and R_2.* □

The fd's of R are not the only fd's that hold in the subschemes. In R_1 for instance the fd $X \to YZ$ holds (which does not hold in R if the decomposition is not trivial, i. e. if $\mathcal{C} \not\models X \to Y$ and $\mathcal{C} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$).

When considering cfd's and ad's there always is the danger of introducing conflict, when modifying the sets of cfd's and ad's. However, the non-trivial horizontal decomposition of a (nonempty) relation cannot generate sets of dependencies that are in conflict, since the subrelations are nonempty too. Indeed, if neither $X \to Y$ nor $X \not\twoheadrightarrow Y$ holds in R, then in most instances r of R the subinstances $\sigma_{X \to Y}(r)$ and $\sigma_{X \not\twoheadrightarrow Y}(r)$ will be nonempty. Therefore we do not have to consider the danger of generating conflict by decomposing a scheme in the sequel.

For cfd's and ad's the inheritance problem is more complicated than for fd's. However, similar inclusions can be easily proved:

Lemma 4.6 *Using Notation 4.1 we have:*

- $\mathcal{C}_1 \subseteq \{T \to U \supset\!\!- T \to V \mid \mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \models T \to U \supset\!\!- T \to V\}$.
- $\mathcal{C}_2 \subseteq \{T \to U \supset\!\!- T \to V \mid \mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \models T \to U \supset\!\!- T \to V\}$.
- $\mathcal{A}_1 \subseteq \{T \not\twoheadrightarrow U \mid \mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \models T \not\twoheadrightarrow U\}$.
- $\mathcal{A}_2 \subseteq \{T \not\twoheadrightarrow U \mid \mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \models T \not\twoheadrightarrow U\}$.

Proof The four cases are somewhat similar:

$\mathcal{C}_1$: Let $T \to U \supset\!\!- T \to V$ be such that $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \not\models T \to U \supset\!\!- T \to V$. We show that $T \to U \supset\!\!- T \to V \notin \mathcal{C}_1$, by constructing an instance r for which $T \to U \supset\!\!- T \to V$ does not hold in $r_1 = \sigma X \to Y(r)$.
$\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \not\models T \not\twoheadrightarrow U$ by Theorem 4.7. Hence by Theorem 4.6 $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \cup \{T \to U\}$ is not in conflict and holds in $r = Arm(FSAT_{\mathcal{C} \cup \{X \to Y\}}(T, U))$. Since $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \not\models T \to U \supset\!\!- T \to V$, $T \not\twoheadrightarrow V$ holds in $Arm(FSAT_{\mathcal{C} \cup \{X \to Y\}}(T, U))$. When this instance is decomposed according to $X \to Y \supset\!\!- X \to Z$ then $r_1 = \sigma_{X \to Y}(r) = Arm(FSAT_{\mathcal{C} \cup \{X \to Y\}}(T, U))$, in which $T \to U \supset\!\!- T \to V$ does not hold (and $r_2 = \emptyset$).

$\mathcal{C}_2$: Let $T \to U \supset\!\!- T \to V$ be such that $\mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \not\models T \to U \supset\!\!- T \to V$. Then $\mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$ by Theorem 4.7

and hence $\mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \cup \{T \to U\}$ is not in conflict by Theorem 4.6. The r we need know is $Arm(FSAT_{\mathcal{C}}(T, U))$.

$\mathcal{A}_1$: If $\mathcal{C} \cup \mathcal{A} \cup \{X \to Y\} \not\models T \not\twoheadrightarrow U$ then we proceed as in the case for $\mathcal{C}_1$.

$\mathcal{A}_2$: If $\mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$ then we proceed as in the case for $\mathcal{C}_2$.

□

Some dependencies are inherited by both R_1 and R_2. They show a close resemblance with Lemma 3.2.

Lemma 4.7 *Using Notation 4.1, an ad holds in both R_1 and R_2 if $\mathcal{C} \cup \mathcal{A} \models T \not\twoheadrightarrow U$ and $\mathcal{C} \models T \to X$. A cfd $T \to U \supset\!\!- T \to V$ holds in both R_1 and R_2 if $\mathcal{C} \cup \mathcal{A} \models T \to U \supset\!\!- T \to V$ and $\mathcal{C} \models T \to U$ or $\mathcal{C} \models T \to X$.*

Proof Suppose $\mathcal{C} \models T \to X$. Consider, in an arbitrary instance r, an arbitrary T-complete set s of tuples, with a unique value on their T-projection, (i. e. a T-unique, T-complete set). Because $T \to X$ holds in r, s is a part of an X-complete set with only one X-value. Hence $s \subseteq \sigma_{X \to Y}(r)$ or $s \subseteq \sigma_{X \not\twoheadrightarrow Y}(r)$, i. e. s cannot be divided into two parts by decomposing r according to $X \to Y \supset\!\!- X \to Z$. This means that all the T-complete sets of r_1 and r_2 are T-complete sets of r. Hence the cfd $T \to U \supset\!\!- T \to V$ (or the ad $T \not\twoheadrightarrow U$) cannot be violated in r_1 or r_2 if it holds in r.

Suppose $\mathcal{C} \models T \to U$ and $\mathcal{C} \models T \to U \supset\!\!- T \to V$. Then $\mathcal{C} \models T \to UV$ and hence $T \to UV$ holds in R_1 and R_2 since an fd cannot be violated by taking a restriction. $T \to UV$ implies $T \to U \supset\!\!- T \to V$, hence $T \to U \supset\!\!- T \to V$ holds in R_1 and R_2.

□

Notation 4.2 We denote the set of the ad's of $\mathcal{A}$, that are inherited because of Lemma 4.7, by $\hat{\mathcal{A}}$. The set of the cfd's of $\mathcal{C}$, that are inherited because of Lemma 4.7 or that are trivial, is denoted by $\hat{\mathcal{C}}$. The trivial cfd's are included because they represent the goals of Chapters 2 and 3. They are of no importance for the inheritance problem, and are neglected in the sequel.

□

In the proof of the inheritance of cfd's and ad's a special instance is needed, of which the construction is partially described below.

Lemma 4.8 *Consider a set $\mathcal{C} \cup \mathcal{A}$, not being in conflict. Let s be an instance in which $\mathcal{C} \cup \mathcal{A}$ holds.*
If $\mathcal{C} \cup \mathcal{A} \cup \{P \not\twoheadrightarrow Q\}$ is not in conflict, then there is an instance r in which $\mathcal{C} \cup \mathcal{A}$ still holds, which contains s as a subset, and in which the ad $P \not\twoheadrightarrow Q$ holds (and hence also every cfd $P \rightarrow Q \supset\!\!- P \rightarrow O$).

Proof A T-complete set of tuples, all having the same T-projection, in which the ad $T \not\twoheadrightarrow U$ does not hold (hence in which $T \rightarrow U$ holds) is called a *violation of* $T \not\twoheadrightarrow U$. A T-complete set of tuples, all having the same T-projection, in which $T \rightarrow U$ and $T \not\twoheadrightarrow V$ hold is called a *violation of* $T \rightarrow U \supset\!\!- T \rightarrow V$.

Let $P \not\twoheadrightarrow Q$ not hold in s. Construct $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$, and suppose that the domains of $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$ and s are disjoint. Suppose also that in $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$ the domains of the attributes all are disjoint.

Let $\overline{P} = \{A \mid \mathcal{C} \models P \rightarrow A\}$. Let t be a tuple in an arbitrary violation of $P \not\twoheadrightarrow Q$ (in s). Let u be an arbitrary tuple of $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$. The domain of $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$ is changed such that $u[\overline{P}] := t[\overline{P}]$. Let the adapted Armstrong relation be called s'.

Let $r = s \cup s'$. In r, $\mathcal{C}$ holds, since $\mathcal{C}$ holds in s and s', and since if $V \not\subset \overline{P}$ then $V \rightarrow W$, for which $\mathcal{C} \models V \rightarrow W$, still holds because no tuple of s' has the same V-projection as any tuple of s, and if $V \subseteq \overline{P}$ then $V \rightarrow W$, for which $\mathcal{C} \models V \rightarrow W$, still holds because also $W \subseteq \overline{P}$ and if the V-projection of a tuple of s and of a tuple of s' are equal then this projection is $t[V]$, and hence for the tuple of s and the tuple of s' the W-projection is $t[W]$ by the construction of s'. If for $V \rightarrow W \supset\!\!- V \rightarrow W' \in \mathcal{C}$ $V \rightarrow W$ does not hold, then in s' $V \not\twoheadrightarrow W$ holds, hence in r $V \rightarrow W \supset\!\!- V \rightarrow W'$ still holds.

In r, $\mathcal{A}$ holds since $\mathcal{A}$ holds in s and s' and since ad's cannot be violated by taking a union.

In s', $P \not\twoheadrightarrow Q$ holds, hence s' does not contain any violation of $P \not\twoheadrightarrow Q$. In r the violation of $P \not\twoheadrightarrow Q$ (in s) that contains t is no longer a violation of $P \not\twoheadrightarrow Q$. Hence in r the number of violations of $P \not\twoheadrightarrow Q$ is (strictly) less than in s.

By repeating the above construction until there are no violations of $P \not\twoheadrightarrow Q$ any more (in the final r), one establishes an instance r in which $P \not\twoheadrightarrow Q$ holds, and which still contains s as a subset.

It is obvious that when $P \not\twoheadrightarrow Q$ holds, then also $P \rightarrow Q \supset\!\!- P \rightarrow O$, by the definition of cfd's (and also expressed by rule $CA2$). □

The above construction is more or less the same as for fd's and ad's. Similar constructions will appear in the next chapters.

Lemma 4.9 *Let $\mathcal{C} \cup \mathcal{A}$ be not in conflict. Let s be an instance in which $\hat{\mathcal{C}} \cup \hat{\mathcal{A}}$ holds. Then there exists an instance r, containing s as a subset, and in which $\mathcal{C} \cup \mathcal{A}$ holds.*

Proof r can be obtained by repeating the construction of the proof of Lemma 4.8 for every ad of $\mathcal{A} - \hat{\mathcal{A}}$ and every cfd of $\mathcal{C} - \hat{\mathcal{C}}$. The final instance r satisfies $\mathcal{C} \cup \mathcal{A}$. □

Theorem 4.8 *Using Notation 4.1 and 4.2, a cfd or ad must hold in R_1 (resp. R_2) iff it is a consequence of $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\}$ (resp. $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\}$).*

Proof From Lemma 4.6 and 4.7 it follows that $(\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\})^* \subseteq (\mathcal{C}_1 \cup \mathcal{A}_1)^* \subseteq (\mathcal{C} \cup \mathcal{A} \cup \{X \rightarrow Y\})^*$ and $(\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\})^* \subseteq (\mathcal{C}_2 \cup \mathcal{A}_2)^* \subseteq (\mathcal{C} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\})^*$.

The proof has to cover 4 cases: a cfd and an ad, in R_1 and in R_2.

1. Consider an ad $T \not\twoheadrightarrow U \in (\mathcal{C} \cup \mathcal{A} \cup \{X \rightarrow Y\})^* - (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\})^*$. We prove that $T \not\twoheadrightarrow U \notin (\mathcal{C}_1 \cup \mathcal{A}_1)^*$. Since $T \not\twoheadrightarrow U \notin (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\})^*$, $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\} \cup \{T \rightarrow U\}$ is not in conflict, by Theorem 4.6. Hence there exists an instance s in which $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\} \cup \{T \rightarrow U\}$ holds. By the construction of Lemma 4.9 an instance r can be build which contains s and in which $\mathcal{C} \cup \mathcal{A}$ holds. In this construction (explained in the proof of Lemma 4.8) a number of modified copies of $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$ are added to s. Since $X \not\twoheadrightarrow Y$ holds in $Arm(FSAT_{\mathcal{C}}(\emptyset, \emptyset))$, and since $\mathcal{C} \not\models T \rightarrow X$, $r_1 = \sigma_{X \rightarrow Y}(r) = s$. Hence in r_1 $T \not\twoheadrightarrow U$ does not hold, which means that $T \not\twoheadrightarrow U \notin (\mathcal{C}_1 \cup \mathcal{A}_1)^*$.
2. Consider a cfd $T \rightarrow U \supset\!\!- T \rightarrow V \in (\mathcal{C} \cup \mathcal{A} \cup \{X \rightarrow Y\})^* - (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\})^*$. We prove that $T \rightarrow U \supset\!\!- T \rightarrow V \notin (\mathcal{C}_1 \cup \mathcal{A}_1)^*$. Since $T \rightarrow U \supset\!\!- T \rightarrow V \notin (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\})^*$, $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\} \cup \{T \rightarrow U\}$ cannot be in conflict. (Otherwise $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\} \models T \not\twoheadrightarrow U$ which contradicts with $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \rightarrow Y\} \not\models T \rightarrow U \supset\!\!- T \rightarrow V$ (by Theorem 4.7).)

$\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \to Y\} \cup \{T \to U\} \cup \{T \not\to V\}$ cannot be in conflict either, since $T \to V \notin FSAT_{\hat{\mathcal{C}} \cup \{X \to Y\}}(T, U)$, (hence $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \to Y\} \cup \{T \to U\} \cup \{T \not\to V\}$ holds in $Arm(FSAT_{\hat{\mathcal{C}} \cup \{X \to Y\}}(T, U))$). There exists a set s of tuples, in which $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \to Y\} \cup \{T \to U\} \cup \{T \not\to V\}$ holds. By Lemma 4.9 one can construct an instance r, containing s, in which $\mathcal{C} \cup \mathcal{A}$ holds. $r_1 = \sigma_{X \to Y}(r) = s$, hence $T \to U \supset\!\!- T \to V$ does not hold in r_1. This means that $T \to U \supset\!\!- T \to V \notin (\mathcal{C}_1 \cup \mathcal{A}_1)^*$.

3. Consider an ad $T \not\to U \in (\mathcal{C} \cup \mathcal{A} \cup \{X \not\to Y\})^* - (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\})^*$. We prove that $T \not\to U \notin (\mathcal{C}_2 \cup \mathcal{A}_2)^*$. Since $T \not\to U \notin (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\})^*$, $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\} \cup \{T \to U\}$ is not in conflict, by Theorem 4.6. Hence there exists an instance s in which $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\} \cup \{T \to U\}$ holds. By a construction, very similar to that of Lemma 4.8 an instance r can be build which contains s and in which $\mathcal{C} \cup \mathcal{A}$ is satisfied. To obtain that $\sigma_{X \not\to Y}(r) = s$ one must use modified copies of $Arm(FSAT_{\hat{\mathcal{C}}}(X, Y))$ instead of $Arm(FSAT_{\hat{\mathcal{C}}}(\emptyset, \emptyset))$. In $r_2 = \sigma_{X \not\to Y}(r)$, $T \to U$ holds, hence $T \not\to U \notin (\mathcal{C}_2 \cup \mathcal{A}_2)^*$.

   The above argument relies on the observation that $T \not\to U$ holds in $Arm(FSAT_{\hat{\mathcal{C}}}(X, Y))$, i. e. that $\hat{\mathcal{C}} \cup \{X \to Y\} \not\models T \to U$. Suppose $\hat{\mathcal{C}} \cup \{X \to Y\} \models T \to U$ (hence $T \to U \in FSAT_{\hat{\mathcal{C}}}(X, Y)$). We also have that $\mathcal{C} \cup \mathcal{A} \cup \{X \not\to Y\} \models T \not\to U$. $T \to U \in FSAT_{\hat{\mathcal{C}}}(X, Y)$ means that $\hat{\mathcal{C}} \models T \to X$, by Lemma4.3. If $\mathcal{C} \cup \mathcal{A} \models T \not\to U$ then $T \not\to U$ is inherited because of Lemma 4.7. Since $T \not\to U \notin \hat{\mathcal{A}}$, this is not the case, hence $\mathcal{C} \cup \mathcal{A} \not\models T \not\to U$. By the proof of Theorem 4.6 $\mathcal{C} \cup \mathcal{A} \cup \{X \not\to Y\} \models T \not\to U$ and $\mathcal{C} \cup \mathcal{A} \not\models T \not\to U$ imply $\mathcal{C} \cup \{X \not\to Y\} \models T \not\to U$, and hence also $\mathcal{C} \cup \{T \to U\} \models X \to Y$ (by the proof of Theorem 4.6 again). To deduce $X \to Y$ from $\mathcal{C} \cup \{T \to U\}$, only cfd's $P \to Q \supset\!\!- P \to O \in \mathcal{C}$ for which $\mathcal{C} \models P \to Q$ or $\mathcal{C} \models P \to T$ can be used. Since $\hat{\mathcal{C}} \models T \to X$ these cfd's all are in $\hat{\mathcal{C}}$ ($P \to T$ and $T \to X$ induce $P \to X$). Hence $\hat{\mathcal{C}} \cup \{T \to U\} \models X \to Y$, or, by the proof of Theorem 4.6, $\hat{\mathcal{C}} \cup \{X \not\to Y\} \models T \not\to U$, which contradicts with $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\} \not\models T \not\to U$.

4. Consider a cfd $T \to U \supset\!\!- T \to V \in (\mathcal{C} \cup \mathcal{A} \cup \{X \not\to Y\})^* - (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\})^*$. Since $T \to U \supset\!\!- T \to V \notin (\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\})^*$, $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\to Y\} \cup \{T \to U\} \cup \{T \not\to V\}$ cannot be in conflict (for the same reason as for a cfd in R_1). The same construction as in case 3 now

produces an instance such that in r_2 $T \to U$ and $T \not\twoheadrightarrow V$ hold. Hence $T \to U \supset\!\!- T \to V \notin (\mathcal{C}_2 \cup \mathcal{A}_2)^*$.

□

From Theorem 4.8 one can easily deduce an algorithm which calculates generating sets for the inherited dependencies, and which takes polynomial time only, since it is not necessary to calculate the closure (i.e. the set of all the consequences) of a set of dependencies.

Algorithm 4.5 *A Horizontal Decomposition Step with cfd's and ad's*

Input: $R = (\Omega, \mathcal{C} \cup \mathcal{A})$ and a cfd $X \to Y \supset\!\!- X \to Z \in \mathcal{C}$. $\mathcal{C} \cup \mathcal{A}$ is assumed not to be in conflict, $\mathcal{C} \not\models X \to Y$ and $\mathcal{C} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$.

Output: An ordered pair of schemes $(R_1 = (\Omega, \mathcal{C}_1 \cup \mathcal{A}_1), R_2 = (\Omega, \mathcal{C}_2 \cup \mathcal{A}_2))$, being the decomposition of R according to $X \to Y \supset\!\!- X \to Z$.

Method:

var $\hat{\mathcal{C}}, \hat{\mathcal{C}}_1, \hat{\mathcal{C}}_2$: set of cfd's $:= \emptyset$
$\hat{\mathcal{A}}$: set of ad's $:= \emptyset$

begin
for each $T \to U \supset\!\!- T \to V \in \mathcal{C}$ **do**
if $\mathcal{C} \models T \to U$ or $\mathcal{C} \models T \to X$ or $V \subseteq TU$
then
$\hat{\mathcal{C}} := \hat{\mathcal{C}} \cup \{T \to U \supset\!\!- T \to V\}$
od
for each $T \not\twoheadrightarrow U \in \mathcal{A}$ **do**
if $\mathcal{C} \models T \to X$
then
$\hat{\mathcal{A}} := \hat{\mathcal{A}} \cup \{T \not\twoheadrightarrow U\}$
od
for each $T \to U \supset\!\!- T \to V \in \hat{\mathcal{C}}$ **do**
if $\hat{\mathcal{C}} \cup \{X \to YZ\} \not\models T \to U$ and $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \to YZ\} \not\models T \not\twoheadrightarrow U$
then
$\hat{\mathcal{C}}_1 := \hat{\mathcal{C}}_1 \cup \{T \to U \supset\!\!- T \to V\}$
od
for each $T \to U \supset\!\!- T \to V \in \hat{\mathcal{C}}$ **do**
if $\hat{\mathcal{C}} \not\models T \to U$ and $\hat{\mathcal{C}} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$
then

$$\hat{\mathcal{C}}_2 := \hat{\mathcal{C}}_2 \cup \{T \to U \supset\!\!- T \to V\}$$

od

return$(R_1 = (\Omega, \hat{\mathcal{C}}_1 \cup \{X \to YZ\} \cup \hat{\mathcal{A}}), R_2 = (\Omega, \hat{\mathcal{C}}_2 \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\}))$

end

□

In this algorithm, we used that the presence of ad's has no effect on the membership problem for fd's (expressed as cfd's), i. e. $\mathcal{C} \models T \to U$ iff $\mathcal{C} \cup \mathcal{A} \models T \to U$. This follows immediately from Theorem 4.7.

If $n = \sharp\mathcal{C}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$, one can easily see that the time-complexity of Algorithm 4.5 is n times the complexity of a cfd-membership test plus m times the complexity of an ad-membership test, i. e. $O(n^4r^2 + n^3m^2r^2)$.

4.4 The "Conditional" Normal Form

In this section we illustrate an algorithm for horizontal decomposition of a relation scheme, according to its cfd's. The algorithm decomposes the scheme (and subschemes) until no subscheme can be decomposed any further. This is formalized by defining a normal form.

Definition 4.5 A scheme $R = (\Omega, \mathcal{C} \cup \mathcal{A})$ is said to be in *Conditional Normal Form (CNF)* iff for all $X \to Y \supset\!\!- X \to Z \in \mathcal{C}$ holds $\mathcal{C} \cup \mathcal{A} \models X \to Y$ or $\mathcal{C} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$.
A decomposition $(R_1, \ldots, R_n)$ is in *CNF* iff all the R_i, $i = 1 \ldots n$, are in *CNF*.

□

Since the horizontal decomposition is based on real constraints, there is no way to define the inheritance differently, as was possible with goals. However, for the "trivial" cfd's, the choice of explicitly inheriting the trivial cfd's is arbitrary. If we would include all trivial cfd's in the sets $\mathcal{C}_1$ and $\mathcal{C}_2$ we obtain the trivial decompositions of Section 3.3. In Algorithm 4.5 we have defined that the trivial cfd's are inherited in the same way as in the inherited decomposition steps of Section 3.3. Hence the horizontal decomposition into Conditional Normal Form is equivalent to the inherited decomposition into Horizontal Normal Form if only trivial cfd's are given.

From this definition the construction of a decomposition algorithm, which decomposes a relation scheme according to a cfd, and then decomposes the

subschemes further on, until all subschemes are in *CNF*, is straightforward.

To illustrate the horizontal decomposition into *CNF*, we modify Example 2.1:

Example 4.2 Recall Example 2.1. In Section 4.1 we showed that the relation scheme *STAFF* is likely to satisfy $emp \rightarrow job\ man\ dep\ div \supset\!\!- emp \supset\!\!- sal$. We represent some of the goals as trivial cfd's: $emp \rightarrow job \supset\!\!- emp \rightarrow job$ and $man \rightarrow div \supset\!\!- man \rightarrow div$. We no longer consider the fd $emp\ job \rightarrow sal$ since that is a stronger constraint than our cfd. Figure 4.1 shows a decomposition tree for the decomposition of *STAFF* into *CNF*. The sets of cfd's and ad's we give are not exactly the sets generated by Algorithm 4.5. We removed redundant elements, and combined constraints, to keep the figure as small as possible.
Table 4.1 shows the instance of Table 2.1, with a small modification: in Example 2.1 we choose to split up an *emp*loyee's *sal*ary if he has more than one *job*. It is only natural to do the same if an *emp*loyee works for more than one *dep*artment or *div*ision. This however was forbidden by the fd $emp\ job \rightarrow sal$, which we have now abandoned. Table 4.1 now shows more than one *sal*ary for Goldstein (but accidently not for Shapiro).
After the decomposition into *CNF* the following subinstances are obtained:

$staff_{111}$ contains the *emp*loyees having only one *job*, one *man*ager, one *dep*artment and one *div*ision (and hence also only one *sal*ary), and whose *man*ager has only one *div*ision for these *emp*loyees.

$staff_{111}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Wallace	accountant	Brown	2000	sales	Los Angeles
Diamond	accountant	Brown	1500	sales	Los Angeles
Brown	sales manager	Goldstein	3000	sales	Los Angeles
Eltman	carrier	Kedesdy	1000	stock	Los Angeles
Kedesdy	accountant	Brown	2000	stock	Los Angeles
Carlson	chauffeur	Kedesdy	1500	stock	Los Angeles
Pike	secretary	Kedesdy	1300	stock	Los Angeles

$staff_{112}$ contains the same kind of *emp*loyees as $staff_{111}$, but whose *man*ager has more than one *div*ision for these *emp*loyees.

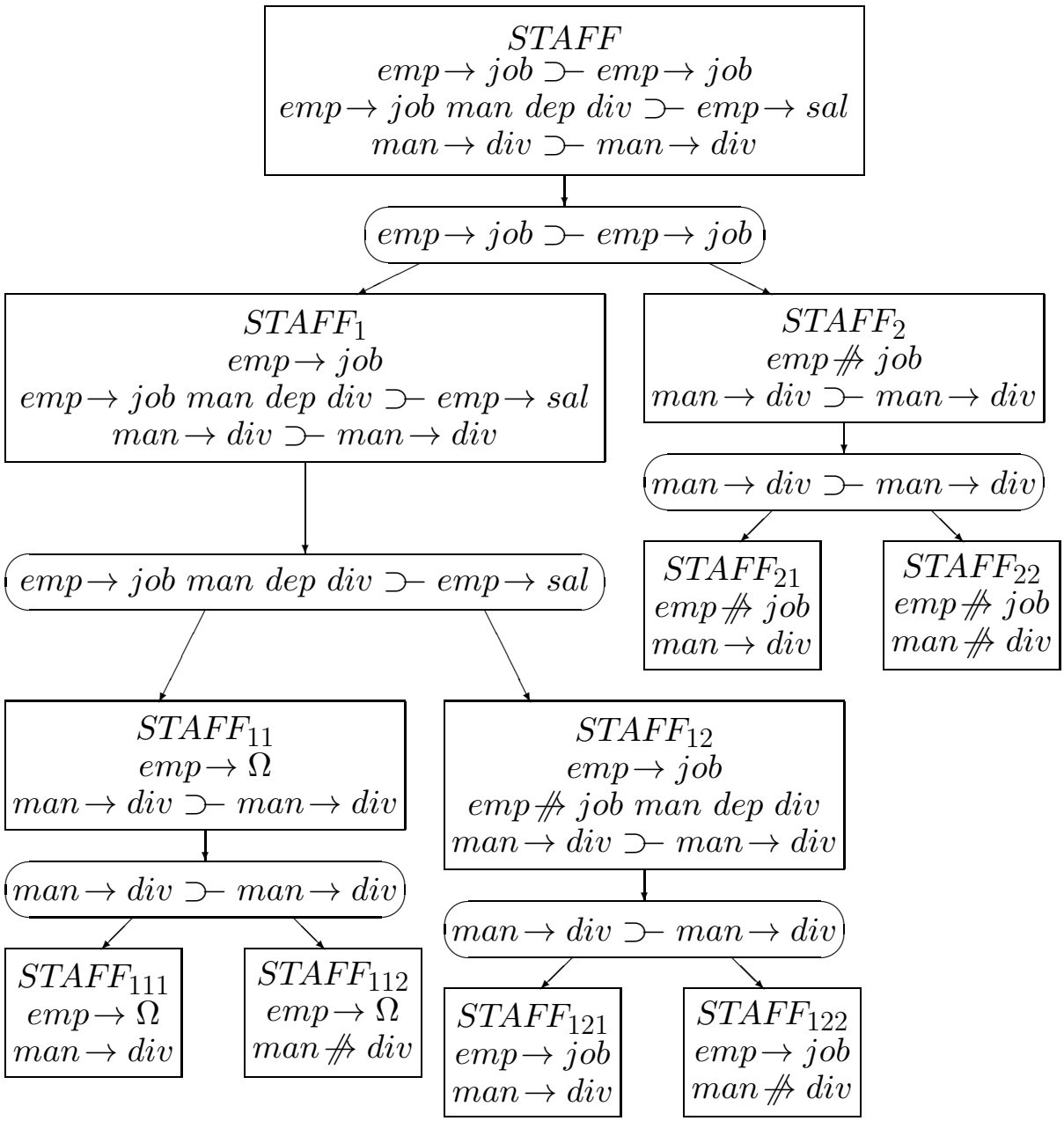


Figure 4.1: A decomposition tree for *STAFF*, into *CNF*.

$staff_{112}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Pierce	accountant	Shapiro	1500	sales	Santa Barbara
Jones	chauffeur	Shapiro	1000	sales	Santa Barbara
Matthews	accountant	Shapiro	1600	sales	Bakersfield

In Example 2.1 $staff_{12}$ could not be decomposed any more because $man \not\rightarrow div$ was not clean. We do not consider "clean cfd's" and hence can decompose $staff_{12}$:

emp	*job*	*man*	*sal*	*dep*	*div*
Wallace	accountant	Brown	2000	sales	Los Angeles
Diamond	accountant	Brown	1500	sales	Los Angeles
Murrel	secretary	Wallace	1000	sales	Los Angeles
Murrel	secretary	Diamond	1000	sales	Los Angeles
Murrel	secretary	Brown	1000	sales	Los Angeles
Brown	sales manager	Goldstein	3000	sales	Los Angeles
Goldstein	gen. manager	Goldstein	1200	sales	Los Angeles
Eltman	carrier	Kedesdy	1000	stock	Los Angeles
Kedesdy	accountant	Brown	2000	stock	Los Angeles
Carlson	chauffeur	Kedesdy	1500	stock	Los Angeles
Pike	secretary	Kedesdy	1300	stock	Los Angeles
Goldstein	gen. manager	Goldstein	800	stock	Los Angeles
Pierce	accountant	Shapiro	1500	sales	Santa Barbara
Goodwin	secretary	Pierce	1000	sales	Santa Barbara
Goodwin	secretary	Shapiro	1000	sales	Santa Barbara
Jones	chauffeur	Shapiro	1000	sales	Santa Barbara
Shapiro	accountant	Shapiro	1000	sales	Santa Barbara
Shapiro	sales manager	Goldstein	1000	sales	Santa Barbara
Goldstein	gen. manager	Goldstein	1000	sales	Santa Barbara
Matthews	accountant	Shapiro	1600	sales	Bakersfield
Tyrrell	chauffeur	Shapiro	1000	sales	Bakersfield
Tyrrell	carrier	Shapiro	800	sales	Bakersfield
Shapiro	sales manager	Goldstein	1000	sales	Bakersfield
Goldstein	gen. manager	Goldstein	1000	sales	Bakersfield

Table 4.1: Instance for *STAFF* (with cfd's).

$staff_{121}$ contains the employees with only one *job*, but with more than one *man*ager, *dep*artment or *div*ision, and whose *man*ager has only one *div*ision for these *emp*loyees.

$staff_{121}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Murrel	secretary	Wallace	1000	sales	Los Angeles
Murrel	secretary	Diamond	1000	sales	Los Angeles
Murrel	secretary	Brown	1000	sales	Los Angeles
Goodwin	secretary	Pierce	1000	sales	Santa Barbara
Goodwin	secretary	Shapiro	1000	sales	Santa Barbara

$staff_{122}$ contains the employees with only one *job*, but with more than one *man*ager, *dep*artment or *div*ision, and whose *man*ager has more than one *div*ision for these *emp*loyees.

