

# On the Relationship of Defeasible Argumentation and Answer Set Programming

Matthias Thimm   Gabriele Kern-Isberner

Technische Universität Dortmund

May 29, 2008

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming
- 3 Properties of warrant
- 4 Answer Set Programming
- 5 Converting a de.l.p. into an answer set program
- 6 Conclusion

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming
- 3 Properties of warrant
- 4 Answer Set Programming
- 5 Converting a de.l.p. into an answer set program
- 6 Conclusion

# The motivation

- Both, Defeasible Logic Programming and Answer Set Programming use logic programming as a representation mechanism
- While logic programming in general is a well understood framework, argumentation frameworks are still under heavy development
- Although the relationship of argumentation and default logic has been investigated using abstract argumentation frameworks, we are trying to investigate a direct link between DeLP and ASP

Our aim is to express the set of warranted literals of a defeasible logic program directly in terms of answer set semantics to get a better understanding of the relationships of their inference mechanisms.

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming**
- 3 Properties of warrant
- 4 Answer Set Programming
- 5 Converting a de.l.p. into an answer set program
- 6 Conclusion

# A very brief overview

- in DeLP (*Defeasible Logic Programming*) we are dealing with facts, strict rules and defeasible rules.
- A defeasible logic program (*de.l.p.*)  $\mathcal{P}$  is a tuple  $\mathcal{P} = (\Pi, \Delta)$  with a set  $\Pi$  of facts and strict rules and a set  $\Delta$  of defeasible rules.
- Using defeasible argumentation via a dialectical analysis one can determine warrants and warranted literals.

## Definition (Warrant)

A literal  $h$  is *warranted*, iff there exists an argument  $\langle \mathcal{A}, h \rangle$  for  $h$ , such that the root of the marked dialectical tree  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}^*$  is marked “undefeated”.  
Then  $\langle \mathcal{A}, h \rangle$  is a *warrant* for  $h$ .

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming
- 3 Properties of warrant**
- 4 Answer Set Programming
- 5 Converting a de.l.p. into an answer set program
- 6 Conclusion

# Warranting arguments

In general, a warrant  $\langle \mathcal{A}, h \rangle$  is not unbeatable, i. e. it does not hold: “If an argument  $\langle \mathcal{A}, h \rangle$  is undefeated in the dialectical tree  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$ , then it is undefeated in every dialectical tree”.

# Warranting arguments

In general, a warrant  $\langle \mathcal{A}, h \rangle$  is not unbeatable, i. e. it does not hold: “If an argument  $\langle \mathcal{A}, h \rangle$  is undefeated in the dialectical tree  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$ , then it is undefeated in every dialectical tree”.

But

## Proposition

*If an argument  $\langle \mathcal{A}, h \rangle$  is undefeated in the dialectical tree  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$ , then it is undefeated in every dialectical tree  $\mathcal{T}_{\langle \mathcal{A}', h' \rangle}$ , where  $\langle \mathcal{A}, h \rangle$  is a child of  $\langle \mathcal{A}', h' \rangle$ .*

and therefore

## Proposition

*If  $h$  and  $h'$  are warranted literals in a de.l.p.  $\mathcal{P}$ , then  $h$  and  $h'$  cannot disagree.*

# Joint disagreement 1/2

Although two warranted literals are consistent, this is not always the case for sets of more than two warranted literals.

## Definition (Joint disagreement)

If  $\{h_1, \dots, h_n\} \cup \Pi \sim \perp$ , then  $h_1, \dots, h_n$  are in *joint disagreement*.

## Example

Let *de.l.p.*  $\mathcal{P} = (\Pi, \Delta)$  with

$$\Pi = \{a, (h \leftarrow c, d), (\neg h \leftarrow e, f)\}$$

$$\Delta = \{(c \multimap a), (d \multimap a), (e \multimap a), (f \multimap a)\}$$

$\Rightarrow c, d, e, f$  are warranted (assuming a suitable preference relation under arguments) and in joint disagreement.

## Joint disagreement 2/2

Some sets of warranted literals can never be in joint disagreement as the following two propositions show.

