Using Clausal Deductive Databases for Defining Semantics in Disjunctive Deductive Databases

Dietmar Seipel

University of Würzburg, Department of Computer Science
Am Hubland, D – 97074 Würzburg, Germany
Tel. +49 931 888 5026, Fax. +49 931 888 4600
E-mail: seipel@informatik.uni-wuerzburg.de

This paper investigates the novel concept of *clausal deductive databases* (cd–databases), which are special normal deductive databases – i.e. deductive databases which may contain default negation in rule bodies – over a meta–language $\mathcal{L}^{\mathsf{cd}}$ with a fixed set of predicate symbols, namely dis, con, and some *built–in* predicate symbols. The arguments of the literals in $\mathcal{L}^{\mathsf{cd}}$ are given by disjunctive and conjunctive clauses of a basic first–order language $\mathcal{L}$ (which are considered as terms in $\mathcal{L}^{\mathsf{cd}}$).

Disjunctive deductive databases (dd–databases) generalize normal deductive databases by allowing for disjunctions of atoms or literals in rule heads – next to default negation in rule bodies. We will present a *transformation* which maps a dd–database $\mathcal{D}$ into a cd–database $\mathcal{D}^{\mathsf{cd}}$ that talks about the clauses of $\mathcal{D}$. CD–databases provide a *flexible framework* for declaratively specifying the semantics of dd–databases: we can fix a standard *control strategy*, e.g. stable model or well–founded semantics, and vary the *logical description* $\mathcal{D}^{\mathsf{cd}}$ for specifying different semantics. The *transformed database* $\mathcal{D}^{\mathsf{cd}}$ usually consists of a part $\mathcal{D}^{\otimes}$ which naturally expresses the rules of $\mathcal{D}$, and two generic parts which are independent of $\mathcal{D}$: $\mathcal{D}^{logic}$ specifies *logical inference rules* like resolution and subsumption, and $\mathcal{D}^{cwa}$ specifies non–monotonic inference rules like *closed–world–assumptions*.

Another program transformation, which uses the idea of bringing sets of clauses to the argument level will be given for *hypothetical logic programs*: a hypothetical logic program $\mathcal{D}$ is mapped to a classical normal logic program $\mathcal{D}^{\odot}$, such that well–known semantics and inference methods for normal logic programs can be used for (disjunctive) hypothetical reasoning as well.

Keywords: disjunctive deductive databases, clausal logic programming, program transformations, non–monotonic reasoning, stable models, closed–world–assumptions, hypothetical reasoning

1. Introduction

In the last twenty years the concept of relational databases has been extended in various ways, e.g. to deductive databases, object–oriented databases, and active databases [19,28]. *Deductive databases* [2] provide a very powerful formalism for representing knowledge in the form of facts – which correspond to the tuples in a relational database – and different forms of inference rules. The *semantics* of normal deductive databases with rules that may contain default negation in rule bodies, is well–understood, and it is defined using special concepts of *non–monotonic reasoning*. The most prominent approaches are the *stable model semantics* [8] and the *well–founded semantics* [27]. *Disjunctive deductive databases* [11] allow for disjunctions – rather than just single atoms or literals – in rule heads, next to default negation in rule bodies; thus it becomes possible to specify knowledge with uncertain conclusions. For disjunctive deductive databases, the situation is much more complicated; there exist various generalizations of the stable model semantics [18,5] and the well–founded semantics [11,1], including some semantics dealing with three or more truth values[1].

In this paper we will investigate the novel concept of *clausal deductive databases*, which are special normal deductive databases over a meta–language $\mathcal{L}^{\mathsf{cd}}$ with a small set of predicate symbols which are talking about the clauses of a basic first–order language $\mathcal{L}$. E.g., the following inference rule is an allowed rule:

$$r = dis(X_d) \leftarrow complement(X_d, X_c) \wedge not\, con(X_c).$$

The arguments of the so–called cd–atoms $dis(X_d)$ and $con(X_c)$ can be arbitrary disjunctive or conjunctive formulas $X_d, X_c \in \mathcal{L}$, respectively. In addition, there are so–called bi–atoms, e.g. $complement(X_d, X_c)$, $subsumes(X_d, X_d')$ or $resolvent(X_1, X_2, X_d)$, with special *built–in* predicate symbols with a fixed, natural interpretation known from predicate logic. The rule r above represents a *closed–world–assumption* for inferring knowledge (the disjunction X_d) in the absence of the complementary knowledge (the conjunction X_c)[2].

CD–databases provide a flexible *framework for specifying semantics* of dd–databases in a *declarative* way. It is based on the idea of transforming a dd–database $\mathcal{D}$ into a cd–database $\mathcal{D}^{\mathsf{cd}}$, the cd–*transformation* of $\mathcal{D}$, and then defining the (disjunctive) semantics of $\mathcal{D}$ based on a well–understood (normal) semantics

[1] *true, false, undefined, overdefined,* etc.
[2] possible instantiations could be $X_d = \neg p(a) \vee \neg q(b)$ and $X_c = p(a) \wedge q(b)$

for $\mathcal{D}^{\mathsf{cd}}$. We will show that the (disjunctive) stable model semantics of $\mathcal{D}$ can be characterized by simply using the (normal) stable model semantics of $\mathcal{D}^{\mathsf{cd}}$. In a more refined approach we will use a special *fixpoint condition* based on $\mathcal{D}^{\mathsf{cd}}$ for defining a new semantics, called *stable state semantics*, which is able to give plausible interpretations for dd–databases for which standard semantics (e.g. stable or partial stable models) are *inconsistent*, i.e. do not provide any interpretations.

For some extensions of deductive databases without default negation we will show that it is possible to use the principle of reasoning known from the definite consequence operator $\mathcal{T}_{\mathcal{D}}$, as given by Lloyd [10]. We will define a *generic consequence operator* $\mathcal{T}_{\mathcal{D}}^{\odot,\geq} : 2^{\mathcal{X}} \to 2^{\mathcal{X}}$ on arbitrary power sets $2^{\mathcal{X}}$ that is parameterized with a natural transformation $\odot$, which maps $\mathcal{D}$ to a definite deductive database $\mathcal{D}^{\odot}$, and with a subsumption relation $\geq$. We will prove that $\mathcal{T}_{\mathcal{D}}^{\odot,\geq}$ is *monotonic* and *continuous* on the complete lattice $\langle 2^{\mathcal{X}}, \subseteq \rangle$. Then the disjunctive hyperresolution consequence operator $\mathcal{T}_{\mathcal{D}}^{s} : 2^{D_{HB_{\mathcal{D}}}} \to 2^{D_{HB_{\mathcal{D}}}}$ of Minker and Rajasekar [14] for dd–databases $\mathcal{D}$ without default negation – whose disjunctive Herbrand base is denoted by $D_{HB_{\mathcal{D}}}$ – is formulated as a definite consequence operator $\mathcal{T}_{\mathcal{D}}^{\odot,\geq}$ for $\mathcal{X} = \{ \, dis(\alpha) \mid \alpha \in D_{HB_{\mathcal{D}}} \, \}$.

The transformation $\mathcal{D}^{\odot}$ can be extended for the case of dd–databases with default negation by adding a definite representation $\mathcal{D}^{logic}$ of the *logical axioms* for binary resolution and subsumption and by adding a set $\mathcal{D}^{cwa}$ of *closed–world–assumptions*, cf. Reiter [20]. We can show that the *stable models* of $\mathcal{D}$ correspond to the stable models of the normal deductive database $\mathcal{D}^{\mathsf{cd}} = \mathcal{D}^{\odot} \cup \mathcal{D}^{logic} \cup \mathcal{D}^{cwa}$ for a suitable set $\mathcal{D}^{cwa}$. Moreover, we will define a new semantics for dd–databases which we will call *stable state semantics*; it is based on Herbrand states rather than Herbrand interpretations. This new semantics can be characterized based on the stable models of another transformed database $\mathcal{D}^{\mathsf{cd,tu}}$.

The advantage of transforming a disjunctive deductive database $\mathcal{D}$ into a normal deductive database $\mathcal{D}^{\mathsf{cd}}$ is that it becomes possible to use *standard techniques* for defining a semantics and *standard tools* for evaluating a database under this semantics [4,16,21,25,29]. In a first, prototypical implementation we have used the system Smodels developed by Niemelä [16], which can compute well–founded and stable model semantics for normal deductive databases.

Also other semantics of disjunctive deductive databases have been characterized or computed based on *transformations*. For some program transformations Figure 1 shows how they map programs from one class of logic programs to

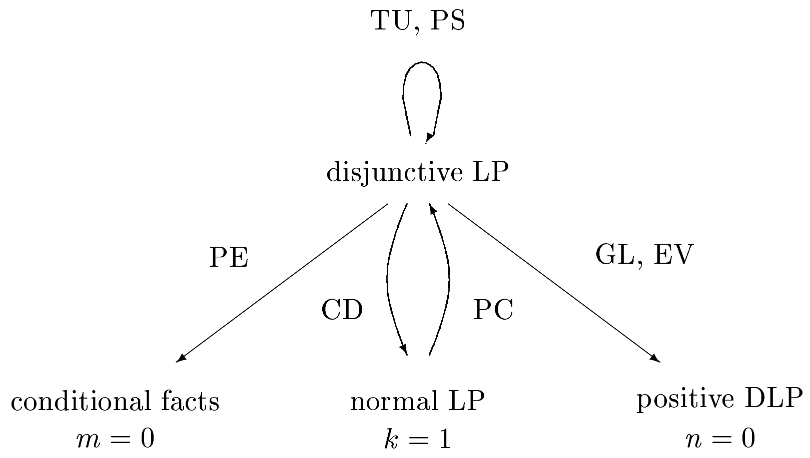

Figure 1. Program Transformations

another class.[3] E.g., Brass and Dix [1] have used *partial evaluation* (PE) and *program simplification* (PS) for defining their *disjunctive well–founded semantics*; Seipel et al. [23] have used the tu*–transformation* (TU) for characterizing *partial stable models*; Fernández et al. [6] and Seipel [24] have used the *evidential transformation* (EV) for characterizing *total stable models*, and *evidential stable models*, respectively. Other program transformations shown in Figure 1 are the *Gelfond–Lifschitz transformation* (GL) and *program completion* (PC).

Another *clausal* program transformation, which also uses the idea of bringing sets of clauses to the object level ("reifying formulas"), will be given for *hypothetical logic programs*, cf. [13,3,17]: a hypothetical logic program $\mathcal{D}$ is mapped to a classical normal logic program $\mathcal{D}^{\odot}$, such that well–known semantics and inference methods for normal logic programs can be used for hypothetical reasoning as well. Moreover, using our techniques we can even extend hypothetical logic programs by allowing for disjunctions in rule heads. The rules of $\mathcal{D}^{\odot}$ contain hypothetical atoms of the form $A \Leftarrow \mathcal{D}'$, where the embedded implication sign "$\Leftarrow$" can be perceived as the predicate symbol, and the arguments are an atom A and a hypothetical logic program $\mathcal{D}'$, respectively.[4] Thus, $\mathcal{D}^{\odot}$ is another form of a clausal deductive database, since sets of clauses – i.e., hypothetical logic programs – occur on the argument level.

The rest of the paper is organized as follows: In Section 2 we present the

[3] The different classes of logic programs will be explained in detail in Section 2.1.
[4] That means that hypothetical logic programs are defined inductively.

syntax of dd–databases and the stable model semantics. In Section 3 we define the syntax of cd–databases, and we also define clausal Herbrand interpretations for dd–databases. In Section 4 we develop the generic $\mathcal{T}_{\mathcal{D}}$–operator, and we show how the disjunctive consequence operator $\mathcal{T}^s_{\mathcal{D}}$ can be formulated. Section 5 deals with a characterization of the stable models using clausal logic programming, and defines a generalization, called *stable states*. We give a *characterization of the stable states* of a dd–database in terms of the stable models of an *annotated* clausal deductive database $\mathcal{D}^{\mathsf{cd,tu}}$ by modifying the concept of the $\mathtt{tu}$–*transformation* known from [23]. In Section 6 another program transformation is given that maps *hypothetical logic programs* to classical normal logic programs.

2. Basic Definitions and Notations

2.1. Syntax of Disjunctive Deductive Databases

Given a first order language $\mathcal{L}$, which may contain function symbols. A *disjunctive deductive database (dd–database)* $\mathcal{D}$ consists of logical inference rules of the form

$$r = A_1 \vee \ldots \vee A_k \leftarrow B_1 \wedge \ldots \wedge B_m \wedge \mathit{not}\, C_1 \wedge \ldots \wedge \mathit{not}\, C_n, \qquad (1)$$

where all A_i, B_i, and C_i are atoms in the language $\mathcal{L}$, $k, m, n \in I\!N_0$, and *not* is the negation–by–default operator.[5] The set of all *ground instances* of the rules in $\mathcal{D}$ is denoted by $\mathit{gnd}\,(\mathcal{D})$. A rule is called a *fact* if $m = n = 0$, it is called a *conditional fact* if $m = 0$, and it is called an *integrity constraint* or a *denial rule* if $k = 0$. A rule (or database) is called *positive–disjunctive* if it does not contain default negation (i.e. $n = 0$). It is called *normal* if $k = 1$, and *definite* if $k = 1$ and $n = 0$. A rule r of the form (1) above is denoted for short as

$$r = \alpha \leftarrow \beta \wedge \mathit{not}\cdot \gamma,$$

where $\alpha = A_1 \vee \ldots \vee A_k$, $\beta = B_1 \wedge \ldots \wedge B_m$, and $\gamma = C_1 \vee \ldots \vee C_n$.[6] α is called the *head* of r, β is called the *positive body*, and $\mathit{not}\cdot \gamma$ is called the *negative body* of r. A fact can simply be denoted by its head $A_1 \vee \ldots \vee A_k$.

[5] By $I\!N_+$ we denote the set $\{\, 1, 2, 3, \ldots \}$ of positive natural numbers, whereas $I\!N_0$ denotes the set $\{\, 0, 1, 2, \ldots \}$ of all natural numbers. $\langle\, n, m \,\rangle$ denotes the interval $\{\, n, n+1, \ldots, m \,\}$.

[6] Note that γ is a disjunction, and, according to the De Morgan pattern, $\mathit{not}\cdot \gamma$ is taken to be a conjunction.

2.2. Examples of Disjunctive Deductive Databases

Disjunctive Transitive Closure

Firstly, the positive–disjunctive deductive database (cf. [22])

$$\begin{aligned}\mathcal{D} \;=\; \{\; & path(X,Y) \leftarrow arc(X,Z) \wedge path(Z,Y),\\ & path(X,Y) \leftarrow arc(X,Y),\\ & arc(a,b) \vee arc(a,c),\;\; arc(b,d),\;\; arc(c,d) \;\}\end{aligned}$$

consists of the classical transitive closure rules and some disjunctive facts for the *arc*–relation, cf. Figure 2, where definite arcs are represented by solid arrows and indefinite arcs are represented by dotted arrows.

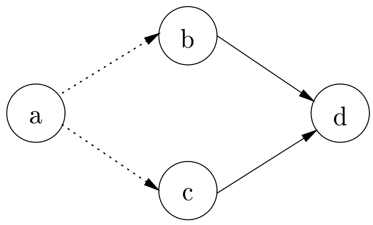

Figure 2. Graph with indefinite arcs

Obviously, there exists a path from node a to node d in the graph of Figure 2, either via b or via c. In Section 4, the disjunctive consequence operator $\mathcal{T}^s_{\mathcal{D}}$ will be defined, which is able to derive the conclusion $path(a,d)$ from $\mathcal{D}$.

Medical Diagnosis

Secondly, we will consider the following disjunctive deductive database for *medical diagnosis*, which deals with the pair of genes that determine a persons blood–group. Every person has got two genotypes – one is inherited from the mother and the father, respectively, cf. Figure 3 – which may be identical.

