ECI 34 VIRTUAL

26-30 / JULIO / 2021

ESCUELA DE CIENCIAS INFORMÁTICAS

Structured Argumentation: Defeasible Logic Programming

- Guillermo R. Simari

LIDIA

Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA)

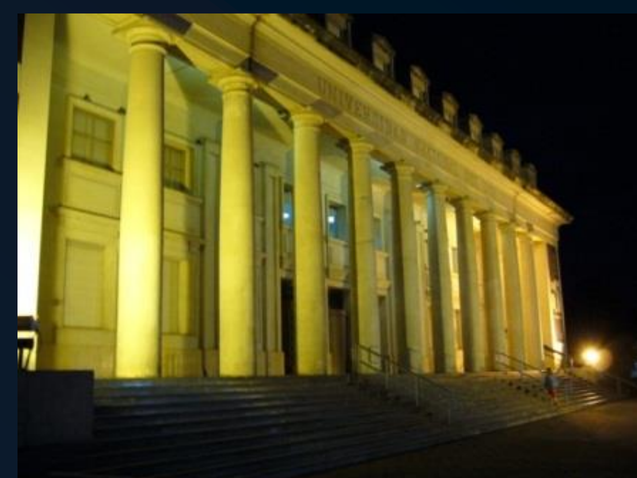
Instituto de Ciencias e Ingeniería de la Computación

Departamento de Ciencias e Ingeniería de la Computación

UNIVERSIDAD NACIONAL DEL SUR

Bahia Blanca - ARGENTINA

ICIC CONICET UNS



Structured Argumentation

See Argument & Computation, Vol 5 No 1, 2014 for a set of tutorials on Structured Argumentation.

Conceptual View

Definition of Status of Arguments
Definition of Defeat among Arguments
Definition of Conflict among Arguments
Definition of Argument
Definition of the Underlying (Logical) Language

Defeasible Logic Programming (DeLP)

García, A.J., Simari, G.R.: Defeasible logic programming: An argumentative approach. Theory and Practice of Logic Programming 4(1-2), 95–138. 2004

Also, see Argument & Computation Vol 5 No 1, 2014 for a set of tutorials on Structured Argumentation.

Arguments and Trees

- *As we have said, an argument is a piece of reasoning that supports, from certain evidence, a claim Q.*
- *The tenability of this claim must be confirmed by analyzing other arguments for (support) and against (interference) such claim in a dialectical process.*
- *These arguments are connected through the defeat relation and could be organized in dialectical trees.*
- *Observe that, given a claim there could exist different arguments that support that claim, and for each argument there will be a different dialectical tree.*

Introduction

- *The inference engine is based in a defeasible argumentation inference mechanism for warranting the conclusions.*
- *In the language of DeLP there is the possibility of representing information in the form of weak rules in a declarative manner.*
- *Weak rules represent a key element for introducing defeasibility and they are used to represent a defeasible relationship between pieces of knowledge.*
- *This connection could be defeated after all things are considered.*

Introduction

- *General Common Sense reasoning should be defeasible in a way that is not explicitly programmed.*
- *Rejection should be the result of the global consideration of the corpus of knowledge that the agent performing such reasoning has at his disposal.*
- *Defeasible Argumentation provides a way of doing that.*

Conceptual View

Let's begin with the accepted view that there are five common elements in systems for defeasible argumentation:

Definition of Status of Arguments

Definition of Defeat among Arguments

Definition of Conflict among Arguments

Definition of Argument

Definition of the Underlying (Logical) Language

Conceptual View

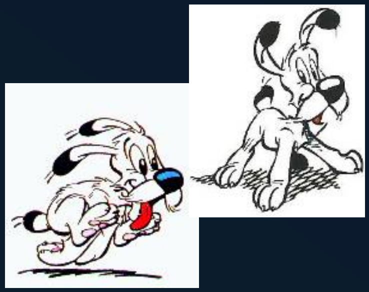
Definition of Status of Arguments
Definition of Defeat among Arguments
Definition of Conflict among Arguments
Definition of Argument
Definition of the Underlying (Logical) Language

DeLP's Language

- *DeLP considers two kinds of program rules: defeasible rules to represent tentative information such as*

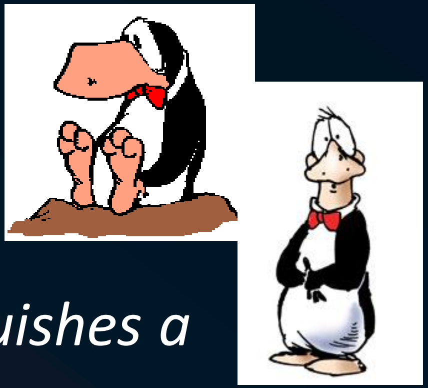
$$\sim flies(dumbo) \prec elephant(dumbo)$$

and strict rules used to represent strict knowledge such as



$$mammal(idefix) \leftarrow dog(idefix)$$

$$bird(opus) \leftarrow penguin(opus)$$



- *Syntactically, the symbol "$\prec$" is all that distinguishes a defeasible rule from a strict one.*

- *Pragmatically, a defeasible rule is used to represent knowledge that could be used when nothing can be posed against it.*

Facts and Strict and Defeasible Rules

- *A Fact is a ground literal:* $innocent(joe)$
- *A Strict Rule is denoted:* $L_0 \leftarrow L_1,\ L_2\ ,\ \ldots,\ L_n$

  where L_0 *is a ground literal called the Head of the rule and* $L_1,\ L_2,\ \ldots,\ L_n$ *are ground literals which form its Body.*
- *This kind of rule is used to represent a relation between the head and the body which is not defeasible.*

  Examples:

$$\sim guilty(joe) \leftarrow innocent(joe)$$

$$mammal(garfield) \leftarrow cat(garfield)$$

Facts and Strict and Defeasible Rules

- A *Defeasible Rule* is denoted: $L_0 \prec L_1, L_2, \ldots, L_n$

- This kind of rule is used to represent a relation between the head and the body of the rule which is tentative and its intuitive interpretation is:

  "Reasons to believe in $L_1, L_2, \ldots, L_n$ are reasons to believe in L_0"

  Examples:

$$flies(tweety) \prec bird(tweety)$$

$$\sim good\_weather(today) \prec low\_pressure(today), wind(south)$$

Defeasible Rules

- *Defeasible rules are not default rules.*
- *In a default rule such as*

$$\varphi : \psi_1, \psi_2, \ldots, \psi_n \,/\, \chi$$

the justification part, $\psi_1, \psi_2, \ldots, \psi_n$, *is a consistency check that controls the applicability of this rule.*

Example: $$\frac{elephant(dumbo) \;:\; {\sim}flies(dumbo)}{{\sim}flies(dumbo)}$$

represents the same knowledge as

$${\sim}flies(dumbo) \prec elephant(dumbo)$$

Defeasible Rules

- *The effect of a defeasible rule comes from a dialectical analysis made by the inference mechanism; therefore, there is no need to encode any particular check, even though could be done if necessary.*

- *Changes in the knowledge represented using DeLP's language is reflected with the sole addition of new knowledge to the representation, thus leading to better elaboration tolerance.*

Defeasible Logic Program

A Defeasible Logic Program (delp) is a set of facts, strict rules and defeasible rules denoted

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

where

- Π *is a set of facts and strict rules, and*
- Δ *is a set of defeasible rules.*

Facts, strict, and defeasible rules are ground, i.e., do not contain variables.

Defeasible Logic Program

- *However, we will use schematic rules containing variables.*

  If R is a schematic rule, $Ground(R)$ stands for the set of all ground instances of R and

$$Ground(\mathcal{P}) = \bigcup_{R \in \mathcal{P}} Ground(R)$$

  in all cases the set of individual constants in the language of $\mathcal{P}$ will be used (see V. Lifschitz, Foundations of Logic Programming, in Principles of Knowledge Representation, G. Brewka, Ed., 1996, FOLLI)

Here is an example of a *Defeasible Logic Program* (*delp*) denoted $\mathcal{P} = (\Pi, \Delta)$, where Π is a set of facts and strict rules, and Δ is a set of defeasible rules.

Π

Strict Rules

$$\left\{ \begin{array}{ll} bird(X) \leftarrow chicken(X) & chicken(tina) \\ bird\ (X) \leftarrow penguin(X) & penguin(opus) \\ {\sim}flies(X) \leftarrow penguin(X) & scared(tina) \end{array} \right.$$

Facts

Δ

Defeasible Rules

$$\left\{ \begin{array}{l} flies(X) \prec bird(X) \\ {\sim}flies(X) \prec chicken(X) \\ flies(X) \prec chicken\ (X),\ scared(X) \end{array} \right.$$

Defeasible Logic Programming: DeLP

Here is another example of a $\mathcal{P} = (\Pi, \Delta)$

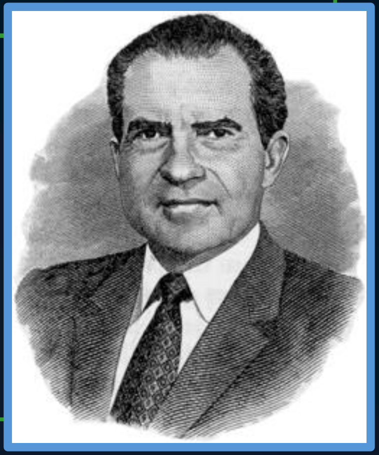
Δ

Defeasible Rules

$$\left\{\begin{array}{l} has\_a\_gun(X) \prec lives\_in\_chicago(X) \\ \sim has\_a\_gun(X) \prec lives\_in\_chicago(X), \\ \qquad\qquad\qquad\qquad pacifist(X) \\ pacifist(X) \prec quaker(X) \\ \sim pacifist(X) \prec republican(X) \end{array}\right.$$

Π

$$\left.\begin{array}{l} lives\_in\_chicago(nixon) \\ quaker(nixon) \\ republican(nixon) \end{array}\right\} \textit{Facts}$$



Adapted from Prakken and Vreeswijk (2000)

Defeasible Logic Programming: DeLP

Still one more example of a $\mathcal{P} = (\Pi, \Delta)$

Δ

Defeasible Rules

$$\left\{\begin{array}{l} buy\_shares(X) \prec good\_price(X) \\ \sim buy\_shares(X) \prec good\_price(X),\ risky(X) \\ risky(X) \prec in\_fusion(X,\ Y) \\ risky(X) \prec in\_debt(X) \\ \sim risky(X) \prec in\_fusion(X,\ Y),\ strong(Y) \end{array}\right.$$

Π

$$\left.\begin{array}{l} good\_price(acme) \\ in\_fusion(acme,\ estron) \\ strong(estron) \end{array}\right\}$$

Facts

Conceptual View

Definition of Status of Arguments
Definition of Defeat among Arguments
Definition of Conflict among Arguments
Definition of Argument
Definition of the Underlying (Logical) Language

Defeasible Derivation

Def: *Let $\mathcal{P}$ = (Π, Δ) be a program and L a ground literal. A defeasible derivation $\mathcal{P} \mid\!\sim L$ of L from $\mathcal{P}$, is a finite sequence of ground literals*

$$L_1,\ L_2,\ \ldots,\ L_n = L,$$

s.t. each literal L_k ($1 \leq i < k \leq n$) is there because:

- *L_k is a fact in Π, or*
- *there is a rule (strict or defeasible) in $\mathcal{P}$ with head L_k and body $B_1, B_2, \ldots, B_j$ where every literal B_j ($1 \leq j$) in the body is some L_i appearing previously in the sequence ($i < k$).*

Defeasible Derivation

- *Notice that defeasible derivation differs from standard logical, or strict, derivation only in the use of defeasible, or weak, rules.*

- *Given a Defeasible Logic Program, a derivation for a literal L is called defeasible because there may exist information that contradicts L, or the way that L is inferred is attacked; in that case, the acceptance of L as a valid conclusion will be questioned.*

- *A few examples of defeasible derivations follow.*

Defeasible Derivation

From the program:

$$
\begin{array}{ll}
bird(X) \leftarrow chicken(X). & chicken(tina). \\
bird(X) \leftarrow penguin(X). & penguin(opus). \\
\sim flies(X) \leftarrow penguin(X). & scared(tina). \\
flies(X) \prec bird(X). & \\
\sim flies(X) \prec chicken(X). & \\
flies(X) \prec chicken(X), scared(X). &
\end{array}
$$

The following are some derivations that could be obtained:

- $chicken(tina), bird(tina), flies(tina).$
- $chicken(tina), \sim flies(tina).$
- $penguin(opus), bird(opus), flies(opus).$
- $penguin(opus), \sim flies(opus).$

i.e., $\mathcal{P} \mathrel{|\!\!\sim} flies(tina)$, and $\mathcal{P} \mathrel{|\!\!\sim} \sim flies(opus)$

Defeasible Derivation

From the program:

$$buy\_shares(X) \prec good\_price(X).$$
$$\sim buy\_shares\ (X) \prec good\_price(X), risky(X).$$
$$risky(X) \prec in\_fusion(X, Y).$$
$$risky(X) \prec in\_debt(X).$$
$$\sim risky(X) \prec in\_fusion(X, Y), strong(Y).$$
$$good\_price(acme).$$
$$in\_fusion(acme, estron).$$
$$strong(estron).$$

The following derivations could be obtained:

- $good\_price(acme), buy\_shares(acme).$
- $in\_fusion(acme, estron), risky(acme),$ $good\_price(acme), \sim buy\_shares(acme)$
- $in\_fusion(acme, estron), risky(acme)$
- $in\_fusion(acme, estron), strong(estron), \sim risky(acme)$

Programs and Derivations

- *A program* $\mathcal{P} = (\Pi, \Delta)$ *is contradictory if it is possible to derive from it a pair of complementary literals.*
- *From the examples it is possible to derive literals, such as* $\boxed{flies(tina), \sim flies(tina)}$ *and* $\boxed{flies(opus), \sim flies(opus)}$ *from the first one, and* $\boxed{risky(acme), \sim risky(acme)}$ *and* $\boxed{buy\_shares(acme), \sim buy\_shares(acme)}$ *from the second.*
- *Contradictory programs are useful for representing knowledge that is potentially contradictory.*
- *On the other hand, as a design restriction, the set* Π *should not be contradictory, because in that case the represented knowledge would be inconsistent.*

Defeasible Argumentation

Def: Let L be a literal and $\mathcal{P} = (\Pi, \Delta)$ be a program, we say that $\mathcal{A}$ is an $argument$ for L, denoted $\langle \mathcal{A}, L \rangle$, if $\mathcal{A}$ is a set of rules in Δ such that:

1) *There exists a defeasible derivation of L from $\Pi \cup \mathcal{A}$;*

2) *The set $\Pi \cup \mathcal{A}$ is non contradictory; and*

3) *There is no proper subset $\mathcal{A}'$ of $\mathcal{A}$ such that $\mathcal{A}'$ satisfies 1) and 2), that is, $\mathcal{A}$ is minimal as the defeasible part of the derivation mentioned in 1).*

Defeasible Argumentation

- *That is to say, an argument $\langle \mathcal{A}, L \rangle$, or an argument $\mathcal{A}$ for L, is a minimal, non contradictory set of defeasible rules, that can be extracted from a defeasible derivation of L.*

- *Strict rules are not part of the argument.*

- *Note that for any L which is derivable from Π alone, the empty set $\emptyset$ is an argument for L (i.e. $\langle \emptyset, L \rangle$); in this case, there is no otherposible argument for L.*

$poor\_perf(john).\quad sick(john).$

$good\_perf(peter).\ unruly(peter).$

$suspend(X) \mathrel{-\!\!\!\!\prec} {\sim}responsible(X).$

$suspend(X) \mathrel{-\!\!\!\!\prec} unruly(X).$

${\sim}suspend(X) \mathrel{-\!\!\!\!\prec} responsible(X).$

${\sim}responsible(X) \mathrel{-\!\!\!\!\prec} poor\_perf(X).$

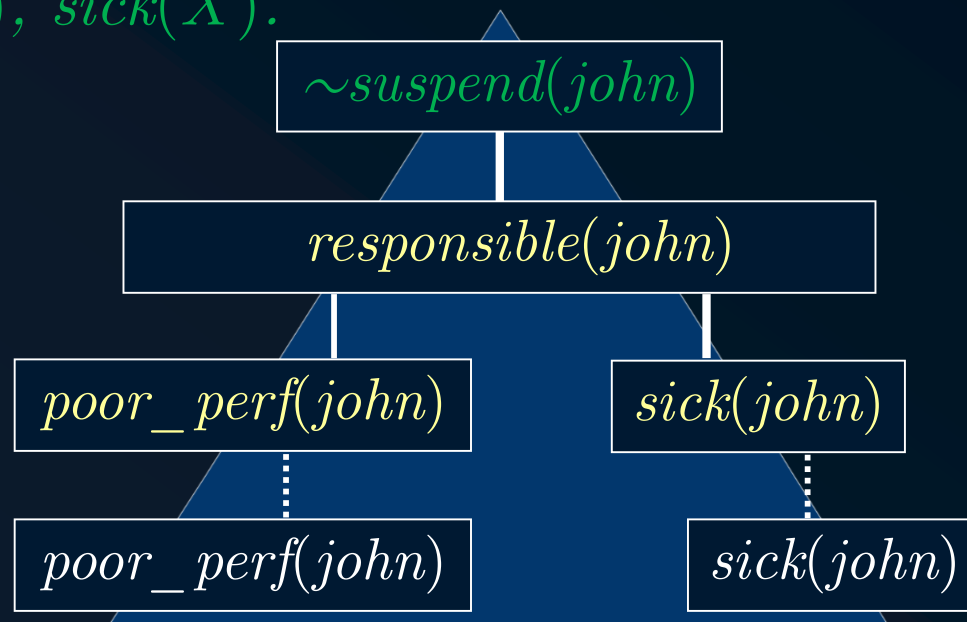
$responsible(X) \mathrel{-\!\!\!\!\prec} good\_perf(X).$

$responsible(X) \mathrel{-\!\!\!\!\prec} poor\_perf(X),\ sick(X).$

An argument for

${\sim}suspend(john)$

built from the program above.



