Ontology-Mediated Neurosymbolic Constraint Acquisition from Multiple Stakeholders

Stefan Bischof, Juliana Kainz and Danilo Valerio

Siemens AG Österreich, Vienna, Austria

Abstract

Neurosymbolic research typically assumes a pre-existing symbolic specification, leaving the upstream challenge of acquiring and formalizing requirements and constraints largely unaddressed. We present an architecture that fills this gap by using an OWL configuration ontology to mediate between neural constraint sources and downstream consumers. In this framework, LLM assistants elicit soft stakeholder preferences, while hardware specifications define hard physical and engineering limits. The ontology unifies these heterogeneous inputs, leverages description logic to identify unsatisfiability, and generates symbolic explanations that enable LLMs to interactively renegotiate terms with users. Any remaining conflicts are resolved downstream via priority-based relaxation. We illustrate our approach on a microgrid use case from the FLEXI project and argue its generalizability to multi-stakeholder domains where constraint acquisition is distributed across human and automated sources of unequal authority.

Keywords

neurosymbolic AI, constraint acquisition, configuration ontology, constraint reconciliation

1. Introduction

Neurosymbolic systems can be categorized by how they couple neural components with symbolic specifications [1]. Most existing literature focuses heavily on the downstream problem: ensuring that neural outputs comply with a given specification. Typically a neural component proposes a candidate output (an answer, a structured string, a proof) and a symbolic layer checks it against logical constraints, rejecting or repairing violations. This paradigm has driven significant advances in verification, constrained decoding, and symbolic verifiers that check neural outputs [2, 3, 4, 5].

A much less explored challenge is the upstream problem of how these symbolic specifications are actually constructed. In many real settings the constraint set is neither fixed nor given by a single authority. It is acquired, assembled from heterogeneous sources whose constraints may be mutually inconsistent. Physical and safety limits come from hardware specifications and are non-negotiable. Other constraints are preferences: what a particular user wants, expressed in natural language and elicited through dialogue. Preferences originate from multiple stakeholder groups whose interests may conflict and whose claims do not carry equal weight; for example, when operating an energy grid, an operator's grid-stability requirement outranks an individual user's preference. Before the specification can be used, these sources must be reconciled into a single, consistent constraint set, and the conflicts must be detected and resolved.

In this position paper, we argue that a configuration ontology is an ideal mediating layer for this upstream acquisition challenge, and outline the architecture it enables. Our work builds on CONTO, a configuration ontology based on the Web Ontology Language (OWL), developed at Siemens for vendor-independent product configuration [6]. CONTO already provides a standards-based representation of components, features, and constraints, together with Description-Logic consistency checking. However it is limited to modeling *hard constraints only*: every constraint must be satisfied, and there is no notion of preference, priority, or optimization objective. Overcoming this limitation is what multi-stakeholder

Third Workshop on Knowledge Graphs and Neurosymbolic AI (KG-NeSy 2026), in ISWC 2026 Workshops Joint Proceedings, October 25–26, 2026, Bari, Italy

bischof.stefan@siemens.com (S. Bischof); juliana.kainz@siemens.com (J. Kainz); danilo.valerio@siemens.com (D. Valerio)
0000-0001-9521-8907 (S. Bischof); 0000-0002-1293-0315 (J. Kainz); 0000-0003-4104-3974 (D. Valerio)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-05

CEUR-WS.org/Vol-4313/kgnesy-paper6.pdf

constraint acquisition demands, and addressing it forms the core of our approach. Concretely, we make the following contributions:

- We frame **multi-stakeholder constraint acquisition** as a distinct neurosymbolic problem, the necessary complement to the well-studied downstream validation task.
- We propose an **ontology-mediated architecture** that integrates LLM assistants and hardware sources into an OWL configuration ontology. By extending this ontology with a hard/soft distinction and priority levels, user preferences become relaxable, weighted constraints alongside hard physical limits (Section 3).
- We describe a **closed neurosymbolic loop** in which description-logic reasoning detects structural conflicts and returns symbolic explanations that drive the assistants' renegotiation with users, and we instantiate the architecture on the microgrid (Section 4).

