Notions of Attack and Justified Arguments for Extended Logic Programs

Ralf Schweimeier and **Michael Schroeder**[1]

Abstract. The concept of argumentation may be used to give a formal semantics to a variety of assumption based reasoning formalisms. In particular, various argumentation semantics have been proposed for logic programming with default negation. For extended logic programming, i.e. logic programming with two kinds of negation, there arise a variety of notions of attack on an argument, and therefore a variety of different argumentation semantics.

The purpose of this paper is to shed some light on these various semantics, and examine the relationship between different semantics. We identify a number of different notions of attack for extended logic programs, and compare the resulting least fixpoint semantics, defined via acceptability of arguments. We investigate the validity of the coherence principle, and notions of consistency for these semantics.

1 Introduction

Argumentation has attracted much interest in the area of AI. On the one hand, argumentation is an important way of human interaction and reasoning, and is therefore of interest for research into intelligent systems. Application areas include automated negotiation via argumentation [15, 14, 17] and legal reasoning [16]. On the other hand, argumentation provides a formal model for various assumption based (or non-monotonic, or default) reasoning formalisms [3, 4]. In particular, various argumentation based semantics have been proposed for logic programming with default negation [3, 7], some of them equivalent with well-known semantics for normal logic programs such as the stable model semantics [9], or the well-founded semantics [8].

Argumentation semantics are elegant since they can be captured in an abstract framework [7, 3, 19, 12], for which an elegant theory of attack, defence, acceptability, and other notions can be developed, without recourse to the concrete instance of the reasoning formalism at hand. This framework can then be instantiated to various assumption based reasoning formalisms. Similarly, a dialectical proof theory can be defined for an abstract argumentation framework, and then applied to any instance of such a framework [7, 11].

In general, an argument A is a proof which may use a set of defeasible assumptions. Another argument B may have a conclusion which contradicts the assumptions or the conclusions of A, and thereby B *attacks* A. There are two fundamental notions of such attacks: *undercut* and *rebut* [16] or equivalently *ground-attack* and *reductio-ad-absurdum attack* [6]. We will use the terminology of undercuts and rebuts. Both attacks differ in that an undercut attacks a premise of an argument, while a rebut attacks a conclusion.

So, given a logic program we can define an argumentation semantics by iteratively collecting those arguments which are acceptable to a proponent, i.e. they can be defended against all opponent attacks. In fact, such a notion of acceptability can be defined in a number of ways depending on which attacks we allow the proponent and opponent to use.

Normal logic programs do not have negative conclusions, which means that we cannot use rebuts. Thus both opponents can only launch undercuts on each other's assumptions. Extended logic programs [10, 2, 20], on the other hand, introduce explicit negation, which states that a literal is explicitly false. As a result, both undercuts and rebuts are possible forms of attack; there are further variations depending on whether any kind of counter-attack is admitted. A variety of argumentation semantics arise if one allows one notion of attack as defence for the proponent, and another as attack for the opponent. The aim of this paper is to examine the relationships between and properties of these different argumentation semantics. In particular, we would like to relate different existing argumentation semantics [6, 16], and find an argumentation semantics which is equivalent to the well-founded semantics for extended logic programs [2].

The paper is organised as follows: First we define arguments and notions of attack and acceptability. The relationships between the fixpoint semantics resulting from the different notions are examined in Section 4. In Sections 5, we examine some properties of these semantics regarding consistency, and the coherence principle which relates explicit and default negation.

2 Extended LP and Argumentation

We summarise the definitions of arguments for extended logic programs, and define various notions of attack between arguments.

2.1 Arguments

Definition 1 *An* objective literal *is an atom A or its explicit negation $\neg A$. We define $\neg\neg L = L$. A* default literal *is of the form $not\ L$ where L is an objective literal. A* literal *is either an objective or a default literal. An* extended logic program *is a (possibly infinite) set of rules of the form $L_0 \leftarrow L_1, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_{m+n} (m, n \geq 0)$, where each L_i is an objective literal ($0 \leq i \leq m+n$). For such a rule r, we call L_0 the* head *of the rule, $head(r)$, and $L_1, \ldots, not\ L_{m+n}$ the* body *of the rule, $body(r)$.*

Our definition of an argument for an extended logic program is based on [16]. Essentially, an argument is a partial proof, resting on a number of *assumptions*, i.e. a set of default literals.[2] Note that we do not consider priorities of arguments, as used e.g. in [16, 19].