A statement $gt(P,T)$ means that person P has genotype T. The predicates *homozygot* and *heterozygot*, which are defined in the first three rules, tell if a person has got two identical or two different genotypes, respectively:

$$\begin{aligned}& heterozygot(P,T_1,T_2) \leftarrow gt(P,T_1) \wedge gt(P,T_2) \wedge T_1 \neq T_2.\\ & heterozygot(P) \leftarrow heterozygot(P,T_1,T_2).\\ & homozygot(P,T) \leftarrow gt(P,T) \wedge not\ heterozygot(P).\end{aligned}$$

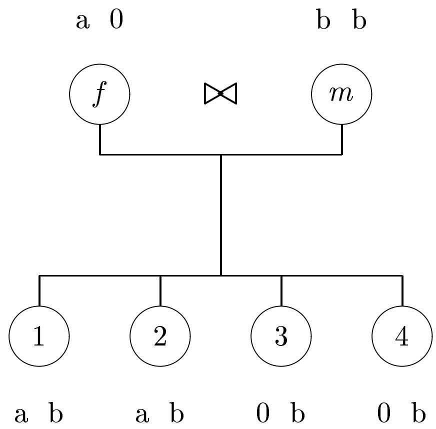

Figure 3. Inheritance of Blood–Groups

E.g., we would derive $heterozygot(f, a, 0)$ and $homozygot(m, b)$ from Figure 3. The rules four and five express that a person must have at least one of the genotypes "a", "b", or "o" and can have at most two different genotypes; rule five is an integrity constraint, which forbids that a single person has got three different genotypes: it has an empty rule head denoting a contradiction:

$$gt(P, a) \vee gt(P, b) \vee gt(P, o) \leftarrow person(P).$$
$$\leftarrow gt(P, T_1) \wedge gt(P, T_2) \wedge gt(P, T_3) \wedge T_1 \neq T_2 \wedge T_1 \neq T_3 \wedge T_2 \neq T_3.$$

The "diagnostic" rules six and seven relate the genotype of a child "C" to the genotypes of the parents "P":

$$gt(C, T_1) \vee gt(C, T_2) \leftarrow parent(C, P) \wedge heterozygot(P, T_1, T_2).$$
$$gt(C, T) \leftarrow parent(C, P) \wedge homozygot(P, T).$$

Finally, the *phenotype*, i.e. the person's actual blood–group, is the result of the two genotypes, cf. rules ten and eleven. Four different phenotypes are possible: "a", "b", "ab", and "o". This is specified by rules eight and nine.

$$pt(P, a) \vee pt(P, b) \vee pt(P, ab) \vee pt(P, o) \leftarrow person(P).$$
$$\leftarrow pt(P, T_1) \wedge pt(P, T_2) \wedge T_1 \neq T_2.$$
$$pt(P, a) \leftarrow heterozygot(P, a, o).$$
$$pt(P, a) \leftarrow homozygot(P, a).$$

Further rules could be given for deriving the phenotypes "*b*", "*ab*", and "*o*", and also for relating the genotype of a person to the genotypes of the parents.

The intuitive understanding of the medical dd–database, which was explained above, can be formalized using stable model semantics (which will be defined in Section 2.4). Simpler semantics like minimal or perfect model semantics cannot be applied here, since the database is non–stratified: *genotype* depends recursively on itself, and there is a negative dependency between *homozygot* and *heterozygot* involved in the recursion.

2.3. Herbrand States

The Herbrand base $\mathit{HB}_{\mathcal{D}}$ of a dd–database $\mathcal{D}$ contains all ground atoms over the language of $\mathcal{D}$, and $\mathit{HL}_{\mathcal{D}} = \mathit{HB}_{\mathcal{D}} \cup \{\, \neg A \mid A \in \mathit{HB}_{\mathcal{D}} \,\}$ contains all literals over $\mathit{HB}_{\mathcal{D}}$. The semantics of dd–databases will be based on a set of ground disjunctions, a so–called Herbrand state (cf. [11]), or a set of Herbrand interpretations.

1. The *disjunctive*, the *negative*, the *general*, and the *conjunctive Herbrand base* of $\mathcal{D}$ are given by $\mathit{DHB}_{\mathcal{D}}$, $\mathit{NHB}_{\mathcal{D}}$, $\mathit{GHB}_{\mathcal{D}}$, and $\mathit{CHB}_{\mathcal{D}}$, respectively:

$$\begin{aligned}
\mathit{DHB}_{\mathcal{D}} &= \{\, A_1 \vee \ldots \vee A_k \mid A_i \in \mathit{HB}_{\mathcal{D}}, \text{ for } i \in \langle 1,k \rangle,\ k \in \mathbb{N}_0 \,\}, \\
\mathit{NHB}_{\mathcal{D}} &= \{\, \neg A_1 \vee \ldots \vee \neg A_k \mid A_i \in \mathit{HB}_{\mathcal{D}}, \text{ for } i \in \langle 1,k \rangle,\ k \in \mathbb{N}_0 \,\}, \\
\mathit{GHB}_{\mathcal{D}} &= \{\, L_1 \vee \ldots \vee L_k \mid L_i \in \mathit{HL}_{\mathcal{D}}, \text{ for } i \in \langle 1,k \rangle,\ k \in \mathbb{N}_0 \,\}, \\
\mathit{CHB}_{\mathcal{D}} &= \{\, L_1 \wedge \ldots \wedge L_k \mid L_i \in \mathit{HL}_{\mathcal{D}}, \text{ for } i \in \langle 1,k \rangle,\ k \in \mathbb{N}_0 \,\}.
\end{aligned}$$

2. An *Herbrand state* is a set S of formulas. It is called *disjunctive*, *negative*, *general* and *conjunctive*, if $S \subseteq \mathit{DHB}_{\mathcal{D}}$, $S \subseteq \mathit{NHB}_{\mathcal{D}}$, $S \subseteq \mathit{GHB}_{\mathcal{D}}$, and $S \subseteq \mathit{CHB}_{\mathcal{D}}$. respectively.

We will sometimes also write $\mathit{HB}_{\mathcal{L}}$, $\mathit{HL}_{\mathcal{L}}$, $\mathit{DHB}_{\mathcal{L}}$, $\mathit{NHB}_{\mathcal{L}}$, $\mathit{GHB}_{\mathcal{L}}$, and $\mathit{CHB}_{\mathcal{L}}$, if we are interested in the language $\mathcal{L}$ rather than the database $\mathcal{D}$. Disjunctions and conjunctions will be treated like sets of literals. Thus, we will identify a disjunction $L_1 \vee \ldots \vee L_k \in \mathit{GHB}_{\mathcal{D}}$ with its permutations $L_{\pi(1)} \vee \ldots \vee L_{\pi(k)}$. The empty disjunction (conjunction) with $k = 0$ literals will be denoted by $\Box_d$ ($\Box_c$). Note that $\Box_d$ is an element of all of $\mathit{DHB}_{\mathcal{D}}$, $\mathit{NHB}_{\mathcal{D}}$, and $\mathit{GHB}_{\mathcal{D}}$, and that $\Box_c$ is an element of $\mathit{CHB}_{\mathcal{D}}$. Note also that singleton disjunctions or conjunctions with $k = 1$ literals are identified with literals; as a consequence we get $\mathit{GHB}_{\mathcal{D}} \cap \mathit{CHB}_{\mathcal{D}} = \mathit{HL}_{\mathcal{D}}$.

The common *subsumption* relation for ground disjunctions $\alpha = L_1 \vee \ldots \vee L_n \in \mathit{GHB}_{\mathcal{D}}$, $\alpha' = L'_1 \vee \ldots \vee L'_m \in \mathit{GHB}_{\mathcal{D}}$, is defined by $\alpha \trianglerighteq \alpha'$ iff $\{\, L_1, \ldots, L_n \,\} \subseteq$

$\{ L'_1, \ldots, L'_m \}$. The *canonical form* of a general Herbrand state $S \subseteq G_{HB_\mathcal{D}}$ is defined as the set $can_{\trianglerighteq}(S)$ of minimal elements of S w.r.t. subsumption, and the *expanded form* of S is defined as the set $exp_{\trianglerighteq}(S)$ of all $\alpha \in G_{HB_\mathcal{D}}$ that are subsumed by some $\alpha' \in S$:

$$can_{\trianglerighteq}(S) = \{ \alpha \in S \mid \not\exists \alpha' \in S : \alpha' \neq \alpha, \alpha' \trianglerighteq \alpha \},$$
$$exp_{\trianglerighteq}(S) = \{ \alpha \in G_{HB_\mathcal{D}} \mid \exists \alpha' \in S : \alpha' \trianglerighteq \alpha \}.$$

Obviously, it holds $can_{\trianglerighteq}(S) \subseteq S \subseteq exp_{\trianglerighteq}(S)$. E.g., for $HB_\mathcal{D} = \{ a, b, c \}$ and $S = \{ a, a \vee b, b \vee c \}$ we get $can_{\trianglerighteq}(S) = S \setminus \{ a \vee b \}$ and $exp_{\trianglerighteq}(S)$ contains – among others – the additional disjunctions $a \vee c$ and $a \vee b \vee \neg c$.

For a literal $L \in HL_\mathcal{D}$ we define the *formal complement* $\neg \cdot L = \neg A$, if $L = A \in HB_\mathcal{D}$ is an atom, and $\neg \cdot L = A$, if $L = \neg A \in HL_\mathcal{D}$ is a negated atom. For a general disjunction $\alpha = L_1 \vee \ldots \vee L_k \in G_{HB_\mathcal{D}}$ we define the conjunction

$$\neg \cdot \alpha = \neg \cdot (L_1 \vee \ldots \vee L_k) = \neg \cdot L_1 \wedge \ldots \wedge \neg \cdot L_k \in C_{HB_\mathcal{D}}$$

to be the *formal complement* of α according to the De Morgan pattern, and likewise we define $\neg \cdot \beta = \neg \cdot L_1 \vee \ldots \vee \neg \cdot L_k$ for $\beta = L_1 \wedge \ldots \wedge L_k \in C_{HB_\mathcal{D}}$.

2.4. Stable Model Semantics

An *Herbrand interpretation* of a dd–database $\mathcal{D}$ is given by a subset $I \subseteq HB_\mathcal{D}$. It defines a mapping I: $HB_\mathcal{D} \to \{ \mathsf{t}, \mathsf{f} \}$ assigning a truth value "t" (*true*) or "f" (*false*) to each $A \in HB_\mathcal{D}$: $I(A) = \mathsf{t} \Leftrightarrow A \in I$. [7] *I satisfies* of a ground rule $r = \alpha \leftarrow \beta \wedge not \cdot \gamma$ if $I(\beta) \wedge I(not \cdot \gamma) = \mathsf{t} \Rightarrow I(\alpha) = \mathsf{t}$, where $I(not \cdot \gamma) = \neg I(\gamma)$. I is an *Herbrand model* of $\mathcal{D}$ if I satisfies all ground instances $r \in gnd(\mathcal{D})$ of the rules of $\mathcal{D}$.

I is a *minimal model* of $\mathcal{D}$ if I is an Herbrand model of $\mathcal{D}$ and there is no other Herbrand model J of $\mathcal{D}$ such that $J \subsetneq I$. The set of all minimal models of $\mathcal{D}$ is denoted by $\mathcal{MM}_2(\mathcal{D})$. For positive–disjunctive deductive databases $\mathcal{D}$ the semantics is given by $\mathcal{MM}_2(\mathcal{D})$. E.g., the minimal models of the transitive closure database of Section 2.2 are given by

$$\mathcal{MM}_2(\mathcal{D}) = \{ I \cup \{ arc(a,b), path(a,b) \}, I \cup \{ arc(a,c), path(a,c) \} \},$$

[7] For $A_i \in HB_\mathcal{D}$, $i \in \langle 1, k \rangle$, and a connective $\oplus \in \{ \vee, \wedge \}$ we define $I(A_1 \oplus \ldots \oplus A_k) = I(A_1) \oplus \ldots \oplus I(A_k)$. For $k = 0$, the empty disjunction $\Box_d$ (i.e. $\oplus = \vee$) evaluates to $I(\Box_d) = \mathsf{f}$, whereas the empty conjunction $\Box_c$ (i.e. $\oplus = \wedge$) evaluates to $I(\Box_c) = \mathsf{t}$.

where $I \;=\; \{\, arc(b,d),\; arc(c,d),\; path(b,d),\; path(c,d),\; path(a,d) \,\}$.

The semantics of dd–databases in general is defined based on a more general class of Herbrand models, the *stable models*. These are defined in terms of the minimal models M of the *Gelfond–Lifschitz transformation* (GL–transformation, [8,18]) of $\mathcal{D}$, which is a special ground positive–disjunctive database $\mathcal{D}^M$ derived from those rules of $gnd\,(\mathcal{D})$ whose negative body is true in M (i.e. $C_i \notin M$, for all C_i in $not\cdot\gamma = not\, C_1 \wedge \ldots \wedge not\, C_n$).

Definition 1 (Stable Models, [8,18]).
Let M be a Herbrand interpretation of a disjunctive deductive database $\mathcal{D}$.

1. The GL–transformation $\mathcal{D}^M$ of $\mathcal{D}$ w.r.t. M is

$$\mathcal{D}^M = \{\; \alpha \leftarrow \beta \mid \text{there exists } \alpha \leftarrow \beta \wedge not\cdot\gamma \in gnd\,(\mathcal{D}), \\ \text{such that } M(not\cdot\gamma) = \mathtt{t} \;\}.$$

2. M is called *stable model* of $\mathcal{D}$ if $M \in \mathcal{MM}_2(\mathcal{D}^M)$. The set of all stable models of $\mathcal{D}$ is denoted by $\mathcal{STABLE}_2(\mathcal{D})$.

Example 2. Consider the dd–database

$$\mathcal{D}_{win} = \{\; win(X) \leftarrow move(X,Y) \wedge not\, win(Y) \;\} \;\cup\; S, \\ S = \{\; move(a,b) \vee move(a,c),\; move(b,d),\; move(c,d) \;\}.$$

The graph for the disjunctive *move*–relation is again given by Figure 2 from Section 2.2. We get a set with two stable models, which coincide on the subset $I = \{\, move(b,d),\; move(c,d),\; win(b),\; win(c) \,\}$ of atoms:

$$\mathcal{STABLE}_2(\mathcal{D}_{win}) = \{\, I \cup \{\, move(a,b) \,\},\; I \cup \{\, move(a,c) \,\} \,\}.$$

For $M = I \cup \{\, move(a,b) \,\}$ we get that $M(not\, win(y)) = \mathtt{t}$ iff $y \in \{a,d\}$, and thus we get $M \in \mathcal{MM}_2(\mathcal{D}^M_{win})$ for the GL–transformation $\mathcal{D}^M_{win}$:

$$\mathcal{D}^M_{win} = \{\; win(x) \leftarrow move(x,y) \mid x \in \{a,b,c,d\},\; y \in \{a,d\} \;\} \;\cup\; S.$$

All of the atoms in I and the disjunction $move(a,b) \vee move(a,c)$ are consequences of $\mathcal{D}_{win}$ under the stable model semantics.

3. Clausal Deductive Databases

In the context of default negation and disjunction one can generalize the notion of a (total) Herbrand interpretation in two ways.

Firstly, one can allow for a third truth value "u" (*undefined*), which leads to partial Herbrand interpretations $I\colon \mathit{HB}_{\mathcal{D}} \to \{ \mathsf{t}, \mathsf{f}, \mathsf{u} \}$. If a semantics is given by a set $\mathcal{I}$ of partial Herbrand interpretations, then a general Herbrand state $S \subseteq \mathit{GHB}_{\mathcal{D}}$ containing all disjunctive consequences $\alpha \in \mathit{GHB}_{\mathcal{D}}$ that hold in all $I \in \mathcal{I}$ can be assigned to $\mathcal{D}$. Partial Herbrand interpretations have been used by Przymusinski, cf. [18], who generalized the concept of stable model to *partial stable models*. There exists a characterization of the partial stable models of a disjunctive deductive database $\mathcal{D}$ in terms of the stable models of a naturally transformed annotated database $\mathcal{D}^{\mathsf{tu}}$, cf. [23].

Secondly, a semantics can also be given by a set $\mathcal{S}$ of so–called *clausal Herbrand interpretations*, which are special general Herbrand states, without refering to total or partial Herbrand interpretations at all. In this paper we will introduce the new concept of a *clausal deductive databases*, which are special normal deductive databases over a restricted set of predicate symbols. In Section 5, we will transform a dd–database $\mathcal{D}$ into a corresponding cd–database $\mathcal{D}^{\mathsf{cd}}$ and then compute the clausal Herbrand models of $\mathcal{D}$ by simply applying the standard stable model semantics to an annotated cd–database $\mathcal{D}^{\mathsf{cd,tu}}$ that is derived from $\mathcal{D}^{\mathsf{cd}}$.

3.1. Partial and Clausal Herbrand Interpretations

A *partial Herbrand interpretation* of a disjunctive deductive database $\mathcal{D}$ is given by a subset $I \subseteq \mathit{HL}_{\mathcal{D}}$, such that I does not contain both an atom $A \in \mathit{HB}_{\mathcal{D}}$ and its negation $\neg A$, i.e. $I \cap \neg I = \emptyset$. It defines a mapping $I\colon \mathit{HB}_{\mathcal{D}} \to \{ \mathsf{t}, \mathsf{f}, \mathsf{u} \}$ on ground atoms by

$$I(A) = \begin{cases} \mathsf{t}, \text{ iff } A \in I \\ \mathsf{f}, \text{ iff } \neg A \in I \\ \mathsf{u}, \text{ otherwise.} \end{cases}$$

Alternatively, I can be represented by a pair $\langle I_t, I_u \rangle$, such that $I_t = \{ A \in \mathit{HB}_{\mathcal{D}} \mid I(A) = \mathsf{t} \}$ and $I_u = \{ A \in \mathit{HB}_{\mathcal{D}} \mid I(A) = \mathsf{t} \vee I(A) = \mathsf{u} \}$. E.g., for the Herbrand base $\mathit{HB}_{\mathcal{D}} = \{ a, b, c, d \}$ and $I(a) = \mathsf{t}$, $I(b) = \mathsf{t}$, $I(c) = \mathsf{f}$, and $I(d) = \mathsf{u}$, we get $I_t = \{ a, b \}$ and $I_u = \{ a, b, d \}$. If for each atom $A \in \mathit{HB}_{\mathcal{D}}$, I contains either A or $\neg A$, then I defines a *total Herbrand interpretation* or Herbrand interpretation for short. In this case, it is sufficient to represent I by its set of true atoms (since there are no undefined atoms), which we did in Section 2.