Throughout, we ground the architecture in the microgrid use case of the FLEXI project,[1] a business park with photovoltaics, stationary batteries, and electric-vehicle (EV) chargers whose charging and storage must be scheduled for sustainability, grid stability, and cost. Its constraints come from sources of unequal authority: EV owners state charging preferences, a charge-point operator sets infrastructure requirements, and hardware specifications fix the hard physical limits. Two requests show what makes reconciliation hard: charging 90 kWh into a battery that holds 60 can never be satisfied at all, whereas a full charge by 16:00 is feasible on its own yet may be unschedulable once many vehicles charge at once.

2. Related Work

We position our approach against three strands of work.

The dominant neurosymbolic pattern in constrained settings uses symbolic knowledge to constrain or verify neural outputs: constrained decoding restricts generation to a formal grammar [2], and symbolic solvers or provers check a proposed solution and feed errors back to the model [4, 5]. These approaches assume the constraint specification is given; our work targets the prior question of how a consistent specification is acquired in the first place. Logic-LM [3] is the closest structural analogue, iterating generation, symbolic checking, and revision, but its feedback corrects the *formalization* of a single given problem, whereas ours reconciles *competing constraints* elicited from multiple stakeholders.

There is a substantial constraint-programming literature on *constraint acquisition*, learning a constraint network from examples of valid and invalid assignments [7]. This includes active methods that acquire constraints through membership queries to a user [8]. This paradigm typically assumes a single implicit authority from which constraints are learned via structured data or query responses. It rarely assumes elicitation through natural-language dialogue, leverages an explicit ontology as the shared representation, or manages multiple stakeholder groups with differing priority.

Recent work combines LLMs with ontologies and knowledge graphs for requirements elicitation and configuration: conversational ontology engineering guides users from informal input to structured requirements [9], and LLMs paired with constraint programming support interactive preference elicitation [10]. Closely related, a configuration copilot integrates an LLM with a constraint solver for interactive product configuration [11]. We build on this line but shift the emphasis from authoring a single product model to reconciling constraints acquired concurrently from several conflicting sources of unequal authority.

3. Approach

We present the architecture in two steps: first the mediating layer itself, the sources that feed it, and the vocabulary extension that makes stakeholder preferences relaxable; then the consistency check this enables and the feedback loop it drives.

[1] https://flexi-cetp.eu/

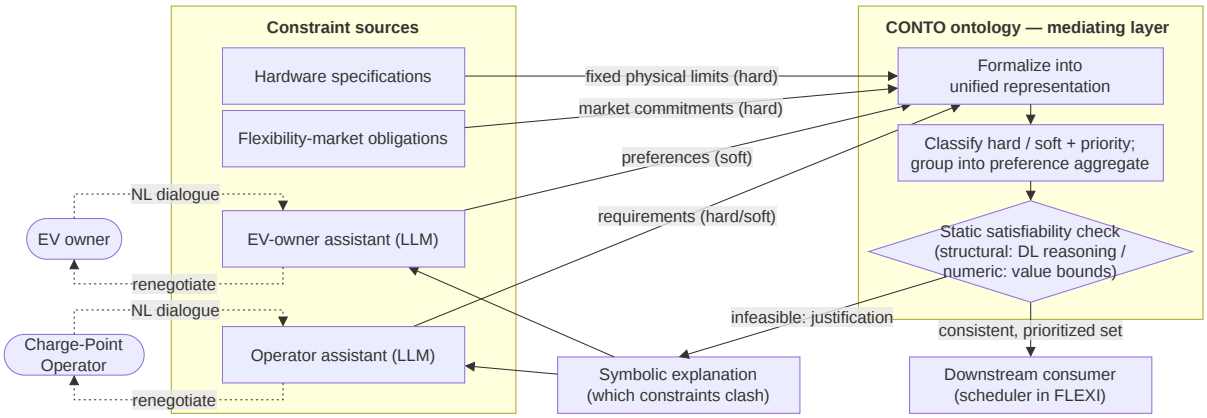


Figure 1: The constraint-acquisition loop, instantiated on the FLEXI microgrid. Constraint sources of unequal authority (left) feed the CONTO mediating layer (right), which this paper addresses; the downstream consumer (a scheduler in FLEXI) is out of scope.