$\langle\{{\sim}suspend(john) \mathrel{-\!\!\!\!\prec} responsible(john).,$
$\quad responsible(john) \mathrel{-\!\!\!\!\prec} poor\_perf(john),\ sick(john).\},\ {\sim}suspend(john)\rangle$

$poor\_perf(john).\quad sick(john).$

$good\_perf(peter).\ unruly(peter).$

$suspend(X) \prec \sim responsible(X).$

$suspend(X) \prec unruly(X).$

$\sim suspend(X) \prec responsible(X).$

$\sim responsible(X) \prec poor\_perf(X).$

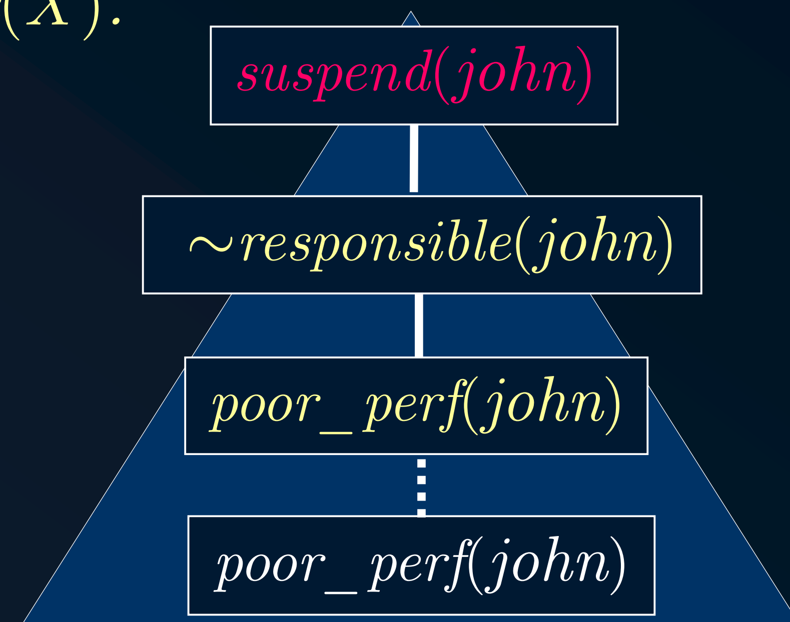
$responsible(X) \prec good\_perf(X).$

$responsible(X) \prec poor\_perf(X),\ sick(X).$

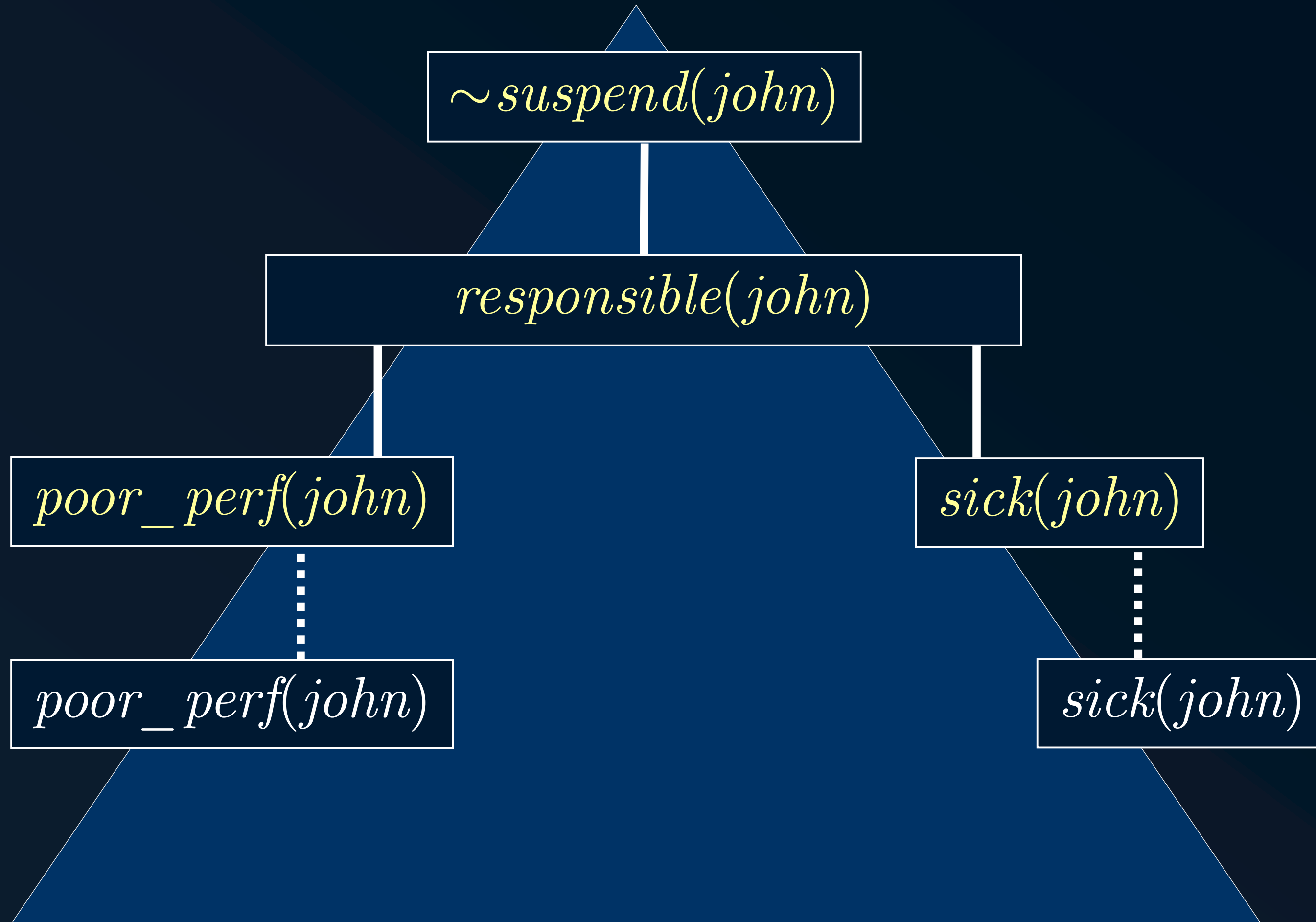
An argument for

$suspend(john)$

built from the program above.

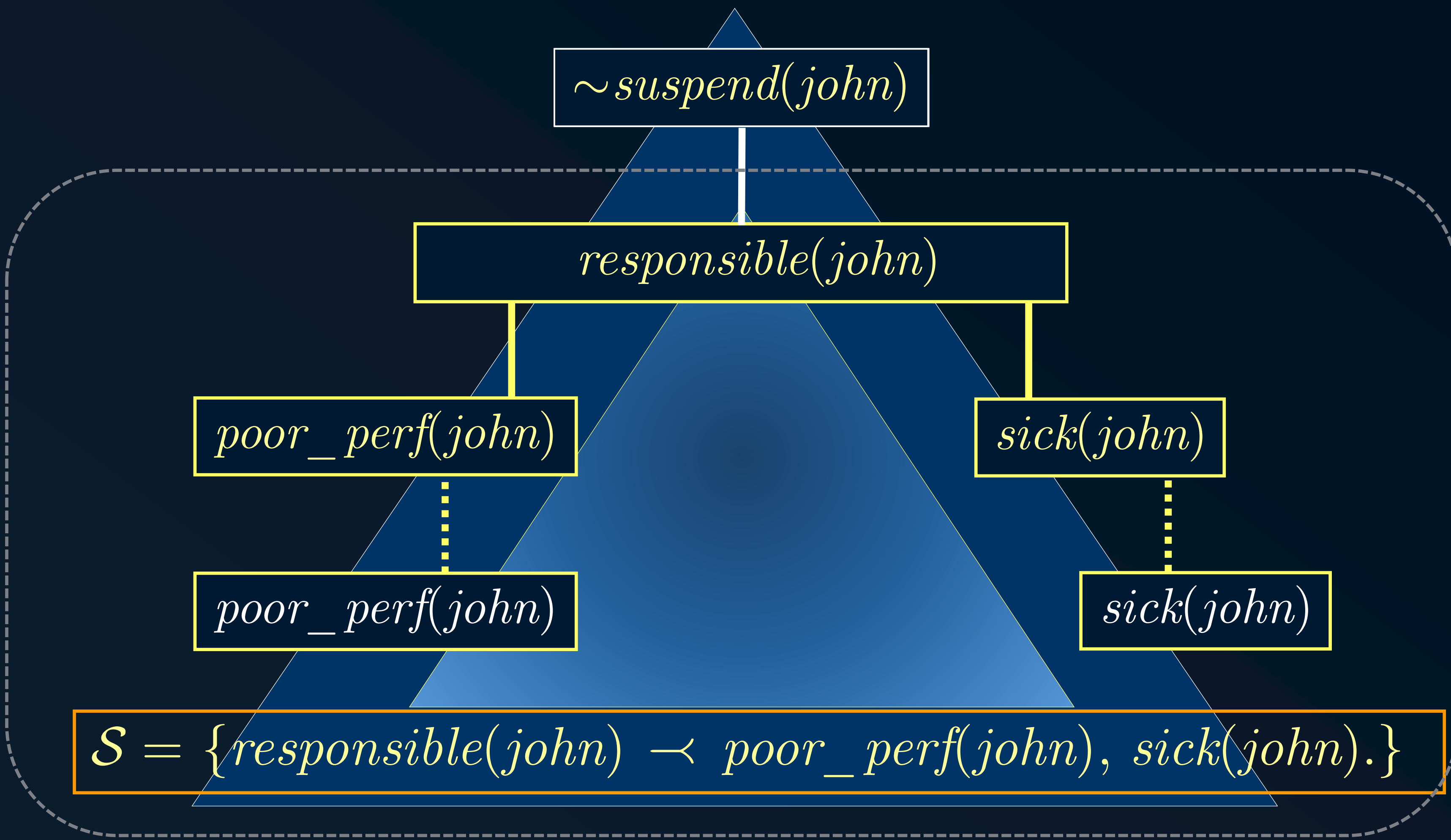


$$\langle\{suspend(john) \prec \sim responsible(john).,\ \sim responsible(john) \prec poor\_perf(john).\},\ suspend(john)\rangle$$



$$\begin{aligned}\mathcal{A} = \{&\sim suspend(john) \prec responsible(john).,\\ &responsible(john) \prec poor\_perf(john),\ sick(john).\}\end{aligned}$$

$\langle \mathcal{S}, Q \rangle$ is a subargument of $\langle \mathcal{A}, L \rangle$ if $\mathcal{S}$ is an argument for Q and $\mathcal{S} \subseteq \mathcal{A}$



$$\mathcal{A} = \{\sim suspend(john) \prec responsible(john).,$$
$$responsible(john) \prec poor\_perf(john), sick(john).\}$$

Conceptual View

Definition of Status of Arguments
Definition of Defeat among Arguments
Definition of Conflict among Arguments
Definition of Argument
Definition of the Underlying (Logical) Language

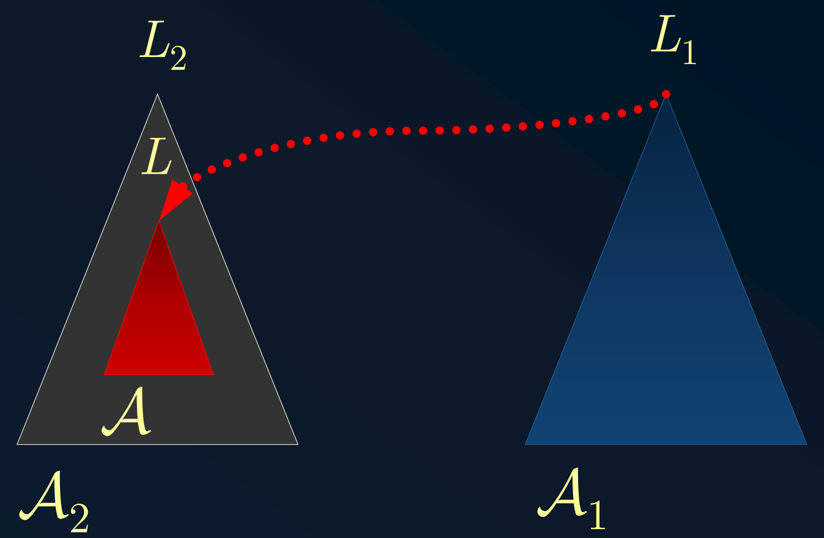
Counter-Arguments or Rebuttals

- *In DeLP, answers are supported by arguments but an argument could be defeated by other arguments.*
- *Informally, a query L will succeed if the supporting argument for it is not defeated.*
- *To study this situation, we consider arguments refered to as counter-arguments or rebuttals.*
- *Counter-arguments are also arguments, and therefore the analysis must be extended to those arguments, and so on.*
- *This analysis is dialectical in nature.*

Def: Let $\mathcal{P} = (\Pi, \Delta)$ *be a program, we will say that two literals* L_1 *and* L_2 $disagree$ *if the set* $\Pi \cup \{L_1, L_2\}$ *is contradictory.*

For example, given $\Pi = \{\sim L_1 \leftarrow L_2,\ L_1 \leftarrow L_3\}$ *the set* $\{L_2,\ L_3\}$ *is contradictory.*

Def: Let $\mathcal{P} = (\Pi, \Delta)$ *be a program, we say that* $\langle \mathcal{A}_1,\ L_1 \rangle$ $counter\text{-}argues$, $rebuts$ or $attacks$ $\langle \mathcal{A}_2,\ L_2 \rangle$ *at literal* L, *if and only if there exists a subargument* $\langle \mathcal{A},\ L \rangle$ *of* $\langle \mathcal{A}_2,\ L_2 \rangle$ *such that* L *and* L_1 *disagree.*



Counter-Arguments or Rebuttals

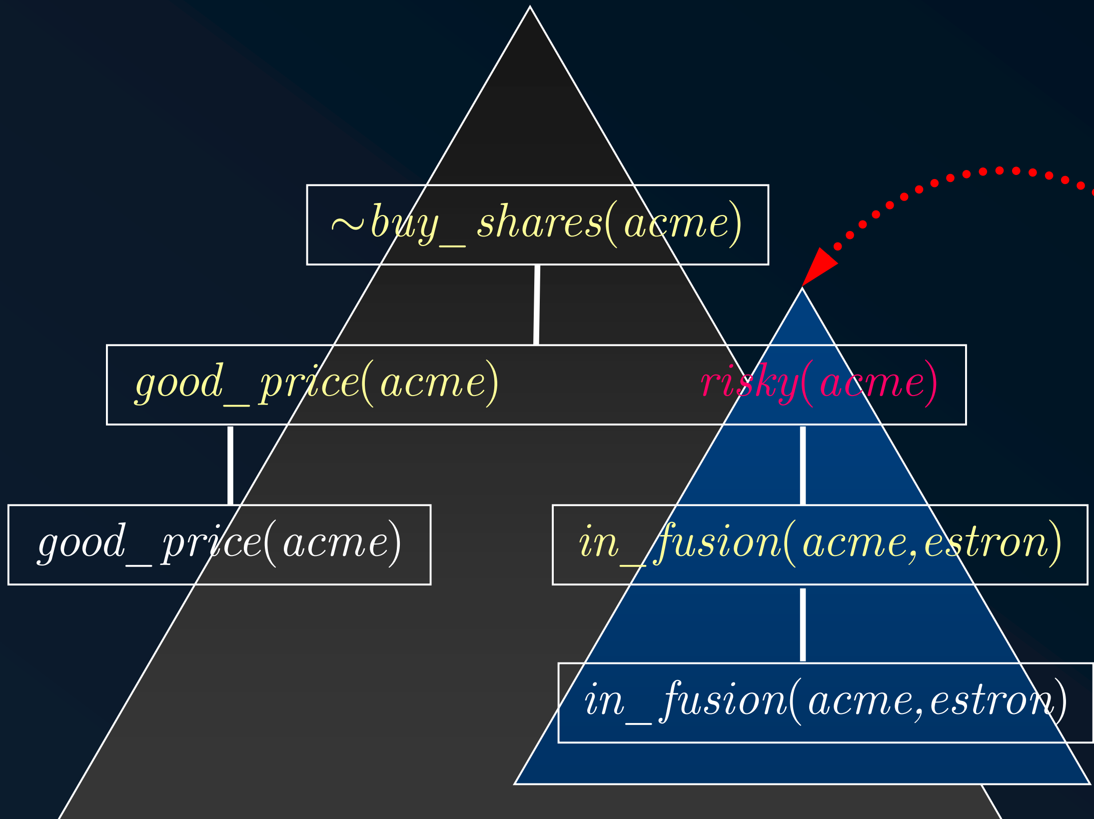
- *Given $\mathcal{P} = (\Pi, \Delta)$, any literal R such that $\Pi \vdash R$, has the support of the empty argument $\langle \emptyset, R \rangle$.*

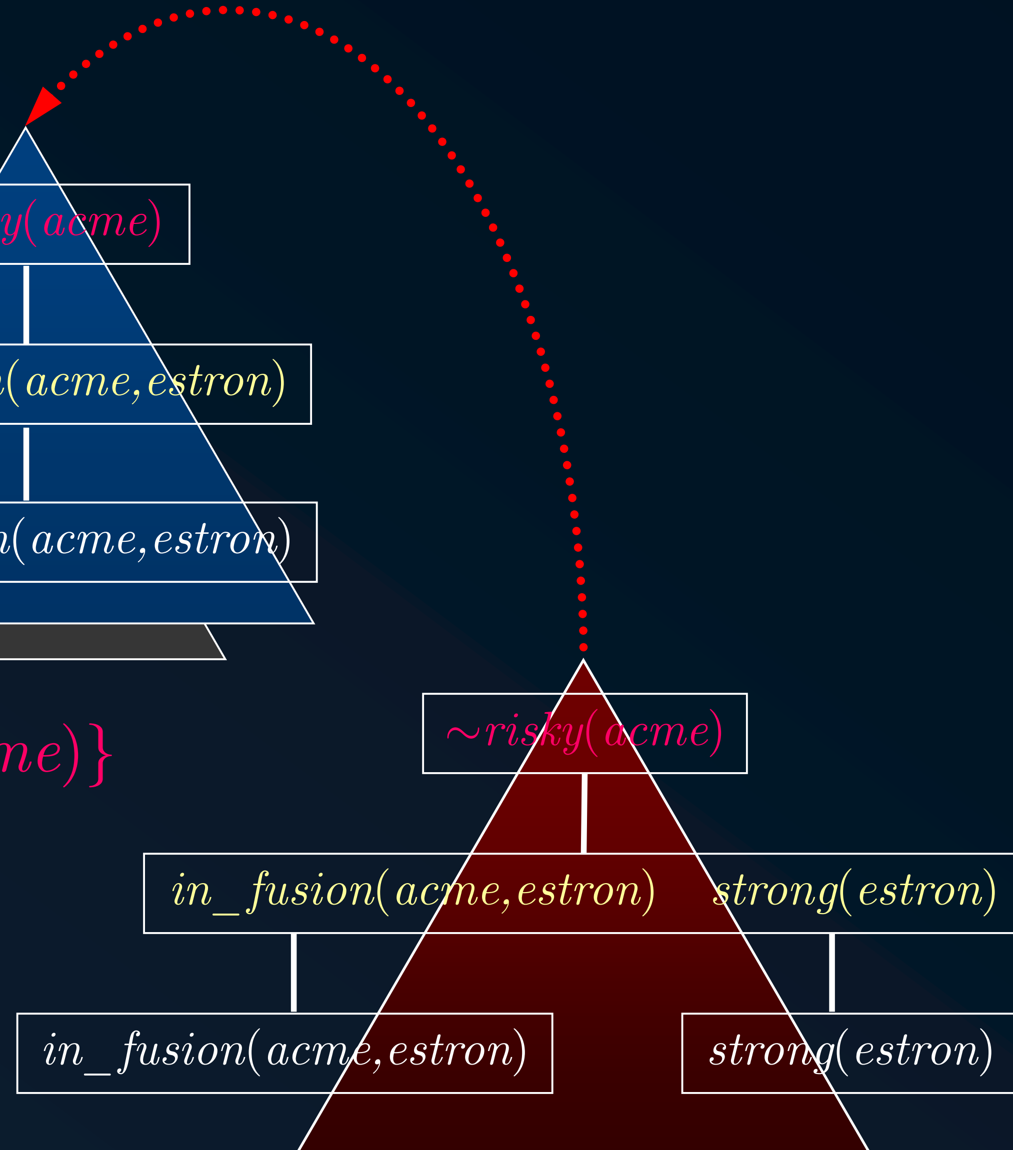
- *There is no posible counter-argument for any of those R since there is no way of constructing an argument which would mention a literal in disagreement with R.*

- *Also, any $\langle \emptyset, R \rangle$ cannot be a counter-argument for any argument $\langle \mathcal{A}, L \rangle$ because of the same reasons.*

- *Note that given an argument $\langle \mathcal{A}, L \rangle$, that argument could contain <u>multiple attack points</u>.*

- *Also, it would be very useful to have some preference criteria to decide between arguments in conflict.*