### Proposition

*Let  $\langle \mathcal{A}, h \rangle$  be an argument such that  $\{h, h_1, \dots, h_n\} = \{\text{head}(r) \mid r \in \mathcal{A}\}$ . Then  $h, h_1, \dots, h_n$  do not jointly disagree.*

It follows

### Proposition

*Let  $\mathcal{P}$  be a de.l.p. If  $h$  is a warranted literal in  $\mathcal{P}$  and  $\langle \mathcal{A}, h \rangle$  is a warrant for  $h$ , then  $h'$  is warranted in  $\mathcal{P}$  for every subargument  $\langle \mathcal{B}, h' \rangle$  of  $\langle \mathcal{A}, h \rangle$ .*

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming
- 3 Properties of warrant
- 4 Answer Set Programming**
- 5 Converting a de.l.p. into an answer set program
- 6 Conclusion

# Overview

Extended logic programs (Gelfond, Lifschitz) use default negation to handle uncertainty and to realize non-monotonic reasoning.

## Definition (Extended logic program)

An *extended logic program* (*program* for short)  $P$  is a finite set of rules of the form

$$h \leftarrow a_1, \dots, a_n, \text{not } b_1, \dots, \text{not } b_m$$

# Answer sets

Let  $X$  be a set of literals.

## Definition (Reduct)

The  $X$ -*reduct* of a program  $P$  ( $P^X$ ) is the union of all rules  $h \leftarrow a_1, \dots, a_n$  such that  $h \leftarrow a_1, \dots, a_n$ , not  $b_1, \dots$ , not  $b_m \in P$  and  $X \cap \{b_1, \dots, b_m\} = \emptyset$ .

The reduct is used to characterize a set of literals as an answer set:

## Definition (Answer set)

A consistent set of literals  $S$  is an *answer set* of a program  $P$ , iff  $S$  is the minimal model of  $P^S$ .

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming
- 3 Properties of warrant
- 4 Answer Set Programming
- 5 Converting a de.l.p. into an answer set program**
- 6 Conclusion

# Minimal disagreement, guard rules

To preserve consistency in answer sets, sets of warranted literals that are in joint disagreement have to be handled appropriately.

## Definition (Minimal disagreement set)

A *minimal disagreement set*  $\mathcal{X}$  is a set of derivable literals such that  $\mathcal{X} \cup \Pi \vdash \perp$  and there is no proper subset  $\mathcal{X}'$  of  $\mathcal{X}$  with  $\mathcal{X}' \cup \Pi \vdash \perp$ . Let  $\mathfrak{X}(\mathcal{P})$  be the set of all minimal disagreement sets of  $\mathcal{P}$ .

## Definition (Guard literals, guard rules)

The set of *guard literals*  $GuardLit(\mathcal{P})$  for  $\mathcal{P}$  is defined as  $GuardLit(\mathcal{P}) = \{\alpha_h \mid h \text{ is a literal in } \mathcal{P}\}$  with new symbols  $\alpha_h$ . The set of *guard rules*  $GuardRules(\mathcal{P})$  of  $\mathcal{P}$  is defined as  $GuardRules = \{\alpha_h \leftarrow h_1, \dots, h_n \mid \{h, h_1, \dots, h_n\} \in \mathfrak{X}(\mathcal{P})\}$ .

# Induced answer set programs

## Definition (*de.lp*-induced answer set program)

The  $\mathcal{P}$ -induced answer set program  $ASP(\mathcal{P})$  is defined as the minimal extended logic program satisfying

- 1 for every  $a \in \Pi$  it is  $a \in ASP(\mathcal{P})$ ,
- 2 for every  $r : h \leftarrow b_1, \dots, b_n \in \Pi$  it is  $r \in ASP(\mathcal{P})$ ,
- 3 for every  $h \prec b_1, \dots, b_n \in \Delta$  it is  $h \leftarrow b_1, \dots, b_n$ , not  $\alpha_h \in ASP(\mathcal{P})$   
and
- 4  $GuardRules(\mathcal{P}) \subseteq ASP(\mathcal{P})$ .

# An example

## Example

Let  $\mathcal{P} = (\Pi, \Delta)$  with

$$\Pi = \{a, b, (h \leftarrow c, d), (\neg h \leftarrow e)\}$$

$$\Delta = \{(p \multimap a), (\neg p \multimap b), (c \multimap b), (d \multimap b), (e \multimap a)\}$$

Here we have  $\{(\alpha_h \leftarrow \neg h), (\alpha_{\neg h} \leftarrow c, d), (\alpha_c \leftarrow d, \neg h), (\alpha_c \leftarrow d, e), (\alpha_d \leftarrow c, e)\} \subseteq \text{GuardRules}(\mathcal{P})$ .