$staff_{122}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Goldstein	gen. manager	Goldstein	1200	sales	Los Angeles
Goldstein	gen. manager	Goldstein	800	stock	Los Angeles
Goldstein	gen. manager	Goldstein	1000	sales	Santa Barbara
Goldstein	gen. manager	Goldstein	1000	sales	Bakersfield

$staff_{21}$ contains the *emp*loyees with more than one *job*, and whose *man*ager has only one *div*ision for these employees. The cfd $emp \rightarrow job\ man\ dep\ div \supset\!\!- emp \rightarrow sal$ cannot be used since $emp \not\twoheadrightarrow job$ implies $emp \not\twoheadrightarrow job\ man\ dep\ div$. The way the *sal*aries are stored is no longer important: the cfd avoids the generation of empty subinstances.

$staff_{21}$ =

emp	*job*	*man*	*sal*	*dep*	*div*
Tyrrell	chauffeur	Shapiro	1000	sales	Bakersfield
Tyrrell	carrier	Shapiro	800	sales	Bakersfield

$staff_{22}$ contains the *emp*loyees having more than one *job* and whose *man*ager has more than one *div*ision.

$staff_{22} =$

emp	*job*	*man*	*sal*	*dep*	*div*
Shapiro	accountant	Shapiro	1000	sales	Santa Barbara
Shapiro	sales manager	Goldstein	1000	sales	Santa Barbara
Shapiro	sales manager	Goldstein	1000	sales	Bakersfield

When one compares the subinstances of *staff* with those of Example 2.1 one notes that with cfd's the same subinstances are obtained without generating any additional empty instances.

□

Chapter 5

Decomposition with ifd's

The horizontal decomposition according to cfd's is capable of representing the decomposition with goals, but the cfd's as constraints are quite restrictive: the two fd's $X \to Y$ and $X \to Z$ in the cfd $X \to Y \supset\!\!- X \to Z$ must have the same left hand side X. The reason for this restriction is that when decomposing the relation to obtain $X \to Y$, we want to generate the "implied" fd $X \to Z$ in the first subrelation. The easiest way to obtain this result is to let both fd's have the same left hand side, but this is certainly not a necessary condition.

This chapter covers [14]. The horizontal decomposition according to $X \to Y \supset\!\!- X \to Z$ splits r into two X-complete sets of tuples. If we make sure that the X-complete sets of tuples are also complete for the left hand side of the implied fd, then this fd will hold in the first subrelation. We formalize this weaker (and hence more general) kind of constraint below.

5.1 Imposed-Functional Dependencies

Let us first recall Example 2.1. In the previous chapter we already indicated that the cfd $emp \to job\ man\ dep\ div \supset\!\!- emp \to sal$ was more appropriate than the fd $emp\ job \to sal$. Some other implications between fd's may hold: $emp \to job \supset\!\!- emp\ man\ dep\ div \to sal$ means that if an employee has only one job, then he can have only one salary for every manager, department and division for which he works. This constraint is not a cfd, and also is not equivalent to the given cfd: it is weaker; a greater part of the database may satisfy the first fd.

Definition 5.1 Let R be a relation scheme, $X, X', Y, Z \subseteq \Omega$ and $X \subseteq X'$.

- A relation instance r of R satisfies the *imposed-functional dependency (ifd)* $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ iff in every X-complete set of tuples in r, in which the fd $X \rightarrow Y$ holds, the fd $X' \rightarrow Z$ must hold too.
- The scheme R satisfies $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ iff all the instances of R satisfy $X \rightarrow Y \supset\!\!- X' \rightarrow Z$.

□

The ifd's contain the (cfd's) since cfd's are ifd's $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ with $X = X'$. They also contain the fd's as a subclass. An fd $T \rightarrow U$ is equivalent to the ifd $\emptyset \rightarrow \emptyset \supset\!\!- T \rightarrow U$ (which is not a cfd) as well as a large number of other ifd's, e. g. $T \rightarrow T \supset\!\!- T \rightarrow U$ (which is a cfd). We usually denote an fd as fd, and not as a special ifd.

The condition $X \subseteq X'$ is still a rather strong condition, and certainly is not a necessary condition. From Lemma 4.7 we already know that $X' \rightarrow X$ is still a sufficient condition for the fd $X' \rightarrow Z$ to hold in $\sigma_{X \rightarrow Y}(R)$. This leads to the following definition:

Definition 5.2 Let R be a relation scheme, $X, X'Y, Z \subseteq \Omega$.

- A relation instance r of R satisfies the *generalized imposed-functional dependency (gifd)* $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ iff $X' \rightarrow X$ holds in r and if in every X-complete set of tuples in r, in which the fd $X \rightarrow Y$ holds, the fd $X' \rightarrow Z$ must hold too.
- The scheme R satisfies $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ iff all the instances of R satisfy $X \rightarrow Y \supset\!\!- X' \rightarrow Z$.

□

The class of the gifd's contains the ifd's as a subclass, and some gifd's cannot be expressed by means of an ifd. However, the following remark shows that we can simulate a gifd by means of an ifd and an fd. Hence the gifd's are not more powerful than the ifd's, so we only discuss ifd's in this chapter.

Remark 5.1 *The gifd $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ (with $X' \rightarrow Z$) is equivalent to the ifd $X \rightarrow Y \supset\!\!- XX' \rightarrow Z$ together with the fd $X' \rightarrow X$.*

□

Definition 5.3 The *horizontal decomposition of a scheme R, according to the ifd $X \rightarrow Y \supset\!\!- X' \rightarrow Z$*, is the ordered pair (R_1, R_2), where $R_1 = \sigma_{X \rightarrow Y}(R)$ and $R_2 = R - R_1$.

□

Note that the horizontal decomposition of a scheme, according to $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ does not depend on X' or Z, but it induces the fd $X' \rightarrow Z$ in R_1, as follows from Lemma 4.7. In R_2, ad $X \not\twoheadrightarrow Y$ holds (but nothing is known about X' and Z).

From now on we shall assume that the set SC of constraints of a relation scheme R consists of a set $\mathcal{I}$ of ifd's and a set $\mathcal{A}$ of ad's.

When $Z - X' \subseteq Y - X$ the ifd $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ is "trivial", and can be used to simulate the horizontal decomposition based on goals.

The calculation of the constraints that hold in the selections of R is described in Section 5.3. It is very similar to the inheritance problem for cfd's.

Since fd's and cfd's are "special" ifd's, the conflict-problem of cfd's and ad's also occurs with ifd's and ad's. Before we give a conflict-detection algorithm, we show the construction of Armstrong relations for fd's, considering the presence of ifd's.

Definition 5.4 Let $\mathcal{I}$ be a set of ifd's over a set Ω of attributes. A *strong Armstrong relation for* $\mathcal{I}$ is a strong Armstrong relation for the set of all fd's that are a consequence of $\mathcal{I}$.

□

The proof of the construction of a (strong) Armstrong relation for $\mathcal{I}$ again relies on the following special set of fd's:

Definition 5.5 $FSAT_{\mathcal{I}}(X, Y)$ is the smallest possible set of fd's, such that:

1. $X \rightarrow Y \in FSAT_{\mathcal{I}}(X, Y)$.
2. If $T \rightarrow U \in FSAT_{\mathcal{I}}(X, Y)$ and $T \rightarrow U \supset\!\!- T' \rightarrow V \in \mathcal{I}$ then $T' \rightarrow V \in FSAT_{\mathcal{I}}(X, Y)$.
3. If $FSAT_{\mathcal{I}}(X, Y) \models T \rightarrow V$ then $T \rightarrow V \in FSAT_{\mathcal{I}}(X, Y)$.

□

$FSAT_{\mathcal{I}}(X, Y)$ can be constructed starting from $\{X \rightarrow Y\}$ by repeatedly trying to satisfy 2 and 3 of the definition.

Lemma 5.1 *Let $\mathcal{I}$ be a set of ifd's over Ω, $T, V, X, Y \subseteq \Omega$. $T \rightarrow V \in FSAT_{\mathcal{I}}(X, Y)$ iff $\mathcal{I} \cup \{X \rightarrow Y\} \models T \rightarrow V\}$.*

Proof This proof is omitted, since it would be almost identical to the proof of Lemma 4.1.

□

From Theorem 2.1 and the above lemma one can easily deduce that

Theorem 5.1 *Let $\mathcal{I}$ be a set of ifd's $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$ is a strong Armstrong relation for $\mathcal{I}$. In other words, for all $T, U \subseteq \Omega$ holds that $\mathcal{I} \models T \rightarrow U$ iff $T \rightarrow U \in FSAT_{\mathcal{I}}(\emptyset, \emptyset)$.*

□

From the above Lemma and Theorem immediately follows that if $Y \subseteq X$ then $FSAT_{\mathcal{I}}(X, Y)$ is the set of all fd's that are consequences of $\mathcal{I}$. We usually denote this set by $FSAT_{\mathcal{I}}(\emptyset, \emptyset)$.

Note also that for all X, Y the sets $FSAT_{\mathcal{I} \cup \{X \rightarrow Y\}}(\emptyset, \emptyset)$ and $FSAT_{\mathcal{I}}(X, Y)$ are equal.

With the Armstrong relation for $\mathcal{I}$, we can prove the following theorem and algorithm, exactly as for cfd's:

Theorem 5.2 *$\mathcal{I} \cup \mathcal{A}$ is in conflict iff for some ad $X \not\Rightarrow Y$ of $\mathcal{A}$, $\mathcal{I} \models X \rightarrow Y$ holds.*

□

Algorithm 5.1 *Conflict Detection*

Input: $\mathcal{I}, \mathcal{A}$, a set of ifd's and a set of ad's.

Output: *true* or *false*.

Method:
for each $T \not\Rightarrow U$ **in** $\mathcal{A}$ **do**
  if $\mathcal{I} \models T \rightarrow U$
    then
      return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

In the above algorithm we did not explain how to verify whether $\mathcal{I} \models T \rightarrow U$. This means verifying whether $T \rightarrow U \in FSAT_{\mathcal{I}}(\emptyset, \emptyset)$, and is part of Algorithm 4.2. The construction of the corresponding algorithm for *FSAT* in the case of ifd's is very similar, and therefor omitted. The time-complexity of that algorithm is $O(n^3 r^2)$ where $n = \sharp\mathcal{I}$ and $r = \sharp\Omega$.

The time- complexity of Algorithm 5.1 then becomes $O(n^3r^2m)$ where $m = \sharp\mathcal{A}$.

5.2 The Implication Problem for ifd's and ad's

As in the previous chapter, we solve the implication problem for ifd's and ad's by means of the tools *FSAT*, the Armstrong relation, and a set of inference rules. We solve the implication problem for ifd's (only) first. For ifd's we have the following inference rules:

$(I0)$: if $Y \subseteq X$, $Y' \subseteq X'$, $X \subseteq X''$, $X' \subseteq X''$ and $X \to Y \supset\!\!- X'' \to Z$ then $X' \to Y' \supset\!\!- X'' \to Z$.

$(I1)$: if $Z - X' \subseteq Y - X$ and $X \subseteq X'$ then $X \to Y \supset\!\!- X' \to Z$.

$(I2)$: if $X \to Y \supset\!\!- X' \to Z$ and $W \subseteq V$ then $X \to Y \supset\!\!- X'V \to ZW$.

$(I3)$: if $X \to Y \supset\!\!- X' \to X''$ and $X \to Y \supset\!\!- X'' \to Z$ then $X \to Y \supset\!\!- X' \to Z$.

$(I4)$: if $X \to Y \supset\!\!- X' \to Z$ and $X' \to Z \supset\!\!- X'' \to T$ then $X \to Y \supset\!\!- X'' \to T$.

$(I5)$: if $X \to Y \supset\!\!- X' \to Z$ and $X \to X''$ then $X'' \to Y \supset\!\!- X'X'' \to Z$. □

As fd's and cfd's are special ifd's, the use of fd's in these rules is allowed. Rules $I0$ shows the equivalence of the different possibilities to represent the same fd. The classical inference rules for fd's and the inference rules for cfd's can be deduced from $I1, \ldots, I5$ as follows:

Lemma 5.2 *The rules $I0, \ldots, I5$ are complete for cfd's (and hence also for fd's).*

Proof To prove this completeness we show how to prove $C1, \ldots, C5$ using $I0, \ldots I5$. We are free to denote fd's in any way we like, according to rule $I0$.

$C1$: There are 2 possibilities:

1. If $Z \subseteq XY$ then $X \to Y \supset\!\!- X \to Z$ holds by $I1$.
2. If $XY \to Z$, then $I0$ lets us write this fd as $X \to X \supset\!\!- XY \to Z$.
   - $X \to Y \supset\!\!- X \to X$ $(I1)$ and $X \to X \supset\!\!- XY \to Z$ induce $X \to Y \supset\!\!- XY \to Z$ by $I4$.

- $X \to Y \supset\!\!- X \to XY$ ($I1$) and $X \to Y \supset\!\!- XY \to Z$ induce $X \to Y \supset\!\!- X \to Z$ by $I3$.

$C2$: Let $X \to Y \supset\!\!- X \to Z$ and $X \to Y \supset\!\!- X \to T$. We prove $X \to Y \supset\!\!- X \to TZ$:

- $X \to Y \supset\!\!- X \to Z$ induces $X \to Y \supset\!\!- X \to XZ$, by $I2$.
- $X \to Y \supset\!\!- X \to T$ induces $X \to Y \supset\!\!- XZ \to T$, by $I2$.
- $X \to Y \supset\!\!- X \to XZ$ and $X \to Y \supset\!\!- XZ \to T$ induce $X \to Y \supset\!\!- X \to T$ by $I3$.

$C3$: Let $X \to Y \supset\!\!- X \to Z$ and $Z \to T$. We prove $X \to Y \supset\!\!- X \to T$:

- $X \to Y \supset\!\!- X \to Z$ induces $X \to Y \supset\!\!- X \to XZ$ by $I2$.
- $\emptyset \to \emptyset \supset\!\!- Z \to T$ (this is $Z \to T$) and $\emptyset \to \emptyset \supset\!\!- XZ \to Z$ ($I1$) induce $\emptyset \to \emptyset \supset\!\!- XZ \to T$ by $I3$.
- Rule $I0$ allows us to write this fd as $X \to X \supset\!\!- XZ \to T$. $X \to Y \supset\!\!- X \to X$ ($I1$) and $X \to X \supset\!\!- XZ \to T$ induce $X \to Y \supset\!\!- XZ \to T$ by $I4$.
- $X \to Y \supset\!\!- X \to XZ$ and $X \to Y \supset\!\!- XZ \to T$ induce $X \to Y \supset\!\!- X \to T$ by $I3$.

$C4$: is a special case of $I4$ where $X = X' = X''$.

$C5$: Let $X \to Y \supset\!\!- X \to Z$, $W \to Y \supset\!\!- W \to X$ and $X \to W$. We prove $W \to Y \supset\!\!- W \to Z$:

- $X \to Y \supset\!\!- X \to Z$ and $X \to W$ induce $W \to Y \supset\!\!- XW \to Z$ by $I5$.
- $W \to Y \supset\!\!- W \to X$ and $W \to X \supset\!\!- W \to XW$ ($I1$) induce $W \to Y \supset\!\!- W \to XW$ by $I4$.
- $W \to Y \supset\!\!- W \to XW$ and $W \to Y \supset\!\!- XW \to Z$ induce $W \to Y \supset\!\!- W \to Z$ by $I3$.

□

Note that for cfd's a rather strange rule $C5$ was necessary to change the left hand side of the fd's, i. e. to change $X \to Y$ into $W \to Y$. With the ifd's this is much easier: rule $I5$ is in fact the "equivalent" to $C5$.

Note also that rules $C2$ and $I2$ differ. This is a variation on the classical theme: augmentation (like I2) versus union rule (C2). One can easily prove that rule $I2$ can be replaced by:

$(I2')$: if $X \to Y \supset\!\!- X' \to Z$ and $X \to Y \supset\!\!- X' \to T$ then $X \to Y \supset\!\!- X' \to ZT$.

Rule $I2'$ can be easily proved:

- $X \to Y \supset\!\!- X' \to Z$ induces $X \to Y \supset\!\!- X' \to X'Z$ by $I2$.
- $X \to Y \supset\!\!- X' \to T$ induces $X \to Y \supset\!\!- X'Z \to ZT$ by $I2$.
- $X \to Y \supset\!\!- X' \to X'Z$ and $X \to Y \supset\!\!- X'Z \to ZT$ induce $X \to Y \supset\!\!- X' \to ZT$ by $I3$.

Also, rule $I2$ can be easily proved using $I2'$:

- $X \to Y \supset\!\!- X' \to Z$ and $X \to Y \supset\!\!- X'V \to X'$ ($I1$) induce $X \to Y \supset\!\!- X'V \to Z$ by $I3$.
- $W \subseteq X'V$ induces $X \to X \supset\!\!- X'V \supset\!\!- W$ by $I1$.
- $X \to Y \supset\!\!- X \to X$ ($I1$) and $X \to X \supset\!\!- X'V \supset\!\!- W$ induce $X \to Y \supset\!\!- X'V \supset\!\!- W$ by $I4$.
- Finally, $X \to Y \supset\!\!- X'V \to Z$ and $X \to Y \supset\!\!- X'V \to W$ induce $X \to Y \supset\!\!- X'V \to ZW$ by $I2'$.

Since $C5$ is not needed for the completeness for fd's and $I5$ is only used in the proof of $C5$, we conclude that $I0 \ldots I4$ are complete for fd's.

Theorem 5.3 *The rules $I0 \ldots I5$ are sound.*

Proof This is very straightforward, and left to the reader. The proof follows the same argumentation as in Theorem 4.3. □

The proof of the completeness of $I0 \ldots I5$ for ifd's relies on similar properties of $FSAT_{\mathcal{I}}(X, Y)$ to those given in Lemmas 4.3 and 4.4.

Lemma 5.3 *If $P \to Q \in FSAT_{\mathcal{I}}(X, Y)$ then $\mathcal{I} \vdash P \to Q$ or $\mathcal{I} \vdash P \to X$.*

Proof Let $\mathcal{I} = \{X_i \to Y_i \supset\!\!- X'_i \to Z_i : i = 1 \ldots n\}$.
For $P \to Q = X \to Y$ this property is trivial. We prove that it remains valid during the construction of $FSAT_{\mathcal{I}}(X, Y)$ by repeatedly trying to satisfy 2 and 3 of Definition 5.5.

- Let part 2 be the reason that $P \to Q$ is in $FSAT_{\mathcal{I}}(X, Y)$. Then for some i, $P = X'_i$ and $Q = Z_i$ and by induction $\mathcal{I} \vdash X_i \to Y_i$ or $\mathcal{I} \vdash X_i \to X$.
  - If $\mathcal{I} \vdash X_i \to Y_i = X_i \to X_i \supset\!\!- X_i \to Y_i$ then $X_i \to X_i \supset\!\!- X_i \to Y_i$ and $X_i \to Y_i \supset\!\!- X'_i \to Z_i$ induce $X_i \to X_i \supset\!\!- X'_i \to Z_i$ by $I4$. Hence $\mathcal{I} \vdash X'_i \to Z_i = P \to Q$.
  - If $\mathcal{I} \vdash X_i \to X$ then $X'_i \to X$ holds by augmentation, (which is a consequence of $I0 \ldots I4$) , since $X \subseteq X'$.

- Let part 3 be the reason for $P \rightarrow Q$ to be in $FSAT_{\mathcal{I}}(X,Y)$. Then there are three possibilities (corresponding to the three well known inference rules for fd's):
  1. If $Q \subseteq P$, then $\mathcal{I} \vdash P \rightarrow P \supset\!\!- P \rightarrow Q = P \rightarrow Q$ by $I1$.
  2. If $P = P_1P_2$, $Q = Q_1Q_2$, $Q_2 \subseteq P_2$ and $P_1 \rightarrow Q_1 \in FSAT_{\mathcal{I}}(X,Y)$ then $\mathcal{I} \vdash P_1 \rightarrow Q_1$ or $\mathcal{I} \vdash P_1 \rightarrow X$ by induction, hence $\mathcal{I} \vdash P_1P_2 \rightarrow Q_1Q_2 = P \rightarrow Q$ or $\mathcal{I} \vdash P_1P_2 \rightarrow X = P \rightarrow X$ by augmentation.
  3. If $P \rightarrow O$ and $O \rightarrow Q \in FSAT_{\mathcal{I}}(X,Y)$ then $\mathcal{I} \vdash P \rightarrow O$ or $P \rightarrow X$ and $\mathcal{I} \vdash O \rightarrow Q$ or $O \rightarrow X$ by induction. Using the transitivity rule for fd's (a consequence of $I0 \ldots I4$) we obtain $\mathcal{I} \vdash P \rightarrow Q$ or $\mathcal{I} \vdash P \rightarrow X$.

The proof is completed by remarking that reflexivity, augmentation and transitivity are complete for fd's.

□

Lemma 5.4 *If $\mathcal{I} \models X' \rightarrow X$ then $\mathcal{I} \vdash X \rightarrow Y \supset\!\!- XX' \rightarrow Z$ iff $X' \rightarrow Z \in FSAT_{\mathcal{I}}(X,Y)$. (We write XX' to make sure that $X \subseteq XX'$).*

Proof If $\mathcal{I} \vdash X \rightarrow Y \supset\!\!- XX' \rightarrow Z$ then $\mathcal{I} \models X \rightarrow Y \supset\!\!- XX' \rightarrow Z$ because $I0 \ldots I5$ are sound. Hence $\mathcal{I} \cup \{X \rightarrow Y\} \models XX' \rightarrow Z$ which implies that $XX' \rightarrow Z \in FSAT_{\mathcal{I}}(X,Y)$ by Lemma 5.1. $XX' \rightarrow Z$ and $X' \rightarrow X$ induce $X' \rightarrow Z$ hence $X' \rightarrow Z \in FSAT_{\mathcal{I}}(X,Y)$.

For the converse we give the same kind of proof as in the previous lemma.

Let $\mathcal{I} = \{X_i \rightarrow Y_i \supset\!\!- X_i' \rightarrow Z_i \mid i = 1 \ldots n\}$.

If $X' \rightarrow Z = X \rightarrow Y$ then $\mathcal{I} \vdash X \rightarrow Y \supset\!\!- XX' \rightarrow Z$ by $I1$.

We prove that the property of the lemma remains valid during the construction of $FSAT_{\mathcal{I}}(X,Y)$ by repeatedly trying to satisfy 2 and 3 of Definition 5.5.

- Let part 2 be the reason for $X' \rightarrow Z$ to be in $FSAT_{\mathcal{I}}(X,Y)$. Then for some i, $X' = X_i'$ and $Z = Z_i$ and $X_i \rightarrow Y_i \in FSAT_{\mathcal{I}}(X,Y)$. By Lemma 5.3 $\mathcal{I} \vdash X_i \rightarrow Y_i$ (and $X_i' \rightarrow Z_i$) or $\mathcal{I} \vdash X_i \rightarrow X$.
  1. If $\mathcal{I} \vdash X_i \rightarrow Y_i$ (and hence $\mathcal{I} \vdash X_i' \rightarrow Z_i = X' \rightarrow Z$ and hence $XX' \rightarrow Z$ by augmentation) then $\mathcal{I} \vdash X \rightarrow Y \supset\!\!- XX' \rightarrow Z$ by $I4$ (on $X \rightarrow Y \supset\!\!- X \rightarrow X$ and $X \rightarrow X \supset\!\!- XX' \rightarrow Z$).

2. If $\mathcal{I} \vdash X_i \to X$ then $X \to Y \supset\!\!- XX_i \to Y_i$ (holding by induction), and $XX_i \to Y_i \supset\!\!- XX_i' \to Z_i$ induce $X \to Y \supset\!\!- XX_i' \to Z_i = X \to Y \supset\!\!- XX' \to Z$ by $I4$. ($XX_i \to Y_i \supset\!\!- XX_i' \to Z_i$ is derived by $I5$ from $X_i \to Y_i \supset\!\!- X_i' \to Z_i$ and $X_i \to X$.)

- Let part 3 be the reason for $X' \to Z$ to be in $FSAT_{\mathcal{I}}(X, Y)$. There are three possibilities:
  1. If $Z \subseteq X'$ then $\mathcal{I} \vdash X \to Y \supset\!\!- XX' \to Z$ by $I1$.
  2. If $X' = X''X'''$, $Z = Z'Z''$, $Z'' \subseteq X'''$ and $X'' \to Z' \in FSAT_{\mathcal{I}}(X, Y)$ then by induction (if $X'' \to X$) or by $I0$ (if $X'' \to Z'$) we know that $\mathcal{I} \vdash X \to Y \supset\!\!- XX'' \to Z'$. Since $Z'' \subseteq X'''$ rule $I2$ generates $X \to Y \supset\!\!- XX''X''' \to Z'Z''$.
  3. If $X' \to U$ and $U \to Z \in FSAT_{\mathcal{I}}(X, Y)$ then $\mathcal{I} \vdash X' \to U$ or $X' \to X$ and $\mathcal{I} \vdash U \to Z$ or $U \to X$, by Lemma 5.3.
     (a) If $\mathcal{I} \vdash X' \to U$ and $\mathcal{I} \vdash U \to Z$ then $\mathcal{I} \vdash X' \to Z$ by transitivity, hence also $XX' \to Z$, written as $X \to Y \supset\!\!- XX' \to Z$ by $I0$.
     (b) If $\mathcal{I} \vdash X' \to X$ then $\mathcal{I} \vdash X \to Y \supset\!\!- XX' \to U$ by induction.
        - If also $\mathcal{I} \vdash U \to Z$ then $\mathcal{I} \vdash X \to Y \supset\!\!- XX' \to Z$ by $I0$, $I2$ and $I3$.
        - If on the other hand $\mathcal{I} \vdash U \to X$ then $\mathcal{I} \vdash X \to Y \supset\!\!- XU \to Z$ by induction, and then $\mathcal{I} \vdash X \to Y \supset\!\!- XX' \to Z$ by $I2$ and $I4$ on $X \to Y \supset\!\!- XX' \to U$ and $X \to Y \supset\!\!- XU \to Z$.

□

Using these two lemmas the link between the implication problem for ifd's and the set $FSAT_{\mathcal{I}}(X, Y)$ can be easily established:

Theorem 5.4 *Let $X \subseteq X'$. $\mathcal{I} \models X \to Y \supset\!\!- X' \to Z$ iff $X' \to Z \in FSAT_{\mathcal{I}}(X, Y)$.*

Proof Let $\mathcal{I} = \{X_i \to Y_i \supset\!\!- X_i' \to Z_i \mid i = 1 \ldots n\}$.
If $\mathcal{I} \models X \to Y \supset\!\!- X' \to Z$ then (by the proof of Lemma 5.1) $X' \to Z \in FSAT_{\mathcal{I}}(X, Y)$. Since $X \subseteq X'$, $X' \to X$ is trivial, hence by Lemma 5.4 $\mathcal{I} \vdash X \to Y \supset\!\!- X' \to Z$. $\mathcal{I} \vdash X \to Y \supset\!\!- X' \to Z$ implies $\mathcal{I} \models X \to Y \supset\!\!- X' \to Z$ since $I0 \ldots I5$ are sound.

□

From the above proof we immediately deduce:

Corollary 5.1 *$I0 \ldots I5$ are complete for ifd's.*

□

Theorem 5.4 shows that the implication problem for ifd's behaves like that of cfd's. This means that one can regard an ifd as an ad-hoc implication between two fd's. The construction of $FSAT_{\mathcal{I}}(X, Y)$ indeed consists of generating the "closure" of a set of fd's, using the normal inference rules (in part 3) and some "additional" rules (the ifd's in part 2). However, this "nice" property seems not to hold any longer if the condition $X \subseteq X'$ in the definition of $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ is dropped. In the following two chapters we shall show more general constraints than the ifd's, but even these constraints do not allow arbitrary sets of attributes in $X \rightarrow Y \supset\!\!- X' \rightarrow Z$. From Theorem 5.4 one can deduce a membership algorithm which verifies whether $X' \rightarrow Z$ is in $FSAT_{\mathcal{I}}(X, Y)$, in $O(n^3 r^2)$ time, where $n = \sharp\mathcal{I}$ and $r = \sharp\Omega$. The algorithm is omitted because it is very similar to Algorithm 4.2, the membership algorithm for cfd's.

We now present a set of inference rules for mixed ifd's and ad's:

$(IA1)$: if $X \rightarrow Y$, $X \rightarrow Z \supset\!\!- X' \rightarrow T$ and $X' \not\Rightarrow T$ then $Y \not\Rightarrow Z$.
$(IA2)$: if $X \subseteq X'$ and $X \not\Rightarrow Y$ then $X \rightarrow Y \supset\!\!- X' \rightarrow Z$ for all $Z \subseteq \Omega$. □

Lemma 5.5 *The rules $IA1$ and $IA2$ are sound.*

Proof We only prove $IA1$. $IA2$ is rather trivial.
Consider an arbitrary Y-complete, Y-unique set s of tuples (in an instance), satisfying $X \rightarrow Y$, $X \rightarrow Z \supset\!\!- X' \rightarrow T$ and $X' \rightarrow T$). Because of $X \rightarrow Y$ s is also X-complete, and because of $X \subseteq X'$ (hence $X' \rightarrow X$) s is also X'-complete. Since $X' \not\Rightarrow T$ holds in this X-complete set of tuples, $X \not\Rightarrow Z$ must hold too (otherwise $X \rightarrow Z \supset\!\!- X' \rightarrow T$ would be violated). Hence in s there are several Z-values, for the only Y-value, which means that $Y \not\Rightarrow Z$ holds in s.

□

Since cfd's are special ifd's we should be able to prove the rules for cfd's and ad's ($CA1$ and $CA2$) using $I0, \ldots, I5, IA1$ and $IA2$. (Otherwise $I0, \ldots, I5, IA1$ and $IA2$ cannot be complete for ifd's and ad's.)

Remark 5.2 *Rules $IA1$ and $IA2$ imply $CA1$ and $CA2$.*

Proof Since $CA1$ is a special case of $IA1$, with $X = X'$ and $CA2$ is a special case of $IA2$, with $X = X'$, this is trivial.

□

Note that rules $I0, \ldots, I4$ and $CA1$ are complete for fd's and ad's only.

Theorem 5.5 *Let $\mathcal{I} \cup \mathcal{A}$ be not in conflict, $X, Y \subseteq \Omega$. Then $\mathcal{I} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$ iff $\mathcal{I} \cup \mathcal{A} \cup \{X \to Y\}$ is in conflict.*

Proof The only-if part is trivial.
For the if-part, suppose that $\mathcal{I} \cup \mathcal{A} \cup \{X \to Y\}$ is in conflict. Then by Theorem 5.2 $\mathcal{I} \cup \{X \to Y\} \models \{T \to U\}$ for some $T \not\twoheadrightarrow U \in \mathcal{A}$. We prove that $\mathcal{I} \cup \{T \not\twoheadrightarrow U\} \models X \not\twoheadrightarrow Y$. (We actually prove $\vdash$ because we use our inference rules.) We give a new and shorter proof than that in [13].

$\mathcal{I} \cup \{X \to Y\} \models T \to U$ implies $T \to U \in FSAT_{\mathcal{I}}(X, Y)$ by Lemma 5.1. By Lemma 5.3 $\mathcal{I} \models T \to X$, (since $\mathcal{I} \not\models T \to U$ unless $\mathcal{I} \cup \mathcal{A}$ was in conflict), hence by Lemma 5.4 $\mathcal{I} \models X \to Y \supset\!\!- XT \to U$.

