Explainable AI via Argumentation: Theory & Practice

Antonis Kakas
University of Cyprus, Cyprus
Nikos Spanoudakis
Technical University of Crete, Greece

Co-founders (with Pavlos Moraitis) of Argument Theory
https://www.argument-theory.com/

ESSAI 2024 School: 22-26 July, Athens

Course Contents & Structure

- Theory of Argumentation: Concepts & Methodologies
  - Lecture 1 – Overall Exposition.

- Hands on Development of XAI Arg-based system(s)
  - Lecture 1 – Student start their Choice of Problem
  - Lecture 2 – Argumentation in Practice & Technology
    - GORGIAS and RAISON
  - Lecture 3 & 4 – Further Study of Practice of Argumentation
    - Student Systems development
  - Lecture 4 & 5 – Student Presentations

- Brief Exposition of Advanced Topics - Lecture 5
  - Explainable Machine Learning via Argumentation: ArgEML
  - Argumentation in Natural Language: COGNICA with LLM

Your Background

- Course on Argumentation?
  - Read on Argumentation?
- Research on Argumentation?
- Practice of Argumentation?
  - Used a System of Argumentation?
  - Studied Application Problem via Argumentation?

Lecture 1

- **Motivation**
  - **Explainable AI (XAI) & Why Argumentation for XAI?**
- **Theory of Argumentation**
  - **Validity of Argument**
  - **Argumentative Reasoning**
- **Argumentation in Practice**
  - **Structured Argumentation for Knowledge Representation.**
  - **Gorgias Argumentation Framework.**
  - **Preview: Basics of a Methodology for Contextual Knowledge Acquisition**
- **Preview: Building Arg-based Systems**
  - **High-level Systems Architecture**
  - **Arg-based Technology – Systems and Authoring Tools**
- **Start of hands-on Development**
  - **Students choose their own application problem.**
  - **Open accounts in Gorgias Cloud**

MOTIVATION

Why Explainable AI (XAI)?

The Era of AI: Industry 4.0 – Automated Decision

Explainable Decision Making

What is Explainable AI?

What is Explainable AI?

"AI concerned not (only) with what is the solution but with how it comes about."

With HOW that is:
Informative, Debatable, Contestable

FOR

Accountability, Trust, Ethicacy
But also, Usefulness with "Human in Loop"

Explainable AI

Explanations for Informative, Debatable/Auditable, Contestable AI Systems

What is an Explanation?

Tim Miller: ***Explanation in artificial intelligence: Insights from the social sciences, 2019.***
Richard Feynman Why?: ***Video***

The *Science* of *Explanations*

Factor	Description
Explanations are contrastive	Explanations are contrastive: people usually don't only ask why a certain prediction was made but rather why this prediction was made instead of another prediction.
Explanations are selective	Explanations are selective and focus on one or two possible causes and not all causes for the recommendation.
Explanations are social	Explanations are part of social interaction between the explainer and the explainee. This means that the social context determines the content, the communication, and the nature of the explanations.
Explanations are contextual	Explainable AI systems should be able to explain their capabilities and understandings, however every explanation is set within a context that depends on the task, abilities, and expectations of the user of the AI systems.
Explanations need to be trustworthy	Trust must be considered in terms of the accuracy and reliability of the system, but also in terms of how much individuals trust the explanations give. Mistrust of the whole system can result from explanations that are too complicated, incomplete or inaccurate.
Explanation recipient	The "intended audience" is another factor that needs to be considered when generating explanations as different user types have different needs. For example, a computer engineer may need more detailed explanations when auditing the system from a patient or a physician.

Evaluation *Metrics* for *Explanations*

	Co-12 Property (*)	Description
Content	Correctness	Describes how faithful the explanation is w.r.t the black box.
	Completeness	Describes how much the black box behavior is described in the explanation.
	Consistency	Describes how deterministic and implementation-invariant the explanation method is.
	Continuity	Describes how continuous and generalizable the explanation function is.
	Contrastivity	Describes how discriminative the explanation is w.r.t. other events or targets.
	Covariate complexity	Describe how complex the (interaction of) features in the explanation are.
presentation	Compactness	Describes the size of the explanation.
	composition	Describes the presentation format and organization of the explanation.
	confidence	Describes the presence and accuracy of probability information in the explanation.
user	Context	Describes how relevant the explanation is to the user and their needs.
	Coherence	Describes how accordant the explanation is with prior knowledge and beliefs.
	Controllability	Describes how interactive and controllable an explanation is for a user.

(*) Nauta, M. *et al.* From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI. (2022).

Properties of Explanation

Explanations need to be:

- **Attributive – Why this solution?**
- **Contrastive – Why not some other solution?**
- **Actionable – Where does this solution lead?**

Why Argumentation for AI & XAI

- **Argumentation: Reasoning Universalis**
  - **Formal Logical & Informal Reasoning**

- **Argumentation: Naturally Explanatory**
  - **Debate for and against a claim/position**

Dietz et al, Argumentation: A calculus for Human-Centric AI, 2022.

The Big Picture

Argumentation is a Natural Calculus for Explainable AI

↓

How do we Reason in Argumentation?

How do we Model and Acquire Knowledge for Argumentation (in a practical way)?

How do we Build Arg-based systems?

Building XAI Systems from Natural Specifications

Example of Contextual Decision Problem

Call Assistant (Personal Policy)

"Normally, allow a call. When at work deny a call from an unknown number. When busy at work also deny a call from a known number unless it is an emergency family call. Always allow a call from my manager. "

Options: allow a call, deny a call.