# An example

## Example

Let  $\mathcal{P} = (\Pi, \Delta)$  with

$$\Pi = \{a, b, (h \leftarrow c, d), (\neg h \leftarrow e)\}$$

$$\Delta = \{(p \multimap a), (\neg p \multimap b), (c \multimap b), (d \multimap b), (e \multimap a)\}$$

Here we have  $\{(\alpha_h \leftarrow \neg h), (\alpha_{\neg h} \leftarrow c, d), (\alpha_c \leftarrow d, \neg h), (\alpha_c \leftarrow d, e), (\alpha_d \leftarrow c, e)\} \subseteq \text{GuardRules}(\mathcal{P})$ .

The  $\mathcal{P}$ -induced answer set program  $\text{ASP}(\mathcal{P})$  arises as

$$\begin{aligned} \text{ASP}(\mathcal{P}) = & \{a, b, (h \leftarrow c, d), (\neg h \leftarrow e), (p \leftarrow a, \text{not } \alpha_p), \\ & (\neg p \leftarrow b, \text{not } \alpha_{\neg p}), (c \leftarrow b, \text{not } \alpha_c), \\ & (d \leftarrow b, \text{not } \alpha_d), (e \leftarrow a, \text{not } \alpha_e)\} \cup \text{GuardRules}(\mathcal{P}) \end{aligned}$$

# Results

It can be shown that sets of warranted literals and answer sets are related:

## Theorem

*Let  $\mathcal{P} = (\Pi, \Delta)$  be a de.l.p. and  $ASP(\mathcal{P})$  the  $\mathcal{P}$ -induced answer set program. If  $h$  is warranted in  $\mathcal{P}$  then there exists at least one answer set  $M$  of  $ASP(\mathcal{P})$  with  $h \in M$ .*

# Results

It can be shown that sets of warranted literals and answer sets are related:

## Theorem

*Let  $\mathcal{P} = (\Pi, \Delta)$  be a de.l.p. and  $\text{ASP}(\mathcal{P})$  the  $\mathcal{P}$ -induced answer set program. If  $h$  is warranted in  $\mathcal{P}$  then there exists at least one answer set  $M$  of  $\text{ASP}(\mathcal{P})$  with  $h \in M$ .*

For a special case it follows

## Corollary

*Let  $\mathcal{P} = (\Pi, \Delta)$  be a de.l.p. and  $\text{ASP}(\mathcal{P})$  the  $\mathcal{P}$ -induced answer set program. If  $\Pi$  does not contain any strict rule and  $M$  is the set of all warranted literals of  $\mathcal{P}$  then there exists an answer set  $M'$  of  $\text{ASP}(\mathcal{P})$  with  $M \subseteq M'$ .*

# Induced\* answer set programs 1/3

## Definition (*de.l.p.*\*-induced answer set program)

The  $\mathcal{P}^*$ -induced answer set program  $\text{ASP}^*(\mathcal{P})$  is defined as the minimal extended logic program satisfying

- ① for every  $a \in \Pi$  it is  $a \in \text{ASP}^*(\mathcal{P})$  and
- ② for every (strict or defeasible) rule  $h \leftarrow\!\!\leftarrow b_1, \dots, b_n \in \Pi \cup \Delta$  it is  $h \leftarrow b_1, \dots, b_n$ , not  $b'_1, \dots$ , not  $b'_m \in \text{ASP}^*(\mathcal{P})$  where  $\{b'_1, \dots, b'_m\} = \{b \mid b \text{ and } h \text{ disagree}\}$ .

## Theorem

Let  $\mathcal{P} = (\Pi, \Delta)$  be a *de.l.p.*. Let furthermore  $\text{ASP}^*(\mathcal{P})$  be the  $\mathcal{P}^*$ -induced answer set program. If  $M$  is the set of all warranted literals of  $\mathcal{P}$ , then there exists an answer set  $M'$  of  $\text{ASP}^*(\mathcal{P})$  with  $M \subseteq M'$ .

# Outline

- 1 Motivation
- 2 Defeasible Logic Programming
- 3 Properties of warrant
- 4 Answer Set Programming
- 5 Converting a de.l.p. into an answer set program
- 6 Conclusion**

# Conclusion

- we studied transformations of defeasible logic programs into answer set programs in order to make relationships between their inference mechanisms explicit
- we proved that for our conversion, warrant implies credulous inference
- for the second type of conversion, all warranted literals are in one answer set

# Conclusion

- we studied transformations of defeasible logic programs into answer set programs in order to make relationships between their inference mechanisms explicit
- we proved that for our conversion, warrant implies credulous inference
- for the second type of conversion, all warranted literals are in one answer set

*Thank you for your attention*

# Appendix I: Comparing arguments

Arguments can be compared, e. g. using *Generalized Specificity*.