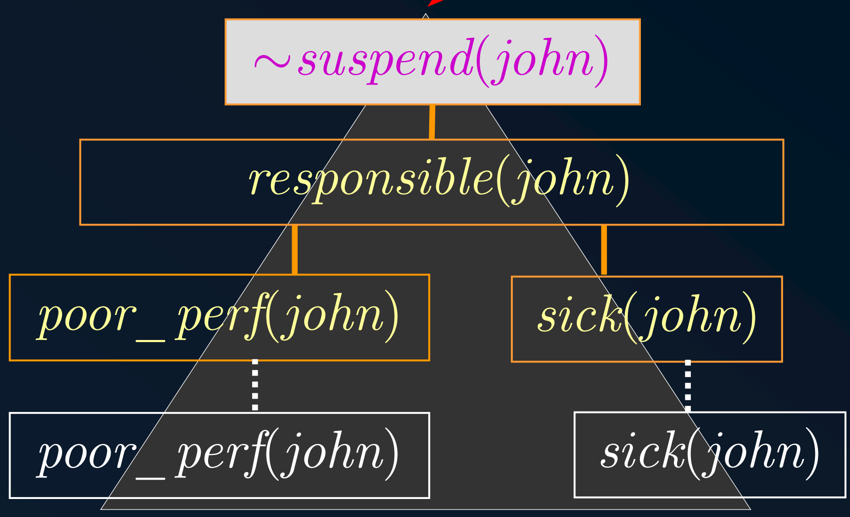


$\Pi \cup \{\ risky(acme),\ \sim risky(acme)\}$

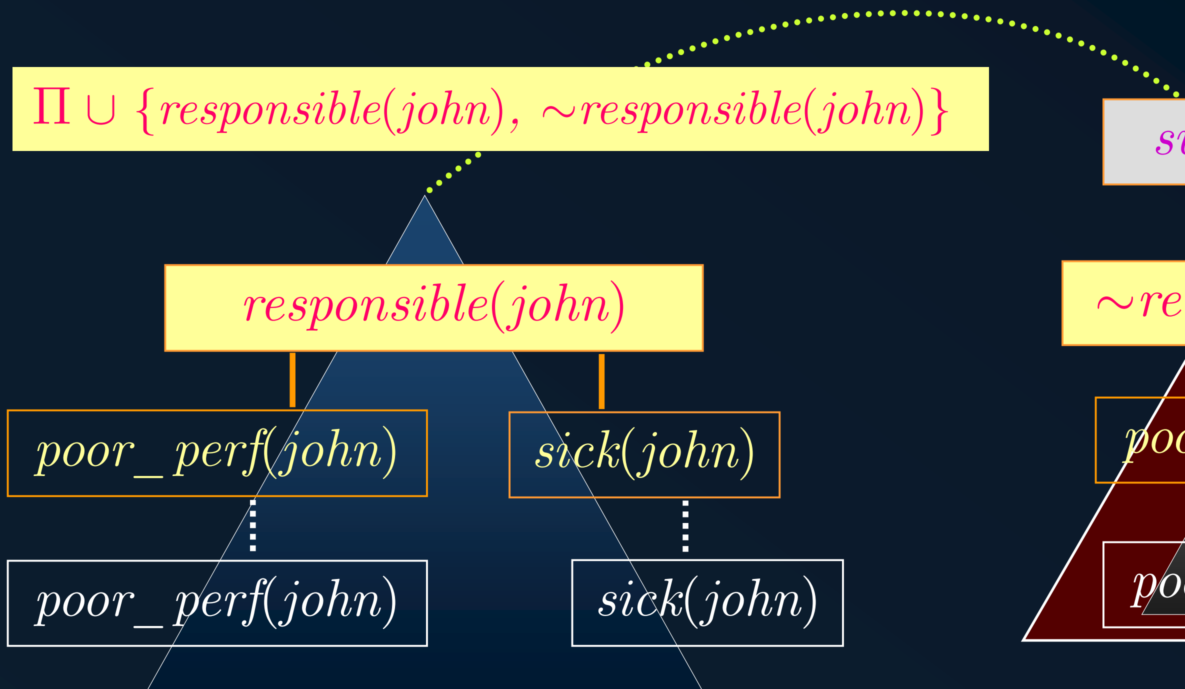
is a contradictory set

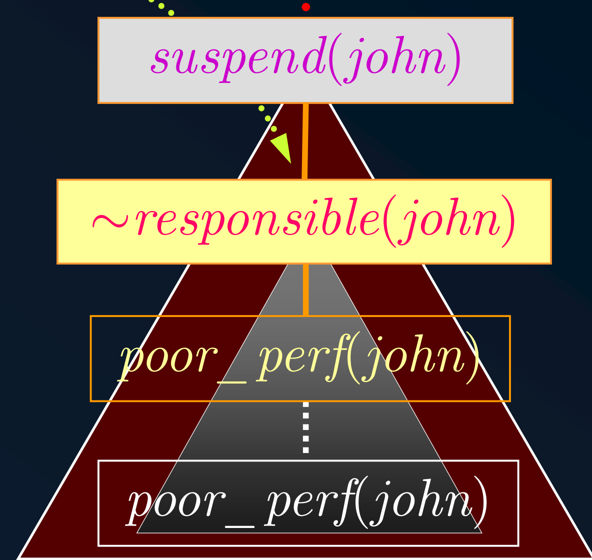
Counter-argument

$\Pi \cup \{suspend(john)\ \sim suspend(john)\}$



$poor\_perf(john).\ \ sick(john).$
$good\_perf(peter).\ \ unruly(peter)$
$suspend(X) \prec \sim responsible(X).$
$suspend(X) \prec unruly(X).$
$suspend(X) \prec \sim responsible(X).$
$\sim suspend(X) \prec responsible(X).$
$\sim responsible(X) \prec poor\_perf(X).$
$responsible(X) \prec good\_perf(X).$
$responsible(X) \prec poor\_perf(X),\ sick(X).$



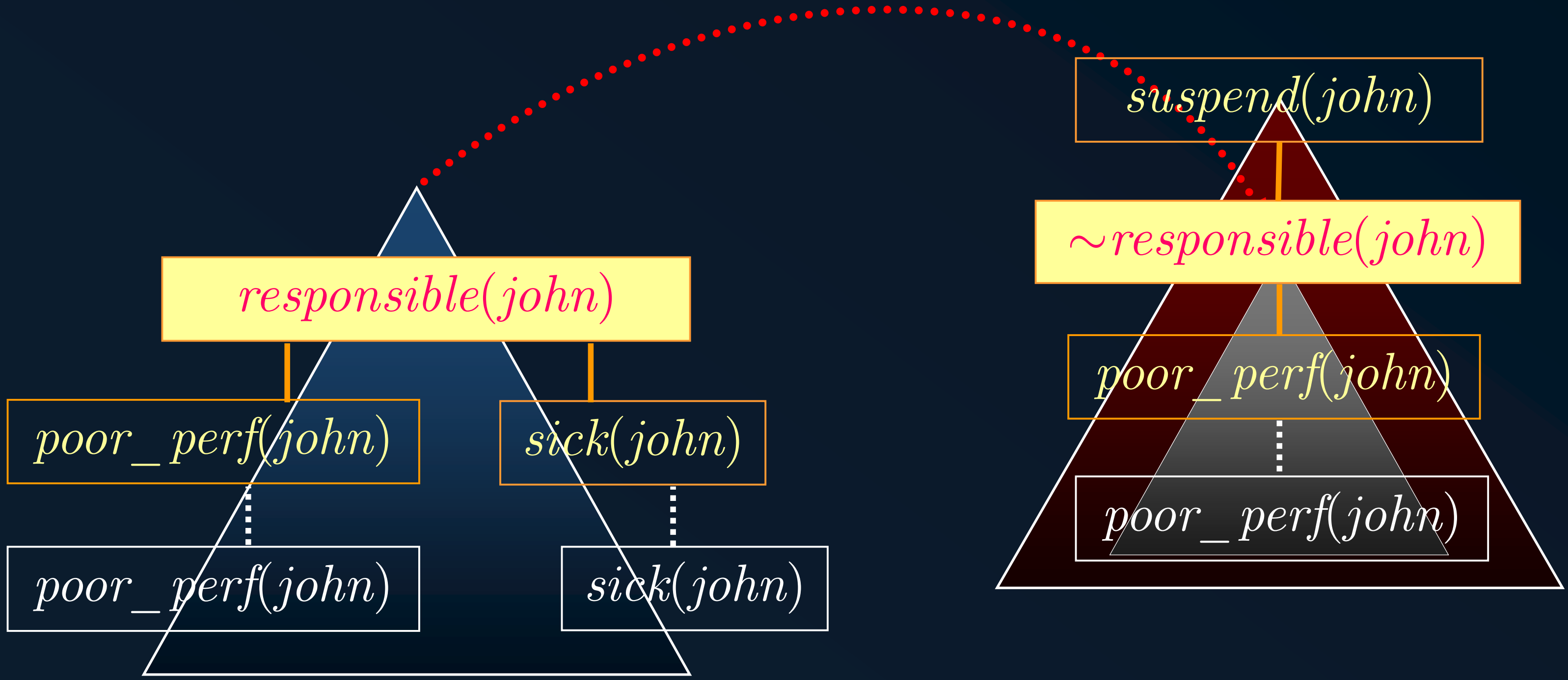


Conceptual View

Definition of Status of Arguments
Definition of Defeat among Arguments
Definition of Conflict among Arguments
Definition of Argument
Definition of the Underlying (Logical) Language

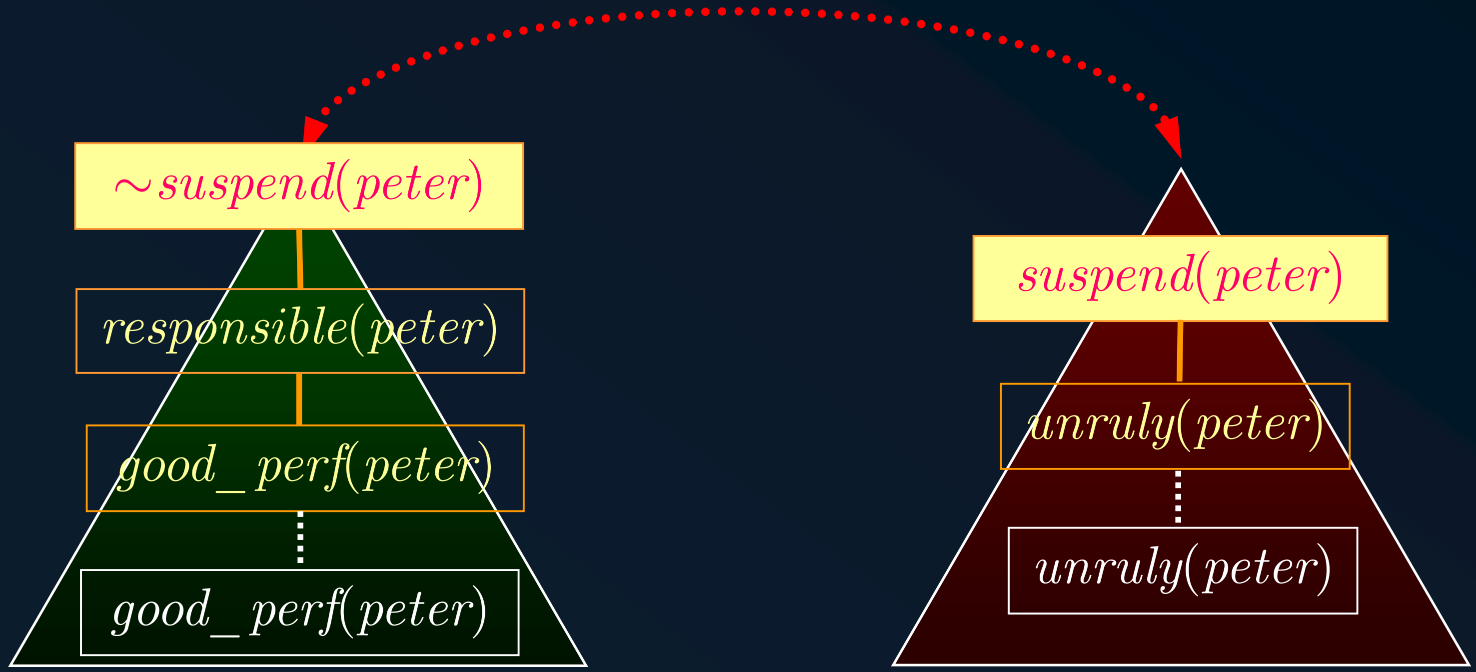
Proper Defeater

An argument $\langle \mathcal{B}, P \rangle$ *is a* $proper\ defeater$ *for* $\langle \mathcal{A}, L \rangle$ *if* $\langle \mathcal{B}, P \rangle$ *is a counter-argument of* $\langle \mathcal{A}, L \rangle$ *that attacks a subargument* $\langle \mathcal{S}, Q \rangle$ *of* $\langle \mathcal{A}, L \rangle$ *and* $\langle \mathcal{B}, P \rangle$ $is\ better\ than$ $\langle \mathcal{S}, Q \rangle$ (*by the chosen comparison criterion*).



Blocking Defeater

An argument $\langle \mathcal{B}, P \rangle$ *is a* $\mathit{blocking\ defeater}$ *for* $\langle \mathcal{A}, L \rangle$ *if* $\langle \mathcal{B}, P \rangle$ *is a counter-argument of* $\langle \mathcal{A}, L \rangle$ *that attacks a subargument* $\langle \mathcal{S}, Q \rangle$ *of* $\langle \mathcal{A}, L \rangle$ *and* $\langle \mathcal{B}, P \rangle$ $\mathit{is\ not\ comparable\ to}$ $\langle \mathcal{S}, Q \rangle$ *(by the chosen comparison criterion).*



Comparison: Generalized Specificity

Def: *Let* $\mathcal{P} = (\Pi, \Delta)$ *be a program, let* Π_G *be the set of strict rules in* Π *and let* $\mathcal{F}$ *be the set of all literals defeasibly derived from* $\mathcal{P}$*. Let* $\langle \mathcal{A}_1, L_1 \rangle$ *and* $\langle \mathcal{A}_2, L_2 \rangle$ *be two arguments built from* $\mathcal{P}$*, where* $L_1, L_2 \in \mathcal{F}$*.*
Then $\langle \mathcal{A}_1, L_1 \rangle$ *is* $\mathit{strictly\ more\ specific\ than}$ $\langle \mathcal{A}_2, L_2 \rangle$ *if:*

1. *For all* $\mathcal{H} \subseteq \mathcal{F}$*, if there exists a defeasible derivation* $\Pi_G \cup \mathcal{H} \cup \mathcal{A}_1 \mid\!\sim L_1$ *while* $\Pi_G \cup \mathcal{H} \not\vdash L_1$ *then* $\Pi_G \cup \mathcal{H} \cup \mathcal{A}_1 \mid\!\sim L_2$*, and*
2. *There exists* $\mathcal{H}' \subseteq \mathcal{F}$ *s.t. there exists a defeasible derivation* $\Pi_G \cup \mathcal{H}' \cup \mathcal{A}_2 \mid\!\sim L_2$ *and* $\Pi_G \cup \mathcal{H}' \not\vdash L_2$ *but* $\Pi_G \cup \mathcal{H}' \cup \mathcal{A}_1 \not\mid\!\sim L_1$

Intuitive view of Specificity - DeLP (Comparison)

More informed arguments (i.e., more precise) are preferred over less informed ones:

$$\langle \{ \sim a \prec b \}, \sim a \rangle \preceq \langle \{ a \prec b, c \}, a \rangle$$

Shorter derivations (i.e. more concise) are preferred over longer derivations:

$$\langle \{ (\sim a \prec b); (b \prec c) \}, \sim a \rangle \preceq \langle \{ a \prec b \}, a \rangle$$

Comparison: Generalized Specificity

For example, from program:

$$bird(X) \leftarrow chicken(X)$$
$$flies(X) \prec bird(X)$$
$$\sim flies(X) \prec chicken(X)$$
$$flies(X) \prec chicken(X),\ scared(X)$$

$$chicken(tina)$$
$$scared(tina)$$

It is possible to obtain

$$\langle \mathcal{A}_1, \sim flies(tina) \rangle, \mathcal{A}_1 = \{\sim flies(tina) \prec chicken(tina)\ \}$$
$$\langle \mathcal{A}_2, flies(tina) \rangle, \mathcal{A}_2 = \{\ flies(tina) \prec bird(tina)\ \}$$
$$\langle \mathcal{A}_3, flies(tina) \rangle, \mathcal{A}_3 = \{\ flies(tina) \prec chicken(tina), scared(tina)\ \}$$

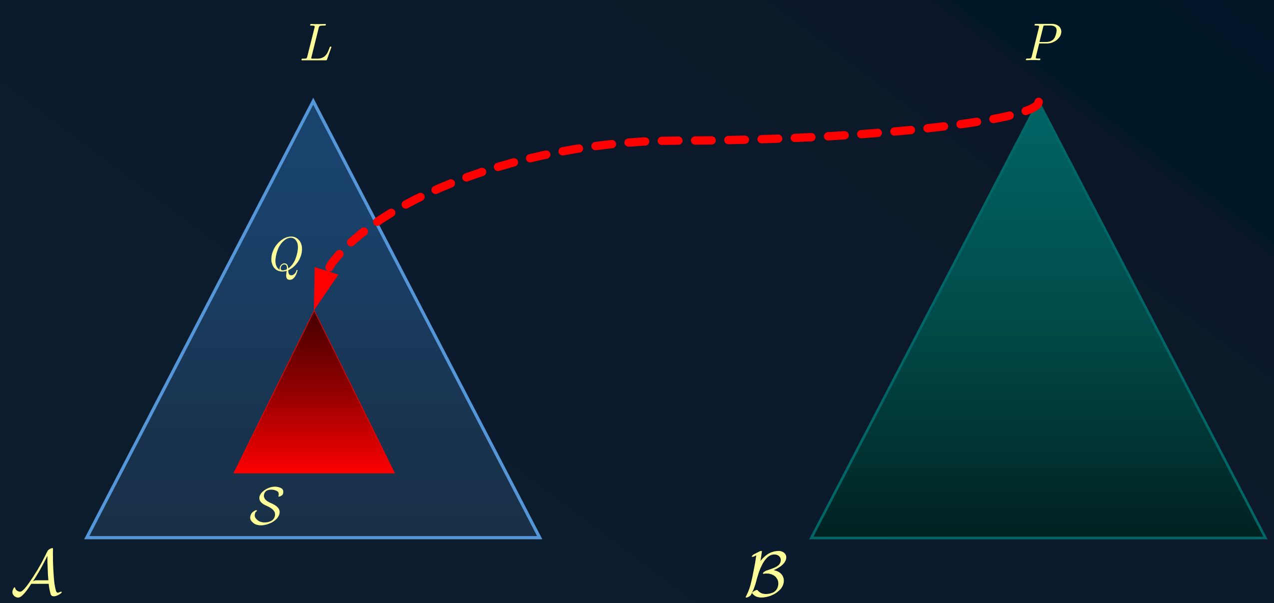
- $\mathcal{A}_3$ *is preferred to* $\mathcal{A}_1$ *because it is more precise.*
- $\mathcal{A}_1$ *is preferred to* $\mathcal{A}_2$ *because it is more concise.*

Defeaters

An argument $\langle \mathcal{B}, P \rangle$ is a defeater for $\langle \mathcal{A}, L \rangle$ if $\langle \mathcal{B}, P \rangle$ is a counter-argument for $\langle \mathcal{A}, L \rangle$ that attacks a subargument $\langle \mathcal{S}, Q \rangle$ of $\langle \mathcal{A}, L \rangle$ and one of the following conditions holds:

a) $\langle \mathcal{B}, P \rangle$ *is better than* $\langle \mathcal{S}, Q \rangle$ (*proper defeater*), or

b) $\langle \mathcal{B}, P \rangle$ *is not comparable to* $\langle \mathcal{S}, Q \rangle$ (*blocking defeater*)



Defeaters: Example

From the program:

$buy\_shares(X) \prec good\_price(X)$	$good\_price(acme)$
$\sim buy\_shares(X) \prec risky(X)$	$in\_fusion(acme, estron)$
$risky(X) \prec in\_fusion(X, Y)$	

With preference:

$$\sim buy\_shares(X) \prec risky(X) > buy\_shares(X) \prec good\_price(X)$$

The argument $\langle \mathcal{A}, \sim buy\_shares(acme)\rangle$ *where*

$$\mathcal{A} = \{\sim buy\_shares(acme) \prec risky(acme),\ risky(acme) \prec in\_fusion(acme, estron)\}$$

is counter-argument of $\langle \mathcal{B}, buy\_shares(acme)\rangle$

where $\mathcal{B} = \{\ buy\_shares(acme) \prec good\_price(acme)\ \}$

that is a proper defeater of it.

