Connecting Proof Theory and Knowledge Representation: 
Sequent Calculi and the Chase with Existential Rules

Tim S. Lyon¹, Piotr Ostropolski-Nalewaja¹²

¹Technische Universität Dresden
²University of Wrocław
{timothy_stephen.lyon, piotr.ostropolski-nalewaja}@tu-dresden.de

Abstract

Chase algorithms are indispensable in the domain of knowledge base querying, which enable the extraction of implicit knowledge from a given database via applications of rules from a given ontology. Such algorithms have proved beneficial in identifying logical languages which admit decidable query entailment. Within the discipline of proof theory, sequent calculi have been used to write and design proof-search algorithms to identify decidable classes of logics. In this paper, we show that the chase mechanism in the context of existential rules is in essence the same as proof-search in an extension of Gentzen’s sequent calculus for first-order logic. Moreover, we show that proof-search generates universal models of knowledge bases, a feature also exhibited by the chase. Thus, we formally connect a central tool for establishing decidability proof-theoretically with a central decidability tool in the context of knowledge representation.

1 Introduction

Existential Rules and the Chase. The formalism of existential rules is a significant sub-discipline within the field of knowledge representation, offering insightful results within the domain of ontology-based query answering (Baget et al. 2009), data exchange and integration (Fagin et al. 2005), and serving a central role in the study of generic decidability criteria (Feller et al. 2023). Ontology-based query answering is one of the principal problems studied within the context of existential rules, and asks if a query is logically entailed by a given knowledge base (KB) \( K = (D, R) \), where \( D \) is a database and \( R \) is a finite set of existential rules (Baget et al. 2011). Databases generally consist of positive atomic facts such as \( \text{Female(Marie)} \) or \( \text{Mother(Zuza, Marie)} \), while existential rules—which are first-order formulae of the form \( \forall x y \exists \alpha(x, y) \rightarrow \exists z \phi(z) \) with \( \beta \) and \( \alpha \) conjunctions of atoms—are used to encode a logical theory or ontology that permits the extraction of implicit knowledge from the encompassing KB.

The primary tool for studying query answering within this setting is the so-called chase, an algorithm that iteratively saturates a given database under applications of existential rules (Beeri and Vardi 1984). The chase is useful in that it generates a universal model satisfying exactly those queries entailed by a KB, and thus, allows for the reduction of query entailment to query checking over the constructed universal model (Deutsch, Nash, and Remmel 2008). In this paper, we show how the chase corresponds to proof-search in an extension of Gentzen’s sequent calculus, establishing a connection between a central tool in the theory of existential rules with the primary decidability tool in proof theory.

Sequent Calculi and Proof-Search. Since its introduction, Gentzen’s sequent formalism (Gentzen 1935a; Gentzen 1935b) has become one of the preferred proof-theoretic frameworks for the creation and study of proof calculi. A sequent is an object of the form \( \Gamma \vdash \Delta \) such that \( \Gamma \) and \( \Delta \) are finite (multi)sets of logical formulae, and a sequent calculus is a set of inference rules that operate over such sequents. Systematic, and generalizations thereof, have proved beneficial in establishing (meta)logical properties with a diverse number of applications, being used to write decision algorithms (Dyckhoff 1992; Slaney 1997), to calculate interpolants (Maehara 1960; Lyon et al. 2020), and have even been applied in knowledge intergration scenerios (Lyon and Gómez Álvarez 2022).

It is well-known that geometric implications, i.e., first-order formula of the form \( \forall x (\phi \rightarrow \exists y_1 \psi_1 \lor \cdots \lor \exists y_n \psi_n) \) with \( \psi \) and \( \psi_i \) conjunctions of atoms, can be converted into an inference rules in a sequent calculus (Simpson 1994, p. 24). Since such formulae subsume the class of existential rules, we may leverage this insight to extend Gentzen’s sequent calculus for first-order logic with such rules to carry out existential rule reasoning. When we do so, we find that sequent systems mimic existential rule reasoning and proof-search (described below) simulates the chase.

Proof-search is the central means by which decidability is obtained with a sequent calculus, and usually operates by applying the inference rules of a sequent calculus bottom-up on an input sequent with the goal of constructing a proof thereof. If a proof of the input is found, the input is confirmed to be valid, and if a proof of the input is not found, a counter-model can typically be extracted witnessing the invalidity of the input. We make the novel observation that counter-models extracted from proof-search (in the context of existential rules) are universal, being homomorphically

¹Existential rules are also referred to as a tuple-generating dependencies (Abiteboul, Hull, and Vianu 1995), conceptual graph rules (Salvat and Mugnier 1996), Datalog² (Gottlob 2009), and \( \forall \exists \)-rules (Baget et al. 2011) in the literature.
equivalent to the universal model generated by the chase.

