Optimizing the Nominal Introduction Rule in (Hyper)Tableau Calculi

Boris Motik, Rob Shearer, and Ian Horrocks

Oxford University Computing Laboratory

1 Introduction

A tableau calculus for $\mathcal{SHIQ}$ has been known for some time, and it forms the basis for several highly successful implementations [2, 9, 10]. Extending this calculus to $\mathcal{SHOIQ}$, however, was notoriously difficult due to an interaction between nominals, inverse roles, and number restrictions, which results in (partial) loss of the forest-model property for models of $\mathcal{SHOIQ}$ knowledge bases. In this paper, we analyze the problems that arise from this combination of features and present an overview of two solutions.

The first solution, described in [4], extends the $\mathcal{SHIQ}$ calculus with an *NN-rule* that extends the non-forest-like portion of a model with new individuals. Applications of this rule can be highly nondeterministic, however, and can substantially increase the size of the generated models, which can lead to inefficiency.

We present an alternative solution based on a new *NI-rule*. Our approach does not introduce new individuals; instead, existing individuals are incorporated into the non-forest-like portion of the model. We implemented our solution in HermiT.[1] As we discuss in [8], this approach seems to work well in practice.

2 Preliminaries

To show that a description logic knowledge base $\mathcal{K} = (\mathcal{R}, \mathcal{T}, \mathcal{A})$ is satisfiable, a tableau algorithm constructs a sequence of ABoxes $\mathcal{A} = \mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_n$ called a *derivation*, where each $\mathcal{A}_i$ is obtained from $\mathcal{A}_{i-1}$ by an application of one *inference rule*.[2] The inference rules make the information implicit in the axioms of $\mathcal{R}$ and $\mathcal{T}$ explicit, and thus evolve the ABox $\mathcal{A}$ towards a (representation of a) model of $\mathcal{K}$. The algorithm terminates either if no inference rule is applicable to some $\mathcal{A}_n$, in which case $\mathcal{A}_n$ represents a model of $\mathcal{K}$, or if $\mathcal{A}_n$ contains an obvious contradiction, in which case the model construction has failed. The following inference rules are commonly used in DL tableau calculi.

- $\sqcup$-rule: Given $(C_1 \sqcup C_2)(s)$, derive either $C_1(s)$ or $C_2(s)$.
- $\sqcap$-rule: Given $(C_1 \sqcap C_2)(s)$, derive $C_1(s)$ and $C_2(s)$.

[1] `http://web.comlab.ox.ac.uk/oucl/work/boris.motik/HermiT/`

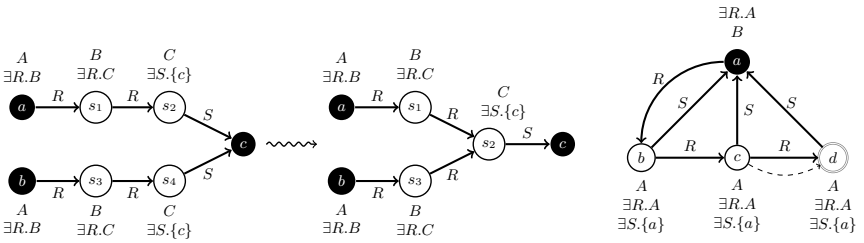
[2] Some formalizations of tableau algorithms work on *completion graphs*, which have a natural correspondence to ABoxes.

- $\exists$-rule: Given $(\exists R.C)(s)$, derive $R(s,t)$ and $C(t)$ for t a fresh individual.
- $\forall$-rule: Given $(\forall R.C)(s)$ and $R(s,t)$, derive $C(t)$.
- $\sqsubseteq$-rule: Given a GCI $C \sqsubseteq D$ and an individual s, derive $(\neg C \sqcup D)(s)$.
- $\leq$-rule: Given $\leq n\, R.C(s)$, $R(s,t_0), ..., R(s,t_n)$, and $C(t_0), ..., C(t_n)$, choose t_i and t_j for some $0 \leq i < j \leq n$ and merge t_i and t_j.

