Building Theories into Instantiation

Alan M Frisch"
Department of Computer Science
University of York
York Y01 5DD
United Kingdom

C David Page Jr '
Oxford University Computing Laboratory
Wolfson Building, Parks Road
Oxford 0X1 3QD
United Kingdom

Abstract

Instantiation orderings over formulas (the relation of one formula bemg an instance of another) have long been central to the study of automated deduction and logic programming, and are of rapidly-growing importance in the study of database systems and machine learning A variety of instantiation ordenngs are now IP use, many of which incorporate some kind of background information in the form of a *constraint theory* Even a casual examination of these instantiation orderings reveals that they are somehow related, but in exactly what way? This paper presents a general instantiation ordering of which all these orderings are special cases, as are other instantiation ordenngs The paper shows that this general ordering has the semantic properties we desire in an instantiation ordering, implying that the special cases have these properties as well The extension to this general ordering is useful in applications to inductive logic programming, automated deduction and logic programming, knowledge-base vivification, and database systems

1 Introduction

Instantiation orderings over formulas (the relation of one formula bemg an instance of another) have long been central to the study of automated deduction and logic programming, and are of rapidly-growmg importance to the study of database systems and machine learning (e g , in inductive logic programming) One common way—perhaps the *most* common way-to build a theory of background information into a computational system based on instantiation is to generalize the ordinary definition of instantiation to take account of the theory The earliest work of this kind was Plotkin's [1972] method of building an equational theory into resolution merely by replacing unification with the more general operation of unification with respect to the theory, an operation he called T-unification" and is today called "E-unification " E-unification can be thought of as an operation used to obtain the greatest lower bound in an instantiation ordering that is relative to the built-in equational theory Sorted Jogics can be thought of as logics that incorporate background information about sorts, which are sets of objects in the domain All automated deduction systems for sorted logic (e g , [Walther, 1987, Conn, 1987, Schmidt-Schauss, 1989, Frisch, 1991]) perform sorted unification, which, once again, can be thought of as an operation used to obtain the greatest lower bound in an instantiation ordering with a built-in theory about sorts

This paper presents an instantiation ordering for con straint logic, a logic that generalizes previous logics that incorporate background information, and proves that previous instantiation ordenngs are special cases of this ordering Furthermore, Section 5 shows the utiliy of this generalized instantiation ordering by discussing applications to (1) induction and inductive logic programming, (2) deduction and logic programming, and (3) knowledge-base vivification We are currently investigating applications to constraint databases and database query languages as well The extension to constraint logic is vital for each of these applications

After defining our generalized instantiation ordering, we identify two semantic properties that one should expect of instantiation orderings, properties that make instantiation orderings useful Theorem 2 establishes that our general instantiation ordering has these properties, consequently (because of the nature of these properties) all special cases of the general ordering must have these properties

A major consequence of defining instantiation in such a general setting is that the traditional substitution-based definitions are inadequate The definition of instantiation presented in this paper (Section 3) does not involve substitutions, it is primarily semantic Two of the principle theorems of this paper, Theorems 3 and 6, each identify conditions under which an instantiation ordering can be based on substitution Examples are presented that show that substitution-based definitions cannot be guaranteed to work when these condi-

"Emall frisch@minster york ac uk World-Wide Web http //dcpul cs york ac uk 9876/ug/frisch/home html

'Email dpageOcomlab ox ac uk Supported by Esprit Basic Research Action 6020 on Inductive Logic Programming

tions are not met Thus, this paper demonstrates that substitution-based definitions, such as that embodied in E-unification, are not applicable in all generalized settings, and the paper identifies some of the limits of applicability For example, we see that the instantiation ordering for sorted logic can be generalized from sorts to arbitrary predicates provided that there are no built-in equations However, if there are built-in equations, then additional conditions, which are identified in the paper, must be met

By showing that sorted logic meets the conditions of Theorem 3, and that logic with built-in equations meets the conditions of Theorem 6, we are able to prove that the instantiation orderings associated with these two logics are special cases of our general instantiation ordering These proofs are simple, suggesting that it also may be simple to prove that other instantiation ordenngs are special cases of the general ordering Other properties, such as the existence of least upper bounds and greatest lower bounds and the finiteness of antichains, ascending chains, ard descending chains, hold only in some instantiation ordenngs Elsewhere sufficient conditions for obtaining these algebraic properties are presented [Page, 1993]

2 Constrained Formulas and Constraint Theories

A *constrained formula* is composed of two ordinary formulae, one called the *head* and the other called the *constraint* The constraint can be any formula whose predicates are drawn from a distinguished set of predicates called *constraint predicates* In addition, *TRUE* is a constraint and has the obvious interpretation We stipulate that the interpreted equality predicate ("=") is a constraint predicate For clarity, constraint predicates (except equality) are written in email capital letters, e g "ELEPHANT " The head of a constrained formula can be any formula that contains no constraint predicates We require that every variable that has a free occurrence in the constraint also has a free occurrence in the head Constraint formulas are written m the form ϕ/C, where ϕ is the head and C is the constraint