- $T \not\twoheadrightarrow U$, $T \to U \supset\!\!- T \to U$ and $T \to XT$ induce $XT \not\twoheadrightarrow U$ by $IA1$.
- $XT \not\twoheadrightarrow U$ and $X \to Y \supset\!\!- XT \to U$ induce $X \not\twoheadrightarrow Y$ by $IA1$.

This shows that $\mathcal{I} \cup \{T \not\twoheadrightarrow U\} \models X \not\twoheadrightarrow Y$.

□

The proof of the above theorem immediately implies that:

Corollary 5.2 *The rules $I0, \ldots, I5$ and $IA1$ are complete for the inference of ad's from a set of ifd's and ad's.*

□

Corollary 5.3 *The rules $I0, \ldots, I5$ and $IA1$ are complete for the inference of ad's from a set of cfd's and ad's.*

□

Using Theorem 5.2 and 5.5 one can easily prove that Algorithm 4.3, with $\mathcal{C}$ replaced by $\mathcal{I}$, is correct for ifd's and ad's. It still takes $O(n^3 r^2 m)$ time, where $n = \sharp\mathcal{I}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$.

As Theorem 5.5 is very similar to Theorem 4.6, the following theorem solves the implication problem for ifd's (with the influence of ad's) as in Theorem 4.7. The proof is omitted since it is exactly the same as for Theorem 4.7, with $\mathcal{C}$ replaced by $\mathcal{I}$.

Theorem 5.6 *Let $\mathcal{I} \cup \mathcal{A}$ be not in conflict. Then $\mathcal{I} \cup \mathcal{A} \models X \to Y \supset\!\!- X' \to Z$ (with $X \subseteq X'$) iff $\mathcal{I} \models X \to Y \supset\!\!- X' \to Z$ or $\mathcal{I} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$.*

□

An immediate consequence of the above theorem is:

Corollary 5.4 *The rules $I0, \ldots, I5, IA1$ and $IA2$ are complete for the inference of ifd's from a set of ifd's and ad's.*

□

This also means:

Corollary 5.5 *The rules* $I0, \ldots, I5, IA1$ *and* $IA2$ *are complete for the inference of cfd's from a set of ifd's and ad's.*

□

Algorithm 4.3, with $\mathcal{C}$ replaced by $\mathcal{I}$, is a membership algorithm for ifd's (considering the presence of ad's), and takes $O(n^3 r^2 m)$ time, with $m = \sharp\mathcal{I}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$.

Rules $IA1$ and $IA2$ are very similar to $CA1$ and $CA2$. Still, $IA1$ and $IA2$ are complete (together with $I0 \ldots I5$) whereas $CA1$ and $CA2$ are not. The reason is that if $\mathcal{C} \cup \{X \to Y\} \models T \to U$ (and $\mathcal{C} \not\models T \to U$) Lemma 5.4 shows that $\mathcal{C} \models X \to Y \supset\!\!- XT \to U$. This ifd cannot be expressed by means of cfd's.

We illustrate the completeness of $I0, \ldots, I5, IA1$ and $IA2$ with Example 4.1:

Example 5.1 Recall Example 4.1. $\mathcal{C} = \{AB \to C \supset\!\!- AB \to E\ ,\ A \to D \supset\!\!- A \to F\}$ and $\mathcal{A} = \{AB \not\twoheadrightarrow EF\}$. We show that $\mathcal{C} \cup \{AB \not\twoheadrightarrow EF\} \vdash A \not\twoheadrightarrow CD$, using the rules for ifd's and ad's.

$A \to CD \supset\!\!- AB \to C$ is trivial. With $AB \to C \supset\!\!- AB \to E$ it induces $A \to CD \supset\!\!- AB \to E$ by $I4$.

$A \to CD \supset\!\!- A \to D$ is trivial. With $A \to D \supset\!\!- A \to F$ it induces $A \to CD \supset\!\!- A \to F$ by $I4$. With the trivial $A \to F \supset\!\!- AB \to F$ it induces $A \to CD \supset\!\!- AB \to F$ by $I4$.

From $A \to CD \supset\!\!- AB \to E$ and $A \to CD \supset\!\!- AB \to F$ we infer $A \to CD \supset\!\!- AB \to EF$ by $I2'$. Rule $IA1$ then generates $A \not\twoheadrightarrow CD$ from $AB \not\twoheadrightarrow EF$.

The problem for cfd's is that the constraint $A \to CD \supset\!\!- AB \to EF$, which is a consequence of $\mathcal{C}$, cannot be expressed by means of cfd's. This indicates that it is unlikely that a complete set of inference rules for cfd's and ad's would exist (which only uses cfd's and ad's).

□

5.3 The Inheritance of ifd's and ad's.

The membership problem must be solved in order to be able to decide whether a decomposition according to an ifd is trivial or not (i. e. whether for $X \to Y \supset\!\!- X' \to Z$ $X \to Y$ or $X \not\Rightarrow Y$ holds). When decomposing the subschemes further on (using other ifd's) one must know which dependencies already hold in these subschemes. This *inheritance problem* is rather similar to the inheritance problem for cfd's and ad's. Where possible, references are made to Section 4.3 instead of repeating very similar proofs.

Notation 5.1 In the sequel we treat the horizontal decomposition of a scheme $R = (\Omega, \mathcal{I} \cup \mathcal{A})$, according to $X \to Y \supset\!\!- X' \to Z \in \mathcal{I}$, into the schemes $R_1 = \sigma_{X \to Y}(R) = (\Omega, \mathcal{I}_1 \cup \mathcal{A}_1)$, and $R_2 = \sigma_{X \not\Rightarrow Y}(R) = (\Omega, \mathcal{I}_2 \cup \mathcal{A}_2)$. We assume that $\mathcal{I} \cup \mathcal{A}$ is not in conflict, $\mathcal{I} \cup \mathcal{A} \not\models X \to Y$ and $\mathcal{I} \cup \mathcal{A} \not\models X \not\Rightarrow Y$. We again do not consider complete sets of dependencies. The sets of dependencies we shall generate for the subschemes are only generating for the set of all dependencies, holding in the subschemes. □

Since fd's cannot be violated by taking a selection of a relation, Remark 4.3 also applies if the decompositions are based on ifd's instead of cfd's.

For ifd's and ad's similar inclusions than in Lemma 4.6 hold, and they can be proved similarly:

Lemma 5.6 *Using Notation 5.1 we have:*

- $\mathcal{I}_1 \subseteq \{T \to U \supset\!\!- T' \to V \mid \mathcal{I} \cup \mathcal{A} \cup \{X \to Y\} \models T \to U \supset\!\!- T' \to V\}$.
- $\mathcal{I}_2 \subseteq \{T \to U \supset\!\!- T' \to V \mid \mathcal{I} \cup \mathcal{A} \cup \{X \not\Rightarrow Y\} \models T \to U \supset\!\!- T' \to V\}$.
- $\mathcal{A}_1 \subseteq \{T \not\Rightarrow U \mid \mathcal{I} \cup \mathcal{A} \cup \{X \to Y\} \models T \not\Rightarrow U\}$.
- $\mathcal{A}_2 \subseteq \{T \not\Rightarrow U \mid \mathcal{I} \cup \mathcal{A} \cup \{X \not\Rightarrow Y\} \models T \not\Rightarrow U\}$.

□

Some dependencies that are inherited by both R_1 and R_2 are given by the following lemma, which can be proved exactly as Lemma 4.7:

Lemma 5.7 *Using Notation 5.1, an ad holds in both R_1 and R_2 if $\mathcal{I} \cup \mathcal{A} \models T \not\Rightarrow U$ and $\mathcal{I} \models T \to X$. An ifd $T \to U \supset\!\!- T' \to V$ holds in both R_1 and R_2 if $\mathcal{I} \cup \mathcal{A} \models T \to U \supset\!\!- T' \to V$ and $\mathcal{I} \models T \to U$ or $\mathcal{I} \models T \to X$.*

□

Notation 5.2 We denote the set of the ad's of $\mathcal{A}$ that are inherited because of Lemma 5.7, by $\hat{\mathcal{A}}$. The set of the ifd's of $\mathcal{I}$ that are inherited because of Lemma 5.7, or that are trivial, is denoted by $\hat{\mathcal{I}}$. We include the trivial ifd's because they represent the goals of Chapters 2 and 3. They are of no importance for the inheritance problem, and are neglected in the sequel.

□

The proof of the inheritance of ifd's and ad's relies on the construction of a special instance with the following property:

Lemma 5.8 *Consider a set $\mathcal{I} \cup \mathcal{A}$, not being in conflict. Let s be an instance in which $\mathcal{I} \cup \mathcal{A}$ holds.*

If $\mathcal{I} \cup \mathcal{A} \cup \{P \not\twoheadrightarrow Q\}$ is not in conflict, then there exists an instance r, in which $\mathcal{I} \cup \mathcal{A}$ still holds, which contains s as a subset, and in which the ad $P \not\twoheadrightarrow Q$ holds (and hence also every ifd $P \rightarrow Q \supset\!\!- P' \rightarrow O$ where $P \subseteq P'$).

Proof This lemma can be proved using the construction of Lemma 4.8. The only additional fact we have to prove is that in $r = s \cup s'$ ($s' = Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$), $\mathcal{I}$ still holds.

Let $T \rightarrow U \supset\!\!- T' \rightarrow V \in \mathcal{I}$. If $T \not\subset \overline{P}$ and $\mathcal{I} \models T \rightarrow U$ then $T \rightarrow U$ and $T' \rightarrow V$ still hold in r since s and s' both satisfy $T \rightarrow U$ and $T' \rightarrow V$ and since s and s' have no common T-value (and hence also no common T' value). If $T \subseteq \overline{P}$ and $\mathcal{I} \models T \rightarrow U$ then $U \subseteq \overline{P}$ too. s and s' have a common $\overline{P}$-value, so $T \rightarrow U$ still holds in r (since the common T-value has the same U-value in s and s', since it is part of the common $\overline{P}$-value). If $T' \subseteq \overline{P}$ then also $V \subseteq \overline{P}$, hence $T' \rightarrow V$ holds in r for the same reason as for $T \rightarrow U$. If $T' \not\subset \overline{P}$ then $T' \rightarrow V$ holds in r since s and s' have no common T'–value.
If for $T \rightarrow U \supset\!\!- T' \rightarrow V \in \mathcal{I}$, $\mathcal{I} \cup \mathcal{A} \models T \not\twoheadrightarrow U$, then $T \not\twoheadrightarrow U$ holds in r since it holds in both s and s', and hence $T \rightarrow U \supset\!\!- T' \rightarrow V$ holds by $IA2$.

□

As in Lemma 4.9 we can repeat the construction of Lemma 5.8 to obtain:

Lemma 5.9 *Let $\mathcal{I} \cup \mathcal{A}$ be not in conflict. Let s be an instance in which $\hat{\mathcal{I}} \cup \hat{\mathcal{A}}$ holds. Then there exists an instance r, containing s as a subset, and in which $\mathcal{I} \cup \mathcal{A}$ holds.*

□

The solution of the inheritance problem is proved in a very similar way as Theorem 4.8:

Theorem 5.7 *Using Notation 5.1 and 5.2, an ifd or ad must hold in* R_1 *(resp.* R_2*) iff it is a consequence of* $\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \to Y\}$ *(resp.* $\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\}$*).*

Proof From Lemma 5.6 and 5.7 it follows that $(\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \to Y\})^* \subseteq (\mathcal{I}_1 \cup \mathcal{A}_1)^* \subseteq (\mathcal{I} \cup \mathcal{A} \cup \{X \to Y\})^*$ and $(\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\})^* \subseteq (\mathcal{I}_2 \cup \mathcal{A}_2)^* \subseteq (\mathcal{I} \cup \mathcal{A} \cup \{X \not\twoheadrightarrow Y\})^*$.

The proof has to cover 4 cases: an ifd and an ad, in R_1 and in R_2. We only prove the case for an ad in R_1. The other cases are similar and also similar to the proof of Theorem 4.8.

Consider an ad $T \not\twoheadrightarrow U \in (\mathcal{I} \cup \mathcal{A} \cup \{X \to Y\})^* - (\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \to Y\})^*$. We prove that $T \not\twoheadrightarrow U \notin (\mathcal{I}_1 \cup \mathcal{A}_1)^*$.

Since $T \not\twoheadrightarrow U \notin (\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \to Y\})^*$, $\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \to Y\} \cup \{T \to U\}$ is not in conflict, by Theorem 5.5. Hence there exists an instance s in which $\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \to Y\} \cup \{T \to U\}$ holds. By the construction of Lemma 5.9 an instance r can be build which contains s and in which $\mathcal{I} \cup \mathcal{A}$ holds. In this construction (explained in the proof of Lemma 4.8) a number of modified copies of $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$ are added to s. Since $X \not\twoheadrightarrow Y$ holds in $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$, and since $\mathcal{I} \not\models T \to X$ (by the definition of $\hat{\mathcal{A}}$), $r_1 = \sigma_{X \to Y}(r) = s$. Hence in r_1, $T \not\twoheadrightarrow U$ does not hold, which means that $T \not\twoheadrightarrow U \notin (\mathcal{I}_1 \cup \mathcal{A}_1)^*$.

□

From Theorem 5.7 one can easily deduce an algorithm which calculates generating sets for the inherited dependencies, in $O(n^4r^2 + n^3m^2r^2)$ time, where $n = \sharp\mathcal{I}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$. (This is n times the complexity of an ifd-membership test plus m times the complexity of an ad-membership test).

Algorithm 5.2 *A Horizontal Decomposition Step with ifd's and ad's*

Input: $R = (\Omega, \mathcal{I} \cup \mathcal{A})$ and an ifd $X \to Y \supset\!\!- X' \to Z \in \mathcal{I}$. $\mathcal{I} \cup \mathcal{A}$ is assumed not to be in conflict, $\mathcal{I} \cup \mathcal{A} \not\models X \to Y$ and $\mathcal{I} \cup \mathcal{A} \not\models X \not\twoheadrightarrow Y$.

Output: An ordered pair of schemes $(R_1 = (\Omega, \mathcal{I}_1 \cup \mathcal{A}_1), R_2 = (\Omega, \mathcal{I}_2 \cup \mathcal{A}_2))$, being the decomposition of R according to $X \to Y \supset\!\!- X' \to Z$.

Method:

var $\hat{\mathcal{I}}, \hat{\mathcal{I}}_1, \hat{\mathcal{I}}_2$: set of ifd's $:= \emptyset$
$\hat{\mathcal{A}}$: set of ad's $:= \emptyset$

begin
for each $T \to U \supset\!\!- T' \to V \in \mathcal{I}$ **do**
if $\mathcal{I} \models T \to U$ or $\mathcal{I} \models T \to X$ or $Z - X' \subseteq Y - X$
then
$\hat{\mathcal{I}} := \hat{\mathcal{I}} \cup \{T \to U \supset\!\!- T' \to V\}$
od
for each $T \not\twoheadrightarrow U \in \mathcal{A}$ **do**
if $\mathcal{I} \models T \to X$
then
$\hat{\mathcal{A}} := \hat{\mathcal{A}} \cup \{T \not\twoheadrightarrow U\}$
od
for each $T \to U \supset\!\!- T' \to V \in \hat{\mathcal{I}}$ **do**
if $\hat{\mathcal{I}} \cup \{X \to Y\} \cup \{X' \to Z\} \not\models T \to U$ and
$\hat{\mathcal{I}} \cup \{X \to Y\} \cup \{X' \to Z\} \cup \hat{\mathcal{A}} \not\models T \not\twoheadrightarrow U$
then
$\hat{\mathcal{I}}_1 := \hat{\mathcal{I}}_1 \cup \{T \to U \supset\!\!- T' \to V\}$
od
for each $T \to U \supset\!\!- T' \to V \in \hat{\mathcal{I}}$ **do**
if $\hat{\mathcal{I}} \not\models T \to U$ and $\hat{\mathcal{I}} \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\} \not\models T \not\twoheadrightarrow U$
then
$\hat{\mathcal{I}}_2 := \hat{\mathcal{I}}_2 \cup \{T \to U \supset\!\!- T' \to V\}$
od
return$(R_1 = (\Omega, \hat{\mathcal{I}}_1 \cup \{X \to Y\} \cup \{X' \to Z\} \cup \hat{\mathcal{A}}),$
$R_2 = (\Omega, \hat{\mathcal{I}}_2 \cup \hat{\mathcal{A}} \cup \{X \not\twoheadrightarrow Y\}))$
end

□

This algorithm again treats the trivial ifd's in such a way that the decomposition that is generated in case of all trivial ifd's is equivalent to the inherited decomposition into *HNF*.

5.4 The "Imposed" Normal Form

In this section we illustrate an algorithm for horizontal decomposition of a relation scheme, according to its ifd's. The algorithm decomposes the scheme (and subschemes) until no subscheme can be decomposed any further. This is formalized by defining a normal form:

Definition 5.6 A scheme $R = (\Omega, \mathcal{I} \cup \mathcal{A})$ is said to be in *Imposed Normal Form (INF)* iff for all $X \rightarrow Y \supset\!\!- X' \rightarrow Z \in \mathcal{I}$ either $\mathcal{I} \models X \rightarrow Y$ or $\mathcal{I} \cup \mathcal{A} \models X \not\twoheadrightarrow Y$.
A decomposition $(R_1, \ldots, R_n)$ is in *INF* iff all the R_i, $i = 1 \ldots n$, are in *INF*.

□

In Algorithm 5.2 we have defined that the trivial ifd's are inherited in the same way as in the inherited decomposition steps of Section 3.3. Hence the horizontal decomposition into Imposed Normal Form is equivalent to the inherited decomposition into Horizontal Normal Form if only trivial ifd's are given. Furthermore, if only cfd's are given, the decomposition into *INF* is equivalent to the decomposition into *CNF*, described in Section 4.4.

From this definition the construction of a decomposition algorithm, which decomposes a relation scheme according to an ifd, and then decomposes the subschemes further on, until all subschemes are in *INF*, is straightforward.

To illustrate the horizontal decomposition into *INF*, we modify Example 2.1:

Example 5.2 Recall Example 2.1. In Section 5.1 we showed that the relation scheme $STAFF$ is likely to satisfy $emp \rightarrow job \supset\!\!- emp\ man\ dep\ div \rightarrow sal$. In Section 4.1 we suggested the cfd $emp \rightarrow job\ man\ dep\ div \supset\!\!- emp \rightarrow sal$. To be consistent with Example 4.2 we also consider the goal $man \stackrel{?}{\not\twoheadrightarrow} div$, which is represented by the ifd $man \rightarrow div \supset\!\!- man \rightarrow div$. The goal $emp \stackrel{?}{\not\twoheadrightarrow} job$ is no longer needed, since it already is represented by our ifd $emp \rightarrow job \supset\!\!- emp\ man\ dep\ div \rightarrow sal$. Figure 5.1 shows a decomposition tree for the decomposition of $STAFF$ into *INF*. The sets of ifd's and ad's we give are not exactly as generated by Algorithm 5.2, to keep the figure smaller.

Recall Table 4.1. The reader is invited to verify that this table satisfies the given ifd's for $STAFF$. After decomposing *staff* according to Figure 5.1 the subinstances that are obtained correspond to the subinstances of Example 4.2. The main reason that we obtain the same subinstances is that *staff* satisfies the fd $emp\ job\ man\ dep\ div \rightarrow sal$, which is not given. This implies that in $staff_{12}$, the fd $emp\ man\ dep\ div \rightarrow sal$ accidently holds, in Figure 4.1.

□

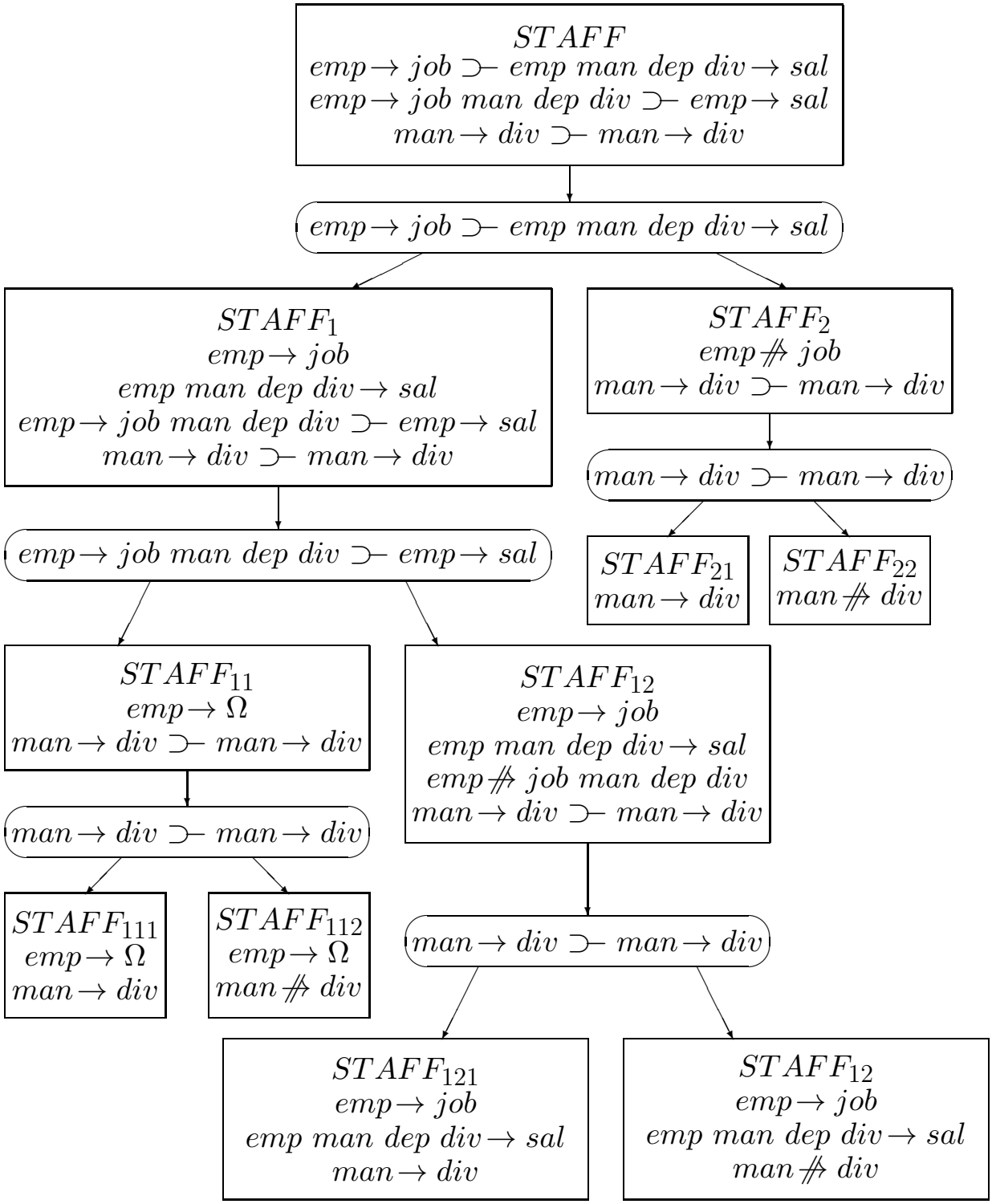


Figure 5.1: A decomposition tree for $STAFF$, into INF.

Chapter 6

Decomposition with fdi's

The horizontal decomposition according to ifd's is capable of representing the decomposition with goals, and with cfd's, and has nicer properties than these previous approaches: the decomposition is based on constraints (unlike the goals) and for these constraints we have a sound and complete axiom system (unlike for cfd's and ad's). However, they may fail to be able to express the constraints that could lead to horizontal decompositions, because in the ifd $X \to Y \supset\!\!- X' \to Z$ the sets of tuples in which the fd $X \to Y$ must hold are the X-unique X-complete sets of tuples, for the same X. This is not at all obvious, especially not with the ifd's in which the second (or implied) fd has a different left hand side: X'. Also the reason why $X \subseteq X'$ or $X' \to X$ must hold is not very clear (except the technical reason).

This chapter covers [15]. In this chapter we remove the restriction that $X \to Y$ must hold in X-unique X-complete sets of tuples, to allow a distinction between the fd we want and the kind of sets we want it to be satisfied in.

6.1 Functional Dependency Implications

Let us first recall Example 2.1. The cfd $emp \to job\ man\ dep\ div \supset\!\!- emp \to sal$ and the ifd $emp \to job \supset\!\!- emp\ man\ dep\ div \to sal$ may generate useful decompositions, but they do not reflect the structure of the company, represented by the relation $STAFF$. The company is divided into several *div*isions, each having a number of *dep*artments, each having a

number of *emp*loyees. The major structural element in this database is the *div*ision. So for both semantic reasons (keeping tuples of the same division together) as for technical reasons (keeping tuples of the same division in the same relation easies the distribution among different computers, such that the computer of each division contains the tuples of that division).

Some constraints about divisions are:

- If in a *div*ision every *emp*loyee works for only one *dep*artment and *man*ager, at only one *job*, then he has only one *sal*ary (for that *div*ision).
- If in a *div*ision every *dep*artment treats only one *job*, then (this *div*ision is so large that) every *emp*loyee has only one *job*, *sal*ary and *man*ager for this *div*ision.
- If in a *div*ision every *dep*artment treats only one *job*, then every *man*ager has only one *dep*artment (for that *div*ision).

We shall not discuss the choice of these constraints. Others can equally likely be satisfied, in other companies.

We now show how to express this constraint formally, and how it leads to a horizontal decomposition.

Definition 6.1 Let R be a relation scheme, $X, Y, Z, T, U \subseteq \Omega$, such that $Z \subseteq X$ and $Z \subseteq T$.

- A relation instance r of R satisfies the *functional dependency implication (fdi)* $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$, iff in every Z-complete set of tuples in r, in which the fd $X \rightarrow Y$ holds, the fd $T \rightarrow U$ must hold too.
- The scheme R satisfies $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ iff all the instances of R satisfy $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$.

□

The ifd's of Chapter 5 are fdi's with $Z = X$. The cfd's of Chapter 4 have $Z = X = T$. Fd's can be expressed as fdi's in many different ways. E. g. $T \rightarrow T \overset{T}{\supset\!\!-} T \rightarrow U$ (a cfd), $\emptyset \rightarrow \emptyset \overset{\emptyset}{\supset\!\!-} T \rightarrow U$ (an ifd) and $T \rightarrow \emptyset \overset{\emptyset}{\supset\!\!-} T \rightarrow U$ (a "true" fdi) all represent the fd $T \rightarrow U$. The goals of Chapters 2 and 3 can be expressed as trivial fdi's, e. g. $T \rightarrow U \overset{T}{\supset\!\!-} T \rightarrow T$.

The fdi's of the *STAFF* example are (in the order in which they were given):

- $emp\ div \rightarrow job\ man\ dep \overset{div}{\supset\!\!-} emp\ div \rightarrow sal$
- $dep\ div \rightarrow job \overset{div}{\supset\!\!-} emp\ div \rightarrow job\ sal\ man$
- $dep\ div \rightarrow job \overset{div}{\supset\!\!-} man\ div \rightarrow dep$

One immediately notices that repeating *div* three times in the fdi is a bit redundant, and one may ask whether the conditions $Z \subseteq X$ and $Z \subseteq T$ in Definition 6.1 are necessary. A first remark is that weakening the conditions to $X \rightarrow Z$ and $T \rightarrow Z$, to obtain *generalized functional dependency implications* does not generate a more powerful class of dependencies (as indicated in the previous chapter, with the introduction of generalized imposed-functional dependencies). However, the following generalization is possible:

Definition 6.2 Let R be a relation scheme, $X, Y, Z, T, U \subseteq \Omega$.

- A relation instance r of R satisfies the *unrestricted functional dependency implication (ufdi)* $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ iff in every Z-unique Z-complete set of tuples in r, in which the fd $X \rightarrow Y$ holds, the fd $T \rightarrow U$ must hold too.
- The scheme R satisfies $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ iff all the instances of R satisfy $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$.

□

This definition considers Z-unique Z-complete sets of tuples, for semantic reasons: the attribute Z is supposed to be the major element in the database, like the *div*ision in the *STAFF* example. We do not want the fd $X \rightarrow Y$ for one Z-value to interfere with $X \rightarrow Y$ for another Z-value, or, in our example, we do not want a constraint on *emp*loyees of one *div*ision to interfere with that constraint on *emp*loyees of another *div*ision. However, the additional condition that we consider Z-unique sets of tuples causes the ufdi's not to be more powerful than the fdi's:

Remark 6.1 *The ufdi* $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ *is equivalent to the fdi* $XZ \rightarrow Y \overset{Z}{\supset\!\!-} TZ \rightarrow U$.

Proof Let $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ hold in r. Consider a Z-complete set of tuples $s \subseteq r$. If $XZ \rightarrow Y$ holds in s, then $XZ \rightarrow Y$ holds in every Z-unique subset of s, hence also $X \rightarrow Y$ holds in every Z-unique subset of s (since they each have only one Z-value). Hence $T \rightarrow U$ must hold in each

Z-unique subset of s. This implies that $TZ \rightarrow U$ must hold in s (since no two Z-unique Z-complete sets of tuples can have the same TZ-value).

Let $XZ \rightarrow Y \overset{Z}{\supset\!\!-} TZ \rightarrow U$ hold in r. Consider a Z-unique Z-complete set of tuples $s \subseteq r$. If $XZ \rightarrow Y$ holds in s, then so does $X \rightarrow Y$ (since s has only one Z-value) and so does $TZ \rightarrow U$ because of the fdi. Hence also $T \rightarrow U$ holds in s (since s has only one Z-value).

□

The ufdi's corresponding to the fdi's of the *STAFF* example look much nicer:

- $emp \rightarrow job\ man\ dep \overset{div}{\supset\!\!-} emp \rightarrow sal$
- $dep \rightarrow job \overset{div}{\supset\!\!-} emp \rightarrow job\ sal\ man$
- $dep \rightarrow job \overset{div}{\supset\!\!-} man \rightarrow dep$

The reason why we build our decomposition around fdi's instead of ufdi's (although the latter look nicer, and are possibly easier to understand), is that using Z-complete sets of tuples is what we have done in the previous chapters. Hence several lemmas and theorems can have very similar proofs (of which some can be easily omitted). If we use ufdi's all theorems would certainly have to be proved explicitly, for clarity.

Definition 6.3 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme, let $X, Y, Z \subseteq \Omega$, such that $Z \subseteq X$.

- For every instance r of R, the *selection for* $X \overset{Z}{\rightarrow} Y$ *of* r, $\sigma_{X \overset{Z}{\rightarrow} Y}(r)$, is the largest Z-complete subset (of tuples) of r in which $X \rightarrow Y$ holds.
- The *selection for* $X \overset{Z}{\rightarrow} Y$ *of* R, $\sigma_{X \overset{Z}{\rightarrow} Y}(R)$, is a scheme $R_1 = (\Omega, \Delta, dom, M_1, SC_1)$. The calculation of SC_1 will be described in Section 6.3. SC_1 contains $X \rightarrow Y$ of course. M_1 explains that all instances of R_1 must be the selection for $X \overset{Z}{\rightarrow} Y$ of the instances of R.

□

We require $Z \subseteq X$ to make sure that X-values of r are not split up by taking the selection for $X \overset{Z}{\rightarrow} Y$ of r.