The truth values of disjunctions $\alpha \in G\mathit{HB}_{\mathcal{D}}$ and conjunctions $\beta \in C\mathit{HB}_{\mathcal{D}}$ are defined based on the *Boolean operations* "$\vee$", "$\wedge$" and "$\neg$" in three–valued logic, which are given in Figure 4. The *truth ordering* on the three truth values is given by $\mathsf{f} \leq_{\mathsf{t}} \mathsf{u} \leq_{\mathsf{t}} \mathsf{t}$.

$\wedge$	t	f	u
t	t	f	u
f	f	f	f
u	u	f	u

$\vee$	t	f	u
t	t	t	t
f	t	f	u
u	t	u	u

$\neg$	
t	f
f	t
u	u

Figure 4. Boolean operations in three–valued logic

A *clausal Herbrand interpretation* is given by a general Herbrand state $S \subseteq G\mathit{HB}_{\mathcal{D}}$ that is closed under binary resolution and subsumption. Obviously, we can *abbreviate* a clausal Herbrand interpretation S simply by its set $can_{\trianglerighteq}(S)$ of minimal elements w.r.t. subsumption. If $can_{\trianglerighteq}(S) \subseteq H\mathit{L}_{\mathcal{D}}$, then we can consider S as a partial Herbrand interpretation $I = can_{\trianglerighteq}(S)$ of $\mathcal{D}$. Conversely, a partial Herbrand interpretation I can be considered as a special clausal Herbrand interpretation $exp_{\trianglerighteq}(I)$. Every set $\mathcal{S}$ of clausal (or partial) Herbrand interpretations defines a clausal Herbrand interpretation $\mathcal{S}^{\cap} = \cap_{S \in \mathcal{S}} S \subseteq G\mathit{HB}_{\mathcal{D}}$, where $\mathcal{S}^{\cap} = G\mathit{HB}_{\mathcal{D}}$, if $\mathcal{S} = \emptyset$

The *semantics* of a disjunctive deductive database $\mathcal{D}$ will be given by a set $\mathcal{S}$ of clausal Herbrand interpretations, and a formula α is a consequence under $\mathcal{S}$ if $\alpha \in \mathcal{S}^{\cap}$.

3.2. Syntax of Clausal Deductive Databases

Consider a first order language $\mathcal{L}$ and the set $Lit_{\mathcal{L}}$ of all literals that can be built in the language $\mathcal{L}$. Additionally, consider a set $\mathcal{V}$ of variable symbols for formulas. Disjunctive (conjunctive) *formulas* are defined inductively: every atom $A \in Lit_{\mathcal{L}}$ and every $X \in \mathcal{V}$ is both a disjunctive and a conjunctive formula, and, if $t_1, \ldots, t_n$ are disjunctive (conjunctive) formulas, then so is $t_1 \vee \ldots \vee t_n$ $(t_1 \wedge \ldots \wedge t_n)$.

A *cd–atom* is given by $dis(t_d)$ or $con(t_c)$, where t_d is a disjunctive formula and t_c is a conjunctive formula. If X_d, X'_d, X''_d are disjunctive formulas, and

X_c, X'_c, X''_c are conjunctive formulas, and $X \in Lit_{\mathcal{L}} \cup \mathcal{V}$, then the following are *built–in atoms* (bi–atoms):

$$\begin{aligned}
&positive(X_d),\ negative(X_d),\\
&subsumes(X_d, X'_d),\ resolvent(X_d, X'_d, X''_d),\\
&conjunction(X_c, X'_c, X''_c),\\
&literal(X),\ complement(X_d, X_c).
\end{aligned}$$

A *clausal deductive database* (*cd–database*) $\mathcal{C}$ consists of *cd–rules* of the form

$$r = A \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not\, C_1 \wedge \ldots \wedge not\, C_n, \tag{2}$$

where A is a cd–atom, and the B_i, $i \in \langle 1, m \rangle$, and C_i, $i \in \langle 1, n \rangle$, are cd–atoms or bi–atoms, and *not* is the negation–by–default operator. A cd–rule is called a cd–fact if $m = n = 0$. For example, the following is a cd–rule:

$$\begin{aligned}
&dis(X'_d \vee \neg p(Y, a)) \leftarrow\\
&\qquad dis(X_d \vee q(c)) \wedge subsumes(X_d, X'_d) \wedge\\
&\qquad complement(X_d, X_c \wedge \neg q(d)) \wedge not\, con(X_c \wedge \neg q(d)) \wedge\\
&\qquad not\, literal(X_c).
\end{aligned}$$

Built–in predicate symbols are not allowed in rule heads, although built–in predicates could of course be defined by logical rules as well. For simplicity reasons, however, we assume that their definition is given by a standard *built–in Herbrand interpretation*

$$\begin{aligned}
I^{bi}_{\mathcal{L}} = &\{\, positive(\alpha) \mid \alpha \in D\mathit{HB}_{\mathcal{L}} \,\} \cup \{\, negative(\alpha) \mid \alpha \in N\mathit{HB}_{\mathcal{L}} \,\} \cup\\
&\{\, literal(\gamma) \mid \gamma \in H\mathit{B}_{\mathcal{L}} \,\} \cup \{\, complement(\alpha, \neg \cdot \alpha) \mid \alpha \in G\mathit{HB}_{\mathcal{L}} \,\} \cup\\
&\{\, subsumes(\alpha_1, \alpha_2) \mid \alpha_1, \alpha_2 \in G\mathit{HB}_{\mathcal{L}},\ \alpha_1 \trianglerighteq \alpha_2 \,\} \cup\\
&\{\, resolvent(A \vee \alpha_1, \neg A \vee \alpha_2, \alpha_1 \vee \alpha_2) \mid A \in H\mathit{B}_{\mathcal{L}},\ \alpha_1, \alpha_2 \in G\mathit{HB}_{\mathcal{L}} \,\} \cup\\
&\{\, conjunction(\beta_1, \beta_2, \beta_1 \wedge \beta_2) \mid \beta_1, \beta_2 \in C\mathit{HB}_{\mathcal{L}} \,\}.
\end{aligned}$$

The built–in predicate symbols *conjunction* and *resolvent* are not really necessary in cd–databases, since they can be replaced using the junctors "$\wedge$" and "$\vee$", respectively. However, it will be very convenient to have them.

The *ground instance* $gnd\,(\mathcal{C})$ of a cd–database $\mathcal{C}$ is obtained replacing all variable symbols for formulas by ground formulas from $G\mathit{HB}_{\mathcal{L}} \cup C\mathit{HB}_{\mathcal{L}}$ and by also grounding the literals from $Lit_{\mathcal{L}}$ in $\mathcal{C}$, such that the resulting rules are proper cd–

rules.[8] For a general Herbrand state $S \subseteq G_{HB_\mathcal{L}}$ we define $dis(S) = \{dis(\alpha) | \alpha \in S\}$ and $con(S) = \{\ con(L_1 \wedge \ldots \wedge L_k) \mid k \in \mathbb{N}_0,\ L_i \in S \cap H_{L_\mathcal{L}},$ for all $i \in \langle 1, k \rangle\ \}$. Note that $con(\Box_c) \in con(S)$. Finally, $S^{\mathsf{cd}} = dis(S) \cup con(S) \cup I^{bi}_\mathcal{L}$ denotes the total Herbrand interpretation of $\mathcal{C}$ that is derived from S.

4. Clausal Deductive Databases without Default Negation

In this section we will show that the disjunctive hyperresolution consequence operator $\mathcal{T}^s_\mathcal{D}$, which has been defined by Minker and Rajasekar [14] for positive–disjunctive deductive databases $\mathcal{D}$, can be formulated using a generic definite hyperresolution consequence operator $\mathcal{T}^{\odot,\geq}_\mathcal{D}$, which is parameterized with a suitable subsumption relation. Thus reasoning with disjunctive hyperresolution can be reduced to reasoning with definite hyperresolution in cd–databases.

4.1. The Disjunctive Consequence Operator $\mathcal{T}^s_\mathcal{D}$

We will consider the following definition[9] of the disjunctive hyperresolution consequence operator $\mathcal{T}^s_\mathcal{D} :\ 2^{D_{HB_\mathcal{D}}} \to 2^{D_{HB_\mathcal{D}}}$:

$$\begin{aligned}\mathcal{T}^s_\mathcal{D}(S) = \{\ \alpha \vee \alpha_1 \vee \ldots \vee \alpha_m \in D_{HB_\mathcal{D}} \mid \\ &\text{there exists a rule } \alpha \leftarrow B_1 \wedge \ldots \wedge B_m \in gnd\,(\mathcal{D}), \\ &\text{such that } \{\ B_1 \vee \alpha_1, \ldots, B_m \vee \alpha_m\ \} \subseteq S\ \}.\end{aligned}$$

$\mathcal{T}^s_\mathcal{D}$ generalizes the well–known consequence operator $\mathcal{T}_\mathcal{D}$ of van Emden and Kowalski, which had been defined for definite logic programs (cf. [10]). Minker and Rajasekar [14] have shown that $\mathcal{T}^s_\mathcal{D}$ is *monotonic* and *continuous* on the complete lattice $\langle 2^{D_{HB_\mathcal{D}}}, \subseteq \rangle$. This ensures that the fixpoint iteration terminates with the least fixpoint $lfp\,(\mathcal{T}^s_\mathcal{D})$ within at most ω steps: the expanded form $exp_{\trianglerighteq}(lfp\,(\mathcal{T}^s_\mathcal{D}))$ coincides with the set of positive ground disjunctions that are logical consequences of $\mathcal{D}$. E.g., for the transtive closure database of Section 2.2 the *fixpoint iteration* produces the following ordinal powers $\mathcal{T}^s_\mathcal{D} \uparrow 0 = \emptyset$ and $\mathcal{T}^s_\mathcal{D} \uparrow n+1 = \mathcal{T}^s_\mathcal{D}(\mathcal{T}^s_\mathcal{D} \uparrow n)$, for $n \geq 0$ (cf. [22]):

$$\mathcal{T}^s_\mathcal{D} \uparrow 1\ =\ \{\ arc(a,b) \vee arc(a,c),\ arc(b,d),\ arc(c,d)\ \},$$

[8] This implies that the obvious typing condition for variable symbols for formulas must be obeyed.

[9] It is equivalent to the definition of [14]. Since disjunctions are considered as sets of atoms, it is not necessary to work with smallest factors.

$$\begin{aligned}
\mathcal{T}^s_{\mathcal{D}} \uparrow 2 \;&=\; \{\, arc(a,b) \vee path(a,c),\; path(a,b) \vee arc(a,c), \\
&\qquad path(b,d),\; path(c,d) \,\} \;\cup\; \mathcal{T}^s_{\mathcal{D}} \uparrow 1, \\
\mathcal{T}^s_{\mathcal{D}} \uparrow 3 \;&=\; \{\, path(a,b) \vee path(a,c), \\
&\qquad path(a,d) \vee arc(a,c),\; path(a,d) \vee arc(a,b), \\
&\qquad path(a,d) \vee path(a,c),\; path(a,d) \vee path(a,b) \,\} \;\cup\; \mathcal{T}^s_{\mathcal{D}} \uparrow 2, \\
\mathcal{T}^s_{\mathcal{D}} \uparrow 4 \;&=\; \{\, path(a,d) \,\} \;\cup\; \mathcal{T}^s_{\mathcal{D}} \uparrow 3, \\
\mathcal{T}^s_{\mathcal{D}} \uparrow n \;&=\; \mathcal{T}^s_{\mathcal{D}} \uparrow 4, \quad \text{for all} \;\; n \geq 4.
\end{aligned}$$

Thus, $lfp(\mathcal{T}^s_{\mathcal{D}}) = \mathcal{T}^s_{\mathcal{D}} \uparrow \omega = \mathcal{T}^s_{\mathcal{D}} \uparrow 4$. This example shows that $\mathcal{T}^s_{\mathcal{D}}$ can also derive definite consequences (e.g. $path(a,d)$) from disjunctive facts: in step 4, $path(a,d)$ is derived from the ground rule $\alpha \leftarrow B_1 \wedge B_2 \in gnd\,(\mathcal{D})$ and the facts $B_1 \vee \alpha_1$, $B_2 \vee \alpha_2$, where $B_1 = arc(a,c)$, $B_2 = path(c,d)$, $\alpha = \alpha_1 = path(a,d)$, and $\alpha_2 = \Box_d$. In [22], $\mathcal{T}^s_{\mathcal{D}}$ has been related to the model–generation operator of Fernández and Minker [7], and it was shown that the latter is monotonic but not continuous.

4.2. The Generic $\mathcal{T}_{\mathcal{D}}$–Operator

Given a deductive database $\mathcal{D}$ without default negation (e.g. a positive–disjunctive, probabilistic or hypothetical deductive database, cf. [22,28,3]): We consider a partial ordering $\mathcal{O} = \langle\, \mathcal{X}, \geq \rangle$ and a transformation "$\odot$" that maps $\mathcal{D}$ to a definite deductive database $\mathcal{D}^{\odot}$, such that $gnd\,(\mathcal{D}^{\odot})$ consists of definite ground rules $\alpha \leftarrow \beta_1 \wedge \ldots \wedge \beta_m$, where $\{\, \alpha, \beta_1, \ldots, \beta_m \,\} \subseteq \mathcal{X}$; examples for $\mathcal{X}$ and $\geq$ will be given in the following Section 4.2. Then a generic $\mathcal{T}_{\mathcal{D}}$–operator

$$\mathcal{T}^{\odot,\geq}_{\mathcal{D}} :\; 2^{\mathcal{X}} \to 2^{\mathcal{X}}$$

is induced by $\langle\, \odot, \geq \rangle$ on the complete lattice $\langle\, 2^{\mathcal{X}}, \subseteq \rangle$; it refers to the expanded form $exp_{\geq}(X) = \{\alpha \in \mathcal{X} \,|\, \exists \alpha' \in X : \alpha' \geq \alpha\} \subseteq \mathcal{X}$, of sets $X \subseteq \mathcal{X}$, cf. Section 2.1:

$$\begin{aligned}
\mathcal{T}^{\odot,\geq}_{\mathcal{D}}(X) = \{\; \alpha \in \mathcal{X} \mid \text{there exists } \alpha \leftarrow \beta_1 \wedge \ldots \wedge \beta_m \in gnd\,(\mathcal{D}^{\odot}), \\
\text{such that } \{\beta_1, \ldots, \beta_m\} \subseteq exp_{\geq}(X) \;\}.
\end{aligned}$$

Theorem 3 (Properties of the Generic $\mathcal{T}_{\mathcal{D}}$–Operator).
The operator $\mathcal{T}^{\odot,\geq}_{\mathcal{D}} :\; 2^{\mathcal{X}} \to 2^{\mathcal{X}}$ is *monotonic* and *continuous* on $\langle\, 2^{\mathcal{X}}, \subseteq \rangle$.

The proof is given in Appendix A. The properties of fixpoint iteration with subsumption have also been analysed in [9] with special interest in the process of *differential* fixpoint iteration.

4.3. Transformation of Positive DD–Databases to Positive CD–Databases

For positive–disjunctive deductive databases $\mathcal{D}$ we consider $\mathcal{X} = dis(D\mathit{HB}_\mathcal{D})$, and we will present two natural ways for formulating the disjunctive consequence operator $\mathcal{T}^s_\mathcal{D}$ as a generic $\mathcal{T}_\mathcal{D}$–operator. This gives an alternative proof that $\mathcal{T}^s_\mathcal{D}$ is monotonic and continuous. Moreover, in the next section the second transformation will be extended to dd–databases with default negation.

Transformation 1.