ARGUMENTATION THEORY

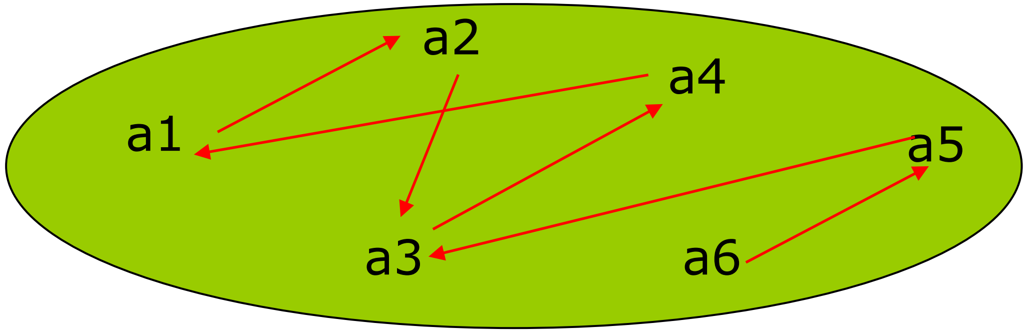
Theory of Argumentation

- Abstract Argumentation
  - Validity of Arguments
  - Admissibility Semantics
- Structured Argumentation
  - Realization of Abstract Argumentation
  - Dynamic - Contextual Argumentation

Abstract Argumentation Frameworks

<Args, ATT>

- **Args : Set of Arguments**
  - **Ex: {a1,a2,a3,a4,a5,a6}**
- **ATT : Relation on Args: Conflict & Strength**



- **Forms a STATIC ARENA for Argumentation**
  - **A Snap-short of a (the current) context of debate**
  - **A current argumentative debate takes place.**

Abstract Argumentation Frameworks
<Arg, Att, Def>

- **Arg is a set of Arguments**
- **Att is weak attack or conflict relation between arguments**
- **Def is a strong defense or defeat relation between arguments.**

In <Args, ATT> ATT combines Att and Def.

(See Extra Slides: Connection of AAFs)

Validity of Arguments <Args, Att, Def>

Acceptable set of Arguments:

Arguments that can defend against (all) their counter-arguments

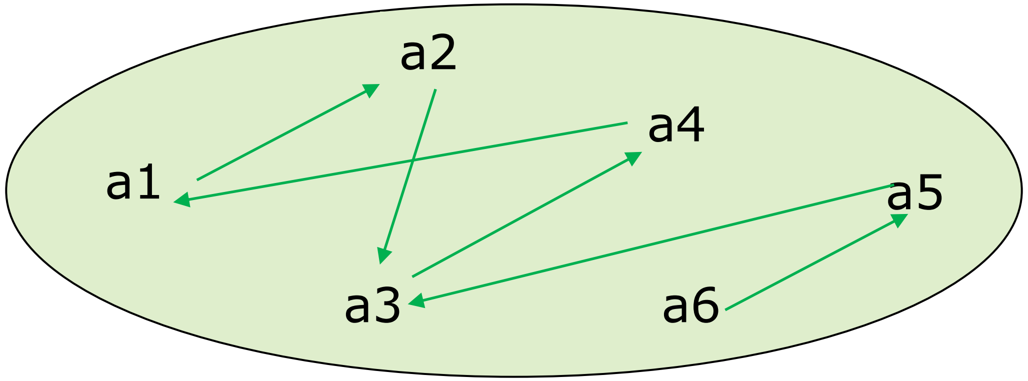
Example: Admissible set of Arguments:

1. Set is not self-attacking
2. Set defends against any attacking set

(In <Arg,ATT>: Defends = ATTACKS back)

Valid Coalitions/Cases of Arguments

Q: Is argument a1 valid/admissible?