The individuals present in the original ABox $\mathcal{A}$ are called *named individuals*, and they can be connected by role assertions in an arbitrary way. The individuals introduced by the $\exists$- and $\geq$-rules are called *blockable individuals*. For example, if $\exists R.C(a)$ is expanded into $R(a,s)$ and $C(s)$, then s is a blockable *successor* of a. *Descendant* is the transitive closure of the successor relation. It is not difficult to see that no $\mathcal{SHIQ}$ tableau inference rule can connect s with an existing element of $\mathcal{A}$ other than a: the individual s can participate only in inferences that derive an assertion of the form $D(s)$ or create a new successor of s. Hence, each ABox $\mathcal{A}'$ derived from $\mathcal{A}$ can be seen as a "forest": each named individual can be arbitrarily connected to other named individuals and to a tree of blockable successors. The *concept label* $\mathcal{L}_{\mathcal{A}}(s)$ is defined as the set of all concepts C such that $C(s) \in \mathcal{A}$, and the *edge label* $\mathcal{L}_{\mathcal{A}}(s,s')$ as the set of all atomic roles such that $R(s,s') \in \mathcal{A}$.

The $\leq$-rule requires that we *merge* two individuals s and t. As described in [3], simply replacing one individual with the other in all assertions can lead to nontermination. Instead, when t is merged into s, all assertions involving descendants of t are removed; this process is called *pruning*. In order to avoid "circular pruning", tableau algorithms never merge an individual into its descendant.

A naïve application of the tableau rules does not terminate if the TBox contains existential quantifiers in cycles: given a TBox $\mathcal{T} = \{\top \sqsubseteq \exists R.\top\}$, naïve applications of the $\exists$-rule produce an infinite chain of blockable individuals. To ensure termination is such cases, tableau algorithms employ *blocking* [5]. DLs such as $\mathcal{SHIQ}$ and $\mathcal{SHOIQ}$ allow for inverse roles and number restrictions, which require *pairwise blocking* [5, 7]: given a strict ordering $\prec$ on the individuals of an ABox $\mathcal{A}$, for s' and t' any individuals and s and t blockable successors of s' and t', respectively, in $\mathcal{A}$, t *blocks* s if and only if $t \prec s$, $\mathcal{L}_{\mathcal{A}}(s) = \mathcal{L}_{\mathcal{A}}(t)$, $\mathcal{L}_{\mathcal{A}}(s') = \mathcal{L}_{\mathcal{A}}(t')$, $\mathcal{L}_{\mathcal{A}}(s,s') = \mathcal{L}_{\mathcal{A}}(t,t')$, and $\mathcal{L}_{\mathcal{A}}(s',s) = \mathcal{L}_{\mathcal{A}}(t',t)$; an individual s is *blocked* if it is blocked by another individual t or if it is a successor of a blocked individual. In tableau algorithms, the $\exists$- and $\geq$-rules are applicable only to nonblocked individuals, which ensures termination: the number of different concept and edge labels is exponential in $|\mathcal{K}|$, so an exponentially long branch in a forest-like ABox must contain a blocked individual, thus limiting the length of each branch in an ABox. Let $\mathcal{A}$ be an ABox to which no tableau inference rule is applicable, and in which s is blocked by t. We can construct a model from $\mathcal{A}$ by *unraveling*—that is, by replicating the fragment between s and t infinitely often. Intuitively, blocking ensures that the part of the ABox between s and s' "behaves" just like the part between t and t', so unraveling indeed generates a model. If our logic were able to connect blockable individuals in a non-tree-like way, then unraveling would not generate a model.



(a) Non-Tree-Like Structures due to Merging (b) Blocking and Unraveling

Fig. 1: Problems caused by nominals. Named individuals are shown in black, blockable individuals in white, and blocking is indicated with a dashed line.

3 Nominals, Inverse Roles, and Number Restrictions

We now discuss the difficulties that arise when tableau calculi are extended to handle nominals, inverse roles, and number restrictions. We summarize both the original solution presented in [4], as well as our novel approach, which we incorporated into our hypertableau calculus.

3.1 The Main Problems

There are two primary problems that must be addressed when extending any of the known $\mathcal{SHIQ}$ tableau calculi with nominals.