Definition 2 *Let P be an extended logic program. An* argument *for P is a finite sequence $A = [r_1, \ldots r_n]$ of ground instances of rules*

[1] Deptartment of Computing, City University, London EC1V 0HB, UK, {ralf,msch}@soi.city.ac.uk

[2] In [3, 6], an argument *is* a set of assumptions; the two approaches are equivalent in that there is an argument with a conclusion L iff there is a set of assumptions from which L can be inferred. See the discussion in [16].

$r_i \in P$ *such that 1. for every* $1 \leq i \leq n$*, for every objective literal* L_j *in the body of* r_i *there is a* $k > i$ *such that* $head(r_k) = L_j$*. 2. No two distinct rules in the sequence have the same head. The head of a rule in* A *is called a* conclusion *of* A*, and a default literal* $not\ L$ *in the body of a rule of* A *is called an* assumption *of* A*.*

The restriction to minimal arguments is not essential, but convenient, since it rules out arguments constructed from several unrelated arguments. Generally, one is interested in the conclusions of an argument, and wants to avoid having rules in an argument which do not contribute to the desired conclusion.

2.2 Notions of Attack

There are two fundamental notions of attack: *undercut*, which invalidates an assumption of an argument, and *rebut*, which contradicts a conclusion of an argument [6, 16]. From these, we may define further notions of attack, by allowing either of the two fundamental kinds of attack, and considering whether any kind of counter-attack is allowed or not. We will now formally define these notions of attacks.

Definition 3 *Let* A_1 *and* A_2 *be arguments. 1.* A_1 undercuts A_2 *if there is an objective literal* L *such that* L *is a conclusion of* A_1 *and* $not\ L$ *is an assumption of* A_2*. 2.* A_1 rebuts A_2 *if there is an objective literal* L *such that* L *is a conclusion of* A_1 *and* $\neg L$ *is a conclusion of* A_2*. 3.* A_1 attacks A_2 *if* A_1 *undercuts or rebuts* A_2*. 4.* A_1 defeats A_2 *if* A_1 *undercuts* A_2*, or* A_1 *rebuts* A_2 *and* A_2 *does not undercut* A_1*. 5.* A_1 strongly attacks A_2 *if* A_1 *attacks* A_2 *and* A_2 *does not undercut* A_1*. 6.* A_1 strongly undercuts A_2 *if* A_1 *undercuts* A_2 *and* A_2 *does not undercut* A_1*.*

The notions of *undercut* and *rebut*, and hence *attack* are fundamental for extended logic programs [6, 16]. The notion of *defeat* is used in [16], along with a notion of *strict defeat*, i.e. a defeat that is not counter-defeated. For arguments without priorities, rebuts are symmetrical, and therefore strict defeat coincides with strict undercut, i.e. an undercut that is not counter-undercut. Similarly, strict attack coincides with strict undercut. For this reason, we use the term *strong undercut* instead of *strict undercut*, and similarly define *strong attack* to be an attack which is not counter-undercut. We will use the following abbreviations for these notions of attack. *r* for rebuts, *u* for undercuts, *a* for attacks, *d* for defeats, *sa* for strongly attacks, and *su* for strongly undercuts.

These notions of attack define for any extended logic program a binary relation on the set of arguments of that program.

Definition 4 *A* notion of attack *is a function* x *which assigns to each extended logic program* P *a binary relation* x_P *on the set of arguments of* P*, i.e.* $x_P \subseteq Args_P^2$*. Notions of attack can be partially ordered by defining* $x \subseteq y$ *iff* $\forall P : x_P \subseteq y_P$

Definition 5 *Let* x *be a notion of attack. Then the* inverse *of* x*, denoted by* x^{-1}*, is defined as* $x_P^{-1} = \{(B, A) \mid (A, B) \in x_P\}$*.*

In this relational notation, Definition 3 can be rewritten as $a = u \cup r$, $d = u \cup (r - u^{-1})$, $sa = (u \cup r) - u^{-1}$, and $su = u - u^{-1}$.