Defeaters: Example

From the program:

$$
\begin{array}{l}
pacifist(X) \prec quaker(X) \\
{\sim}pacifist(X) \prec republican(X) \\
quaker(nixon) \\
republican(nixon)
\end{array}
$$

With the preference defined by specificity:

$\langle \mathcal{A}, {\sim}pacifist(nixon) \rangle$ *where*

$$\mathcal{A} = \{{\sim}pacifist(nixon) \prec republican(nixon) \}$$

is a blocking defeater for

$\langle \mathcal{B}, pacifist(nixon) \rangle$ *where*

$$\mathcal{B} = \{ pacifist(nixon) \prec quaker(nixon) \}$$

Conceptual View

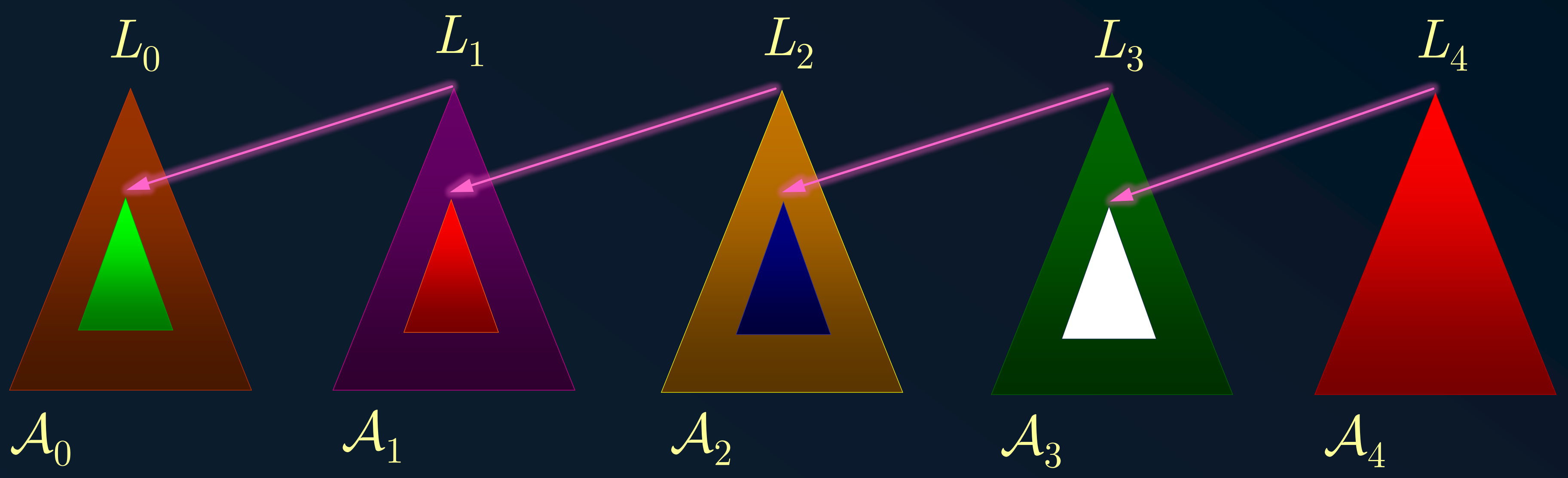
Definition of Status of Arguments
Definition of Defeat among Arguments
Definition of Conflict among Arguments
Definition of Argument
Definition of the Underlying (Logical) Language

Argumentation Line

Given $\mathcal{P} = (\Pi, \Delta)$, *and* $\langle \mathcal{A}_0, L_0 \rangle$ *an argument obtained from* $\mathcal{P}$. *An argumentation line for* $\langle \mathcal{A}_0, L_0 \rangle$ *is a sequence of arguments obtained from* $\mathcal{P}$, *denoted*

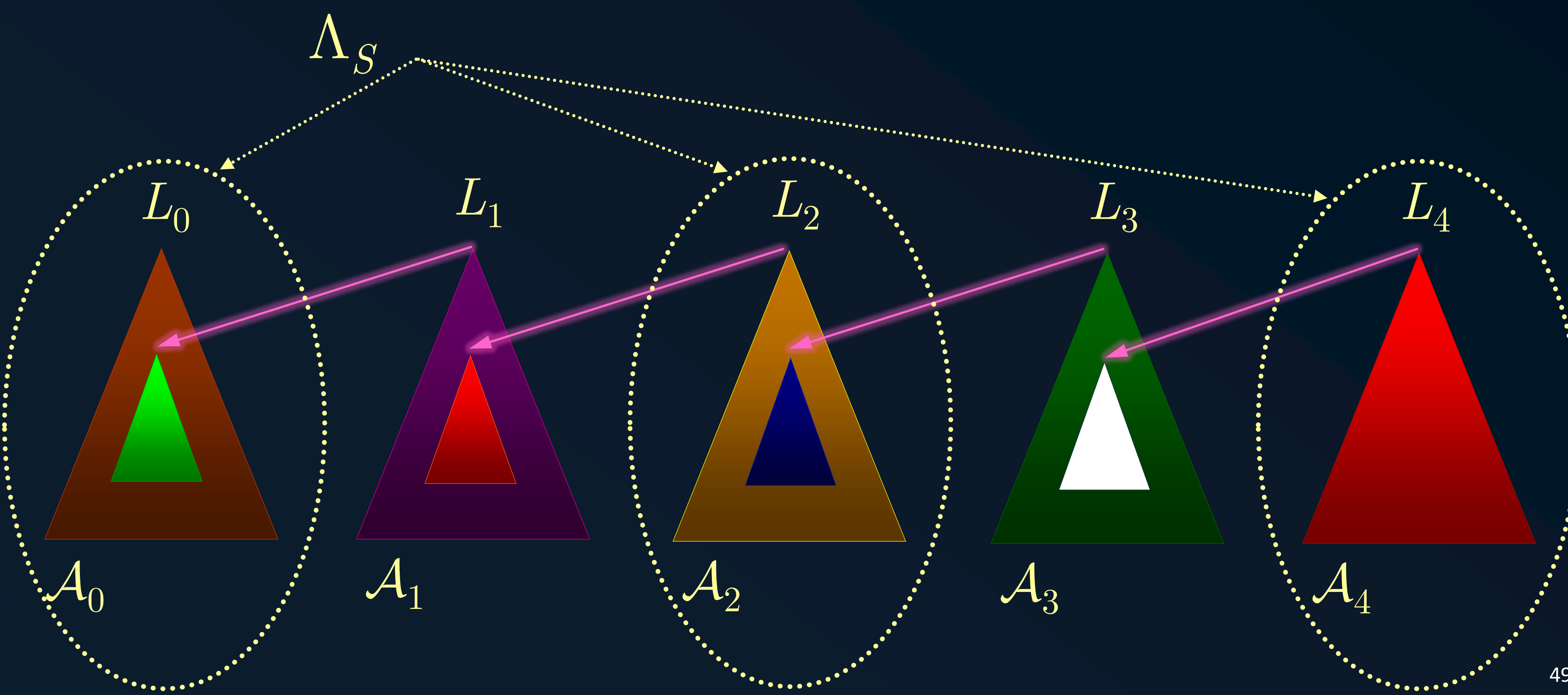
$$\Lambda = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_1, L_1 \rangle, \ldots]$$

where each element in the sequence $\langle \mathcal{A}_i, L_i \rangle, i > 0$ *is a defeater for* $\langle \mathcal{A}_{i\text{-}1}, L_{i\text{-}1} \rangle$.



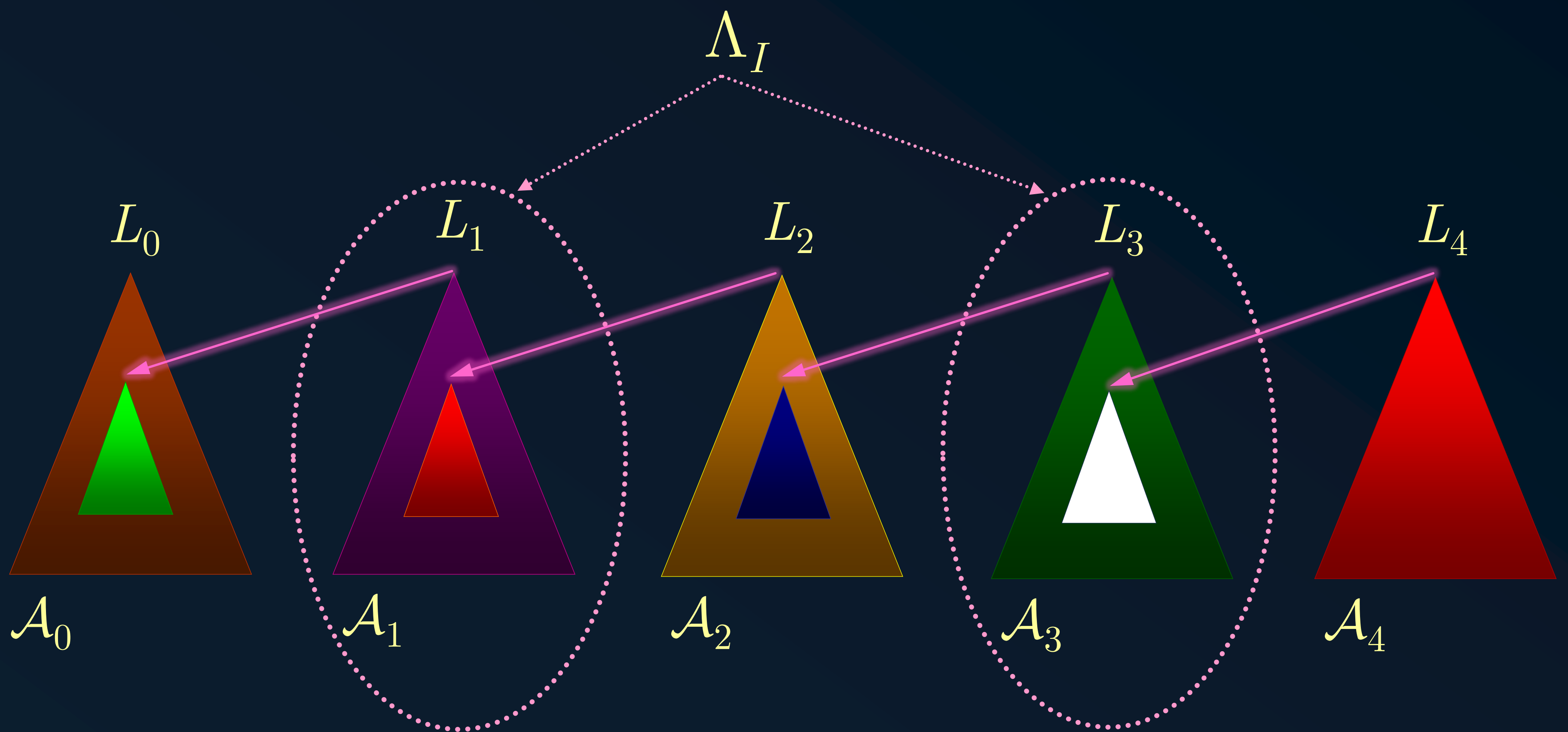
Argumentation Line

Given an argumentation line $\Lambda = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_1, L_1 \rangle, \ldots]$*, the subsequence* $\Lambda_S = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_2, L_2 \rangle, \ldots]$ *contains supporting arguments and* $\Lambda_I = [\langle \mathcal{A}_1, L_1 \rangle, \langle \mathcal{A}_3, L_3 \rangle, \ldots]$ *are interfering arguments.*



Argumentation Line

Given an argumentation line $\Lambda = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_1, L_1 \rangle, \ldots]$, *the subsequence* $\Lambda_S = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_2, L_2 \rangle, \ldots]$ *contains* ***supporting arguments*** *and* $\Lambda_I = [\langle \mathcal{A}_1, L_1 \rangle, \langle \mathcal{A}_3, L_3 \rangle, \ldots]$ *are* ***interfering arguments***.



Argumentation Lines

Assume a program $\mathcal{P}$ where:

$$\begin{array}{lcl}
\langle \mathcal{A}_1, L_1 \rangle & \textit{defeats} & \langle \mathcal{A}_0, L_0 \rangle \\
\langle \mathcal{A}_2, L_2 \rangle & \textit{defeats} & \langle \mathcal{A}_0, L_0 \rangle \\
\langle \mathcal{A}_3, L_3 \rangle & \textit{defeats} & \langle \mathcal{A}_1, L_1 \rangle \\
\langle \mathcal{A}_4, L_4 \rangle & \textit{defeats} & \langle \mathcal{A}_2, L_2 \rangle \\
\langle \mathcal{A}_5, L_5 \rangle & \textit{defeats} & \langle \mathcal{A}_2, L_2 \rangle
\end{array}$$

From $\langle \mathcal{A}_0, L_0 \rangle$ there exist several argumentation lines:

$$\Lambda_1 = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_1, L_1 \rangle, \langle \mathcal{A}_3, L_3 \rangle]$$

$$\Lambda_2 = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_2, L_2 \rangle, \langle \mathcal{A}_4, L_4 \rangle]$$

$$\Lambda_3 = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_2, L_2 \rangle, \langle \mathcal{A}_5, L_5 \rangle]$$

Acceptable Argumentation Line

Given a program $\mathcal{P} = (\Pi, \Delta)$, an argumentation line $\Lambda = [\langle \mathcal{A}_0, L_0 \rangle, \langle \mathcal{A}_1, L_1 \rangle, \ldots]$ will be acceptable if:

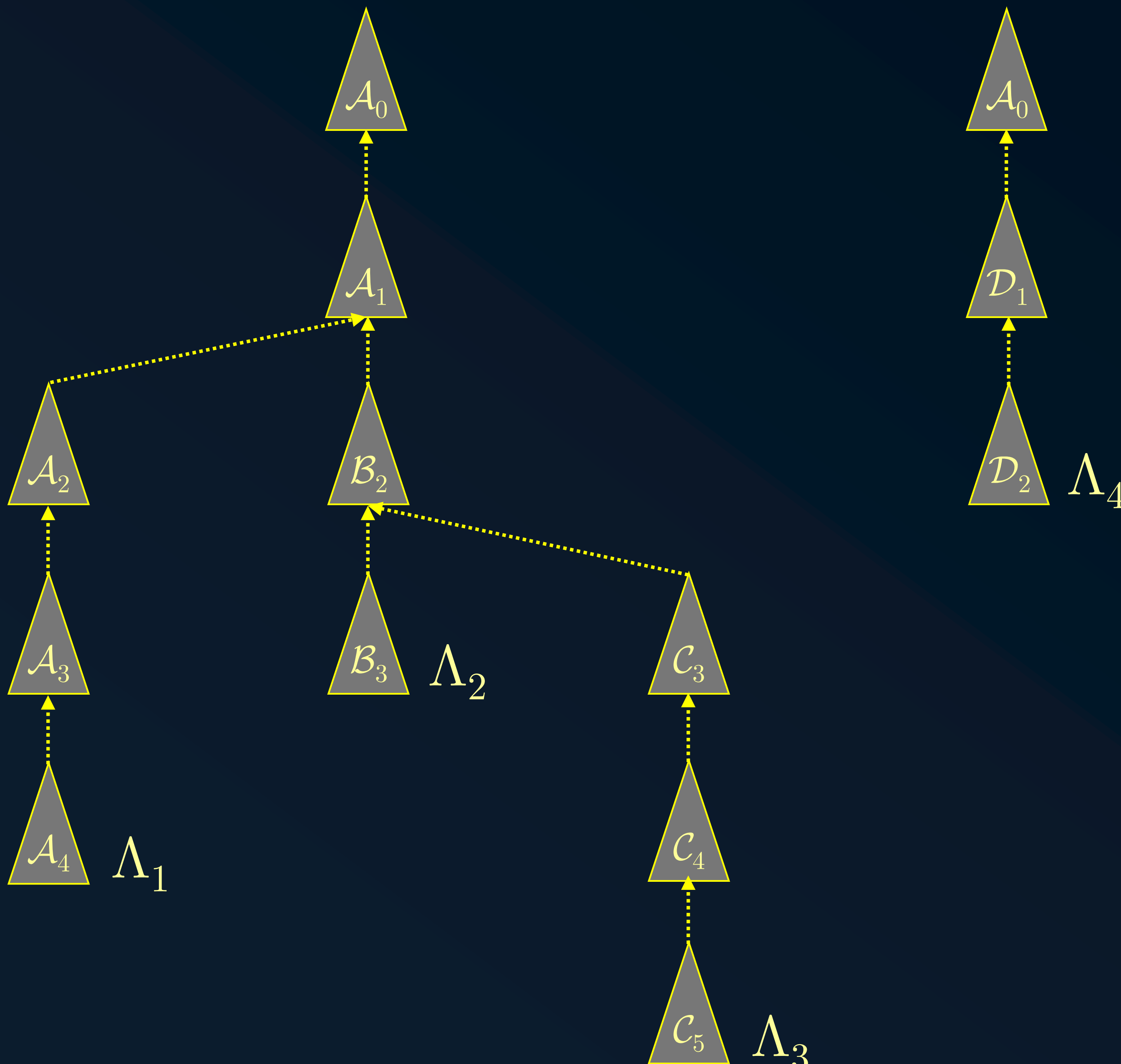
1. Λ *is a finite sequence (non circularity).*
2. *The set Λ_S of supporting arguments is concordant, and the set Λ_I of interfering arguments is concordant.*
3. *No argument $\langle \mathcal{A}_k, L_k \rangle$ in Λ that is a subargument of a preceding argument $\langle \mathcal{A}_i, L_i \rangle$, $i < k$.*
4. *For all i, s.t. $\langle \mathcal{A}_i, L_i \rangle$ is a blocking defeater for $\langle \mathcal{A}_{i\text{-}1}, L_{i\text{-}1} \rangle$, if there exists $\langle \mathcal{A}_{i+1}, L_{i+1} \rangle$ then $\langle \mathcal{A}_{i+1}, L_{i+1} \rangle$ is a proper defeater for $\langle \mathcal{A}, L_i \rangle$ (i.e., $\langle \mathcal{A}, L_i \rangle$ could not be blocked).*