Nonforest models. Nominals can make ABoxes non-forest-like, as the following simple example demonstrates.

$$\begin{aligned} \mathcal{A}_1 &= \{\ A(a), \quad A(b)\ \} \\ \mathcal{T}_1 &= \{\ A \sqsubseteq \exists R.B,\ \ B \sqsubseteq \exists R.C,\ \ C \sqsubseteq \exists S.\{c\}\ \} \end{aligned} \tag{1}$$

Successive rule applications on $\mathcal{A}_1$ and $\mathcal{T}_1$ can produce the ABox $\mathcal{A}_1^1$ shown in the left-hand side of Figure 1a. This ABox is clearly not forest-shaped: the two role-paths in $\mathcal{A}_1^1$ start at the named individuals a and b, and end in a named individual c. If role relations between blockable individuals remain forest-like, however, termination of model construction can be ensured easily. Some DLs that include nominals produce only such *extended forest-like* ABoxes, but the property is lost if the DL also includes both inverse roles and number restrictions [3].

Assume now that we extend $\mathcal{T}_1$ with the axiom $\top \sqsubseteq \leq 1\, S^-$. On $\mathcal{A}_1^1$, this forces us to merge s_2 and s_4. Mergine s_4 into s_2 produces the ABox $\mathcal{A}_1^2$ shown in the right-hand side of Figure 1a. The assertion $R(s_3, s_2)$ makes the blockable portion of $\mathcal{A}_1^2$ non-forest-shaped. By extending the example, it is possible to use nominals, inverse roles, and number restrictions to arrange blockable individuals in cycles. This loss of forest-shaped models prevents the use of blocking and pruning, thus invalidating the standard termination arguments.

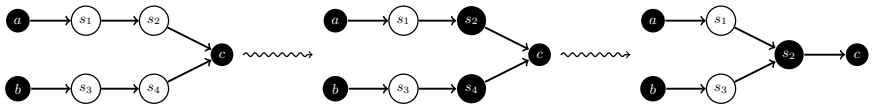


Fig. 2: The Introduction of Root Individuals

Blocking and unraveling. Even if the blockable individuals in a tableau maintain a forest structure, nominals can prevent the successful application of blocking and unraveling. Consider the following knowledge base.

$$\mathcal{A}_2 = \{\ A(a),\ (\exists R.B)(a)\ \} \qquad \mathcal{T}_2 = \left\{ \begin{array}{l} A \sqsubseteq \forall R^-.\bot,\ \ B \sqsubseteq \exists R.B \sqcap \exists S.\{a\}, \\ \top \sqsubseteq \leq 1\, R^-,\ \ \top \sqsubseteq \leq 3\, S^- \end{array} \right\} \tag{2}$$

This knowledge base contains an individual a that can have no R^--neighbors, and that serves as the root of an infinite chain of individuals, each of which must be an S^--neighbor of a.

A tableau expansion of $\mathcal{A}_2$ and $\mathcal{T}_2$ would produce the ABox $\mathcal{A}_2^1$ shown in Figure 1b. The individual d is blocked in $\mathcal{A}_2^1$ by the individual c, so the derivation terminates. Note that the last axiom from $\mathcal{T}_2$ is satisfied: a is the only individual in $\mathcal{A}_2^1$ that has S^--neighbors and it has only three such neighbors. To construct a model from $\mathcal{A}_2^1$, we unravel the blocked parts of the ABox—that is, we construct an infinite path that extends past d by "duplicating" the fragment of the model between c and d an infinite number of times. This, however, creates additional S^--neighbors of a, which invalidates the last axiom from $\mathcal{T}_2$; thus, the unraveled ABox does not define a model of $\mathcal{A}_2$ and $\mathcal{T}_2$.

3.2 A Naïve Solution

To solve the above problems, we need to extend the arbitrarily interconnected part of the ABox. To this end, in addition to named and blockable individuals, we introduce *root individuals*—freshly introduced individuals that can be arbitrarily connected to named individuals and other root individuals. We apply the following test (*): if an ABox $\mathcal{A}$ contains assertions $R(s, a)$, $A(s)$, and $\leq n\, R^-.A(a)$, with a a root or a named individual and s a blockable individual that is not a successor of a, then we change s into a root individual.

Applied to $\mathcal{A}_1^2$, this condition changes the status of s_2 and s_4 from a blockable individuals into a root individuals. After this change, only s_1 and s_3 are blockable in $\mathcal{A}_1^2$, so the ABox has the extended forest-like shape and we can apply blocking and pruning as usual. This is schematically shown in Figure 2.

Promotion of blockable individuals to roots also elegantly solves the blocking and unraveling problem. In $\mathcal{A}_2$ and $\mathcal{T}_2$, because a must satisfy an at-most restriction of the form $\leq 3\, S^-$, as soon as $S(d, a)$ is derived, condition (*) is fulfilled and we turn d into a root individual, which prevents premature blocking.