## Example

- Let  $a, b$  be facts. Then

$$\begin{aligned} \langle \{(c \leftarrow a, b)\}, c \rangle &\succ_{spec} \langle \{(\neg c \leftarrow a)\}, \neg c \rangle \\ \langle \{(d \leftarrow a)\}, d \rangle &\succ_{spec} \langle \{(c \leftarrow a), (\neg d \leftarrow c)\}, \neg d \rangle \end{aligned}$$

→ *proper attacks*

- Arguments might be incomparable

$$\begin{aligned} \langle \{(c \leftarrow a)\}, c \rangle &\not\succeq_{spec} \langle \{(\neg c \leftarrow b)\}, \neg c \rangle \\ \langle \{(c \leftarrow a)\}, c \rangle &\not\succeq_{spec} \langle \{(\neg c \leftarrow b)\}, \neg c \rangle \end{aligned}$$

→ *blocking attacks*

# Appendix II: Induced\* answer set programs

## Example

Let  $\mathcal{P} = (\Pi, \Delta)$  with

$$\Pi = \{a, b, (h \leftarrow c, d), (\neg h \leftarrow e)\}$$

$$\Delta = \{(p \multimap a), (\neg p \multimap b), (c \multimap b), (d \multimap b), (e \multimap a)\}$$

$\rightarrow \{a, b, c, d\}$  are warranted (using *Generalized Specificity*)

The  $\mathcal{P}^*$ -induced answer set program  $\text{ASP}^*(\mathcal{P})$  arises as

$$\begin{aligned} \text{ASP}^*(\mathcal{P}) = \{ & a, b, (h \leftarrow c, d, \text{not } \neg h, \text{not } e), (\neg h \leftarrow e, \text{not } h, ), \\ & (p \multimap a, \text{not } \neg p), (\neg p \multimap b, \text{not } p), (c \multimap b), \\ & (d \multimap b), (e \multimap a, \text{not } h)\} \end{aligned}$$

$\rightarrow$  The answer sets of  $\text{ASP}^*(\mathcal{P})$  are  $\{a, b, c, d, e, \neg h, p\}$ ,  
 $\{a, b, c, d, e, \neg h, \neg p\}$ ,  $\{a, b, c, d, h, p\}$ ,  $\{a, b, c, d, h, \neg p\}$

## Appendix III: Proofs 1/7

## Proposition

*If an argument  $\langle \mathcal{A}, h \rangle$  is undefeated in the dialectical tree  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$ , then it is undefeated in every dialectical tree  $\mathcal{T}_{\langle \mathcal{A}', h' \rangle}$ , where  $\langle \mathcal{A}, h \rangle$  is a child of  $\langle \mathcal{A}', h' \rangle$ .*

## Proof.

- the subtree rooted at  $\langle \mathcal{A}, h \rangle$  after  $\langle \mathcal{A}', h' \rangle$  is a subtree of  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$
- every “needed” supporting argument of  $\langle \mathcal{A}, h \rangle$  in  $\mathcal{T}_{\langle \mathcal{A}, h \rangle}$  is in  $\mathcal{T}_{\langle \mathcal{A}', h' \rangle}$
- $\langle \mathcal{A}, h \rangle$  is undefeated in  $\mathcal{T}_{\langle \mathcal{A}', h' \rangle}$



## Appendix III: Proofs 2/7

## Proposition

*If  $h$  and  $h'$  are warranted literals in a de.l.p.  $\mathcal{P}$ , then  $h$  and  $h'$  cannot disagree.*

## Proof.

- suppose  $h, h'$  disagree
- let  $\langle \mathcal{A}, h \rangle, \langle \mathcal{A}', h' \rangle$  be warrants
- wlog  $\langle \mathcal{A}, h \rangle$  attacks  $\langle \mathcal{A}', h' \rangle$
- due to last proposition,  $\langle \mathcal{A}, h \rangle$  is undefeated in dial. tree of  $\langle \mathcal{A}', h' \rangle$
- $\langle \mathcal{A}', h' \rangle$  is defeated, hence no warrant.