Definition 6.4 The *horizontal decomposition of a scheme* R*, according to the fdi* $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$, is the ordered pair (R_1, R_2), where $R_1 = \sigma_{X \overset{Z}{\rightarrow} Y}(R)$ and $R_2 = R - R_1$.

□

Note that the horizontal decomposition of a scheme, according to $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ does not depend on T or U, but it induces the fd $T \rightarrow U$ in R_1. In R_2, which contains the exceptions, the fd $X \rightarrow Y$ is violated for every Z-value. This is formalized by means of the following constraint:

Definition 6.5 Let R be a relation scheme, $X, Y, Z \subseteq \Omega$ and $Z \subseteq X$.

- A relation instance r of R satisfies the *anti-functional dependency (afd)* $X \overset{Z}{\not\Longrightarrow} Y$ iff every nonempty Z-complete set of tuples of r, the functional dependency $X \rightarrow Y$ does not hold.
- The scheme R satisfies $X \overset{Z}{\not\Longrightarrow} Y$ iff all the instances of R satisfy $X \overset{Z}{\not\Longrightarrow} Y$.

□

Definition 6.6 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme. Let r be an instance of R.

- The *selection for* $X \overset{Z}{\not\Longrightarrow} Y$ *of* r, $\sigma_{X \overset{Z}{\not\Longrightarrow} Y}(r)$, is the largest Z-complete set of tuples of r in which $X \overset{Z}{\not\Longrightarrow} Y$ holds.
- The *selection for* $X \overset{Z}{\not\Longrightarrow} Y$ *of* R, $\sigma_{X \overset{Z}{\not\Longrightarrow} Y}(R)$ is the scheme $R_2 = (\Omega, \Delta, dom, M_2, SC_2)$. The calculation of SC_2 will be described in Section 6.3. M_2 explains that all instances of R_2 must be the selection for $X \overset{Z}{\not\Longrightarrow} Y$ of the instances of R.

□

One can easily see that $R - \sigma_{X \overset{Z}{\rightarrow} Y}(R) = \sigma_{X \overset{Z}{\not\Longrightarrow} Y}(R)$, hence the horizontal decomposition of R according to $X \rightarrow Y \overset{Z}{\rightarrow} T \rightarrow U$ is the couple of schemes $(\sigma_{X \overset{Z}{\rightarrow} Y}(R), \sigma_{X \overset{Z}{\not\Longrightarrow} Y}(R))$.

The afunctional dependencies used in the previous chapters, are afd's with $X = Z$.

From now on we shall assume that the set of constraints SC of a relation scheme R consists of a set $\mathcal{I}$ of fdi's and a set $\mathcal{A}$ of afd's. (Note that $\mathcal{I}$ contains the fd's, cfd's and ifd's, and that $\mathcal{A}$ contains the ad's).

As for fd's, cfd's and ifd's, we start the decomposition theory with the proof of the existence of strong Armstrong relations for fd's, considering the presence of fdi's:

Definition 6.7 Let $\mathcal{I}$ be a set of fdi's over a set Ω of attributes. A *strong Armstrong relation for* $\mathcal{I}$ is a strong Armstrong relation for the set of all fd's that are a consequence of $\mathcal{I}$.

□

The proof of the construction of (strong) Armstrong relations for $\mathcal{I}$ again relies on the following special set of fd's:

Definition 6.8 $FSAT_{\mathcal{I}}(X, Y)$ is the smallest possible set of fd's, such that:

1. $X \to Y \in FSAT_{\mathcal{I}}(X, Y)$.
2. If $X_i \to Y_i \in FSAT_{\mathcal{I}}(X, Y)$ and $X_i \to Y_i \overset{Z_i}{\supset\!-} T_i \to U_i \in \mathcal{I}$ then $T_i \to U_i \in FSAT_{\mathcal{I}}(X, Y)$.
3. If $FSAT_{\mathcal{I}}(X, Y) \models T \to U$ then $T \to U \in FSAT_{\mathcal{I}}(X, Y)$.

□

$FSAT_{\mathcal{I}}(X, Y)$ can be constructed starting from $\{X \to Y\}$ and by repeatedly trying to satisfy 2 and 3 of the definition.

Lemma 6.1 *Let $\mathcal{I}$ be a set of fdi's over Ω, $T, U, X, Y \subseteq \Omega$. $FSAT_{\mathcal{I}}(X, Y) = \{P \to Q \mid \mathcal{I} \cup \{X \to Y\} \models P \to Q\}$.*

Proof For $X \to Y$ it is obvious that $\mathcal{I} \cup \{X \to Y\} \models X \to Y$. For other fd's we prove the lemma by induction.

- Suppose $T \to V$ is added by trying to satisfy part 2 of the definition. Then (by induction) there is an fdi $P \to Q \overset{U}{\supset\!-} T \to V \in \mathcal{I}$ for which $\mathcal{I} \cup \{X \to Y\} \models P \to Q$. Obviously $P \to Q$ and $P \to Q \overset{U}{\supset\!-} T \to V$ imply $T \to V$.
- Suppose $T \to V$ is added by trying to satisfy part 3 of the definition. Then it is a consequence of a number of fd's for which the lemma already holds (by induction hypothesis). Hence $T \to V$ also is a consequence of $\mathcal{I} \cup \{X \to Y\}$.

This proves that all the fd's of $FSAT_{\mathcal{I}}(X, Y)$ are consequences of $\mathcal{I} \cup \{X \to Y\}$. To prove the opposite inclusion, consider $Arm(FSAT_{\mathcal{I}}(X, Y))$. In $Arm(FSAT_{\mathcal{I}}(X, Y))$, all the fd's of $FSAT_{\mathcal{I}}(X, Y)$ hold, and (since by part 3 of the definition $FSAT_{\mathcal{I}}(X, Y)^* = FSAT_{\mathcal{I}}(X, Y)$) no other fd's hold. Suppose some fdi $X_i \to Y_i \overset{Z_i}{\supset\!-} T_i \to U_i$ of $\mathcal{I}$ does not hold in $Arm(FSAT_{\mathcal{I}}(X, Y))$. Then there exists a Z_i-complete set of tuples in

$Arm(FSAT_{\mathcal{I}}(X,Y))$ in which $X_i \to Y_i$ holds and in which $T_i \to U_i$ does not hold. By Theorem 2.1 this means that $FSAT_{\mathcal{I}}(X,Y) \models X_i \to Y_i$, and hence $X_i \to Y_i \in FSAT_{\mathcal{I}}(X,Y)$ (by part 4 of the definition). By part 2 of the definition this implies that $T_i \to U_i \in FSAT_{\mathcal{I}}(X,Y)$ and hence $T_i \to U_i$ must hold in $Arm(FSAT_{\mathcal{I}}(X,Y))$, a contradiction. □

From Theorem 2.1 and the above lemma one can easily deduce that

Theorem 6.1 *Let $\mathcal{I}$ be a set of fdi's. $Arm(FSAT_{\mathcal{I}}(\emptyset,\emptyset))$ is a strong Armstrong relation for $\mathcal{I}$. In other words, for all $T,U \subseteq \Omega$ holds that $\mathcal{I} \models T \to U$ iff $T \to U \in FSAT_{\mathcal{I}}(\emptyset,\emptyset)$.* □

From this theorem we learn that the set Z in $X \to Y \overset{Z}{\supset} T \to U$ has no importance for the implication problem for fd's. Hence fdi's can represent arbitrary implications between fd's (in the whole relation), by letting $Z = \emptyset$. The cfd's and ifd's could not express these implications. This suggests that the fdi's are the largest class of constraints for studying "partial" implications between fd's.

With the Armstrong relation for $\mathcal{I}$, we can prove the following theorem for detecting conflict, in exactly the same way as for cfd's:

Theorem 6.2 *$\mathcal{I} \cup \mathcal{A}$ is in conflict iff for some afd $X \overset{Z}{\not\Rightarrow} Y$ of $\mathcal{A}$, $\mathcal{I} \models X \to Y$ holds.*

This Theorem shows that the set Z also is not important for the conflict detection.

Algorithm 6.1 *Conflict Detection*

Input: $\mathcal{I}, \mathcal{A}$, a set of fdi's and a set of afd's.

Output: *true* or *false*.

Method:
for each $T \overset{Z}{\not\Rightarrow} U$ **in** $\mathcal{A}$ **do**
  if $\mathcal{I} \models T \to U$
    then
      return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

In the above algorithm we did not explain how to verify whether $\mathcal{I} \models T \to U$. This means verifying whether $T \to U \in FSAT_{\mathcal{I}}(\emptyset, \emptyset)$, which can be done by an algorithm, very similar to Algorithm 4.2. The time-complexity of that algorithm is $O(n^3r^2)$ where $n = \sharp\mathcal{I}$ and $r = \sharp\Omega$. The time complexity of Algorithm 6.1 then becomes $O(n^3r^2m)$ where $m = \sharp\mathcal{A}$.

6.2 The Implication Problem for fdi's and afd's

As in the previous chapters, we solve the implication problem for fdi's and afd's by means of the tools *FSAT*, the Armstrong relation, and a set of inference rules. However we need an additional tool: a construction similar to that of Lemma 4.8. We solve the implication problem for fdi's (only) first. For fdi's we have the following inference rules:

$(FI0)$ **:** if $Z \subseteq X$, $Z \subseteq T$, $Y \subseteq X$, $Z' \subseteq X'$, $Z' \subseteq T$, $Y' \subseteq X'$, and $X \to Y \overset{Z}{\supset\!\!-} T \to U$ then $X' \to Y' \overset{Z}{\supset\!\!-} T \to U$.

$(FI1)$ **:** if $Z \subseteq X$ and $Z \subseteq T$ and either $U \subseteq T$ or (both) $X \subseteq T$ and $U - T \subseteq Y - X$ then $X \to Y \overset{Z}{\supset\!\!-} T \to U$.

$(FI2)$ **:** if $X \to Y \overset{Z}{\supset\!\!-} T \to U$ and $W \subseteq V$ then $X \to Y \overset{Z}{\supset\!\!-} TV \to UW$.

$(FI3)$ **:** if $X \to Y \overset{Z}{\supset\!\!-} T \to U$ and $X \to Y \overset{Z}{\supset\!\!-} U \to V$ then $X \to Y \overset{Z}{\supset\!\!-} T \to V$.

$(FI4)$ **:** if $X \to Y \overset{Z}{\supset\!\!-} T \to U$ and $T \to U \overset{Z}{\supset\!\!-} V \to W$ then $X \to Y \overset{Z}{\supset\!\!-} V \to W$.

$(FI5)$ **:** if $X \to Y \overset{Z}{\supset\!\!-} T \to U$ and $Z \to Z'$ then $XZ' \to Y \overset{Z'}{\supset\!\!-} TZ' \to U$. □

As fd's are special fdi's the use of fd's in these rules is allowed. In fact $FI0$ shows how to represent fd's as fdi's (in some equivalent ways). The classical inference rules for fd's and the inference rules for cfd's and ifd's can be deduced from $FI0, \ldots, FI5$ as follows:

Lemma 6.2 *The rules $FI0, \ldots, FI5$ are complete for ifd's (and hence also for cfd's and fd's).*

Proof To prove this completeness we show how to prove $I0, \ldots, I5$ using $FI0, \ldots, FI5$. For $I0, \ldots, I4$ this is trivial since these rules are special cases of $FI0, \ldots, FI5$. The only nontrivial rule is $I5$: Let $X \to Y \overset{X}{\supset\!\!-} X' \to Z$

and $X \to X''$. Rule $FI5$ induces $XX'' \to Y \overset{X''}{\supset\!\!-} X'X'' \to Z$. Rule $FI1$ induces $X'' \to Y \overset{X''}{\supset\!\!-} XX'' \to Y$. Rule $FI4$ then generates $X'' \to U \overset{X''}{\supset\!\!-} X'X'' \to Z$.

□

Theorem 6.3 *The rules $FI0 \ldots FI5$, are sound.*

Proof This is very straightforward, and left to the reader. The proof of Theorem 4.3 shows the arguments that can be applied to the rules for fdi's as well.

□

Lemma 6.1 does not describe the implication problem for fd's completely (considering the presence of fdi's). The sets Z_i in the fdi's $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i$ are of no importance in the construction of $FSAT_{\mathcal{I}}(X, Y)$, although they have some influence on the implication problem for fdi's. The following lemma shows the influence of the Z_i on the implication problem for fd's:

Lemma 6.3 *Let $\mathcal{I}_Z = \{X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i \in \mathcal{I} \mid \mathcal{I} \models X_i \to Y_i$ or $\mathcal{I} \models Z_i \to Z\}$. Let $P, Q \subseteq \Omega$. Then $\mathcal{I} \models P \to Q$ iff $\mathcal{I}_Z \models P \to Q$ (for every $Z \subseteq \Omega$).*

Proof In the construction of $FSAT_{\mathcal{I}}(\emptyset, \emptyset)$ only those $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i \in \mathcal{I}$ are used for which $\mathcal{I} \models X_i \to Y_i$. Hence $FSAT_{\mathcal{I}}(\emptyset, \emptyset) = FSAT_{\mathcal{I}_Z}(\emptyset, \emptyset)$ for every set $Z \subseteq \Omega$. The lemma then follows directly from Lemma 6.1.

□

The following lemma is similar to Lemmas 4.3 and 5.3.

Lemma 6.4 *Let $Z \subseteq X$, let $\mathcal{I}_Z$ be as in Lemma 6.3. If $P \to Q \in FSAT_{\mathcal{I}_Z}(X, Y)$ then $\mathcal{I} \models P \to Q$ or $\mathcal{I} \models P \to Z$.*

Proof We prove that the property remains valid throughout the construction of $FSAT_{\mathcal{I}_Z}(X, Y)$.

- If $P \to Q = X \to Y$ then $P \to Z$ is trivial.
- If $P \to Q$ is added in part 2 of Definition 6.8 then $P = T_i$ and $Q = U_i$ for some $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i \in \mathcal{I}_Z$. There are two possibilities (by the definition of $\mathcal{I}_Z$): $\mathcal{I} \models X_i \to Y_i$ or $\mathcal{I} \models Z_i \to Z$.
  - If $\mathcal{I} \models X_i \to Y_i$ then $\mathcal{I} \models T_i \to U_i = P \to Q$.

  - If $\mathcal{I} \models Z_i \to Z$ then $\mathcal{I} \models T_i \to Z = P \to Z$ by augmentation (since $Z_i \subseteq T_i$).
- If $P \to Q$ is added in part 3 then it is derived from other fd's (already in $FSAT_{\mathcal{I}_Z}(X, Y)$), by reflexivity, augmentation or transitivity.
  - If $Q \subseteq P$ then $P \to Q$ is trivial.
  - If $P = P'P''$, $Q = Q'Q''$, with $P' \to Q'$ already in $FSAT_{\mathcal{I}_Z}(X, Y)$ and $Q'' \subseteq P''$, then $P \to Q$ or $P \to Z$ is deduced from $P' \to Q'$ or $P' \to Z$ by augmentation.
  - If $P \to O$ and $O \to Q$ already are in $FSAT_{\mathcal{I}_Z}(X, Y)$ then $P \to Q$ or $P \to Z$ is deduced from $P \to O$ or $P \to Z$ and $O \to Q$ or $O \to Z$ by transitivity.

□

The following lemma provides part of the typical solution of the implication problem, as it occurred in the previous chapters.

Lemma 6.5 *Let $Z \subseteq X$, $\mathcal{I} \models T \to Z$, and let $\mathcal{I}_Z$ be as in Lemma 6.3. Then $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} TZ \to U$ iff $T \to U \in FSAT_{\mathcal{I}_Z}(X, Y)$.*

Proof If $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} TZ \to U$ then clearly $TZ \to U \in FSAT_{\mathcal{I}_Z}(X, Y)$. Since $\mathcal{I} \models T \to Z$ we know $\mathcal{I}_Z \models T \to Z$ by Lemma 6.3. Clearly $T \to Z$ and $TZ \to U$ induce $T \to U$. Lemma 6.1 implies that $T \to U \in FSAT_{\mathcal{I}_Z}(X, Y)$. For the converse we prove that the property remains valid throughout the construction of $FSAT_{\mathcal{I}_Z}(X, Y)$.

- If $X \to Y = T \to U$ then $X \to Y \overset{Z}{\supset\!-} TZ \to U$ holds by rule $FI1$.
- If $T \to U$ is added in part 2 of Definition 6.8 then $T = T_i$ and $U = U_i$ for some $X_i \to Y_i \overset{Z_i}{\supset\!-} T_i \to U_i \in \mathcal{I}_Z$. There are 2 possibilities: $\mathcal{I} \models Z_i \to Z$ or $\mathcal{I} \models X_i \to Y_i$.
  - If $\mathcal{I} \models X_i \to Y_i$ then also $\mathcal{I} \models T_i \to U_i$. From Lemma 6.3 we know that $\mathcal{I}_Z \models T_i \to U_i$. Augmentation gives $\mathcal{I}_Z \models T_iZ \to U_i$. From $X \to Y \overset{Z}{\supset\!-} X \to X$ ($FI1$) and $X \to X \overset{Z}{\supset\!-} T_iZ \to U_i$ ($FI0$, provided that $Z \subseteq T = T_i$) we infer $X \to Y \overset{Z}{\supset\!-} T_iZ \to U_i = X \to Y \overset{Z}{\supset\!-} TZ \to U$ by $FI4$.
  - If $\mathcal{I} \models Z_i \to Z$ then $\mathcal{I} \models X_i \to Z$ by augmentation. By induction we know that $\mathcal{I} \models X \to Y \overset{Z}{\supset\!-} X_iZ \to Y_i$. $X_i \to Y_i \overset{Z_i}{\supset\!-} T_i \to U_i$ and $Z_i \to Z$ induce $X_iZ \to Y_i \overset{Z}{\supset\!-} T_iZ \to U_i$ by $FI5$. $\mathcal{I} \models X \to Y \overset{Z}{\supset\!-}$

$X_iZ \to Y_i$ and $X_iZ \to Y_i \overset{Z}{\supset\!-} T_iZ \to U_i$ induce $X \to Y \overset{Z}{\supset\!-} T_iZ \to U_i$ by $FI4$.

- If $T \to U$ is added in part 3 of Definition 6.8 then it is inferred from other fd's of $FSAT_{\mathcal{I}_Z}(X,Y)$ by reflexivity, augmentation or transitivity.
  - If $U \subseteq T$ then $X \to Y \overset{Z}{\supset\!-} TZ \to U$ holds by $FI1$.
  - If $T \to U = T'T'' \to U'U''$ with $T' \to U'$ already in $FSAT_{\mathcal{I}_Z}(X,Y)$ and $U'' \subseteq T''$, then there are two possibilities (by Lemma 6.4): $\mathcal{I} \models T' \to U'$ or $\mathcal{I} \models T' \to Z$.
    - If $\mathcal{I} \models T' \to U'$ then $TZ \to U$ follows by augmentation. $X \to Y \overset{Z}{\supset\!-} X \to X$ ($FI1$) and $X \to X \overset{Z}{\supset\!-} TZ \to U$ ($FI0$) then induce $X \to Y \overset{Z}{\supset\!-} TZ \to U$ by $FI4$.
    - If $\mathcal{I} \models T' \to Z$ then $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} T'Z \to U'$ holds by induction. $X \to Y \overset{Z}{\supset\!-} TZ \to U$ then follows by $FI2$.
  - If $T \to V$ and $V \to U$ already are in $FSAT_{\mathcal{I}_Z}(X,Y)$ then there are four possibilities (Lemma 6.5):
    - If $\mathcal{I} \models T \to V$ and $\mathcal{I} \models V \to Z$ then $\mathcal{I} \models T \to U$, hence $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} TZ \to U$ is deduced as shown above.
    - If $\mathcal{I} \models T \to V$ and $\mathcal{I} \models V \to Z$ then $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} VZ \to U$ holds by induction. $X \to Y \overset{Z}{\supset\!-} TZ \to V$ is deduced as above. $FI2$ yields $X \to Y \overset{Z}{\supset\!-} TZ \to VZ$. $X \to Y \overset{Z}{\supset\!-} TZ \to U$ is then deduced by $FI3$.
    - If $\mathcal{I} \models T \to Z$ and $\mathcal{I} \models V \to U$ then $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} TZ \to V$ by induction. $FI2$ yields $X \to Y \overset{Z}{\supset\!-} TZ \to VZ$. $X \to Y \overset{Z}{\supset\!-} VZ \to U$ is deduced as above. $FI3$ yields $X \to Y \overset{Z}{\supset\!-} TZ \to U$.
    - If $\mathcal{I} \models T \to Z$ and $\mathcal{I} \models V \to Z$ then $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} TZ \to V$ and $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!-} VZ \to U$ hold by induction. Again $FI2$ and $FI3$ yield $X \to Y \overset{Z}{\supset\!-} TZ \to U$.

□

From Lemma 6.2 and the proof of Lemma 6.5 one can easily deduce:

Corollary 6.1 *Let $\mathcal{I}_Z$ be as in Lemma 6.3, let $Z \subseteq X$, $Z \subseteq T$. Then $\mathcal{I}_Z \vdash X \to Y \overset{Z}{\supset\!-} T \to U$ iff $T \to U \in FSAT_{\mathcal{I}_Z}(X,Y)$.*

□

To complete the proof of the completeness of $FI0 \ldots FI5$ for fdi's it remains to show that $\mathcal{I} \models X \to Y \overset{Z}{\supset\!\!-} T \to U$ iff $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!\!-} T \to U$. In the proof we need a complicated construction of an instance, similar to that in Lemma 4.8, which we shall be needing also in the completeness proof for mixed fdi's and afd's and in the solution to the inheritance problem. Therefor we give the construction in a somewhat more general way than we need for the fdi's only.

Lemma 6.6 *Let $\mathcal{I} \cup \mathcal{A}$ be not in conflict. Let $\mathcal{I}_Z$ be as in Lemma 6.3. Let $\mathcal{A}_Z = \{T_i \overset{Z_i}{\not\longrightarrow} U_i \in \mathcal{A} \mid \mathcal{I} \models Z_i \to Z\}$. Let $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models X \to Y \overset{Z}{\supset\!\!-} T \to U$ (or let $\mathcal{I}_z \cup \mathcal{A}_Z \not\models X \overset{Z}{\not\longrightarrow} Y$). Then we can construct an instance in which $\mathcal{I} \cup \mathcal{A}$ holds but in which $X \to Y \overset{Z}{\supset\!\!-} T \to U$ (or $X \overset{Z}{\not\longrightarrow} Y$) does not hold.*

Proof Let $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models X \to Y \overset{Z}{\supset\!\!-} T \to U$. We shall see later that in $Arm(FSAT_{\mathcal{I}_Z}(X,Y))$, $\mathcal{A}_Z$ holds. Since also $\mathcal{I}_Z \cup \{X \to Y\}$ holds $T \to U$ cannot hold, otherwise (by Theorem 2.1) $T \to U \in FSAT_{\mathcal{I}_Z}(X,Y)$ which implies $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!\!-} T \to U$ by Lemma 6.5. By Theorem 6.1 this means that $T \overset{T}{\not\longrightarrow} U$ holds in $Arm(FSAT_{\mathcal{I}_Z}(X,Y))$. In $Arm(FSAT_{\mathcal{I}_Z}(X,Y))$ a number of fdi's of $\mathcal{I} - \mathcal{I}_Z$ and a number of afd's of $\mathcal{A} - \mathcal{A}_Z$ may not hold.

Suppose $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i \in \mathcal{I} - \mathcal{I}_Z$ does not hold, hence $X_i \to Y_i$ and $T_i \overset{T_i}{\not\longrightarrow} U_i$ hold in $r_1 = Arm(FSAT_{\mathcal{I}_Z}(X,Y))$.

Consider $r_2 = Arm(FSAT_{\mathcal{I}}(\emptyset,\emptyset))$. In r_2 $\mathcal{I} \cup \mathcal{A}$ holds. Let $s = r_1 \cup r_2$, after renaming one value of the domain of r_2, such that $\exists t_1 \in r_1$, $\exists t_2 \in r_2$: $t_1[\overline{Z_i}] = t_2[\overline{Z_i}]$, (and such that $X_i \to Y_i$ holds in the Z_i-value containing t_1).

- In s, $\mathcal{I}_Z$ still holds: suppose $X_j \to Y_j \overset{Z_j}{\supset\!\!-} T_j \to U_j \in \mathcal{I}_Z$ not hold. There are two cases: $\mathcal{I} \models X_j \to Y_j$ (and $T_j \to U_j$) or $\mathcal{I} \models Z_j \to Z$.
  - If $\mathcal{I} \models X_j \to Y_j$ and $T_j \to U_j$ and if $T_j \to U_j$ no longer holds in s then $\exists t'_1 \in r_1$, $\exists t'_2 \in r_2$: $t'_1[T_j] = t'_2[T_j]$ and $t'_1[U_j] \neq t'_2[U_j]$. $t'_1[T_j] = t'_2[T_j]$ is only possible if $T_j \subseteq \overline{Z_i}$ but then also $U_j \subseteq \overline{Z_i}$ since $\mathcal{I} \models T_j \to U_j$, hence $t'_1[U_j] = t'_2[U_j]$, a contradiction.
  - If $\mathcal{I} \models Z_j \to Z$, then $T_j \subseteq \overline{Z_i}$ (which must hold to violate $T_j \to U_j$ for the same reason as in the first case) and $Z_j \to Z$ would induce

$Z \subseteq \overline{Z_i}$, a contradiction with $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i \notin \mathcal{I}_Z$.

- In s, $\mathcal{A}_Z$ still holds since it is impossible to violate an afd by taking a union of two instances in which that afd holds.
- In s, every fdi of $\mathcal{I} - \mathcal{I}_Z$ and every afd of $\mathcal{A} - \mathcal{A}_Z$ which already holds in r_1 (and also in r_2 of course) still holds. For the afd's the reason is the same as for those of $\mathcal{A}_Z$. For the fdi's we have that if $X_k \to Y_k \overset{Z_k}{\supset\!\!-} T_k \to U_k \in \mathcal{I} - \mathcal{I}_Z$ holds in r_1 then $X_k \overset{Z_k}{\not\Longrightarrow} Y_k$ holds in r_1, and such an afd is not violated any more.
- In s, $X \overset{Z}{\not\Longrightarrow} Y$ still does not hold, since $X \to Y$ holds in r_1 and since (as explained above) r_1 and r_2 do not "share" a common Z- value (otherwise $Z \subseteq \overline{Z_i}$).
- In s, $T \overset{Z}{\not\Longrightarrow} U$ may no longer hold, because it may not hold in r_2. But in the "r_1-part" of s, $T \overset{Z}{\not\Longrightarrow} U$ still holds, since r_1 and r_2 do not share a Z-value (and hence also no T-value).
- But in s the number of Z_i-values for which $X_i \to Y_i$ holds (and not $T_i \to U_i$) is decreased by one, since the Z_i-value containing t_1 collapses with the Z_i-value of R_2, containing t_2, (in which $X_i \overset{Z_i}{\not\Longrightarrow} Y_i$ holds), and since R_2 has no Z_i-values with $X_i \to Y_i$.

By repeating the above construction for all other Z_i-values for which $X_i \to Y_i$ holds one can generate a relation in which $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i$ holds (since $X_i \overset{Z_i}{\not\Longrightarrow} Y_i$ holds). By then repeating the above construction for all fdi's of $\mathcal{I} - \mathcal{I}_Z$ which do not hold in $Arm(FSAT_{\mathcal{I}_Z}(X,Y))$ one generates a relation in which $\mathcal{I}$ holds (and $X \to Y \overset{Z}{\supset\!\!-} T \to U$ still does not hold).

For the afd's of $\mathcal{A} - \mathcal{A}_Z$ the construction proceeds in a similar way:

If $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models X \overset{Z}{\not\Longrightarrow} Y$ then we shall see later that in $Arm(FSAT_{\mathcal{I}_Z}(X,Y))$, $\mathcal{A}_Z$ holds, as well as $\mathcal{I}_Z \cup \{X \to Y\}$. The same construction as above leads to an instance in which $\mathcal{I} \cup \mathcal{A}$ holds, and in which $X \overset{Z}{\not\Longrightarrow} Y$ does not hold. □

Theorem 6.4 *FI*0 ... *FI*5 *are complete for fdi's. Furthermore, let* $Z \subseteq X$, $Z \subseteq T$, *then* $\mathcal{I} \models X \to Y \overset{Z}{\supset\!\!-} T \to U$ *iff* $T \to U \in FSAT_{\mathcal{I}_Z}(X,Y)$.

Proof If $\mathcal{I} \models X \to Y \overset{Z}{\supset\!\!-} T \to U$ then by Lemma 6.6 $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!\!-} T \to U$. (The converse is trivial.) Lemma 6.5 and Corollary 6.1

imply that $T \to U \in FSAT_{\mathcal{I}_Z}(X,Y)$, $\mathcal{I}_Z \vdash X \to Y \overset{Z}{\supset\!\!-} T \to U$ and $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!\!-} T \to U$ are equivalent.

□

Before we solve the implication problem for mixed fdi's and afd's we first propose a set of inference rules:

*FIA*1 **:** if $Z \to V$, $X \to Y \overset{Z}{\supset\!\!-} T \to U$ and $T \overset{Z}{\not\longrightarrow} U$ then $V \overset{Z}{\not\longrightarrow} Y$.
*FIA*2 **:** if $X \overset{Z}{\not\longrightarrow} Y$ and $Z \subseteq T$ then $X \to Y \overset{Z}{\supset\!\!-} T \to U$ for all $U \subseteq \Omega$.
*FIA*3 **:** if $X \overset{Z}{\not\longrightarrow} Y$ and $Z \to Z'$ then $XZ' \overset{Z'}{\not\longrightarrow} Y$.

□

Lemma 6.7 *Rules FIA*1*, FIA*2 *and FIA*3 *are sound.*

Proof Rule *FIA*1 is very similar to *IA*1. *FIA*2 is very similar too *IA*2. We only prove *FIA*3:

Let $X \overset{Z}{\not\longrightarrow} Y$ and $Z \to Z'$ hold in an instance r. Consider an arbitrary Z'-complete set of tuples s in r. Since $Z \to Z'$, s also is Z-complete. Hence $X \to Y$ does not hold in s (because of $X \overset{Z}{\not\longrightarrow} Y$). Since $Z \to Z'$, $X \to XZ'$ holds by augmentation. $XZ' \to Y$ cannot hold in s, since $X \to XZ'$ and $XZ' \to Y$ would induce $X \to Y$ by transitivity. Since $XZ' \to Y$ does not hold in this arbitrary Z'-complete set of tuples of r, $XZ' \overset{Z'}{\not\longrightarrow} Y$ holds in r.

□

Remark 6.2 *Rules FI*0*, . . . , FI*5*, FIA*1 *and FIA*2 *are complete for ifd's and ad's.*

Proof This is fairly trivial since *IA*1 and *IA*2 are special cases of *FIA*1 and *FIA*2 respectively.

□

Theorem 6.5 *Let* $\mathcal{I} \cup \mathcal{A}$ *be not in conflict,* $X, Y, Z \subseteq \Omega$ *and* $Z \subseteq X$*.* $\mathcal{I} \cup \mathcal{A} \models X \overset{Z}{\not\longrightarrow} Y$ *iff* $\mathcal{I}_Z \cup \mathcal{A}_Z \models X \overset{Z}{\not\longrightarrow} Y$*.*

Proof The if-part is trivial.
For the only-if-part, suppose $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models X \overset{Z}{\not\longrightarrow} Y$. We first show that $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\}$ cannot be in conflict .