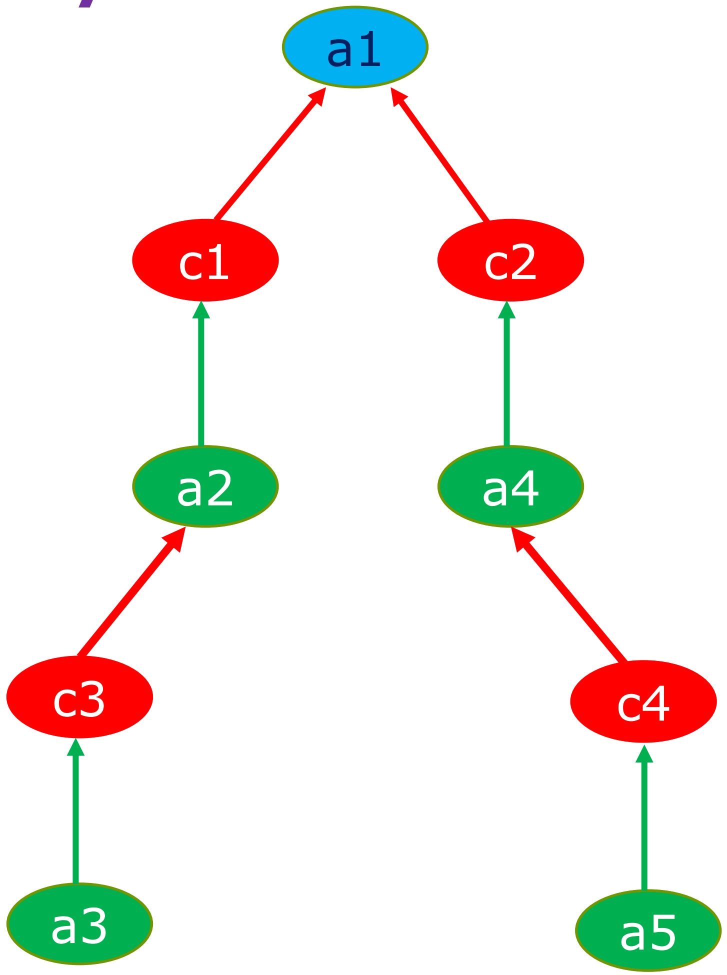


- **Only in coalition {a1,a3,a6}: a CASE for a1!**

- **Is {a1, a2} a valid coalition/case?**
  - **No – it is self-attacking!**

Reasoning/Computation in <Args, Att, Def>

FINDING/FORMING A VALID CASE



{a1,a2,a4,a3,a5}

is **acceptable:**

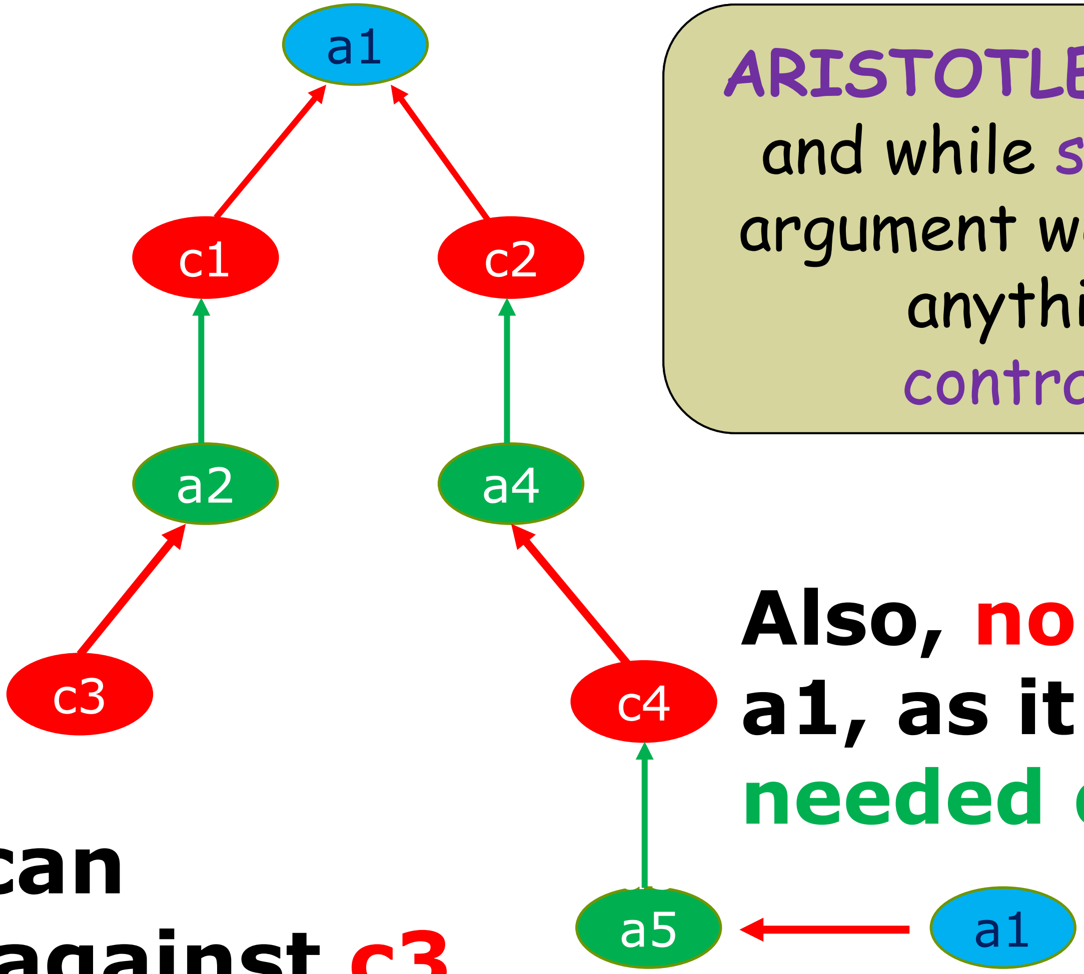
A valid case

for a1.

Reasoning/Computation in <Args, Att, Def>

FIND/FORM A CASE

- {a1}is **not** **Acceptable. No Valid Case for it.**



ARISTOTLE (Topics): " … and while sustaining our argument we avoid saying anything self-contradictory.

Does not defend against c1, since a2, a needed defence can not be defended against c3.

Also, no valid case for a1, as it attacks the needed defence a5.

Reasoning in Argumentation

- Reasoning: Building valid (admissible) Cases

- Reasoning for φ:
  - Build a case for φ
  - Show no case′ for ¬φ

- Could be in Dilemma:
  - Have a case for φ
  - Have a case′ for ¬φ

Reasoning in Argumentation

Building valid (admissible) Cases

- **Practical Features of Arg-based Reasoning**
  - **On Demand/Lazy computation**
    - **Argumentation Arena is NOT static but Dynamic**
  - **Depends on the Current Environment/Audience to:**
    - **Produce Counter-Arguments**
    - **Strengthen the subsequent defense arguments.**

ARGUMENTATION THEORY for PRACTICE

Realizing Argumentation

Abstract ⇒ Structured Argumentation

- **Arguments build via a set of Argument Schemes, AS: a1=AS1(t), a2=AS2(t) … t some world parameters.**

- **Att ≡ C, an application dependent conflict notion, C.**

- **Def via an application relative strength, ב, on AS**

"a defends against b" iff in conflict and not weaker

(a,b) ∈ Def iff they are *conflict* (*i.e.* (a,b) ∈ *Att*) & a is not weaker than b, i.e. if (b,a) ∈ ב then (a,b) ∈ ב.

Application Argumentation Frameworks

<Args, Att, Def> ⇨ <AS, C, ב>

- Conflict, C, is Static

- Strength, ב, is Dynamic/Application Dependent

Strength, ב, is Context Dependent - Conditional on perceived current environment

Construction of Arguments (Args)

Arguments are constructed as instantiations of argument schemes As

As="Premises --> Position/Claim"

- **Argument Schemes are programmed or authored or learned from data/experience**

Construction of Arguments (Args)

- Argument Schemes: Licenses/Topoi for arguments

- These are links (not rules):
  - Premises/ἔνδοξα ----→ Position
    - E.g. arg1: Ambulance ----→ Serious_Injury

- Arguments "enter" activated dynamically from "sensory" premise information
  - E.g. Activated from the text: "An ambulance arrived."