Argumentation Lines

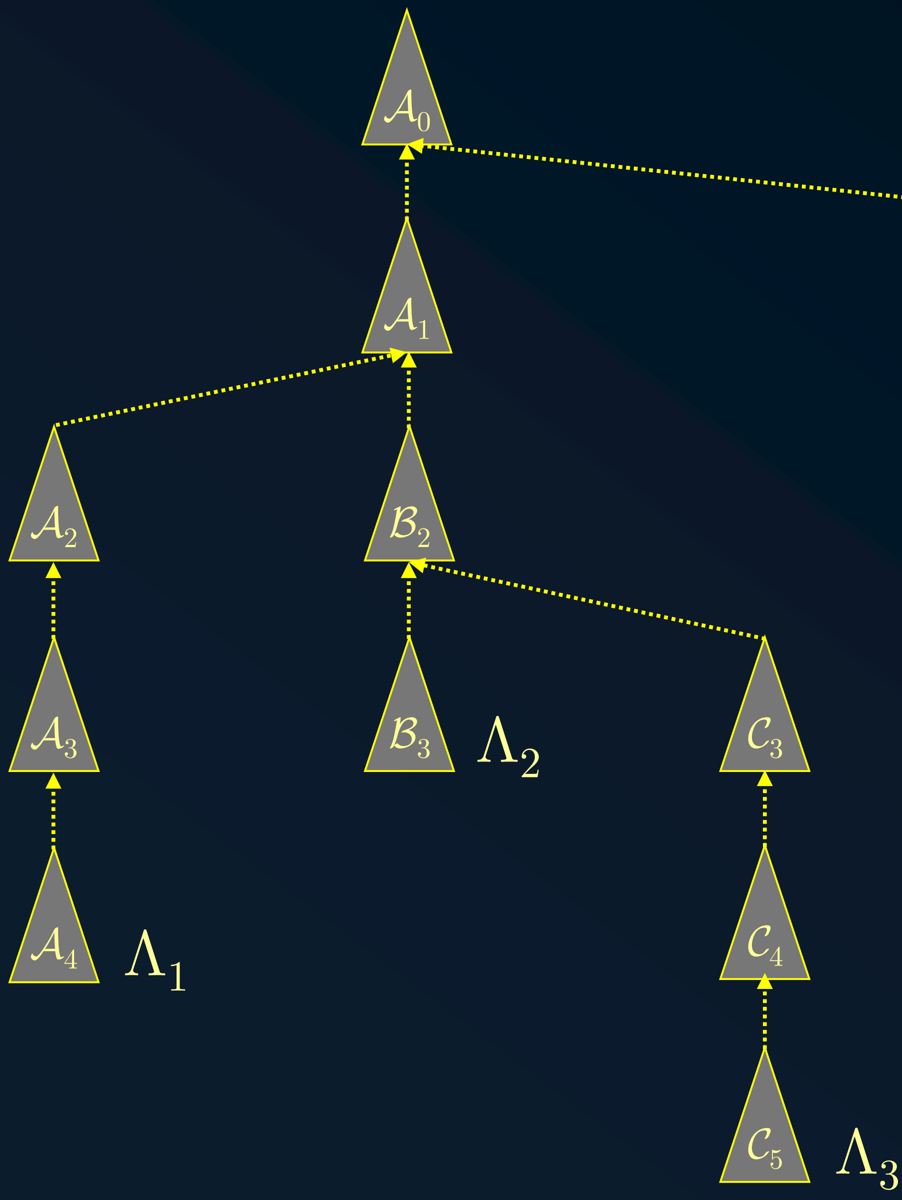
Argumentation Lines (Bundle)

Argumentation Lines (Bundle)

Argumentation Lines (Bundle)



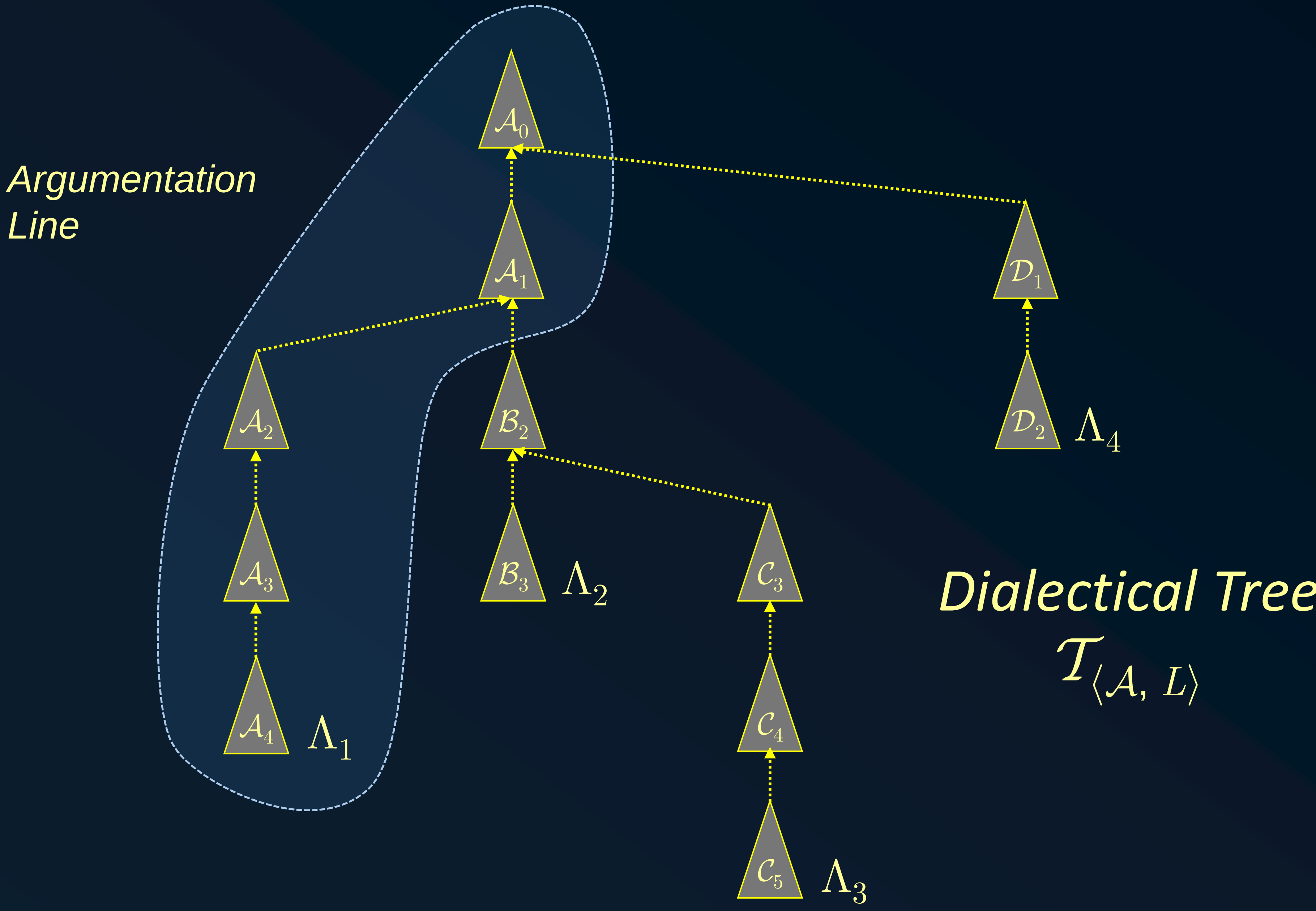
Argumentation Lines (Bundle)



Dialectical Tree

$\mathcal{T}_{\langle \mathcal{A},\, L \rangle}$

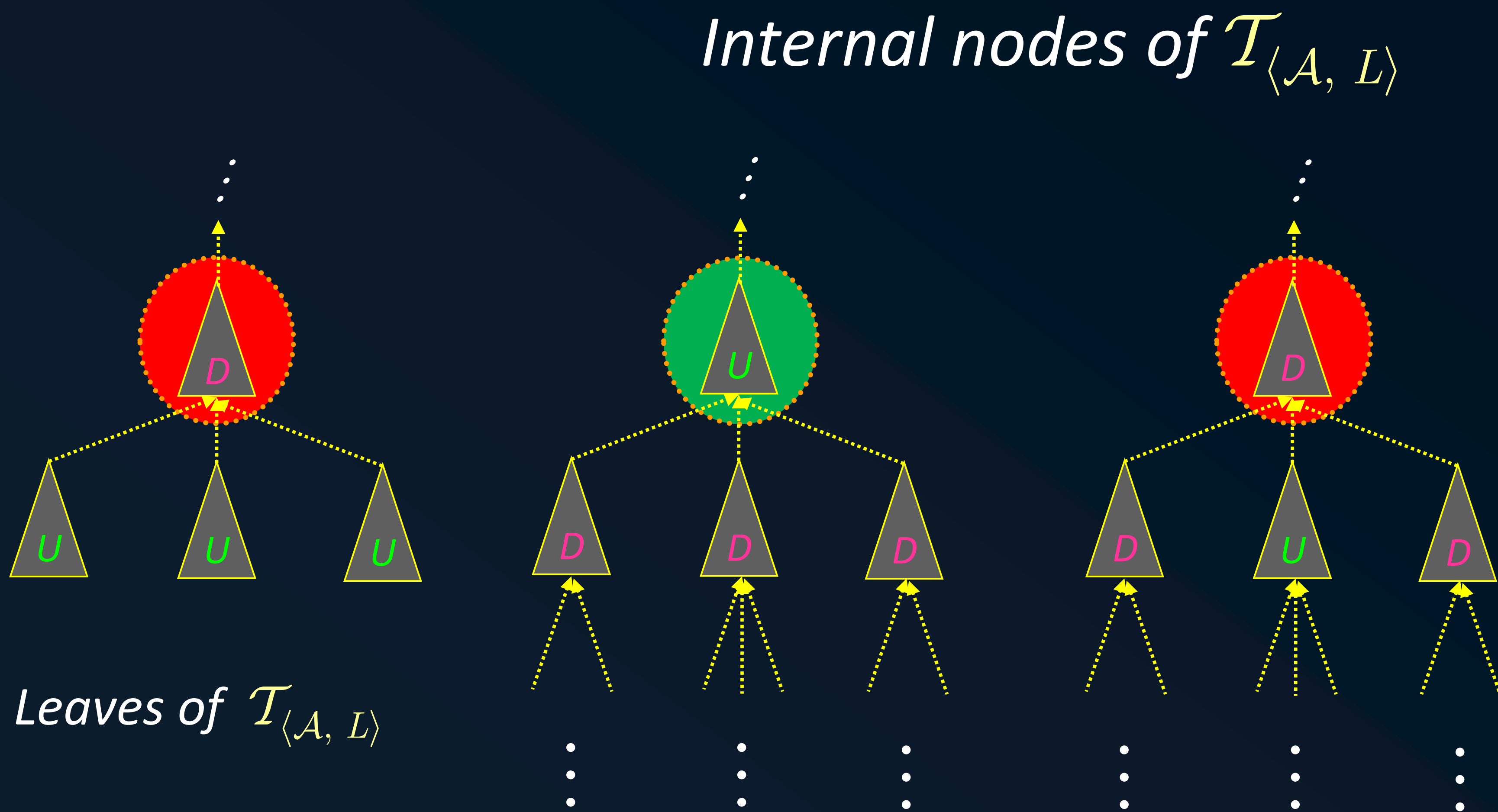
Argumentation Lines (Bundle)



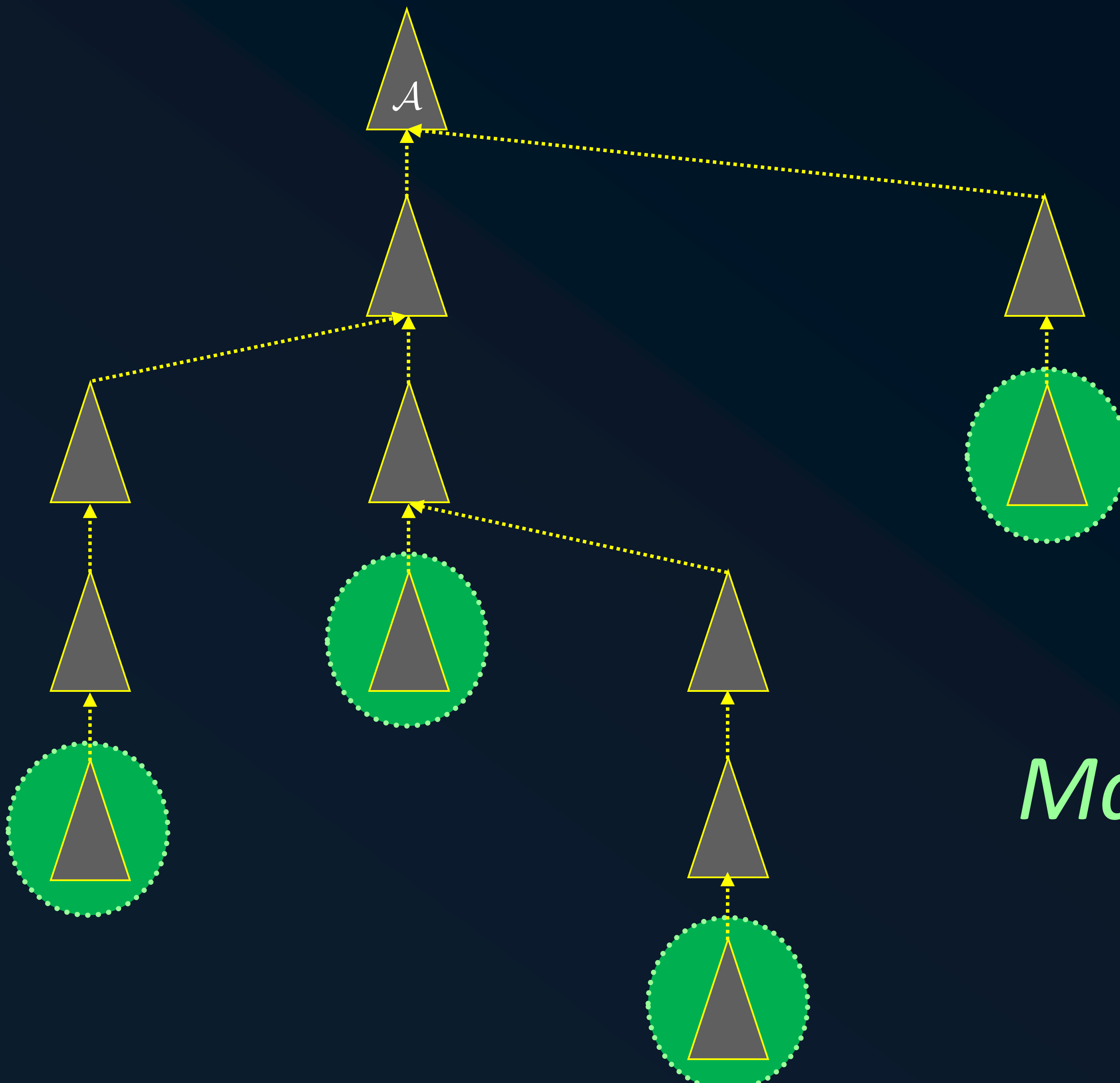
Dialectical Tree

- *A Dialectical Tree is the conjoint representation of all the acceptable argumentation lines.*
- *Given an argument $\mathcal{A}$ for a literal L, its dialectical tree contains all acceptable argumentation lines that start with that argument.*
- *Thus, analyzing the defeat status for a given argument could be done on the dialectical tree.*
- *As every argumentation line is admissible, and therefore finite, every dialectical tree is also finite.*

