Reasoning with Inconsistency

Mike Giancola

Selmer Bringsjord



Intermediate Formal Logic & AI (IFLAI-2) September 24, 2020



Sponsored by





• Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies

- Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies
 - Underlying logic could be inconsistent

- Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies
 - Underlying logic could be inconsistent
 - Inconsistency could be inherent to the domain

- Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies
 - Underlying logic could be inconsistent
 - Inconsistency could be inherent to the domain
 - e.g. Belief Revision

- Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies
 - Underlying logic could be inconsistent



- Inconsistency could be inherent to the domain
 - e.g. Belief Revision

 Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies

Hard!

- Underlying logic could be inconsistent
- Inconsistency could be inherent to the domain
 - e.g. Belief Revision

 Al agents utilizing automated reasoning will undoubtedly encounter inconsistencies

Hard!

- Underlying logic could be inconsistent
- Inconsistency could be inherent to the domain
 - e.g. Belief Revision
- Goal: Build an Al agent which can detect inconsistencies and find solutions



Т















• Each pilot's display has its own set of sensors





- Each pilot's display has its own set of sensors
 - A faulty sensor feeding the PF's display gave an incorrect reading



- Each pilot's display has its own set of sensors
 - A faulty sensor feeding the PF's display gave an incorrect reading
 - Typically, a Comparator Function continuously monitors sensor readings



- Each pilot's display has its own set of sensors
 - A faulty sensor feeding the PF's display gave an incorrect reading
 - Typically, a Comparator Function continuously monitors sensor readings
 - This was disabled by a Declutter Function

• Instantly detect the inconsistency

- Instantly detect the inconsistency
 - In this case, notice that Pilot I's sensor reading seems unusual, and that Pilot 2's reading matches the backup instruments.

- Instantly detect the inconsistency
 - In this case, notice that Pilot I's sensor reading seems unusual, and that Pilot 2's reading matches the backup instruments.
- Find a solution

- Instantly detect the inconsistency
 - In this case, notice that Pilot I's sensor reading seems unusual, and that Pilot 2's reading matches the backup instruments.
- Find a solution
 - In this case, send sensor readings from Pilot 2's sensors to Pilot I's display, ignoring faulty data

A Solution in OSCAR



🕜 OSCAR_Files — -bash — 204×54

A Solution in OSCAR



🕜 OSCAR_Files — -bash — 204×54

```
Problem #1
This is a solution to the airplane crash scenario
Given premises:
      ~(ReadsNormal iru1)
                                justification = 1.0
      (ReadsNormal iru2)
                               justification = 1.0
      (MatchesBackup iru2)
                               justification = 1.0
      (all i1)(all i2)(((~(ReadsNormal i1) & (ReadsNormal i2)) & (MatchesBackup i2)) -> NormalAttitude)
                                                                                                                         justification = 0.9
Ultimate epistemic interests:
     NormalAttitude
                          interest = 0.9
    FORWARDS PRIMA FACIE REASONS
                        {~(ReadsNormal iru1)} ||=> ~NormalAttitude strength = 0.6
       PF-REASON_1.1:
 Interest in NormalAttitude
  is answered affirmatively by node 14
Elapsed time = 0.022 sec
 ARGUMENT #1
 This is an undefeated argument of strength 0.9 for:
      NORMALATTITUDE
 which is of ultimate interest.
3. (MatchesBackup iru2)
                          GIVEN
1. ~(ReadsNormal iru1)
                         GIVEN
4. (all i1)(all i2)(((~(ReadsNormal i1) & (ReadsNormal i2)) & (MatchesBackup i2)) -> NormalAttitude)
                                                                                              GIVEN
7. (all i2)(((~(ReadsNormal x0) & (ReadsNormal i2)) & (MatchesBackup i2)) -> NormalAttitude)
                                                                                       UI from { 4 }
8. (((~(ReadsNormal x0) & (ReadsNormal x1)) & (MatchesBackup x1)) -> NormalAttitude)
                                                                               UI from { 7 }
9. ((~(ReadsNormal x0) & (ReadsNormal x1)) -> ((MatchesBackup x1) -> NormalAttitude))
                                                                                exportation from { 8 }
11. (~(ReadsNormal x0) -> ((ReadsNormal x1) -> ((MatchesBackup x1) -> NormalAttitude)))
                                                                                  exportation from { 9 }
12. ((ReadsNormal x1) -> ((MatchesBackup x1) -> NormalAttitude)) modus-ponens1 from { 11 , 1 }
 2. (ReadsNormal iru2)
                        GIVEN
13. ((MatchesBackup iru2) -> NormalAttitude)
                                            modus-ponens1 from { 12 , 2 }
                     modus-ponens1 from { 13 , 3 }
 14. NormalAttitude
 Argument #2 support defeaters for this argument.
 This argument supports defeaters for { link 5 for node 6 } thereby providing defeaters for argument #2
 ARGUMENT #2
 This is a defeated argument for:
      (~
       (ALL I1
        (ALL I2
        (-> (& (& (~ (READSNORMAL I1)) (READSNORMAL I2)) (MATCHESBACKUP I2))
         NORMALATTITUDE))))
1. ~(ReadsNormal iru1)
                         GIVEN
                     PF-REASON_1.1 from { 1 }
6. ~NormalAttitude
15. ~(all i1)(all i2)(((~(ReadsNormal i1) & (ReadsNormal i2)) & (MatchesBackup i2)) -> NormalAttitude)
                                                                                                INVERSION_FROM_CONTRADICTORY_NODES_14_AND_6 from { 6 }
2. (ReadsNormal iru2)
                        GIVEN
 4. (all i1)(all i2)(((~(ReadsNormal i1) & (ReadsNormal i2)) & (MatchesBackup i2)) -> NormalAttitude)
                                                                                              GIVEN
 7. (all i2)(((~(ReadsNormal x0) & (ReadsNormal i2)) & (MatchesBackup i2)) -> NormalAttitude)
                                                                                      UI from { 4 }
8. (((~(ReadsNormal x0) & (ReadsNormal x1)) & (MatchesBackup x1)) -> NormalAttitude)
                                                                               UI from { 7 }
9. ((~(ReadsNormal x0) & (ReadsNormal x1)) -> ((MatchesBackup x1) -> NormalAttitude))
                                                                                exportation from { 8 }
 11. (~(ReadsNormal x0) -> ((ReadsNormal x1) -> ((MatchesBackup x1) -> NormalAttitude)))
                                                                                exportation from { 9 }
 12. ((ReadsNormal x1) -> ((MatchesBackup x1) -> NormalAttitude)) modus-ponens1 from { 11 , 1 }
 13. ((MatchesBackup iru2) -> NormalAttitude)
                                            modus-ponens1 from { 12 , 2 }
3. (MatchesBackup iru2)
                          GIVEN
 14. NormalAttitude
                     modus-ponens1 from { 13 , 3 }
 Arguments #1, #2 support defeaters for this argument.
```