But what does a constrained formula *mean*? In all the applications of constrained formulas that we know about, the variables in the formulas are either all universally quantified or all existentially quantified [1] Where ϕ is any formula (ordinary or constrained), we say that the *universal closure* of ϕ is the result of universally quantifying all free variables in ϕ, and we denote the universal closure of ϕ by $\bar{\forall}\phi$ Similarly, the *existential closure* of ϕ is the result of existentially quantifying all free variables in ϕ, and it is denoted by $\bar{\exists}\phi$ If ϕ/C is a constrained formula, then we define $\bar{\forall}(\phi/C)$ to be logically equivalent to $\bar{\forall}(C \to \phi)$, and $\bar{\exists}(\phi/C)$ to be logically equivalent to $\bar{\exists}(C \land \phi)$

Operations on constrained formulae act with respect to background information about the constraint predicates This background information takes the form of a

[1] In fact, substantia] difficulties confront attempts to mix the quantifiers on constrained formulas

$\Sigma_1 = \{\forall x$ UNIV(x), ELEPHANT$(clyde) \lor$ IN-CIRCUS$(clyde)$,
ELEPHANT$(jumbo) \lor$ IN-CIRCUS$(jumbo)$,
GRAY$(mom(clyde))$, GRAY$(mom(jumbo))$,
$\forall x$ ELEPHANT$(x) \to$ ELEPHANT$(mom(x))$,
$\forall x$ ELEPHANT$(x) \to$ MAMMAL$(x)\}$

$\Sigma_2 = \{party(arnold) = republican$,
$spouse(maria) = arnold$, $spouse(arnold) = maria$,
BIGGER$(arnold, maria)$, KENNEDY$(maria)$,
$\forall x$ KENNEDY$(x) \to (party(x) = democratic)$,
LEGISLATURE$(us\text{-}house)$, LEGISLATURE$(us\text{-}senate)\}$

$\Sigma_3 = \{\forall x \forall y\, (f(x,y) = f(y,x)),\ (c = a) \lor (c = b)\}$

Figure 1 Three Constraint Theories

constraint theory A constraint theory is any (possibly infinite) satisfiable set of sentences all of whose predicates are constraint predicates Figure 1 gives three examples of constraint theories, called Σ_1, Σ_2 and Σ_3 Notice that Σ_1 contains monadic constraint predicates only Therefore, we also call it a *sort theory* [2]

The following definitions are used throughout the paper A *value assignment* is an assignment of individuals from the domain of a given model to all free variables Let Σ be a constraint theory, and let M be a model We say that M is a *Σ-model* if and only if M satisfies Σ We say that a constraint that is satisfied by some Σ-model and some value assignment is *Σ-satisfiable* Where Σ is, more generally, any (possibly infinite) first-order theory, and ψ_1 and ψ_2 are logical sentences, we say that ψ_1 *Σ-entails* ψ_2 if and only if $\Sigma \cup \{\psi_1\} \models \psi_2$, and we write $\psi_1 \models_\Sigma \psi_2$ More generally, where ϕ_1 and ϕ_2 are logical formulas, we say that ϕ_1 Σ-entails ϕ_2 if and only if any Σ-model and value assignment that satisfy ϕ_1 also satisfy ϕ_2 A *quasi-ordering*, or *preorder*, is a relation that is reflexive and transitive Alternatively, a quasi-ordering may be thought of as a relation that partitions a set into equivalence classes and partially orders those equivalence classes It is straightforward to verify that $\models_\Sigma$ is a quasi-ordering on constraints

A constrained formula is *admissible* with respect to constraint theory Σ, or *Σ-admissible*, if its constraint is Σ-satisfiable Otherwise, it is *Σ-inadmissible* The remainder of the paper considers only constrained formulas that are admissible with respect to the constraint theory under consideration Thus "constrained formula" always means "Σ-admissible constrained formula "

Let ϕ be any formula with n *top-level term occurrences*, that is, n occurrences of terms as arguments to predicates Number the occurrences from 1 to n in left-to-right order, as they appear in ϕ We use $\phi[t_1, \ldots, t_n]$ to denote a formula whose ith top-level term occurrence is t_i, for all $1 \leq i \leq n$ Subsequently we use $\phi[t'_1, \ldots, t'_n]$ to denote the formula that results from replacing the ith top-level term occurrence in $\phi[t_1, \ldots, t_n]$ with t'_i, for all $1 \leq i \leq n$ For example, if $\phi[f(a), f(a), g(c,x)]$ denotes the formula $p(f(a)) \land q(f(a), g(c,x))$ then $\phi[g(a,b), c, f(x)]$ denotes