Marking of a Dialectical Tree

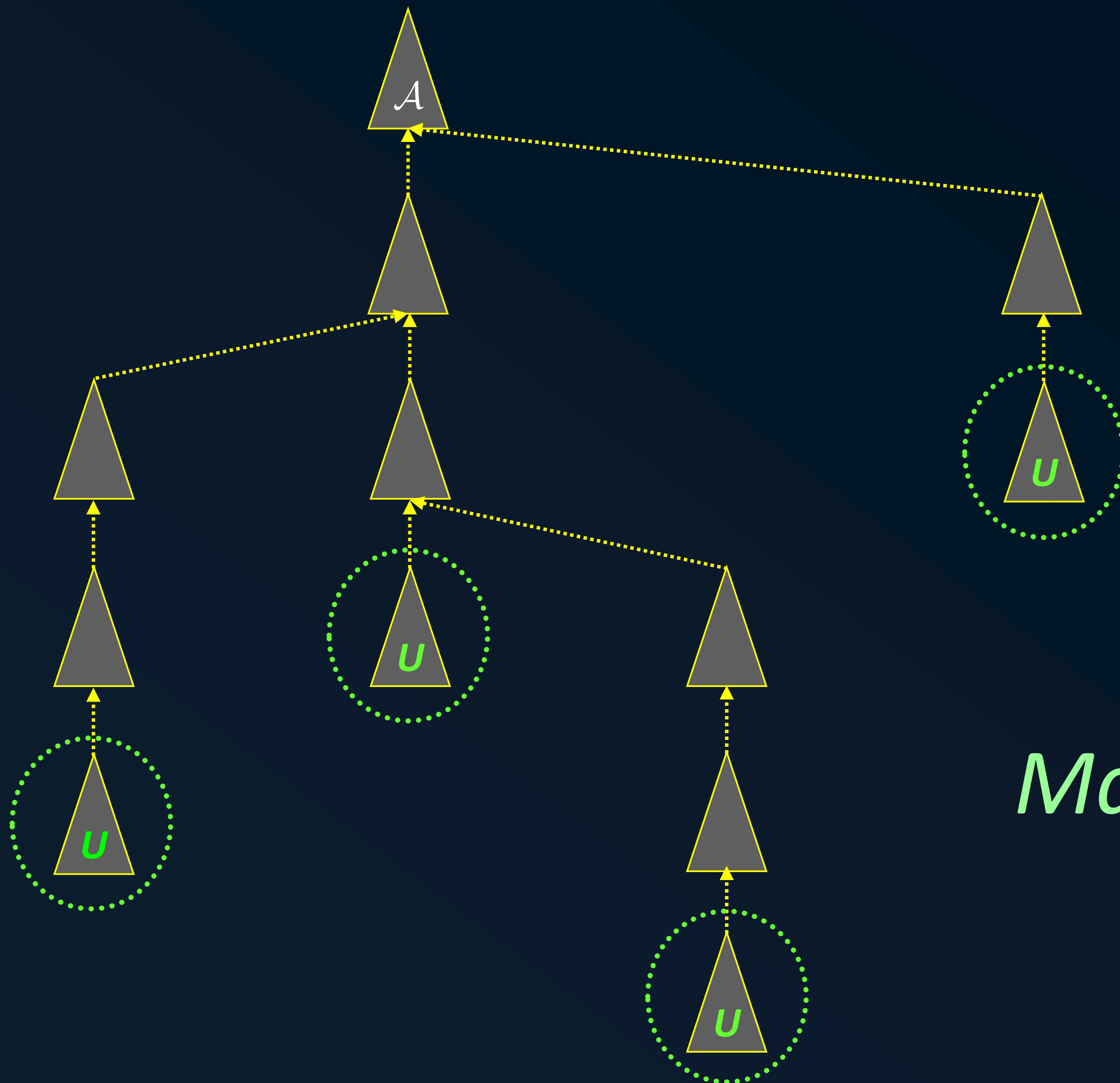


Dialectical Tree



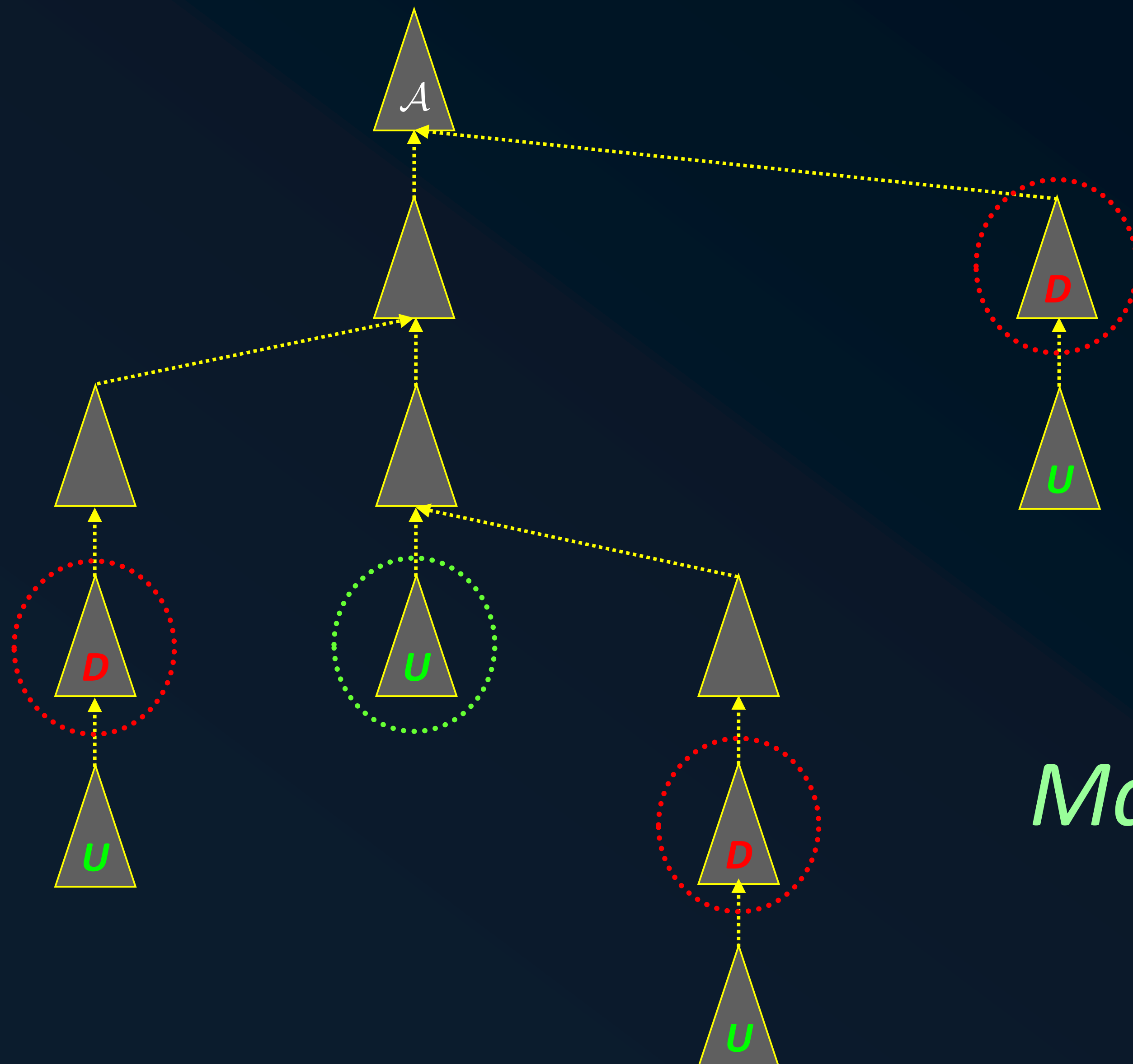
Marking

Dialectical Tree



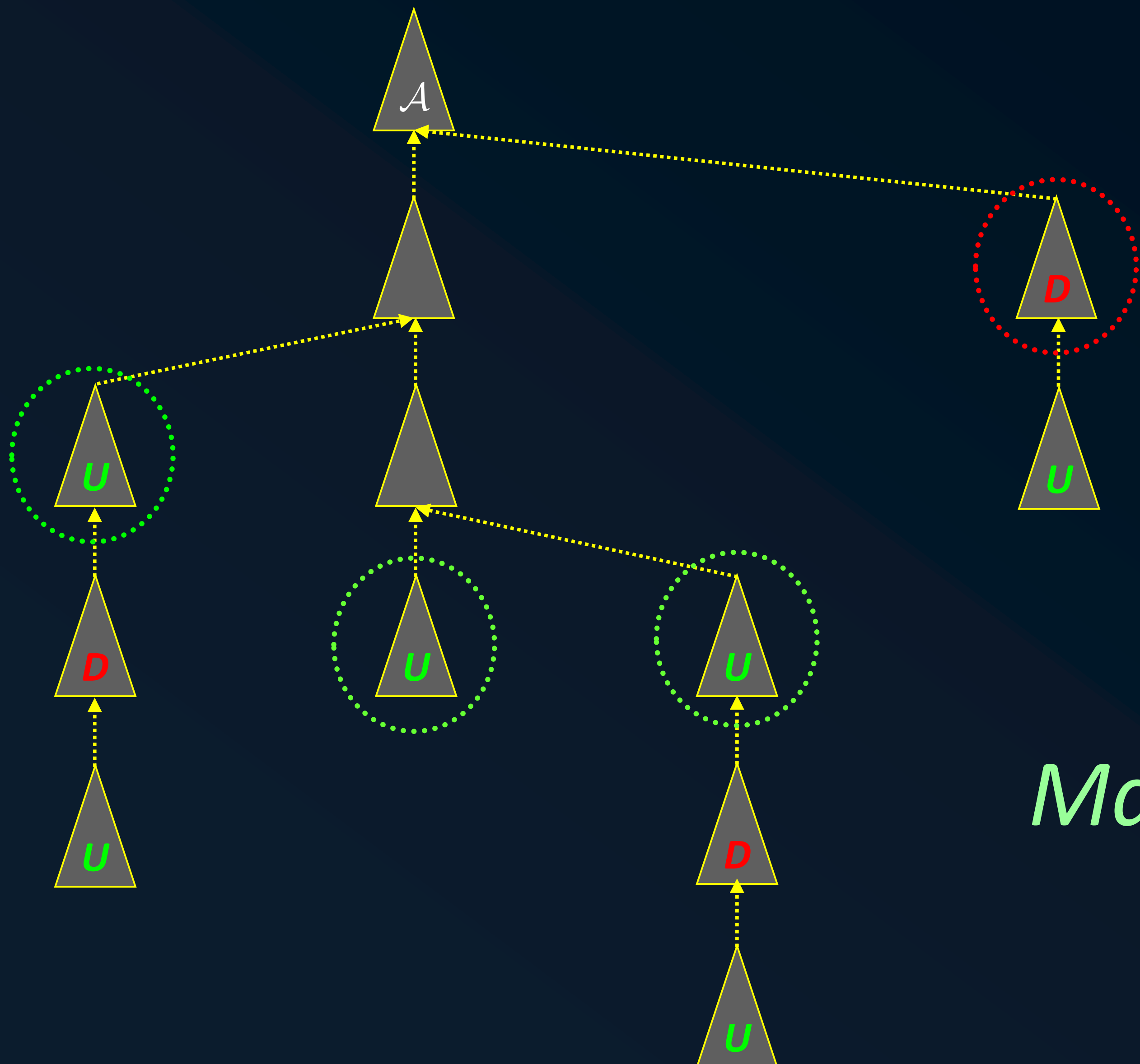
Marking

Dialectical Tree



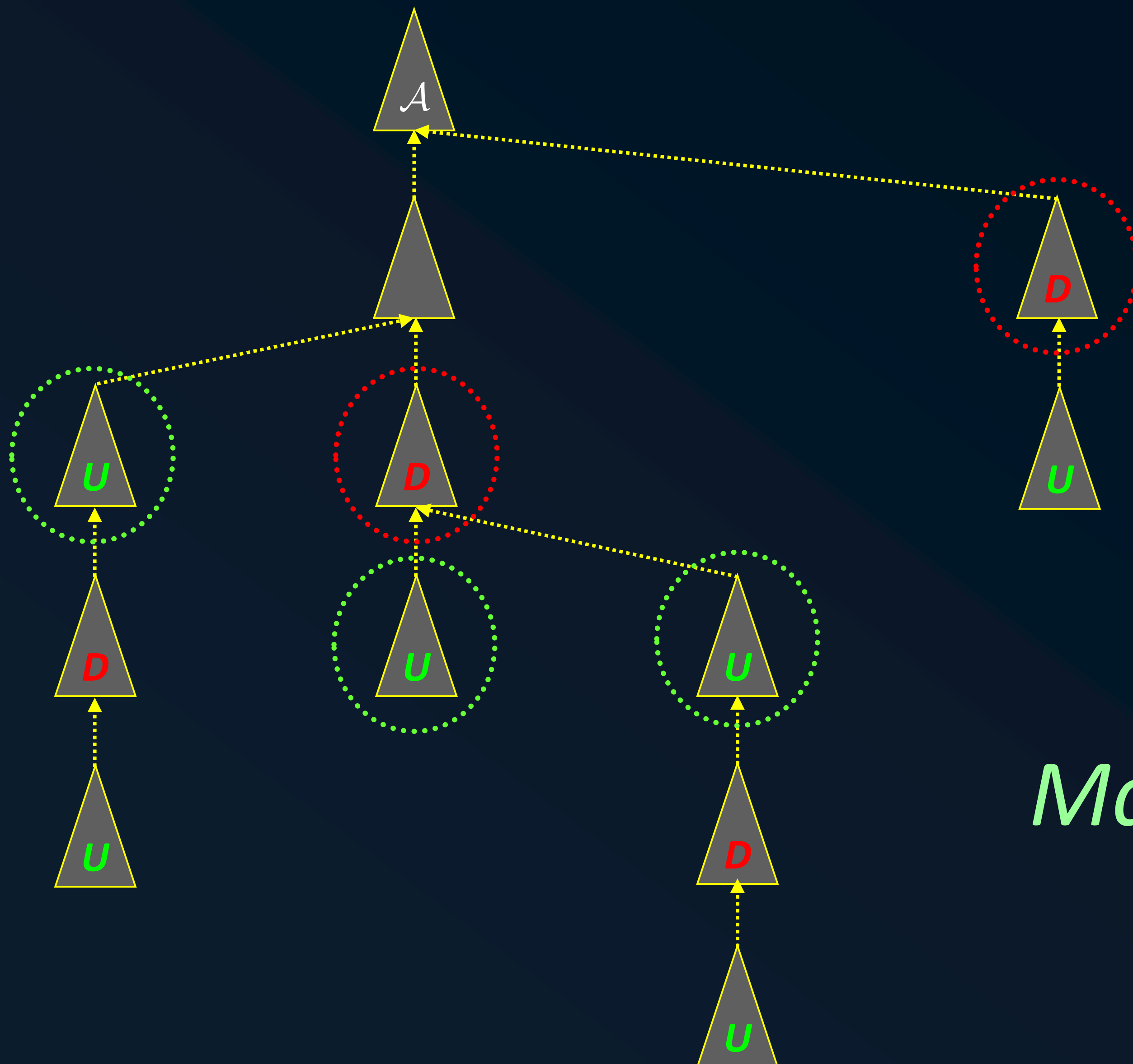
Marking

Dialectical Tree



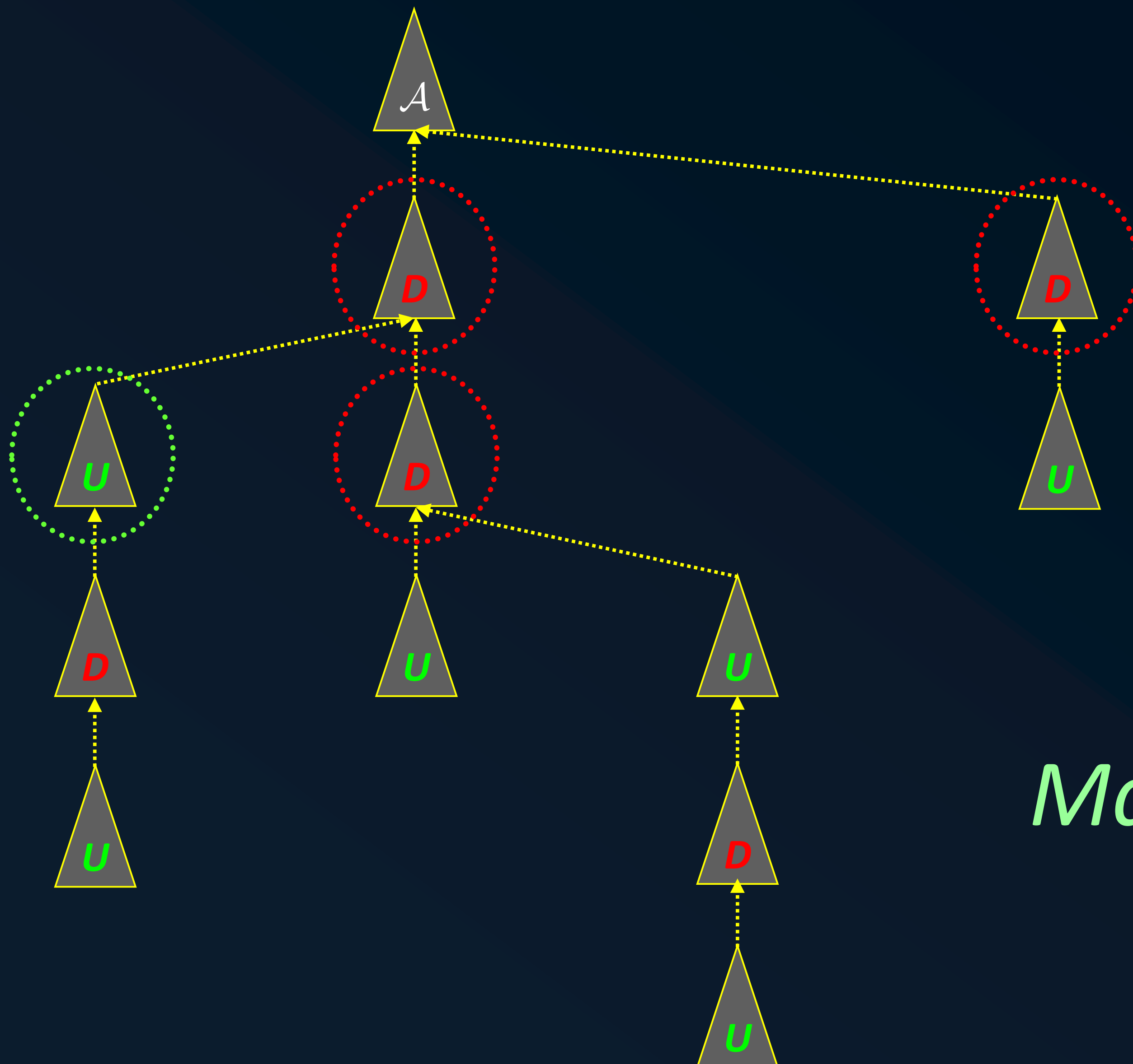
Marking

Dialectical Tree



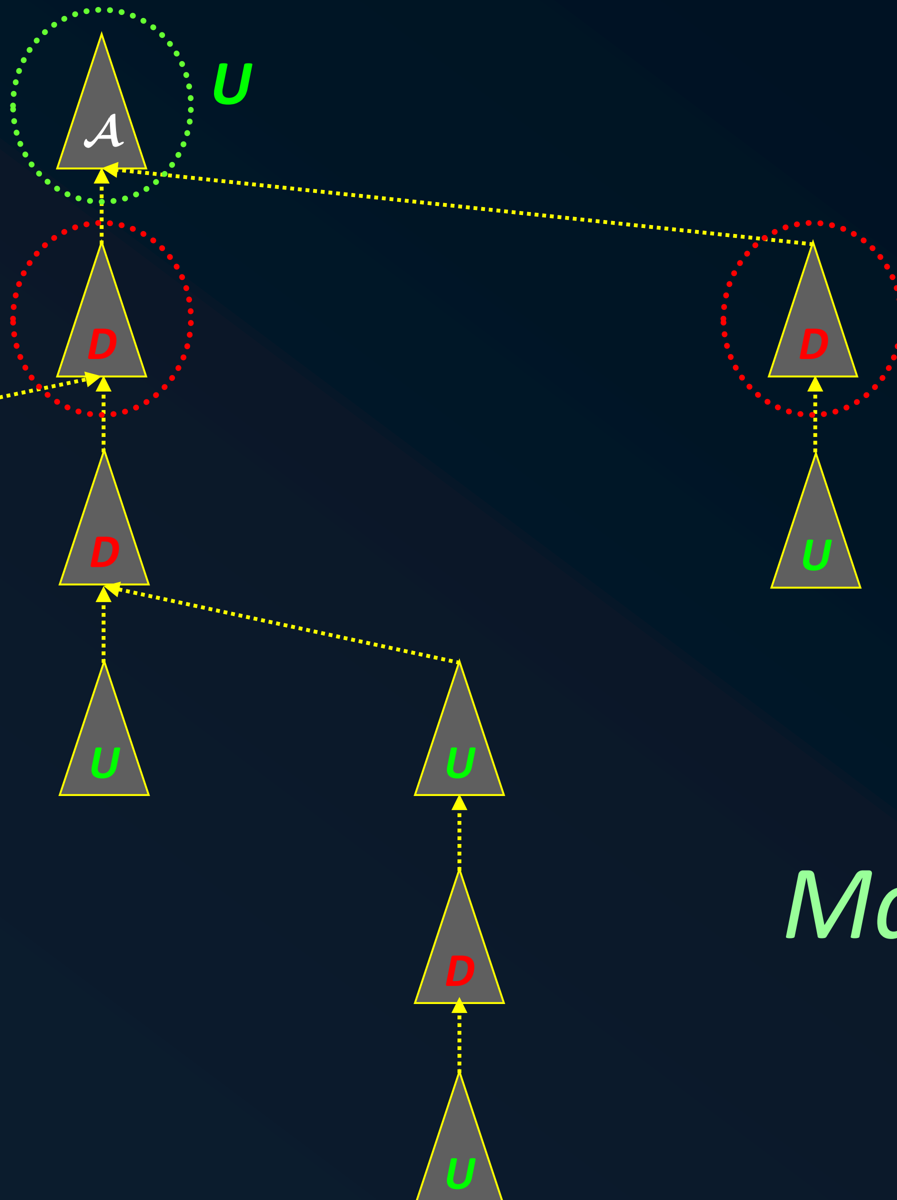
Marking

Dialectical Tree



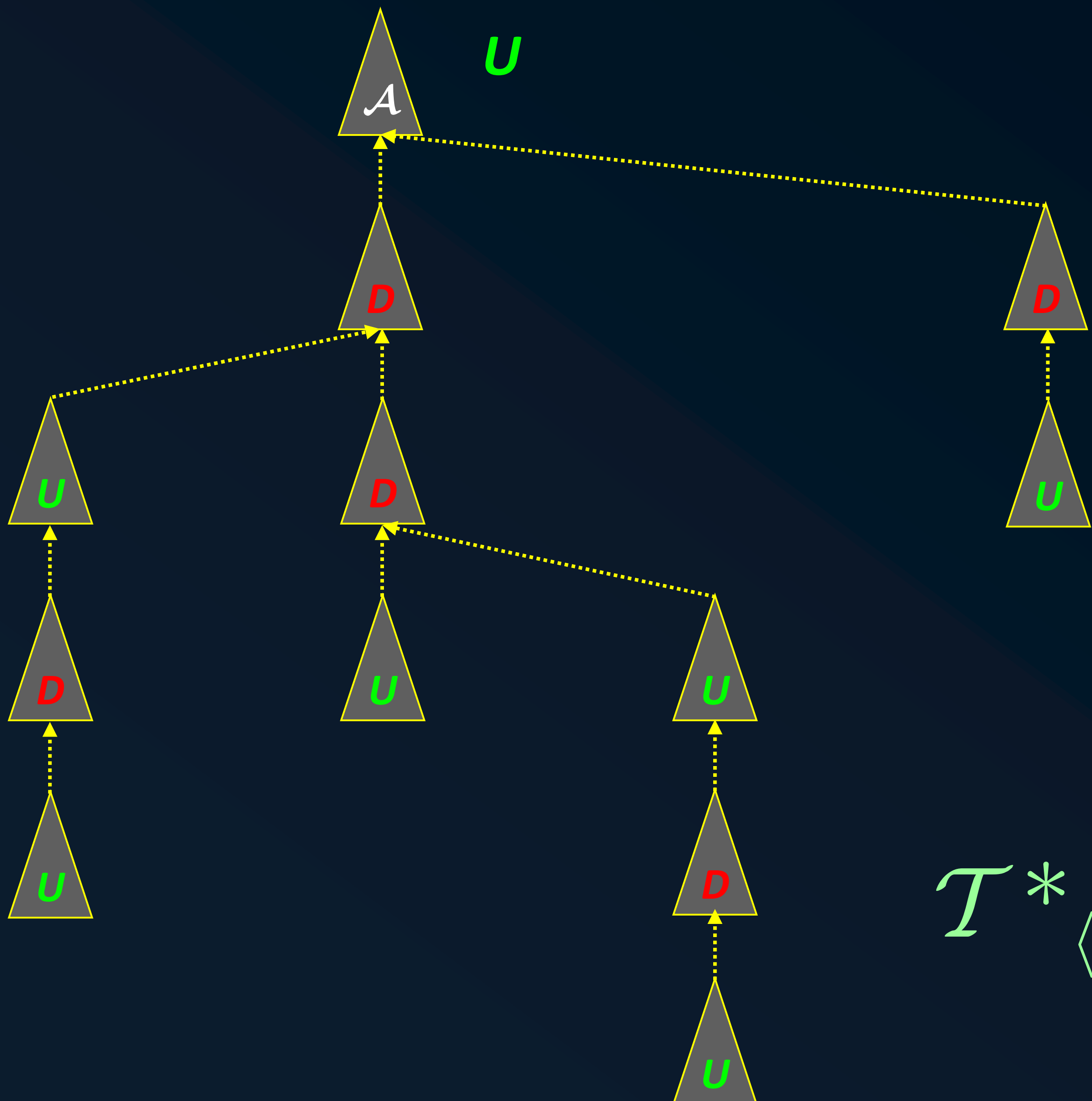
Marking

Dialectical Tree



Marking

Dialectical Tree (Marked)