Proposition 1 *The notions of attack* su, u, sa, d, a *of Definition 3 are partially ordered according to the following Hasse diagram in Fig 1.*

Proof. Apply set-theoretic laws $A - B \subseteq A \subseteq A \cup C$ and $(A \cup B) - C = (A - C) \cup (B - C)$ (for all sets A, B, and C), to the definitions. □

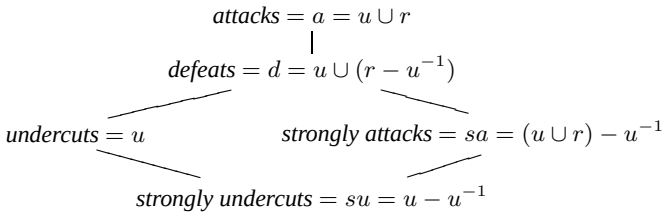

Figure 1. Hasse diagramme for notions of attack

This diagram in Fig. 1 contains the notions of attack used in [6, 16], plus *strongly attacks* which is a natural intermediate notion between *strongly undercuts* and *defeats*. Undercuts are present in all the notions of attack, since they are fundamental when dealing with default literals. In the absence of priorities, rebuts are symmetric, i.e. if A rebuts B, B also rebuts A, while undercuts are not symmetric in general.

3 Acceptability and Justified Arguments

Given the above notions of attack, we can deploy them to define the acceptability of an argument. Basically, an argument is acceptable if it can be defended against any attack. Depending on which particular notion of attack we use as defence and which for the opponent's attacks, we obtain a host of acceptability notions.

Acceptability forms the basis for our argumentation semantics, which is defined as the least fixpoint of a function, which collects all acceptable arguments. The *least* fixpoint is of particular interest [16, 6], because it provides a canonical fixpoint semantics and it can be constructed inductively.

Definition 6 *Let* x *and* y *be notions of attack. Let* A *be an argument, and* S *a set of arguments. Then* A *is* x/y-acceptable wrt. S *if for every argument* B *such that* $(B, A) \in x$ *there exists an argument* $C \in S$ *such that* $(C, B) \in y$*.*

Based on the notion of acceptability, we can then define a fixpoint semantics for arguments.

Definition 7 *Let* x *and* y *be notions of attack, and* P *an extended logic program. The operator* $F_{P,x/y} : \mathcal{P}(Args_P) \rightarrow \mathcal{P}(Args_P)$ *is defined as* $F_{P,x/y}(S) = \{A \mid A$ *is* x/y*-acceptable wrt.* $S\}$*.*

The set of x/y-justified arguments *is the least fixpoint of* $F_{P,x/y}$*. We denote the set of* x/y*-justified arguments by* $J_{P,x/y}$*. If the program* P *is clear from the context, we omit the subscript* P*.*

For any program P*, the least fixpoint exists by the Knaster-Tarski fixpoint theorem [18], because* $F_{P,x/y}$ *is monotone. It can be constructed by transfinite induction as follows:*

$$\begin{array}{lcll} J^0_{x/y} & = & \emptyset & \\ J^{\alpha+1}_{x/y} & = & F_{P,x/y}(J^\alpha_{x/y}) & \text{for } \alpha+1 \text{ a successor ordinal} \\ J^\lambda_{x/y} & = & \bigcup_{\alpha<\lambda} J^\alpha_{x/y} & \text{for } \lambda \text{ a limit ordinal} \end{array}$$

Then there exists a least ordinal λ_0 *such that* $F_{x/y}(J^{\lambda_0}_{x/y}) = J^{\lambda_0}_{x/y} = J_{x/y}$*.*

4 Relationships of Notions of Justifiability

This section is devoted to an analysis of the relationship between the different notions of justifiability, leading to a hierarchy of notions of justifiability illustrated in Figure 3.

First of all, it is easy to see that the least fixpoint increases if we weaken the attacks, or strengthen the defence.

Proposition 2 *Let $x' \subseteq x$, $y \subseteq y'$ be notions of attack, then $J_{x/y} \subseteq J_{x'/y'}$.*

Theorem 3 states that it does not make a difference if we allow only the strong version of the defence. This is because an argument need not defend itself on its own, but it may rely on other arguments to defend it.

We only give a formal proof for the first theorem; the proofs for the other theorems are similar, and we provide an intuitive informal explanation instead.

Theorem 3 *Let $x \supseteq undercuts$ and y be notions of attack, and $sy = y - undercuts^{-1}$. Then $J_{x/y} = J_{x/sy}$.*

Proof. Informally, every x-attack B to an x/y-justified argument A is y-defended by some x/sy-justified argument C (by induction). Now if C was *not* a sy-attack, then it is undercut by B, and because $x \supseteq undercuts$ and C is justified, there exists a *strong* defence for C against B, which is also a defence of the original argument A against C.

The formal proof is by transfinite induction. By Proposition 2, we have $J_{x/sy} \subseteq J_{x/y}$. We prove the inverse inclusion by showing that for all ordinals α: $J^{\alpha}_{x/y} \subseteq J^{\alpha}_{x/sy}$, by transfinite induction on α.