The first approach uses the built–in predicate *subsumes*:

$$\mathcal{D}^\odot = \{\ dis(X) \leftarrow dis(B_1 \vee X) \wedge \ldots \wedge dis(B_m \vee X) \wedge subsumes(\alpha, X) \mid \text{there exists } \alpha \leftarrow B_1 \wedge \ldots \wedge B_m \in \mathcal{D}\ \} \ \cup\ I^{bi}_\mathcal{D},$$

where $X \in \mathcal{V}$ is a variable symbol for formulas, and the common concept of subsumption for ground disjunctions $\alpha, \alpha' \in D\mathit{HB}_\mathcal{D}$ is used, i.e. $\alpha \geq \alpha'$ iff $\alpha \trianglerighteq \alpha'$. Then the $\mathcal{T}^s_\mathcal{D}$–operator on disjunctive Herbrand states S can be characterized using the generic $\mathcal{T}_\mathcal{D}$–operator and the expansion for disjunctive Herbrand states:

$$\mathcal{T}^{\odot,\trianglerighteq}_\mathcal{D}(dis(S) \cup I^{bi}_\mathcal{D}) = dis(exp_\trianglerighteq(\mathcal{T}^s_\mathcal{D}(S))) \cup\ I^{bi}_\mathcal{D}.$$

Example 4. For $\mathcal{D} = \{\, a \vee b \leftarrow c,\ c \vee d \,\}$ we get

$$\mathcal{D}^\odot = \{\ dis(X) \leftarrow dis(c \vee X) \wedge subsumes(a \vee b, X),\quad dis(X) \leftarrow subsumes(c \vee d, X)\ \} \ \cup\ I^{bi}_\mathcal{D}.$$

The rule $dis(X) \leftarrow dis(B_1 \vee X) \wedge subsumes(\alpha, X)$ that is derived from $a \vee b \leftarrow c$ has one disjunction $B_1 \vee X$ in its body, which extends $B_1 = c$ by some disjunction X that must be subsumed by $\alpha = a \vee b$. In a fixpoint iteration with $\mathcal{T}^{\odot,\geq}_\mathcal{D}$, first the fact $dis(c \vee d)$ is derived (amoung other facts). In the second iteration, using the ground instance $dis(a \vee b \vee d) \leftarrow dis(c \vee a \vee b \vee d) \wedge subsumes(a \vee b, a \vee b \vee d)$, which is obtained for $X = a \vee b \vee d$, and the fact $dis(c \vee d)$, it is possible to derive the fact $dis(a \vee b \vee d)$, since $c \vee d \trianglerighteq c \vee a \vee b \vee d$ and $subsumes(a \vee b, a \vee b \vee d) \in I^{bi}_\mathcal{D}$.

Transformation 2.

Another way of transforming a positive–disjunctive deductive database $\mathcal{D}$ is

$$\mathcal{D}^\odot = \{\ dis(\alpha \vee X_1 \vee \ldots \vee X_m) \leftarrow dis(B_1 \vee X_1) \wedge \ldots \wedge dis(B_m \vee X_m) \mid \text{there exists } \alpha \leftarrow B_1 \wedge \ldots \wedge B_m \in \mathcal{D}\ \},$$

with variable symbols $X_i \in \mathcal{V}$ ($i \in I\!N_+$) for formulas. If subsumption "$\geq$" for ground disjunctions $\alpha, \alpha' \in D_{HB_\mathcal{D}}$ is defined as identity (where disjunctions are treated as sets), i.e. $\alpha \geq \alpha'$ iff $\alpha = \alpha'$, then the disjunctive consequence operator $\mathcal{T}^s_\mathcal{D}$ on disjunctive Herbrand states S coincides with the generic $\mathcal{T}_\mathcal{D}$–operator:

$$\mathcal{T}^{\odot,\trianglerighteq}_\mathcal{D}(S) = dis(\mathcal{T}^s_\mathcal{D}(S)).$$

Example 5. For $\mathcal{D} = \{\, a \vee b \leftarrow c,\ c \vee d \,\}$ we get

$$\mathcal{D}^\odot = \{\ dis(a \vee b \vee X_1) \leftarrow dis(c \vee X_1),\ dis(c \vee d)\ \}.$$

In a fixpoint iteration with $\mathcal{T}^{\odot,\trianglerighteq}_\mathcal{D}$, first the fact $dis(c \vee d)$ is derived. In the second iteration, using the ground instance $dis(a \vee b \vee d) \leftarrow dis(c \vee d)$ of the first rule (obtained for $X_1 = d$) and the fact $dis(c \vee d)$, it is possible to derive the fact $dis(a \vee b \vee d)$.

5. Clausal Deductive Databases with Default Negation

We will transform a disjunctive deductive database $\mathcal{D}$ into a corresponding clausal deductive database $\mathcal{D}^{\mathsf{cd}} = \mathcal{D}^\odot \cup \mathcal{D}^{logic} \cup \mathcal{D}^{cwa} \cup I^{bi}_\mathcal{D}$, which we call the cd–*transformation* of $\mathcal{D}$. It consists of four parts: a set

$$\begin{aligned} \mathcal{D}^\odot = \{\ & dis(\alpha \vee X_1 \vee \ldots \vee X_m) \leftarrow \\ & \qquad dis(B_1 \vee X_1) \wedge \ldots dis(B_m \vee X_m) \wedge \\ & \qquad not\, con(C_1) \wedge \ldots \wedge not\, con(C_n) \mid \\ & \text{there exists } \ \alpha \leftarrow B_1 \wedge \ldots \wedge B_m \wedge not\, C_1 \wedge \ldots \wedge not\, C_n \in \mathcal{D}\ \}, \end{aligned}$$

of rules obtained from $\mathcal{D}$ (by extending *Transformation 2* of Section 4.2), with variable symbols $X_i \in \mathcal{V}$ ($i \in I\!N_+$) for formulas, a set $\mathcal{D}^{logic}$ of logical rules for relating the predicate symbols dis and con, a set $\mathcal{D}^{cwa} = \{\, r^{cwa}_{neg} \,\}$ with a *closed–world–assumption*, and the built–in interpretation $I^{bi}_\mathcal{D}$. The cd–facts in $I^{bi}_\mathcal{D}$ and the cd–rules in $\mathcal{D}^{logic} \cup \mathcal{D}^{cwa}$ are *generic*; they depend only on the Herbrand base $H_{B_\mathcal{D}}$, and not on the structure of the rules of $\mathcal{D}$.

We will prove that the stable models of the cd–transformation $\mathcal{D}^{\mathsf{cd}}$ correspond to the stable models of $\mathcal{D}$. Thus, we have shown a very natural way of computing the stable models of a disjunctive deductive database $\mathcal{D}$ based on the stable models of a normal deductive database $\mathcal{D}^{\mathsf{cd}}$. Furthermore, we will extend the set $\mathcal{STABLE}_2(\mathcal{D})$ to a set $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D})$ of so–called *stable states*, which are clausal Herbrand interpretations of $\mathcal{D}$ that are derived using a special *fixpoint condition* for $\mathcal{D}^{\mathsf{cd}}$. Stable states form a useful new semantics for dd–databases, which is different from the stable model semantics. In contrast, the well–founded model I_{wfs} of $\mathcal{D}^{\mathsf{cd}}$, which is the least partial stable model in the knowledge ordering, is not suitable for defining a state semantics.

5.1. The Generic Rules

The following cd–rules represent *closed–world–assumptions*; they allow for deriving certain disjunctions if their complemantary conjunctions cannot be derived:

$$\begin{aligned} r^{cwa}_{neg} &= dis(X_d) \leftarrow negative(X_d) \wedge complement(X_d, X_c) \wedge not\ con(X_c), \\ r^{cwa}_{pos} &= dis(X_d) \leftarrow positive(X_d) \wedge complement(X_d, X_c) \wedge not\ con(X_c), \\ r^{cwa}_{gen} &= dis(X_d) \leftarrow complement(X_d, X_c) \wedge not\ con(X_c). \end{aligned}$$

In $\mathcal{D}^{\mathsf{cd}}$ we will only use the rule r^{cwa}_{neg} for deriving negative disjunctions $\alpha \in N\mathit{HB}_{\mathcal{D}}$. In a more general framework for extended–disjunctive deductive databases ([15]) one could as well derive positive or general disjunctions by a closed–world–assumption.

The following axiomatic clauses in the set

$$\mathcal{D}^{logic} = \{\, r^{logic}_{res},\, r^{logic}_{sub},\, r^{logic}_{con},\, r^{logic}_{lit},\, r^{logic}_{emp} \,\}$$

describe the correlation between the atoms for disjunctions and conjunctions:

$$\begin{aligned} r^{logic}_{res} &= dis(X_d) \leftarrow dis(X'_d) \wedge dis(X''_d) \wedge resolvent(X'_d, X''_d, X_d), \\ r^{logic}_{sub} &= dis(X_d) \leftarrow dis(X'_d) \wedge subsumes(X'_d, X_d), \\ r^{logic}_{con} &= con(X_c) \leftarrow con(X'_c) \wedge con(X''_c) \wedge conjunction(X'_c, X''_c, X_c), \\ r^{logic}_{lit} &= con(X_c) \leftarrow literal(X_c) \wedge dis(X_c), \\ r^{logic}_{emp} &= con(\Box_c). \end{aligned}$$

The Rule/Goal–Graph for a Small Example

In Figure 5 a condensed version of the *rule/goal–graph* of *gnd* $(\mathcal{D}^{logic} \cup \{\, r^{cwa}_{neg} \,\})$ is given for a dd–database $\mathcal{D}$ over a language $\mathcal{L}$ with only two ground atoms, namely with $\mathit{HB}_{\mathcal{D}} = \{\, a, b \,\}$. Many atoms are omitted, e.g., $dis(\neg a \vee b)$ and $con(a \wedge \neg b)$. The atoms $dis(a)$ and $con(a)$, which both stand for a, are identified, and similarly $dis(b)$ and $con(b)$, which both stand for b. Furthermore, an arc to

$dis(\neg a)$ is drawn as a negative arc to $con(a)$, and an arc to $dis(\neg b)$ is drawn as a negative arc to $con(b)$; this is motivated by the ground instances $r^{cwa}_{neg}\theta_a$ and $r^{cwa}_{neg}\theta_b$ of the closed–world–assumption, where $\theta_a = \{ X_d \mapsto a,\, X_c \mapsto \neg a \}$ and $\theta_b = \{ X_d \mapsto b,\, X_c \mapsto \neg b \}$. Figure 5 is based on the following ground instances:

$$\begin{aligned} r^{cwa}_{neg}\theta_1 = dis(\neg a \vee \neg b) \leftarrow\; & negative(\neg a \vee \neg b) \wedge \\ & complement(\neg a \vee \neg b, a \wedge b) \wedge not\; con(a \wedge b), \end{aligned}$$

$$\begin{aligned} r^{logic}_{res}\theta_2 &= dis(a) \leftarrow dis(a \vee b) \wedge dis(\neg b) \wedge resolvent(a \vee b, \neg b, a), \\ r^{logic}_{res}\theta_3 &= dis(b) \leftarrow dis(a \vee b) \wedge dis(\neg a) \wedge resolvent(a \vee b, \neg a, b), \end{aligned}$$

$$r^{logic}_{con}\theta_4 = con(a \wedge b) \leftarrow con(a) \wedge con(b) \wedge conjunction(a, b, a \wedge b).$$

It can be seen that there is a negative cycle involving $dis(a)$ and $dis(b)$, if $dis(a \vee b)$

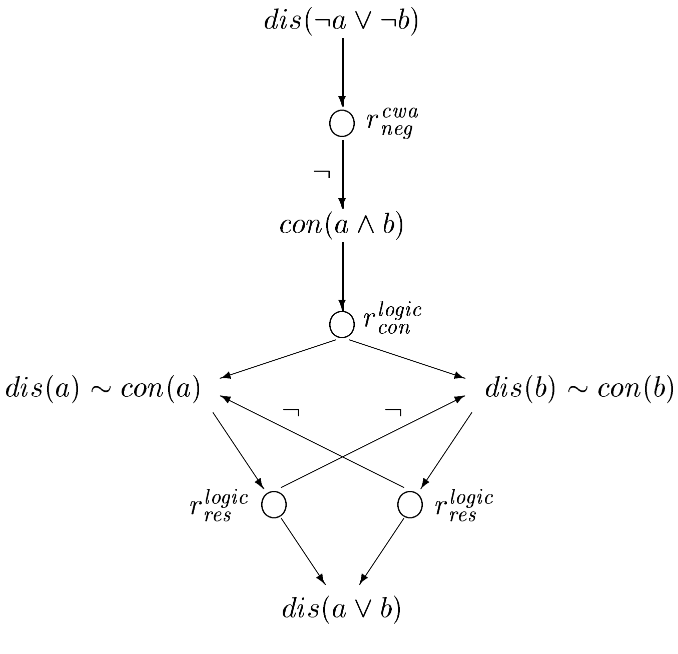

Figure 5. A Condensed Version of the Rule/Goal–Graph

happens to be *true*. In stable model semantics this cycle would lead to alternative assignments of *true* and *false* to $dis(a)$ and $dis(b)$, respectively. In well–founded semantics (partial stable model semantics) the cycle would lead to the assignment of *undefined* to $dis(a)$, $dis(b)$, $con(a)$, and $con(b)$. Consequently, also $con(a \wedge b)$

would become *undefined* because of the rule r^{logic}_{con}, and $dis(\neg a \vee \neg b)$ would become *undefined*, because of the rule r^{cwa}_{neg}. This behaviour will be analysed in the following Sections 5.2 and 5.3. In Section 5.4 a three–valued semantics will be proposed, which does not use the rule r^{logic}_{con} for propagating *undefinedness*. Thus, it will be possible to assign *false* to $con(a \wedge b)$, although both $con(a)$ and $con(b)$ are *undefined*.

5.2. Two–Valued Analysis of $\mathcal{D}^{\mathsf{cd}}$

We want to investigate the total Herbrand models I of the clausal deductive database $\mathcal{D}^{\mathsf{cd}}$. Since I is total, it must assign one of the truth values $\mathtt{t}$ or $\mathtt{f}$ to each of the atoms $dis(\alpha)$ and $con(\beta)$, for $\alpha \in \mathit{GHB}_{\mathcal{D}}$, $\beta \in \mathit{CHB}_{\mathcal{D}}$ – in particular also if α and β are literals $A, \neg A \in \mathit{HL}_{\mathcal{D}}$. Due to the subsumption rule r^{logic}_{sub} it must hold $I(\alpha_1 \vee \alpha_2) = I(\alpha_1) \vee I(\alpha_2)$, for all $\alpha_1, \alpha_2 \in \mathit{GHB}_{\mathcal{D}}$.

If $I(dis(\Box_d)) = \mathtt{t}$, then we define $I^{\otimes} = J^{\top}$, where $J^{\top}(A) = \top$, for all $A \in \mathit{HB}_{\mathcal{D}}$, with a fourth truth value $\top$ (*overdefined*) denoting *inconsistency*. $J^{\top}$ is considered as a special total Herbrand interpretation of $\mathcal{D}$, where all atoms A are both true and false. If $I(dis(\Box_d)) = \mathsf{f}$, then I can be represented by an (ordinary) total Herbrand interpretation $I^{\otimes}$ of $\mathcal{D}$, where $I^{\otimes}(A) = I(dis(A))$, for all $A \in \mathit{HB}_{\mathcal{D}}$. Conversely, a total Herbrand interpretation J of $\mathcal{D}$ defines an Herbrand interpretation $J^{\odot}$ of $\mathcal{D}^{\mathsf{cd}}$ by $J^{\odot}(dis(\alpha)) = J(\alpha)$, for all $\alpha \in \mathit{GHB}_{\mathcal{D}}$, and $J^{\odot}(con(\beta)) = J(\beta)$, for all $\beta \in \mathit{CHB}_{\mathcal{D}}$.

For a total Herbrand model I of $\mathcal{D}^{\mathsf{cd}}$, it holds $(I^{\otimes})^{\odot} = I$, and for a total Herbrand model J of $\mathcal{D}$, it holds $(J^{\odot})^{\otimes} = J$. A set $\mathcal{I}$ of total Herbrand models of $\mathcal{D}^{\mathsf{cd}}$ can be represented by the set $\mathcal{I}^{\otimes} = \{\, I^{\otimes} \mid I \in \mathcal{I}\,\}$ of Herbrand interpretations of $\mathcal{D}$.

Theorem 6 (Characterization of Stable Models).
Given a disjunctive deductive database $\mathcal{D}$, then it holds

$$\mathcal{STABLE}_2(\mathcal{D}^{\mathsf{cd}})^{\otimes} = \mathcal{STABLE}_2(\mathcal{D}) \cup \mathcal{J}, \quad \text{where} \quad \mathcal{J} = \emptyset \quad \text{or} \quad \mathcal{J} = \{\, J^{\top} \,\}.$$

The proof is given in Appendix A. It would be possible to forbid $(J^{\top})^{\odot}$ as a stable model of $\mathcal{D}^{\mathsf{cd}}$ by adding one of the following (technical) rules to $\mathcal{D}^{\mathsf{cd}}$:

$$r^{logic}_{cons} = dis(c_{\top}) \leftarrow dis(\Box_d) \wedge not\ dis(c_{\top}),$$

$$r^{logic'}_{cons} = \leftarrow dis(\Box_d).$$

$c_\top$ is a new 0–ary predicate symbol such that the atom $c_\top$ does not unify with any formula $\alpha \in G_{HB_\mathcal{D}}$. r^{logic}_{cons} is a cd–rule, and $r^{logic\,\prime}_{cons}$ is an equivalent denial rule, which is easier to understand, but is no cd–rule.