Contributions. Our contributions in this paper are as follows: (1) We establish a strong connection between tools in the domain of existential rules with that of proof theory; in particular, we show how to transform derivations with existential rules into sequent calculus proofs and vice versa. (2) We establish a correspondence between the chase and sequent-based proof-search, and (3) we recognize that proof-search, like the chase, generates universal models for knowledge bases, which is a novel, previously unknown insight regarding the capability of sequent systems.

Organization. The preliminaries are located in Section 2. In Section 3, we present the sequent calculus framework and write a proof-search algorithm that simulates the chase. Correspondences between existential rule reasoning and sequent-based reasoning are explicated in Section 4, and in Section 5, we conclude and discuss future research. The reader may consult the appended version of this paper (Lyon and Ostropolski-Nalewaja 2023) for proofs of claims.

2 Preliminaries and Existential Rules

Formulae and Syntax. We let C and V be two disjoint denumerable sets of constants and variables. We use a, b, c, . . . to denote constants and x, y, z, . . . to denote variables. We define the set of terms to be T = C ∪ V , and we denote terms by t and annotated versions thereof. Moreover, we let P = {p, q, r, . . . } be a denumerable set of predicates containing denumerably many predicates of each arity n ∈ N, and use ar(p) = n to denote that p ∈ P is of arity n. An atom is a formula of the form p(t1, . . . , tn) such that t1, . . . , tn ∈ T and ar(p) = n. We will often write atoms as p(t) with t = t1, . . . , tn. The first-order language L is defined via the following grammar in Backus–Naur form:

φ ::= p(t) | ¬φ | φ ∧ ψ | 3x φ

such that p ∈ P , t ∈ T , and x ∈ V . We use ϕ, ψ, χ, . . . to denote formulae from L , and define ϕ ∨ ψ := ¬(¬ϕ ∧ ¬ψ), ϕ → ψ := ¬ϕ ∨ ψ, and ∀x ϕ := ¬∃x ¬ϕ. The occurrence of a variable is free in a formula ϕ when it does not occur within the scope of a quantifier. We let ϕ(t/x) represent the formula obtained by substituting the term t for every free occurrence of the variable x in ϕ. We use Γ, Δ, Σ, . . . to denote sets of formulae from L , let V (Γ) denote the set of free variables in the formulae of Γ, and let T (Γ) denote the set of set of free variables and constants occurring in the formulae of Γ. We let i ∈ [n] represent 1 ≤ i ≤ n, and define a ground atom to be an atom p(t1, . . . , tn) such that for each i ∈ [n], ti ∈ C. An instance I is defined to be a (potentially infinite) set of atoms, and a database D is defined to be a finite set of ground atoms. We let T be a special unary predicate and define I ∪ T = I ∪ {T(c) | c ∈ C}. An instance I is referred to as an interpretation iff I ⊇ T = I.

Substitutions. A substitution σ is defined to be a partial function over T. A homomorphism from an instance I to an instance J is a substitution π from the terms of I to the terms of J such that (1) if p(t1, . . . , tn) ∈ I, then p(π(t1), . . . , π(tn)) ∈ J, and (2) π(a) = a, for each a ∈ C. We say that an instance I homomorphically maps into an instance J iff a homomorphism exists from I to J. Two instances I and J are defined to be homomorphically equivalent, written I ≡ J, iff each instance can be homomorphically mapped into the other. An I-assignment is defined to be a substitution µ such that (1) µ(x) ∈ T(I), for each x ∈ V , and (2) µ(a) = a, for each a ∈ C. For an I-assignment µ, we let µ(φ) denote the formula obtained by replacing each free variable of φ with its value under µ, and we let µ[t/x] be the same as µ, but where the variables x are respectively mapped to t ∈ T.

Models and Satisfaction. Given an interpretation I and an I-assignment µ, we recursively define satisfaction := as:

1. I, µ |= p(t1, . . . , tn) iff p(µ(t1), . . . , µ(tn)) ∈ I;
2. I, µ |= ¬ϕ iff I, µ ̸|= ϕ;
3. I, µ |= ϕ ∧ ψ iff I, µ |= ϕ and I, µ |= ψ;
4. I, µ |= 3x ϕ iff t ∈ T(I) exists and I, µ[t/x] |= ϕ.

