Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-26)
Early Career Spotlight Track

Towards Reasonable AI: Foundations for Abstraction and Generalized Reasoning

Zeynep G. Saribatur
Institute of Logic and Computation, TU Wien, Austria
zeynep.saribatur@tuwien.ac.at

Abstract

Human reasoning relies on abstraction and generalization, in order to make decisions flexible under changing conditions while ignoring irrelevant details and focusing on the essence. Developing AI systems with such abilities, while ensuring transparency and explainability on the reasoning behind the made decision, remains a central challenge. Symbolic AI provides transparent knowledge representations and formal reasoning guarantees, yet lacks principled mechanisms for abstracting away irrelevant details while preserving the information required for reasoning, explainability, and generalization. This paper presents an overview of my research towards addressing this challenge by developing formal and computational methods for abstraction in Answer Set Programming, one of the core formalisms in symbolic AI, and related logic-based frameworks. The contributions span from foundational abstraction techniques that simplify reasoning representations while preserving essential solution properties, to investigations on how such abstractions can both improve computational reasoning and support human understanding of AI decision processes.

1 Introduction

Abstracting over and forgetting (ir)relevant details are key cognitive mechanisms that humans unwittingly employ when tackling complex problems. They enable flexible decision-making under changing conditions, in particular through the ability to generalize to previously unseen situations. In cognitive science, humans' ability of generalization is considered to involve concept learning and abstraction over similarities [Harnad, 2017; Bruner, 1986]. While humans naturally handle such capabilities, developing them within AI systems has proven a challenging problem. The goal is generally to construct from a given set of problem instances a model that can be used for solving new instances, or to derive a generalized solution to be used in similar problem instances [Yuxiao *et al.*, 2011; Jiménez *et al.*, 2019; Cropper *et al.*, 2022]. Modern machine learning methods that rely on statistical models excel at complex computer vision and natural language processing tasks, while the learned models provide little-to-no insights on their generalization ability. In addition to the lack of transparency, there are generally no formal guarantees on the learned generalizations, as witnessed, e.g., by hallucinations of large language models [Ji *et al.*, 2023], or by simplicity bias of neural networks that ignore meaningful features [Shah *et al.*, 2020]. These limitations have renewed interest in combining statistical learning with symbolic reasoning. While learning is effective at acquiring representations from data, symbolic methods provide knowledge representation and formal reasoning capabilities.

Symbolic AI refers to methods that explicitly represent knowledge and solve problems through logical reasoning. Besides providing transparent and inspectable representations, symbolic approaches offer formal semantics, making it possible to establish correctness guarantees about the reasoning process and the computed solutions. However, reasoning over increasingly large knowledge bases remains computationally demanding, and understanding the resulting reasoning processes or explanations can itself become difficult due to their size and complexity. Thus, equipping symbolic AI with abstraction mechanisms is essential for simplifying reasoning, understanding complex decision processes, and enabling generalized reasoning across related problems.

My research investigates these questions in the setting of Answer Set Programming (ASP), a declarative logic programming paradigm for knowledge representation and reasoning. In ASP, a problem is represented by a set of rules of the form $H \leftarrow B$ in a first-order language, collectively called a program, where intuitively, the head H must hold whenever the body B is satisfied. The models of such programs, called answer sets [Gelfond and Lifschitz, 1991], correspond to the solutions of the encoded problem. ASP encodings often follow a "guess-and-check" methodology, where solution candidates are generated and invalid candidates ruled out through constraints. A distinguishing feature of ASP is its nonmonotonic semantics, which naturally supports reasoning with incomplete information, defaults, and exceptions. ASP combines expressive knowledge representation with precise formal semantics, making it an ideal setting for studying abstraction and generalized reasoning.

The central idea underlying my research is that abstraction provides the formal machinery for simplifying reasoning while preserving the semantic structure required for explain-

ability and generalized reasoning. This paper presents the progression of this research in ASP, from over-approximation techniques that simplify reasoning by omitting and clustering details, to dependency-preserving abstractions that characterize irrelevancy in reasoning, enabling empirically validated improvements in explainability, and ultimately towards a formal foundation for generalized reasoning. I view these abstraction mechanisms as a key ingredient for building reasonable AI systems capable of logical reasoning and generalization, with cognitively aligned explanations of their decisions.