Construction of Attacks/Conflict (Att/C)

- **Attacks result from a conflict on the claim or the premises of an argument (that is attacked).**

arg: Premises --- > Claim

Three types of Attacks:

- **Rebuttal: Conflict on the Claim**
- **Undermining: Conflict on a/the Premises**
- **Undercutting: Conflict on the Link**

Construction of Defense/Strength (ב)

- The strength/priority relation, ב, between arguments is Context Sensitive - Not statically Global.

- Dynamically Conditional on the (partially) perceived state of current environment.

Need to decide on ב at any given situation!
HOW?

In the GORGIAS FRAMEWORK
via Argumentation ⇒ Priority Arguments

GORGIAS ARGUMENTATION

GORGIAS Argumentation Framework

- **As – a set of Object-level argument schemes**
- **C – negation and other application incompatibilities**
- **ב - a Set of Priority Argument Schemes, *Prs, of the form:***

“Premises/Conditions ---> as1 > as2”

<As, C, ב>

<As, C, *Prs*> ≡ <As U *Prs*, C >

GORGIAS Argumentation Framework

<As U *Prs*, C >

- **Composite Arguments Δ =(O,P)**

- **Δ =(O,P) is admissible iff**

1) **Δ is Conflict-free**
2) **Δ defends against any attack, A=(O1,P1): if P1 supports a>δ then P supports δ'>a'**

Δ defends against A: (1) it contains a stronger argument or (2) Δ and A are non-comparable

GORGIAS Reasoning/Decision Making
EXAMPLE SAF1

Gorgias theory SAF1 = <As U *Prs*, *C* >:

- ***C*** given by two conflicting **options**: opt1 & opt2
- *As* = {r1: cond1 ---> opt1 & r2 : cond1 ---> opt2}
- *Prs* = {R1: true ---> r1>r2 & R2: cond2 ---> r2>r1}

1. First, consider a Scenario where only cond1 holds.

2. Then extend the Scenario with cond2 also holding.

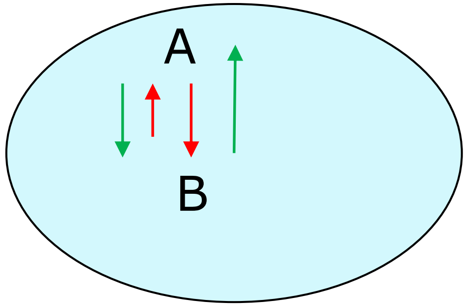
GORGIAS Reasoning/Decision Making
SAF1 = <As U *Prs*, C >

Consider a Scenario where cond1 holds.

- *As* = {r1: cond1 ---> opt1 & r2 : cond1 ---> opt2}
- *Prs* = {R1: true ---> r1>r2 & ~~R2: cond2 ---> r2>r1~~}

Phase 1: Reasoning at Object-level:

Composite arguments A=(r1, {}) , B= (r2, {})
attack and defend against each other.



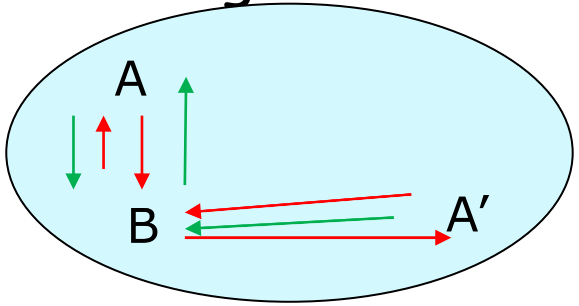
Phase 2: Reasoning at Higer/Priority-level:

Can these arguments be strengthened?

GORGIAS Reasoning/Decision Making
SAF1 = <As U Prs, C >

Phase 2: Reasoning at Higer/Priority-level:

- A'={r1, R1}, strengthens A
- A' **defends against** B but B does **not defend against** A'
  - R1 makes r1>r2 but B does **not** make its argument r2 stronger.
- Also, B cannot be strengthened.



Therefore, only admissible arguments for **opt1**.

Hence Definite Decision of **opt 1**.

Consider an Extended Scenario where cond2 holds.

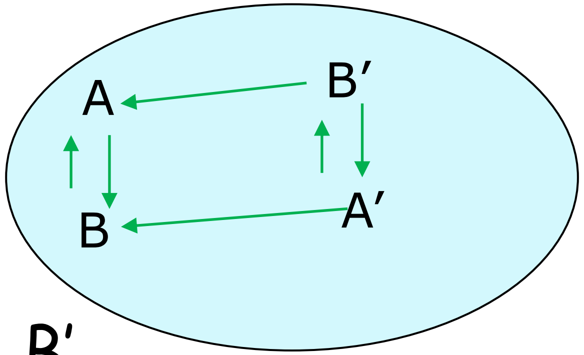
Can B can be strengthened?

GORGIAS Reasoning/Decision Making
SAF1 = <As U Prs, C >

Consider the extended Scenario where cond2 also holds.

Then R2: cond2 ---> r2>r1 is also active.

Phase 2: Reasoning at Priority-level:



- B′={r2, R2} strengthens B
- B′ **defends against** A but A does **not defend** B′
- A′ **defends** B′ (R1 in A′ makes r1>r2) and **vice versa** (R2 in B′ makes r2>r1)

Now A′ and B′ are admissible: both options are validly supported.

Hence the Decision is in Dilemma

Explanations in Argumentation

- Argumentation generates Explanations!
  - Explanations are directly extracted from the Valid/Admissible set, S, of arguments, i.e. from the Case for a conclusion
  - Argument in S in support : Attributive part of Explanation
  - Defending arguments in S: Contrastive part of Explanation

- Explanation from GORGIAS
  - Attributive: from Object-level arguments
  - Contrastive: from Priority Arguments
  - Actionable: from Hypotheses/Abducibles

Exercise

Consider the story below in the following 3 scenes:

- Mary was very busy at the office.
- She did not want to be distracted.
- **Her phone rang.**

- It was her mother phoning.

- Mary's mother fell ill last week.
- She was still (very) ill in the hospital.