[2] Monadic constraint predicates are often referred to as *Boris*

$p(g(a,b)) \wedge q(c, f(x))$

3 The General Instantiation Ordering

The ordinary instantiation ordering on ordinary formulas is particularly useful for three reasons First, if $\phi_1 \geq \phi_2$ then the set of ground instances of ϕ_1 is a superset of the set of ground instances of ϕ_2 Second, if $\phi_1 \geq \phi_2$ then $\overline{\forall}\phi_1 \models \overline{\forall}\phi_2$ and $\overline{\exists}\phi_2 \models \overline{\exists}\phi_1$ Third, for *atomic* formulas (atoms), this second statement is an *if and only if*, that is, the following three statements are equivalent (1) $\phi_1 \geq \phi_2$, (2) $\overline{\forall}\phi_1 \models \overline{\forall}\phi_2$, and (3) $\overline{\exists}\phi_2 \models \overline{\exists}\phi_1$ We seek an instantiation ordering for constrained formulas that has analogous properties, where we replace entailment with Σ-entailment and ground instances with ground "Σ-instances", or ground instances under the new ordering

Definition 1 (Σ-more general) *Let Σ be a constraint theory Let ϕ_1/C_1 and ϕ_2/C_2 be constrained formulas and let $\vec{v}$ be the free variables of ϕ_1 Let ϕ_2'/C_2' be a variant of ϕ_2/C_2 that shares no free variables with ϕ_1 We say that ϕ_1/C_1 is* Σ-more general *than ϕ_2/C_2 (written $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$) if and only if ϕ_1 is of the form $\phi[t_1, \ldots, t_n]$, ϕ_2' is of the form $\phi[t_1', \ldots, t_n']$ and $C_2' \models_\Sigma \exists\vec{v}\, t_1 = t_1' \wedge \ldots \wedge t_n = t_n' \wedge C_1$*

As an example, the constrained formula

$$controls(party(spouse(arnold)), x, 1993)/\text{LEGISLATURE}(x)$$

is Σ_2-more general than the constrained formula

$$controls(democratic, y, 1993)/(y = us\ house)$$

because

$$y = us\ house \models_{\Sigma_2} \exists x\ party(spouse(arnold)) = democratic \wedge x = y \wedge 1993 = 1993 \wedge \text{LEGISLATURE}(x)$$

For another example, observe that $p(f(c,x)) \geq_{\Sigma_3} p(f(a,b))$ because $\models_{\Sigma_3} \exists x\ f(c,x) = f(a,b)$ This example illustrates the difficulty with using a substitution-based definition of instantiation in this general setting For example, under the standard instantiation ordering associated with E-unification $p(f(a,b))$ is not an instance of $p(f(c,x))$, there exists no substitution that maps $f(c,x)$ to a term that is equal to $f(a,b)$ according to the equational theory Σ_3

Like the usual instantiation ordering for formulas of FOPC, the $\geq_\Sigma$ ordering on constrained formulas is not a partial ordering but a *quasi-ordering*, or *preorder*, for any choice of Σ Because $\geq_\Sigma$ is a quasi-ordering, if $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$ then every ground instance of ϕ_2/C_2 is also a ground instance of ϕ_1/C_1 Thus the $\geq_\Sigma$ ordering is analogous to the $\geq$ ordering on ordinary atoms in at least one of the three ways we wanted it to be What about the other two? Theorem 2, which follows, reveals that the ordering $\geq_\Sigma$ for constrained formulas is analogous to the ordering $\geq$ for ordinary formulas in these ways as well

Theorem 2 *Let ϕ_1/C_1 and ϕ_2/C_2 be constrained formulas, and let Σ be a constraint theory If $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$ then $\overline{\forall}(\phi_1/C_1) \models_\Sigma \overline{\forall}(\phi_2/C_2)$ and $\overline{\exists}(\phi_2/C_2) \models_\Sigma \overline{\exists}(\phi_1/C_1)$ Furthermore, if ϕ_1 and ϕ_2 are atoms then the following three statements are equivalent (1) $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$, (2) $\overline{\forall}(\phi_1/C_1) \models_\Sigma \overline{\forall}(\phi_2/C_2)$, (3) $\overline{\exists}(\phi_2/C_2) \models_\Sigma \overline{\exists}(\phi_1/C_1)$* [3]

If we wanted to extend the Σ-more general ordering to include Σ-inadmissible constrained formulas as well, all three of the desired properties of the ordering are retained if we specify that the Σ-inadmissible constrained formulas are the *least* formulas in the ordering In other words, we retain the properties if for any constrained formulas ϕ_1/C_1 and ϕ_2/C_2, if ϕ_2/C_2 is Σ-inadmissible then $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$

4 The Relationship to Substitution-based Orderings

This section shows that other, established instantiation orderings for various restricted classes of constrained formulas and constraint theories are equivalent to the Σ-more general ordering (given the appropriate restrictions) Our analysis is based on Theorem 3 and Theorem 6, which under specified conditions provide alternative, substitution-based characterizations of the $\geq_\Sigma$ ordering Using these theorems it is easy to equate the $\geq_\Sigma$ ordering with various established orderings For brevity, we consider only two of the best known orderings