2 Foundations of Abstraction in ASP

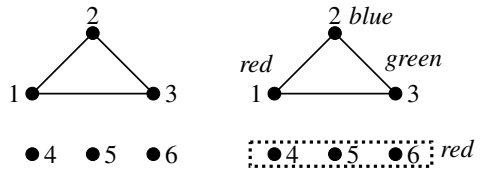
The central idea behind abstraction is to intentionally discard details while preserving the essential structure for reasoning. In the context of ASP, we introduced abstraction as a means to simplify a program into a coarser program while preserving the original solutions, thus achieving *over-approximation*. Formally, a program P' over $\mathcal{U}'$ is called an *abstraction* of a program P over $\mathcal{U}$, if there exists a mapping $m : \mathcal{U} \to \mathcal{U}'$ such that for each answer set I of P, $I' = \{m(l) \mid l \in I\}$ is an answer set of P', i.e., $m(AS(P)) \subseteq AS(P')$. This ensures that every answer set of the original program is represented at the abstract level, although additional spurious answer sets may be introduced. Such abstractions reduce the size of the search space and make it possible to reason about a simplified representation of the problem.

Approaches for two kinds of abstraction mappings are introduced. The first, *abstraction by omission* [Saribatur and Eiter, 2021], omits selected atoms from the vocabulary (by mapping the atoms to $\top$) and focuses reasoning on the remaining information. The second, *domain abstraction* [Saribatur *et al.*, 2021], clusters domain elements into abstract representatives and reasons over the resulting abstract domain.

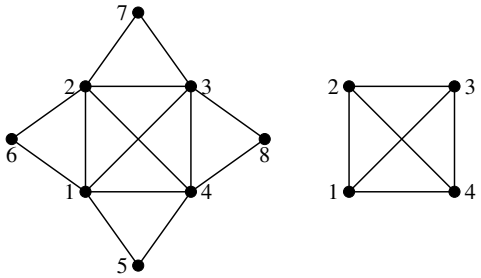
Next example illustrates how both forms of abstraction produce a simpler problem representation that captures essential aspects of the original problem.

Example 1. *Figure 1a shows a graph instance that has 162 possible 3-colorings of the nodes, with 6 possible colorings of nodes 1-2-3. Meanwhile, the graph in Figure 1b is not 3-colorable due to the clique 1-2-3-4. We consider two possible abstractions over the nodes in the problem. For Figure 1b, omitting all the nodes except 1-2-3-4 preserves the uncolorability. As for Figure 1a clustering the nodes 4-5-6 into one abstract node results in 3-coloring the nodes 1-2-3 and assigning some colors to the cluster node without focusing on the details of the colors of each individual node.*

Depending on the abstraction mapping, we provide syntactic operators to construct an abstract ASP program that ensures over-approximation. As spurious answer sets can be introduced, one may need to go over all abstract answer sets until one that can be instantiated to an original answer set is found. If the original program has no answer sets, all encountered abstract answer sets will be spurious. Eliminating the spurious answer sets is possible by refining the abstraction. However, checking the concreteness of an abstract answer set returns unsatisfiability, if it is spurious, without any reason on why this is the case. In order to obtain hints on spuriousness, we make use of ASP debugging approaches [Brain *et al.*, 2007; Oetsch *et al.*, 2010] to be able to debug checking the existence of an original answer set that can be mapped to the abstract answer set. The refinement decisions are then made using the debugging hints obtained from the checking. We introduce an abstraction-&-refinement methodology (inspired from CEGAR [Clarke *et al.*, 2003]) that starts with an initial abstraction and refines it repeatedly using hints that are obtained from checking the abstract answer sets, until a concrete solution (or unsatisfiability) is encountered. We have implemented the framework in tools that are able to automatically achieve an abstraction mapping that creates an abstract program where either a concrete answer set is encountered, or unsatisfiability is achieved.



(a) 3-coloring of a graph, where nodes with no edges are clustered into one cluster node (right)



(b) Non-3-colorable graph, where nodes irrelevant to the non-colorability are omitted (right)

Figure 1: Illustration of abstraction through omission and clustering.