For **each scene** consider the question **"Will Mary answer the phone, Yes or No?"** Construct the arguments **for** and **against** answering the phone, showing also the attack and defense or priority relations between the arguments.

Draw the argumentation arena for each scene and in each case find the acceptable (set of) arguments supporting the two possible options/conclusions of Yes or No.

PREVIEW

Argumentation-based Software Methodologies & Systems Design/Architectures

Computational Argumentation: a "Roadmap"

From <Args, ATT> ... to <Args, Att, Def> ... to

... to <As, C, ב> ...

... to GORGIAS <As U *Prs*, C > ...

From Theory to Practice

... to *SoDA* Methodology for Knowledge Acquisition

... to rAIson

... to Applications

From Theory to Practice

"Normally, allow a call. When at work deny a call from an unknown number. When busy at work also deny a call from a known number unless it is an emergency family call. Always allow a call from my manager. "

Methodology for Knowledge Representation

Options: allow a call, deny a call.

Factors: at work, known/unknown, busy, …, manager

Keys for Preferences: Normally, unless, always, …

STEP 1: Identify 2 + 1 groups of information

Challenge of Acquiring Knowledge

- **Acquisition of Contextual Knowledge**
  - **From Experts or Policy Document**
  - **From Data of Examples**

CHALLENCE: Facilitate the extraction of the hidden/implied preferences in high-level specs.

- ***SoDA Methodology***
  - **Acquisition from high-level Problem Specs**
  - **From Natural Language specs.**

From Theory to Practice

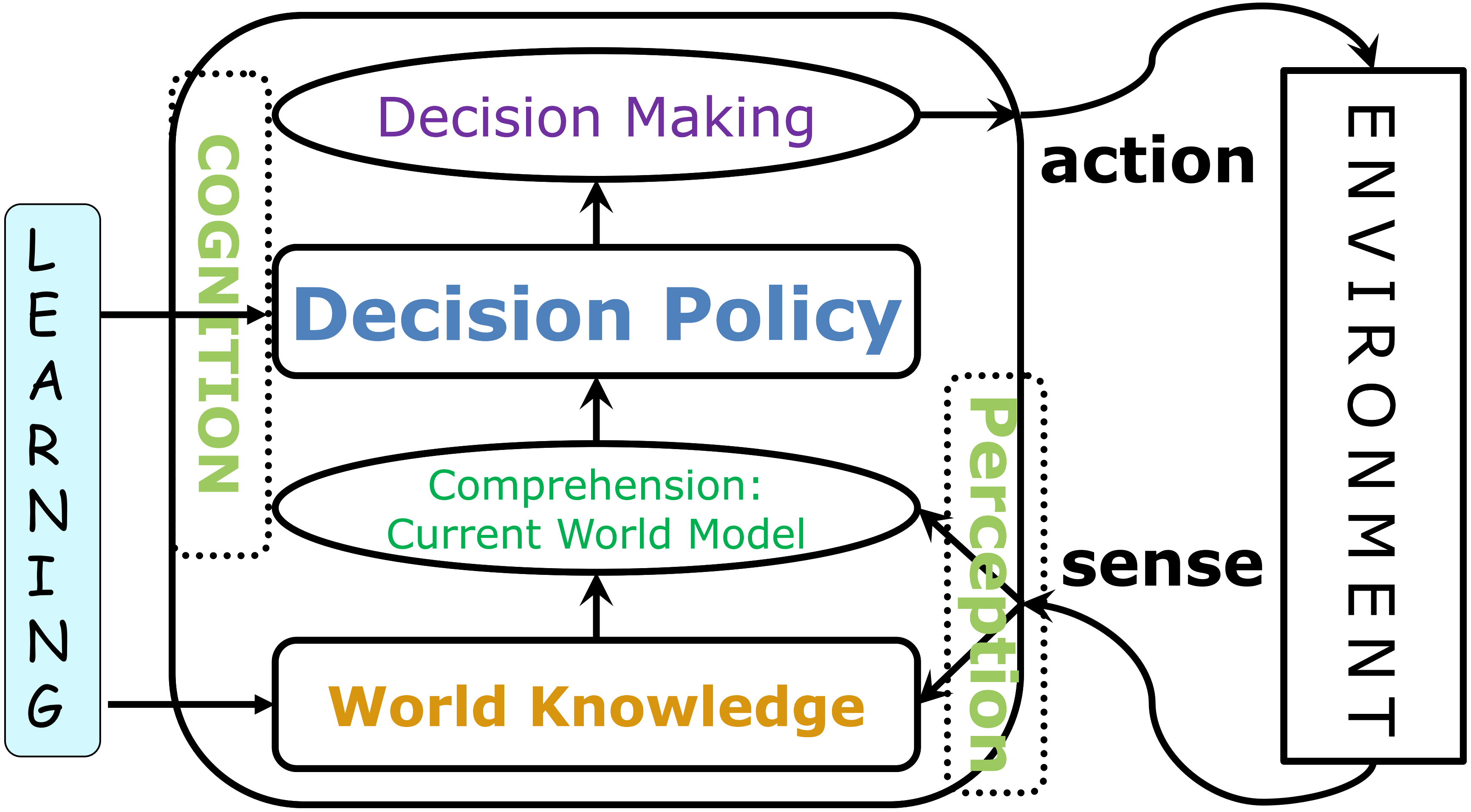
"Normally, allow a call. When at work deny a call from an unknown number. When busy at work also deny a call from a known number unless it is an emergency family call. Always allow a call from my manager. "

***SoDA:* Methodology for Knowledge Representation**

STEP 2: Identify Scenario-based Preferences

[SEE EXTRA SLIDES DETAILS OF THIS EXAMPLE:
STEP2: Scenario-based Preferences
STEP3: Gorgias Representation/Code]

Building Decision Machines
System Architecture



HANDS-ON DEVELOPMENT

Your Decision Problem/Policy

- **Professional Problem/Policy**
  - **Insurance Policy, Risk Management, Liability, Marketing, …**
- **Personal Policy – Cognitive Assistant**
  - **Hotel Assistant, …**
  - **Email/Social Media/Calendar Assistant**

Submit a one paragraph high-level description of your Decision Policy

Email us with Subject "Hands on Day 1"

Example Problems (1)

"Normally, allow a call. When at work deny a call from an unknown number. When busy at work also deny a call from a known number unless it is an emergency family call. Always allow a call from my manager. "

Methodology for Knowledge Representation

Options: allow a call, deny a call.