Suppose $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\}$ is in conflict. Then in $Arm(FSAT_{\mathcal{I}_Z}(X,Y))$ for some afd $T \overset{V}{\not\longrightarrow} U$ of $\mathcal{A}_Z$ $T \to U$ holds (since $T \overset{V}{\not\longrightarrow} U$ does not hold).

Hence (Theorem 6.1) $\mathcal{I}_Z \cup \{X \to Y\} \models T \to U$. Since $\mathcal{I}_Z \models V \to Z$ (hence also $\mathcal{I}_Z \models T \to Z$) Lemma 6.5 yields $\mathcal{I}_Z \models X \to Y \overset{Z}{\supset\!\!-} TZ \to U$.

$T \overset{V}{\not\Longrightarrow} U$ induces $TZ \overset{V}{\not\Longrightarrow} U$ by rule *FIA*1 (since $T \to TZ$ by augmentation on $T \to Z$, and since $T \to U \overset{V}{\supset\!\!-} T \to U$ is trivial). Rule *FIA*3 yields $TZ \overset{Z}{\not\Longrightarrow} U$, and *FIA*1 then induces $X \overset{Z}{\not\Longrightarrow} Y$ (from $X \to Y \overset{Z}{\supset\!\!-} TZ \to U$), a contradiction with $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models X \overset{Z}{\not\Longrightarrow} Y$.

So $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\}$ cannot be in conflict. Hence (as we promised in the proof of Lemma 6.6) $\mathcal{A}_Z$ holds in $Arm(FSAT_{\mathcal{I}_Z}(X, Y))$. Lemma 6.6 then says that there exists an instance in which $\mathcal{I} \cup \mathcal{A}$ holds and in which $X \overset{Z}{\not\Longrightarrow} Y$ does not hold. Hence $\mathcal{I} \cup \mathcal{A} \not\models X \overset{Z}{\not\Longrightarrow} Y$.

□

The proof of the above theorem immediately implies that:

Corollary 6.2 *The rules FI*$0, \ldots, FI5, FIA1$ *and FIA*3 *are complete for the inference of afd's from a set of fdi's and afd's.*

□

Using Theorem 6.2 and 6.5 one can easily prove the following membership algorithm for afd's (considering the presence of fdi's):

Algorithm 6.2 *Membership Detection for afd's with fdi's*

Input: $\mathcal{I}, \mathcal{A}$, a set of fdi's and a set of afd's, not in conflict; $X \overset{Z}{\not\Longrightarrow} Y$ an afd.

Output: *true* or *false*

Method:

var $\mathcal{I}_Z$: set of fdi's := $\emptyset$
$\mathcal{A}_Z$: set of afd's := $\emptyset$

begin
for each $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i$ **in** $\mathcal{I}$ **do**
if $\mathcal{I} \models X_i \to Y_i$ or $\mathcal{I} \models Z_i \to Z$
then
$\mathcal{I}_Z := \mathcal{I}_Z \cup \{X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i\}$
od
for each $X_j \overset{Z_j}{\not\Longrightarrow} Y_j$ **in** $\mathcal{A}$ **do**
if $\mathcal{I} \models Z_j \to Z$

then
$\mathcal{A}_Z := \mathcal{A}_Z \cup \{X_j \overset{Z_j}{\not\rightarrow} Y_j\}$
od
for each $X_j \overset{Z_j}{\not\rightarrow} Y_j$ **in** $\mathcal{A}_Z$ **do**
if $\mathcal{I}_Z \cup \{X \rightarrow Y\} \models X_j \rightarrow Y_j$
then
return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

Rule *FIA*2 shows the influence of afd's on the implication problem for fdi's:

Theorem 6.6 *Let* $\mathcal{I} \cup \mathcal{A}$ *be not in conflict,* $X, Y, Z, T, U \subseteq \Omega$, $Z \subseteq X$ *and* $Z \subseteq T$. $\mathcal{I} \cup \mathcal{A} \models X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ *iff* $\mathcal{I}_Z \models X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ *or* $\mathcal{I}_Z \cup \mathcal{A}_Z \models X \overset{Z}{\not\rightarrow} Y$.

Proof The if part is trivial (using *FIA*2 if $\mathcal{I}_Z \cup \mathcal{A}_Z \vdash X \overset{Z}{\not\rightarrow} Y$).
For the only-if-part, assume that $\mathcal{I}_Z \not\models X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ and $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models X \overset{Z}{\not\rightarrow} Y$. By the proof of Theorem 6.5 $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \rightarrow Y\}$ is not in conflict. Hence by Lemma 6.6 there exists an instance in which $\mathcal{I} \cup \mathcal{A}$ holds and in which $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ does not hold. Hence $\mathcal{I} \cup \mathcal{A} \not\models X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$.
□

An immediate consequence of the above theorem is:

Corollary 6.3 *The rules* $FI0, \ldots, FI5, FIA1, \ldots, FIA3$ *are complete for the inference of fdi's from a set of fdi's and afd's.*

□

A membership algorithm for fdi's (considering the presence of afd's) is easily deduced:

Algorithm 6.3 *Membership Detection for fdi's with afd's*

Input: $\mathcal{I}, \mathcal{A}$, a set of fdi's and a set of afd's, not in conflict; $X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$ an fdi.

Output: *true* or *false*

Method:
if $\mathcal{I} \models X \rightarrow Y \overset{Z}{\supset\!\!-} T \rightarrow U$
then

return(*true*) { and exit }
if $\mathcal{I} \cup \mathcal{A} \models X \not\stackrel{Z}{\Rightarrow} Y$
then
return(*true*) { and exit }
else
return(*false*)

□

Algorithms 6.2 and 6.3 both take $O(n^4r^2 + n^3r^2m)$ time, where $n = \sharp\mathcal{I}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$. This time-complexity is actually the time needed for calculating $\mathcal{I}_Z$ and $\mathcal{A}_Z$, and is based on a membership algorithm for *FSAT* which takes $O(n^3r^2)$ time.

6.3 The Inheritance of fdi's and afd's.

For the (further) decomposition of the subschemes that result from a horizontal decomposition step, we need to know the fdi's and afd's that hold in the subschemes. This *inheritance problem* is very similar to that for ifd's and ad's.

Notation 6.1 In the sequel we always treat the horizontal decomposition of a scheme $R = (\Omega, \mathcal{I} \cup \mathcal{A})$, according to $X \to Y \stackrel{Z}{\supset} T \to U \in \mathcal{I}$, into the subschemes $R_1 = \sigma_{X \stackrel{Z}{\to} Y}(R) = (\Omega, \mathcal{I}_1 \cup \mathcal{A}_1)$, and $R_2 = \sigma_{X \not\stackrel{Z}{\Rightarrow} Y}(R) = (\Omega, \mathcal{I}_2 \cup \mathcal{A}_2)$. We assume that $\mathcal{I} \cup \mathcal{A}$ is not in conflict, and also that $\mathcal{I} \cup \mathcal{A} \not\models X \to Y$ and $\mathcal{I} \cup \mathcal{A} \not\models X \not\stackrel{Z}{\Rightarrow} Y$. We only require $\mathcal{I}_1 \cup \mathcal{A}_1$ and $\mathcal{I}_2 \cup \mathcal{A}_2$ to be generating for the sets of all dependencies which hold in R_1 and R_2.

□

Since fd's cannot be violated by taking a selection of a relation, Remark 4.3 also applies if the decompositions are based on fdi's instead of cfd's.

The inclusions of Lemma 5.6 are easily translated into the following inclusions for fdi's and afd's, which also express Lemma 5.7:

Lemma 6.8 *Using Notation 6.1 we have:*

- $\mathcal{I}_Z \cup \{X \to Y\} \subseteq \mathcal{I}_1 \subseteq \{X' \to Y' \stackrel{Z'}{\supset} T' \to U' \mid \mathcal{I} \cup \mathcal{A} \cup \{X \to Y\} \models X' \to Y' \stackrel{Z'}{\supset} T' \to U'\}$.

- $\mathcal{I}_Z \subseteq \mathcal{I}_2 \subseteq \{X' \to Y' \overset{Z'}{\supset} T' \to U' \mid \mathcal{I} \cup \mathcal{A} \cup \{X \not\overset{Z}{\to} Y\} \models X' \to Y' \overset{Z'}{\supset} T' \to U'\}$.
- $\mathcal{A}_Z \subseteq \mathcal{A}_1 \subseteq \{X' \not\overset{Z'}{\to} Y' \mid \mathcal{I} \cup \mathcal{A} \cup \{X \to Y\} \models X' \not\overset{Z'}{\to} Y'\}$.
- $\mathcal{A}_Z \cup \{X \not\overset{Z}{\to} Y\} \subseteq \mathcal{A}_2 \subseteq \{X' \not\overset{Z'}{\to} Y' \mid \mathcal{I} \cup \mathcal{A} \cup \{X \not\overset{Z}{\to} Y\} \models X' \not\overset{Z'}{\to} Y'\}$.

□

The proof of the inheritance of fdi's and afd's relies on the construction of Lemma 6.6, which has already been used to solve the implication problem.

Theorem 6.7 *Using Notation 6.1, an fdi or afd must hold in R_1 (resp. $\mathcal{R}_2$) iff it is a consequence of $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\}$ (resp. $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \not\overset{Z}{\to} Y\}$).*

Proof From Lemma 6.8 it follows that $(\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\})^* \subseteq (\mathcal{I}_1 \cup \mathcal{A}_1)^* \subseteq (\mathcal{I} \cup \mathcal{A} \cup \{X \to Y\})^*$ and also that $(\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \not\overset{Z}{\to} Y\})^* \subseteq (\mathcal{I}_2 \cup \mathcal{A}_2)^* \subseteq (\mathcal{I} \cup \mathcal{A} \cup \{X \not\overset{Z}{\to} Y\})^*$. We prove that the first inclusions are equalities.

Let $\mathcal{I} \cup \mathcal{A} \cup \{X \to Y\} \models T \not\overset{V}{\to} U$ but $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\} \not\models T \not\overset{V}{\to} U$. We prove that $T \not\overset{V}{\to} U$ does not hold in R_1.

$\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\} \not\models T \not\overset{V}{\to} U$ implies that $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \{X \to Y\} \cup \{T \to U\}$ is not in conflict, by the proof of Theorem 6.5. By the proof of Lemma 6.6 one can construct an instance r in which $\mathcal{I} \cup \mathcal{A}$ holds but in which
$T \not\overset{V}{\to} U$ does not hold. One starts with $r_1 = Arm(FSAT_{\mathcal{I}_Z \cup \{X \to Y\}}(T, U))$ this time, and adds copies of $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$ to obtain r. From the construction, used in the proof of Lemma 6.6 one can easily see that $r_1 = \sigma_{X \overset{Z}{\to} Y}(r)$, since the copies of $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$ have $X \not\overset{Z}{\to} Y$ and have different Z-values than those occurring in r_1. Hence we obtain an instance in which $\mathcal{I} \cup \mathcal{A}$ holds, and such that $T \not\overset{V}{\to} U$ does not hold in r_1. Hence $T \not\overset{V}{\to} U \notin (\mathcal{I}_1 \cup \mathcal{A}_1)^*$.

The proof of the other three cases (an fdi in R_1, and afd in R_2 and an fdi in R_2) is similar, and also similar to the proof for cfd's an ad's, and therefore left to the reader.

□

From Theorem 6.7 one can easily deduce an algorithm which calculates generating sets for the inherited dependencies in $O(n^4r^2 + n^3r^2m)$ time

where $n = \sharp\mathcal{I}$, $m = \sharp\mathcal{A}$ and $r = \sharp\Omega$. Hence this also is the time-complexity of the following algorithm:

Algorithm 6.4 *A Horizontal Decomposition Step with fdi's and afd's*

Input: $R = (\Omega, \mathcal{I} \cup \mathcal{A})$ and an fdi $X \to Y \overset{Z}{\supset\!\!-} T \to U \in \mathcal{I}$. $\mathcal{I} \cup \mathcal{A}$ is assumed not to be in conflict, $\mathcal{I} \not\models X \to Y$ and $\mathcal{I} \cup \mathcal{A} \not\models X \overset{Z}{\not\Rightarrow} Y$.

Output: An ordered pair of schemes $(R_1 = (\Omega, \mathcal{I}_1 \cup \mathcal{A}_1), R_2 = (\Omega, \mathcal{I}_2 \cup \mathcal{A}_2))$, being the decomposition of R according to $X \to Y \overset{Z}{\supset\!\!-} T \to U$.

Method:

var $\mathcal{I}_Z, \mathcal{I}_{Z1}, \mathcal{I}_{Z2}$: set of fdi's := $\emptyset$
$\mathcal{A}_Z$: set of afd's := $\emptyset$

begin
for each $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i$ **in** $\mathcal{I}$ **do**
if $\mathcal{I} \models X_i \to Y_i$ or $\mathcal{I} \models Z_i \to Z$ or $U \subseteq T$ or
({both} $X \subseteq T$ and $U - T \subseteq Y - X$)
then
$\mathcal{I}_Z := \mathcal{I}_Z \cup \{X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i\}$
od
for each $X_j \overset{Z_j}{\not\Rightarrow} Y_j$ **in** $\mathcal{A}$ **do**
if $\mathcal{I} \models Z_j \to Z$
then
$\mathcal{A}_Z := \mathcal{A}_Z \cup \{X_j \overset{Z_j}{\not\Rightarrow} Y_j\}$
od
for each $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i$ **in** $\mathcal{I}_Z$ **do**
if $\mathcal{I}_Z \cup \{X \to Y\} \cup \{T \to U\} \not\models X_i \to Y_i$ and
$\mathcal{I}_Z \cup \{X \to Y\} \cup \{T \to U\} \cup \mathcal{A}_Z \not\models X_i \overset{Z_i}{\not\Rightarrow} Y_i$
then
$\mathcal{I}_{Z1} := \mathcal{I}_{Z1} \cup \{X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i\}$
od
for each $X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i$ **in** $\mathcal{I}_Z$ **do**
if $\mathcal{I}_Z \not\models X_i \to Y_i$ and
$\mathcal{I}_Z \cup \{X \not\to Y\} \cup \mathcal{A}_Z \not\models X_i \overset{Z_i}{\not\Rightarrow} Y_i$
then
$\mathcal{I}_{Z2} := \mathcal{I}_{Z2} \cup \{X_i \to Y_i \overset{Z_i}{\supset\!\!-} T_i \to U_i\}$
od
return$(R_1 = (\Omega, \mathcal{I}_{Z1} \cup \{X \to Y\} \cup \{T \to U\} \cup \mathcal{A}_Z)$,

$$R_2 = (\Omega, \mathcal{I}_{Z2} \cup \mathcal{A}_Z \cup \{X \not\twoheadrightarrow Y\}))$$

end

□

This algorithm again treats the trivial fdi's in such a way that the decomposition that is generated in case of all trivial fdi's is equivalent to the inherited decomposition into *HNF*.

6.4 The "FDI" Normal Form

In this section we illustrate an algorithm for horizontal decomposition of a relation scheme, according to its fdi's. The algorithm decomposes the scheme (and subschemes) until no subscheme can be decomposed any further. This is formalized by defining the following normal form:

Definition 6.9 A scheme $R = (\Omega, \mathcal{I} \cup \mathcal{A})$ is said to be in *FDI-Normal Form, (FDINF)* iff for all $X \to Y \overset{Z}{\supset\!\!-} T \to U \in \mathcal{I}$ either $\mathcal{I} \models X \to Y$ or $\mathcal{I} \cup \mathcal{A} \models X \not\overset{Z}{\twoheadrightarrow} Y$.

A decomposition $(R_1, \ldots, R_n)$ is in *FDINF* iff all the $R_i, i = 1 \ldots n$ are in *FDINF*.

□

If only trivial fdi's are given the horizontal decomposition into *FDINF* generates the inherited decomposition into *HNF*. If all fdi's are cfd's, the horizontal decomposition into *CNF* (Conditional Normal Form) is obtained. If all fdi's are ifd's, the horizontal decomposition into *INF* is obtained. So the decomposition using fdi's generalizes all previous decompositions. (We could also create "clean" decompositions using fdi's).

To illustrate the horizontal decomposition into *FDINF*, we modify Example 2.1 once more:

Example 6.1 Recall Example 2.1. In Section 6.1 we showed three fdi's for *STAFF*:

- $emp\ div \to job\ man\ dep \overset{div}{\supset\!\!-} emp\ div \to sal$
- $dep\ div \to job \overset{div}{\supset\!\!-} emp\ div \to job\ sal\ man$
- $dep\ div \to job \overset{div}{\supset\!\!-} man\ div \to dep$

The goals given in Example 2.1 are not all represented in this example, to keep it reasonably small. Figure 6.1 shows a decomposition tree for the decomposition of *STAFF* into *FDINF*.

Note that in $STAFF_{11}$ the fdi $dep\ div \rightarrow job \overset{div}{\supset\!\!-} emp\ div \rightarrow job\ sal\ man$ has been applied, although the fdi $dep\ div \rightarrow job \overset{div}{\supset\!\!-} man\ div \rightarrow dep$ would have given the same result, since the decomposition only depends on the "left fd" of an fdi.

In $STAFF_2$, $dep\ div \not\twoheadrightarrow job$ holds: $dep\ div \rightarrow job \overset{div}{\supset\!\!-} job\ sal\ man$ and $dep\ div \rightarrow job \overset{div}{\supset\!\!-} man\ div \rightarrow dep$ induce $dep\ div \rightarrow job \overset{div}{\supset\!\!-} emp\ div \rightarrow job\ sal\ man\ dep$. But since $emp\ div \not\twoheadrightarrow job\ man\ dep$, $emp\ div \not\twoheadrightarrow job\ sal\ man\ dep$ also holds, which induces that $dep\ div \not\overset{div}{\longrightarrow} job$ holds in $STAFF_2$. Hence $STAFF_2$ cannot be decomposed further on.

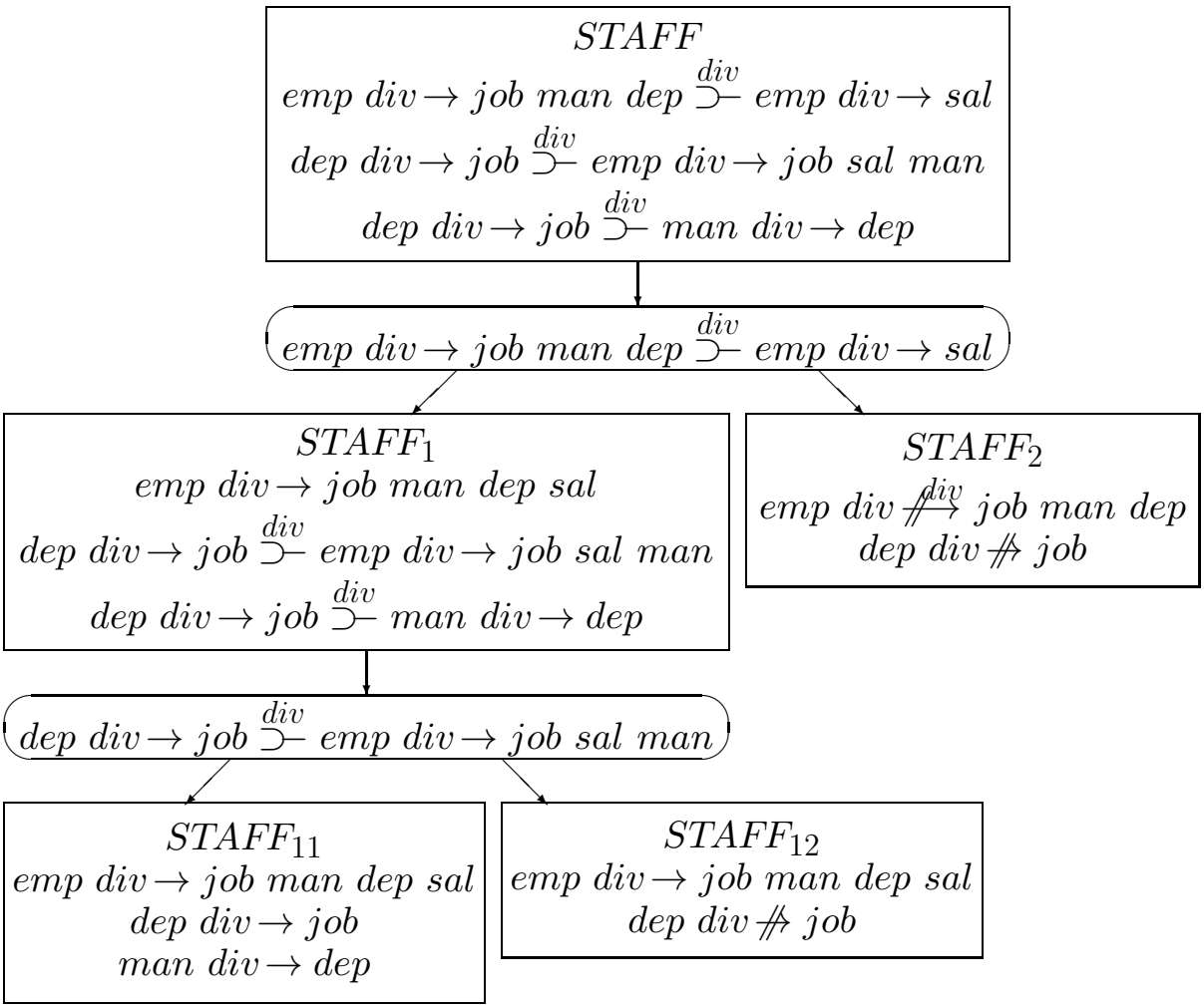


Figure 6.1: A decomposition tree for *STAFF*, into *FDINF*.

Recall Table 4.1. After the decomposition of *staff* into *FDINF*, as shown in Figure 6.1, one obtains a very strange result: all tuples belong to $staff_2$. The reason for this result is that our instance does not reflect the structure of a company for which the database scheme is chosen: In the "large" *div*isions every *emp*loyee should have only one *job*, one *man*ager and one *dep*artment for every *div*ision he works for. If we consider the Los Angeles *div*ision as a large *div*ision, the secretary Murrel should not be working for three *man*agers. If we delete the tuples

Murrel	secretary	Wallace	1000	sales	Los Angeles
Murrel	secretary	Diamond	1000	sales	Los Angeles

the Los Angeles *div*ision becomes a "large" *div*ision, and will be included in $staff_{12}$. But for becoming a part of $staff_{11}$ each *dep*artment should have only one *job*, which means that the meaning of the attribute *job* should relate to the *dep*artment, whereas in our example it relates to *emp*loyees. □

Chapter 7

Decomposition with fsi's

In the previous chapter we claimed that the functional dependency implication was the most general partial implication between fd's. In this chapter however we introduce a still more general constraint: the *functional dependency set implication*.
This chapter covers [16]. The horizontal decomposition according to an fdi not only induces the "right" fd, but also the "right" fd of all other fdi's with the same "left fd". In Example 6.1 we have seen this: the fdi $dep\ div \rightarrow job \overset{div}{\supset\!\!-} emp\ div \rightarrow job\ sal\ man$ and $dep\ div \rightarrow job \overset{div}{\supset\!\!-} man\ div \rightarrow dep$ have the same condition: $dep\ div \rightarrow job$. In this chapter we introduce a constraint which enables us to combine such fdi's into one new constraint.

7.1 Functional Dependency Set Implications

The general idea behind the functional dependency implication is that if some fd holds in part of the database, this may induce some other fd in that part of the database. In this chapter we generalize this idea: if some set of fd's holds in a part of the database, this may induce another set of fd's in that part of the database.
There are two reasons for introducing this new constraint:

1. The first reason is a semantic one: if several fdi's have the same left fd, then the decomposition according to one of these fdi's induces the right fd's of all these fdi's. Hence it is better to combine these fdi's into one constraint.

2. The second reason is both semantic and technical: the semantic reason is that by putting more than one fd in the left part we obtain a new and more general class of constraints. But this also has a technical advantage, even if we combine two fdi's into one of these new constraints (and obtain a weaker constraint): the maximal number of subrelations that result from a horizontal decomposition is exponential in the number of fdi's. In our examples we have had to be careful of not generating too many subrelations. If several fdi's are combined into one constraint the number of subrelations decreases (but the number of different kinds of exceptions that can be distinguished also decreases).

Let us first recall Example 2.1 and 6.1. If in a *div*ision every *dep*artment treats only one *job*, every *emp*loyee has only one *job* and every *man*ager supervises only one *job* (for this *div*ision), then (the *div*ision is so large that) every *emp*loyee works in only one *dep*artment and has only one *sal*ary, and every *man*ager supervises (*emp*loyees) in only one *dep*artment (for that *div*ision).

This constraint will be written as:

$$\{div\ dep \rightarrow job, div\ emp \rightarrow job, div\ man \rightarrow job\} \overset{div}{\supset\!\!-} \{div\ emp \rightarrow dep\ sal, div\ man \rightarrow dep\}$$

Note that none of the fd's of the second (or "implied") set are logical consequences of the first set of fd's. They are said to be implied by the first set of fd's by observing the real world.

We now define this constraint, and the horizontal decomposition induced by it, more formally.

Definition 7.1 Let R be a relation scheme, $Z \subseteq \Omega$, and let $\mathcal{F}_1$ and $\mathcal{F}_2$ be sets of fd's over Ω, such that $\forall X \rightarrow Y \in \mathcal{F}_1 \cup \mathcal{F}_2 : Z \subseteq X$.

- A relation instance r of R satisfies the *functional dependency set implication (fsi)* $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$, iff in every Z-complete set of tuples in r, in which all the fd's of $\mathcal{F}_1$ hold, all the fd's $\mathcal{F}_2$ must hold too.
- The scheme R satisfies $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ iff all the instances of R satisfy $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$.

□

The requirement that all "left hand sides" of the fd's of $\mathcal{F}_1$ and $\mathcal{F}_2$ must include Z is necessary because we want to generate the fd's in the subrelations that result from a horizontal decomposition step. We can eliminate this restriction if we modify the definition as for fdi's:

Definition 7.2 Let R be a relation scheme, $Z \subseteq \Omega$, and let $\mathcal{F}_1$ and $\mathcal{F}_2$ be sets of fd's over Ω.

- A relation instance r of R satisfies the *unrestricted functional dependency set implication (ufsi)* $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ iff in every Z-unique Z-complete set of tuples in r, in which all the fd' of $\mathcal{F}_1$ hold, all the fd's of $\mathcal{F}_2$ must hold too.
- The scheme R satisfies $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ iff all the instances of R satisfy $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$.

□

The reason for considering Z-unique Z-complete sets of tuples is the same as for ufdi's, and makes that the ufsi's are not more powerful than the fsi's:

Remark 7.1 *The ufsi* $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ *is equivalent to the fsi* $\mathcal{F}_1' \overset{Z}{\supset\!\!-} \mathcal{F}_2'$ *where* $\mathcal{F}_1' = \{XZ \to Y \mid X \to Y \in \mathcal{F}_1\}$ *and* $\mathcal{F}_2' = \{XZ \to Y \mid X \to Y \in \mathcal{F}_2\}$. □

The fdi's of Chapter 6 are special fsi's where $\mathcal{F}_1$ and $\mathcal{F}_2$ each contain only one fd. Since fd's, cfd's and ifd's are special fdi's they are fsi's too. In particular fd's can be expressed in many ways as fsi's, some of which are fdi's. $X \to Y$ is equivalent to $\{X \to X\} \overset{X}{\supset\!\!-} \{X \to Y\}$ for instance, which is an fdi (in fact even a cfd). But $X \to Y$ is also equivalent to $\emptyset \overset{X}{\supset\!\!-} \{X \to Y\}$ for instance, which is not an fdi.

Definition 7.3 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme, $Z \subseteq \Omega$, $\mathcal{F}$ a set of fd's over Ω, such that $\forall X \to Y \in \mathcal{F} : Z \subseteq X$.

- For every instance r of R, the *selection for* $\mathcal{F}_Z$ *of* r, $\sigma_{\mathcal{F}_Z}(r)$, is the largest Z-complete subset (of tuples) of r in which all fd's of $\mathcal{F}$ hold.
- The *selection for* $\mathcal{F}_Z$ *of* R, $\sigma_{\mathcal{F}_Z}(\mathcal{R})$, is a scheme $R_1 = (\Omega, \Delta, dom, M_1, SC_1)$. The calculation of SC_1 will be described in Section 7.3. SC_1 contains $\mathcal{F}$ of course. M_1 explains that all instances of R_1 must be the selection for $\mathcal{F}_Z$ of the instances of R.

□

We require $Z \subseteq X$ for all $X \to Y \in \mathcal{F}$ to make sure that X-values of r are not split up by taking a selection for $\mathcal{F}_Z$.

Definition 7.4 The *horizontal decomposition of an instance r, according to the fsi $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$*, is the ordered pair (r_1, r_2), where $r_1 = \sigma_{\mathcal{F}_{1_Z}}(r)$ and $r_2 = r - r_1$.
The *horizontal decomposition of a scheme R, according to the fsi $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$*, is the ordered pair (R_1, R_2), where $R_1 = \sigma_{\mathcal{F}_{1_Z}}(R)$ and $R_2 = R - R_1$. □

Note from Definition 7.4 that the horizontal decomposition of a scheme, according to $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ does not depend on $\mathcal{F}_2$, but it induces the $\mathcal{F}_2$ in R_1. Hence one can always perform a horizontal decomposition to generate a subrelation with a "desirable" set of fd's $\mathcal{F}_1$, by using the "trivial" fsi $\mathcal{F}_1 \overset{Z}{\supset\!-} \emptyset$.

In r_2, which contains the exceptions, for every Z-value at least one of the fd's of $\mathcal{F}_1$ must not hold. The following definition formalizes this new notion of "exception".

Definition 7.5 Let R be a relation scheme, $Z \subseteq \Omega$, and let $\mathcal{F}$ be a set of fd's, such that $\forall X \to Y \in \mathcal{F} : Z \subseteq X$.

- A relation instance r of R satisfies the *anti-functional dependency set (afs)* $\not\mathcal{F}_Z$ iff in every nonempty Z-complete set of tuples, in r, at least one fd of $\mathcal{F}$ does not hold.
- The scheme R satisfies $\not\mathcal{F}_Z$ iff all the instances of R satisfy $\not\mathcal{F}_Z$. □

Definition 7.6 Let $R = (\Omega, \Delta, dom, M, SC)$ be a relation scheme. Let r be an instance of R.

- The *selection for $\not\mathcal{F}_Z$ of r*, $\sigma_{\not\mathcal{F}_Z}(R)$, is the largest Z-complete set of tuples of r in which $\not\mathcal{F}_Z$ holds.
- The *selection for $\not\mathcal{F}_Z$ of R*, $\sigma_{\not\mathcal{F}_Z}(R)$ is the scheme $R_2 = (\Omega, \Delta, dom, M_2, SC_2)$. The calculation of SC_2 will be described in Section 7.3. M_2 explains that all instances of R_2 must be the selection for $\not\mathcal{F}_Z$ of the instances of R. □

One can easily see that $R - \sigma_{\mathcal{F}_Z}(R) = \sigma_{\not\mathcal{F}_Z}(R)$, hence the horizontal decomposition of R according to $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ is $(\sigma_{\mathcal{F}_{1_Z}}(R), \sigma_{\not\mathcal{F}_{1_Z}}(R))$.