Base case $\alpha = 0$: $J_{x/y} = \emptyset = J_{x/sy}$.

Successor ordinal $\alpha \rightsquigarrow \alpha + 1$: Let $A \in J^{\alpha+1}_{x/y}$, and $(B, A) \in x$. By definition, there exists $C \in J^{\alpha}_{x/y}$ such that $(C, B) \in y$. By induction hypothesis, $C \in J^{\alpha}_{x/sy}$.

If B does not undercut C, then we are done. If, however, B undercuts C, then because $C \in J^{\alpha}_{x/sy}$, and $undercuts \subseteq x$, there exists $D \in J^{\alpha_0}_{x/sy}(\emptyset)(\alpha_0 < \alpha)$ such that $(D, B) \in sy$. It follows that $A \in J^{\alpha+1}_{x/sy}$.

Limit ordinal λ: Assume $J^{\alpha}_{x/y} \subseteq J^{\alpha}_{x/sy}$ for all $\alpha < \lambda$. Then $J^{\lambda}_{x/y} = \bigcup_{\alpha<\lambda} J^{\alpha}_{x/y} \subseteq \bigcup_{\alpha<\lambda} J^{\alpha}_{x/sy} = J^{\lambda}_{x/sy}$ □

In particular, the previous Theorem states that undercut and strong undercut are equivalent as a defence, as are attack and strong attack. This may be useful in an implementation, where we may use the stronger notion of defence without changing the semantics, thereby decreasing the number of arguments to be checked. The following Corollary shows that because defeat lies between attack and strong attack, it is equivalent to both as a defence.

Corollary 4 *Let $x \supseteq undercuts$. Then $J_{x/a} = J_{x/d} = J_{x/sa}$.*

Proof. With Proposition 2 and Theorem 3, we have $J_{x/a} \subseteq J_{x/d} \subseteq J_{x/sa} = J_{x/a}$. □

Theorem 5 *Let $x \supseteq$ strongly attacks. Then $J_{x/u} = J_{x/d} = J_{x/a}$.*

Proof. Every x-attack B to a x/a-justified argument a is attacked by some x/u-justified argument C (by induction). If C is a rebut, but not an undercut, then because B strongly attacks C, and because $x \supseteq$ *strongly attacks*, there must have been an argument defending C by undercutting B, thereby also defending A against B.
The statement for *defeats* follows in a similar way to Corollary 4. □

Theorem 6 $J_{sa/su} = J_{sa/sa}$

The proof is similar to Theorem 5.

$P_1 =$
$p \leftarrow not\ q$
$q \leftarrow not\ p$

$P_2 =$
$p \leftarrow not\ q$
$q \leftarrow not\ p$
$\neg p$

$P_3 =$
$p \leftarrow not\ q$
$q \leftarrow not\ r$
$r \leftarrow not\ s$
$s \leftarrow not\ p$
$\neg p$

$P_4 =$
$p \leftarrow not\ q$
$q \leftarrow not\ p$
$r \leftarrow not\ p$

$P_5 =$
$p \leftarrow not\ \neg p$
$\neg p$

$P_6 =$
$\neg p \leftarrow not\ q$
$\neg q \leftarrow not\ p$
p
q

Figure 2. Examples

Theorem 7 $J_{su/a} = J_{su/d}$

Proof. Every strong undercut B to a su/a-justified argument A is attacked by some su/d-justified argument C (by induction). If C does not defeat A, then there is some argument D defending C by defeating B, thereby also defending A against B. □

We will now present some example programs which distinguish various notions of justifiability.

Example 1 *Consider P_1 in Figure 2. For any notion of attack x, we have $J_{su/x} = J_{sa/x} = \{[p \leftarrow not\ q], [q \leftarrow not\ p]\}$, because there is no strong undercut or strong attack to any of the arguments. However, $J_{a/x} = J_{d/x} = J_{u/x} = \emptyset$, because every argument is undercut (and therefore defeated and attacked).*