The following theorem gives a substitution-based characterization of the instantiation ordering when neither the constraints nor the constraint theory contains equality

Theorem 3 *Let ϕ_1/C_1 and ϕ_2/C_2 be constrained formulas and let Σ be a constraint theory, none of which contains the equality predicate Then $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$ if and only if there exists a substitution θ such that $\phi_1\theta = \phi_2$ and $C_2 \models_\Sigma (C_1\theta)$*

The following is an example of this characterization

$$loves(x,y)/\text{MAMMAL}(x) \wedge \text{MAMMAL}(y) \geq_{\Sigma_1} loves(z, mom(z))/\text{ELEPHANT}(z)$$

To see this consider any substitution θ that maps x to z and y to $mom(z)$

This characterization of $\geq_\Sigma$ is in fact the instantiation ordering for a restricted class of equality-free constrained formulas that Page and Frisch [1992] used in their study of *constrained generalization.* Thus this theorem tell us that the ordering used by Page and Frisch is a special case of the ordering defined in this paper

To see why equality is forbidden, consider an example where $\Sigma = \{\forall x \forall y(f(x,y) = f(y,x))\}$ Then $p(f(b,x)) \geq_\Sigma p(f(a,b))$, but not according to the characterization Similarly, consider a case where Σ is empty but equality appears in a constraint $p(a) \geq_\Sigma p(b)/(a = b)$, but not according to the characterization

Taking a hint from the definition of E-unification, it appears that the above examples could be handled if the characterization in the above theorem were to test not

[3] All proofs are omitted from this paper, but appear in a longer, unpublished version of this paper The proofs of Theorems 2, 3 and 6 are long and involved Theorems 9 and 10 follow straightforwardly from Theorems 3 and 6, respectively

for simple equality, but rather for equality according to the theory £ We now formulate a substitution-based characterization motivated by this idea, prove that it is correct under certain conditions, and show by example that this approach cannot always work The characten zation allows equality in the const ram t theory, E, though not in the constrained formulas

We now provide several additional definitions that are needed The definition of *homomorphum* between models, which follows, is standard The definition of *initial model,* taking into account the truth values of predicates, is taken from Goguen and Meseguer [1986]

Definition 4 (Homomorphism between models) *Let M and M' be models with domains D_M and D'_M, respectively Then a function h $D_M \to D_{M'}$ is a homomorphism from M to M' if and only if*

- *for every function symbol f of arity $n \geq 0$, $h([f]^M(d_1, \ldots, d_n)) = [f]^{M'}(h(d_1), \ldots, h(d_n))$ and*
- *for every predicate symbol p of arity $n \geq 0$, if $[p]^M(d_1, \ldots, d_n) = True$ then we also have $[p]^{M'}(h(d_1), \ldots, h(d_n)) = True$,*

where $\langle d_1, \ldots, d_n\rangle$ is any n-tuple of individuals in D_M

Definition 5 (Initial Model) *Let Σ be any theory of first-order logic with equality A Σ-model M is an* initial model *of Σ if and only if there exists a unique homomorphism from M to any other Σ-model*

The reader can confirm that Σ_2 has an initial model, but that neither Σ_1 nor Σ_3 do To see that Σ_1 has no initial model observe that there is no homomorphism in either direction between any Σ_1-model that satisfies ELEPHANT(*jumbo*) and falsifies IN-CIRCUS(*jumbo*) and any Σ_1-model that falsifies ELEPHANT(*jumbo*) and satisfies IN-CIRCUS(*jumbo*)

Theorem 6 *Let Σ be a constraint theory in Skolem Normal Form that has an initial model Let $\phi[t_1, \ldots, t_n]/C_1$ and $\phi[t'_1, \ldots, t'_n]/C_2$ be constrained formulas, where C_1 and C_2 are conjunctions of atoms Then $\phi[t_1, \ldots, t_n]/C_1 \geq_\Sigma \phi[t'_1, \ldots, t'_n]/C_2$ if and only if there exists a substitution θ such that $C_2 \models_\Sigma (C_1\theta)$ and $C_2 \models_\Sigma t_i\theta = t'_i$, $1 \leq i \leq n$* [4]

Let's return to an earlier example to illustrate this characterization Recall the constraint theory Σ_2, and consider the following pair of constrained formulas

$$controls(party(spouse(x)), y, 1993)/\text{BIGGER}(x, maria) \wedge \text{LEGISLATURE}(y)$$

$$controls(party(z), us\text{-}house, 1993)/\text{KENNEDY}(z)$$

In the notation of Theorem 6, t_1 is $party(spouse(x))$, t_2 is y, t_3 is 1993, t'_1 is $party(z)$, t'_2 is *us-house*, and t'_3 is 1993 In addition, C_1 is $(\text{BIGGER}(x, maria) \wedge \text{LEGISLATURE}(y))$ and C_2 is KENNEDY(z) Let θ map x to *arnold* and y to *us-house* Clearly $C_2 \models_{\Sigma_2} (C_1\theta)$, $C_2 \models_{\Sigma_2} (t_2\theta = t'_2)$, and $C_2 \models_{\Sigma_2} (t_3\theta = t'_3)$, these would hold even if C_2 were simply *TRUE* The interesting part of this example is that $C_2 \models_{\Sigma_2} (t_1\theta = t'_1)$, note that both C_2 and θ are involved nontrivially