We say that I is a model of Γ and write I |= Γ for every ϕ ∈ Γ and I-assignment µ, we have I, µ |= ϕ. We define an instance I to be a universal model of Γ iff for any model J of Γ there exists a homomorphism from I to J.

Existential Rules. An existential rule is a first-order formula ρ = ∀xy β(x, y) → ∃z α(y, z) such that β(x, y) = body(ρ) (called the body) and α(y, z) = head(ρ) (called the head) are conjunctions of atoms over constants and the variables x, y, and z, respectively. We call a finite set R of existential rules a rule set. We define Γ to be R-valid iff for every interpretation I, if I |= R, then I |= Γ.

Derivations and the Chase. We say that an existential rule ρ is applicable to an instance I iff there exists an I-assignment µ such that µ(β(x, y)) ⊆ I, and when this is the case, we say that τ = (ρ, µ) is a trigger in I. Given a trigger τ = (ρ, µ) in I we define an application of the trigger τ to the instance I to be the instance I(τ) = I ∪ α(µ(y), z) where z is a tuple of fresh variables. We define a chase derivation (I0, τ1), . . . , (In, τn) such that for every i ∈ [n], τi is a trigger in In and τi(Σn) = Ii+1. For an instance I and a rule set R, we define the one-step chase to be:

C1(H, R) = ∪ τ(Σ).

τ is a trigger in I

We let Cn(I, R) = I as well as let Cn+1(I, R) = C1(Hn(i, R), R). Finally, we define the chase to be Cn(I, R) = (U i∈[n] Cn(I, R))1, which serves as a universal model of I ∪ R in Deutsch, Nash, and Remmel (2008).

Queries and Entailment. A Boolean conjunctive query (or BQ) is a formula 3x q(x) such that q(x) is a conjunction of atoms over the variables x and constants. We define a knowledge base (or KB) to be an ordered pair K = (D, R) with D a database and R a rule set, and let I be a model of K, written I |= K, iff I |= D ∪ R. We write K |= 3x q(x) to mean that for every I, if I |= K, then I |= 3x q(x). A chase derivation (I0, τ1), . . . , τn witnesses (D, R) |= 3x q(x).

2 We use a restricted variant of the chase; cf. (Fagin et al. 2005).
iff \( I_1 = D \), only rules from \( \mathcal{R} \) are applied, and there exists an \( I_{n+1} \)-assignment \( \mu \) such that \( \mu(q(x)) \subseteq I_{n+1} \).

3 Sequent Systems and Proof-Search

We define a \textit{sequent} to be an object of the form \( \Gamma \vdash \Delta \) such that \( \Gamma \) and \( \Delta \) are \textit{finite} sets of formulae from \( \mathcal{L} \). Typically, multisets are used in sequents rather than sets, however, we are permitted to use sets in the setting of classical logic; cf. (Kleene 1952). For a sequent \( \Gamma \vdash \Delta \), we call \( \Gamma \) the \textit{antecedent} and \( \Delta \) the \textit{consequent}. We define the \textit{formula interpretation} of a sequent to be \( f(\Gamma \vdash \Delta) = \bigwedge \Gamma \rightarrow \bigvee \Delta \).

The sequent calculus G3 (Kleene 1952) for first-order logic is defined to be the set of inference rules presented in Figure 1. It consists of the initial rule \((id)\) along with \textit{logical rules} that introduce complex logical formulae in either the antecedent or consequent of a sequent. The \((\exists_L)\) rule is subject to a side condition, stating that the rule is applicable only if \( y \) is \textit{fresh}, i.e. \( y \) does not occur in the surrounding context \( \Gamma, \Delta \). The \((\exists_R)\) rule allows for the bottom-up instantiation of an existentially quantified formula with a term \( t \). An \textit{application} of a rule is obtained by instantiating the rule with formulae from \( \mathcal{L} \). We call an application of a rule \textit{top-down (bottom-up)} whenever the conclusion (premises) is (are) obtained from the premises (conclusion).