Factors: at work, known/unknown, busy, …, manager

Keys for Preferences: Normally, unless, always, …

STEP 1: Identify 2 + 1 groups of information

Example Problems (2)

- **Example D2: Travel Assistant (Personal Policy)**

"For long distance travel it is possible to use all means of transport. If the bus stop is near, I prefer to get the bus. If it is a cold day, I can take the metro or a taxi. If the bus stop is near and it is a cold day, I prefer to take the metro, except if it rains, in which case I will take a taxi. I do not take the taxi when I am short on funds."

Options: take a taxi, take the bus, take the metro.

Example Problems (3)

◻ Example D3: Seller Policy

"The primary choice is to sell at regular price. However, if a customer has spent more than 200 euros during the last month then sell at a promotional price. During the high season still sell at regular price. If the quantity of the product is low and the customer is not regular, then cannot sell. Special products are not sold at promotional price."

Options: sell at regular price, sell at promotional price, cannot sell

Example Problems (4)

◻ Example D3: Medical Liability (Legal Policy)

"When a professional misconduct is committed by a doctor in a public sector establishment then we have either personal accountability of the doctor, or public sector support. If the doctor is tenured then we have public sector support, except if the doctor has committed the misconduct while practicing outside their specialty, in which case the doctor is personally accountable. When the professional misconduct is committed by a doctor in a private sector establishment and the doctor is tenured then the doctor has private sector support. Always, the doctor has personal accountability if they practice as an independent entity."

Options: personal accountability of a doctor, private sector support, public sector support

EXTRA SLIDES

READING for Details

Theory of Argumentation
Some References

- Kakas, Mancarella, Dung & Dimopoulos (1994 & 1995), "Logic Programing without Negation as Failure", ICLP94 and ISLP95.
- A. C. Kakas, P. Moraitis (2003), Argumentation based decision making for autonomous agents. AAMAS 2003: 883-890.
- N. I. Spanoudakis, A. C. Kakas & A. Koumi (2022), Application Level Explanations for Argumentation-based Decision Making. ArgXAI@COMMA 2022.

Extra READING

Theory of Argumentation
Extra READING

- Dung (1995), On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, JAI Vol. 77(2): 321-357
- Tutorials on Structured Argumentation, Argument & Computation, vol. 5, no. 1, 2014.
- E. Dietz et al, (2021) Computational Argumentation and Cognitive AI, ACAI 2021: 363-388.

SoDA Methodology:
Example of Call Assistant

SoDA: Example Call Assistant

"Normally, allow a call. When at work deny a call from an unknown number. When busy at work also deny a call from a known number unless it is an emergency family call. Always allow a call from my manager. "

Methodology for Knowledge Representation

STEP 1: Identify 2 + 1 groups of information

Options: allow a call, deny a call.

Factors: at work, known/unknown, busy, …, manager

Keys for Preferences: Normally, unless, always, …

SoDA: Example Call Assistant

"Normally, allow a call."

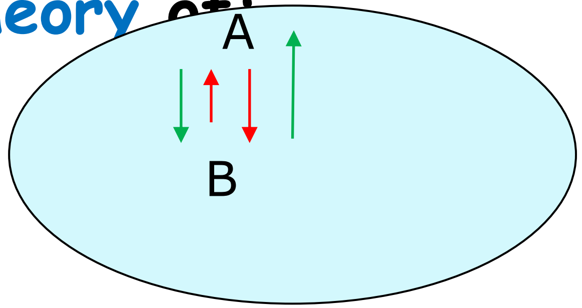
This asks us to consider a scenario with no extra information.

In which there is a preference of allow over deny.

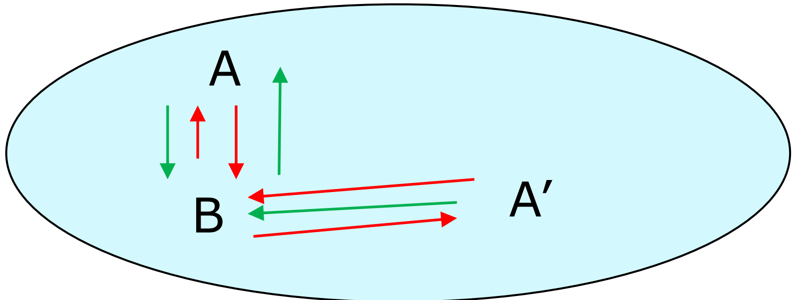
We express this via Scenario-based Preference: <1, {}, allow(call)>

This is automatically translated to GORGIAS Arg. Theory of:

- As = {r1: call ---> allow; r2 : call ---> deny; …}
- Prs = {R1: true ---> r1>r2; … }



Then follow Reasoning as above (opt1=allow, opt2=deny) to give allow as a definite decision. [Given in next 6 Slides.]



Reasoning/Decision Making in GORGIAS: Call Assistant

"Normally, allow a call."

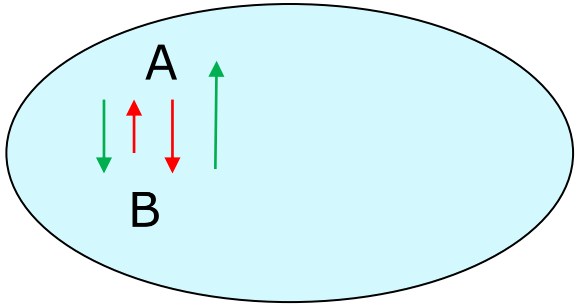
This asks us to consider a scenario with no extra information.