The afd's (of Chapter 6) are afs' $\not{\mathcal{F}}_Z$ for which $\mathcal{F}$ contains only one fd.

From now on we shall assume that the set of constraints SC of a relation scheme R consists of a set $\mathcal{I}$ of fsi's and a set $\mathcal{A}$ of afs'. (Note that $\mathcal{I}$ contains the fd's, cfd's, ifd's and fdi's, and that $\mathcal{A}$ contains the ad's and afd's).

As in the previous chapters we first show how to detect conflict. Therefore we need the following special set of fd's:

Definition 7.7 $FSAT_{\mathcal{I}}(\mathcal{F})$ is the smallest possible set of fd's, such that:

1. $\mathcal{F} \subseteq FSAT_{\mathcal{I}}(\mathcal{F})$.
2. If $\mathcal{F}_{i_1} \subseteq FSAT_{\mathcal{I}}(\mathcal{F})$ and $\mathcal{F}_{i_1} \stackrel{Z_i}{\supset\!\!-} \mathcal{F}_{i_2} \in \mathcal{I}$ then $\mathcal{F}_{i_2} \subseteq FSAT(\mathcal{F})$.
3. $FSAT_{\mathcal{I}}(\mathcal{F}) = (FSAT_{\mathcal{I}}(\mathcal{F}))^*$.

□

$FSAT_{\mathcal{I}}(\mathcal{F})$ can be constructed starting from $\mathcal{F}$ and by repeatedly trying to satisfy 2 and 3 of the definition.

Note that $FSAT_{\mathcal{I}}(\mathcal{F}) = FSAT_{\mathcal{I}\cup\mathcal{F}}(\emptyset)$. This equality will be used several times without further notice.

Lemma 7.1 $FSAT_{\mathcal{I}}(\mathcal{F}) = \{P \rightarrow Q \mid \mathcal{I} \cup \mathcal{F} \models P \rightarrow Q\}$.

Proof Consider $Arm(FSAT_{\mathcal{I}}(\mathcal{F}))$. By Definition 7.7 it is clear that $\mathcal{I} \cup \mathcal{F}$ holds in $Arm(FSAT_{\mathcal{I}}(\mathcal{F}))$. Hence all the fd-consequences of $\mathcal{I} \cup \mathcal{F}$ also hold. By Theorem 2.1 this implies that all these fd's are in $(FSAT_{\mathcal{I}}(\mathcal{F}))^*$. Part 3 of Definition 7.7 implies that these fd's are in $FSAT_{\mathcal{I}}(\mathcal{F})$.

The opposite inclusion is obvious from Definition 7.7.

□

From the above lemma we immediately deduce that for all sets of fd's $\mathcal{F}$ over Ω holds that $\mathcal{I} \models \mathcal{F}$ iff $\mathcal{F} \subseteq FSAT_{\mathcal{I}}(\emptyset)$.

With the above lemma we can easily show how to detect conflict:

Theorem 7.1 *$\mathcal{I} \cup \mathcal{A}$ is in conflict iff for some afs $\not{\mathcal{F}}_Z$ of $\mathcal{A}$, $\mathcal{I} \models \mathcal{F}$ holds.*

Proof The if-part is trivial.
For the only-if-part, consider $Arm(FSAT_{\mathcal{I}}(\emptyset))$, in which $\mathcal{I}$ holds. Hence if $\mathcal{I} \cup \mathcal{A}$ is in conflict some $\not{\mathcal{F}}_Z$ of $\mathcal{A}$ does not hold. By Theorem 2.1 this implies that all fd's of $\mathcal{F}$ are consequences of $FSAT_{\mathcal{I}}(\emptyset)$, which implies that $\mathcal{F} \subseteq FSAT_{\mathcal{I}}(\emptyset)$ by part 3 of Definition 7.7. Lemma 7.1 then says that $\mathcal{I} \models \mathcal{F}$.

□

This Theorem shows that the set Z of an afs $\not\!\!A_{\setminus Z}$ is not important for the conflict detection.

Algorithm 7.1 *Conflict Detection*

Input: $\mathcal{I}, \mathcal{A}$, a set of fsi's and a set of afs'.

Output: *true* or *false*.

Method:

for each $\not\!\!A_{\setminus Z}$ **in** $\mathcal{A}$ **do**
  if $\mathcal{I} \models \mathcal{F}$
    then
      return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

The if-test $\mathcal{I} \models \mathcal{F}$ can be calculated by a membership algorithm for $FSAT_{\mathcal{I}}(\emptyset)$. This is essentially the same as for cfd's, given in Algorithm 4.2. We shall describe the algorithm in the next section. The time-complexity becomes $O(n^3 r^2 m)$ where $r = \sharp\Omega$, $n =$ the number of the fd's in all fsi's of $\mathcal{I}$ together, and $m =$ the number of the fd's in all afs' of $\mathcal{A}$ together.

7.2 The Implication Problem for fsi's and afs'

The implication problem for fsi's is very similar to that of fdi's, except that we use a different set of inference rules.

We denote the set of all fd's which are consequences of a set $\mathcal{F}$ of fd's by $\mathcal{F}^*$. The set of all fd's $X \to Y$ of $\mathcal{F}^*$ for which $Z \subseteq X$ is denoted by $\mathcal{F}^{*Z}$.

For fsi's we have the following inference rules:

$(FS1)$: if $\mathcal{F}_2 \subseteq \mathcal{F}_1^{*Z}$ and $\forall X \to Y \in \mathcal{F}_1 : Z \subseteq X$ then $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$.

$(FS2)$: if $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ and $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_3$ then $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2 \cup \mathcal{F}_3$.

$(FS3)$: if $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ and $\mathcal{F}_2 \overset{Z}{\supset\!\!-} \mathcal{F}_3$ then $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_3$.

$(FS4)$: if $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ and $Z \to Z'$ then $\mathcal{F}_1^{*Z'} \overset{Z'}{\supset\!\!-} \mathcal{F}_2^{*Z'}$.

$(FS5)$: if $\mathcal{F}_1$ holds and $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ then $\mathcal{F}_2$ holds and if $\mathcal{F}_2$ holds and $\forall X \to Y \in \mathcal{F}_1 \cup \mathcal{F}_2 : Z \subseteq X$ then $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ holds.

□

As fd's are special fsi's the use of fd's in these rules is allowed. In fact $FS5$ shows all representations of fd's as fsi's.

Before we prove the soundness of these rules we first show how to deduce the rules for fdi's from $F1, \ldots, F3, FS1, \ldots, FS5$. We include $F1, \ldots, F3$, since they are needed to calculate $\mathcal{F}^{*Z}$ from $\mathcal{F}$.

Lemma 7.2 *The rules $F1, \ldots, F3, FS1, \ldots, FS5$ are complete for fdi's (and hence also for ifd's, cfd's and fd's).*

Proof We show how to deduce the rules $FI0, \ldots, FI5$ from $FS1, \ldots, FS5$:

$FI0$: Let $Z \subseteq X$, $Z \subseteq T$, $Y \subseteq X$, $Z' \subseteq X'$, $Z' \subseteq T$, $Y' \subseteq X'$, $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U\}$. Then $X \to Y$ holds (by $F1$), hence $T \to U$ holds by the first part of $FS5$. Since $Z' \subseteq T$ and $Z' \subseteq X'$ and $T \to U$ hold, the second part of $FS5$ induces that $\{X' \to Y'\} \overset{Z'}{\supset\!-} \{T \to U\}$ holds.

$FI1$: Let $Z \subseteq X$ and $Z \subseteq T$ and either $U \subseteq T$ or both $X \subseteq T$ and $U - T \subseteq Y - X$. Then either $T \to U$ is trivial ($F1$) or $T \to U$ follows from $T \to UY$ and $UY \to Y$ by $F3$, where $T \to UY$ follows from $X \to Y$ by $F2$ and $UY \to Y$ holds by $F1$. Hence $\{T \to U\} \subseteq \{X \to Y\}^{*Z}$. $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U\}$ then follows by $FS1$.

$FI2$: Let $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U\}$ and $W \subseteq V$. Then $\{T \to U\} \overset{Z}{\supset\!-} \{TV \to UW\}$ holds by $FS1$ and together these two fsi's induce $\{X \to Y\} \overset{Z}{\supset\!-} \{TV \to UW\}$ by $FS3$.

$FI3$: Let $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U\}$ and $\{X \to Y\} \overset{Z}{\supset\!-} \{U \to V\}$. Then $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U \ , \ U \to V\}$ holds by $FS2$. Together with $\{T \to U \ , \ U \to V\} \overset{Z}{\supset\!-} \{T \to V\}$ (which holds by $F3$ and $FS1$) this induces $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to V\}$ by $FS3$.

$FI4$: is a special case of $FS3$.

$FI5$: Let $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U\}$ and $Z \to Z'$. Then $\{X \to Y\}^{*Z'} \overset{Z'}{\supset\!-} \{T \to U\}^{*Z'}$ holds by $FS4$. $\{XZ' \to Y\} \overset{Z}{\supset\!-} \{Z \to Z'\}$ holds by $FS5$ and together with $\{XZ' \to Y\} \overset{Z}{\supset\!-} \{XZ' \to Y\}$ ($FS1$) this induces $\{XZ' \to Y\} \overset{Z}{\supset\!-} \{XZ' \to Y \ , \ Z \to Z'\}$ by $FS2$. From the two fd's $XZ' \to Y$ and $Z \to Z'$ and the inclusion $Z \subseteq X$ one easily

deduces $X \to Y$. Hence by $FS1$ and $FS3$ one obtains $\{XZ' \to Y\} \overset{Z}{\supset\!-} \{X \to Y\}$. Hence $\{XZ' \to Y\}^{*Z'} \overset{Z'}{\supset\!-} \{X \to Y\}^{*Z'}$ holds by $FS4$. $FS3$ applied to $\{XZ' \to Y\}^{*Z'} \overset{Z'}{\supset\!-} \{X \to Y\}^{*Z'}$ and $\{X \to Y\}^{*Z'} \overset{Z'}{\supset\!-} \{T \to U\}^{*Z'}$ gives $\{XZ' \to Y\}^{*Z'} \overset{Z'}{\supset\!-} \{T \to U\}^{*Z'}$. Applying $FS3$ again, with $\{XZ' \to Y\} \overset{Z'}{\supset\!-} \{XZ' \to Y\}^{*X'}$ ($FS1$) gives $\{XZ' \to Y\} \overset{Z'}{\supset\!-} \{T \to U\}^{*Z'}$. Since $TZ' \to U \in \{T \to U\}^{*Z'}$ $FS1$ and $FS3$ give $\{XZ' \to Y\} \overset{Z'}{\supset\!-} \{TZ' \to U\}$. □

The most important element in the above proof (used for $FI5$) is described by the following remark:

Remark 7.2 *If $Z \to Z'$ and $\forall X \to Y \in \mathcal{F} : Z \subseteq X$ then $\mathcal{F}$ and $\mathcal{F}^{*Z'}$ are equivalent. In general however, (if $Z \not\to Z'$) $\mathcal{F}$ is more powerful than $\mathcal{F}^{*Z'}$.*

□

We now turn to the proof of the soundness of the inference rules.

Theorem 7.2 *The rules $FS1, \ldots, FS5$ are sound.*

Proof This is very similar to the proof for fdi's. We give the proof for $FS4$ as an example (since this is the most complicated one).

Let $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ and $Z \to Z'$ hold in an instance r. Consider an arbitrary Z'-complete set of tuples s in r, in which all fd's of $\mathcal{F}_1^{*Z'}$ hold. (For the other Z'-complete sets there is nothing to prove). We prove that $\mathcal{F}_2^{*Z'}$ holds in this set of tuples.

Since $Z \to Z'$ s is also Z-complete. Suppose some fd $X \to Y$ of $\mathcal{F}_1$ does not hold in s. Then $XZ' \to Y$ does not hold since $XZ' \to Y$ and $Z \to Z'$ (and $Z \subseteq X$) would induce $X \to Y$. Since $XZ' \to Y \in \mathcal{F}_1^{*Z'}$ (since $Z' \subseteq XZ'$ and $X \to Y$ induces $XZ' \to Y$ by augmentation) we have a contradiction that $\mathcal{F}_1^{*Z'}$ holds in s.

The fsi $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ then induces that $\mathcal{F}_2$ holds in this Z-complete set of tuples. Since all fd's of $\mathcal{F}_2^{*Z'}$ are consequences of $\mathcal{F}_2$, $\mathcal{F}_2^{*Z'}$ holds in s.

□

Lemma 7.1 shows that $FSAT$ is a set which characterizes the fd's that are a consequence of $\mathcal{I}$. The following Lemma shows that these fd's can be generated using our inference rules.

Lemma 7.3 $FSAT_{\mathcal{I}}(\mathcal{F}) = \{P \rightarrow Q \mid \mathcal{I} \cup \mathcal{F} \vdash P \rightarrow Q\}$.

Proof From Lemma 7.1 and Theorem 7.2 we know that: $\{P \rightarrow Q \mid \mathcal{I} \cup \mathcal{F} \vdash P \rightarrow Q\} \subseteq FSAT_{\mathcal{I}}(\mathcal{F})$.
For the opposite inclusion we show that the property, that all elements of $FSAT_{\mathcal{I}}(\mathcal{F})$ can be deduced from $\mathcal{I} \cup \mathcal{F}$, remains valid throughout the construction of $FSAT_{\mathcal{I}}(\mathcal{F})$.

- If $P \rightarrow Q \in \mathcal{F}$ then the property is trivial.
- If $P \rightarrow Q$ is added to $FSAT_{\mathcal{I}}(X, Y)$ in part 2 of Definition 7.7 then $P \rightarrow Q \in \mathcal{F}_{i_2}$ for some $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2} \in \mathcal{I}$. By the induction hypothesis all fd's of $\mathcal{F}_{i_1}$ can be inferred from $\mathcal{I} \cup \mathcal{F}$. Hence by rule $FS5$ $\mathcal{I} \cup \mathcal{F} \vdash \mathcal{F}_{i_2}$. Rule $FS5$ applied to $\mathcal{F}_{i_2}$ and $\mathcal{F}_{i_2} \overset{Z_i}{\supset\!-} \{P \rightarrow Q\}$ (holding by $FS1$) gives $\mathcal{I} \cup \mathcal{F} \vdash P \rightarrow Q$.
- If $P \rightarrow Q$ is added in part 3 of Definition 7.7 then it can be deduced from fd's for which the property holds, by using $F1 \ldots F3$. Hence $\mathcal{I} \cup \mathcal{F} \vdash P \rightarrow Q$.

□

If one chooses $\mathcal{F} = \emptyset$ then the following result becomes obvious:

Corollary 7.1 *$F1 \ldots F3$, $FS1 \ldots FS5$ are complete for the inference of fd's from a set of fsi's.*

□

In the construction of $FSAT_{\mathcal{I}}(\emptyset)$ only those fsi's $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2}$ of $\mathcal{I}$ are used (in part 2) for which $\mathcal{I} \models \mathcal{F}_{i_1}$ (and hence also $\mathcal{I} \models \mathcal{F}_{i_2}$). This leads to the following lemma:

Lemma 7.4 *Let $\mathcal{I}_Z = \{\mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2} \in \mathcal{I} : \mathcal{I} \models Z_i \rightarrow Z$ or $\mathcal{I} \models \mathcal{F}_{i_1}\}$. Then $\mathcal{I} \models P \rightarrow Q$ iff $\mathcal{I}_Z \models P \rightarrow Q$ (for any Z).*

□

The proof of the completeness of our inference rules for fsi's uses a similar technique as for fdi's.

Lemma 7.5 *Let $\mathcal{I}_Z$ be as in Lemma 7.4. Let $\mathcal{F}$ be such that $\forall X \rightarrow Y \in \mathcal{F} : Z \subseteq X$. If $P \rightarrow Q \in FSAT_{\mathcal{I}_Z}(\mathcal{F})$ then $\mathcal{I} \models P \rightarrow Q$ or $\mathcal{I} \models P \rightarrow Z$.*

Proof We prove that the property remains valid throughout the construction of $FSAT_{\mathcal{I}_Z}(\mathcal{F})$.

- If $P \rightarrow Q \in \mathcal{F}$ then $P \rightarrow Z$ is trivial.

- If $P \to Q$ is added in part 2 of Definition 7.7 then $P \to Q \in \mathcal{F}_{i_2}$ for some $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2} \in \mathcal{I}_Z$. There are two possibilities (by the definition of $\mathcal{I}_Z$): $\mathcal{I} \models \mathcal{F}_{i_1}$ or $\mathcal{I} \models Z_i \to Z$.
  - If $\mathcal{I} \models \mathcal{F}_{i_1}$ then $\mathcal{I} \models \mathcal{F}_{i_2}$ by $FS5$, hence obviously $\mathcal{I} \models P \to Q \in \mathcal{F}_{i_2}$.
  - If $\mathcal{I} \models Z_i \to Z$ then $\mathcal{I} \models P \to Z$ by augmentation (since $Z_i \subseteq P$ if $P \to Q \in \mathcal{F}_{i_2}$).
- If $P \to Q$ is added in part 3 then it is derived from other fd's (already in $FSAT_{\mathcal{I}_Z}(\mathcal{F})$) by $F1, \ldots, F3$.
  - If $Q \subseteq P$ then $P \to Q$ is trivial.
  - If $P = P'P''$, $Q = Q'Q''$, with $P' \to Q'$ already in $FSAT_{\mathcal{I}_Z}(\mathcal{F})$ and $Q'' \subseteq P''$, then $P \to Q$ or $P \to Z$ is deduced from $P' \to Q'$ or $P' \to Z$ by augmentation.
  - If $P \to O$ and $O \to Q$ already are in $FSAT_{\mathcal{I}_Z}(\mathcal{F})$ then $P \to Q$ or $P \to Z$ is deduced from $P \to O$ or $P \to Z$ and $O \to Q$ or $O \to Z$ by transitivity.

□

The following lemma partially solves the membership problem for fsi's:

Lemma 7.6 *Let $\mathcal{I}_Z$ be as in Lemma 7.4. Let $\mathcal{F}_1$ be such that $\forall X \to Y \in \mathcal{F}_1 : Z \subseteq X$, $\mathcal{F}_2$ such that $\forall X \to Y \in \mathcal{F}_2 : Z' \subseteq X$, and let $\mathcal{I} \models Z' \to Z$. Then $\mathcal{I}_Z \models \mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2^{*Z}$ iff $\mathcal{F}_2 \subseteq FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$.*

Proof If $\mathcal{I}_Z \models \mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2^{*Z}$ then obviously $\mathcal{F}_2^{*Z} \subseteq FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$. Since $\mathcal{I} \models Z' \to Z$ $\mathcal{F}_2^{*Z}$ is equivalent to $\mathcal{F}_2$ by Remark 7.2. Hence also $\mathcal{F}_2 \subseteq FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$ by part 3 of Definition 7.7.
For the converse we proceed as in Lemma's 7.3 and 7.5, by proving that the property remains valid throughout the construction of $FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$.

- If $\mathcal{F}_2 = \mathcal{F}_1$ then rule $FS1$ gives $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2^{*Z}$.
- If $\mathcal{F}_2$ is added to $FSAT_{\mathcal{I}_Z}(\mathcal{F})$ in part 2 of Definition 7.7 then $\mathcal{F}_2 = \mathcal{F}_{i_2}$ for some $\mathcal{F}_{i_1} \overset{Z}{\supset\!-} \mathcal{F}_{i_2} \in \mathcal{I}_Z$. There are 2 possibilities (by the definition of $\mathcal{I}_Z$):
  - If $\mathcal{I} \models \mathcal{F}_{i_1}$ then $\mathcal{I}_Z \models \mathcal{F}_{i_1}$ by Lemma 7.4. Hence $\mathcal{I}_Z \models \mathcal{F}_{i_2}$ by rule $FS5$. Hence also $\mathcal{I}_Z \models \mathcal{F}_{i_2}^{*Z}$ since $\mathcal{F}_{i_2}$ induces $\mathcal{F}_{i_2}^{*Z}$. $\mathcal{F}_1 \overset{Z}{\supset\!-} \emptyset$

(holding by $FS1$) and $\emptyset \overset{Z}{\supset\!\!-} \mathcal{F}_{i_2}^{*Z}$ (a representation for fd's, by $FS5$) induce $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_{i_2}^{*Z} = \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2^{*Z}$ by rule $FS3$.

– If $\mathcal{I} \models Z_i \rightarrow Z$ then we have that $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_{i_1}^{*Z}$ by induction. Since $Z_i \rightarrow Z$, $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$ induces $\mathcal{F}_{i_1}^{*Z} \overset{Z}{\supset\!\!-} \mathcal{F}_{i_2}^{*Z}$ by $FS4$. Hence $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_{i_2}^{*Z} = \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2^{*Z}$ by $FS3$.

- If $\mathcal{F}_2$ is added to $FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$ in part 3 of Definition 7.7 then $\mathcal{F}_2 \subseteq \mathcal{F}^*$ for some $\mathcal{F}$ that was a part of $FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$ already.
  From Lemma 7.5 we know that for all $X \rightarrow Y \in \mathcal{F} : \mathcal{I}_Z \vdash X \rightarrow Y$ or $\mathcal{I}_Z \vdash X \rightarrow Z$.
  – If $\mathcal{I}_Z \vdash X \rightarrow Y$ then $\mathcal{I}_Z \vdash XZ \rightarrow Y$ (by $F2$), hence $\mathcal{I}_Z \vdash \mathcal{F}_1 \overset{Z}{\supset\!\!-} \{XZ \rightarrow Y\}$ by $FS3$ on $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \emptyset$ and $\emptyset \overset{Z}{\supset\!\!-} XZ \rightarrow Y$ ($FS5$).
  – If $\mathcal{I}_Z \vdash X \rightarrow Z$ then $\mathcal{I}_Z \vdash \mathcal{F}_1 \overset{Z}{\supset\!\!-} \{XZ \rightarrow Y\}$ holds by induction (since $XZ \rightarrow Y \in \{X \rightarrow Y\}^{*Z}$).
  Let $\mathcal{F}' = \{XZ \rightarrow Y \in \mathcal{F} \mid \mathcal{I}_Z \vdash X \rightarrow Z$ or $\mathcal{I}_Z \vdash X \rightarrow Y\}$, then by $FS2$ $\mathcal{I}_Z \vdash \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}'$. One can easily see that $\mathcal{F}^{*Z} = \mathcal{F}'^{*Z}$ (using $F1 \ldots F3$), hence $\mathcal{I}_Z \vdash \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}^{*Z}$ by $FS3$ on $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}'$ and $\mathcal{F}' \overset{Z}{\supset\!\!-} \mathcal{F}'^{*Z} = \mathcal{F}^{*Z}$ ($FS1$). Hence by $FS3$ and $FS1$ one infers $\mathcal{I}_Z \vdash \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$.

□

From the proof of Lemma 7.6 one can see that:

Corollary 7.2 *Let $\mathcal{I}_Z$ be as in Lemma 7.4, let $\mathcal{F}_1$ and $\mathcal{F}_2$ be such that $\forall X \rightarrow Y \in \mathcal{F}_1 \cup \mathcal{F}_2$, $Z \subseteq X$. Then $\mathcal{I}_Z \vdash \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ iff $\mathcal{F}_2 \subseteq FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$.* □

To complete the proof of the completeness of $F1 \ldots F3$, $FS1 \ldots FS5$ for fsi's it remains to show that the fsi's of $\mathcal{I} - \mathcal{I}_Z$ have no influence on $\mathcal{I} \models \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$. To prove this we need a complicated construction of an instance, similar to that of Lemma 6.6, which we shall also be needing to prove the completeness for mixed fsi's and afs'. Therefore we include the properties of this instance, related to afs', in the following lemma:

Lemma 7.7 *Let $\mathcal{I} \cup \mathcal{A}$ be not in conflict. Let $\mathcal{I}_Z$ be as in Lemma 7.4. Let $\mathcal{A}_Z = \{\not\!\mathcal{A}_{i Z_i} \in \mathcal{A} \mid \mathcal{I} \models Z_i \rightarrow Z\}$. Let $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ (or let $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \not\!\mathcal{A}_{1 Z}$). Then we can construct an instance in which $\mathcal{I} \cup \mathcal{A}$ holds but in which $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ (resp. $\not\!\mathcal{A}_{1 Z}$) does not hold.*

Proof Suppose $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$. We shall see later that in $r_1 = Arm(FSAT_{\mathcal{I}_Z}(\mathcal{F}_1))$ $\mathcal{A}_Z$ holds. By Lemma 7.6 $\mathcal{F}_2 \not\subset FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$, hence $\mathcal{F}_2$ does not hold in r_1. From Theorem 2.1 we can easily deduce that this means that for some $X \to Y \in \mathcal{F}_2$, $X \overset{X}{\not\Longrightarrow} Y$ holds in r_1, hence $X \overset{Z}{\not\Longrightarrow} Y$ holds (by $F1$ and $FA3$). We also know (from rule $FA2$) that $\not\Join_{1_Z}$ cannot hold in r_1.

In r_1 a number of fsi's of $\mathcal{I} - \mathcal{I}_Z$ and a number of afs' of $\mathcal{A} - \mathcal{A}_Z$ may not hold. This will be solved by "adding" copies of $s = Arm(FSAT_{\mathcal{I}}(\emptyset))$. In s $\mathcal{I} \cup \mathcal{A}$ holds, as one can easily see.

Let some $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2} \in \mathcal{I} - \mathcal{I}_Z$ not hold in r_1. Then (Theorem 2.1) all fd's of $\mathcal{F}_{i_1}$ hold and some fd's of $\mathcal{F}_{i_2}$ do not hold. Since $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2} \notin \mathcal{I}_Z$ for some fd $T \to U \in \mathcal{F}_{i_1} : \mathcal{I} \not\models T \to U$.
Let the values that occur in s be renamed such that they all become different from the values of r_1, except that for some $t_1 \in r_1$, $t_2 \in s$: $t_1[\overline{Z_i}] = t_2[\overline{Z_i}]$, where $\overline{Z_i} = \{A \mid \mathcal{I} \models Z_i \to A\}$. The "modified" union of r_1 and s satisfies the following properties:

- In $r_1 \cup s$, $\mathcal{I}_Z$ still holds: let $\mathcal{F}_{j_1} \overset{Z_j}{\supset\!\!-} \mathcal{F}_{j_2} \in \mathcal{I}_Z$ not hold. Then there are two cases:
  1. either $\mathcal{I} \models \mathcal{F}_{j_1}$, hence $\mathcal{I} \models \mathcal{F}_{j_2}$ by $FS5$, and if $\mathcal{F}_{i_2}$ no longer holds then $\exists t_1' \in r_1$, $\exists t_2' \in s$, $\exists T_j \to U_j \in \mathcal{F}_{j_2}$: $t_1'[T_j] = t_2'[T_j]$ and $t_1'[U_j] \neq t_2'[U_j]$, and hence $T_j \subseteq \overline{Z_i}$ and $t_1'[T_j] = t_2'[t_j] = t_1[T_j]$, but then also $U_j \subseteq \overline{Z_i}$ since $\mathcal{I} \models T_j \to U_j$, hence $t_1'[U_j] = t_2'[U_j] = t_1[U_j]$, a contradiction.
  2. or $\mathcal{I} \models Z_i \to Z$, but then $T_j \subseteq \overline{Z_i}$ (which holds for some $T_j \to U_j \in \mathcal{F}_{j_2}$ that does not hold in $r_1 \cup s$) and $Z_j \to Z$ would induce $Z \subseteq \overline{Z_i}$, a contradiction with $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2} \notin \mathcal{I}_Z$.
- In $r_1 \cup s$, $\mathcal{A}_Z$ still holds since it is impossible to violate an afs by taking a union of two instances in which that afs holds.
- In $r_1 \cup s$ every fsi of $\mathcal{I} - \mathcal{I}_Z$ and every afs of $\mathcal{A} - \mathcal{A}_Z$ which already holds in r_1 (and also in s of course) still holds. For the afs' the reason is the same as for those of $\mathcal{A}_Z$. For the fsi's we have that if $\mathcal{F}_{k_1} \overset{Z_k}{\supset\!\!-} \mathcal{F}_{k_2} \in \mathcal{I} - \mathcal{I}_Z$ holds in R_1 then $\not\Join_{k_1 Z_k}$ holds in r_1 and s, and such an afs is not

violated by the union. $FA2$ shows that this afs implies $\mathcal{F}_{k_1} \stackrel{Z_k}{\supset\!\!-} \mathcal{F}_{k_2}$.

- In $r_1 \cup s$, $\not\mathcal{F}_{1_Z}$ still does not hold, since it does not hold in r_1 and since (as explained above) r_1 and r_2 do not "share" a common Z-value which could influence $\not\mathcal{F}_{1_Z}$ in R_1 (otherwise $Z \subseteq \overline{Z_i}$).
- In $r_1 \cup s$, $\not\mathcal{F}_{2_Z}$ may no longer hold, because it may not hold in s. But in the "r_1-part" of $r_1 \cup s$, $\mathcal{F}_2^Z$ still holds, since r_1 and s do not share a Z-value (and hence also no T-value for any $T \rightarrow U \in \mathcal{F}_2$).
- But in $r_1 \cup s$ the number of Z_i-values for which all $X_{i_n} \rightarrow Y_{i_n} \in \mathcal{F}_1$ hold (and for which some $T_{i_n} \rightarrow U_{i_n} \in \mathcal{F}_2$ does not hold) is decreased by one, since the Z_i-value containing t_1 collapses with the Z_i-value of s, containing t_2, (in which $\not\mathcal{F}_{i1_{Z_i}}$ holds), and since s has no Z_i-values for which all $X_{i_n} \rightarrow Y_{i_n} \in \mathcal{F}_{i_1}$ hold.

By repeating the above construction for all other Z_i-values for which $\mathcal{F}_{i_1}$ holds one can generate a relation in which $\mathcal{F}_{i_1} \stackrel{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$ holds (since $\not\mathcal{F}_{i1_{Z_i}}$ holds).

By then repeating the above construction for all fsi's of $\mathcal{I} - \mathcal{I}_Z$ which do not hold in $Arm(FSAT_{\mathcal{I}_Z}(\mathcal{F}_1))$ one generates a relation in which $\mathcal{I}$ holds (and $\mathcal{F}_1 \stackrel{Z}{\supset\!\!-} \mathcal{F}_2$ still does not hold).

For the afs' of $\mathcal{A} - \mathcal{A}_Z$ the construction proceeds in a similar way.

If $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \not\mathcal{F}_Z$ then we shall see later that in $Arm(FSAT_{\mathcal{I}_Z}(\mathcal{F}))$ $\mathcal{A}_Z$ holds, as well as $\mathcal{I}_Z \cup \{\mathcal{F}\}$. The same construction as above leads to an instance in which $\mathcal{I} \cup \mathcal{A}$ holds, and in which $\not\mathcal{F}_Z$ does not hold.

□

Theorem 7.3 *$F1 \ldots F3$, $FS1 \ldots FS5$ are complete for fsi's.*
Furthermore, let $\mathcal{F}_1$ and $\mathcal{F}_2$ be such that $\forall X \rightarrow Y \in \mathcal{F}_2 \cup \mathcal{F}_2 : Z \subseteq X$, then $\mathcal{I} \vdash \mathcal{F}_1 \stackrel{Z}{\supset\!\!-} \mathcal{F}_2$ iff $\mathcal{F}_2 \subseteq FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$.