It is well-known that every \textit{geometric implication}, which is a formula of the form \( \forall x(\varphi \rightarrow \exists y_1 \psi_1 \lor \cdots \lor \exists y_n \psi_n) \) with \( \varphi \) and \( \psi_i \) conjunctions of atoms, can be converted into an inference rule; see (Simpson 1994, p. 24) for a discussion. We leverage this insight to transform existential rules (which are special instances of geometric implications) into inference rules that can be added to the sequent calculus G3. For an existential rule \( \rho = \forall x y \beta(x, y) \rightarrow \exists z \alpha(y, z) \), we define its corresponding \textit{sequent rule} \( s(\rho) \) to be:

\[
\frac{\Gamma, \beta(x, y), \alpha(y, z) \vdash \Delta}{\Gamma, \beta(x, y) \vdash \Delta} s(\rho) \quad z \text{ fresh}
\]

Note that we take the body \( \beta(x, y) \) and head \( \alpha(y, z) \) to be sets of atoms, rather than conjunctions of atoms, and we note that \( x, y \) may be instantiated with terms in rules application. Also, \( s(\rho) \) is subject to the side condition that the rule is applicable only if all variables \( z \) are fresh. We define the sequent calculus G3(\( \mathcal{R} \)) as \( G3 \cup \{ s(\rho) \mid \rho \in \mathcal{R} \} \). We define a \textit{derivation} to be any sequence of applications of rules in G3(\( \mathcal{R} \)) to arbitrary sequents, define an \( \mathcal{R} \)-\textit{derivation} to be a derivation that only applies \( s(\rho) \) rules, and define a \textit{proof} to be a derivation starting from applications of the \((id)\) rule. An example of a proof is shown on the left side of Figure 2.

Theorem 1 (Soundness and Completeness). \( f(\Gamma \vdash \Delta) \) is \( \mathcal{R} \)-valid iff there exists a proof of \( \Gamma \vdash \Delta \) in G3(\( \mathcal{R} \)).

Algorithm 1 The proof-search algorithm Prove.

\textbf{Input:} A sequent \( \Gamma \vdash \Delta \).
\textbf{Output:} A Boolean True and False.

If no rule is applicable to \( \Gamma \vdash \Delta \), Return False;
If there exists a \( p(t) \in \Gamma \cap \Delta \), Return True;
If \( \varphi \land \psi \in \Delta \), but \( \varphi, \psi \notin \Delta \),
\begin{align*}
\text{Set } & \Delta_1 := \varphi, \Delta \text{ and } \Delta_2 := \psi, \Delta; \\
\text{If } & \text{Prove}(\Gamma \vdash \Delta_i) = \text{False for some } i \in \{1, 2\}, \\
\text{Return } & \text{False;}
\end{align*}
\text{Else}
\begin{align*}
\text{Return } & \text{True;}
\end{align*}
\begin{align*}
\text{If } & \exists x \varphi \in \Delta \text{ and } t \notin T(\Gamma), \text{but } \varphi(t/x) \notin \Delta, \\
\text{Set } & \Delta := \varphi(t/x), \Delta; \text{Return Prove}(\Gamma \vdash \Delta); \\
\text{Let } & \rho = \forall x y \beta(x, y) \rightarrow \exists z \alpha(y, z) \text{ be the next rule according to } \prec (\text{if no rule has yet been picked, choose one in } \mathcal{R}); \\
\text{If } & \exists \gamma \in \Delta \text{ and } \rho(t) \notin \Delta, \text{but } \gamma(t) \notin \Delta, \\
\text{Set } & \Delta := \gamma(t), \Delta; \text{Return Prove}(\Gamma \vdash \Delta); \\
\text{Else}
\begin{align*}
\text{If } & \exists x \varphi \in \Delta \text{ and } t \notin T(\Gamma), \text{but } \varphi(t/x) \notin \Delta, \text{and } \rho(t) \notin \Delta, \\
\text{Set } & \Delta := \gamma(t), \Delta; \text{Return Prove}(\Gamma \vdash \Delta);
\end{align*}
\end{align*}