In which there is a preference of allow over deny.

We express this via Scenario-based Preference: <1, {}, allow(call)>

Phase 1: Reasoning at Object-level:

- A={r1(call1)} object-level argument supports **allow**.
- B={r2(call1)} object-level argument supports **deny**.
  - A attacks and defends B and vice versa.

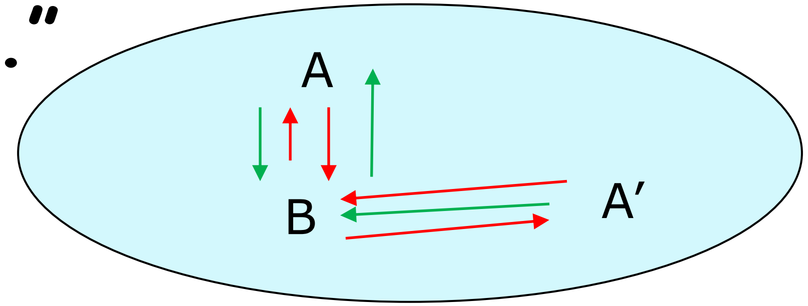


Phase 2: Reasoning at Higer/Priority-level:

- Can these arguments be strengthened?

Reasoning/Decision Making in GORGIAS Argumentation

"Normally, allow a call."



Phase 2: Reasoning at Higer/Priority-level:

- Can these arguments be strengthened?
- A′={r1(call1), R1(call1)}, with R1: true --> r1 > r2 strengthens A
- A′ **defends against** B but B does **not defend against** A′
- B cannot be strengthened (in this scenario no other active priority arg.)
- Hence A′ admissible and B cannot be made admissible

Hence **definite decision**: **allow** call 1.

SoDA: Example Call Assistant cnt

"When busy … deny a call … unless it is an emergency family call."

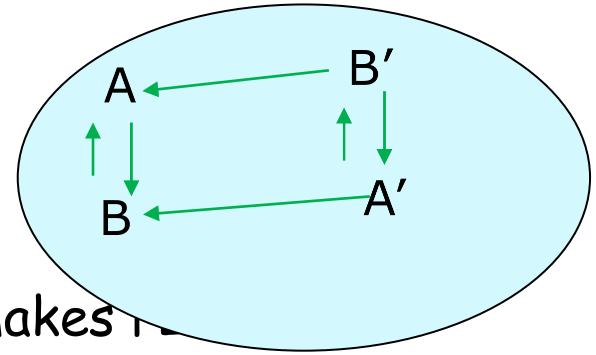
Hierarchy of Scenario-based Preferences:

<1, {}, allow(call)>

<2, {busy}, deny(call)>

<3, {busy, efamily(call)}, allow(call)>

- **New Priority arg.:** R2: busy ---> r2>r1
- **Priority-level Reasoning in Scenario {busy}:**
  - A′={r1(call), R1(call)} strengthens A and B′={r2(call), R2(call)}
  - A′ defends B′ (R1 in A′ makes r1>r2) and vice versa (R2 in B′ makes r



A′ and B′ are admissible, i.e. both options are validly supported: In Dilemma

[See next three slides for details & extension of reasoning when efamily holds]

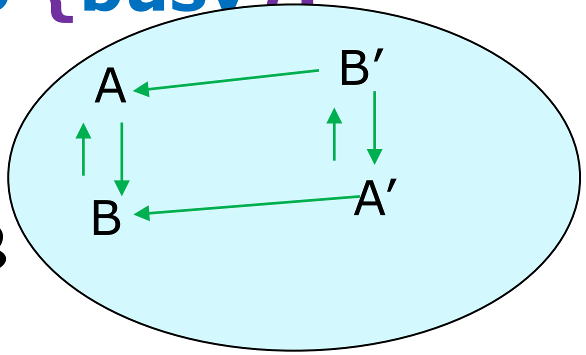
SoDA: Example Call Assistant cnt

"When busy … deny a call … unless it is an emergency family call."

Hierarchy of Scenario-based Preferences:

<1, {}, allow(call)>
<2, {busy}, deny(call)>
<3, {busy, efamily(call)}, allow(call)>

Priority-level Reasoning in Scenario {busy}:



- A′={r1(call), R1(call)} strengthens A
  - A′ defends B but B does **not** defend A′
- B′={r2(call), R2(call)} R2: busy ---> r2>r1 strengthens B
  - B′ defend A but A does **not** defend B′
- A′ defends B′ (R1 in A′ makes r1>r2) and vice versa (R2 in B′ makes r2>r1)

A′ and B′ are admissible, i.e. both options are validly supported: in Dilemma

Reasoning/Decision Making in GORGIAS Argumentation

Hierarchy of Scenario-based Preferences:

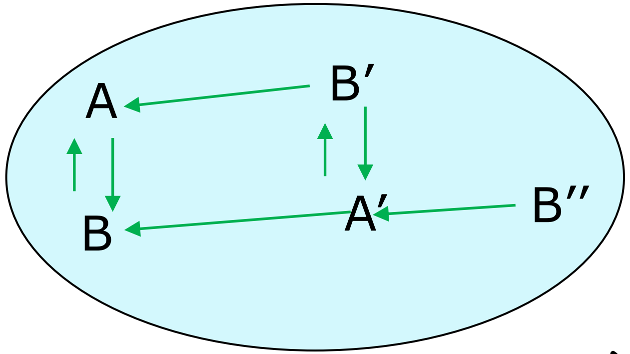
<1, {}, allow(call)>

<2, {busy}, deny(call)>

Priority-level Phase in scenario {busy}:

We should NOT be in a Dilemma

- B''={r2(call), R2(call), C2(call)} with the higher-level priority argument C2: true --- > R2>R1 strengthens B'
- B'' **defends against** A' but not vice-versa
  - **Their conflict is on priority between R1 and R2**
- Also, A' cannot be strengthened (in this scenario by any active priority arg.)