Proof If $\mathcal{I} \models \mathcal{F}_1 \stackrel{Z}{\supset\!\!-} \mathcal{F}_2$ then by Lemma 7.7 $\mathcal{I}_Z \models \mathcal{F}_1 \stackrel{Z}{\supset\!\!-} \mathcal{F}_2$. Lemma 7.6 yields $\mathcal{F}_2 \in FSAT_{\mathcal{I}_Z}(\mathcal{F}_1)$, while Corollary 7.2 implies $\mathcal{I}_Z \vdash \mathcal{F}_1 \stackrel{Z}{\supset\!\!-} \mathcal{F}_2$.

The converse is trivial.

□

Before we solve the implication problem for mixed fsi's and afs' we first propose a set of inference rules:

*FSA*1 : if $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ and $\not\!\!\!\!\backslash\!\!\!\!\backslash_{2_Z}$ then $\not\!\!\!\!\backslash\!\!\!\!\backslash_{1_Z}$.

*FSA*2 : if $\not\!\!\!\!\backslash\!\!\!\!\backslash_{1_Z}$ and $\forall X \to Y \in \mathcal{F}_2 : Z \subseteq X$ then $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$.

*FSA*3 : if $\not\!\!\!\!\backslash\!\!\!\!\backslash_Z$ and $Z \to Z'$ then $\not\!\!\!\!\backslash\!\!\!\!\backslash^{*Z'}_Z$.

*FSA*4 : if $\not\!\!\!\!\backslash\!\!\!\!\backslash_{1_Z}$ and $\mathcal{F}_1^{*Z} \subseteq \mathcal{F}_2^{*Z}$ and $\forall X \to Y \subseteq \mathcal{F}_2 : Z \subseteq X$ then $\not\!\!\!\!\backslash\!\!\!\!\backslash_{2_Z}$.

Lemma 7.8 *Rules FSA*1, . . . , *FSA*4 *are sound.*

Proof This is fairly straightforward, using a similar argument as for the rules for fsi's. We prove rule *FSA*3 as an example.

Suppose $\not\!\!\!\!\backslash\!\!\!\!\backslash^{*Z'}_{Z'}$ does not hold. (The notation may be a bit confusing: $\not\!\!\!\!\backslash\!\!\!\!\backslash^{*Z'}_{Z'}$ means that in every Z' complete set of tuples (in an instance satisfying the constraint) at least one fd of $\mathcal{F}^{*Z'}$ must not hold.) Then in some Z'-complete set of tuples s all fd's of $\mathcal{F}^{*Z'}$ hold. Since s is also Z-complete (because of $Z \to Z'$) it remains to prove that all fd's of $\mathcal{F}$ hold in s.

Let $X \to Y \in \mathcal{F}$, then $XZ' \to Y \in \mathcal{F}^{*Z'}$. $Z \to Z'$ and $Z \subseteq X$ induce $X \to XZ'$ by $F2$. By $F3$ we infer that $X \to Y$ holds in s. □

Before we show the completeness of the rules for fsi's and afs', we show how to deduce the inference rules for fdi's and afd's from the rules for fsi's and afs'.

Remark 7.3 $F1, \ldots, F3, FS1, \ldots, FS5, FSA1, \ldots FSA4$ *are complete for fdi's and afs'.*

Proof We prove the rules *FIA*1, . . . , *FIA*3:

*FIA*1 : Suppose $\{Z \to V\}$, $\{X \to Y\} \overset{Z}{\supset\!-} \{T \to U\}$ and $T \overset{Z}{\not\!\!\!\!\longrightarrow} U$ hold. (We should write $\not\!\!\!\!\backslash\!\!\!\!\backslash_Z$ where $\mathcal{F} = \{T \to U\}$ to be exact.) By *FSA*1 we infer $X \overset{Z}{\not\!\!\!\!\longrightarrow} Y$. $X \to V$ holds by $F2$ (augmentation) on $Z \to V$. Hence $XV \to Y \in \{X \to Y\}^{*Z}$. Rule *FSA*4 then generates $XV \overset{Z}{\not\!\!\!\!\longrightarrow} Y$ from $X \overset{Z}{\not\!\!\!\!\longrightarrow} Y$. Rules *FSA*3 and *FSA*1 induce $XV \overset{V}{\not\!\!\!\!\longrightarrow} Y$ (from the trivial $\{XV \to Y\} \overset{V}{\supset\!-} \{XV \to Y\}^{*V}$. Rules *FSA*4 and *FSA*1 then induce $V \overset{V}{\not\!\!\!\!\longrightarrow} Y$ (using the trivial $\{V \to Y\} \overset{V}{\supset\!-} \{V \to Y\}^{*V}$.

*FIA*2 : This is a special case of *FSA*2.

*FIA*3 : This is a special case of *FSA*3.

□

Theorem 7.4 *Let* $\mathcal{I} \cup \mathcal{A}$ *be not in conflict.* $\mathcal{I} \cup \mathcal{A} \models \not\!\!A_Z$ *iff* $\mathcal{I}_Z \cup \mathcal{A}_Z \models \not\!\!A_Z$.

Proof The if-part is trivial.

For the only-if-part, suppose $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \not\!\!A_Z$. We first show that $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}$ cannot be in conflict.

Suppose $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}$ is in conflict. Then in $Arm(FSAT_{\mathcal{I}_Z}(\mathcal{F}))$ for some afs $\not\!\!A'_V$ of $\mathcal{A}_Z$, $\mathcal{F}'$ holds (by Theorem 7.1 and Theorem 2.1). Hence $\mathcal{I}_Z \cup \mathcal{F} \models \mathcal{F}'$. Since $\mathcal{I}_Z \models V \to Z$ Lemma 7.6 yields $\mathcal{I}_Z \models \mathcal{F} \overset{Z}{\supset\!\!-} \mathcal{F}'^{*Z}$.

$\not\!\!A'_V$ induces ${\not\!\!A'_Z}^{*Z}$ by rule *FSA*3. Rule *FSA*1 (with $\mathcal{F} \overset{Z}{\supset\!\!-} \mathcal{F}'^{*Z}$ induces $\not\!\!A_Z$, a contradiction with $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \not\!\!A_Z$.

Hence $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \not\!\!A_Z$ and $\mathcal{A}_Z$ holds in $Arm(FSAT_{\mathcal{I}_Z}(\mathcal{F}))$ (this was used in Lemma 7.7, considering that *FSA*2 deduces $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ from $\not\!\!A_{1_Z}$. Lemma 7.7 then says that there exists an instance in which $\mathcal{I} \cup \mathcal{A}$ holds and in which $\not\!\!A_Z$ does not hold. Hence $\mathcal{I} \cup \mathcal{A} \not\models \not\!\!A_Z$.

□

The proof of the above theorem immediately implies that:

Corollary 7.3 *The rules* $F1, \ldots, F3$,$FS1, \ldots, FS5$,$FSA1, FSA3$ *and* $FSA4$ *are complete for the inference of afs' from a set of fsi's and afs'.*

□

For the sake of completeness we show a generalization of Algorithm 4.2 for fsi's.

Algorithm 7.2 *Membership Detection for fsi's*

Input: $\mathcal{I} = \{\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2} \mid i = 1 \ldots n\}$; $\mathcal{F}_{0_1} \overset{Z_0}{\supset\!\!-} \mathcal{F}_{0_2}$.

Output: *true* or *false*. (meaning $\mathcal{I} \models \mathcal{F}_{0_1} \overset{Z}{\supset\!\!-} \mathcal{F}_{0_2}$ or not.)

Method:

var $P_1, \ldots, P_k$: set of attributes
{ one for each left hand side of each fd in any fsi $(0 \ldots n)$ }
change : boolean
i, j : integer

begin
  for $j := 1$ **to** k **do**
    $P_j := \overline{X_j}^{\mathcal{F}_{0_1}}$ { saturation for $\mathcal{F}_{0_1}$ }
  od
  repeat
    $change := false$
    for $j := 0$ **to** k **do**
      for $i := 0$ **to** k **do**
        if $X_j \subseteq P_i$
          then
            if $P_j \not\subset P_i$
              then
                begin
                  $P_i := P_i \cup P_j$
                  $change := true$
                end
      od
    od
    for $i := 1$ **to** n **do**
      if for all $X_j \rightarrow Y_j$ **of** $\mathcal{F}_{i_1}$ $Y_j \subseteq P_j$
        then
          if for some $X_j \rightarrow Y_j$ **of** $\mathcal{F}_{i_2}$ $Y_j \not\subset P_i$
            then
              begin
                $P_i := P_i \cup Z_j$
                $change := true$
              end
    od
  until $change = false$
  if for all $X_j \rightarrow Y_j$ **of** $\mathcal{F}_{0_2}$ $Y_j \subseteq P_j$
    then
      return($true$)
    else
      return($false$)
end

□

The correctness of Algorithm 7.2 can be proved exactly as for cfd's. The time-complexity is something like $O(n^3r^2)$, where $r = \sharp\Omega$ and $n =$ the number of fd's in the input.
Using Theorem 7.1 and 7.4 one can easily prove the following membership algorithm for afs' (considering the presence of fsi's):

Algorithm 7.3 *Membership Detection for afs' with fsi's*

Input: $\mathcal{I}, \mathcal{A}$, a set of fsi's and a set of afs, not in conflict; $\not\!\!\mathbb{A}_Z$ an afs.

Output: *true* or *false*

Method:

var $\mathcal{I}_Z$: set of fsi's := $\emptyset$
$\mathcal{A}_Z$: set of afs' := $\emptyset$
begin
for each $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2}$ **in** $\mathcal{I}$ **do**
if $\mathcal{I} \models \mathcal{F}_{i_1}$ or $\mathcal{I} \models Z_i \rightarrow Z$
then
$\mathcal{I}_Z := \mathcal{I}_Z \cup \mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2}$
od
for each $\not\!\!\mathbb{A}_{j_{Z_j}}$ **in** $\mathcal{A}$ **do**
if $\mathcal{I}_Z \cup \mathcal{F} \models \mathcal{F}_j$
then
return(*true*) { and exit }
od
return(*false*) { only reached if for-loop is done }

□

Rule *FSA*2 shows the influence of afs' on the implication problem for fsi's:

Theorem 7.5 *Let $\mathcal{I} \cup \mathcal{A}$ be not in conflict. $\mathcal{I} \cup \mathcal{A} \models \mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ iff $\mathcal{I}_Z \models \mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ or $\mathcal{I}_Z \cup \mathcal{A}_Z \models \not\!\!\mathbb{A}_{1_Z}$.*

Proof The if part is trivial (using *FSA*2 if $\mathcal{I}_Z \cup \mathcal{A}_Z \models \not\!\!\mathbb{A}_{1_Z}$).
For the only-if-part, assume that $\mathcal{I}_Z \not\models \mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ and $\mathcal{I}_Z \cup \mathcal{A}_Z \not\models \not\!\!\mathbb{A}_{1_Z}$. By the proof of Theorem 7.4 $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}_1$ is not in conflict. Hence by Lemma 7.7 there exists an instance in which $\mathcal{I} \cup \mathcal{A}$ holds and in which $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$ does not hold. Hence $\mathcal{I} \cup \mathcal{A} \not\models \mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2$.

□

An immediate consequence of the proof of the above theorem is:

Corollary 7.4 *The rules* $F1 \ldots F3, FS1 \ldots FS5$ *and* $FSA1 \ldots FSA4$ *are complete for the inference of fsi's from a set of fsi's and afs'.*

□

A membership algorithm for fsi's (considering the presence of afs') is easily deduced:

Algorithm 7.4 *Membership Detection for fsi's with afs'*

Input: $\mathcal{I}, \mathcal{A}$, a set of fsi's and a set of afs, not in conflict; $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$ an fdi.

Output: *true* or *false*

Method:
if $\mathcal{I} \models \mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$
  then
    return(*true*) { and exit }
if $\mathcal{I} \cup \mathcal{A} \models \not\mathbb{A}_{1_Z}$
  then
    return(*true*) { and exit }
  else
    return(*false*)

□

Algorithms 7.3 and 7.4 both take $O(n^4r^2 + n^3r^2m)$ time, where n is the number of fd's that occur in the fsi's of of $\mathcal{I}$, m is the number of fd's that occur in the afs' of $\mathcal{A}$ and $r = \sharp\Omega$. This time-complexity is actually the time needed for calculating $\mathcal{I}_Z$ and $\mathcal{A}_Z$, and is based on a membership algorithm for *FSAT* which takes $O(n^3r^2)$ time.

7.3 The Inheritance of fsi's and afs'

For the (further) decomposition of the subschemes that result from a horizontal decomposition step, we need to know the fsi's and afs' that hold in the subschemes. This *inheritance problem* is very similar to that for fdi's and afd's.

Notation 7.1 In the sequel we always treat the horizontal decomposition of a scheme $R = (\Omega, \mathcal{I} \cup \mathcal{A}$, according to $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2 \in \mathcal{I}$, into the subschemes $R_1 = \sigma_{\mathcal{F}_{1_Z}}(R) = (\Omega, \mathcal{I}_1 \cup \mathcal{A}_1)$, and $R_2 = \sigma_{\not\mathbb{A}_{1_Z}}(R) = (\Omega, \mathcal{I}_2 \cup \mathcal{A}_2)$. We

assume that $\mathcal{I} \cup \mathcal{A}$ is not in conflict, and also that $\mathcal{I} \cup \mathcal{A} \not\models \mathcal{F}_1$ and $\mathcal{I} \cup \mathcal{A} \not\models \not\!\!\mathbb{A}_{1_Z}$ (otherwise R_1 resp. R_2 would always be empty). We only require $\mathcal{I}_1 \cup \mathcal{A}_1$ and $\mathcal{I}_2 \cup \mathcal{A}_2$ to be generating for the sets of all dependencies which hold in R_1 and R_2.

□

Since fd's cannot be violated by taking a selection of a relation, Remark 4.3 also applies if the decompositions are based on fsi's instead of cfd's. This means that all the fd's which hold in R also hold in R_1 and $\mathcal{R}_2$.
The inclusions of Lemma 6.8 are easily translated into the following inclusions for fsi's and afs':

Lemma 7.9 *Using Notation 7.1 we have:*

- $\mathcal{I}_Z \cup \mathcal{F}_1 \subseteq \mathcal{I}_1 \subseteq \{\mathcal{F}'_1 \overset{Z'}{\supset\!-} \mathcal{F}'_2 \mid \mathcal{I} \cup \mathcal{A} \cup \mathcal{F}_1 \models \mathcal{F}'_1 \overset{Z'}{\supset\!-} \mathcal{F}'_2\}$.
- $\mathcal{I}_Z \subseteq \mathcal{I}_2 \subseteq \{\mathcal{F}'_1 \overset{Z'}{\supset\!-} \mathcal{F}'_2 \mid \mathcal{I} \cup \mathcal{A} \cup \not\!\!\mathbb{A}_{1_Z} \models \mathcal{F}'_1 \overset{Z'}{\supset\!-} \mathcal{F}'_2\}$.
- $\mathcal{A}_Z \subseteq \mathcal{A}_1 \subseteq \{\not\!\!\mathbb{A}'_{Z'} \mid \mathcal{I} \cup \mathcal{A} \cup \mathcal{F}_1 \models \not\!\!\mathbb{A}'_{Z'}\}$.
- $\mathcal{A}_Z \cup \not\!\!\mathbb{A}_{1_Z} \subseteq \mathcal{A}_2 \subseteq \{\not\!\!\mathbb{A}'_{Z'} \mid \mathcal{I} \cup \mathcal{A} \cup \not\!\!\mathbb{A}_{1_Z} \models \not\!\!\mathbb{A}'_{Z'}\}$.

□

The proof of the inheritance of fsi's and afs' relies on the construction of Lemma 7.7, which has already been used to solve the implication problem.

Theorem 7.6 *Using Notation 7.1, an fsi or afs must hold in R_1 (resp. $\mathcal{R}_2$) iff it is a consequence of $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}_1$ (resp. $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \not\!\!\mathbb{A}_{1_Z}$).*

Proof From Lemma 7.9 it follows that $(\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}_1)^* \subseteq (\mathcal{I}_1 \cup \mathcal{A}_1)^* \subseteq (\mathcal{I} \cup \mathcal{A} \cup \mathcal{F}_1)^*$ and also that $(\mathcal{I}_Z \cup \mathcal{A}_Z \cup \not\!\!\mathbb{A}_{1_Z})^* \subseteq (\mathcal{I}_2 \cup \mathcal{A}_2)^* \subseteq (\mathcal{I} \cup \mathcal{A} \cup \not\!\!\mathbb{A}_{1_Z})^*$. We prove that the first inclusions are equalities.

Let $\mathcal{I} \cup \mathcal{A} \cup \mathcal{F}_1 \models \not\!\!\mathbb{A}'_{Z'}$ but $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}_1 \not\models \not\!\!\mathbb{A}'_{Z'}$. We prove that $\not\!\!\mathbb{A}'_{Z'}$ does not hold in R_1.

$\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}_1 \not\models \not\!\!\mathbb{A}'_{Z'}$ implies that $\mathcal{I}_Z \cup \mathcal{A}_Z \cup \mathcal{F}_1 \cup \mathcal{F}'_1$ is not in conflict, by the proof of Theorem 7.4. By the proof of Lemma 7.7 one can construct an instance R in which $\mathcal{I} \cup \mathcal{A}$ holds but in which $\not\!\!\mathbb{A}'_{Z'}$ does not hold. One starts with $r_1 = Arm(FSAT_{\mathcal{I}_Z \cup \mathcal{F}_1}(\mathcal{F}'))$ this time, and adds copies of $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$ to obtain r. From the construction, used in the proof of Lemma 7.7 one can easily see that $r_1 = \sigma_{\mathcal{F}_{1_Z}}(r)$, since the copies of $Arm(FSAT_{\mathcal{I}}(\emptyset, \emptyset))$ have $\not\!\!\mathbb{A}_{1_Z}$ and have different Z-values than those occurring in r_1. Hence we obtain an instance in which $\mathcal{I} \cup \mathcal{A}$ holds, and such that $\not\!\!\mathbb{A}'_{Z'}$ does not hold in r_1. Hence $\not\!\!\mathbb{A}'_{Z'} \notin (\mathcal{I}_1 \cup \mathcal{A}_1)^*$.

The proof of the other three cases (an fsi in R_1, an afs in R_2 and an fsi in R_2) is similar, and also similar to the proof for cfd's and ad's, and therefore left to the reader.

□

From Theorem 7.6 one can easily deduce an algorithm which calculates the inherited dependencies in the same time as the membership algorithm. Hence this also is the time-complexity of the following algorithm:

Algorithm 7.5 *A Horizontal Decomposition Step with fsi's and afs'*

Input: $R = (\Omega, \mathcal{I} \cup \mathcal{A})$ and an fsi $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2 \in \mathcal{I}$. $\mathcal{I} \cup \mathcal{A}$ is assumed not to be in conflict, $\mathcal{I} \cup \mathcal{A} \not\models \mathcal{F}_1$ and $\mathcal{I} \cup \mathcal{A} \not\models \not\mathbb{A}_Z$.

Output: An ordered pair of schemes $(R_1 = (\Omega, \mathcal{I}_1 \cup \mathcal{A}_1), R_2 = (\Omega, \mathcal{I}_2 \cup \mathcal{A}_2))$, being the decomposition of R according to $\mathcal{F}_1 \overset{Z}{\supset\!\!-} \mathcal{F}_2$.

Method:

var $\mathcal{I}_Z, \mathcal{I}_{Z_1}, \mathcal{I}_{Z_2}$: set of fsi's := $\emptyset$
$\mathcal{A}_Z$: set of afs' := $\emptyset$
begin
for each $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$ **in** $\mathcal{I}$ **do**
if $\mathcal{I} \models \mathcal{F}_{i_1}$ or $\mathcal{I} \models Z_i \to Z$ or $\mathcal{F}_1 \models \mathcal{F}_2$
then
$\mathcal{I}_Z := \mathcal{I}_Z \cup \mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$
od
for each $\not\mathbb{A}_{j_{Z_j}}$ **in** $\mathcal{A}$ **do**
if $\mathcal{I} \models Z_j \to Z$
then
$\mathcal{A}_Z := \mathcal{A}_Z \cup \not\mathbb{A}_{j_{Z_j}}$
od
for each $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$ **in** $\mathcal{I}_Z$ **do**
if $\mathcal{I}_Z \cup \mathcal{F}_1 \cup \mathcal{F}_2 \not\models \mathcal{F}_{i_1}$ and $\mathcal{I}_Z \cup \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{A}_Z \not\models \not\mathbb{A}_{i_{1Z_i}}$
then
$\mathcal{I}_{Z_1} := \mathcal{I}_{Z_1} \cup \mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$
od
for each $\mathcal{F}_{i_1} \overset{Z_i}{\supset\!\!-} \mathcal{F}_{i_2}$ **in** $\mathcal{I}_Z$ **do**
if $\mathcal{I}_Z \not\models \mathcal{F}_{i_1}$ and $\mathcal{I}_Z \cup \not\mathbb{A}_1 \cup \mathcal{A}_Z \not\models \not\mathbb{A}_{i_{1Z_i}}$

then

$\mathcal{I}_{Z_2} := \mathcal{I}_{Z_2} \cup \mathcal{F}_{i_1} \overset{Z_i}{\supset\!-} \mathcal{F}_{i_2}$

od

return$(R_1 = (\Omega, \mathcal{I}_{Z_1} \cup \mathcal{F}_1 \cup \mathcal{F}_1 \cup \mathcal{A}_Z)$, $R_2 = (\Omega, \mathcal{I}_{Z_2} \cup \mathcal{A}_Z \cup \not\mathbb{1}_Z))$

end

□

7.4 The "FSI" Normal Form

In this section we illustrate an algorithm for horizontal decomposition of a relation scheme, according to its fsi's. The algorithm decomposes the scheme (and subschemes) until no subscheme can be decomposed any further. This is formalized by defining the following normal form:

Definition 7.8 A scheme $R = (\Omega, \mathcal{I} \cup \mathcal{A})$ is said to be in *FSI-Normal Form, (FSINF)* iff for all $\mathcal{F}_1 \overset{Z}{\supset\!-} \mathcal{F}_2 \in \mathcal{I}$ either $\mathcal{I} \models \mathcal{F}_1$ or $\mathcal{I} \cup \mathcal{A} \models \not\mathbb{1}_Z$.

A decomposition $(R_1, \ldots, R_n)$ is in *FSINF* iff all the $R_i, i = 1 \ldots n$ are in *FSINF*.

□

If only trivial fsi's are given the horizontal decomposition into *FSINF* generates the inherited decomposition into *HNF*. If all fsi's are cfd's, the horizontal decomposition into *CNF* (Conditional Normal Form) is obtained. If all fsi's are ifd's, the horizontal decomposition into *INF* is obtained. If all fsi's are fdi's, the horizontal decomposition into *FDINF* is obtained. So the decomposition using fdi's generalizes all previous decompositions. (We could also create "clean" decompositions using fsi's).

To illustrate the horizontal decomposition into *FSINF*, we modify Example 2.1 once more:

Example 7.1 Recall Example 2.1. In Section 7.1 we described the following fsi for $STAFF$:

$$\{div\ dep \rightarrow job, div\ emp \rightarrow job, div\ man \rightarrow job\} \overset{div}{\supset\!-} \{div\ emp \rightarrow dep\ sal, div\ man \rightarrow dep\}$$

In addition to this fsi, we also consider an fdi of Example 6.1: $emp\ div \rightarrow job\ man\ dep \overset{div}{\supset\!-} emp\ div \rightarrow sal$. The other fdi's are combined in our fsi.

The meaning of the job attribute has changed in this example: since we consider the *div*isions in which each *dep*artment has only one job as the "big" *div*isions, the job must actually belong to the *dep*artment, and not to the *emp*loyees. Otherwise, we would not have any "large" *div*isions in any real company, since every *dep*artment normally has a manager and some other employees like secretaries. So the fd $dev\ dep \rightarrow job$ would never be satisfied.

Figure 7.1 shows a decomposition tree for the decomposition of $STAFF$ into *FSINF*. The constraints that are shown are not exactly as generated by the decomposition algorithm.

We do not decompose the instance given in previous chapters, since these instances do not correspond to the new meaning of the attribute job. □

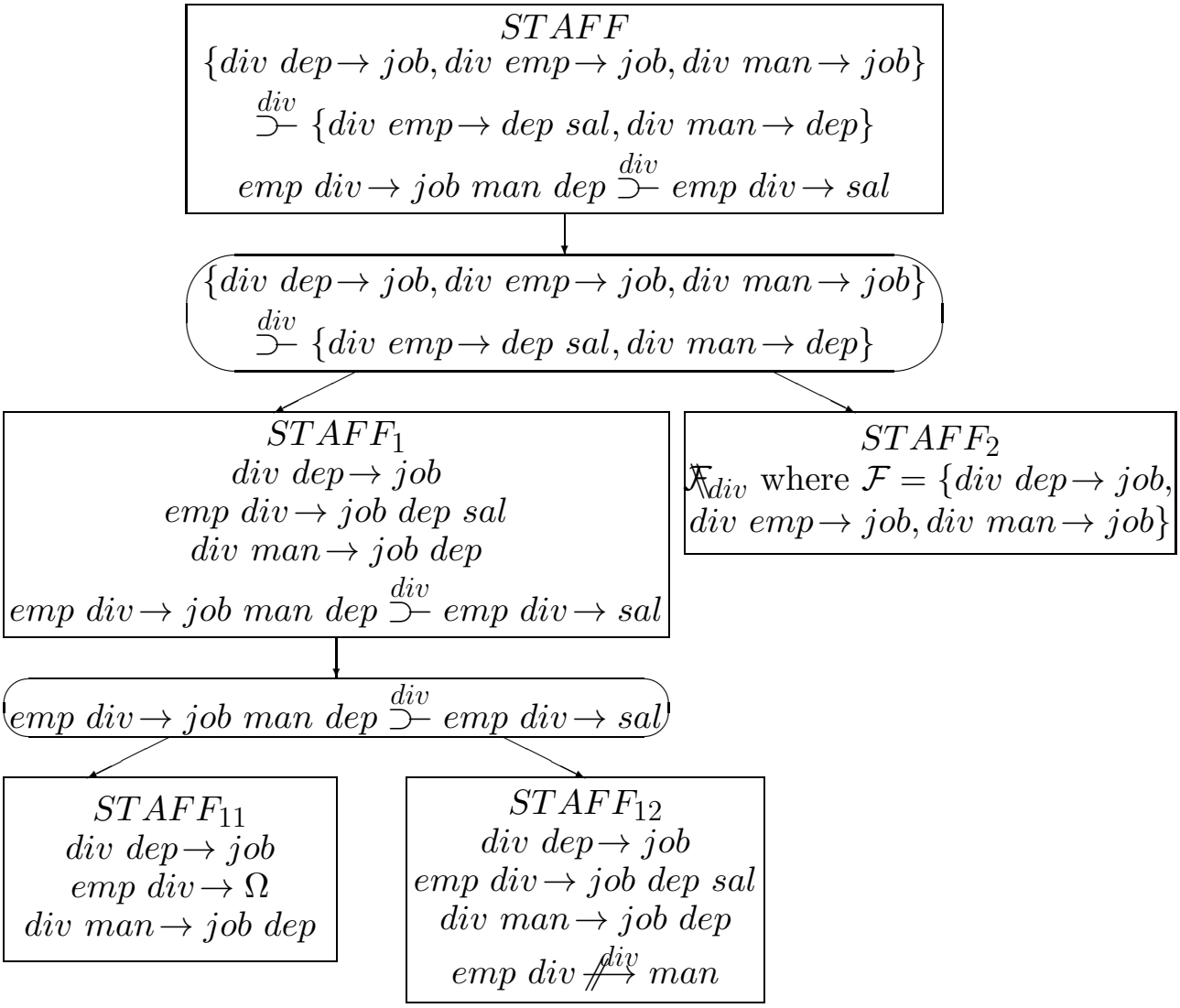


Figure 7.1: A decomposition tree for $STAFF$, into *FSINF*.

Chapter 8

The Update Problem

In this chapter we briefly discuss the effect of the horizontal decomposition on the problem of updating relations. This effect is negative of course, since the creation or removal of exceptions causes data to move from one subrelation to another, whereas the update in one big relation does not require this. Furthermore, depending on the implementation of instances, even just scanning for an item goes faster in one relation than in several relations (with the one relation as union), unless multiple processors are working in parallel.

However, the horizontal decomposition does allow an improvement in update-efficiency, for two reasons: the subrelations can be decomposed vertically, using the fd's, generated by the horizontal decomposition, and if the update algorithm is smart enough, the subrelations can be updated in parallel.

In this chapter we shall ignore the benefit of both the vertical decomposition and parallelism, when calculating the time-complexity. However, we shall develop a new normal form which allows an increased amount of parallelism in the update algorithm. Since this normal form is rather restrictive we show how to maximize the number of fd's that can be generated from the goals. The update-study is based on the decomposition using goals, so we do not consider cfd's, ifd's, fdi's and afs'.

This chapter covers [17]. It is not meant as an exhaustive study on how to optimize updating using horizontal decompositions, but it shows some general strategies that can be used in actual implementations.

8.1 The update algorithm

Before we present the complicated update algorithm, we first discuss an intuitive way to solve the update problem, using the decomposition tree for the relation scheme. We only consider insertions and deletions. Modifications can be (artificially) converted to a deletion followed by an insertion. We concentrate on the decomposition of an abstract relation scheme R, shown in Figure 8.1.

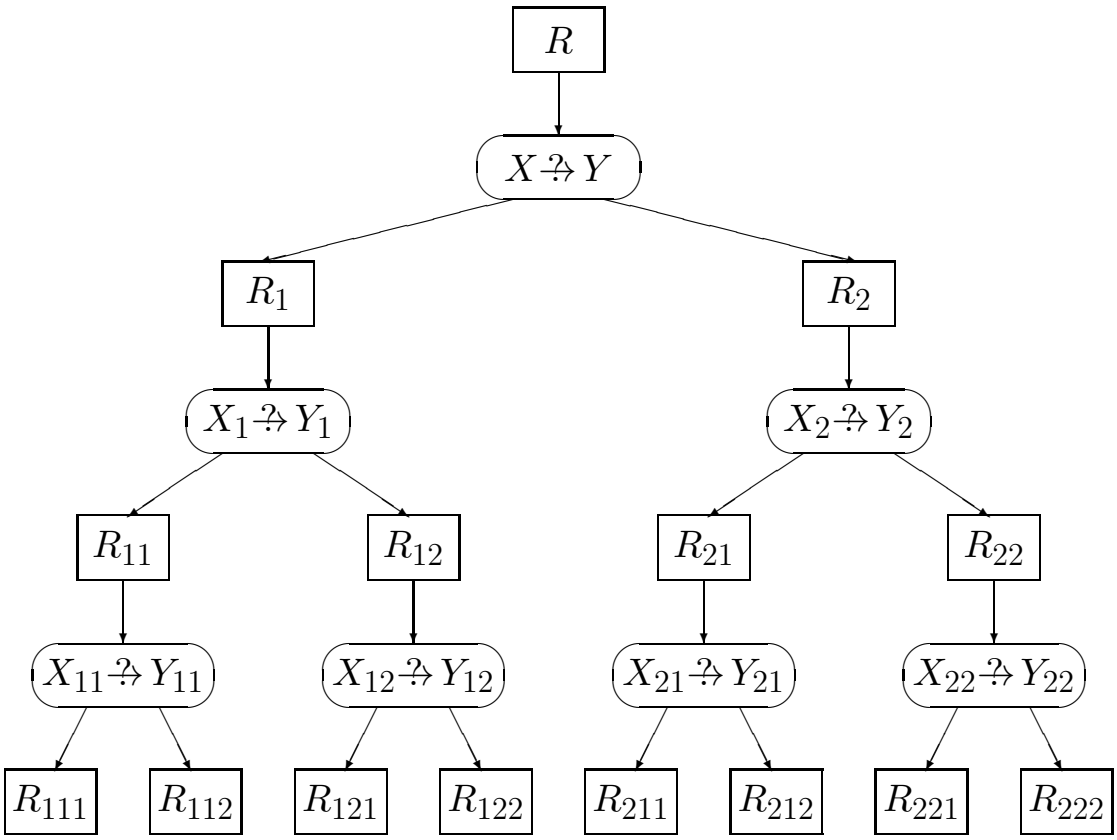


Figure 8.1: Decomposition of an abstract relation.