3.1. Architecture Overview

We propose an architecture in which a configuration ontology sits between the sources that supply constraints and the downstream consumer that uses them, shown as the mediating layer of Figure 1. Constraints enter from two kinds of source: LLM-based assistants that elicit preferences from human stakeholders in dialogue, and hardware specifications that contribute fixed physical limits. The ontology formalizes every incoming constraint into a single representation, classifies it as hard or soft and attaches a priority, and checks whether each soft preference can be satisfied at all given the hard limits. A preference that cannot is turned into a symbolic explanation and returned to the originating assistant, which renegotiates with the user. Acquisition is continuous rather than a one-off batch: constraints may arrive or change at any time, so each is checked as it arrives and, whenever the feasible set changes, the scheduler is re-invoked. This paper concerns the acquisition loop; the scheduler is downstream, and we treat it here as a consumer of the acquired constraint set. It is in the acquisition loop that the neural and symbolic components are coupled: the ontology mediates between them, mapping heterogeneous, partially conflicting inputs to a single consistent and prioritized constraint set.

Our mediator builds on CONTO, a suite of OWL ontologies for vendor-independent product configuration [6]. CONTO separates three layers: a generic meta-model (L2) defining the vocabulary of components, features, and constraints; a product line (L1) that defines a specific configurable artifact against L2; and a configuration (L0), a concrete configured instance. It offers two complementary formalizations of a product model. The *Configuration Vocabulary* (ConfigVoc, prefix `cv:`) is an instance-based OWL 2 QL vocabulary in which a constraint is reified as data: a table constraint of rows and cells, or a formula constraint carrying an expression tree. The *Configuration DL* ontology (ConfigDL, prefix `cdl:`) is a class-based formalization in which a constraint *is* an OWL axiom, so a description-logic reasoner enforces it natively through consistency checking.

In both formalizations, however, every constraint is *hard*: it must hold, and there is no notion of a preference that may be relaxed, no priority among constraints, and no optimization objective. For single-authority product configuration this is adequate; for multi-stakeholder acquisition, where user preferences must coexist with, and yield to, physical limits, it is the gap we propose to close. Whether the preference semantics are better added to the instance-based ConfigVoc (as additional data) or to the axiom-based ConfigDL (where relaxable constraints do not map cleanly onto OWL's all-or-nothing axiom semantics) is itself an open question we return to in Section 5; for concreteness we illustrate the extension on ConfigVoc below.

Each stakeholder group is served by its own instance of an LLM-based assistant, configured for that group, which elicits constraints in natural-language dialogue and maps them into the ontology's vocabulary; because the assistants are the interface to human stakeholders, they also carry back the explanations produced downstream (Section 3.2). We treat the elicited preferences as *soft* constraints:

Listing 1: Modeling in the FLEXI microgrid: an existing ConfigVoc hard constraint (`cv:`) and the proposed preference extension (`pref:`) marking a soft, prioritized constraint and a preference aggregate. Formula bodies are abbreviated; ConfigVoc encodes them as expression trees.

```
# Hard constraint (physical limit) -- existing ConfigVoc
flexi:fuseCapacity a cv:FormulaConstraint ;
  cv:usesFeature flexi:sumChargerPower, flexi:fuseCapacity ;
  cv:formula [ ... "sumChargerPower <= fuseCapacity" ... ] .  # tree - abbreviated here

# Soft constraint (user preference) -- proposed extension
flexi:desiredSOC a cv:FormulaConstraint ;
  cv:usesFeature flexi:actualSOC, flexi:targetSOC ;
  cv:formula  [ ... "actualSOC >= targetSOC" ... ] ;          # tree - abbreviated here
  pref:constraintType pref:Soft ;                             # proposed
  pref:priority 1000 .                                        # proposed

flexi:evOwnerPreferences a pref:PreferenceAggregate ;         # proposed
  pref:aggregates flexi:desiredSOC .
```

statements of what a stakeholder wants that should be satisfied where possible but may be relaxed. In contrast, hardware and infrastructure specifications contribute *hard* constraints, fixed physical and safety limits that must always hold; these enter the ontology directly, without elicitation, and are never relaxed.

Preferences as soft constraints. The central element of our approach is a small vocabulary extension that introduces the hard/soft distinction, priorities, and preference aggregates. We propose to classify each constraint as hard or soft, attach a numeric priority (and, optionally, a weight) to soft constraints, and group a stakeholder's soft constraints into a *preference aggregate* that the downstream consumer should try to satisfy. A hard constraint must hold: to *violate* it renders the set *infeasible*. A soft constraint need not; leaving it unsatisfied is not a violation but a *relaxation* that incurs a *violation cost* in proportion to its priority, in the sense of valued and semiring-based constraint-satisfaction problems (CSPs) and of weak constraints in answer set programming (ASP) [12, 13, 14]. Modeling preferences as first-class, weighted, relaxable constraints, rather than as an out-of-band scoring function, keeps them in the same formal representation as the hard limits, so that the same satisfiability check and conflict explanation apply to both.

