Computational Models of Argument
F. Toni et al. (Eds.)

doi:10.3233/FAIA220167

PyArg for Solving and Explaining Argumentation in Python: Demonstration

AnneMarie BORG [a], Daphne ODEKERKEN [a,b,1]
[a] *Department of Information and Computing Sciences, Utrecht University*
[b] *National Police Lab AI, Netherlands Police*
ORCiD ID: AnneMarie Borg https://orcid.org/0000-0002-7204-6046 , Daphne Odekerken https://orcid.org/0000-0003-0285-0706

Abstract. We introduce `PyArg`, a Python-based solver and explainer for both abstract argumentation and ASPIC$^+$. A large variety of extension-based semantics allows for flexible evaluation and several explanation functions are available.

Keywords. Abstract Argumentation, Structured Argumentation, Explainable Artificial Intelligence, Python

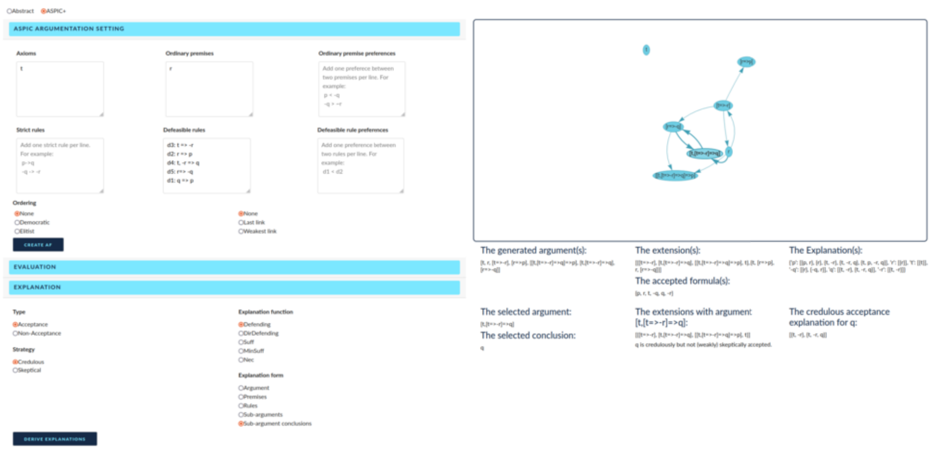
Introduction. Deriving extensions and conclusions from argumentation settings is an essential part of computational argumentation. Moreover, in recent years the interest in argumentation-based explainable artificial intelligence has increased considerably [1]. Since the derivation of conclusions and explanations tends to become intractable when the number of arguments and attacks (in the abstract setting [2]) or the size of the knowledge base and the set of rules (in the structured setting, e.g., [3]) increases, it is useful to have a computational tool that does this for us. To this end, we introduce `PyArg`, which, in addition to being a solver, can also derive explanations.

The Demonstration. We introduce `PyArg` [4], a solver designed for researchers and students who are used to work with Python. The package provides various implementations of formalisms and algorithms in both abstract argumentation and ASPIC$^+$ and comes equipped with an integrated, interactive visualization.

- Selection between abstract argumentation [2] and ASPIC$^+$ [3]. In the abstract setting, users can provide arguments and the attacks between them, in the ASPIC$^+$ setting users can provide axioms, ordinary premises with their preferences, strict rules, defeasible rules with their preferences and a choice in how to derive an ordering from these preferences.
- Evaluation based on a large variety of extension-based semantics [5]. The admissible, complete, grounded, preferred, ideal, stable, semi-stable and eager semantics are available as well as a credulous and skeptical strategy.
- Explanations for (non-)accepted arguments and formulas (in the case of an ASPIC$^+$-setting), based on the explanations from [6,7]. There are functions based on the notion of defense as well as based on necessity and sufficiency.

[1] Author order alphabetical.

`PyArg`, the code and a link to the browser app, is available open source on `https://git.science.uu.nl/D.Odekerken/py_arg`. We hope that it will turn into a community project where `PyArg` becomes a complete solver for many argumentation formalisms, which can be used for teaching and research purposes and that is easily extendable for anyone interested to implement their own ideas.