Example 2 *Consider P_2 in Figure 2. Let x be a notion of attack. Then $J_{d/x} = J_{a/x} = \emptyset$, because every argument is defeated (hence attacked). $J_{sa/su} = J_{sa/sa} = \{[q \leftarrow not\ p]\}$, because $[q \leftarrow not\ p]$ is the only argument which is not strongly attacked, but it does not strongly attack any other argument. $J_{u/su} = J_{u/u} = \{[\neg p]\}$, because there is no undercut to $[\neg p]$, but $[\neg p]$ does not undercut any other argument. $J_{u/a} = \{[\neg p], [q \leftarrow not\ p]\}$, because there is no undercut to $[\neg p]$, and the undercut $[p \leftarrow not\ p]$ to $[q \leftarrow not\ p]$ is attacked by $[\neg p]$. We also have $J_{sa/u} = \{[\neg p], [q \leftarrow not\ p]\}$, because $[q \leftarrow not\ p]$ is not strongly attacked, and the strong attack $[p \leftarrow not\ q]$ on $[\neg p]$ is undercut by $[q \leftarrow not\ p]$. Finally, $J_{su/x} = \{[\neg p], [p \leftarrow not\ q], [q \leftarrow not\ p]\}$, because none of the arguments is strongly undercut.*

Example 3 *Consider P_3 in Figure 2. Let x be a notion of attack. Then $J_{sa/x} = \emptyset$, because every argument is strongly attacked.*

$J_{su/u} = J_{su/su} = \{[\neg p]\}$, because all arguments except $[\neg p]$ are strongly undercut, but $[\neg p]$ does not undercut any argument. And $J_{u/a} = J_{su/sa} = J_{su/a} = \{[\neg p], [q \leftarrow not\ r], [s \leftarrow not\ p]\}$.

Example 4 *Consider P_4 in Figure 2. Let x be a notion of attack. Then $J_{u/x} = J_{d/x} = J_{a/x} = \emptyset$, because every argument is undercut. $J_{su/su} = J_{su/sa} = J_{sa/su} = J_{sa/sa} = \{[p \leftarrow not\ q], [q \leftarrow not\ p]\}$ In this case, the strong attacks are precisely the strong undercuts; The argument $[r \leftarrow not\ p]$ is not justified, because the strong undercut $[p \leftarrow not\ q]$ is undercut, but not strongly undercut, by $[q \leftarrow not\ p]$. $J_{su/u} = J_{su/a} = J_{sa/u} = J_{sa/a} = \{[p \leftarrow not\ q], [q \leftarrow not\ p], [r \leftarrow not\ p]\}$ Again, undercuts and attacks, and strong undercuts and strong attacks, coincide; but now $[r \leftarrow not\ p]$ is justified, because non-strong undercuts are allowed as defence.*

Example 5 *Consider P_5 in Figure 2. Then $J_{a/x} = \emptyset$, because both arguments attack each other, while $J_{d/x} = \{[\neg p]\}$, because $[\neg p]$ defeats $[p \leftarrow not\ \neg p]$, but not vice versa.*

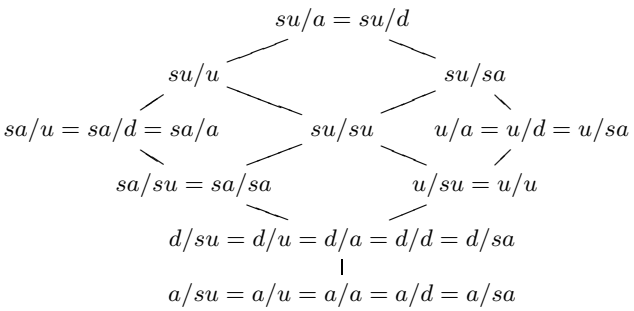

Figure 3. Hierarchy of Notions of Justifiability

Example 6 *Consider P_6 in Figure 2. Let x be a notion of attack. Then $J_{sa/x} = J_{d/x} = J_{a/x} = \emptyset$, because every argument is strongly attacked (hence defeated and attacked), while $J_{u/x} = J_{su/x} = \{[p], [q]\}$.*

Theorem 8 *The notions of justifiability are ordered (by set inclusion) according to the Hasse diagram in Figure 3.*

By definition, Dung's grounded argumentation semantics [6] is exactly a/u-justifiability, while Prakken and Sartor's semantics [16], if we disregard priorities, amounts to d/su-justifiability. As corollaries to Theorem 8, we obtain relationships of these semantics to the other notions of justifiability.

Corollary 9 *Let J_D be the set of justified arguments according to Dung's grounded argumentation semantics [6]. Then $J_D = J_{a/su} = J_{a/u} = J_{a/a} = J_{a/d} = J_{a/sa}$ and $J_D \subsetneq J_{x/y}$ for all $x \neq a$ and y.*

Corollary 10 *Let J_{PS} be the set of justified arguments according to Prakken and Sartor's argumentation semantics [16], where all arguments have the same priority. Then $J_{PS} = J_{d/su} = J_{d/u} = J_{d/a} = J_{d/d} = J_{d/sa}$, $J_{PS} \subsetneq J_{x/y}$ for all $x \notin \{a, d\}$ and y, and $J_{PS} \supsetneq J_{a/y}$ for all notions of attack y.*

For normal logic programs, it has been established that the least fixpoint argumentation semantics is equivalent to the well-founded semantics, see e.g. [13]. One of the aims of this paper is to clarify which notions of attack would be appropriate to obtain a least fixpoint argumentation semantics equivalent to the well-founded semantics for extended logic programming [2]. The following theorem presents the solution.