This argument supports defeaters for { link 4 for node 4 } thereby providing defeaters for arguments #1, #2



 A nascent automated reasoner for generating and adjudicating arguments

RAIRLab/ShadowAdjudicator: A X	+
\leftrightarrow \rightarrow C $rac{1}{2}$	🗊 🔒 https:// github.com /RAIRLab/ShadowAdjudicator 🖂 🏠 🖳 🚱 💆 😹 🗰
Search or jump to	Pull requests Issues Marketplace Explore
	RAIRLab / ShadowAdjudicator
	<> Code ① Issues 0 11 Pull requests 0 ⊙ Actions III Projects 0 □ Wiki ⑦ Security 0 ∠ Insights ⑧ Settings
	A system for adjudicating arguments amongst two or more AI agents reasoning in a quantified modal logic. Edit Manage topics
	-০- 11 commits 🐉 1 branch 💮 0 packages 🚫 0 releases মে 1 contributor ক্র্রি AGPL-3.0
	Branch: master + New pull request Create new file Upload files Find file Clone or download +
	Migiancola Added ability to (roughly) track inputs to ShadowProver for outputtin Latest commit 250ffc8 15 days ago
	adjudicator Added ability to (roughly) track inputs to ShadowProver for outputtin 15 days ago
	prover @ 714b7ec Added ShadowProver as a submodule last month
	.gitignore Initial commit 4 months ago
	LICENSE Initial commit 4 months ago
	README.md Implemented modus ponens over SF beliefs, created demo last month
	D README.md
	ShadowAdjudicator
	A system for adjudicating arguments amongst two or more AI agents reasoning in a quantified modal logic.



- A nascent automated reasoner for generating and adjudicating arguments
- Builds upon ShadowProver

RAIRLab/ShadowAdjudicator: A X	+
-> C û	・ 回 🔒 https://github.com/RAIRLab/ShadowAdjudicator … 🖂 🕹 🕷 色 🦉 参 開
Search or jump to	Pull requests Issues Marketplace Explore
	RAIRLab / ShadowAdjudicator
	↔ Code 🕕 Issues 0 👘 Pull requests 0 ⊙ Actions 🛄 Projects 0 🛄 Wiki ⑦ Security 0 🗠 Insights ⑧ Settings
	A system for adjudicating arguments amongst two or more Al agents reasoning in a quantified modal logic. Edit Manage topics
	->- 11 commits 💱 1 branch 🛞 0 packages 🛇 0 releases ৪২ 1 contributor ক্রি AGPL-3.0
	Branch: master - New pull request Create new file Upload files Find file Clone or download -
	Ng mjgiancola Added ability to (roughly) track inputs to ShadowProver for outputtin Latest commit 250ffc8 15 days ago
	adjudicator Added ability to (roughly) track inputs to ShadowProver for outputtin 15 days ago
	prover @ 714b7ec Added ShadowProver as a submodule last month
	.gitignore Initial commit 4 months ago
	LICENSE Initial commit 4 months ago
	README.md Implemented modus ponens over SF beliefs, created demo last month
	TREADME.md
	ShadowAdjudicator
	A system for adjudicating arguments amongst two or more AI agents reasoning in a quantified modal logic.