This solution, however, introduces another problem: the number of root individuals can now grow arbitrarily, as shown in the following example.

$$\mathcal{A}_3 = \{\ A(b)\ \} \quad \mathcal{T}_3 = \{\ A \sqsubseteq \exists R.A,\ A \sqsubseteq \exists S.\{a\}\ \top \sqsubseteq \leq 2\, S\ \} \tag{3}$$

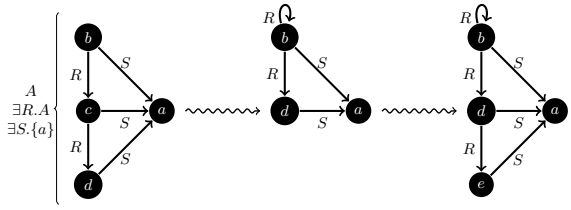


Fig. 3: A Yo-Yo With Root Individuals

On $\mathcal{A}_3$ and $\mathcal{T}_3$, rule applications can produce the derivation sequence shown in Figure 3: after c and d are introduced as descendents of b, they satisfy condition (*) so we change them into root individuals. Subsequently, the third axiom from $\mathcal{T}_3$ is not satisfied, so we merge two neighbors of a; we choose to merge c into b. Since d is now not a blockable individual, we cannot prune it. The first axiom of $\mathcal{T}_3$ is not satisfied for d, so we must extend the ABox with a new successor e. This fresh (blockable) individual also satisfies (*) and becomes a root individual. If we continue to merge the R-neighbors of b into b, we can repeat the same inferences forever by introducing an infinite number of root individuals as S^--neighbors of a in the course of the derivation.

Even if only a finite number of root individuals were introduced as neighbors for any given root individual, a different problem can cause the introduction of an arbitrarily large number of root individuals. Consider the following knowledge base:

$$\begin{aligned} \mathcal{A}_4 &= \left\{ \begin{array}{l} S(a,a), \\ \exists R.B(a) \end{array} \right\} \\ \mathcal{T}_4 &= \left\{ \begin{array}{ll} B \sqsubseteq \exists R.C, & C \sqsubseteq \exists S.D, \\ D \sqsubseteq \{a\}, & \top \sqsubseteq \leq 1\, S^- \end{array} \right\} \end{aligned} \tag{4}$$

Applications of the $\exists$-rule introduce two new blockable individuals, the second of which satisfies (*) and thus becomes a root individual. The resulting ABox, $\mathcal{A}_4^1$, is shown in the left-hand side of Figure 4. Due to the inverse-functionality of S, individuals a and c are merged. Both are root individuals, so neither is a descendant of the other; hence, we can choose which of the two individuals to merge into the other. Suppose we merge a into c. Then the blockable individual b is pruned; the result is shown in the center of Figure 4. Due to pruning, the existential $\exists R.B$ is no longer satisfied, so applications of the $\exists$-rule produce two more blockable individuals, the second of which again becomes a root individual. The resulting ABox, shown in the right-hand side of Figure 4, is isomorphic to $\mathcal{A}_4^1$, and the same inferences can be repeated forever with fresh individuals.

3.3 The Existing Solution

To guarantee termination, the calculus from [4] uses an NN-rule that refines condition (*). Assume that an ABox $\mathcal{A}$ contains assertions $R(s,a)$ and $A(s)$ where s is a blockable individual that is not a successor of a root or named individual a; furthermore, assume that a must satisfy an at-most restriction of

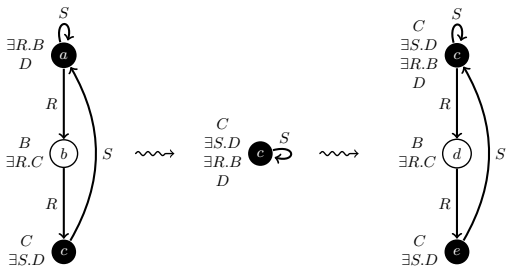


Fig. 4: A Caterpillar Example

the form $\leq n\, R^{-}.A$. If $\mathcal{A}$ already contains root individuals $z_1, ..., z_n$ such that $\bigcup_{1\leq i\leq n}\{A(z_i), R(z_i, a)\} \cup \{z_i \neq z_j \mid 1 \leq i < j \leq n\} \subseteq \mathcal{A}$, then the $\leq$-rule simply merges s into some z_i; no new root individual needs to be introduced. If $\mathcal{A}$ does not contain such $z_1, ..., z_n$, the NN-rule nondeterministically guesses the exact number $m \leq n$ of R^{-}-neighbors of a that are members of A, generates m fresh root individuals $w_1, ..., w_m$, and extends $\mathcal{A}$ with the assertions