$$\mathcal{T}^*_{\langle \mathcal{A},\, \mathrm{L} \rangle}$$

Warranted Literals

- *Let $\mathcal{P} = (\Pi, \Delta)$ be a defeasible program.*

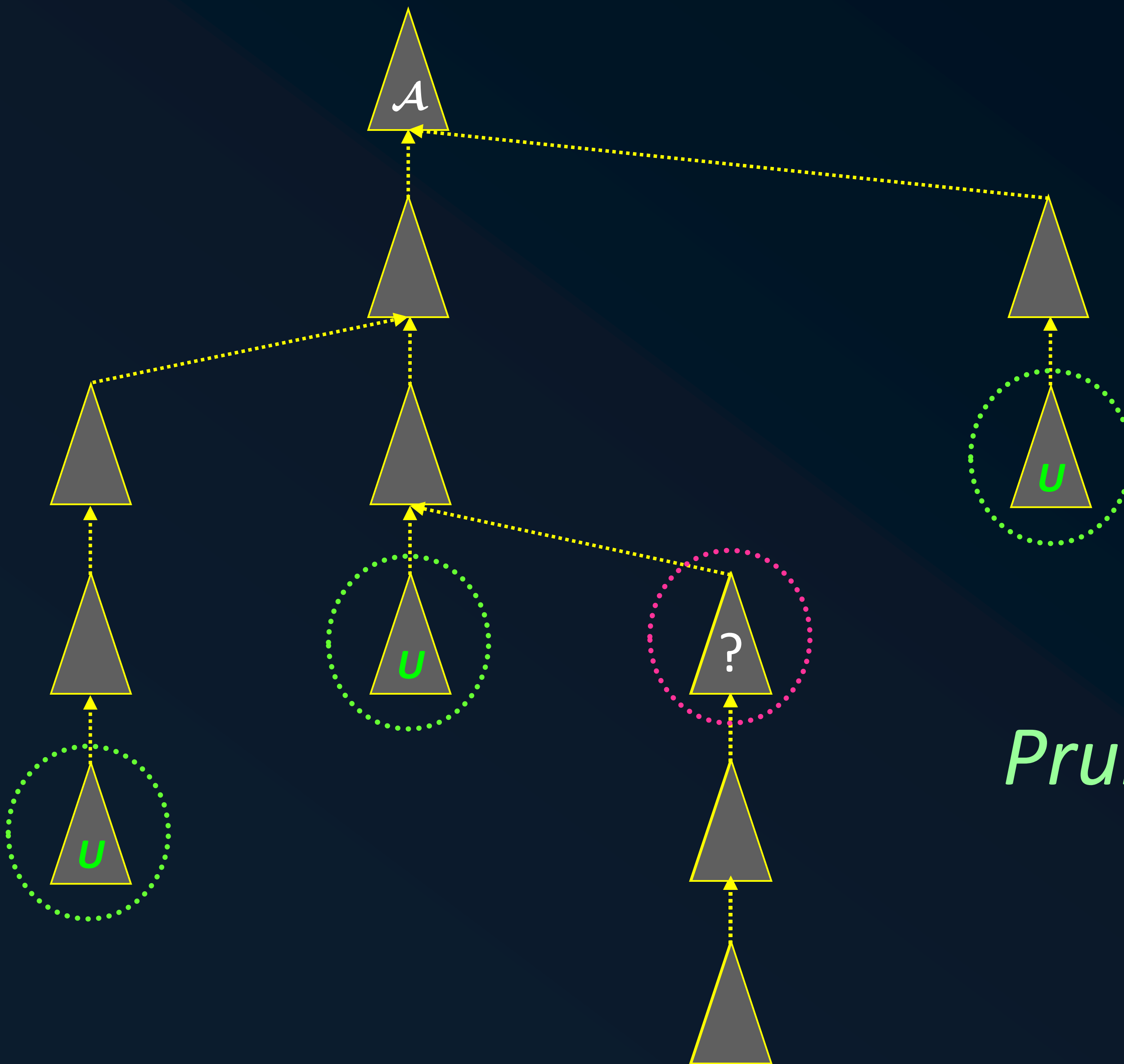
  Let $\langle \mathcal{A}, L \rangle$ be an argument and let $\mathcal{T}^*_{\langle \mathcal{A}, L \rangle}$ be its associated dialectical tree.

  A literal L is warranted if and only if the root of $\mathcal{T}^*_{\langle \mathcal{A}, L \rangle}$ is marked as "U".

- *That is, the argument $\langle \mathcal{A}, L \rangle$ is an argument such that each possible defeater for it has been defeated.*

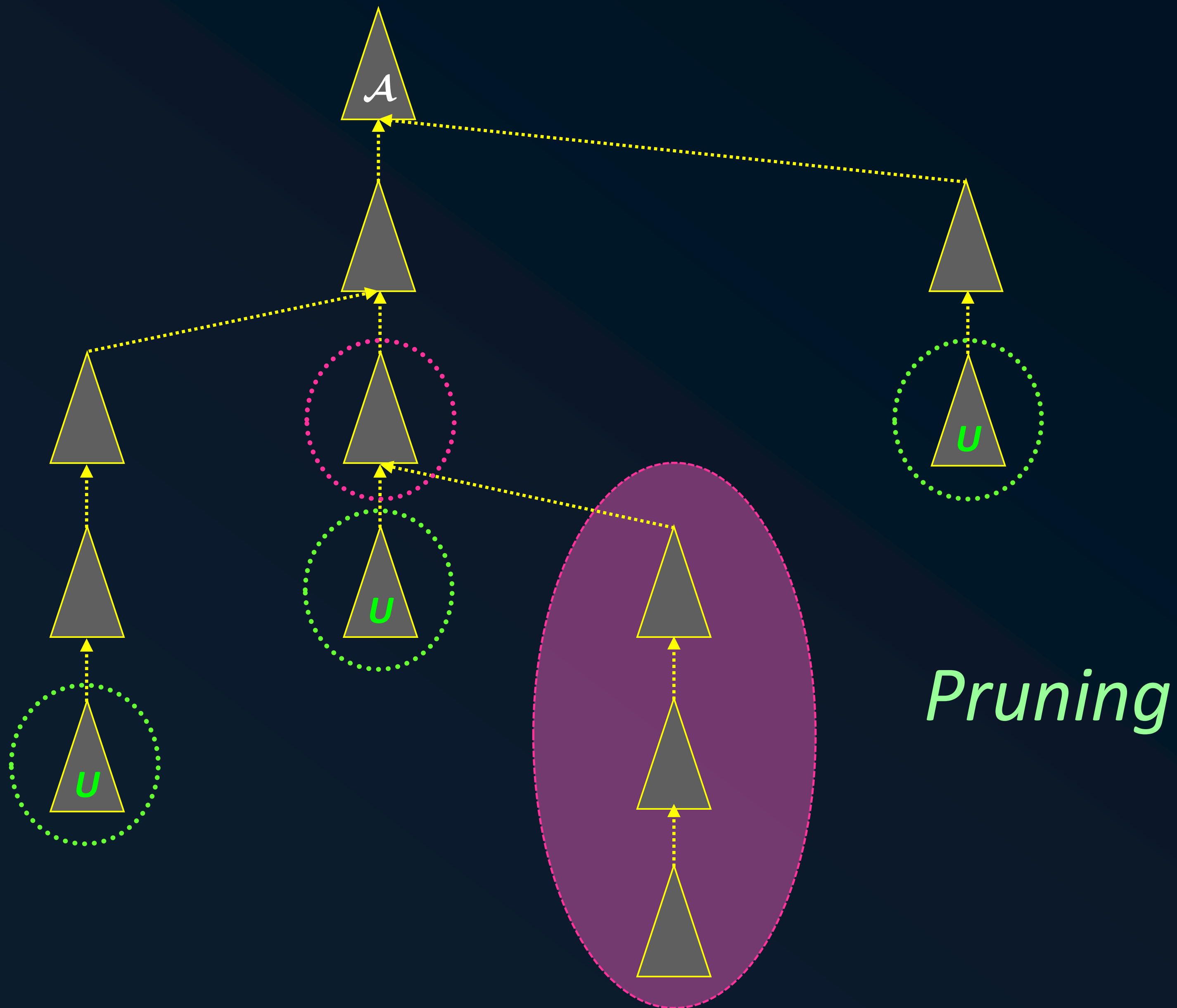
- *We will say that $\mathcal{A}$ is a warrant for L.*

Dialectical Tree: Pruning

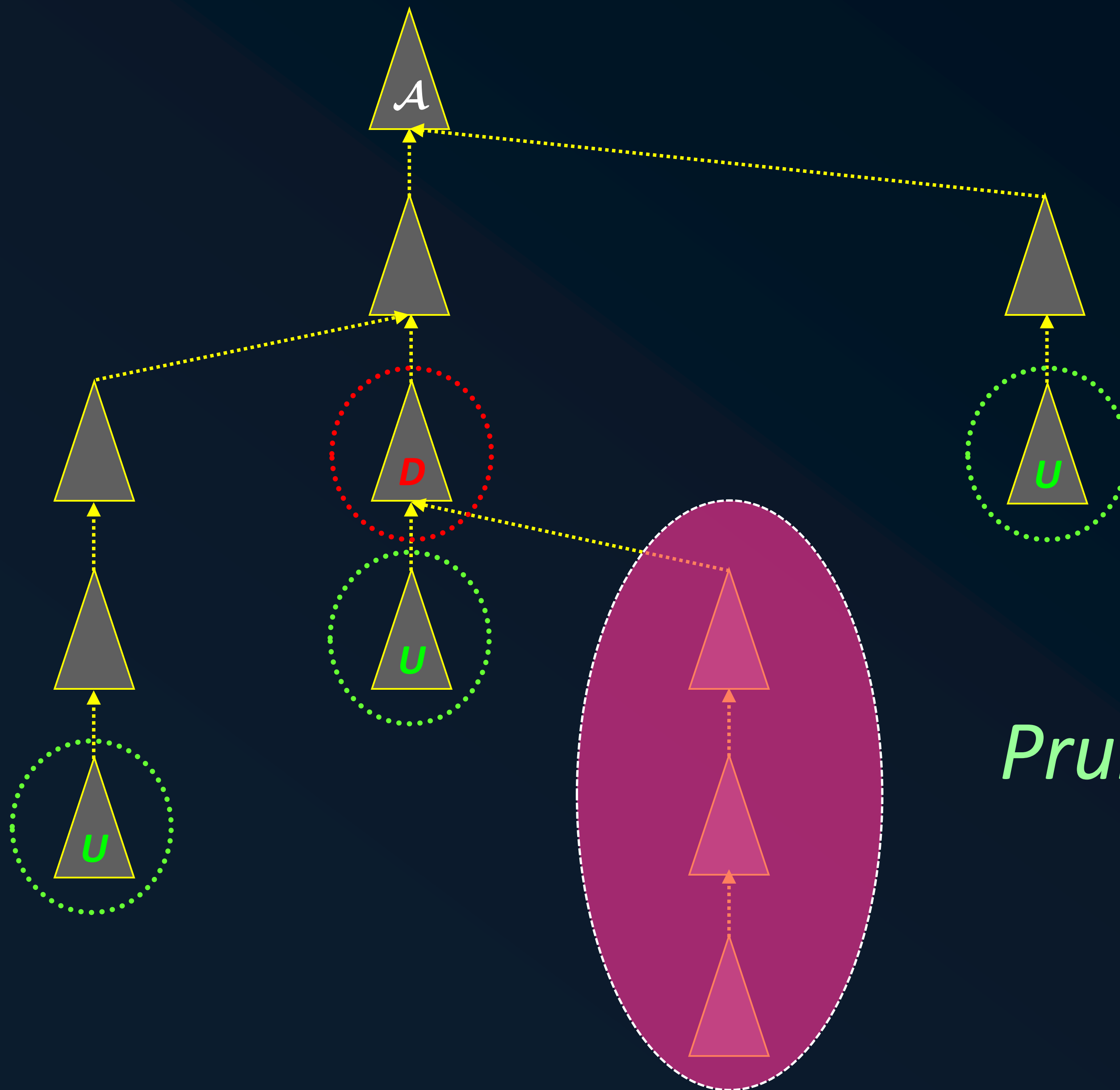


Pruning

Dialectical Tree: Pruning

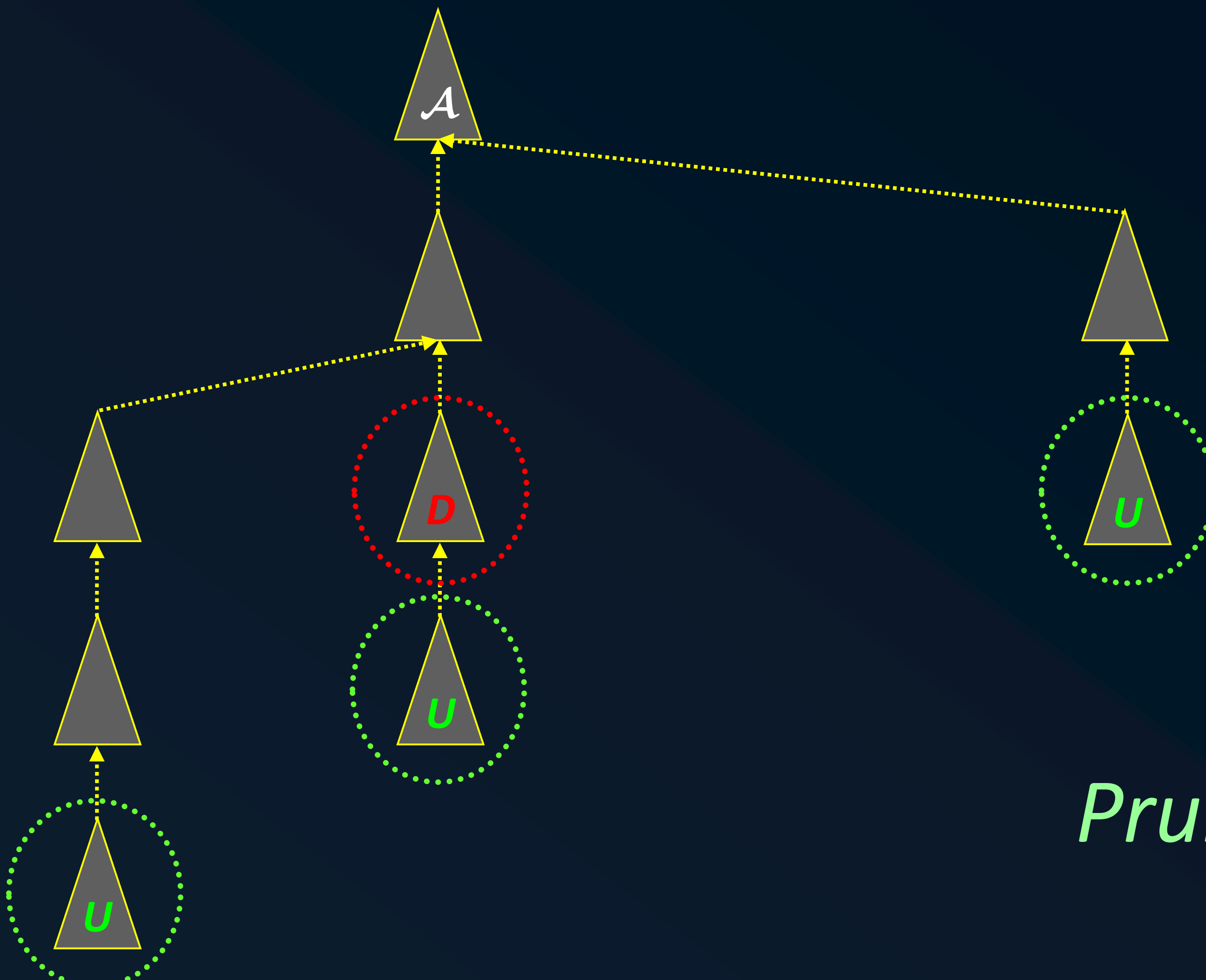


Dialectical Tree: Pruning



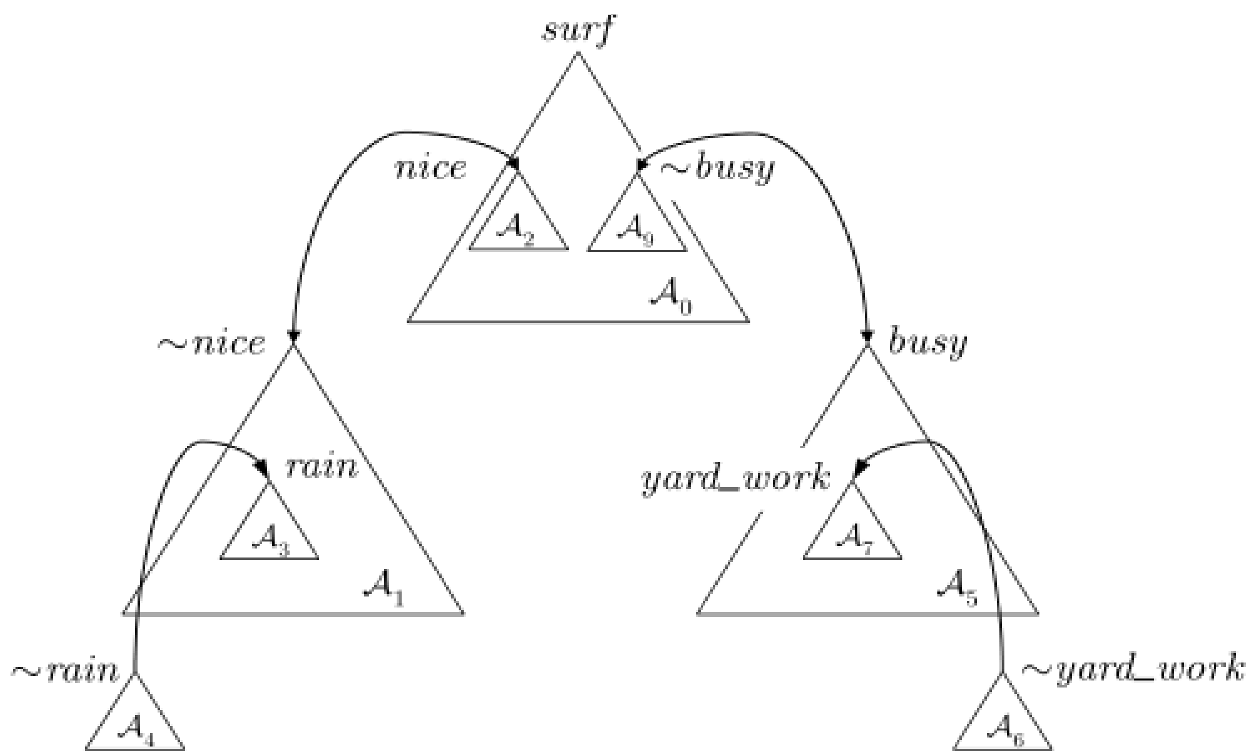
Pruning

Dialectical Tree: Pruning



Pruning

$$\Pi_{2.1} = \left\{ \begin{array}{l} monday \\ cloudy \\ dry\_season \\ waves \\ grass\_grown \\ hire\_gardener \\ vacation \\ \sim working \leftarrow vacation \\ few\_surfers \leftarrow \sim many\_surfers \\ \sim surf \leftarrow ill \end{array} \right\} \quad \Delta_{2.1} = \left\{ \begin{array}{l} surf \prec nice, spare\_time \\ nice \prec waves \\ \sim nice \prec rain \\ rain \prec cloudy \\ \sim rain \prec dry\_season \\ spare\_time \prec \sim busy \\ \sim busy \prec \sim working \\ cold \prec winter \\ working \prec monday \\ busy \prec yard\_work \\ yard\_work \prec grass\_grown \\ \sim yard\_work \prec hire\_gardener \\ many\_surfers \prec waves \\ \sim many\_surfers \prec monday \end{array} \right\}$$