## Appendix III: Proofs 3/7

## Proposition

Let  $\langle \mathcal{A}, h \rangle$  be an argument such that  $\{h, h_1, \dots, h_n\} = \{\text{head}(r) \mid r \in \mathcal{A}\}$ . Then  $h, h_1, \dots, h_n$  do not jointly disagree.

## Proof.

As  $\langle \mathcal{A}, h \rangle$  is an argument,  $\Pi \cup \mathcal{A}$  is non-contradictory and thus does not cause the derivation of complementary literals. As  $\Pi \cup \mathcal{A} \vdash h, h_1, \dots, h_n$  the literals  $h, h_1, \dots, h_n$  do not jointly disagree.  $\square$

## Appendix III: Proofs 4/7

## Proposition

*Let  $\mathcal{P}$  be a de.i.p. If  $h$  is a warranted literal in  $\mathcal{P}$  and  $\langle \mathcal{A}, h \rangle$  is a warrant for  $h$ , then  $h'$  is warranted in  $\mathcal{P}$  for every subargument  $\langle \mathcal{B}, h' \rangle$  of  $\langle \mathcal{A}, h \rangle$ .*

Show the contraposition:

## Proposition

*Let  $\mathcal{P}$  be a de.i.p. and  $\langle \mathcal{B}, h' \rangle$  an argument. If  $\langle \mathcal{B}, h' \rangle$  is defeated in a dialectical process, every argument  $\langle \mathcal{A}, h \rangle$ , such that  $\langle \mathcal{B}, h' \rangle$  is a subargument of  $\langle \mathcal{A}, h \rangle$ , is also defeated in a dialectical process.*

## Appendix III: Proofs 5/7

## Proof.

- let  $\langle \mathcal{B}, h' \rangle$  be defeated in its dialectical tree and  $\langle \mathcal{C}, h'' \rangle$  a defeater
- $\langle \mathcal{C}, h'' \rangle$  is also an attack on  $\langle \mathcal{A}, h \rangle$
- the tree rooted at  $\langle \mathcal{C}, h'' \rangle$  under  $\langle \mathcal{A}, h \rangle$  is a subtree of the tree rooted at  $\langle \mathcal{C}, h'' \rangle$  under  $\langle \mathcal{B}, h' \rangle$
- there is no  $\langle \mathcal{D}, g \rangle$  in the tree rooted at  $\langle \mathcal{C}, h'' \rangle$  and interfering with  $\langle \mathcal{B}, h' \rangle$  in the dial. tree of  $\langle \mathcal{B}, h' \rangle$  that is not in the dial. tree of  $\langle \mathcal{A}, h \rangle$ , provided its parentnode exists in the dial. tree of  $\langle \mathcal{A}, h \rangle$
- hence the subtree rooted at  $\langle \mathcal{C}, h'' \rangle$  under  $\langle \mathcal{A}, h \rangle$  “loses” no needed interfering arguments
- hence  $\langle \mathcal{C}, h'' \rangle$  defeats  $\langle \mathcal{A}, h \rangle$



## Appendix III: Proofs 6/7

## Theorem

Let  $\mathcal{P} = (\Pi, \Delta)$  be a de.l.p. and  $\text{ASP}(\mathcal{P})$  the  $\mathcal{P}$ -induced answer set program. If  $h$  is warranted in  $\mathcal{P}$  then there exists at least one answer set  $M$  of  $\text{ASP}(\mathcal{P})$  with  $h \in M$ .

## Proof.

- the set  $S$  of literals appearing in a warrant  $\langle \mathcal{A}, h \rangle$  do not jointly disagree
- hence  $S$  can be extended to a consistent set  $M$ , such that  $M$  is an answer set of  $\text{ASP}(\mathcal{P})$



## Appendix III: Proofs 7/7

## Corollary

Let  $\mathcal{P} = (\Pi, \Delta)$  be a de.l.p. and  $\text{ASP}(\mathcal{P})$  the  $\mathcal{P}$ -induced answer set program. If  $\Pi$  does not contain any strict rule and  $M$  is the set of all warranted literals of  $\mathcal{P}$  then there exists an answer set  $M'$  of  $\text{ASP}(\mathcal{P})$  with  $M \subseteq M'$ .

## Proof.

- there can be no disagreement sets with cardinality  $> 2$
- no two warranted literals can disagree
- hence  $M$  is consistent and can consistently be extended to an answer set  $M'$