Theorem 11 *Let P be an extended logic program and $WFM(P)$ its well-founded model [2]. Then $WFM(P) = \{L \mid$ there exists a u/a-justified argument for $L\} \cup \{not\ L \mid$ all arguments for L are attacked by a u/a-justified argument$\}$.*

Proof (Sketch). For the sake of brevity we can only sketch the proof. In [1], the well-founded model is defined as the least fixpoint of the operator $\Gamma\Gamma_s$, where Γ is the Gelfond-Lifschitz operator [9], and Γ_s is the same operator applied to the *semi-normal* version of the program, obtained by adding $not\ \neg L$ to the body of each rule with head L. The fixpoint is generated by a transfinite sequence of interpretations $\{I_\alpha\}$. We show that I_alpha corresponds to $J^\alpha_{u/a}$ as defined in Def. 7 by showing that undercuts correspond to the Γ operator, because they attack default literals and attacks correspond to the Γ_s operator, because they attack either the head of a rule or a default literal in the body. □

Note also the following observations:

1. The notions of a/x-, d/x- and sa/x-justifiability are very skeptical in that a *fact* p may not be justified, if there is a rule $\neg p \leftarrow B$ (where $not\ p \notin B$) that is not x-attacked. On the other hand this is useful in terms of avoiding inconsistency.

2. sx/y-justifiability is very credulous, because it does not take into account non-strong attacks, so e.g. the program $\{p \leftarrow not\ q, q \leftarrow not\ p\}$ has the justified arguments $[p \leftarrow not\ q]$ and $[q \leftarrow not\ p]$.

5 The Coherence Principle and Consistency

The coherence principle for extended logic programming [2] states that "explicit negation implies implicit negation". If the intended meaning of $not\ L$ is "if there is no evidence for L, assume that L is false", and the intended meaning of $\neg L$ is "there is evidence for the falsity of L", then the coherence principle states that explicit evidence is preferred over assumption of the lack of evidence. Formally, this can be stated as: if $\neg L$ is in the semantics, then $not\ L$ is also in the semantics. In an argumentation semantics, we have not defined what it means for a default literal to be "in the semantics". This can easily be remedied, though.

Definition 8 *Let P be an extended logic program, and x and y notions of attack, and let L be an objective literal. Then L is x/y-*justified *if there exists a x/y-justified argument for L.*

*Let nL be a fresh atom, and $P' = P \cup \{nL \leftarrow not\ L\}$. Then $not\ L$ is x/y-*justified *if $[nL \leftarrow not\ L]$ is a x/y-justified argument for P'.*

Note that because nL is fresh, $J_{x/y}(P') = J_{x/y}(P)$ or $J_{x/y}(P') = J_{x/y}(P) \cup \{[nL \leftarrow not\ L]\}$.

Definition 9 *A least fixpoint semantics $J_{x/y}$* satisfies the coherence principle *if for every objective literal L, if $\neg L$ is x/y-justified, then $not\ L$ is x/y-justified.*

The following result states that a least fixpoint semantics satisfies the coherence principle exactly if we allow any attack for the defence. Informally, this is because the only way of attacking a default literal $not\ L$ is by undercut, i.e. an argument for L, and in general, such an argument can only be attacked by an argument for $\neg L$ by a rebut.

Theorem 12 *Let $x, y \in \{a, u, d, su, sa\}$. Then $J_{x/y}$ satisfies the coherence principle iff $J_{x/y} = J_{x/a}$.*

Proof. For the "if" direction, we show that for those notions of justifiability $x/y \neq x/a$, the coherence principle does not hold.

Consider P_3 in Figure 2. Then $J_{u/u}(P') = J_{su/u}(P') = J_{su/su}(P') = \{[\neg p]\}$. Now Consider P_5 in Figure 2. Then $J_{su/sa}(Q') = J_{sa/sa}(Q') = \{[p \leftarrow not\ \neg p], [\neg p \leftarrow not\ p]\}$.