5.3. Three–Valued Analysis of $\mathcal{D}^{\mathsf{cd}}$

In the following we will investigate the effects of the propagation rules r^{logic}_{res} and r^{logic}_{con} for disjunctions and conjunctions, respectively, and the closed–world–assumption r^{cwa}_{neg} in three–valued logic. We will see that the well–founded model I_{wfs} of the normal deductive database $\mathcal{D}^{\mathsf{cd}}$ is not appropriate for designing a useful state semantics. Likewise, also the partial stable models of $\mathcal{D}^{\mathsf{cd}}$ are not useful, since the well–founded model I_{wfs} is the least partial stable model of $\mathcal{D}^{\mathsf{cd}}$ in the knowledge–ordering[10].

The Conjunction Rule r^{logic}_{con}.

Due to r^{logic}_{con} the partial stable models I exhibit the following behaviour for conjunctions $\beta = L_1 \wedge \ldots \wedge L_k \in C_{HB_\mathcal{D}}$:

$$I(con(\beta)) \geq_{\mathsf{t}} \mathsf{u} \iff (\,\forall i \in \langle\, 1, k\,\rangle : I(con(L_i)) \geq_{\mathsf{t}} \mathsf{u}\,).$$

r^{cwa}_{neg} supports $dis(\neg{\cdot}L_1 \vee \ldots \vee \neg{\cdot}L_k)$, if $I(con(\beta)) = \mathsf{f}$. In this case $I(con(L_i)) = \mathsf{f}$ must hold for some $i \in \langle\, 1, k\,\rangle$, i.e. r^{cwa}_{neg} also supports $dis(\neg{\cdot}L_i)$. In a state–based semantics it should be possible that the closed–world–assumption can support a negative disjunction $dis(\neg A_1 \vee \neg A_2)$ without supporting one of $dis(\neg A_1)$ and $dis(\neg A_2)$. For three–valued models I, such a behaviour is impossible.[11]

Example 7. The *stable states semantics* of $\mathcal{D} = \{\, a \leftarrow not\, b,\, b \leftarrow not\, a\,\}$ will be given by the three clausal Herbrand interpretations $S_1 = \{\, a, \neg b\,\}$, $S_2 = \{\, \neg a, b\,\}$, and $S_3 = \{\, \neg a \vee \neg b\,\}$. S_3 corresponds to the partial Herbrand interpretation $I = S_3^{\mathsf{cd}}$ of $\mathcal{D}^{\mathsf{cd}}$, where $dis(\neg a \vee \neg b)$ is derived since $con(a \wedge b)$ cannot be derived.

The Resolution Rule r^{logic}_{res}.

r^{logic}_{res} has the following consequence for disjunctions $\alpha_1, \alpha_2 \in G_{HB_\mathcal{D}}$ and atoms $A \in H_{B_\mathcal{D}}$:

[10] i.e. I_{wfs} exactly is the set of consequences under the semantics of partial stable models

[11] $dis(\neg A_1 \vee \neg A_2)$ is supported, if $I(con(A_1 \wedge A_2)) = \mathsf{f}$. On the other hand, assuming that $dis(\neg A_1)$ and $dis(\neg A_2)$ are not supported, it must hold $I(con(A_i)) \geq_{\mathsf{t}} \mathsf{u}$, for $i = 1, 2$, which implies $I(con(A_1 \wedge A_2)) \geq_{\mathsf{t}} \mathsf{u}$, and contradicts $I(con(A_1 \wedge A_2)) = \mathsf{f}$.

If $\alpha_1 \vee \alpha_2$ is the binary resolvent of two disjunctions $A \vee \alpha_1$, $\neg A \vee \alpha_2$, such that $I(dis(A \vee \alpha_1)) \geq_{\mathsf{t}} \mathsf{u}$, $I(dis(\neg A \vee \alpha_2)) \geq_{\mathsf{t}} \mathsf{u}$, then $I(dis(\alpha_1 \vee \alpha_2)) \geq_{\mathsf{t}} \mathsf{u}$.

This is undesirable and unituitive since it could be the case that $I(dis(A)) = \mathsf{u}$, and it does not make sense to resolve on undefined atoms.

Example 8. The *stable state semantics* of $\mathcal{D} = \{\, a \leftarrow not\ a,\ b \leftarrow c \,\}$ will be given by the clausal Herbrand interpretation $S = \{\neg b, \neg c\}$ corresponding to the partial Herbrand interpretation $I = S^{\mathsf{cd}}$ of $\mathcal{D}^{\mathsf{cd}}$. Here $I(dis(a)) = I(dis(a \vee b)) = I(dis(\neg a \vee c)) = \mathsf{u}$, but for the resolvent we get $I(dis(b \vee c)) = \mathsf{f}$.

5.4. Stable State Semantics

Stable states are clausal Herbrand interpretations S of a dd–database $\mathcal{D}$ that are defined by a special *fixpoint condition* $\Gamma_{\mathcal{C}^{\mathsf{t}}}(\Gamma_{\mathcal{C}^{\mathsf{u}}}(S^{\mathsf{cd}})) = S^{\mathsf{cd}}$ in terms of the $\Gamma_{\mathcal{C}}$–operators [20] for the cd–transformation $\mathcal{C}^{\mathsf{t}} = \mathcal{D}^{\mathsf{cd}}$ and a variant $\mathcal{C}^{\mathsf{u}}$ of $\mathcal{D}^{\mathsf{cd}}$.

Definition 9 (Stable States).

Given a cd–database $\mathcal{C}$ and a dd–database $\mathcal{D}$.

1. The (unique) minimal model – considered as a set of atoms – of the GL–transformation $\mathcal{C}^{I_t}$ of $\mathcal{C}$ w.r.t. a total Herbrand interpretation $I_t \subseteq H\!B_{\mathcal{C}}$ of $\mathcal{C}$ is denoted by $\Gamma_{\mathcal{C}}(I_t)$.
2. A clausal Herbrand interpretation S of $\mathcal{D}$ is called a *stable state* of $\mathcal{D}$ if for the set $I_t = S^{\mathsf{cd}}$ of atoms and the cd–databases

$$\mathcal{C}^{\mathsf{t}} = \mathcal{D}^{\mathsf{cd}}, \quad \mathcal{C}^{\mathsf{u}} = I_t \cup \mathcal{D}^{\mathsf{cd}} \setminus \{\, r^{logic}_{res},\, r^{logic}_{con} \,\}$$

it holds $\Gamma_{\mathcal{C}^{\mathsf{t}}}(\Gamma_{\mathcal{C}^{\mathsf{u}}}(I_t)) = I_t$. The set of all stable states of $\mathcal{D}$ is denoted by $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D})$.

The underlying idea is derived from the well–known approach that has been used for defining the *well–founded semantics* I_{wfs} of a normal deductive database $\mathcal{C}$: $I_{wfs} = \langle\, I_t, I_u \,\rangle$, where $I_t \subseteq I_u \subseteq H\!B_{\mathcal{C}}$, is the least[12] partial Herbrand interpretation satisfying the following *alternating fixpoint condition*: $\Gamma_{\mathcal{C}}(I_u) = I_t$ and $\Gamma_{\mathcal{C}}(I_t) = I_u$, which is equivalent to $\Gamma_{\mathcal{C}}(\Gamma_{\mathcal{C}}(I_t)) = I_t$ and $\Gamma_{\mathcal{C}}(I_t) = I_u$.

In Definition 9 this idea is applied to the normal deductive database $\mathcal{C}^{\mathsf{t}} = \mathcal{D}^{\mathsf{cd}}$ and a set $I_t = S^{\mathsf{cd}}$ of atoms. Based on the observations of Section 5.3,

[12] in the knowledge ordering: $\langle\, I_t^1, I_u^1 \,\rangle \leq_{\mathsf{k}} \langle\, I_t^2, I_u^2 \,\rangle$ iff $I_t^1 \subseteq I_t^2$ and $I_u^2 \subseteq I_u^1$

the condition is modified such that the rules r^{logic}_{res} and r^{logic}_{con} are not used for computing I_u. Instead, the set I_t of atoms is included in $\mathcal{C}^{\mathsf{u}}$ to assure that $I_t \subseteq I_u = \Gamma_{\mathcal{C}^{\mathsf{u}}}(I_t)$.

According to the approach taken by the well–founded semantics, all atoms in $HB_{\mathcal{D}^{\mathsf{cd}}} \setminus I_u$ could be assumed to be false based on the closed–world assumption. Stable state semantics, however, applies the closed–world assumption to atoms $con(\gamma)$ for positive conjunctions $\gamma = A_1 \wedge \ldots \wedge A_k$ only: if $con(A_1 \wedge \ldots \wedge A_k) \notin I_u$, then $dis(\neg A_1 \vee \ldots \vee \neg A_k) \in I_t$ due to the rule r^{cwa}_{neg}.

The stable models I of $\mathcal{C}^{\mathsf{t}} = \mathcal{D}^{\mathsf{cd}}$ are characterized by the fixpoint condition $\Gamma_{\mathcal{C}^{\mathsf{t}}}(I_t) = I_t$. In the following we will represent clausal Herbrand interpretations (and thus stable states) S by their canonical form $can_{\trianglerighteq}(S)$. As a consequence of Theorem 6, for a disjunctive deductive database $\mathcal{D}$, the representation of a stable model of $\mathcal{D}$ as a partial Herbrand interpretation $J \subseteq HL_{\mathcal{D}}$ is a stable state of $\mathcal{D}$.

Theorem 10 (Stable States).
Given a disjunctive deductive database $\mathcal{D}$, then it hols

$$\mathcal{STABLE}_2(\mathcal{D}) \subseteq \mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}).$$

This shows that the semantics of stable states is at most as strong as the semantics of stable models, i.e. $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D})^{\cap} \subseteq \mathcal{STABLE}_2(\mathcal{D})^{\cap}$. The following Example 11 will give some dd–databases ($\mathcal{D}_2$, $\mathcal{D}_3$, $\mathcal{D}_4$, $\mathcal{D}_5$) where it is strictly weaker. The predicate dependency graphs of $\mathcal{D}_i$, $i \in \langle 2, 4 \rangle$, can be found in Figure 6. For many prototypical databases which have been investigated frequently in literature we will compare the stable states to the stable models and – for normal databases – to the well–founded model.

Example 11 (Stable States).

1. For the positive–disjunctive deductive database $\mathcal{D}_1 = \{\, a \vee b \,\}$ we get

$$\mathcal{STABLE}_2(\mathcal{D}_1) = \mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_1) = \{\, \{\, a, \neg b \,\},\ \{\, \neg a, b \,\} \,\}.$$

   The intersection is $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_1)^{\cap} = \mathcal{STABLE}_2(\mathcal{D}_1)^{\cap} = \{\, a \vee b,\, \neg a \vee \neg b \,\}$.

2. For the normal deductive database $\mathcal{D}_2 = \{\, a \leftarrow \mathit{not}\, b,\ b \leftarrow \mathit{not}\, a \,\}$, we get

$$\begin{aligned} \mathcal{STABLE}_2(\mathcal{D}_2) &= \{\, \{\, a, \neg b \,\},\ \{\, \neg a, b \,\} \,\}, \\ \mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_2) &= \mathcal{STABLE}_2(\mathcal{D}_2) \cup \{\, \{\, \neg a \vee \neg b \,\} \,\}. \end{aligned}$$

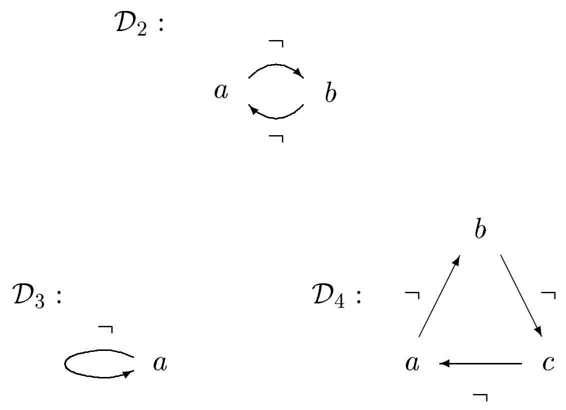

Figure 6. Predicate Dependency Graphs

$\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_2)^{\cap} = \{\, \neg a \vee \neg b \,\}$, is weaker than the state $\mathcal{STABLE}_2(\mathcal{D}_2)^{\cap} = \{\, a \vee b,\ \neg a \vee \neg b \,\}$ derived from the stable models, but stronger than the well–founded model $\emptyset$.

3. For the normal deductive database $\mathcal{D}_3 = \{a \leftarrow not\, a\}$, which does not possess any (total) stable model, there is unique stable state $S_3 = \emptyset$, which is identical to the well–founded model. A similar structure may for instance *become active* for "$a = win(x)$" within the database $\mathcal{D}_{win}$ of Subsection 2.2, if the state S contains cyclic moves "$move(x,x)$", namely $win(x) \leftarrow move(x,x) \wedge not\ win(x)$.

4. For the normal deductive database $\mathcal{D}_4 = \{\, a \leftarrow not\, b,\ b \leftarrow not\, c,\ c \leftarrow not\, a \,\}$, we get $\mathcal{STABLE}_2(\mathcal{D}_4) = \emptyset$, but there is a unique stable state

$$S_4 = \{\, \neg a \vee \neg b,\ \neg b \vee \neg c,\ \neg a \vee \neg c \,\},$$

which is weaker than the inconsistent state $\mathcal{STABLE}_2(\mathcal{D}_4)^{\cap} = \mathit{GHB}_{\mathcal{D}_4}$ derived from the stable models, but stronger than the empty well–founded model $\emptyset$.

5. For the dd–database $\mathcal{D}_5 = \mathcal{D}_4 \cup \{\, a \vee b \vee c \,\}$ which does not possess any stable or partial stable models, there is a unique stable state $S_5 = S_4 \cup \{\, a \vee b \vee c \,\}$.

6. For the stratified–disjunctive deductive database $\mathcal{D}_6 = \{\, a \leftarrow not\, b \,\}$, we get a unqiue stable state $S_6 = \{\, a,\ \neg b \,\}$, which is the unique (total) stable model of $\mathcal{D}_6$, namely the perfect model of $\mathcal{D}_6$.

7. For both of the stratified–disjunctive deductive databases $\mathcal{D}_7' = \{\, c \leftarrow a,\ c \leftarrow b,\ a \vee b \,\}$ and $\mathcal{D}_7'' = \{\, c \leftarrow \mathit{not}\, a,\ c \leftarrow \mathit{not}\, b,\ a \vee b \,\}$ we get (for $\mathcal{D}_7 \in \{\, \mathcal{D}_7', \mathcal{D}_7'' \,\}$)

$$\mathcal{STABLE}_2(\mathcal{D}_7) = \mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_7) = \{\ \{\, a,\ \neg b,\ c \,\},\ \{\, \neg a,\ b,\ c \,\}\ \}.$$

   The intersection is $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_7)^{\cap} = \mathcal{STABLE}_2(\mathcal{D}_7)^{\cap} = \{\, a \vee b,\ \neg a \vee \neg b,\ c \,\}$.

8. For the dd–database $\mathcal{D}_8 = \{\ a,\ \leftarrow a\ \}$, which has got no Herbrand models, there exists one stable state $S_8 = \{\, \Box_d \,\}$, i.e. $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}_8)^{\cap} = \mathit{GHB}_{\mathcal{D}_8}$.

Especially for dd–databases like $\mathcal{D}_3$, $\mathcal{D}_4$, $\mathcal{D}_5$, which are inconsistent under stable model semantics, stable states are very helpful since they provide a plausible, consistent semantics.

For positive–disjunctive deductive databases the intersection $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D})^{\cap}$ of all stable states corresponds to the union of the *minimal model state* $\mathit{MS}_{\mathcal{D}}$ and the *extended generalized closed–world-assumption* $\mathit{EGCWA}_{\mathcal{D}}$, as defined by [11]. The general disjunctions in $\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D})^{\cap}$ can be derived by repeatedly applying binary resolution to disjunctions from $\mathit{MS}_{\mathcal{D}}$ – which is a disjunctive Herbrand state – and $\mathit{EGCWA}_{\mathcal{D}}$ – which is represented as a negative Herbrand state.

5.5. A Characterization of Stable State Semantics

In this section, we will give a *characterization* of the *stable states* of a dd–database in terms of the stable models of an *annotated* cd–database $\mathcal{D}^{\mathsf{cd,tu}}$ by modifying the concept of the tu–*transformation* known from [23].