Listing 1 illustrates the intended modeling in the FLEXI setting, reusing existing ConfigVoc terms (prefix `cv:`) and introducing the proposed preference vocabulary (prefix `pref:`): a hard physical limit (charger power must not exceed fuse capacity), a prioritized soft preference (the vehicle should reach its desired state of charge), and a preference aggregate grouping that stakeholder's soft constraints. The formula bodies are abbreviated for readability; ConfigVoc represents them as expression trees.

3.2. Consistency Checking and the Feedback Loop

Acquiring a usable constraint set involves relaxation at two distinct points, which must be distinguished. The first is the *static satisfiability check* of Figure 1, run as constraints are acquired: for each soft preference it asks whether that preference could be honored *at all* given the hard limits, before any schedule is computed. Because soft constraints are relaxable, such a preference does not make the set inconsistent; it simply can never be met. Operationally the check treats the candidate preference as if it were hard and tests it against the hard set, which exposes two kinds of clash. One is *structural*: a preference may contradict the product model or a hard constraint at the ontological level, such as a request for a forbidden option, which description-logic reasoning decides. The other is *numeric*: a preference may violate a value bound outright, as in the 90 kWh request above, which description logic cannot decide and which requires reasoning over the arithmetic of the formula constraints. Where this combined check runs depends on where the preference extension is hosted. On the axiom-based ConfigDL, an OWL DL reasoner decides the structural case while DL-safe rules in the Semantic Web Rule Language (SWRL) with arithmetic built-ins check the numeric formula constraints over

already-grounded values (feasibility over open variables is beyond them), keeping the check inside OWL, though relaxable constraints have no native axiomatic home there. On the instance-based ConfigVoc, the formula constraints export to a constraint solver such as MiniZinc (a path CONTO already provides [6]), which checks them numerically and, through reified constraints aggregated into a weighted objective, gives the soft-constraint semantics a natural target; ASP is a further transformation target, realizing priorities directly as weak constraints with a weight and a level. Either way, passing the static check guarantees only that each preference is individually satisfiable, not that all can be met together. Soft preferences are deliberately not tested against one another: because each may be relaxed, no pair of them is unsatisfiable, so which of two competing preferences yields is a question of priority for the scheduler, not of consistency.

At this stage, acquisition is a two-way exchange between the neural and symbolic components, mediated by the ontology. Each assistant grounds an open-ended natural-language utterance into typed constraints over the shared vocabulary, so the check operates not on text but on the same formal representation the hardware sources populate. Because that grounding is fallible, the coupling has a useful side effect: a mis-grounded constraint (one the assistant translated incorrectly, say mapping a requested energy to the wrong feature or unit) that clashes with the hard limits is caught by the same check rather than passing silently downstream. When a preference fails the check, it returns a *justification*: a minimal set of the hard constraints responsible for the clash. Both hosting paths supply one, and this is what selects them: an OWL DL reasoner with SWRL support (Pellet/Openllet) computes the justification of an inconsistency, the minimal set of clashing axioms and rules, and a constraint solver such as MiniZinc reports the analogous minimal unsatisfiable subset of the numeric constraints. A tool that returned only a verdict, without the clashing set, could not drive the loop. Either way the result is a set of *named* constraints in the shared vocabulary, not free text. It is handed back to the assistant that supplied the offending preference as a structured tool-call result, and the assistant is prompted to surface those named constraints to the user, renegotiate, and feed a revised constraint into the loop. The assistant's next turn is thus conditioned on a symbolic artifact rather than on retrieved text or a scalar signal: the justification determines which constraint is questioned and how the trade-off is framed. Iterating this loop restores a statically feasible set.

The second relaxation point is the scheduler, which resolves quantitative and resource feasibility. Even a structurally consistent set can be jointly unsatisfiable over the planning horizon, as in the 16:00 request above. Here the priorities and weights determine which soft constraints are relaxed, and by how much, so as to minimize total violation cost while the hard constraints hold. The outcome is reported back to the user through the same assistant, closing the loop a second time at the level of the realized schedule rather than the constraint set.