$$\{A(w_i), R(w_i, a) \mid 1 \leq i \leq m\} \cup \{w_i \neq w_j \mid 1 \leq i < j \leq m\} \cup \{\leq m\, R.A(a)\}.$$

This allows the NN-rule to be applied at most once for each concept of the form $\leq n\, R^{-}.A$ and each root individual, which ensures termination in the "yo-yo" case: the number of neighbors introduced for each root individual is clearly finite. The "caterpillar" case is avoided by requiring that, whenever two root individuals are merged, the "newer" individual is always merged into the "older" one.

For example, Figure 5 shows applications of the NN- and $\leq$-rules to the ABox $\mathcal{A}_5 = \{\ \exists R.\exists R.\{c\}(a),\ \leq 3\, R^{-}.\top(c)\ \}$. In $\mathcal{A}_5^1$, shown in the left-hand side of the figure, the named individual c has a blockable R^{-}-neighbor and must satisfy the restriction $\leq 3\, R^{-}.\top$. The NN-rule nondeterministically chooses whether to introduce one, two, or three fresh root individuals; the parallel branches of the derivation are shown in the center of the figure. The introduction of a single individual, shown in the top branch of Figure 5, results in deterministic application of the $\leq$-rule as the blockable individual b is merged into the fresh root individual z_1, shown in the upper right of the figure. For derivation paths on which more than one fresh root individual is introduced, application of the $\leq$-rule is nondeterministic: b can be merged into any of the new root individuals, with each choice resulting in a new branch of the derivation. Such branching can be costly in practice: all derivation paths must be fully explored in order to identify an unsatisfiable knowledge base.

Although the NN-rule does ensure termination of the tableau algorithm, it is a potential source of inefficiency in knowledge bases in which large numbers appear within at-most concepts: an application of the NN-rule involving a concept $\leq n\, R.C$ guesses among n different possible sizes for the neighbor set, and subsequent applications of the $\leq$-rule must choose how to merge the new roots with blockable individuals. In the case of just a single blockable neighbor, this

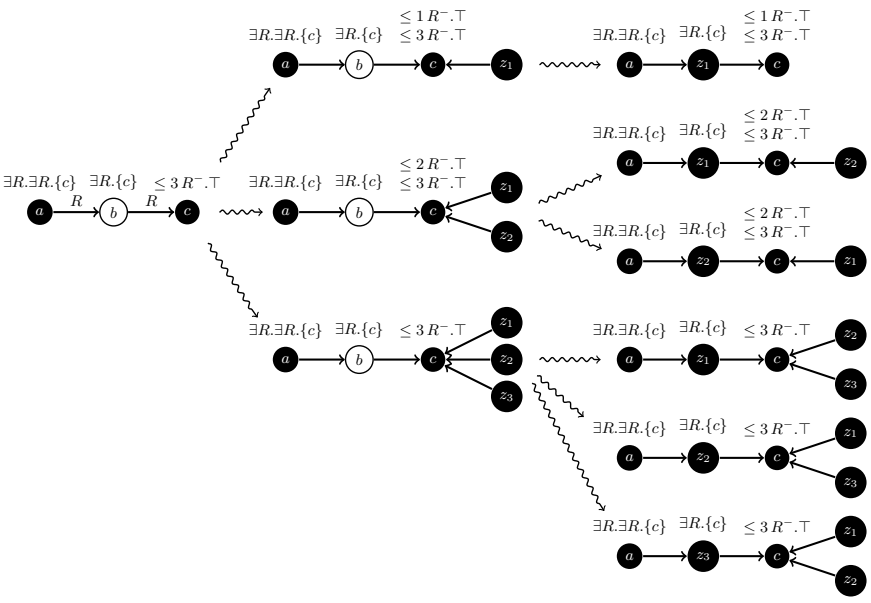


Fig. 5: An application of the NN-rule

results in a derivation tree with n^2 branches. Furthermore, the introduction of new root individuals can result in unnecessary processing and large models.