The tu–*Transformation*

The tu–*representation* of a partial Herbrand interpretation I of a dd–database $\mathcal{D}$ is given by

$$I^{\mathsf{tu}} = \{\ A^{\mathsf{t}} \mid A \in \mathit{HB}_{\mathcal{D}} \wedge I(A) \geq_{\mathsf{t}} \mathsf{t}\ \} \cup \{\ A^{\mathsf{u}} \mid A \in \mathit{HB}_{\mathcal{D}} \wedge I(A) \geq_{\mathsf{t}} \mathsf{u}\ \}.$$

E.g., for the Herbrand base $\mathit{HB}_{\mathcal{D}} = \{\, a, b, c, d \,\}$ and $I(a) = \mathsf{t}$, $I(b) = \mathsf{t}$, $I(c) = \mathsf{f}$, and $I(d) = \mathsf{u}$, we get $I^{\mathsf{tu}} = \{\, a^{\mathsf{t}}, a^{\mathsf{u}}, b^{\mathsf{t}}, b^{\mathsf{u}}, d^{\mathsf{u}} \,\}$.

For dd–databases we will use a special concept of annotating disjunctive rules, which will encode the condition that partial Herbrand models have to fulfill in terms of their tu–representation. Given a truth value $\mathsf{v} \in \{\, \mathsf{t}, \mathsf{u} \,\}$, for a disjunction $\alpha = A_1 \vee \ldots \vee A_k$ and a conjunction $\beta = B_1 \wedge \ldots \wedge B_m$ of atoms we define $\alpha^{\mathsf{v}} = A_1^{\mathsf{v}} \vee \ldots \vee A_k^{\mathsf{v}}$ and $\beta^{\mathsf{v}} = B_1^{\mathsf{v}} \wedge \ldots \wedge B_m^{\mathsf{v}}$.

Definition 12 (tu–Transformation, [23]).

1. For a disjunctive rule $r = \alpha \leftarrow \beta \wedge not \cdot \gamma$ we define the *annotated rules*
$$r^{\mathsf{t}} = \alpha^{\mathsf{t}} \leftarrow \beta^{\mathsf{t}} \wedge not \cdot \gamma^{\mathsf{u}}, \quad r^{\mathsf{u}} = \alpha^{\mathsf{u}} \leftarrow \beta^{\mathsf{u}} \wedge not \cdot \gamma^{\mathsf{t}}.$$
2. For a dd–database $\mathcal{D}$ we define $\mathcal{D}^{\mathsf{t}} = \{ r^{\mathsf{t}} \mid r \in \mathcal{D} \}$, $\mathcal{D}^{\mathsf{u}} = \{ r^{\mathsf{u}} \mid r \in \mathcal{D} \}$, and the *annotated database* $\mathcal{D}^{\mathsf{tu}} = \mathcal{D}^{\mathsf{t}} \cup \mathcal{D}^{\mathsf{u}} \cup \{ A^{\mathsf{u}} \leftarrow A^{\mathsf{t}} \mid A \in H\!B_{\mathcal{D}} \}$ is called the tu–*transformation* of $\mathcal{D}$.

Example 13 (tu–Transformation).
For the dd–database $\mathcal{D} = \{ a \vee b,\ c \leftarrow b \wedge not\, a,\ c \leftarrow a \wedge not\, b \}$, we get

$$\begin{aligned}
\mathcal{D}_1^{\mathsf{t}} &= \{ a^{\mathsf{t}} \vee b^{\mathsf{t}},\ c^{\mathsf{t}} \leftarrow b^{\mathsf{t}} \wedge not\, a^{\mathsf{u}},\ c^{\mathsf{t}} \leftarrow a^{\mathsf{t}} \wedge not\, b^{\mathsf{u}} \}, \\
\mathcal{D}_1^{\mathsf{u}} &= \{ a^{\mathsf{u}} \vee b^{\mathsf{u}},\ c^{\mathsf{u}} \leftarrow b^{\mathsf{u}} \wedge not\, a^{\mathsf{t}},\ c^{\mathsf{u}} \leftarrow a^{\mathsf{u}} \wedge not\, b^{\mathsf{t}} \}, \\
\mathcal{D}_1^{\mathsf{tu}} &= \mathcal{D}_1^{\mathsf{t}} \cup \mathcal{D}_1^{\mathsf{u}} \cup \{ a^{\mathsf{u}} \leftarrow a^{\mathsf{t}},\ b^{\mathsf{u}} \leftarrow b^{\mathsf{t}},\ c^{\mathsf{u}} \leftarrow c^{\mathsf{t}} \}.
\end{aligned}$$

The construction of $\mathcal{D}^{\mathsf{tu}}$ is motivated by condition $M(\alpha) \geq_t M(\beta) \wedge \neg M(\gamma)$ which every partial Herbrand model M of a ground rule $r = \alpha \leftarrow \beta \wedge not \cdot \gamma$ must fulfill. This condition is encoded in the annotated rules r^{t} and r^{u}, since it is equivalent to the following:

$$\begin{aligned}
&(\ (\ M(\beta) \geq_{\mathsf{t}} \mathsf{t}\ \wedge\ \neg(M(\gamma) \geq_{\mathsf{t}} \mathsf{u}\ )\ \Longrightarrow\ M(\alpha) \geq_{\mathsf{t}} \mathsf{t}\ )\ \wedge \\
&(\ (\ M(\beta) \geq_{\mathsf{t}} \mathsf{u}\ \wedge\ \neg(M(\gamma) \geq_{\mathsf{t}} \mathsf{t}\ )\ \Longrightarrow\ M(\alpha) \geq_{\mathsf{t}} \mathsf{u}\ ).
\end{aligned}$$

The rules $A^{\mathsf{u}} \leftarrow A^{\mathsf{t}}$ are due to the fact that we want to perceive the Herbrand models of $\mathcal{D}^{\mathsf{tu}}$ as Herbrand models of $\mathcal{D}$ in tu–representation.

Characterization of Partial Stable Models

The following theorem, which was proved in [23], shows that the partial stable models of a dd–database $\mathcal{D}$ correspond to the total stable models of the annotated database $\mathcal{D}^{\mathsf{tu}}$. For any total Herbrand interpretation I of $\mathcal{D}^{\mathsf{tu}}$ we introduce the notation I^{∇} for the partial Herbrand interpretation of $\mathcal{D}$ that is induced by I, i.e. for $A \in H\!B_{\mathcal{D}}$

$$I^{\nabla}(A) = \begin{cases} \mathsf{t} \text{ if } I(A^{\mathsf{t}}) = \mathsf{t} \\ \mathsf{u} \text{ if } I(A^{\mathsf{u}}) = \mathsf{t} \text{ and } I(A^{\mathsf{t}}) = \mathsf{f} \\ \mathsf{f} \text{ if } I(A^{\mathsf{u}}) = \mathsf{f} \end{cases}$$

For a collection $\mathcal{I}$ of total Herbrand interpretations of $\mathcal{D}^{\mathsf{tu}}$, let $\mathcal{I}^{\nabla} = \{ I^{\nabla} \mid I \in \mathcal{I} \}$.

Theorem 14 (Partial Minimal and Partial Stable Models, [23]).
Given a disjunctive deductive database $\mathcal{D}$, then

$$\mathcal{STABLE}_3(\mathcal{D}) = \mathcal{STABLE}_2(\mathcal{D}^{\mathsf{tu}})^{\nabla}, \quad \text{and} \quad \mathcal{MM}_3(\mathcal{D}) = \mathcal{MM}_2(\mathcal{D}^{\mathsf{tu}})^{\nabla}.$$

The $(\mathsf{cd},\ \mathsf{tu})$*–Transformation*

Given a disjunctive deductive database $\mathcal{D}$, the following annotated clausal deductive database $\mathcal{D}^{\mathsf{cd},\mathsf{tu}}$ is called the $(\mathsf{cd},\ \mathsf{tu})$*–transformation* of $\mathcal{D}$:

$$\mathcal{D}^{\mathsf{cd},\mathsf{tu}} = (\mathcal{D}_t^{\mathsf{cd}})^{\mathsf{t}} \cup (\mathcal{D}_u^{\mathsf{cd}})^{\mathsf{u}} \cup \{\, r_{dis}^{\mathsf{tu}},\, r_{con}^{\mathsf{tu}} \,\}.$$

where $\mathcal{D}_t^{\mathsf{cd}} = \mathcal{D}^{\mathsf{cd}}$, $\mathcal{D}_u^{\mathsf{cd}} = \mathcal{D}^{\mathsf{cd}} \setminus \{\, r_{res}^{logic},\, r_{con}^{logic} \,\}$ and

$$r_{dis}^{\mathsf{tu}} = dis^{\mathsf{u}}(X_d) \leftarrow dis^{\mathsf{t}}(X_d), \quad r_{con}^{\mathsf{tu}} = con^{\mathsf{u}}(X_c) \leftarrow con^{\mathsf{t}}(X_c).$$

$\mathcal{D}^{\mathsf{cd},\mathsf{tu}}$ encodes the fixpoint condition $\Gamma_{\mathcal{C}^{\mathsf{t}}}(\Gamma_{\mathcal{C}^{\mathsf{u}}}(I_t)) = I_t$, where $I_t = S^{\mathsf{cd}}$, given in Definition 9, which stable states S must fulfill. The rules r_{dis}^{tu} and r_{con}^{tu} reflect the fact that $\mathcal{C}^{\mathsf{u}}$ must be a superset of I_t in Definition 9. Note that it obviously holds $\mathcal{D}^{\mathsf{cd},\mathsf{tu}} \subsetneq (\mathcal{D}^{\mathsf{cd}})^{\mathsf{tu}}$, since $\mathcal{D}_u^{\mathsf{cd}} \subsetneq \mathcal{D}^{\mathsf{cd}}$, which was motivated in Section 5.3.

Thus, $\mathcal{D}^{\mathsf{cd},\mathsf{tu}}$ contains the annotated rules from $(\mathcal{D}^{\odot})^{\mathsf{cd}}$ and annotated generic rules, namely closed–world–assumptions, and logical rules. The *negative closed–world–assumptions* are given by

$$\mathcal{D}^{cwa,\mathsf{tu}} = \{\, (r_{neg}^{cwa})^{\mathsf{t}} \,\} \cup \{\, (r_{neg}^{cwa})^{\mathsf{u}} \,\},$$

where

$$(r_{neg}^{cwa})^{\mathsf{t}} = dis^{\mathsf{t}}(X_d) \leftarrow negative^{\mathsf{t}}(X_d) \wedge complement^{\mathsf{t}}(X_d, X_c) \wedge not\ con^{\mathsf{u}}(X_c),$$
$$(r_{neg}^{cwa})^{\mathsf{u}} = dis^{\mathsf{u}}(X_d) \leftarrow negative^{\mathsf{u}}(X_d) \wedge complement^{\mathsf{u}}(X_d, X_c) \wedge not\ con^{\mathsf{t}}(X_c).$$

The *logical rules* are given by

$$\begin{aligned}\mathcal{D}^{logic,\mathsf{tu}} = \{\, &(r_{res}^{logic})^{\mathsf{t}},\, (r_{sub}^{logic})^{\mathsf{t}},\, (r_{con}^{logic})^{\mathsf{t}},\, (r_{lit}^{logic})^{\mathsf{t}},\, (r_{emp}^{logic})^{\mathsf{t}} \,\} \cup \\ \{\, &(r_{sub}^{logic})^{\mathsf{u}},\, (r_{lit}^{logic})^{\mathsf{u}},\, (r_{emp}^{logic})^{\mathsf{u}} \,\},\end{aligned}$$

which describe the correlation between the atoms for disjunctions and conjunctions:

$$(r_{res}^{logic})^{\mathsf{t}} = dis^{\mathsf{t}}(X_d) \leftarrow dis^{\mathsf{t}}(X_1) \wedge dis^{\mathsf{t}}(X_2) \wedge resolvent^{\mathsf{t}}(X_1, X_2, X_d),$$
$$(r_{sub}^{logic})^{\mathsf{t}} = dis^{\mathsf{t}}(X_d) \leftarrow dis^{\mathsf{t}}(X_1) \wedge subsumes^{\mathsf{t}}(X_1, X_d),$$

$$(r_{con}^{logic})^{\mathsf{t}} = con^{\mathsf{t}}(X_c) \leftarrow con^{\mathsf{t}}(X_1) \wedge con^{\mathsf{t}}(X_2) \wedge conjunction^{\mathsf{t}}(X_1, X_2, X_c),$$

$$(r^{logic}_{lit})^{\mathsf{t}} = con^{\mathsf{t}}(X_c) \leftarrow literal^{\mathsf{t}}(X_c) \wedge dis^{\mathsf{t}}(X_c),$$
$$(r^{logic}_{emp})^{\mathsf{t}} = con^{\mathsf{t}}(\Box_c),$$
$$(r^{logic}_{sub})^{\mathsf{u}} = dis^{\mathsf{u}}(X_d) \leftarrow dis^{\mathsf{u}}(X_1) \wedge subsumes^{\mathsf{u}}(X_1, X_d),$$
$$(r^{logic}_{lit})^{\mathsf{u}} = con^{\mathsf{u}}(X_c) \leftarrow literal^{\mathsf{u}}(X_c) \wedge dis^{\mathsf{u}}(X_c),$$
$$(r^{logic}_{emp})^{\mathsf{u}} = con^{\mathsf{u}}(\Box_c).$$

Note that it would not be necessary to annotate the atoms for the built–in predicate symbols, since these are defined without default–negation by the built–in Herbrand interpretation. Finally, the annotated database $(\mathcal{D}^{\odot})^{\mathsf{tu}}$ contains the rules for relating the predicate dis^{t} to dis^{t} and con^{t} to con^{t}, respectively. Thus, it holds $\mathcal{D}^{\mathsf{cd,tu}} = (\mathcal{D}^{\odot})^{\mathsf{tu}} \cup \mathcal{D}^{logic,\mathsf{tu}} \cup \mathcal{D}^{cwa,\mathsf{tu}} \cup (I^{bi}_{\mathcal{D}})^{\mathsf{t}} \cup (I^{bi}_{\mathcal{D}})^{\mathsf{u}}$.

Characterization of Stable States

For any total Herbrand interpretation I of $\mathcal{D}^{\mathsf{cd,tu}}$ we use the notation I^{∇} for the partial Herbrand interpretation of $\mathcal{D}^{\mathsf{cd}}$ that is induced by I, as we had done for tansforming total Herbrand interpretations I of annotated dd–databases $\mathcal{D}^{\mathsf{tu}}$ into partial Herbrand interpretations of the underlying dd–databases $\mathcal{D}$. Subsequently, a partial Herbrand interpretation I^{∇} of $\mathcal{D}^{\mathsf{cd}}$ is transformed to a clausal Herbrand interpretation

$$I^{\nabla^{\otimes}} = \{ \alpha \in G_{HB_{\mathcal{D}}} \mid I^{\nabla}(dis(\alpha)) = \mathsf{t} \}$$

of $\mathcal{D}$, and a set $\mathcal{I}^{\nabla}$ of partial Herbrand interpretations of $\mathcal{D}^{\mathsf{cd}}$ is transformed to the set $\mathcal{I}^{\nabla^{\otimes}} = \{ I^{\nabla^{\otimes}} \mid I^{\nabla} \in \mathcal{I}^{\nabla} \}$ of clausal Herbrand interpretations of $\mathcal{D}$. Note, that the difference between undefined and false atoms $dis(\alpha) \in HB_{\mathcal{D}^{\mathsf{cd}}}$ is ignored, which is due to the fact that Definition 9 ignores the part I_u of the solution $\langle I_t, I_u \rangle$ to the system $\Gamma_{\mathcal{C}^{\mathsf{t}}}(\Gamma_{\mathcal{C}^{\mathsf{u}}}(I_t)) = I_t$ and $\Gamma_{\mathcal{C}^{\mathsf{u}}}(I_t) = I_u$ of conditions.

The following theorem, which is a consequence of the construction of $\mathcal{D}^{\mathsf{cd,tu}}$, states that the semantics of stable states for dd–databases $\mathcal{D}$ can be characterized based on the semantics of stable models for normal deductive databases $\mathcal{D}^{\mathsf{cd,tu}}$.

Theorem 15 (Characterization of Stable States).
Given a disjunctive deductive database $\mathcal{D}$, then

$$\mathcal{STABLE}_{\mathsf{cd}}(\mathcal{D}) = \mathcal{STABLE}_2(\mathcal{D}^{\mathsf{cd,tu}})^{\nabla^{\otimes}}.$$

6. Hypothetical Reasoning

A *hypothetical logic program* $\mathcal{D}$ over a first–order language $\mathcal{L}$ is a set of hypothetical inference rules, where hypothetical inference rules are defined inductively as follows:

1. A normal logic rule r over $\mathcal{L}$ is also a hypothetical logic rule over $\mathcal{L}$.
2. If $\mathcal{D}_i$, $i \in \langle 1, n+m \rangle$, are hypothetical logic programs and A_i, B_i, C_i, are atoms over $\mathcal{L}$, then

$$\begin{aligned} r = A \leftarrow (B_1 \Leftarrow \mathcal{D}_1) \wedge \ldots \wedge (B_m \Leftarrow \mathcal{D}_m) \wedge \\ \mathit{not}\,(C_1 \Leftarrow \mathcal{D}_{m+1}) \wedge \ldots \wedge \mathit{not}\,(C_n \Leftarrow \mathcal{D}_{m+n}), \end{aligned} \tag{3}$$

   is a hypothetical logic rule over $\mathcal{L}$.