These contributions established formal foundations for abstraction in ASP and demonstrated its usefulness for both reasoning and analysis. The resulting ideas influenced later work on abstraction in argumentation [Saribatur *et al.*, 2020; Saribatur and Wallner, 2021; Apostolakis *et al.*, 2024] and, more broadly, motivated further research in abstraction as a mechanism for explainability and generalized reasoning.

3 Dependency-Preserving Abstractions

The abstraction frameworks discussed in the previous section simplify reasoning through over-approximation. However, supporting generalized reasoning requires stronger guarantees. Abstractions must preserve not only the existence of solutions, but also the semantic dependencies that determine how solutions change across related problem instances.

The subsequent research thus focuses on theoretical foundations for abstractions preserving stronger aspects of a program's semantics and investigate abstractions as a way to generalize within programs [Saribatur and Woltran, 2023; Saribatur *et al.*, 2024]. The main idea is to represent a program P over a given language $\mathcal{U}$ with a program Q over a language $\mathcal{U}'$ of smaller size (compared to $\mathcal{U}$), in a way that pre-

serves the semantics of P in Q. This is achieved by mapping atoms from P to atoms in Q in such a way that the answer sets of the original program P and the resulting abstraction Q correspond in the spirit of uniform equivalence [Eiter and Fink, 2003], independently of the facts, i.e., the instance data, added to the programs. Formally, we define Q to be a uniform m-abstraction of P if for any set F of facts over $\mathcal{U}$, we have $m(AS(P \cup F)) = AS(Q \cup m(F))$. Intuitively, this notion allows an abstract program to reason about classes of similar instances, providing the basis for generalized reasoning.

Next example illustrates the motivation.

Example 2. *Let us consider the following program P.*

$$\begin{aligned} registerEarly &\leftarrow not\ registerLate \\ registerLate &\leftarrow not\ registerEarly \\ registered &\leftarrow registerEarly \\ registered &\leftarrow registerLate \\ reachBremen &\leftarrow takePlane \\ reachBremen &\leftarrow takeTrain \\ attendIJCAI &\leftarrow reachBremen, registered \end{aligned}$$

We register to IJCAI either early or late, and reach Bremen by plane or train; once there, being registered allows us to attend IJCAI. Now for $P \cup \{takeTrain\}$, the answer sets would be $\{register Early, registered, takeTrain, reachBremen, attendIJCAI\}$ and $\{registerLate, registered, takeTrain, reachBremen, attendIJCAI\}$. Since $registered$ holds in both cases, the registration details are not of key importance to attend IJCAI, and can be omitted. Furthermore, in P, since both taking the plane and the train allow us to reach Bremen, the details of how the travel is made could be clustered, e.g, by mapping $takeTrain$ and $takePlane$ to $travel$. Now consider the program Q below.

$$\begin{aligned} reachBremen &\leftarrow travel \\ attendIJCAI &\leftarrow reachBremen \end{aligned}$$

We can see that Q preserves all the dependencies between the abstracted atoms and the rest of the atoms. In other words, if P is extended with further facts F, e.g., $\{takeTrain\}$, then Q together with the abstracted facts $m(F)$, e.g., $\{travel\}$, would still yield corresponding abstracted answer sets.

This illustrates the key idea behind generalized reasoning: once irrelevant details for reasoning have been abstracted away, the same abstract representation remains applicable across multiple related problem instances.

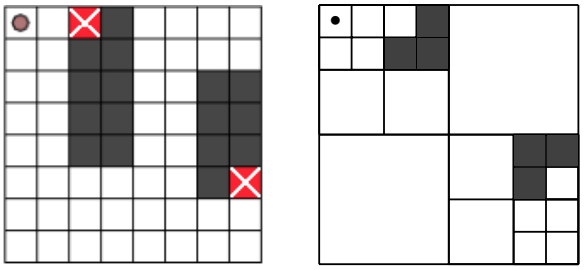
Initial focus has been on omission of atoms, where the resulting abstractions are referred to as *simplifications* [Saribatur and Woltran, 2023], making them closely related to the widely studied notion of forgetting [Eiter and Kern-Isberner, 2019; Gonçalves *et al.*, 2023]. Forgetting constructs, from a program P, a program over a reduced vocabulary that forgets a set of atoms while preserving its semantics in the presence of any additional set of rules/facts not containing the atoms to be forgotten. Building on this connection, we showed that allowing the added facts or rules to range over the full vocabulary captures the irrelevancy of atoms, which can be disregarded both from the original program and from the context while preserving the semantics of the original program. The same observation holds when considering clustering mappings, allowing semantically indistinguishable details to be abstracted into a single concept [Saribatur *et al.*, 2024]. As not every program is abstractable, we provided necessary and sufficient conditions for abstractability together with model-theoretic characterizations of the resulting abstractions. Importantly, whenever possible, these abstractions can be obtained from the original program by simple syntactic operators that project away omitted atoms or map atoms to their clusters.