Abstract [✓ ASPIC+]

ASPIC ARGUMENTATION SETTING

Axioms: t

Ordinary premises: r

Ordinary premise preferences: Add one preferece between two premises per line. For example: p < -q, -q > -r

Strict rules: Add one strict rule per line. For example: p->q, -q -> -r

Defeasible rules:
d3: t => -r
d2: r => p
d4: t, -r => q
d5: r=> -q
d1: q => p

Defeasible rule preferences: Add one preference between two rules per line. For example: d1 < d2

Ordering: [✓ None] Democratic Elitist — [✓ None] Last link Weakest link

EVALUATION

EXPLANATION

Type: [✓ Acceptance] Non-Acceptance

Strategy: [✓ Credulous] Skeptical

Explanation function: [✓ Defending] DirDefending Suff MinSuff Nec

Explanation form: Argument Premises Rules Sub-arguments [✓ Sub-argument conclusions]

The generated argument(s):
[t, r, [t=>-r], [r=>p], [[t,[t=>-r]=>q]=>p], [t,[t=>-r]=>q], [r=>-q]]

The extension(s):
[[[t=>-r], [t,[t=>-r]=>q], [[t,[t=>-r]=>q]=>p], t], [t, [r=>p], r, [r=>-q]]]

The accepted formula(s):
[p, r, t, -q, q, -r]

The Explanation(s):
{'p': [[p, r], [r], [t, -r], [t, -r, q], [t, p, -r, q]], 'r': [[r]], 't': [[t]], '-q': [[r], [-q, r]], 'q': [[t, -r], [t, -r, q]], '-r': [[t, -r]]}

The selected argument:
[t,[t=>-r]=>q]

The selected conclusion:
q

The extensions with argument [t,[t=>-r]=>q]:
[[[t=>-r], [t,[t=>-r]=>q], [[t,[t=>-r]=>q]=>p], t]]
q is credulously but not (weakly) skeptically accepted.

The credulous acceptance explanation for q:
[[t, -r], [t, -r, q]]

Figure 1. Screenshot of `PyArg`, in the ASPIC$^+$ setting, based on [6, Example 3].

Future Work. We intend to extend `PyArg` by implementing additional argumentation formalisms, semantics and explanation functions as well as by introducing dynamic settings. In particular, we will implement algorithms for stability and relevance for incomplete argumentation frameworks in an upcoming release [8].

References

[1] Čyras K, Rago A, Albini E, Baroni P, Toni F. Argumentative XAI: A Survey. In: Proceedings of IJCAI'21. ijcai.org; 2021. p. 4392-9. doi:10.24963/ijcai.2021/600.

[2] Dung PM. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence. 1995;77(2):321-57. doi:10.1016/0004-3702(94)00041-X.

[3] Prakken H. An abstract framework for argumentation with structured arguments. Argument & Computation. 2010;1(2):93-124. doi:10.1080/19462160903564592.

[4] Odekerken D, Borg A. PyArg; 2022. Available from: `https://git.science.uu.nl/D.Odekerken/py_arg`.

[5] Baroni P, Caminada M, Giacomin M. Abstract Argumentation Frameworks and Their Semantics. In: Handbook of Formal Argumentation. College Publications; 2018. p. 159-236.

[6] Borg A, Bex F. A Basic Framework for Explanations in Argumentation. IEEE Intelligent Systems. 2021;36(2):25-35. doi:doi: 10.1109/MIS.2021.3053102.

[7] Borg A, Bex F. Necessary and Sufficient Explanations for Argumentation-Based Conclusions. In: Proceedings of ECSQARU'21. Springer; 2021. p. 45-58. doi:10.1007/978-3-030-86772-0_4.

[8] Odekerken D, Borg A, Bex F. Stability and Relevance in Incomplete Argumentation Frameworks. In: Proceedings of COMMA'22; 2022. Forthcoming.