Hence B cannot be made admissible. Hence **sceptical decision: deny** the call.

Reasoning/Decision Making in GORGIAS Argumentation

"When busy … deny a call … unless it is an emergency family call."

Hierarchy of Scenario-based Preferences:

<1, {}, allow(call)>
<2, {busy}, deny(call)>
<3, {busy, efamily(call)}, allow(call)>

Reasoning in Scenario {busy, efamily(call2)}:

Exercise or Lecture 2

Connection between Abstract Argumentation Frameworks

Abstract Argumentation Frameworks

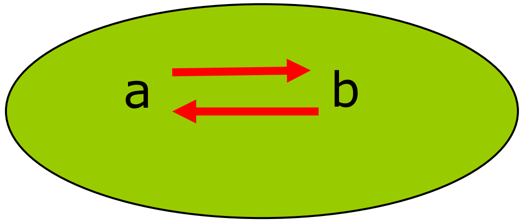
Connection

Argumentation Frameworks:

<Args, ATT> & <Args, Att, Def>

- **One connection:**
  - **Argument "a" Attacks "b" : (a,b) ∈ ATT iff:**
    - **a** and **b** *are in conflict* (*i.e.* **(a,b)** ∈ ***Att***)
    - **If (b,a)** ∈ ***Def*** **then (a,b)** ∈ ***Def***
  - **If (a,b) ∈ ATT and (b,a) ∉ ATT**

    then b is weaker than a, i.e. (b,a) ∉ *Def*



Commonsense Reasoning in Argumentation

Commonsense Reasoning

- **Example: Reasoning about Action & Change**

"Bob came home and found the house in darkness. He turned on the light switch."

Is the house still in darkness?

"The power cut had turned the house into darkness. Bob came home and turned on the light switch."

Defense/Strength (ב)

- The strength relation, ב, between arguments
  - Application Dependent - Sensitive

- Conditional on the partially perceived state of current environment.

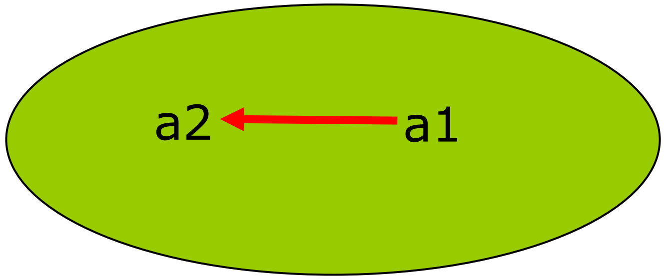
- In some rare cases, we have almost global priorities:
  - Causal Arguments ב Persistence Arguments
  - Precondition Arguments ב Causal Arguments
    - Necessary Conditions ב Sufficient Conditions
  - Pragmatic Arguments ב Motivational Arguments

Commonsense Example 1

"Bob came home and found the house in darkness. He turned on the light switch. …"

- **a1**={turn_on_switch --> light_on ; light_on --> no darkness}
  - **a1 supports no darkness@T^+**
- **a2**={darkness@T --> darkness@T^+}
  - **a2 supports darkness@T^+**

- **a1** is a **causal** argument; **a2** is a **persistence** argument **=> a1 ב a2**



{a1} acceptable

Case for no darkness at T^+

No Case for darkness at T^+

Commonsense Example 1 (alt)

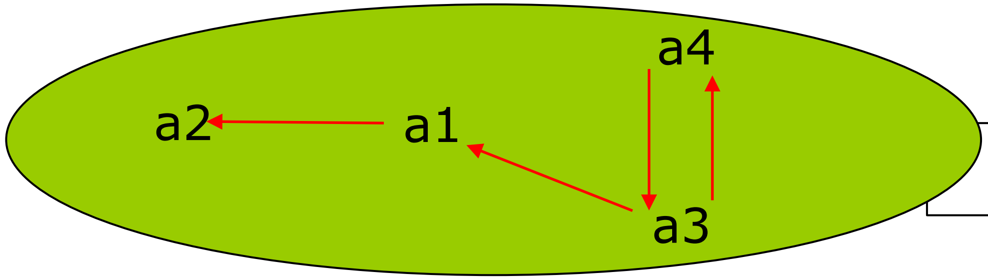
"The power cut had turned the house into darkness. Bob came home and turned on the light switch."

- **a1**={turn_on_switch **-→** light_on, light_on **-→** no darkness}
- **a2**={darkness@T **-→** darkness@T+}

New arguments:

- **a3**={power_cut@T, power_cut **-→** no electricity}
- **a4** ={no power_cut@T}

With a3 > a1 and **a3 ≈ a4 (subjective!)**.



{a2,a3} acceptable
{a1,a4} acceptable

Case for either darkness or not at T^+

Dilemma for darkness at T^+

SoDA Methodology

SoftwareDevelopment via Argumentation
SoDA Methodology

Identify the Language of Options & Factors for Preference

- **Consider application scenarios** and **state** the preferred/desired option(s) in each scenario.
  - **Identify** different **initial scenarios.**
- Successively **refine** the **scenarios**, **restating** at each **refinement** the new **preferred option(s).**
- Considering **combinations** of conflicting of **scenarios**

Hierarchies of Scenario-based Preferences (SBPs)

*So*ftware *D*evelopment via *A*rgumentation
SoDA Methodology

Hierarchies of Scenario-based Preferences (SBPs)

↓

GORGIAS Argumentation Framework

Authoring - no coding - Knowledge Representation

[See Extra Slides for Call Assistant Example]

Practical Challenges

Building Decision Machines

Two major challenges

1. **Acquisition** of problem Knowledge - **Decision Policy**
   - **At a Language Level of the Application – Natural Language?**
   - **Extracting Hidden Preferences from Natural Language Specs**

2. **Middleware** from Sensory Information to Policy Concepts
   - **Comprehension** of current **Context** of the application environment from its low-level sensory information

Intelligence is in the Abstraction of the Decision Policy
Large number of cases grouped into high-level concepts