For the "only if" direction, let x be a notion of attack. Let P be an extended logic program, and $\neg L$ a x/a-justified literal, i.e. there is an argument $A = [\neg L \leftarrow Body, \ldots]$ and an ordinal α s.t. $A \in J^\alpha_{x/a}$.

Let $A' = [nL \leftarrow not\ L]$, and $(B, A') \in x$. Because nL is fresh, the only possible attack on A' is a strong undercut, i.e. L is a conclusion of B. Then A attacks B, and so $[nL \leftarrow not\ L] \in J^{\alpha+1}_{x/a}$. □

Consistency is an important property of a logical system. It states that the system does not support contradictory conclusions. In classical logic "ex falso quodlibet", i.e. if both A and $\neg A$ hold, then any formula holds. In paraconsistent systems [5], this property does not

hold, thus allowing both A and $\neg A$ to hold for a particular formula A, while not supporting any other contradictions.

A set of arguments is consistent if it does not contain two arguments such that one attacks the other. There are several notions of consistency, depending on which notion of attack is considered undesirable.

Definition 10 *Let x be a notion of attack, and P an extended logic program. Then a set of arguments for P is called x-consistent if it does not contain arguments A and B such that $(A, B) \in x_P$.*

The argumentation semantics of an extended logic program need not necessarily be consistent; because of explicit negation, there exist contradictory programs such as $\{p, \neg p\}$, for which there exist sensible, but inconsistent arguments ($[p]$ and $[\neg p]$ in this case).

A general result identifies cases in which the set of justified arguments for a program is consistent. It states that if we allow the attack to be at least as strong as the defence, i.e. if we are *sceptical*, then the set of justified arguments is consistent.

Theorem 13 *Let $x \supseteq y$ be notions of attack, and P an extended logic program. Then the set of x/y-justified arguments is x-consistent.*

Proof. We show that $J^{\alpha}_{x/y}$ is x-consistent for all ordinals α, by transfinite induction on α.
Base case $\alpha = 0$: Trivial.
Successor ordinal $\alpha \rightsquigarrow \alpha + 1$: Assume $A, B \in J^{\alpha+1}_{x/y}$ and $(A, B) \in x$. Then there exists $C \in J^{\alpha}_{x/y}$ such that $(C, A) \in y \subseteq x$. Then by induction hypothesis, because $C \in J^{\alpha}_{x/y}$, then $A \notin J^{\alpha}_{x/y}$. Because $A \in J^{\alpha+1}_{x/y}$, there exists $D \in J^{\alpha}_{x/y}$ such that $(D, C) \in y \subseteq y$. This contradicts the induction hypothesis, so we have to retract the assumption and conclude that $J^{\alpha+1}_{x/y}$ is x-consistent.
Limit ordinal λ: Assume $A, B \in J^{\lambda}_{x/y}$ and $(A, B) \in x$. Then there exist $\alpha, \beta < \lambda$ s.t. $A \in J^{\alpha}_{x/y}$ and $B \in J^{\beta}_{x/y}$. W.l.o.g. assume that $\alpha \leq \beta$. Then because $J^{\alpha}_{x/y} \subseteq J^{\beta}_{x/y}$, we have $A \in J^{\beta}_{x/y}$, contradicting the induction hypothesis that $J^{\beta}_{x/y}$ is x-consistent. □

The following example shows that, in general, the set of justified arguments may well be inconsistent.

Example 7 *Consider $P = \{q \leftarrow not\ p.\quad p.\quad \neg p.\}$ Then $J_{u/a} = \{[q \leftarrow not\ p], [p], [\neg p]\}$, and $[p]$ and $[\neg p]$ rebut each other, and $[p]$ strongly undercuts $[q \leftarrow not\ p]$.*

6 Conclusion and Further Work

We have identified various notions of attack for extended logic programs. Based on these notions of attack, we defined notions of acceptability and least fixpoint semantics. These fixpoint semantics were related by establishing a lattice of justified arguments, based on set inclusion. In particular, we identified an argumentation semantics $J_{u/a}$ equal to the well-founded semantics for logic programs with explicit negation, $WFSX$ [2], and established that $J_D \subseteq J_{PS} \subseteq J_{u/a} = WFSX$, where J_D and J_{PS} are the least fixpoint argumentation semantics of Dung [6] and Prakken and Sartor [16]. Finally, we identified a sufficient and necessary criterion for when a least fixpoint semantics satisfies the coherence principle, and a sufficient criterion for a fixpoint semantics to be consistent.