3.4 The *NI*-rule

As a replacement for the NN-rule described above, we introduce a new NI-rule, which refines the generation of root individuals. Let us again consider the ABox $\mathcal{A}$ containing assertions $R(s, a)$ and $A(s)$, and $\leq n\, R^-.A(a)$, where s is a blockable individual that is not a successor of the root or named individual a. In any model of $\mathcal{A}$, we can have at most n different individuals b_i that participate in assertions of the form $R(b_i, a)$ and $A(b_i)$. Hence, we associate with a in advance a set of n fresh root individuals $\{b_1, \ldots, b_n\}$; unlike the root individuals introduced by the NN-rule, we do not assume that $b_i \neq b_j$. Instead of choosing some subset of these individuals to introduce and relying upon the $\leq$-rule to merge them with blockable neighbors, however, we promote blockable individuals to root individuals directly: to turn s into a root individual, we nondeterministically choose b_j from this set and merge s into b_j. In this way, the number of new root individuals that can be introduced for a number restriction in the label of an individual a is limited to n. An application of our NI-rule to the ABox $\mathcal{A}_5 = \{\ \exists R.\exists R.\{c\}(a),\ \leq 3\, R^-.\top(c)\ \}$ is given in Figure 6. As with the NN-rule, by exploiting a bound on the length of paths of blockable individuals in an ABox, we can establish a bound on the number of root individuals introduced in a derivation, which ensures termination of the algorithm.

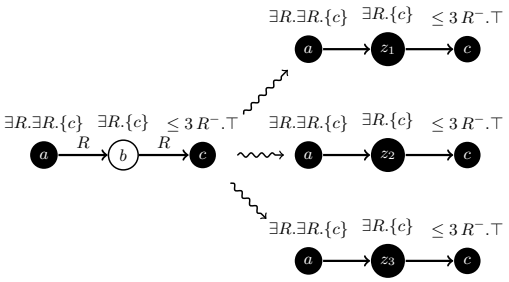


Fig. 6: An application of the NI-rule

4 A Hypertableau *NI*-Rule

We have extended the hypertableau calculus for $\mathcal{SHIQ}$ described in [7] and [6] with the NI-rule in order to support nominals. We summarize here only the aspects of the algorithm relevant to the new NI-rule; a complete discussion of the hypertableau calculus for $\mathcal{SHOIQ}$ is given in [8]. Our reasoning procedure consists of two phases: *preprocessing* and *hypertableau reasoning*.

4.1 Preprocessing

Our algorithm first preprocesses a $\mathcal{SHOIQ}$ knowledge base into an ABox and a set of *DL-clauses*—implications of the form $\bigwedge_{i=1}^{n} U_i \rightarrow \bigvee_{j=1}^{m} V_j$, where U_i are of the form $R(x,y)$ or $A(x)$, and V_j are of the form $R(x,y)$, $A(x)$, $\geq n\, R.C(x)$, or $x \approx y$. In fact, the preprocessing produces *HT-clauses*—DL-clauses of a certain syntactic structure which guarantees termination of the model construction. Due to lack of space, we leave the details of the transformation and the precise definition of HT-clauses to [8]. Roughly speaking, HT-clauses can have the form (5), where R_i and S_i are roles, A_i and B_i are atomic concepts, and C_i and D_i are either atomic concepts or concepts of the form $\geq n\, R.A$ or $\geq n\, R.\neg A$. Furthermore, ar is a function defined as $\mathsf{ar}(R,s,t) = R(s,t)$ and $\mathsf{ar}(R^-,s,t) = R(t,s)$ for R an atomic role and s and t individuals or variables.

$$\begin{aligned}(5)\qquad & \bigwedge A_i(x) \wedge \bigwedge \mathsf{ar}(R_i,x,y_i) \wedge \bigwedge B_i(y_i) \wedge \bigwedge O_{a_i}(y_{a_i}) \rightarrow \\ & \quad \bigvee C_i(x) \vee \bigvee D_i(y_i) \vee \bigvee \mathsf{ar}(S_i,x,y_i) \vee \bigvee x \approx y_{a_i} \vee \bigvee y_i \approx y_j \; @^{x}_{\leq n\, R.C}\end{aligned}$$