Later, we investigated a relativized version of these abstraction notions, where added facts/rules are restricted to a subset of the vocabulary [Saribatur and Woltran, 2024; Knorr *et al.*, 2026]. Besides allowing abstractions for programs that were previously not abstractable, this relaxation bridges the gap between forgetting [Gonçalves *et al.*, 2020; Gonçalves *et al.*, 2019] and (relativized) strong/uniform equivalence [Lifschitz *et al.*, 2001; Eiter *et al.*, 2007] , providing a unified view of these notions in the literature.

More recently, we examined the computational complexity of checking and the existence of abstractions by clustering, and investigated the role of symmetries in determining abstractions [Saribatur *et al.*, 2026a]. We show that symmetry and abstraction notions are orthogonal, due to the difference of *interchangeability* and *indistinguishability*. The idea of symmetry is to detect interchangeable objects, to reduce the search space, such as pigeons and holes in the pigeonhole problem. However, identifying these objects is necessary to ensure that the constraints are still satisfied. Abstraction, on the other hand, is about losing the identification of the objects as they do not matter in finding a solution, such as having checkpoints to pass through in a package delivery where the choice of which middle point the truck passes through is not important. Under suitable syntactic restrictions, syntactically symmetric atoms become semantically indistinguishable, yielding uniform abstractions.

4 Abstraction for Explainability

In the cognitive science of explanation, it is well-established that humans prefer simple, contrastive explanations over exhaustive traces for the made decision [Miller, 2019; Lombrozo, 2007]. Just as image classification explanations use saliency maps to highlight relevant pixels while treating the rest as irrelevant [Ribeiro *et al.*, 2016], symbolic representations must distinguish between essential logical pivots and distracting details adhering with Grice's Maxim of Quantity [Grice, 1975]. Existing explanation approaches in Answer Set Programming (ASP) [Eiter *et al.*, 2026] typically justify why an atom is present or absent in an answer set by tracing the relevant rules and derivations. As domains become more complex, however, such explanations often suffer from information density, containing details that are technically correct but cognitively unhelpful. The challenge is therefore not only to generate explanations, but to identify which information is



(a) Human explanation for unsolvability marking the two dead-ends

(b) Abstraction obtained by the tool that focuses on the region that preserves unsolvability

Figure 2: Explanatory abstractions on an unsolvable Visitall instance, where the agent on the top-left corner needs to visit all the cells by visiting each cell once.

relevant for human understanding.

A first indication that abstraction can support this goal comes from problems with multi-dimensional structures such as grid-cell domains, where human observers naturally focus on the part of the grid relevant for the problem. For these structures, we developed a hierarchical form of domain abstraction that automatically adjusts the level of granularity toward the relevant parts of the instance [Eiter *et al.*, 2019; Saribatur *et al.*, 2021]. Figure 2 illustrates the resulting effect on an unsolvable Visitall problem. The abstraction produced by the tool preserves unsolvability while zooming in on the region that is responsible (Figure 2b). The resulting view resemble the way humans focus on certain parts of the grid when explaining unsolvability (Figure 2a).

This observation motivated a line of work that empirically validates the impact of abstraction over explanations on human cognition [Saribatur *et al.*, 2026b]. The abstraction notions introduced in Section 3 yield abstracted programs, in which unnecessary details are omitted and semantically similar concepts are clustered, that can be used to obtain traces for the decision. As an illustration, Figure 3 shows the explanations for $attendIJCAI$ from Example 2 obtained with the `xclingo` system [Cabalar and Muñiz, 2024]. Default explanation (Figure 3a) is obtained from $P \cup \{takeTrain\}$ while the abstract explanation (Figure 3b) is obtained from $Q \cup \{travel\}$.[1] The question then becomes: Do these theoretically simpler traces actually result in measurable improvements in human understanding and reductions in cognitive effort?