The following examples show that without the additional restrictions the characterization of Theorem 6 is incomplete The examples are similar, in the first example, the constraint theory does not have an initial model, because of the disjunction, and in the second, the constraint of one constrained formula contains disjunction For the first example, let Σ be

$$\{\text{BABY}(ralph) \vee \text{BOY}(ralph), \forall x \forall y(f(x,y) = f(y,x))\}$$

Then

$$p(f(x,y))/(\text{BABY}(x) \wedge \text{BOY}(y)) \geq_\Sigma p(f(ralph, z))/(\text{BABY}(z) \wedge \text{BOY}(z)),$$

but not according to the characterization For the second example, let Σ be the theory $\{\forall x \forall y(f(x,y) = f(y,x))\}$ Then

$$p(f(x,y))/(\text{BABY}(x) \wedge \text{BOY}(y)) \geq_\Sigma p(f(w,z))/((\text{BABY}(w) \vee \text{BOY}(w)) \wedge \text{BABY}(z) \wedge \text{BOY}(z))$$

but not according to the characterization

We are now prepared to relate the $\geq_\Sigma$ ordering to two established orderings

4 1 The Instantiation Ordering for Sorted Logic

As noted in the Section 1, if the predicates in a constraint theory are monadic, the theory is called a *sort theory* In sorted logic, the constraint theory is a sort theory, and the formulas are *sorted formulas* A *sorted formula* is a constrained formula whose constraint is a conjunction of atoms built from monadic constraint predicates and variables that appear in the head, such that each variable occurs at most once in the constraint Because of these additional restrictions on the constraints of sorted formulas, sorted formulas are often represented in an alternative, *in-line* syntax, in which the constraints are attached directly to the variables Thus, for example, the sorted formula *eats*(x,y)/ELEPHANT(x) $\wedge$ VEGETABLE(y) may be expressed as *eats*$(x$ ELEPHANT$,y$ VEGETABLE$)$

Definition 7 (In-line Representation) *Let $\phi/(\tau_1(x_1) \wedge \ldots \wedge \tau_n(x_n))$ be a sorted formula Then α is the* in-line representation *of ϕ/C if and only if α results from replacing every occurrence of the variable x_i in ϕ with an occurrence of the sorted variable x_i τ_i, for all $1 \leq i \leq n$*

The additional restrictions on sorted formulas make possible another definition of an instantiation ordering on sorted formulas (called S-more general, or $\geq_S$, in Definition 8), in which substitutions are central, this is the established instantiation ordering for sorted formulas Theorem 9 states that this established ordering on sorted formulas is the same as the $\geq_\Sigma$ ordering, provided a small additional assumption is made The assumption is that every sort contains at least one individual, and it is made for the established ordering anyway [Frisch, 1991] The definition of the established ordering for sorted formulas, given below, is based on the alternative syntax for sorted formulas

[4] We found it quite surprising that this last condition ($C_2 \models_\Sigma t_i\theta = t'_i$) had to involve C_2 even though C_2 is nonequational An example illustrating the need for this will be presented

Definition 8 (*S*-**more general**) *Let S be a sort theory such that for every sort τ we have $S \models \exists x\ \tau(x)$ (according to S every sort contains some individual) A substitution θ is well-sorted with respect to S if and only if for any variable $x\ \tau$, $x\ \tau\theta = t$ where $S \models \overline{\forall}\tau(t)$ A sorted formula α_1 is S-more general than another, α_2, if and only if $\alpha_1\theta = \alpha_2$ for some substitution θ that is well-sorted with respect to S If α_1 is S-more general than α_2, we write $\alpha_1 \geq_S \alpha_2$*

It is worth noting that because we require that $S \models \exists x\ \tau(x)$ for each sort τ, every sorted formula is Σ-admissible

Theorem 9 *Let α_1 and α_2 be the in-line representations of sorted formulas ϕ_1/C_1 and ϕ_2/C_2, respectively Let $S = \Sigma$ be a sort theory such that for every sort τ, $S \models \exists x\ \tau(x)$ Then $\phi_1/C_1 \geq_\Sigma \phi_2/C_2$ if and only if $\alpha_1 \geq_S \alpha_2$*