We now define a proof-search algorithm that decides (under certain conditions) if a BCQ is entailed by a knowledge base. The algorithm Prove (shown above) takes a sequent of the form \( \mathcal{D} \vdash \exists x \varphi(x) \) as input and bottom-up applies inference rules from G3(\( \mathcal{R} \)) with the goal of constructing a proof thereof. Either, Prove returns a proof witnessing that \( (\mathcal{D}, \mathcal{R}) \models \exists x \varphi(x) \), or a counter-model to this claim can be extracted from failed proof search. Due to the shape of the input \( \mathcal{D} \vdash \exists x \varphi(x) \), only \((id)\), \((\exists_L)\), \((\exists_R)\), and \( s(\rho) \) rules are applicable during proof search. We note that a rule during proof-search is applicable to a sequent whenever a bottom-up application of the rule will introduce a new formula into the sequent. Moreover, we let \( \prec \) be an arbitrary cyclic order over \( \mathcal{R} = \{\rho_1, \ldots, \rho_n\} \), that is, \( \rho_1 \prec \rho_2 \cdots \rho_{n-1} \prec \rho_n \prec \rho_1 \). We use \( \prec \) to ensure the fair application of \( s(\rho) \) rules during proof-search, meaning that no bottom-up rule application is delayed indefinitely.

Theorem 2. Let \( \mathcal{R} \) be a rule set, \( \mathcal{D} \) be a database, and \( \exists x \varphi(x) \) be a BCQ. Then,

1. If Prove\( (\mathcal{D} \vdash \exists x \varphi(x)) = \text{True} \), then a proof in G3(\( \mathcal{R} \)) can be constructed witnessing that \( (\mathcal{D}, \mathcal{R}) \models \exists x \varphi(x) \);
2. If Prove\( (\mathcal{D} \vdash \exists x \varphi(x)) \neq \text{True} \), then a universal model can be constructed witnessing that \( (\mathcal{D}, \mathcal{R}) \not\models \exists x \varphi(x) \).

We refer to the universal model of \( (\mathcal{D}, \mathcal{R}) \) stated in the second claim of Theorem 2 as the \textit{witnessing counter-model}. 771
Let \( \Gamma \vdash \exists x (A(x, a) \land F(x)) \), \( A(c, a) \) \((id)\)

\[
\begin{align*}
\Gamma \vdash \exists x (A(x, a) \land F(x)) & \quad (id) \\
\Gamma \vdash \exists x (A(x, a) \land F(x)) & \quad (\land) \\
M(b, a), A(b, a), F(b), M(c, b), A(c, b), F(c), A(c, a) & \vdash \exists x (A(x, a) \land F(x)) \quad s(\rho_2) \\
M(b, a), A(b, a), F(b), M(c, b), A(c, b), F(c) & \vdash \exists x (A(x, a) \land F(x)) \quad s(\rho_1) \\
M(b, a), M(c, b) & \vdash \exists x (A(x, a) \land F(x)) 
\end{align*}
\]

Theorem 9. I

If \( \Gamma = M(b, a), A(b, a), F(b), M(c, b), A(c, b), F(c), A(c, a) \), above right is an illustration showing that the BCQ \( \exists x (A(x, a) \land F(x)) \) (to the right) can be mapped into the chase \( \text{Ch}_\infty (D, R) \) (to the left) via the \( \text{Ch}_\infty (D, R) \)-assignment \( \mu \) (dotted arrows).

4 Simulations and Equivalences

We present a sequence of results which culminate in the establishment of two main theorems: (1) Theorem 8, which confirms that chase derivations are mutually transformable with certain proofs in \( G_3(R) \), and (2) Theorem 9, which confirms an equivalence between prove and the chase. We end the section by providing an example illustrating the latter correspondence between proofs and the chase.

Observation 3. Let \( R \) be a rule set. If \( \rho \in R \), then any application of \((\land_R)\) and \((\exists_R)\) permute above \( s(\rho) \).

Proof. It is straightforward to confirm the permutation of such rules as the \( s(\rho) \) rules operate on the antecedent of a sequent, and \((\land_R)\) and \((\exists_R)\) operate on the consequent.

Observation 4. If \( I \) is an instance, then only \( s(\rho) \) rules of \( G_3(R) \) can be bottom-up applied to \( I \vdash \emptyset \). Moreover, such an application yields a sequent \( I' \vdash \emptyset \) with \( I' \) an instance.