We conducted an experiment that utilized concept-learning tasks in which participants performed a binary classification task after being presented with explanations. Understanding and effort is measured by accuracy and answer times, respectively. The empirical study was based on a complete between-subject design, where participants were randomly assigned to one of the four groups: default, cluster, removal, and cluster and removal. and the data between these groups were compared to calculate the effect on the dependent variables.

[1]For the empiricial study, the clustered atom was presented explicitly as $takePlaneOrTrain$ rather than $travel$ to avoid introducing an additional abstraction layer for participants.

```
|__attendIJCAI
| |__registered
| | |__registerEarly
| |__reachBremen
| | |__takeTrain
```

(a) Default

```
|__attendIJCAI
| |__reachBremen
| | |__travel
```

(b) Abstracted

Figure 3: `xclingo` explanations for $attendIJCAI$ from Ex. 2

The findings confirm that abstractions help with understanding of the explanations and reduce the cognitive effort, though the type of abstraction influences the nature of the improvement. When explanations were abstracted into clusters, participants made significantly fewer errors. The clustered information in the explanations likely provided structural organization that allowed to form more clear models of the classification rules, leading to fewer errors. On the other hand, removal of the irrelevant details in explanations results in reaching decisions faster. As they were not displayed in the explanations, this helped in the participants to ignore the undecisive attributes when making the classification decisions.

We now have empirical evidence for the double benefit of symbolic abstraction in explanations: structural organization via clustering facilitates understanding, while the removal of irrelevant details reduces cognitive effort. This shows that there is more than just obtaining simpler explanations, as different abstraction operations serve distinct cognitive effects, and also highlights the importance of tailoring symbolic explanations to cognitive processes.

5 Conclusion and Outlook

This research establishes abstraction as a fundamental principle for symbolic AI, providing the theoretical foundations to bridge complex logical reasoning with human-aligned generalization and explainability.

Building on these foundations, ongoing work extends abstraction towards reasoning over specific sets of problem instances [Saribatur *et al.*, 2026c]. Lifting these abstraction notions to temporal theories [Aguado *et al.*, 2013; Aguado *et al.*, 2023], will then allow to represent generalized reasoning in dynamic domains, while also capturing generalized planning. The complementary direction investigates how to learn good abstractions by detecting the underlying patterns [Law *et al.*, 2020; Frances *et al.*, 2021] guided by the developed theoretical results. Another challenge is to automatically derive human-understandable concepts for the learned abstractions, bridging formally computed abstractions with cognitively meaningful symbolic representations.

My long-term vision is to establish cognitively aligned abstraction and generalized reasoning [Ilievski *et al.*, 2025] as core capabilities of symbolic AI systems. Achieving this vision requires developing formal frameworks that can derive generalized solutions across related problem instances, while ensuring that the resulting explanations remain cognitively meaningful for humans. Realizing this vision ultimately relies on close collaboration with social scientists to align formal models of generalization with human reasoning processes.

Acknowledgments

This research was funded in whole or in part by the Austrian Science Fund (FWF) grants 10.55776/T1315 and 10.55776/COE12, and the Vienna Science and Technology Fund (WWTF) grant 10.47379/ICT25044.

References

[Aguado *et al.*, 2013] Felicidad Aguado, Pedro Cabalar, Martín Diéguez, Gilberto Pérez, and Concepción Vidal. Temporal equilibrium logic: a survey. *Journal of Applied Non-Classical Logics*, 23(1-2):2–24, 2013.

[Aguado *et al.*, 2023] Felicidad Aguado, Pedro Cabalar, Martín Diéguez, Gilberto Pérez, Torsten Schaub, Anna Schuhmann, and Concepción Vidal. Linear-time temporal answer set programming. *Theory and Practice of Logic Programming*, 23(1):2–56, 2023.

[Apostolakis *et al.*, 2024] Iosif Apostolakis, Zeynep G. Saribatur, and Johannes P. Wallner. Abstraction in assumption-based argumentation. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024*, 2024.

[Brain *et al.*, 2007] Martin Brain, Martin Gebser, Jörg Pührer, Torsten Schaub, Hans Tompits, and Stefan Woltran. Debugging ASP programs by means of ASP. In *Logic Programming and Nonmonotonic Reasoning, 9th International Conference, LPNMR 2007, Tempe, AZ, USA, May 15-17, 2007, Proceedings*, Lecture Notes in Computer Science, pages 31–43. Springer, 2007.