4 2 E-Unification

E-unification is a form of unification that takes into account a set of equations (universally closed atomic formulas formed with the equality predicate symbol), E [5] Following standard definitions,[6] if s and s' are terms, we say that $s =_E s'$ if and only if the pair $(s\ s')$ is a member of the finest congruence on the term algebra containing all pairs $(t\theta\ t'\theta)$, where $\overline{\forall}(t = t')$ is a member of E and θ is a substitution We say that θ is an E-unifier of s and t if and only if $s\theta =_E t\theta$ Associated with this notion of E-unifier, is an instantiation ordering over the set of terms $s \geq_E t$ if and only if $s\theta =_E t$ for some substitution θ We can straightforwardly extend these standard definitions to generate an ordering over formulas $\phi[t_1, \quad , t_n] \geq_E \psi$ if and only if ψ is of the form $\phi[t'_1, \quad , t'_n]$ and for some θ, $t_i\theta =_E t'_i$ for every $1 \leq i \leq n$

Observe that $t =_E t'$ if and only if $E \models t = t'$ Also observe that any set of equations is in Skolem Normal Form and has an initial model From these observations and Theorem 6, the correspondence between $\leq_E$ and $\leq_\Sigma$ follows immediately

Theorem 10 *Let $E = \Sigma$ be a set of equations and ϕ and ψ be formulas Then $\phi \geq_E \psi$ if and only if ϕ/TRUE $\geq_\Sigma$ ψ/TRUE*

5 Applications

The introduction to this paper mentioned many applications of the instantiation ordenngs for ordinary first-order logic, for logic with built-in equations! theories and for sorted logic This section extends the discussion to applications that employ the generalized orderings characterized by Definition 1 and by Theorems 3 and 6 The application areas discussed are (1) induction and inductive logic programming, (2) deduction and logic programming, and (3) knowledge-base vivification

Inductive Logic Programming HP focuses on inductive learning using a first-order representation, specifically a definite clause representation [Muggleton, 1992] At the foundation of ILP is the work of Plotkin [1970] and Reynolds [1970] on the computation of least upper bounds for ordinary logic This foundational work has been extended by Fnsch and Page to cover sorted logic (based on Theorem 9) [Pnech and Page, 1990], and then further to constraint logic (using a special case of the characterizations of Theorem 3 and Theorem 6) [Page and Fnsch, 1992] The extension to constraint logic can also be viewed as an extension of Buntine's [1986] definition of *generalized subsumption.*

One active area of ILP research over the last four years has been the study of PAC-leamability of restricted classes of definite clause concepts, relative to various classes of definite clause background theories, this work is summarized in [Cohen and Page, 1995] The earliest results (both positive and negative) within this area actually were proven using the aforementioned extensions of the work of Plotkin and Reynolds based on special cases of the ordenngs developed in this paper [Page and Frisch, 1992] Crucial to these results are the semantic properties of these ordenngs, as provided in the theorems of this paper

These early results on learnabuity in ILP have been extended significantly to yield a positive result that applies to structural domains, such as molecular biology or blocks world problem solving [Page, 1993] The foundation of this extended result is the characterization of instantiation in Theorem 6, which allows equality in the background theory The extended result generalizes Haussler's [1989] learnabihty result for structural domains with subset queries in a number of ways, the most significant of which are (1) the ability to learn disjunctive concepts, and (2) the use of much richer background theories (for example, background theories that are not restricted to use only unary predicates) Furthermore, it can be shown that this result subsumes most of the positive PAC-learnability results for ILP (though not the positive *PAC-predtction* results) [7]

Deductive Systems Constraint logic has been used as the basis of constraint logic programming [Jaffar and Lassez, 1987, Hohfeld and Smolka, 1988] and in generalizations of certain deductive systems [Burckert, 1991, Friech, 1994] Such systems typically employ a resolution inference rule that generalizes the ordinary rule of resolution Let us first observe how the ordinary rule of resolution is based on the ordinary instantiation ordering, and then consider how this inference rule can be generalized to constraint logic by basing it on our instantiation ordering for constraint logic

The ordinary rule of resolution operates by taking most-general instances of the two parent clauses such that the two literals being resolved upon become complements (that is, identical but opposite m sign)

[6]Jaffar, Lanez, and Maher [Jaffar *et ai* , 1986] relax this reatnctioD to allow E to be any set of definite clauses whose only predicate is the equality predicate

°For example, see the survey by Siekmann [1989]

[7]For PAC-predictability as opposed to PAC-learnability, the final hypothesis need not have any particular form (e g, a logic program or a propositional DNF formula)

This is accomplished by computing the most-general unifier and applying it to the clauses being resolved This idea can be generalized to obtain a resolution rule for constrained clauses To resolve constrained clauses $P(t_1, \ldots, t_n) \vee \phi/C$ and $\neg P(t'_1, \ldots, t'_n) \vee \phi'/C'$ we need to find the most general common instance of $P(t_1, \ldots, t_n)/C$ and $\neg P(t'_1, \ldots, t'_n)/C'$ Using Definition 1 it can be confirmed that $P(t_1, \ldots, t_n)/\mathcal{C}$ (or, equivalently, $P(t'_1, \ldots, t'_n)/\mathcal{C}$), where $\mathcal{C}$ is $t_1 = t'_1 \wedge \ldots \wedge t_n = t'_n \wedge C \wedge C'$, is a most-general common instance, provided $\mathcal{C}$ is Σ-satisfiable Consequently, the desired resolvent is $\phi \vee \phi'/\mathcal{C}$