There have been some results [7, 13] showing that certain argumentation semantics for normal logic programs coincide with well-known semantics such as the stable model semantics [9] or the well-founded semantics [8]. For extended logic programs, it has been shown in [6] that the stable argumentation semantics coincides with the answer set semantics of [10]; a similar result [6] is stated for the well-founded semantics, albeit restricted to the case of normal logic programs. One aim we achieved with this paper is the definition of an argumentation semantics equivalent to the well-founded semantics for extended logic programs [2]. The results concerning the relationships as depicted in Fig. 3 have been valuable in identifying such a semantics, as well as relating it to existing semantics. Furthermore the equivalence of $J_{u/a}$ and $WFSX$ allow the use of the efficient top-down proof procedure for $WFSX$ [1] to compute justified arguments in $J_{u/a}$.

Future research will determine how to adapt this proof procedure to different argumentation semantics and its relation to dialogue games as defined in [7, 13, 16, 11]. It is also an open question how the hierarchy changes when priorities are added as defined in [16, 19].

REFERENCES

[1] J. J. Alferes, C. V. Damásio, and L. M. Pereira, 'A logic programming system for non-monotonic reasoning', *Journal of Automated Reasoning*, **14**(1), 93–147, (1995).
[2] J. J. Alferes and L. M. Pereira, *Reasoning with Logic Programming*, (LNAI 1111), Springer-Verlag, 1996.
[3] A. Bondarenko, P.M. Dung, R.A. Kowalski, and F. Toni, 'An abstract, argumentation-theoretic approach to default reasoning', *Artificial Intelligence*, **93**(1-2), 63–101, (1997).
[4] C.I. Chesñevar, A.G. Maguitman, and R.P. Loui, 'Logical models of argument', *ACM Computing Surveys*, **32**(4), 337–383, (Dec 2000).
[5] C. Damasio and L. Pereira, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 2, chapter A Survey on Paraconsistent Semantics for Extended Logic Programs, 241–320, Kluwer, 1998.
[6] P. M. Dung, 'An argumentation semantics for logic programming with explicit negation', in *Proc. of Intl. Conf. on Logic Programming*, pp. 616–630. MIT Press, (1993).
[7] P. M. Dung, 'On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games', *Artificial Intelligence*, **77**(2), 321–357, (1995).
[8] A. Van Gelder, K.A. Ross, and J.S. Schlipf, 'The well-founded semantics for general logic programs', *Journal of the ACM*, **38**(3), 620–650, (July 1991).
[9] M. Gelfond and V. Lifschitz, 'The stable model semantics for logic programming', in *5th International Conference on Logic Programming*, eds., R. A. Kowalski and K. A. Bowen, 1070–1080, MIT Press, (1988).
[10] M. Gelfond and V. Lifschitz, 'Logic programs with classical negation', in *Proc. of ICLP90*, pp. 579–597. MIT Press, (1990).
[11] H. Jakobovits and D. Vermeir, 'Dialectic semantics for argumentation frameworks', in *Proc. of Intl. Conf. on Artificial Intelligence and Law*, pp. 53–62, (1999).
[12] H. Jakobovits and D. Vermeir, 'Robust semantics for argumentation frameworks', *Journal of Logic and Computation*, **9**(2), 215–261, (1999).
[13] A.C. Kakas and F. Toni, 'Computing argumentation in logic programming', *Journal of Logic and Computation*, **9**, 515–562, (1999).
[14] S. Kraus, K. Sycara, and A. Evenchik, 'Reaching agreements through argumentation: a logical model and implementation', *Artificial Intelligence*, **104**(1-2), 1–69, (1998).
[15] S. Parsons, C. Sierra, and N. Jennings, 'Agents that reason and negotiate by arguing', *Journal of Logic and Computation*, **8**(3), 261–292, (1998).
[16] H. Prakken and G. Sartor, 'Argument-based extended logig programming with defeasible priorities', *Journal of Applied Non-Classical Logics*, **7**(1), (1997).
[17] M. Schroeder, 'An efficient argumentation framework for negotiating autonomous agents', in *Proc. of Modelling Autonomous Agents in a Multi-Agent World MAAMAW99*. LNAI1647, Springer-Verlag, (1999).
[18] A. Tarski, 'A lattice-theoretical fixpoint theorem and its applications', *Pacific Journal of Mathematics*, **5**, 285–309, (1955).
[19] G.A.W. Vreeswijk, 'Abstract argumentation systems', *Artificial Intelligence*, **90**(1–2), 225–279, (1997).
[20] G. Wagner, *Vivid Logic – Knowledge-Based Reasoning with Two Kinds of Negation*, volume LNAI 764, Springer–Verlag, 1994.