[Bruner, 1986] Jerome Bruner. *A study of thinking*. Routledge, 1986.

[Cabalar and Muñiz, 2024] Pedro Cabalar and Brais Muñiz. Model explanation via support graphs. *Theory and Practice of Logic Programming*, 24(6):1109–1122, 2024.

[Clarke *et al.*, 2003] Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM*, 50(5):752–794, 2003.

[Cropper *et al.*, 2022] Andrew Cropper, Sebastijan Dumančić, Richard Evans, and Stephen H Muggleton. Inductive logic programming at 30. *Machine Learning*, 111(1):147–172, 2022.

[Eiter and Fink, 2003] Thomas Eiter and Michael Fink. Uniform equivalence of logic programs under the stable model semantics. In *Logic Programming, 19th International Conference, ICLP 2003, Mumbai, India, December 9-13, 2003, Proceedings*, Lecture Notes in Computer Science, pages 224–238. Springer, 2003.

[Eiter and Kern-Isberner, 2019] Thomas Eiter and Gabriele Kern-Isberner. A brief survey on forgetting from a knowledge representation and reasoning perspective. *KI – Künstliche Intelligenz*, 33(1):9–33, 2019.

[Eiter *et al.*, 2007] Thomas Eiter, Michael Fink, and Stefan Woltran. Semantical characterizations and complexity of equivalences in answer set programming. *ACM Transactions on Computational Logic (TOCL)*, 8(3):17, 2007.

[Eiter *et al.*, 2019] Thomas Eiter, Zeynep G. Saribatur, and Peter Schüller. Abstraction for zooming-in to unsolvability reasons of grid-cell problems. In *Proc. of the IJCAI 2019 Workshop on Explainable Artificial Intelligence (XAI)*, 2019.

[Eiter *et al.*, 2026] Thomas Eiter, Tobias Geibinger, and Zeynep G. Saribatur. An XAI view on explainable ASP: Methods, systems, and perspectives. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2026, (To appear)*, 2026.

[Frances *et al.*, 2021] Guillem Frances, Blai Bonet, and Hector Geffner. Learning general planning policies from small examples without supervision. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Event, February 2-9, 2021*, pages 11801–11808. AAAI Press, 2021.

[Gelfond and Lifschitz, 1991] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3):365–385, 1991.

[Gonçalves *et al.*, 2019] Ricardo Gonçalves, Tomi Janhunen, Matthias Knorr, Joao Leite, and Stefan Woltran. Forgetting in modular answer set programming. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, volume 33, pages 2843–2850, 2019.

[Gonçalves *et al.*, 2020] Ricardo Gonçalves, Matthias Knorr, João Leite, and Stefan Woltran. On the limits of forgetting in Answer Set Programming. *Artificial Intelligence*, 286:103–307, 2020.

[Gonçalves *et al.*, 2023] Ricardo Gonçalves, Matthias Knorr, and João Leite. Forgetting in answer set programming - A survey. *Theory and Practice of Logic Programming*, 23(1):111–156, 2023.

[Grice, 1975] Herbert Paul Grice. Logic and conversation. *Syntax and semantics*, 3:43–58, 1975.

[Harnad, 2017] Stevan Harnad. To cognize is to categorize: cognition is categorization. In *Handbook of categorization in cognitive science*, pages 21–54. Elsevier, 2017.

[Ilievski *et al.*, 2025] Filip Ilievski, Barbara Hammer, Frank van Harmelen, Benjamin Paassen, Sascha Saralajew, Ute Schmid, Michael Biehl, Marianna Bolognesi, Xin Luna Dong, Kiril Gashteovski, Pascal Hitzler, Giuseppe Marra, Pasquale Minervini, Martin Mundt, Axel-Cyrille Ngonga Ngomo, Alessandro Oltramari, Gabriella Pasi, Zeynep G. Saribatur, Luciano Serafini, John Shawe-Taylor, Vered Shwartz, Gabriella Skitalinskaya, Clemens Stachl, Gido M. van de Ven, and Thomas Villmann. Aligning generalization between humans and machines. *Nature Machine Intelligence*, 7(9):1378–1389, 2025.