In some simple constraint logics, such as some sorted logics, a most general common instance may not exist, in which case the above re9olutioD rule must generate many resolvents to take account of the multiple maximally-general, but incomparable, common instances The longer version of this paper identifies conditions sufficient for the existence of greatest lower bounds

The completeness of resolution and similar inference systems is usually proved by a Herbrand Theorem, which relates the satisfiability of non-ground clauses to the satisfiability of their ground instances, and a Lifting Theorem, which relates non-ground derivations to then-ground instances In a series of papers, Frisch has shown how, under certain conditions, the proofs of these theorems can be systematically transformed to obtain proofs of the corresponding theorems for inference systems based on instantiation with built-in theories These results have been formulated for sorted logic based on the characterization of Theorem 9 [Frisch, 1991], for modal logic based on the characterization of Theorem 3 [Frisch and Scherl, 1991], and for arbitrary constraint logic based *on* a characterization *equivalent to that* of Definition 1 [Frisch, 1994]

Though not immediately obvious, the preceding discussion applies equally to many systems for automated deduction in modal logic As is well-known, modal logic often can be viewed as implicit discourse about possible worlds and, therefore, can be translated to non-modal logic that explicitly discusses possible worlds Frisch and Scherl [1991] show that for many modal logics the sentences resulting from this translation can be transformed into constrainted formulas in which the accessibility conditions among possible worlds appear solely in the constraints From this point of view, the path unification algorithms employed by many modal deduction systems [Ohlbach, 1988, Jackson and Reichgelt, 1989, Wallen, 1990] can be seen to be solvere for such constraints In other words, these path unification algorithms compute the greatest lower bounds m the instantiation ordering

Knowledge Base Viviflcation The premise of vivlfication is that much of the complexity of automated deduction arises from incomplete knowledge m knowledge bases (KBs), m particular from disjunctions leading to reasoning by cases [Borgida and Etherington, 1989, Ethenngton *et ol,* 1989, Levesque, 1988] To use an example from Levesque [1988], suppose our KB includes *age{jreo\7\)* V *age{Jrea\72)* Many of the interesting consequences of this fact follow from Fred being in his early seventies or, even more generally, being a senior citizen If we know that 71 and 72 belong to the category low-seventies, we might replace $age(fred, 71) \vee age(fred, 72)$ with $\exists x\ (age(fred, x)/\text{LOW-SEVENTIES}(x))$ Of course, examples that involve binary predicates in the background information (rather than just categories such as low-seventies) or function symbols require more care

Based on the characterization of instantiation in Theorem 3, an efficient vivification algorithm can be defined that handles higher-arity predicates and function-symbols [Page, 1993] *As a simple example of the algorithm's behavior*, suppose our KB contains the following sentence

$$intimidates(son(jumbo), son(clyde)) \vee$$
$$intimidates(son(fred), son(joe))$$

Suppose our knowledge base also tells us that Jumbo is bigger than Clyde, Jumbo's son is bigger than Clyde's son, Fred is bigger than Joe, and Fred's son is bigger than Joe's son Then the vivification algorithm replaces the preceding sentence with the following existentially-closed constrained formula

$$\exists x \exists y\ (intimidates(son(x), son(y))/$$
$$\text{BIGGER}(x, y) \wedge \text{BIGGER}(son(x), son(y)))$$

While some information is lost in the replacement, much of the useful information is retained and the disjunction is eliminated

6 Conclusion

This paper has presented a general instantiation ordering for constrained formulas to which the established instantiation orderings for various rest ricted classes of constrained formulas are equivalent This ordering allows us to prove, at once, semantic properties of all these instantiation orderngs The utility of building theories into instantiation has been established by a long history of applications in automated deduction and a short history of applications m automated induction We anticipate that instantiation with built-in theories will continue its key role in deductive reasoning and will play an increasing role in non-deductive reasoning

References

[Borgida and Etherington, 1989] Alex Borgada and David W Etherington Hierarchical knowledge bases and efficient disjunctive reasoning In Ronald J Brachman, Hector J Levesque, and Raymond Reiter, editors, *Principles of Knowledge Representation and Reasoning Proceedings of the First International Conference,* pages 33-43, Toronto, May 1989

[Buntine, 1988] W Buntine Generalized subsumption and *its* applications to induction and redundancy *Artificial Intelligence,* 36(2) 149-176, 1988

[Burckert, 199l] Hans-Jiirgen Burckert *A Resolution Principle for a Logic with Restricted Quantifiers* volume 568 of *Lecture Notes in Artificial Intelligence.* Springer-Verlag, Germany, 1991

[Cohen and Page, 1995] W W Cohen and C D Page Polynomial learnability and inductive logic programming- Methods and results New Generation Computing, May 1995

[Conn, 1987] Anthony G Conn A more expressive formulation of many sorted logic Journal of Automated Reasoning, 3(2) 113-200, 1987