The ground instance $\mathit{gnd}\,(\mathcal{D})$ of a hypothetical logic program $\mathcal{D}$ is obtained by replacing all variable symbols in the rules by ground terms from the underlying language $\mathcal{L}$. The intuitive meaning of a ground rule $r \in \mathit{gnd}\,(\mathcal{D})$ of the form (3) is that A should be derivable from $\mathcal{D}$ if all atoms B_i are derivable from $\mathcal{D} \cup \mathcal{D}_i$ and none of the atoms C_i is derivable from $\mathcal{D} \cup \mathcal{D}_{m+i}$. If $\mathcal{D}_i = \emptyset$, for all $i \in \langle 1, n+m \rangle$, then r can be perceived as a normal rule.

Embedded implications $A \Leftarrow \mathcal{D}'$ provide extra knowledge $\mathcal{D}'$ for proving individual atoms A: this knowledge can only be used locally for A; it is *nested* within the overall program $\mathcal{D}$ and not a *subset* of $\mathcal{D}$. In [3,17], hypothetical logic rules are considered under the restriction that all hypotheses $\mathcal{D}'$ must be normal logic programs rather than hypothetical logic programs. Miller [13] conducted a logical analysis of modules in logic programming, and he showed that hypothetical logic programming can provide a logical explanation of parametric modules and abstract data types.

If A is an atom and $\mathcal{D}'$ is a hypothetical logic program, then $A \Leftarrow \mathcal{D}'$ is called a hypothetical atom and $\mathcal{D}'$ is called its hypothesis. If $\mathcal{D}' = \emptyset$, then we can simply write A instead of $A \Leftarrow \mathcal{D}'$. In a hypothetical atom, the embedded implication sign "$\Leftarrow$" can be perceived as the predicate symbol, and A and $\mathcal{D}'$ are *reifyied* and perceived as the arguments of "$\Leftarrow$".

Two Examples

A nice example for hypothetical logic programs was given by Dung [3]: The *British Nationality Act* rules that *a child born in the Kingdom after the com-*

mencement can become a British citizen if his father, who is dead at the time of his birth, would have been entitled to register as a British citizen, had he been alive at that time. This rule can be formalized as the following hypothetical rule:

$$\begin{aligned} &\mathit{become\_citizen}(X) \leftarrow \mathit{born\_in\_UK}(X) \wedge \\ &\qquad (\mathit{become\_citizen}(\mathit{father}(X)) \Leftarrow \{\mathit{alive}(\mathit{father}(X))\}). \end{aligned}$$

The following example $\mathcal{D}_{even}$ is another hypothetical logic program that is well–known from literature [3,17]:

$$\begin{aligned} r_1 &= \mathit{even} \leftarrow \mathit{not}\ \mathit{odd}, \\ r_2 &= \mathit{odd} \leftarrow \mathit{select}(X) \wedge (\mathit{even} \Leftarrow \{\mathit{mark}(X)\}), \\ r_3 &= \mathit{select}(X) \leftarrow r(X) \wedge \mathit{not}\ \mathit{mark}(X), \\ e_i &= r(i),\ i \in \langle 1, n \rangle. \end{aligned}$$

The meaning of $\mathcal{D}_{even}$ – which will be explained later – is that *even* is derivable if the number n of atoms $r(i)$ in $\mathcal{D}_{even}$ is even.

Hypothetical Interpretations

In [3,17], a hypothetical interpretation I is defined as a mapping from the set $\mathcal{W}$ of all hypothetical logic programs to the set of all (partial) Herbrand interpretations, i.e., a hypothetical logic program $\mathcal{D}$ is mapped to another mapping $I(\mathcal{D})$ on atoms, such that $I(\mathcal{D})(A) \in \{\mathsf{t}, \mathsf{u}, \mathsf{f}\}$ for all atoms A. In [3] only the set $\mathcal{W}_\mathcal{D}$ of all hypothetical logic programs that are relevant to $\mathcal{D}$ is considered: $\mathcal{D} \in \mathcal{W}_\mathcal{D}$, and if $\mathcal{D}' \in \mathcal{W}_\mathcal{D}$ and there exists a ground instance[13] $r \in \mathit{gnd}(\mathcal{D})$ of the form (3), then $\mathcal{D}' \cup \mathcal{D}_i \in \mathcal{W}_\mathcal{D}$ for all $i \in \langle 1, n+m \rangle$.

In this paper we will perceive a hypothetical interpretation I simply as a mapping $I^\odot$ on hypothetical atoms:

$$I^\odot(A \Leftarrow \mathcal{D}') = I(\mathcal{D}')(A).$$

We will limit our attention to total hypothetical interpretations $I^\odot$, which can be represented by the set of all hypothetical ground atoms $A \Leftarrow \mathcal{D}'$ which are satisfied by $I^\odot$, i.e., $I^\odot(A \Leftarrow \mathcal{D}') = \mathsf{t}$. Note that $I^\odot$ must satisfy all hypothetical ground atoms $A \Leftarrow \mathcal{D}'$, such that $A \in \mathit{gnd}(\mathcal{D}')$. Note also that embedded implications

[13] The original definition of [3] seems to assume ground rules implicitely. Otherwise, for the program $\mathcal{D}_{even}$ we would get $\mathcal{W}_\mathcal{D} = \{\mathcal{D}, \mathcal{D} \cup \{\mathit{mark}(X)\}\}$, and the desired semantics of $\mathcal{D}_{even}$ could not be obtained.

have got a nonmonotonic behaviour due to the presence of default negation; i.e., $I^{\odot}(A \Leftarrow \mathcal{D}') = \mathsf{t}$ does not imply $I^{\odot}(A \Leftarrow \mathcal{D}'') = \mathsf{t}$ for $\mathcal{D}' \subseteq \mathcal{D}''$.

E.g., for $\mathcal{D}_{even}$ and $n = 2$ we get $\mathcal{W}_{\mathcal{D}} = \{\, \mathcal{D}_M \mid M \subseteq \{1, 2\} \,\}$, where $\mathcal{D}_M = \mathcal{D}_{even} \cup \{\, mark(i) \mid i \in M \,\}$. The following set $I^{\odot}_{even}$ defines a total hypothetical interpretation of $\mathcal{D}_{even}$:

$$\begin{aligned} I^{\odot}_{even} \;=\; \{\, & even \Leftarrow \mathcal{D}_{\{\}},\; odd \Leftarrow \mathcal{D}_{\{1\}},\; odd \Leftarrow \mathcal{D}_{\{2\}},\; even \Leftarrow \mathcal{D}_{\{1,2\}}, \\ & select(1) \Leftarrow \mathcal{D}_{\{\}},\; select(2) \Leftarrow \mathcal{D}_{\{\}}, \\ & select(2) \Leftarrow \mathcal{D}_{\{1\}},\; mark(1) \Leftarrow \mathcal{D}_{\{1\}}, \\ & mark(2) \Leftarrow \mathcal{D}_{\{2\}},\; select(1) \Leftarrow \mathcal{D}_{\{2\}}, \\ & mark(1) \Leftarrow \mathcal{D}_{\{1,2\}},\; mark(2) \Leftarrow \mathcal{D}_{\{1,2\}} \,\} \cup \\ \{\, & r(i) \Leftarrow \mathcal{D}_M \mid i \in \{1,2\},\; M \subseteq \{1,2\} \,\}. \end{aligned}$$

Later we will see that $I^{\odot}$ is the unique stable model of a suitable program transformation $\mathcal{D}^{\odot}_{even}$ of $\mathcal{D}_{even}$. The atoms A that are derivable from $\mathcal{D}_{even}$ are given by the hypothetical atoms $(A \Leftarrow \mathcal{D}_{even}) \in I^{\odot}_{even}$. Since $\mathcal{D}_{\{\}} = \mathcal{D}_{even}$, these atoms are *even*, *select*(1), *select*(2), $r(1)$, and $r(2)$. I.e., it is correctly computed that the number $n = 2$ of atoms $r(i) \in \mathcal{D}_{even}$ is even.

Transformation and Generalization

We transform a hypothetical rule of the form (3) to a normal rule $r^{\odot}$ over hypothetical atoms as follows:

$$\begin{aligned} r^{\odot} = (A \Leftarrow X) \leftarrow & (B_1 \Leftarrow \mathcal{D}_1 \cup X) \wedge \ldots \wedge (B_m \Leftarrow \mathcal{D}_m \cup X) \wedge \\ & not\,(C_1 \Leftarrow \mathcal{D}_{m+1} \cup X) \wedge \ldots \wedge not\,(C_n \Leftarrow \mathcal{D}_{m+n} \cup X). \end{aligned} \tag{4}$$

Here X is a new variable symbol that can be instantiated by an arbitrary hypothetical logic program. Note that for all hypothetical atoms in the body of $r^{\odot}$ the instantiation of $\mathcal{D}_i \cup X$ is a superset of the instantiation of the variable symbol X in the head atom. The transformation of a hypothetical logic program $\mathcal{D}$ is given by $\mathcal{D}^{\odot} = \{\, r^{\odot} \mid r \in \mathcal{D} \,\} \cup \{\, A \Leftarrow \mathcal{D}' \mid \mathcal{D}' \in \mathcal{W}_{\mathcal{D}},\; A \in \mathcal{D}' \cap \mathcal{L} \,\}$.

One can think of generalizing the type of rules given in (4) by allowing for more general hypotheses in the body, as well as disjunctions of hypothetical atoms in the head:

$$\begin{aligned} r^{\bullet} = (A_1 \Leftarrow T^A_1) \vee \ldots \vee (A_k \Leftarrow T^A_k) \leftarrow & \\ (B_1 \Leftarrow T^B_1) \wedge \ldots \wedge (B_m \Leftarrow T^B_m) \wedge & \end{aligned}$$

$$not\,(C_1 \Leftarrow T_1^C) \wedge \ldots \wedge not\,(C_n \Leftarrow T_n^C). \qquad (5)$$

Here T_i^A, T_i^B, and T_i^C are terms over variable symbols (that can be instantiated by arbitrary hypothetical logic programs), hypothetical logic programs, and the operator set union "$\cup$" (even the operators set intersection "$\cap$" and set difference "$\setminus$" could be allowed in these terms).

A *hypothetical disjunctive logic program* $\mathcal{D}$ is given by a set of rules of the form (5). This generalizes the approaches of [3,17] in two ways. Firstly, we allow for repeated nesting of embedded implications, since our hypotheses T_i^A, T_i^B, and T_i^C contain hypothetical logic programs themselves. Secondly, we allow for disjunctive rule heads.

Semantics

The *semantics* of a hypothetical logic program $\mathcal{D}$ can be defined based on its transfromation $\mathcal{D}^{\odot}$. In principle, any semantics that is known for normal logic programs can be applied to $\mathcal{D}^{\odot}$. In the following we will consider the semantics of stable models. If $I^{\odot}$ is a stable model of $\mathcal{D}^{\odot}$, then $I = \{\, A \mid (A \Leftarrow \mathcal{D}) \in I^{\odot} \,\}$ is called a *hypothetical stable model* of $\mathcal{D}$. This gives a simpler and more general characterization of hypothetical stable models than the one given in [3]. In [17] other semantics for hypothetical logic programs are introduced.

E.g., it can be shown that $I^{\odot}_{even}$ is the unique stable model of $\mathcal{D}^{\odot}_{even}$; the *Gelfond–Lifschitz transformation* $(\mathcal{D}^{\odot}_{even})^{I^{\odot}_{even}}$ is given in Appendix B. Thus,

$$I_{even} = \{\, even,\ select(1),\ select(2),\ r(1),\ r(2) \,\}$$

is the unique hypothetical stable model of $\mathcal{D}_{even}$.

The semantics of a hypothetical disjunctive logic program $\mathcal{D}^{\odot}$ can be defined equivalently: If $I^{\odot}$ is a stable model of $\mathcal{D}^{\odot}$, then $I = \{\, A \mid (A \Leftarrow \mathcal{D}) \in I^{\odot} \,\}$ is called a *hypothetical stable model* of $\mathcal{D}^{\odot}$.

The Generic $\mathcal{T}_{\mathcal{D}}$–Operator

Hypothetical logic programs $\mathcal{D}$ without default negation are another example where the concept of the generic $\mathcal{T}_{\mathcal{D}}$–operator $\mathcal{T}_{\mathcal{D}}^{\odot,\geq}$, which was introduced in Section 4.1, can be applied. Here the generic $\mathcal{T}_{\mathcal{D}}$–operator can be defined based on $\mathcal{D}^{\odot}$ and $exp_{\geq}(I^{\odot}) = I^{\odot}$ for hypothetical interpretations. Thus, $\mathcal{T}_{\mathcal{D}}^{\odot,\geq}$ reduces to the classical definite consequence operator of van Emden and Kowalski: $\mathcal{T}_{\mathcal{D}}^{\odot,\geq}(I^{\odot}) = \mathcal{T}_{\mathcal{D}^{\odot}}(I^{\odot})$.

For hypothetical logic programs $\mathcal{D}$ with default negation a hypothetical interpretation I of the form $I = \{ A \mid (A \Leftarrow \mathcal{D}) \in I^{\odot} \}$ is a hypothetical stable model of $\mathcal{D}$ if $I^{\odot} = \mathit{lfp}\,(\mathcal{T}_{(\mathcal{D}^{\odot})^{I^{\odot}}})$, i.e., $I^{\odot}$ is the least fixpoint of $\mathcal{T}_{(\mathcal{D}^{\odot})^{I^{\odot}}}$.

7. Conclusions

Extended–disjunctive deductive databases allow literals to occur in all places where atoms are possible in disjunctive deductive databases cf. [15]. Marek and Truszczyński have introduced *clausal logic programming*, cf. [12], as a further extension: clausal logic programs allow for general disjunctions – rather than atoms – in rule bodies, next to general disjunctions in rule heads. But clausal logic programs do not allow for variable symbols that stand for general disjunctions in their meta–language $\mathcal{L}^{\mathsf{cd}}$, and the ideas of transforming disjunctive logic programs to clausal logic programs or relating the semantics of disjunctive logic programs and corresponding clausal logic programs have not been investigated.

In the future we will consider the complexity issue of stable state semantics, and we will work on an *efficient implementation*. So far we have used the system Smodels developed by Niemelä [16] for computing well–founded and stable model semantics for normal deductive databases. Smodels was applied to the (cd, tu)–transformation $\mathcal{D}^{\mathsf{cd,tu}}$. It uses a *grounding procedure*, which does not work for databases with function symbols like $\mathcal{D}^{\mathsf{cd,tu}}$. Thus, we have written a new grounding procedure for computing the ground instance $\mathit{gnd}\,(\mathcal{D}^{\mathsf{cd,tu}})$, if the Herbrand base $\mathit{HB}_{\mathcal{D}}$ of $\mathcal{D}$ is finite. For a more efficient implementation, the goal is to *back transform* the declarative approach that is working with cd–databases into a procedure that is directly working with dd–databases. This would allow us to use systems that can handle disjunctions directly, e.g., the dlv–system of Eiter, Leone et al. [4], or our system DisLog [25] for reasoning in disjunctive deductive databases.

We are planning to investigate the new semantics w.r.t. further desirable properties, such as *modularity* or *independence under partial evaluation (GPPE)*, and compare it with some more semantics. In another paper we will show how reasoning in *probabilistic deductive databases* [28] can be formulated easily in terms of a suitable generic consequence operator $\mathcal{T}_{\mathcal{D}}^{\odot,\geq}$, which gives immediate results about the continuity of the underlying consequence operators without requiring extra proofs.

Acknowledgements

The author would like to thank the organizers and all the participants of the International Symposium on Foundations of Information and Knowledge Systems (FoIKS'2000) for the stimulating discussions at the conference, and Jack Minker and Rouven Lepenis for their helpful comments on earlier versions of this paper.

References

[1] *S. Brass, J. Dix:* A Disjunctive Semantics Based on Unfolding and Bottom–Up Evaluation, GI–Jahrestagung / IFIP World Computer Congress 1994: Fachgespräch "Disjunktive logische Programmierung und disjunktive Datenbanken", Springer, 1994, pp. 83–91.

[2] *S. Ceri, G. Gottlob, L. Tanca:* Logic Programming and Databases, Springer, 1990.

[3] *P.M. Dung*: Declarative Semantics of Hypothetical Logic Programming with Negation as Failure, Proc. Workshop on Extensions of Logic Programming 1992 (ELP'92), 1992, pp. 45–58.

[4] *T. Eiter, N. Leone, C. Mateis, G. Pfeifer, F. Scarcello:* A Deductive System for Non–Monotonic Reasoning, Proc. Fourth Intl. Conf. on Logic Programming and Non–Monotonic Reasoning 1997 (LPNMR'97), Springer LNAI 1265, 1997, pp. 363–374.