If a tuple t_1 has to be added to r (which is decomposed using $X \twoheadrightarrow Y$) it is possible that this insertion creates an exception to $X \to Y$. This means that there exist tuples $t_2 \ldots t_k$ (for some k) in r_1 with the same X-value as t_1, but with another Y-value. These k tuples plus the new one form an exception to $X \to Y$ and hence belong in r_2. This means that the k tuples have to be deleted from r_1 and inserted into r_2. Let us forget about that insertion into r_2 for a moment and concentrate on the deletion from r_1. Deleting the k tuples from r_1 can be easily done one by one. (We can use a procedure for inserting or deleting a tuple recursively.) r_1 may be

decomposed by $X_1 \overset{?}{\twoheadrightarrow} Y_1$ into r_{11} and r_{12}. Each time a tuple is deleted from r_1 this may cause an exception to $X_1 \rightarrow Y_1$ to disappear, causing m tuples to move from r_{12} to r_{11} (for some m). However, a tuple moving from r_{12} to r_{11} may be removed from r_{11} later on, when removing the next of the k tuples from r_1.

One can easily prove that an update algorithm, based on the above schedule, may cause many tuples to move between many subrelations, before they finally arrive in the right subrelation. In fact, a tuple may (in worst case) visit all the "final" subrelations of the decomposition, a number which is exponential in the number of goals. Hence this algorithm would take exponential time, not in the number of tuples, but in the number of goals.

We now present the algorithm for inserting and deleting tuples, which does not require exponential time. The main property of the algorithm is that the number of tuples that are moved between the subinstances is minimal. In particular, every tuple is inserted or deleted at most once at every "level" of the decomposition tree. (Since the depth of the tree is at most the number of goals this is linear). This is especially useful when the different subinstances are physically stored on different disks or computers, causing the tuples that move from one subrelation to another to move through a network. The "trivial" approach to the update-problem (shown above) could well saturate the network hardware and software.

In the algorithm we use the abbreviation r^{t_X} for $\sigma_{X=t[X]}(r)$, the selection of r containing the tuples having $t[X]$ as X-projection.

Algorithm 8.1 *Insertion and deletion of a set of tuples.*

Input: $rins, rdel$: sets of tuples to be inserted resp. deleted.

Output: No output; only an update to the instance is performed.

Method:

procedure *adjust* (r : (sub)instance {decomposed according to $X \overset{?}{\twoheadrightarrow} Y$};
$rins, rdel$: set of tuples {to be inserted resp. deleted});

var $r_1ins, r_2ins, r_1del, r_2del$: set of tuples := $\emptyset$

begin {*adjust*}
if r is a final subrelation

then
$r := r \cup rins - rdel$
else $\{r$ *is decomposed acc. to* $X \stackrel{?}{\twoheadrightarrow} Y\}$
begin
for each t **in** $rins$ **do**
if $t[X]$ occurs in r_2
then
if $X \not\twoheadrightarrow Y$ holds in $r_2 \cup rins^{t_X} - rdel^{t_X}$
then
$r_2ins := r_2ins \cup rins^{t_X} - rdel^{t_X}$
else
begin
$r_2del := r_2del \cup {r_2}^{t_X}$
$r_1ins := r_1ins \cup r_2^{t_X} \cup rins^{t_X} - rdel^{t_X}$
end
else $\{t[X]$ *does not occur in* $r_2\}$
if $X \to Y$ holds in $r_1 \cup rins^{t_X} - rdel^{t_X}$
then
$r_1ins := r_1ins \cup rins^{t_X} - rdel^{t_X}$
else
begin
$r_1del := r_1del \cup r_1^{t_X}$
$r_2ins := r_2ins \cup r_1^{t_X} \cup rins^{t_X} - rdel^{t_X}$
end
$rins := rins - rins^{t_X}$
$rdel := rdel - rdel^{t_X}$
od
for each t **in** $rdel$ **do**
if $t[X]$ occurs in r_2
then
if $X \not\twoheadrightarrow Y$ holds in $r_2 \cup r_2ins^{t_X} - rdel^{t_X}$
then
$r_2del := r_2del \cup rdel^{t_X}$
else
begin

$r_2del := r_2del \cup r_2^{t_X}$
$r_1ins := r_1ins \cup r_2^{t_X} - rdel^{t_X}$
end
else
$r_1del := r_1del \cup rdel^{t_X}$
$rdel := rdel - rdel^{t_X}$
od
$adjust(r_1, r_1ins, r_1del)$
$adjust(r_2, r_2ins, r_2del)$
end
end; $\{adjust\}$

□

Theorem 8.1 *Algorithm 8.1 correctly performs the insertions and deletions, in $O(m^2nr)$ time, where m is the number of tuples ($\sharp r + \sharp rins + \sharp rdel$), n is the number of goals ($\sharp\mathcal{G}$) and r is the number of attributes ($\sharp\Omega$).*

Proof The correctness of Algorithm 8.1 is easy to prove after the following observation: If t has to be inserted or deleted, then there are 4 possibilities after the part of the update, concerning all tuples (of *rins* and *rdel*) with the same X-value as t:

- $t[X]$ did occur in r_1 before the update (hence $X \to Y$ did hold for this X-value) and $X \to Y$ still holds for this X-value after the update. Then the update is performed in r_1.
- $t[X]$ did occur in r_1 before the update but after the update $X \to Y$ does not hold any more for this X-value, after the update. Then the tuples with this X-value must be moved to r_2.
- $t[X]$ did occur in r_2 before the update (hence $X \not\to Y$ did hold for this X-value) and $X \not\to Y$ still holds for this X=value after the update. Then the update is performed in r_2.
- $t[X]$ did occur in r_2 before the update but after the update $X \not\to Y$ does not hold any more for this X-value, after the update. Then the tuples with this X-value must be moved to r_1.

For the time-complexity we first estimate the time needed to perform the non-recursive actions which are performed in the algorithm. It is easy to

see that, if the tuples are stored in a reasonable way, all operations like "if $X \to Y$ holds in $r_2 \cup rins^{t_X} - rdel^{t_X}$" can be performed in $O(mr)$ time.

It remains to show how much time is involved in the "for each" loops and in the recursive calls to the procedure *adjust*. It is clear that the body of the for-loops for r is executed at most m times. Now assume that the insertions and deletions in r cause $m_1 \leq m$ tuples to move between r_1 and r_2. Since these tuples are deleted from r_1 (or r_2) and inserted into r_2 (resp. r_1) they cannot be moved between r_{11} and r_{12} and/or r_{21} and r_{22}. Hence if m_2 is the number of tuples moved between r_{11} and r_{12} or r_{21} and r_{22} we have that $m_1 + m_2 \leq m$. We can proceed this argument by induction, obtaining numbers $m_1, m_2, \ldots m_n$, with the property $m_1 + m_2 + \ldots + m_n \leq m$. The update stops at level n since the decomposition cannot be "deeper" than the number of goals, if we assume that no goal may be used twice in the same branch of the decomposition tree, which is guaranteed for the decompositions of Chapter 3 (both for *CNF* and for the inherited decomposition into *HNF*.) The total number of elements that cause execution of the body of a for-loop is at most 2 times the number of tuples that have to move between subinstances (once for deleting them somewhere and once for inserting them somewhere else) multiplied by the number of levels they go through (which is at most n). Hence the number of elements in all loops (on all levels) is $O(nm)$, which means that the time-complexity of the entire algorithm is $O(m^2nr)$. □

Looking at the time-complexity, Algorithm 8.1 (although polynomial) may not seem to be efficient, since the square of the number of tuples rapidly becomes a very large number. However, this is not only a "very"-worst case time-complexity and it also does not take into account any possible optimization for verifying fd's, or for finding tuples in an instance. It also does not consider vertical decompositions which can be applied to the final subrelations.

Note also that although Algorithm 8.1 is not linear in the number of tuples, the number of tuples that move between subinstances still is linear. This becomes very important if the subinstances are located in different sites of a computer network, since this is the number of tuples that goes through the network.

8.2 Normal Forms for Updating in Parallel

In this section we discuss how Algorithm 8.1 can be improved by updating the subinstances in parallel, as much as possible. This may not lead to an improvement of the worst-case time-complexity, but it certainly will lead to a better "average" update-performance, especially if the subinstances are distributed among different storage media, or even different computers.

The bottleneck of Algorithm 8.1 is that the sets r_1ins, r_1del, r_2ins and r_2del are calculated completely before the procedure *adjust* is called to update r_1 and r_2. It would be much more efficient if some updating of r_1 and r_2 could be done as soon as an X-complete subset of the tuples that have to move between r_1 and r_2 is determined. While this partial update is completed in the subinstances the next X-complete set can be calculated.

However, the updating process of r_1 (which is decomposed according to say $X_1 \twoheadrightarrow Y_1$) can only start if an X_1-complete set of tuples that has to move between r_{11} and r_{12} is calculated (otherwise we may get an exponential algorithm again). And since there is no relationship between X and X_1 in general, no such X_1-complete set of tuples can be found before r_1ins and r_1del are completely calculated.

The following definition puts a restriction on the decomposition steps that are allowed (i. e. on the goals that can be used), such that the subinstances R_1 and R_2 can be partially updated while the update of R is still not calculated completely.

Definition 8.1 Let the (sub)scheme R have a set $\mathcal{G}$ of goals, and let $X \twoheadrightarrow Y \in \mathcal{G}$ be such that neither $X \rightarrow Y$ not $X \not\twoheadrightarrow Y$ holds in R.

Decomposing R according to $X \twoheadrightarrow Y$ is called a *very safe decomposition step* if for all goals $T \twoheadrightarrow U$ of $\mathcal{G}$ for which $T \rightarrow U$ or $T \not\twoheadrightarrow U$ (already) holds in R, the fd $T \rightarrow X$ also holds in R.

The goal $X \twoheadrightarrow Y$ is then called a *very safe goal* for R.

□

From Theorem 3.3 and Definition 8.1 we conclude that if a subscheme $R_{i\ldots j}$ is decomposed according to a very safe goal, then all fd's or ad's that correspond to goals that have been used earlier in the decomposition

(or that hold in R already) are preserved. The condition for a goal to be very safe is a little stronger than the definition of a "safe goal" of [11].

Definition 8.2 A (sub)scheme R is said to be in *Very Safe Normal Form (VSNF)* if no goal of $\mathcal{G}$ is very safe for R.

A decomposition $R_{1\ldots1} \ldots R_{2\ldots2}$ is said to be in *Very Safe Normal Form* if it is obtained by very safe decomposition steps, and if all final subschemes are in *VSNF*.

□

Note that also for the *VSNF* we have the property that the depth of the decomposition tree is at most the number of goals.

If we have a kind of "lexicographic" ordering in the subinstances, first according the last X that is used, then the previous one etc., then the subinstances can be updated in parallel as follows:

Recall Figure 8.1. The tuples in r_{111} and r_{112} are first sorted on their X_{11}-value, then on their X_1-value, and finally on their X-value. This is possible in a very safe decomposition since $X \to X_1$ holds in R, R_1 and R_{11} and $X_1 \to X_{11}$ holds in R_1 and R_{11}. Since the tuples are sorted on their X_{11}-value first, the first X_{11}-value that is scanned forms an X_{11}-complete set of tuples as soon as a second X_{11}-value occurs. This X_{11}-complete set of tuples is also X_1-complete and X-complete, because of $X \to X_1 \to X_{11}$. Hence the updating procedure can continue with that X-complete set of tuples, before *rins* and *rdel* are calculated completely.

The main problem with the *VSNF* is that the condition for a decomposition to be in *VSNF* is rather strong, i. e. a number of goals may not be used for decomposition. The order in which the goals of $\mathcal{G}$ are used is very important. Using the goals in a different order may produce a different number of subrelations. Therefore we introduce an optimization strategy, which tries to maximize the number of goals that can be used. Since we are mainly interested in the leftmost part of a decomposition tree, i. e. the main part without the exceptions, we emphasize our optimization strategy on that part, and we do not care about how many goals are actually used, but for how many goals does the corresponding fd hold in the leftmost subscheme.

This problem is similar to a well known class of problems, called *NP* (for nondeterministic polynomial). One can nondeterministicly choose the "optimal" order on the goals, which produces the largest number of fd's. Many problems in *NP* are called *NP-complete*, meaning that all problems in *NP* can be reduced to such *NP-complete* problem in polynomial time. It is generally accepted that the *NP-complete* problems cannot be solved in (deterministic) polynomial time (although no one has been able to prove that yet). Early attempts to reduce some *NP-complete* problems to the optimization problem for the horizontal decomposition have led to the conclusion that this is not an *NP-complete* problem. In the sequel we develop a polynomial-time solution for the optimization problem.

Definition 8.3 A very safe decomposition of a scheme R is said to be an *Optimal Decomposition* if the number of goals for which the corresponding fd holds in the "leftmost branch" of the decomposition tree is maximal, and if the same property holds in all its subtrees.

The decomposition is then said to be in the *Optimal Normal Form (ONF)*. □

The optimization problem then is: given a set $\{X_1 \stackrel{?}{\twoheadrightarrow} Y_1, \ldots, X_n \stackrel{?}{\twoheadrightarrow} Y_n\}$ of n goals, and a set $\mathcal{F}$ of fd's, find a maximal sequence $X_{i_1} \stackrel{?}{\twoheadrightarrow} Y_{i_1} \ldots X_{i_k} \stackrel{?}{\twoheadrightarrow} Y_{i_k}$ such that

1. all $i_j \neq i_l$ if $j \neq l$.
2. for all j holds: $\mathcal{F} \cup \{X_{i_1} \rightarrow Y_{i_1}, \ldots, X_{i_j} \rightarrow Y_{i_j}\} \models X_{i_j} \rightarrow X_{i_{j+1}}$.

Calculating an Optimal Decomposition takes exponential time, since the generated decomposition may have 2^n subschemes. However, calculating only the leftmost branch of a decomposition tree takes polynomial time, for a very safe decomposition, or any of the normal forms of Chapter 3. We show that the optimization problem (for one branch of the tree only) can also be solved in polynomial time.

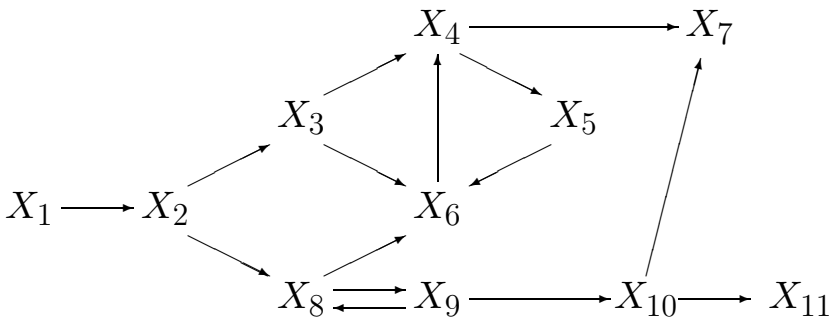
In the sequel we shall neglect the presence of a set $\mathcal{F} \cup \mathcal{A}$ of dependencies, to keep lemmas readable.

Lemma 8.1 *Let* $V = \{X_1, \ldots, X_n\}$, $A = \{X_i \rightarrow X_j \mid \mathcal{F}' = \{X_1 \rightarrow Y_1, \ldots, X_n \rightarrow Y_n\} \models X_i \rightarrow X_j$ *and* $i \neq j\}$. *The directed graph* $\mathcal{G}_0 = (V, A)$ *can be constructed in polynomial time.*

Proof This is obvious since only $O(n^2)$ fd-membership tests have to be executed.

□

The figure below shows part of an example of such a directed graph. A number of "transitive" arcs have been omitted, to keep the figure readable.



We cannot use this graph for finding the longest path immediately, since condition 2 of the update problem says that not $\mathcal{F}'$ but only the already used goals may induce the fd's $X_i \to X_j$. Therefore we remove the "impossible" arcs in an inductive way.

Lemma 8.2 *The directed graph $\mathcal{G}$ is defined as the fixpoint of the following inductively constructed series of graphs:*

We start with $\mathcal{G}_0$.

Let $X_i \to X_j$ be an arc of $\mathcal{G}_p$. Find for each X_i all the X_k for which an $X_k \to X_i$ is an arc of $\mathcal{G}_p$. If the fd's $X_k \to Y_k$ united with $X_i \to Y_i$ imply $X_i \to X_j$ then $X_i \to X_j$ is an arc of $\mathcal{G}_{p+1}$ else it is not.

The directed graph $\mathcal{G}$ can be generated in polynomial time.

Proof The inductive process stops after at most n^2 steps, since each graph must contain at least one arc less than the previous graph, and the graph $\mathcal{G}_0$ contains at most n^2 arcs. Furthermore, every step requires only $O(n)$ fd-membership tests.

□

The directed graph, generated by Lemmas 8.1 and 8.2 may have some cycles, indicated in the example graph above. We first show how to remove those cycles.

Lemma 8.3 *Let $\mathcal{F}_1 \subseteq \mathcal{F}'$ be the set of fd's corresponding to the X-es of a cycle of the graph $\mathcal{G}$, not contained in another cycle. Then for all $X_i \to X_j$ in that cycle $\mathcal{F}_1 \models X_i \to X_j$.*

Proof Suppose that for some $X_i \to X_j$ in the cycle we have that $\mathcal{F}_1 \not\models X_i \to X_j$, hence some fd $X_l \to Y_l$ of $\mathcal{F}' - \mathcal{F}_1$ is needed in the deduction of $X_i \to X_j$. Let $\mathcal{F}'' \subseteq \mathcal{F}'$ be the minimal set of fd's such that $\mathcal{F}'' \models X_i \to X_j$.

Remark 3.1 implies that $\mathcal{F}'' - \{X_l \to Y_l\} \models X_i \to X_l$. Since $\mathcal{F}'' - \{X_l \to Y_l\} \models X_i \to X_l$ some $X_p \to Y_p \in \mathcal{F}'' - \mathcal{F}_1$ is needed to deduce $X_i \to X_l$. Hence $\mathcal{F}'' - \{X_l \to Y_l\} - \{X_p \to Y_p\} \models X_i \to X_p$. We can continue this argument (by induction) to obtain an $X_m \to Y_m \in \mathcal{F}'' - \mathcal{F}_1$ such that $\mathcal{F}_1 \models X_i \to X_m$, which implies that $X_i \to X_m$ is an arc of $\mathcal{G}$. Since $X_m \to Y_m \in \mathcal{F}''$ we know that $X_m \to Y_m$ is needed to deduce $X_i \to X_j$ (because $\mathcal{F}''$ is minimal), hence $X_m \to X_i$ must be an arc of $\mathcal{G}$ too. (Otherwise $\mathcal{G}$ would not be the fixpoint of its construction.) Since $X_i \to X_m$ and $X_m \to X_i$ are arcs of $\mathcal{G}$, and X_m is not in the cycle (since $X_m \to Y_m \notin \mathcal{F}_1$) the cycle is contained in a bigger cycle (by inserting X_m in the cycle), a contradiction.

□

Lemma 8.4 *In a cycle there exists an order, i. e. a sequence $X_{i_1} \dots X_{i_l}$ such that for each $j \in 1 \dots l-1$ $\{X_{i_1} \to Y_{i_1}, \dots, X_{i_j} \to Y_{i_j}\} \models X_{i_j} \to X_{i_{j+1}}$. This order can be calculated in polynomial time.*

Proof Consider a cycle $X_1 \dots X_l$. Let $\mathcal{F}_1 = \{X_1 \to Y_1, \dots, X_l \to Y_l\}$. Consider $X_1 \to X_2$. $\mathcal{F}_1 \models X_1 \to X_2$ (by Lemma 8.3). Let $X_i \to Y_i$ be needed in the deduction of $X_1 \to X_2$, then (by Remark 3.1) $\mathcal{F}_1 - \{X_i \to Y_i\} \models X_i \to X_1$. For $X_i \to X_1$ we can repeat this argument, and obtain an $X_j \to Y_j$ such that $\mathcal{F}_1 - \{X_i \to Y_i\} - \{X_j \to Y_j\} \models X_j \to X_i$. We can repeat this process until we obtain an $X_p \to Y_p$ which (all by itself) induces $X_q \to X_p$ (for the q of the previous step). The order, constructed by this inductive process (which starts with X_p and ends with X_1) does not necessarily include all $X - es$ of $\mathcal{F}_1$ yet: some fd's may not be used. These can be added to the end of the order (since they are implied by the other fd's). The order thus obtained satisfies the lemma. It is obvious that this construction takes only polynomial time.

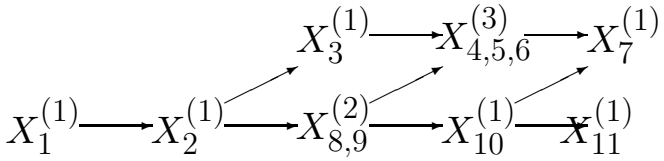
□

Since we now have shown that all elements of a cycle are equivalent we can construct a graph $\mathcal{G}_1$, in which all cycles are replaced by a single vertex, and in which every vertex has a "weight", which is the number of vertices of $\mathcal{G}$ it represents.

Every (directed) path in $\mathcal{G}_1$ (not contained in a longer path) represents a possible (leftmost branch of a) decomposition tree. Now all we have to do is choose the path with the highest weight. It is well known that this can be done in polynomial time (for an acyclic directed graph). This completes the proof of the main theorem:

Theorem 8.2 *The optimization problem can be solved in polynomial time.* □

The picture below shows the graph for the example, given earlier in this section.



The path with highest weight is $X_1^{(1)} \to X_2^{(1)} \to X_{8,9}^{(2)} \to X_{4,5,6}^{(3)} \to X_7^{(1)}$.

Chapter 9

Further Investigations

The most obvious question is whether the horizontal decomposition theory, presented in this thesis, can be extended to cover other constraints besides fd's. Several "candidate-classes" of constraints exist:

- *multivalued dependencies (mvd's)* and *join dependencies (jd's).*
  For mvd's one can easily define a horizontal decomposition, based on X-complete sets:
  The *horizontal decomposition of* r*, according to* $X \not\rightarrow\!\!\!\!\!\rightarrow Y$ is the ordered pair $(r_1 = \sigma_{X \rightarrow\!\!\!\!\!\rightarrow Y}(r), r_2 = r - r_1)$, where $\sigma_{X \rightarrow\!\!\!\!\!\rightarrow Y}(r)$ is the largest X-complete set of tuples for which $X \rightarrow\!\!\!\!\!\rightarrow Y$ holds.
  This definition then leads to a new constraint: the *amultivalued dependency (amd)* $X \not\rightarrow\!\!\!\!\!\rightarrow Y$, with the obvious meaning. One can also easily construct a strong Armstrong relation for mvd's (cfr. Theorem 2.1), and show that conflict can be easily detected (cfr. Theorem 2.3). Also, the amd's have no influence on the implication problem for mvd's (cfr. Theorem 2.2), as long as there is no conflict. The implication problem for amd's becomes more difficult, and we do not yet know if Theorem 3.1 can be converted to mvd's, i. e. we do not know whether $\mathcal{M} \cup \mathcal{A} \models X \not\rightarrow\!\!\!\!\!\rightarrow Y$ iff $\mathcal{M} \cup \mathcal{A} \cup \{X \rightarrow\!\!\!\!\!\rightarrow Y\}$ is in conflict.
  An additional problem with mvd's is that the inheritance problem has an unpleasant solution: all mvd's $T \rightarrow\!\!\!\!\!\rightarrow U$ with $T \neq X$ (or $T \not\rightarrow X$ if we also consider fd's) are lost by decomposing R according to $X \not\rightarrow\!\!\!\!\!\rightarrow Y$. Maybe the real reason for this failure of generalizing the horizontal decomposition to mvd's is that an mvd is in fact a special case of a jd: $X \rightarrow\!\!\!\!\!\rightarrow Y$ is equivalent to the jd $XY \bowtie X(\Omega - Y)$. For a jd

$X_1 \bowtie X_2 \bowtie \ldots \bowtie X_n$ it is not at all obvious how to define the notion of "exceptions".

In Chapter 1 we already indicated that mvd's and jd's are constraints on the structure of the database rather than on the data. Therefore the presence of exceptions in the real world is an indication for a bad database design.

- *partition dependencies (pd's).* These constraints have been introduced recently [9] as a new way of generalizing the concept of fd's. A set X of attributes defines a partition of an instance into its X-values (X-unique X-complete sets of tuples). On these partitions we have the "natural" operators $\bullet$ and $+$. The fd $X \to Y$ is equivalent to $X = X \bullet Y$. A "general" pd is $e = e'$ where e and e' are expressions (with sets of attributes, $\bullet$'s and $+$'s). The notion of an exception cannot be easily generalized to "general" pd's, but maybe some subclass (like $X = Y \bullet Z$) turns out to be useful. No research has been done on exceptions to pd's yet.
- *inclusion dependencies (ind's).* An exception to an ind $X \subseteq Y$ can be easily defined as the set of tuples with an X-value which is no Y-value. The horizontal decomposition then generates a "generalized" ind $R_1[X] \subseteq R_1[Y] \cup R_2[Y]$, and an *exclusion dependency (exd)* [5] $R_2[X] \mid\mid R_1[Y] \cup R_2[Y]$ (meaning that $R_2[X] \cap (R_1[Y] \cup R_2[Y]) = \emptyset$). It seems that most results on ind's and exd's can be generalized to dependencies between unions of relations, but this requires further study. Also, the decomposition for ind's makes the fd's more complicated: $X \to Y$ in R becomes $R_1 \cup R_2 : X \to Y$. We could also generalize ad's to $R_1 \cup R_2 : X \not\twoheadrightarrow Y$, which would prevent ad's from being lost by the horizontal decomposition. This matter still requires further study, but some negative results which are known about fd's and ind's already indicate that one cannot expect a "nice" theory, with only decidability results and polynomial algorithms.

Besides the theoretical generalization of the horizontal decomposition theory to other constraints besides fd's, some practical work needs to be done concerning the implementation of the horizontal decomposition in relational database machines. The theoretical advantage of the horizontal decomposition is somewhat reduced by the more complex update algo-

rithms and the increased number of relations that have to be accessed. The overall result can only be measured with some “real world” databases.

Bibliography

[1] Armstrong W., Dependency structures of database relationships. *IFIP 74 Conf.*, North Holland, pp. 580–583, 1974.

[2] Beeri C., P. A. Bernstein, Computational Problems related to the Design of Normal Form Relation Schemes *ACM TODS* **4:1**, pp. 30–59, 1979.

[3] Bernstein P. A., Normalization and Functional Dependencies in the Relational Database Model. *CSRG-60*, 1975.

[4] Casanova M. A., R. Fagin, C. H. Papadimitriou, Inclusion Dependencies and Their Interaction with Functional Dependencies. *Journal of Computer and System Sciences* **28:1**

[5] Casanova M. A., V. M. P. Vidal, Towards a Sound View Integration Methodology. *Proc. of the* 2nd *Symposium on Principles of Database Systems, ACM*, pp. 36–47, 1983.

[6] Codd E. F., A Relational Model of Data for Large Shared Data Banks. *CACM* **13:6**, pp. 377–387, June 1970.

[7] Codd E. F., Further Normalizations of the Database Relational Model, in *Data Base Systems* Courant Inst. Computer Science Symp. 6, Englewood Cliffs, N. J., Prentice Hall, (R. Rustin, ed.), pp. 33–64, 1972.

[8] Codd E. F., Recent Investigations in Relational Database Systems. *IFIP 74 Conf.*, North Holland, pp. 1017–1021, 1974.

[9] Cosmadakis S. S., P. C. Kanellakis, N. Spyratos, Partition Semantics for Relations. *Proc. of the* 4th *Symposium on Principles of Database Systems, ACM*, pp. 261–275, 1985.

[10] De Bra P., J. Paredaens, The Membership and the Inheritance of Functional and Afunctional Dependencies. *Proc. of the Colloquium on Algebra, Combinatorics and Logic in Computer Science, Gyor, Hungary*, 1983.

[11] De Bra P., J. Paredaens, Horizontal Decompositions for Handling Exceptions to Functional Dependencies. in *Advances in Database Theory*, Vol. **II**, pp. 123–144, 1983.

[12] De Bra P., J. Paredaens, An Algorithm for Horizontal Decompositions. *Information Processing Letters* **17**, pp. 91–95, North-Holland, 1983.

[13] De Bra P., J. Paredaens, Conditional Dependencies for Horizontal Decompositions. in *Lecture Notes in Computer Science*, Vol. **154**, pp. 67–82, (10-th *ICALP*), Springer-Verlag, 1983.

[14] De Bra P., Imposed-Functional Dependencies Inducing Horizontal Decompositions. in *Lecture Notes in Computer Science*, Vol. **194**, pp. 158–170, (12-th *ICALP*), Springer-Verlag, 1985.

[15] De Bra P., Functional Dependency Implications, Inducing Horizontal Decompositions. *UIA technical report* 85-30, 1985.

[16] De Bra P., Horizontal Decomposition Based on Functional-Dependency-Set-Implications. *Proc. of the first ICDT*, Rome, 1986.

[17] De Bra P., The Update Problem for Horizontally Decomposed Databases., *UIA technical report* 84-27, 1984.

[18] Fagin R., Armstrong Databases. *IBM RJ 3440*, 1982.

[19] Fagin R., Multivalued Dependencies and a New Normal Form for Relational Databases. *ACM TODS* **2:3**, pp. 262–278, September 1977.

[20] Fagin R., A. O. Mendelzon, J. D. Ullman, A Simplified Universal Relation Assumption and its Properties. *ACM TODS* **7:3**, pp. 343–360, September 1982.

[21] Fagin R., M. Y. Vardi, Armstrong Databases for Functional and Inclusion Depencencies. *IBM RJ 3500*, 1982.

[22] Maier D., The Theory of Relational Databases. Computer Science Press, Rockville, MD, 1983.

[23] Paredaens J., P. De Bra, On Horizontal Decompositions. *XP2-Congress*, State Univ. of Pennsylvania, 1981.

[24] Paredaens J., P. De Bra, M. Gyssens, The Structure of the Relational Database Model. to appear in *EATCS Monographs on Theoretical Computer Science*, Springer-Verlag, 1987.

[25] Rissanen J., Independent Components of Relations. *ACM TODS* **2:4**, pp. 317–325, December 1977.

[26] Ullman J., Principles of Database Systems, Computer Science Press, Rockville, MD, 1980.