[Ethenngton et at, 1989] David W Ethenngton, Alex Borgida, Ronald J Brachman, and Henry Hautz Vivid knowledge and tractable reasoning A preliminary report In IJCAI-89, pages 1146-1152, Detroit, 1989

[FVisch and Page, 1990] A M Fnsch and C D Page Generalization with taxonomic information In Proceedings of AAAI-90, pages 755-761, Boston, MA, 1990

[FVisch and Scherl, 1991] Alan M Fnsch and Richard B Scherl A general framework for modal deduction In James Allen, Richard Fikes, and Erik Sandewall, editors, Principles of Knowledge Representation and Reasoning Proceedings of the Second International Conference, pages 196-207 Morgan Kaufman, San Mateo, CA, 1991

[FVisch, 1991] Alan M Frisch The substitutional framework for sorted deduction Fundamental results on hybrid reasoning Artificial Intelligence, 49 161-198, 1991

[FVisch, 1994] Alan M Fnsch Deduction with constraints The substitutional framework for hybrid reasoning In Proc of the CADE-12 Workshop on Theory Reasoning in Automated Deduction, pages 44-50, Nancy, France, June 1994

[Goguen and Meseguer, 1986] J A Goguen and J Meseguer EQLOG Equality, types, and generic modules for logic programming In D DeGroot and G Landstrom, editors, Logic Programming Functions, Relations, and Equations, pages 295-363 Prentice-Hall, London, 1986

[Haussler, 1989] D Haussler Learning conjunctive concepts in structural domains Machine Learning, 4 7-40, 1989

[Hohfeld and Smolka, 1988] Markus Hohfeld and Gert Smolka, Definite relations over constraint languages Ldog-Report 53, IBM Deutschland, October 1988

[Jackson and Reichgelt, 1969] Peter Jackson and Han Reichgelt A general proof method for modal predicate logic In Peter Jackson, Han Reichgelt, and Frank van Harmelen, editors, Logic-Based Knowledge Representation, chapter 8, pages 177-228 MIT Press, Cambridge, Massachusetts, 1989

[Jaffar and Lassez, 1987] Joxan Jaffar and Jean-Louis Lassez Constraint logic programming In Proceedings of the 14th ACM Principles of Programming Lan guages Conference, pages 111-119, Munich, January 1987

[Jaffar et aL, 1986] Joxan Jaffar, Jean-Louis Lassez, and Michael J Maher A logic programming language scheme In Doug DeGroot and Gary Lindstrom, editors, Logic Programming Functions, Relations and Equations, pages 441-467 Prentice-Hall, Englewood Cliffe, NJ, 1986

[Levesque, 1988] H J Levesque. Logic and the complexity of reasoning Journal of Philosophical Logic, 17 355-389, 1988

[Muggleton, 1992] S H Muggleton Inductive logic programming In S H Muggleton, editor, Inductive Logic Programming, pages 3-27 Academic PresB, London, 1992

[Ohlbach, 1988] Hans Jurgen Ohlbach A resolution calculus for modal logic In E Lusk and R. Overbeek, editors, 9th International Conference on Automated Deduction, volume 310 of Lecture Notes in Computer Science, pages 500-516 Springer-Verlag, Berlin, May 1988

[Page and FVisch, 1992] C D Page and A M Fnsch Generalization and learnability A study of constrained atoms In S H Muggleton, editor, Inductive Logic Programming, pages 29-61 Academic Press, London, 1992

[Page, 1993] C D Page Anti-unification in constraint logics foundations and applications to learnability in first-order logic, to speed-up learning, and to deduc tion. PhD thesis, University of Illinois at Urbana-Champaign, 1993

[Plotkin, 1970] G D Plotkin A note on inductive gen eralization In B Meltzer and D Michie, editors, Machine Intelligence 5, pages 153-163 Edinburgh University Press, Edinburgh, 1970

[Plotkin, 1972] G D Plotkin Building-in equational theones In B Meltzer and D Michie, editors, Machine Intelligence 7, pages 73-90 Edinburgh University Press, 1972

[Reynolds, 1970] J C Reynolds Transformational systems and algebraic structure of atomic formulas In B Meltzer and D Michie, editors, Machine Intelligence 5, pages 135-52 Edinburgh University Press, Edinburgh, 1970

[Schmidt-Schauss, 1989] Manfred Scamidt-Schauss Computational Aspects of an Order-Sorted Logic with Term Declarations, volume 395 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1989

[Siekmann, 1989] J Siekmann Unification theory Journal of Symbolic Computation, 7(3) 207-274, 1989

[Wallen, 1990] Lincoln A Wallen Automated Proof Search in Non-Classical Logics Efficient Matrix Proof Methods for Modal and Intuitionistic Logics The MIT Press, Cambndge, Massachusetts, 1990

[Walther, 1987] Christoph Walther A Many-Sorted Calculus Based on Resolution and Paramodulation. Morgan Kaufman, Los Altos, CA, 1987