[5] *T. Eiter, N. Leone, D. Sacca:* On the Partial Semantics for Disjunctive Deductive Databases, Annals of Mathematics and Artificial Intelligence, vol. 17(1/2), 1997, pp. 59–96.

[6] *J.A. Fernández, J. Lobo, J. Minker, V.S. Subrahmanian:* Disjunctive LP + Integrity Constrains = Stable Model Semantics, Annals of Math. and AI, vol. 8 (3–4), 1993, pp. 449–474.

[7] *J.A. Fernández and J. Minker:* Bottom–up computation of perfect models for disjunctive theories, Journal of Logic Programming, vol. 25 (1), 1995, pp. 33–51.

[8] *M. Gelfond, V. Lifschitz:* The Stable Model Semantics for Logic Programming, Proc. Fifth Intl. Conference and Symposium on Logic Programming 1988 (ICSLP'88), MIT Press, 1988, pp. 1070–1080.

[9] *G. Köstler, W. Kießling, H. Thöne, U. Güntzer:* The differential fixpoint operator with subsumption, Proc. Intl. Conference on Deductive and Object–Oriented Databases 1993 (DOOD'93), Springer LNCS 760, 1993, pp. 35–48.

[10] *J.W. Lloyd:* Foundations of Logic Programming, second edition, Springer, 1987.

[11] *J. Lobo, J. Minker, A. Rajasekar:* Foundations of Disjunctive Logic Programming, MIT Press, 1992.

[12] *V.W. Marek, M. Truszczyński:* Nonmonotonic Logic – Context–Dependent Reasoning, Springer, 1993.

[13] *D. Miller*: Logical Analysis of Modules in Logic Programming, Journal of Logic Programming, 1989, pp. 79–108.

[14] *J. Minker, A. Rajasekar*: A Fixpoint Semantics for Disjunctive Logic Programs, Journal of Logic Programming, vol. 9(1), 1990, pp. 45–74.

[15] *J. Minker, C. Ruiz:* On Extended Disjunctive Logic Programs, Proc. Intl. Symposium on Methodologies for Intelligent Systems 1993 (ISMIS'93), Springer LNAI 689, 1993, pp. 1–18.

[16] *I. Niemelä, P. Simons:* Efficient Implementation of the Well–founded and Stable Model Semantics, Technical Report 7/96, Univ. Koblenz – Landau, 1996.

[17] *L. Giordano, N. Olivetti*: Combining Negation–as–Failure and Embedded Implications in Logic Programs, Journal of Logic Programming, vol. 36, 1998, pp. 91–147.

[18] *T.C. Przymusinski:* Stable Semantics for Disjunctive Programs, New Generation Computing, vol. 9, 1991, pp. 401–424.

[19] *R. Ramakrishnan:* Database Management Systems, McGraw–Hill, 1998.

[20] *R. Reiter:* A Logic for Default Reasoning, Artificial Intelligence, vol. 13, 1980, pp. 81–132.

[21] *K. Sagonas, T. Swift, D. Warren:* XSB as an Efficient Deductive Database Engine, Proc. ACM SIGMOD Intl. Conf. on the Management of Data 1994 (SIGMOD'94), pp. 442–453.

[22] *D. Seipel, J. Minker, C. Ruiz*: Model Generation and State Generation for Disjunctive Logic Programs, Journal of Logic Programming, vol. 32(1), 1997, pp. 48–69.

[23] *D. Seipel, J. Minker, C. Ruiz*: A Characterization of Partial Stable Models for Disjunctive Deductive Databases, Proc. Intl. Logic Programming Symposium 1997 (ILPS'97), MIT Press, 1997, pp. 245–259.

[24] *D. Seipel*: Partial Evidential Stable Models For Disjunctive Databases, Proc. Workshop on Logic Programming and Knowledge Representation (LPKR'97) at the International Symposium on Logic Programming 1997 (ILPS'97), Springer LNAI 1471, 1998, pp. 66–84.

[25] *D. Seipel:* DisLog – A Disjunctive Deductive Database Prototype, Proc. Twelfth Workshop on Logic Programming (WLP'97), 1997, pp. 136–143. DisLog is available at "http://www-info1.informatik.uni-wuerzburg.de/databases/DisLog".

[26] *D. Seipel:* Clausal Deductive Databases and a General Framework for Semantics in Disjunctive Deductive Databases. Proc. International Symposium on Foundations of Information and Knowledge Systems (FoIKS'2000), Springer LNCS 1762, 2000, pp. 241–259.

[27] *A. Van Gelder, K.A. Ross, J.S. Schlipf:*, Unfounded Sets and Well–Founded Semantics for General Logic Programs, Proc. Seventh ACM Symposium on Principles of Database Systems 1988 (PODS'88), pp. 221–230.

[28] *C. Zaniolo, S. Ceri, C. Faloutsos, R.T. Snodgrass, V.S. Subrahmanian, R. Zicari:* Advanced Database Systems, Morgan Kaufmann, 1997.

[29] *U. Zukowski, S. Brass, B. Freitag:* Improving the Alternating Fixpoint: The Transformation Approach, Proc. 4th Intl. Conf. on Logic Programming an Non–Monotonic Reasoning 1997 (LPNMR'97), Springer LNAI 1265, 1997, pp. 40–59.

Appendix A

Theorem 3 (Properties of the Generic $\mathcal{T}_{\mathcal{D}}$–Operator)
The operator $\mathcal{T}_{\mathcal{D}}^{\odot,\geq} : 2^{\mathcal{X}} \to 2^{\mathcal{X}}$ is *monotonic* and *continuous* on $\langle 2^{\mathcal{X}}, \subseteq \rangle$.

Proof.

1. Firstly, we prove *monotonicity*:

$$X_1 \subseteq X_2 \;\Rightarrow\; exp_{\geq}(X_1) \subseteq exp_{\geq}(X_2) \;\Rightarrow\; \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(X_1) \subseteq \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(X_2).$$

2. Secondly, for proving *continuity*, consider a directed set $\mathcal{Y} \subseteq 2^{\mathcal{X}}$ of subsets of $\mathcal{X}$. Let $X = \cup_{Y \in \mathcal{Y}} Y$. Then $\cup_{Y \in \mathcal{Y}} exp_{\geq}(Y) = exp_{\geq}(X)$. Obviously, it holds $\cup_{Y \in \mathcal{Y}} \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(Y) \subseteq \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(X)$. Conversely, if $\alpha \in \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(X)$, then there must exist a rule $\alpha \leftarrow \beta_1 \wedge \ldots \wedge \beta_m \in gnd\,(\mathcal{D}^{\odot})$, such that $\{\, \beta_1, \ldots, \beta_m \,\} \subseteq exp_{\geq}(X)$. Thus, there must exist sets $Y_{j_1}, \ldots, Y_{j_m} \in \mathcal{Y}$, such that $\beta_i \in exp_{\geq}(Y_{j_i})$. I.e. there exist elements $\beta'_i \in Y_{j_i}$, such that $\beta'_i \geq \beta_i$. Since $\mathcal{Y}$ is directed, there must exist a set $Y_{j_0} \in \mathcal{Y}$, such that $\{\, \beta'_1, \ldots, \beta'_m \,\} \subseteq Y_{j_0}$. Thus, $\{\, \beta_1, \ldots, \beta_m \,\} \subseteq exp_{\geq}(Y_{j_0})$, and $\alpha \in \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(Y_{j_0}) \subseteq \cup_{Y \in \mathcal{Y}} \mathcal{T}_{\mathcal{D}}^{\odot,\geq}(Y)$.

$\Box$

Theorem 6 (Characterization of Stable Models)
Given a disjunctive deductive database $\mathcal{D}$, then it holds

$$\mathcal{STABLE}_2(\mathcal{D}^{\mathsf{cd}})^{\otimes} = \mathcal{STABLE}_2(\mathcal{D}) \cup \mathcal{J}, \quad \text{where} \quad \mathcal{J} = \emptyset \;\text{ or }\; \mathcal{J} = \{\, J^{\top} \,\}.$$

Proof.

1. Consider a total model I of $\mathcal{D}^{\mathsf{cd}}$, such that $I(dis(\Box_d)) = \mathsf{f}$, and consider the corresponding Herbrand interpretation $I^{\otimes}$ of $\mathcal{D}$. It is not necessary to deal with atoms $dis(\alpha)$ for non–singleton negative disjunctions $\alpha \in N\mathit{HB}_{\mathcal{D}}$ during the fixpoint evaluation of $\mathcal{D}^{\mathsf{cd}}$: amoung the rules $(r^{cwa}_{neg})^I$ the ones in $\{\, dis(\neg A) \leftarrow negative(\neg A) \wedge complement(\neg A, A) \mid A \in \mathit{HB}_{\mathcal{D}},\, con(A) \notin I \,\}$ for minimal, i.e. atomary, disjunctions "$\neg A$" are *subsuming* the others. The resolution rule r^{logic}_{res} will consume the atoms $dis(\neg A)$ and produce atoms $dis(\alpha)$ over positive disjunctions $\alpha \in D\mathit{HB}_{\mathcal{D}}$. The evaluation is equivalent to evaluating the disjunctive deductive database $\mathcal{D}^{I^{\otimes}}$ together with the set $\mathcal{D}' = \{\, \leftarrow A \mid A \in \mathit{HB}_{\mathcal{D}},\, con(A) \notin I \,\}$ of denial rules, using the disjunctive consequence operator $\mathcal{T}^{s}_{\mathcal{D}''}$ for $\mathcal{D}'' = \mathcal{D}^{I^{\otimes}} \cup \mathcal{D}'$.

2. $I \neq (J^{\top})^{\odot}$ is a stable model of $\mathcal{D}^{\mathsf{cd}}$ iff it holds $can_{\trianglerighteq}(\mathcal{T}^{s}_{\mathcal{D}''} \uparrow \omega) = \{\, A \in H\!B_{\mathcal{D}} \,|\, I^{\otimes}(A) = \mathsf{t} \,\}$, i.e. iff $I^{\otimes}$ is a minimal model $\mathcal{D}^{I^{\otimes}}$, i.e. iff $I^{\otimes} \in \mathcal{STABLE}_{\mathit{2}}(\mathcal{D})$.
3. As a consequence of 2. we get $\mathcal{STABLE}_{\mathit{2}}(\mathcal{D}^{\mathsf{cd}})^{\otimes} \subseteq \mathcal{STABLE}_{\mathit{2}}(\mathcal{D}) \cup \{\, J^{\top} \,\}$. Every stable model $J \in \mathcal{STABLE}_{\mathit{2}}(\mathcal{D})$ can be perceived as $J = I^{\otimes}$ for the total Herbrand interpretation $I = J^{\odot}$ of $\mathcal{D}^{\mathsf{cd}}$. Thus, according to 2. we get $J \in \mathcal{STABLE}_{\mathit{2}}(\mathcal{D}^{\mathsf{cd}})^{\otimes}$.

□

Appendix B

In the following we will consider the hypothetical logic program $\mathcal{D}_{even}$ that was introduced in Section 6. Recall that $\mathcal{D}_M = \mathcal{D}_{even} \cup \{\, mark(i) \mid i \in M \,\}$, for subsets $M \subseteq \{1,2\}$. The *Gelfond–Lifschitz transformation* $(\mathcal{D}^{\odot}_{even})^{I^{\odot}_{even}}$ is derived by dropping the default–negated literals – which are satisfied in $I^{\odot}_{even}$ and which are displayed only for illustration purposes – in the following rules $g_1, \dots, g_{24}$:

$$g_1 = (even \Leftarrow \mathcal{D}_{\{\}}) \leftarrow not\,(odd \Leftarrow \mathcal{D}_{\{\}}),$$
$$g_2 = (even \Leftarrow \mathcal{D}_{\{1,2\}}) \leftarrow not\,(odd \Leftarrow \mathcal{D}_{\{1,2\}}),$$

$$g_3 = (odd \Leftarrow \mathcal{D}_{\{1\}}) \leftarrow (select(2) \Leftarrow \mathcal{D}_{\{1\}}) \wedge (even \Leftarrow \mathcal{D}_{\{1,2\}}),$$
$$g_4 = (odd \Leftarrow \mathcal{D}_{\{2\}}) \leftarrow (select(1) \Leftarrow \mathcal{D}_{\{2\}}) \wedge (even \Leftarrow \mathcal{D}_{\{1,2\}}),$$

$$g_5 = (select(1) \Leftarrow \mathcal{D}_{\{2\}}) \leftarrow (r(1) \Leftarrow \mathcal{D}_{\{2\}}) \wedge not\,(mark(1) \Leftarrow \mathcal{D}_{\{2\}}),$$
$$g_6 = (select(2) \Leftarrow \mathcal{D}_{\{1\}}) \leftarrow (r(2) \Leftarrow \mathcal{D}_{\{1\}}) \wedge not\,(mark(2) \Leftarrow \mathcal{D}_{\{1\}}),$$
$$g_7 = (select(1) \Leftarrow \mathcal{D}_{\{\}}) \leftarrow (r(1) \Leftarrow \mathcal{D}_{\{\}}) \wedge not\,(mark(1) \Leftarrow \mathcal{D}_{\{\}}),$$
$$g_8 = (select(2) \Leftarrow \mathcal{D}_{\{\}}) \leftarrow (r(2) \Leftarrow \mathcal{D}_{\{\}}) \wedge not\,(mark(2) \Leftarrow \mathcal{D}_{\{\}}),$$

$$g_9 = (mark(1) \Leftarrow \mathcal{D}_{\{1\}}),\ g_{10} = (mark(2) \Leftarrow \mathcal{D}_{\{2\}}),$$
$$g_{11} = (mark(1) \Leftarrow \mathcal{D}_{\{1,2\}}),\ g_{12} = (mark(2) \Leftarrow \mathcal{D}_{\{1,2\}}),$$

$$g_{13} = (r(1) \Leftarrow \mathcal{D}_{\{\}}),\ g_{14} = (r(2) \Leftarrow \mathcal{D}_{\{\}}),$$
$$g_{15} = (r(1) \Leftarrow \mathcal{D}_{\{1\}}),\ g_{16} = (r(2) \Leftarrow \mathcal{D}_{\{1\}}),$$
$$g_{17} = (r(1) \Leftarrow \mathcal{D}_{\{2\}}),\ g_{18} = (r(2) \Leftarrow \mathcal{D}_{\{2\}}),$$
$$g_{19} = (r(1) \Leftarrow \mathcal{D}_{\{1,2\}}),\ g_{20} = (r(2) \Leftarrow \mathcal{D}_{\{1,2\}}),$$

$$g_{21} = (odd \Leftarrow \mathcal{D}_{\{\}}) \leftarrow (select(1) \Leftarrow \mathcal{D}_{\{\}}) \wedge (even \Leftarrow \mathcal{D}_{\{1\}}),$$

$$\begin{aligned}
g_{22} &= (odd \Leftarrow \mathcal{D}_{\{\}}) \leftarrow (select(2) \Leftarrow \mathcal{D}_{\{\}}) \wedge (even \Leftarrow \mathcal{D}_{\{2\}}), \\
g_{23} &= (odd \Leftarrow \mathcal{D}_{\{1,2\}}) \leftarrow (select(1) \Leftarrow \mathcal{D}_{\{1,2\}}) \wedge (even \Leftarrow \mathcal{D}_{\{1,2\}}), \\
g_{24} &= (odd \Leftarrow \mathcal{D}_{\{1,2\}}) \leftarrow (select(2) \Leftarrow \mathcal{D}_{\{1,2\}}) \wedge (even \Leftarrow \mathcal{D}_{\{1,2\}}).
\end{aligned}$$

The following ground rules $g_{25}, \ldots, g_{28}$, are blocked, since their default–negated body literals are not satisfied in $I^{\odot}_{even}$:

$$\begin{aligned}
g_{25} &= (even \Leftarrow \mathcal{D}_{\{1\}}) \leftarrow not\,(odd \Leftarrow \mathcal{D}_{\{1\}}), \\
g_{26} &= (even \Leftarrow \mathcal{D}_{\{2\}}) \leftarrow not\,(odd \Leftarrow \mathcal{D}_{\{2\}}),
\end{aligned}$$

$$\begin{aligned}
g_{27} &= (select(1) \Leftarrow \mathcal{D}_{\{1,2\}}) \leftarrow (r(1) \Leftarrow \mathcal{D}_{\{1,2\}}) \wedge not\,(mark(1) \Leftarrow \mathcal{D}_{\{1,2\}}), \\
g_{28} &= (select(2) \Leftarrow \mathcal{D}_{\{1,2\}}) \leftarrow (r(2) \Leftarrow \mathcal{D}_{\{1,2\}}) \wedge not\,(mark(2) \Leftarrow \mathcal{D}_{\{1,2\}}).
\end{aligned}$$

Thus, the ground rules $g_{21}, \ldots, g_{24}$, cannot fire during fixpoint iteration with $(\mathcal{D}^{\odot}_{even})^{I^{\odot}_{even}}$.