4. Use Case: The Microgrid

The Siemens microgrid is a use case in the FLEXI project. Downstream of acquisition, a scheduler forecasts variable factors such as PV production and grid energy CO_2 intensity and seeks a schedule that satisfies all hard constraints and as many soft constraints as possible; making that scheduler neurosymbolic is outside the scope of this paper.

Constraints are acquired through two assistants. An *EV-owner assistant* elicits user preferences such as desired departure time and desired energy or state of charge, and, where legacy hardware cannot report it automatically, prompts the user for the current state of charge or battery size. An *operator assistant* captures infrastructure-level requirements from the charge-point operator, such as stationary-battery reserves and campus power limits. Hardware specifications add the remaining physical limits: fuse and transformer capacities, battery safety parameters, and committed flexibility-market obligations (e.g., day-ahead bids).

A worked example. The two relaxation points appear in turn. Suppose an EV owner requests an option the product model forbids for the installed hardware (a structural conflict), or requests to charge

90 kWh into a battery that holds 60 (a numeric one). Either way the preference cannot be honored under the hard limits no matter how anything else is scheduled; the static check flags it, and the returned justification lets the assistant ask the user to revise the request before any scheduling is attempted. Now suppose instead the request is individually feasible, a full charge by 16:00, but that morning many vehicles charge simultaneously and the stationary battery is drawn down, so the transformer limit and the battery's reserved power leave too little capacity to meet every EV owner's target in time. No single request is infeasible, so the static check passes; the shortfall surfaces only in the scheduler, which cannot satisfy all soft preferences jointly. Guided by the priorities, it relaxes the lower-priority ones, delivering an 80% charge by 16:00, while holding the hard limits, and reports the outcome to the affected owner.

5. Discussion and Conclusion

We have argued that multi-stakeholder *constraint acquisition* is a neurosymbolic problem worth treating in its own right, and instantiated an ontology-mediated architecture for it on the FLEXI microgrid. Nothing in the architecture, however, is specific to energy. It applies wherever a specification must be assembled from constraints acquired across several human and automated sources of unequal authority, with the ontology as the shared representation, the hard/soft distinction marking which constraints may be relaxed, and the explanation-and-renegotiation loop resolving the conflicts that arise during acquisition. Multi-tenant building climate control is one such setting, far from energy scheduling yet structurally identical: tenants express comfort preferences (soft) through their own assistants, a facility manager contributes building-wide energy caps and safety requirements of higher authority, and equipment datasheets fix actuator and plant limits (hard); tenants compete with one another for the same capped supply, as EV owners do for transformer capacity.

Limitations and open questions. We describe an architecture rather than an evaluated system, and several questions remain open. The preference extension and the explanation feedback loop are proposed: the extension and the assistants are not yet built, and the loop has been exercised only on the OWL path. A first evaluation would target the acquisition loop rather than schedule quality: how often the static check catches a preference that could never be satisfied, how many renegotiation turns follow, and whether checking and solver encoding stay within interactive time. A first design question is where the preference semantics belong, the ConfigVoc-versus-ConfigDL choice detailed in Section 3.2; because both paths supply inconsistency justifications, it governs how preferences are checked, not whether the loop can run. Modeling preferences as weighted soft constraints further raises the questions of how to set and compare priorities across stakeholder groups, and how to handle multiple competing soft constraints (multi-objective optimization); because the groups are self-interested, priority elicitation also carries a social-choice dimension, with the risk of strategic misreporting. The approach also assumes the LLM assistants translate faithfully between natural language and the ontology vocabulary; a mis-grounding that clashes with the hard limits is caught, but a consistent yet wrong one would propagate silently, making faithful grounding the make-or-break assumption. For grounding we would adopt existing practice rather than extend it: constrained decoding against the vocabulary, as in the configuration copilot [11], which yields well-formed constraints but not necessarily the intended ones. Furthermore, description-logic reasoning must remain fast enough for interactive renegotiation. Resolving the ConfigVoc-versus-ConfigDL placement and establishing priority semantics, faithful elicitation, and interactive-time reasoning is the research agenda this work motivates; a first prototype of the preference extension is our immediate next step.

Acknowledgments

This research project is funded by CETPartnership, the Clean Energy Transition Partnership under the 2024 joint call for research proposals, co-funded by the European Commission (GA N°101069750)

and with the funding organisations FFG Austrian Research Promotion Agency (Austria), NWO (Dutch Research Council) (the Netherlands), Swedish Energy Agency (Sweden) and GSRI (Greece).