Answers in DeLP

- *If the strict part* Π *of a program* $\mathcal{P} = (\Pi, \Delta)$ *is inconsistent, any literal can be derived.*
- *If a pair of complementary literals* $\{\ L, \sim\!L\ \}$ *can be derived, it is possible to introduce a way to try to decide whether to accept one of them.*
- *Therefore, there are three different possible answers: accept* L*, accept* $\sim\!L$*, or to reject both.*
- *Also, if the program is used to ànswer queries, there is a fourth possibility: the literal for which the query is made is unknown to the program.*

How it Works - DeLP

Query

$?\text{- } flies(opus)$

DeLP Interpreter (Abstract Machine)

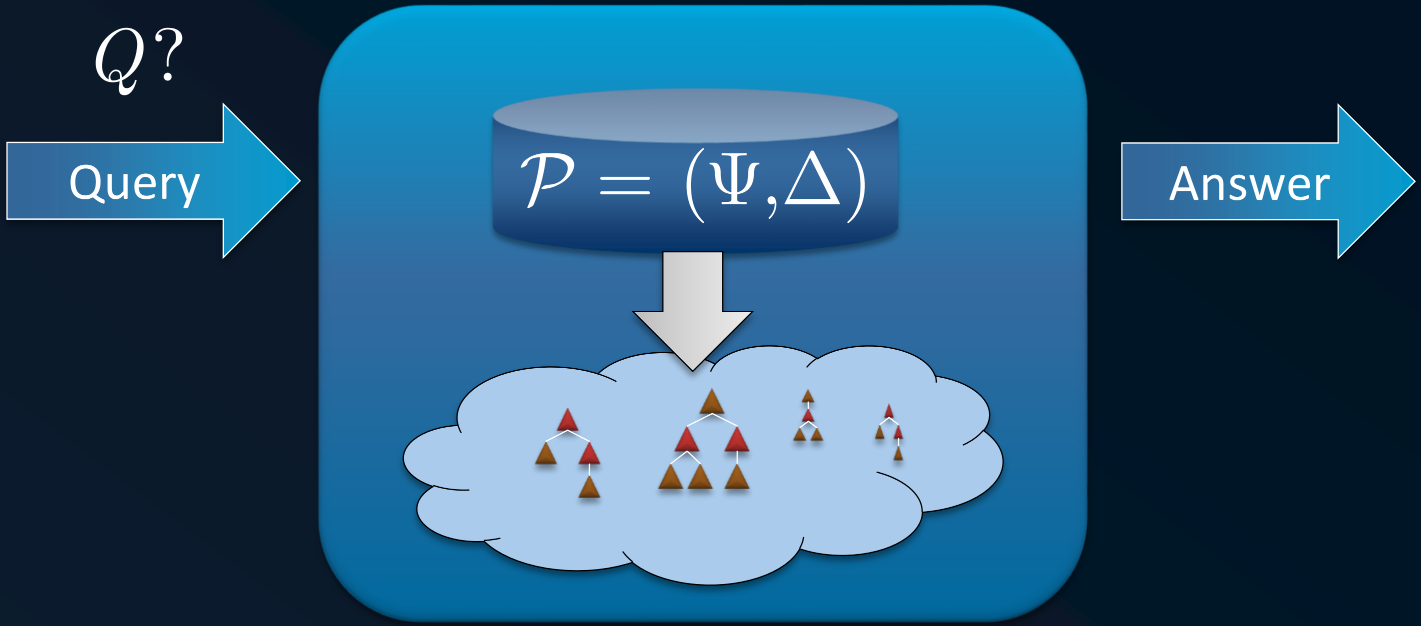
Knowledge Base

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

Possible Answers to Query Q

- YES if there exists a warranted argument $\langle \mathcal{A}, Q \rangle$
- NO if there exists a warranted argument for $\langle \mathcal{B}, \sim Q \rangle$
- UNDECIDED when none of the above cases hold

DeLP -Interpreter



- *YES if there exists a warranted argument* $\langle \mathcal{A}, Q \rangle$*, denoted* $\mathcal{P} \mathrel{|\!\!\sim}_{\mathrm{w}} Q$
- *NO if there exists a warranted argument for* $\langle \mathcal{B}, \sim Q \rangle$*,*
  denoted $\mathcal{P} \mathrel{|\!\!\sim}_{\mathrm{w}} \sim Q$
- *UNDECIDED when none of the above cases hold*
- *UNKNOWN if* Q *is not in the language of the program.*

Specification of the Warrant Procedure

```
warrant(Q, A) :-                                    % Q is a warranted literal
        find_argument(Q, A),                        % if A is an argument for Q
        \+ defeated(A, [support(A, Q)]).            % and A is not defeated

defeated(A, ArgLine) :-                             % A is defeated
        find_defeater(A, D, ArgLine),               % if there is a defeater D for A
        acceptable(D, ArgLine, NewLine),            % acceptable within the line
        \+ defeated(D, NewLine).                    % and D is not defeated

find_defeater(A, D) :-                              % C is a defeater for A
        find_counterarg(A, D, SubA),                % if C counterargues A in SubA
        \+ better(SubA, D).                         % and SubA is not better than C
```

References

- J. Pollock. Defeasible Reasoning, Cognitive Science, 11, 481-518, 1987.
- G. R. Simari, R. P. Loui. A Mathematical Treatment of Defeasible Reasoning and Its Implementation, Artificial Intelligence, 53, 125-157, 1992.
- P. Dung. *On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games.* Artificial Intelligence, 77(2):321-358, 1995.
- G. Vreeswijk, Abstract Argumentation Systems, Artificial Intelligence, 90, 225-279, 1997.
- C. Chesñevar, A. Maguitman, R. Loui. "*Logical Models of Argument*. ACM Computing Surveys, 32(4):337-383, 2000.
- H. Prakken, G. Vreeswijk. Logical Systems for Defeasible Argumentation, in D. Gabbay (Ed.), Handbook of Philosophical Logic, 2nd Edition, 2002.
- A. J. García, G.R. Simari. *Defeasible Logic Programming: An Argumentative Approach,* Theory and Practice of Logic Programming. Vol 4(1), 95-138, 2004.
- C. Chesñevar; G. Simari; T. Alsinet; L. Godo - *A Logic Programming Framework for Possibilistic Argumentation with Vague Knowledge* . Procs. of UAI-2004, Canada, 2004.
- G. R. Simari, A. García, M. Capobianco. Actions, Planning and Defeasible Reasoning. In Proc. 10th Intl. NMR 2004, Whistler BC, Canada. Pp. 377-384, 2004.
- M. Capobianco, C. Chesñevar, G. R. Simari. *Argumentation and the Dynamics of Warranted Beliefs in Changing Environments.* In Intl. Journal on Autonomous Agents and Multiagent Systems, 2005.
- I. Rahwan, G. R. Simari, *Argumentation in Artificial Intlligence*, 2009, Springer.
- P. Besnard, A. Hunter, *Elements of Argumentation*, 2008, MIT Press.

General References

ELSEVIER

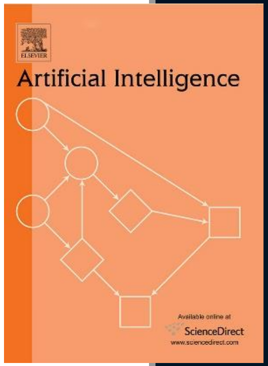
Available online at www.sciencedirect.com

ScienceDirect

Artificial Intelligence 171 (2007) 619–641

Artificial Intelligence

www.elsevier.com/locate/artint



Argumentation in artificial intelligence

T.J.M. Bench-Capon, Paul E. Dunne *

Department of Computer Science, University of Liverpool, Liverpool, United Kingdom

Received 27 April 2007; received in revised form 27 April 2007; accepted 1 May 2007

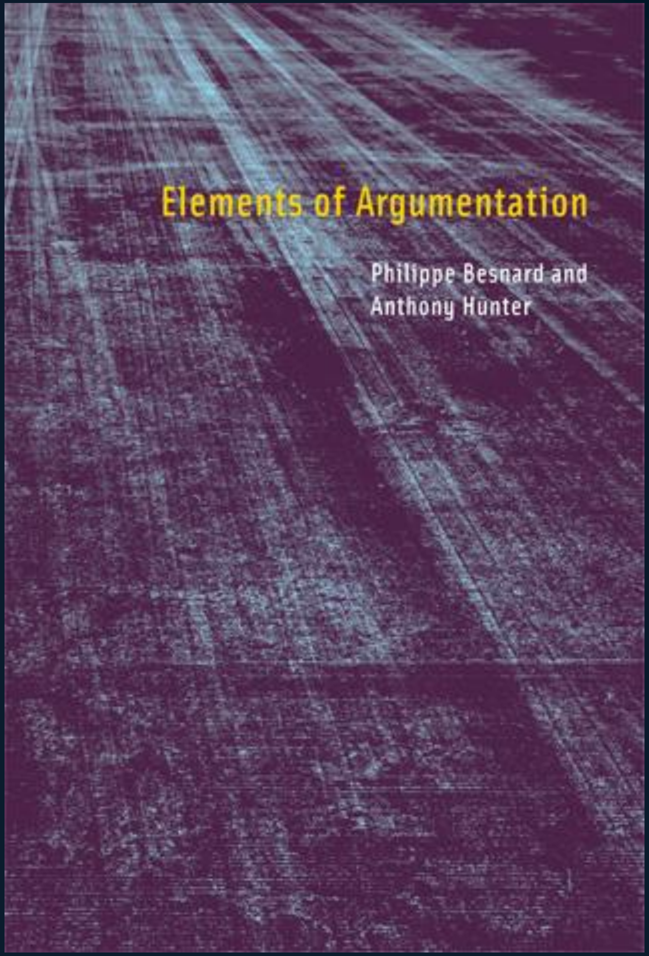
Available online 10 May 2007

Abstract

Over the last ten years, argumentation has come to be increasingly central as a core study within Artificial Intelligence (AI). The articles forming this volume reflect a variety of important trends, developments, and applications covering a range of current topics relating to the theory and applications of argumentation. Our aims in this introduction are, firstly, to place these contributions in the context of the historical foundations of argumentation in AI and, subsequently, to discuss a number of themes that have emerged in recent years resulting in a significant broadening of the areas in which argumentation based methods are used. We begin by presenting a brief overview of the issues of interest within the classical study of argumentation: in particular, its relationship—in terms of both similarities and important differences—to traditional concepts of logical reasoning and mathematical proof. We continue by outlining how a number of foundational contributions provided the basis for the formulation of argumentation models and their promotion in AI related settings and then consider a number of new themes that have emerged in recent years, many of which provide the principal topics of the research presented in this volume.

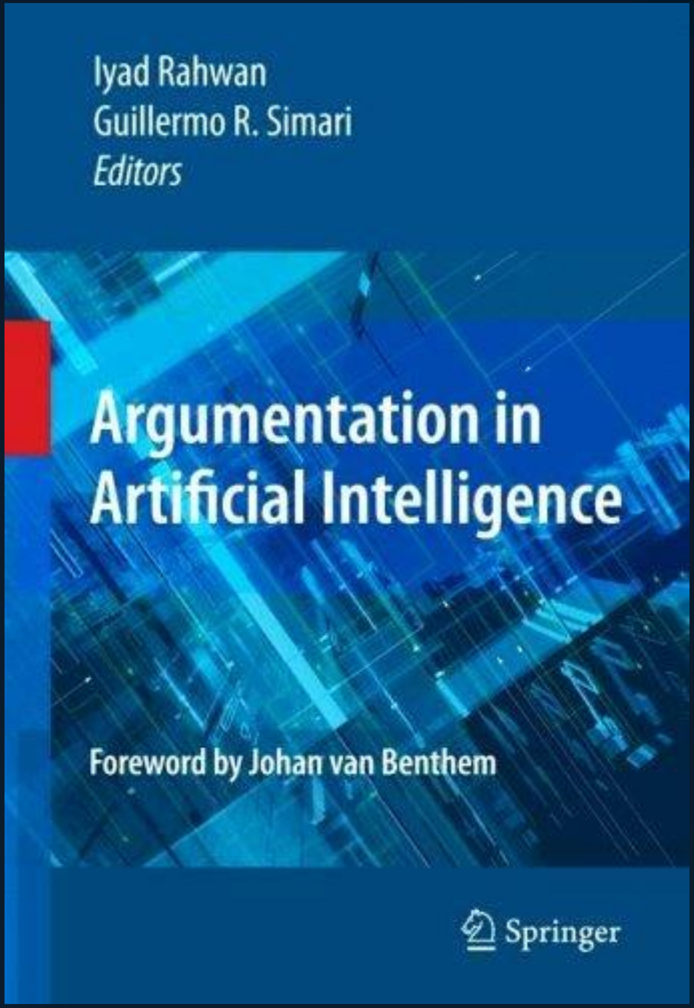
Keywords: Argumentation models; Dialogue processes; Argument diagrams and schemes; Agent-based negotiation; Practical reasoning

General References



Elements of Argumentation
Philippe Besnard and Anthony Hunter
MIT Press, 2008
ISBN: 978-0-262-02643-7

General References



Argumentation in Artificial Intelligence
Iyad Rahwan and Guillermo R. Simari
Springer, 2009
ISBN: 978-0-387-98196-3

General References

Towards Artificial Argumentation

Katie Atkinson[1], Pietro Baroni[2], Massimiliano Giacomin[2], Anthony Hunter[3], Henry Prakken[4,5], Chris Reed[6], Guillermo Simari[7], Matthias Thimm[8], and Serena Villata[9]

[1]Department of Computer Science, University of Liverpool, UK
[2]Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Brescia, Italy
[3]Department of Computer Science, University College London, UK
[4]Department of Information and Computing Sciences, Utrecht University, The Netherlands
[5]Faculty of Law, University of Groningen, The Netherlands
[6]Department of Computer Science, University of Dundee, UK
[7]Department of Computer Science & Engineeering, Universidad Nacional del Sur, Argentina
[8]Institute for Web Science and Technologies, Universität Koblenz-Landau, Germany
[9]Université Côte dAzur, CNRS, Inria, I3S, France

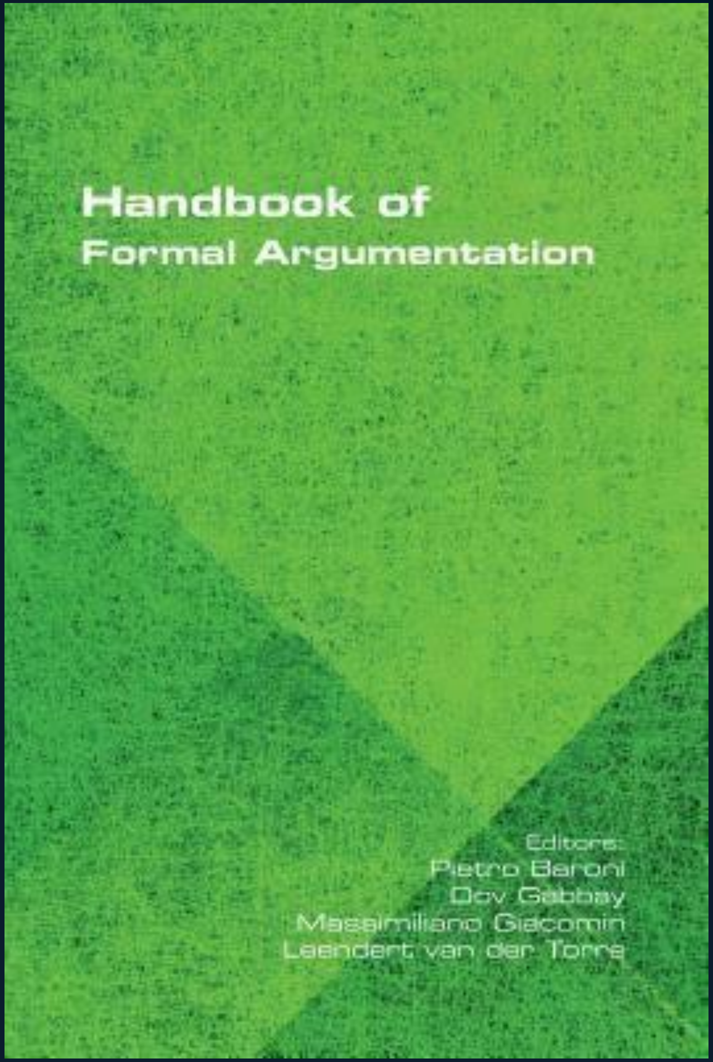
March 15, 2017

AI Magazine - Vol 38 No 3: Fall 2017, pp. 25-36.

Abstract

The field of computational models of argument is emerging as an important aspect of artificial intelligence research. The reason for this is based on the recognition that if we are to develop robust intelligent systems, then it is imperative that they can handle incomplete and inconsistent information in a way that somehow emulates the way humans tackle such a complex task. And one of the key ways that humans do this is to use argumentation — either internally, by evaluating arguments and counterarguments — or externally, by for instance entering into a discussion or debate where arguments are exchanged. As we report in this review, recent developments in the field are leading to technology for artificial argumentation, in the legal, medical, and e-government domains, and interesting tools for argument mining, for debating technologies, and for argumentation solvers are emerging.

General References



Handbook of Formal Argumentation

Pietro Baroni, Dov Gabbay, Massimilino Giacomin

Volume 1, College Publications, Feb 28, 2018.

Also Volume 2 (forthcoming August 2021).

Several more volumes planned.

Thank you!

Questions?