Observation 5. The inference shown below left is a correct application of \( s(\rho) \) iff the inference shown below right is:

\[
\begin{align*}
\Gamma I' & \vdash \emptyset \quad s(\rho) \\
\Gamma I & \vdash \emptyset \\
\Gamma I' & \vdash \Delta \quad s(\rho) \\
\Gamma I & \vdash \Delta
\end{align*}
\]

Observation 6. Let \( I \) and \( I' \) be instances with \( \tau = (\rho, \mu) \) a trigger on \( I \). Then, \( (I, \tau) \), \( (I', \emptyset) \) is a chase derivation iff the following is a correct application of \( s(\rho) \):

\[
\begin{align*}
I' \vdash \emptyset \\
I \vdash \emptyset \\
I' \vdash \Delta \\
I \vdash \Delta
\end{align*}
\]

Lemma 7. For every rule set \( R \), \( n \in \mathbb{N} \), and instances \( I_1, \ldots, I_n \) there exists a chase derivation \((I_1, \tau_1)_{i \in [n-1]} \) iff there exists an \( R \)-derivation of \( I_1 \vdash \emptyset \) from \( I_n \vdash \emptyset \).

To prove the following theorem, one shows that every chase derivation can be transformed into a proof in \( G_3(R) \) and vice-versa.

Theorem 8. Let \( R \) be a rule set. A chase derivation \((I_1, \tau_1)_{i \in [n]}\) witnessing \( (\text{Ch}, D, R) \) \( \models \exists x q(x) \) exists iff a proof in \( G_3(R) \) of \( D \vdash \exists x q(x) \) exists.

Leveraging Theorems 2 and 8, it is straightforward to prove the first claim of the theorem below. The second claim is immediate as \( I \) and \( \text{Ch}_\infty (D, R) \) are universal models.

Theorem 9. Let \( R \) be a rule set, \( D \) be a database, and \( \exists x q(x) \) be a BCQ. Then,

1. \( \text{Prove}(D \vdash \exists x q(x)) = \text{True} \) iff there is an \( n \in \mathbb{N} \) such that \( \text{Ch}_n (D, R) \models \exists x q(x) \) iff \( \text{Ch}_\infty (D, R) \models \exists x q(x) \):

2. If \( \text{Prove}(D \vdash \exists x q(x)) \neq \text{True} \), then \( I \equiv \text{Ch}_\infty (D, R) \) with \( I \) the witnessing counter-model.

Example 10. We provide an example demonstrating the relationship between a proof and the chase. We read \( F(x) \) as ‘\( x \) is female’, \( M(x, y) \) as ‘\( x \) is the mother of \( y \)’ and \( A(x, y) \) as ‘\( x \) is the ancestor of \( y \)’. We let \( K = (D, R) \) be a knowledge base such that \( D = \{ M(b, a), M(c, b), A(b, a) \} \), \( R = \{ \rho_1, \rho_2 \} \), and \( \rho_1 = \forall xy (M(x, y) \rightarrow A(x, y) \land \forall F(x)) \); \( \rho_2 = \forall xy (A(x, y) \land \forall z \rightarrow A(x, z)) \).

In Figure 2, \( K \models \exists x (A(x, a) \land F(x)) \) is witnessed and verified by the proof shown left. The graph shown right demonstrates that the BCQ \( \exists x (A(x, a) \land F(x)) \) (to the right) can be mapped into the chase \( \text{Ch}_\infty (D, R) \) (to the left) via a \( \text{Ch}_\infty (D, R) \)-assignment \( \mu \) (depicted as dotted arrows). (NB. We have omitted the points \{ \( \Gamma (c) \mid c \in C \} \) in the picture of \( \text{Ch}_\infty (D, R) \) for simplicity.)

5 Concluding Remarks

We have formally established an equivalence between existential rule reasoning and sequent calculus proofs, effectively showing that proof-search simulates the chase. This work is meaningful as it uncovers and connects two central reasoning tasks and tools in the domain of existential rules and proof theory. Moreover, we have found that the counter-models extracted from failed proof-search are universal, implying their homomorphic equivalence to the chase—a previously unrecognized observation.

For future work, we aim to examine the relationship between the disjunctive chase (Bourhis et al. 2016) and proof-search in sequent calculi with disjunctive inference rules. It may additionally be worthwhile to investigate if our sequent systems can be adapted to facilitate reasoning with non-classical variants or extensions of existential rules. For example, we could merge our sequent calculi with those of (Lyon and Gómez Álvarez 2022) for standpoint logic—a modal logic used in knowledge integration to reason with diverse and potentially conflicting knowledge sources (Gómez Álvarez and Rudolph 2021). Finally, as this paper presents a sequent calculus for querying with existential rules, we plan to further explore its utility; e.g. by identifying admissible rules or applying loop checking techniques to uncover new classes of existential rules with decidable query entailment.
Acknowledgments

Work supported by the European Research Council (ERC) Consolidator Grant 771779 (DeciGUT).

References