[Ji *et al.*, 2023] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, et al. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

[Jiménez *et al.*, 2019] Sergio Jiménez, Javier Segovia-Aguas, and Anders Jonsson. A review of generalized planning. *The Knowledge Engineering Review*, 34:e5, 2019.

[Knorr *et al.*, 2026] Matthias Knorr, Zeynep G. Saribatur, and Ricardo Gonçalves. Beyond uniform: To boldly abstract what has not been abstracted before. In *Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning, KR 2026, (To appear)*, 2026.

[Law *et al.*, 2020] Mark Law, Alessandra Russo, and Krysia Broda. The ILASP system for inductive learning of answer set programs. *CoRR*, abs/2005.00904, 2020.

[Lifschitz *et al.*, 2001] Vladimir Lifschitz, David Pearce, and Agustín Valverde. Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2(4):526–541, October 2001.

[Lombrozo, 2007] Tania Lombrozo. Simplicity and probability in causal explanation. *Cognitive psychology*, 55(3):232–257, 2007.

[Miller, 2019] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[Oetsch *et al.*, 2010] Johannes Oetsch, Jörg Pührer, and Hans Tompits. Catching the ouroboros: On debugging non-ground answer-set programs. *Theory and Practice of Logic Prog.*, 10(4-6):513–529, 2010.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. ACM, 2016.

[Saribatur and Eiter, 2021] Zeynep G. Saribatur and Thomas Eiter. Omission-based abstraction for answer set programs. *Theory and Practice of Logic Programming*, 21(2):145–195, 2021.

[Saribatur and Wallner, 2021] Zeynep G. Saribatur and Johannes P. Wallner. Existential abstraction on argumentation frameworks via clustering. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021*, pages 549–559, 2021.

[Saribatur and Woltran, 2023] Zeynep G. Saribatur and Stefan Woltran. Foundations for projecting away the irrelevant in ASP programs. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece, September 2-8, 2023*, pages 614–624. IJCAI Organization, 2023.

[Saribatur and Woltran, 2024] Zeynep G. Saribatur and Stefan Woltran. A unified view on forgetting and strong equivalence notions in answer set programming. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, February 20-27, 2024, Vancouver, Canada*, pages 10687–10695. AAAI Press, 2024.

[Saribatur *et al.*, 2020] Zeynep G. Saribatur, Johannes P. Wallner, and Stefan Woltran. Explaining non-acceptability in abstract argumentation. In *Proceedings of the 24th European Conference on Artificial Intelligence, ECAI 2020, 29 August-8 September 2020, Santiago de Compostela, Spain*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 881–888. IOS Press, 2020.

[Saribatur *et al.*, 2021] Zeynep G. Saribatur, Thomas Eiter, and Peter Schüller. Abstraction for non-ground answer set programs. *Artificial Intelligence*, 300:103563, 2021.

[Saribatur *et al.*, 2024] Zeynep G. Saribatur, Matthias Knorr, Ricardo Gonçalves, and João Leite. On abstracting over the irrelevant in answer set programming. In Pierre Marquis, Magdalena Ortiz, and Maurice Pagnucco, editors, *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024*, 2024.

[Saribatur *et al.*, 2026a] Zeynep G. Saribatur, Markus Hecher, and Johannes K. Fichte. Abstracting the indistinguishable in ASP. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2026, (To appear)*, 2026.

[Saribatur *et al.*, 2026b] Zeynep G. Saribatur, Johannes Langer, and Ute Schmid. The dual role of abstracting over the irrelevant in symbolic explanations: Cognitive effort vs. understanding. In *Proceedings of the 48th Annual Meeting of the Cognitive Science Society, CogSci 2026, (To appear)*, 2026.

[Saribatur *et al.*, 2026c] Zeynep G. Saribatur, Kaan Unalan, and Stefan Woltran. Model-theoretic characterization of programs under specific inputs. In *Proceedings of the 18th International Conference on Logic Programming and Nonmonotonic Reasoning, LPNMR 2026, (To appear)*, 2026.

[Shah *et al.*, 2020] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[Yuxiao *et al.*, 2011] Hu Yuxiao, Giuseppe DE GIACOMO, et al. Generalized planning: Synthesizing plans that work for multiple environments. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 918–923. IJCAI/AAAI, 2011.