Declaration on Generative AI

During the preparation of this work, the authors used Claude in order to: Paraphrase and reword, Improve writing style, Grammar and spelling check. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] F. van Harmelen, A. ten Teije, A boxology of design patterns for hybrid learning and reasoning systems, Journal of Web Engineering 18 (2019) 97–124. URL: https://ieeexplore.ieee.org/document/10247288.

[2] T. Scholak, N. Schucher, D. Bahdanau, PICARD: Parsing incrementally for constrained auto-regressive decoding from language models, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2021, pp. 9895–9901. doi:10.18653/v1/2021.emnlp-main.779.

[3] L. Pan, A. Albalak, X. Wang, W. Y. Wang, Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning, in: Findings of the Association for Computational Linguistics: EMNLP, 2023, pp. 3806–3824. doi:10.18653/v1/2023.findings-emnlp.248.

[4] T. X. Olausson, A. Gu, B. Lipkin, C. E. Zhang, A. Solar-Lezama, J. B. Tenenbaum, R. Levy, LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2023, pp. 5153–5176. doi:10.18653/v1/2023.emnlp-main.313.

[5] X. Ye, Q. Chen, I. Dillig, G. Durrett, SatLM: Satisfiability-aided language models using declarative prompting, in: Advances in Neural Information Processing Systems (NeurIPS), volume 36, 2023, pp. 45548–45580. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/8e9c7d4a48bdac81a58f983a64aaf42b-Abstract-Conference.html.

[6] S. Bischof, A. Falkner, E. Filtz, P. Schneider, S. Steyskal, M.-L. Topa, CONTO: An ontology-based approach for interoperable configuration knowledge, in: D. Chaves-Fraga, I. Heibi, D. Garijo, D. Collarana, A. Salatino, S. Vahdati (Eds.), Posters and Demos Track of SEMANTiCS, volume 4064 of *CEUR Workshop Proceedings*, 2025. URL: https://ceur-ws.org/Vol-4064/PD-paper13.pdf.

[7] C. Bessiere, F. Koriche, N. Lazaar, B. O'Sullivan, Constraint acquisition, Artificial Intelligence 244 (2017) 315–342. doi:10.1016/j.artint.2015.08.001.

[8] C. Bessiere, R. Coletta, E. Hebrard, G. Katsirelos, N. Lazaar, N. Narodytska, C.-G. Quimper, T. Walsh, Constraint acquisition via partial queries, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2013, pp. 475–481. URL: https://www.ijcai.org/Proceedings/13/Papers/078.pdf.

[9] B. Zhang, V. A. Carriero, K. Schreiberhuber, S. Tsaneva, L. Sánchez González, J. Kim, J. de Berardinis, OntoChat: A framework for conversational ontology engineering using language models, in: The Semantic Web: ESWC Satellite Events, 2024, pp. 102–121. doi:10.1007/978-3-031-78952-6_10.

[10] C. Lawless, J. Schoeffer, L. Le, K. Rowan, S. Sen, C. St. Hill, J. Suh, B. Sarrafzadeh, "I want it that way": Enabling interactive decision support using large language models and constraint programming, ACM Transactions on Interactive Intelligent Systems 14 (2024) 22:1–22:33. doi:10.1145/3685053.

[11] P. Kogler, W. Chen, A. A. Falkner, A. Haselböck, S. Wallner, Configuration copilot: Towards integrating large language models and constraints, in: E. Vareilles, C. Grosso, J. M. Horcas, A. Felfernig (Eds.), Proceedings of the International Workshop on Configuration (ConfWS), volume 3812 of *CEUR Workshop Proceedings*, 2024, pp. 101–110. URL: https://ceur-ws.org/Vol-3812/paper14.pdf.

[12] T. Schiex, H. Fargier, G. Verfaillie, Valued constraint satisfaction problems: Hard and easy problems, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 1995, pp. 631–639. URL: https://www.ijcai.org/Proceedings/95-1/Papers/083.pdf.

[13] S. Bistarelli, U. Montanari, F. Rossi, Semiring-based constraint satisfaction and optimization, J. ACM 44 (1997) 201–236. doi:10.1145/256303.256306.

[14] F. Buccafurri, N. Leone, P. Rullo, Enhancing disjunctive datalog by constraints, IEEE Trans. Knowl. Data Eng. 12 (2000) 845–860. doi:10.1109/69.877512.