- A nascent automated reasoner for generating and adjudicating arguments
- Builds upon ShadowProver
 - Uses ShadowProver for subproofs of modal/FOL/PL formulae

 → C û Search or jump to 	🖸 🔓 https://github.com/RAIRLab/ShadowAdjudicator		
Search or jump to		⊌ ☆	⊻ II\ 🖸 Θ 📲 🛪 IIII
· · · · · · · · · · · · · · · · · · ·	Pull requests Issues Marketplace Explore		4 + • 🚦
ſ	RAIRLab / ShadowAdjudicator	ⓒ Unwatch ▼ 7 7	0
	C>Code ① Issues 0 □ Pull requests 0	⑦ Security 0 ∠ Insights I Settings	
,	A system for adjudicating arguments amongst two or more AI agents reasoning in a qu danage topics	antified modal logic.	Edit
	->- 11 commits 🐉 1 branch 💮 0 packages 🚫 0 releases	ম 1 contributor ক্রা AGPL-3.0	
[Branch: master - New pull request Create ne	w file Upload files Find file Clone or download	ad 🗸
	mjgiancola Added ability to (roughly) track inputs to ShadowProver for outputtin	Latest commit 250ffc8 15 days a	ago
	Added ability to (roughly) track inputs to ShadowProver for out	tputtin 15 days a	igo
	prover @ 714b7ec Added ShadowProver as a submodule	last mor	nth
	.gitignore Initial commit	4 months a	ago
	.gitmodules Added ShadowProver as a submodule	last mor	nth
	LICENSE Initial commit	4 months a	igo
	README.md Implemented modus ponens over SF beliefs, created demo	last mor	nth
	印 README.md		P
	ShadowAdjudicator	a quantified modal lonic	



- A nascent automated reasoner for generating and adjudicating arguments
- Builds upon ShadowProver
 - Uses ShadowProver for subproofs of modal/FOL/PL formulae
 - Implements an algorithm and inference schemata for generating arguments with strength factors

→ C û	🛛 🔒 https://github.com/RAIRLab/ShadowAdjudicator 🚥 🖂 📩 🔟 🖸 🕹 🛸 🛱
	Pull requests Issues Marketplace Explore
	RAIRLab / ShadowAdjudicator
	<>Code ① Issues 0 11 Pull requests 0 ② Actions □ Projects 0 □ Wiki ③ Security 0 ∠ Insights Security.com
	A system for adjudicating arguments amongst two or more AI agents reasoning in a quantified modal logic. Edit Manage topics
	->-11 commits 🐉 1 branch 🛞 0 packages 🖓 0 releases 🙉 1 contributor ⊕ AGPL-3.0
	Branch: master - New pull request Create new file Upload files Find file Clone or download -
	Nigiancola Added ability to (roughly) track inputs to ShadowProver for outputtin Latest commit 250ffc8 15 days ago
	adjudicator Added ability to (roughly) track inputs to ShadowProver for outputtin 15 days ago
	prover @ 714b7ec Added ShadowProver as a submodule last month
	Initial commit 4 months ago
	LICENSE Initial commit 4 months ago
	B README.md Implemented modus ponens over SF beliefs, created demo last month
	TREADME.md
	ShadowAdjudicator
	A system for adjudicating arguments amongst two or more AI agents reasoning in a quantified modal logic.



- Selmer Bringsjord, M. Giancola, N.S. Govindarajulu. "Toward Defeasible Multi-Operator Argumentation Systems for Culturally Aware Social Robots that Carry Humans Inside Them". In *Robophilosophy* 2020. Forthcoming.
 - Link to preprint
 - Link to final copy will be available <u>here</u> as soon as it's finished

- Selmer Bringsjord, M. Giancola, N.S. Govindarajulu. "Toward Defeasible Multi-Operator Argumentation Systems for Culturally Aware Social Robots that Carry Humans Inside Them". In *Robophilosophy* 2020. Forthcoming.
 - Link to <u>preprint</u>
 - Link to final copy will be available <u>here</u> as soon as it's finished
- https://rair.cogsci.rpi.edu/projects/automated-reasoners/oscar/
 - Software to run OSCAR

- Selmer Bringsjord, M. Giancola, N.S. Govindarajulu. "Toward Defeasible Multi-Operator Argumentation Systems for Culturally Aware Social Robots that Carry Humans Inside Them". In *Robophilosophy* 2020. Forthcoming.
 - Link to <u>preprint</u>
 - Link to final copy will be available <u>here</u> as soon as it's finished
- https://rair.cogsci.rpi.edu/projects/automated-reasoners/oscar/
 - Software to run OSCAR
 - For files to run example from today, email me: mike.j.giancola@gmail.com.
- Licato, John. "Formalizing deceptive reasoning in breaking bad: Default reasoning in a doxastic logic." 2015 AAAI Fall Symposium Series. 2015.
 - <u>https://www.aaai.org/ocs/index.php/FSS/FSS15/paper/download/</u> <u>11669/11486</u>