A simple transformation would encode nominals in DL-clauses using constants: the axiom $C \sqsubseteq \{a\}$ could be translated into $C(x) \rightarrow x \approx a$. We prefer, however to "push" all individuals from DL-clauses into the ABox; this avoids the need to rewrite our clause set if individuals are merged. To this end, we associate with each nominal $\{a\}$ an atomic *nominal guard* concept O_a and add the assertion $O_a(a)$ to the ABox. Nominals are then encoded in DL-clauses using a new variable y_a which can bind only to the (single) member of the appropriate nominal guard: the axiom $C \sqsubseteq \{a\}$ is translated into the HT-clause $C(x) \wedge O_a(y_a) \rightarrow x \approx y_a$.

When formulating the NI-rule, we are faced with a technical problem: concepts of the form $\leq n\, R.A$ are translated in our calculus into DL-clauses, which

makes testing the condition from Section 3.4 difficult. For example, an application of the *Hyp*-rule to the third DL-clause in (3) (obtained from the axiom $\top \sqsubseteq \leq 2\, S^-.\top$) can produce an equality such as $c \approx b$. This equality alone does not reflect the fact that a must satisfy the at-most restriction $\leq 2\, S^-.\top$. To enable the application of the *NI*-rule, we introduce a notion of *annotated equalities*, in which the annotations establish an association with the at-most restriction.

The translation of at-most concepts differs from the $\mathcal{SHIQ}$ case only in that, instead of generating equalities $y_i \approx y_j$, our $\mathcal{SHOIQ}$ transformation produces annotated equalities $y_i \approx y_j \, @^x_{\leq n\, R.C}$. For example, $\top \sqsubseteq \leq 1\, R.\top$ is translated into $R(x, y_1) \wedge R(x, y_2) \rightarrow y_1 \approx y_2 \, @^x_{\leq 1\, R.\top}$. The *annotation* $@^x_{\leq 1\, R.\top}$ does not affect the meaning of the equality; it merely records its provenance, and we shall discuss the usage of this provenance information shortly.

4.2 Hypertableau Reasoning

The hypertableau reasoning phase decides satisfiability of a set of HT-clauses $\mathcal{C}$ and an ABox $\mathcal{A}$.

Individuals In our algorithm, we call the individuals that occur in the input ABox *named*. Furthermore, for a named individual a, the *NI*-rule might need to introduce individuals that are unique for a, a role R, a literal concept B, and some integer i; we represent such individuals as $a.\langle R, B, i\rangle$. Since the *NI*-rule might be applied to these individuals as well, we introduce the notion of *root individuals*—finite strings of the form $a.\gamma_1. \ldots .\gamma_n$ where a is a named individual and each γ_ℓ is of the form $\langle R.B.i\rangle$.

In standard tableau algorithms, the tree structure of the model is encoded in the edges between individuals, which always point from parents to children. In contrast, our algorithm encodes the parent-child relationships into individuals themselves: it represents individuals as finite strings of the form $s.i_1, i_2, \ldots, i_n$, where s is a root individual and i_j are integers. For example, $a.2$ is the second child of the named individual a. Individuals with $n \geq 1$ are called *blockable*.

ABoxes The hypertableau algorithm operates on *generalized ABoxes*, which can contain *renamings* of the form $a \mapsto b$ where a and b are root individuals. The relation $\mapsto$ in $\mathcal{A}$ must be acyclic, $\mathcal{A}$ can contain at most one renaming $a \mapsto b$ for an individual a, and, if $\mathcal{A}$ contains $a \mapsto b$, then a must not occur in any assertion or (in)equality in $\mathcal{A}$. An individual b is the *canonical name* of a root individual a in $\mathcal{A}$, written $b = \mathsf{can}_{\mathcal{A}}(a)$, iff $a \mapsto^*_{\mathcal{A}} b$ and no individual $c \neq b$ exists such that $b \mapsto^*_{\mathcal{A}} c$, where $\mapsto^*_{\mathcal{A}}$ is the transitive-reflexive closure of $\mapsto$ in $\mathcal{A}$.

Derivation Rules Table 1 specifies *derivation rules* that, given an ABox $\mathcal{A}$ and a set of HT-clauses $\mathcal{C}$, derive the ABoxes $\mathcal{A}_1, \ldots, \mathcal{A}_n$. The main derivation rule is similar to the one of the hypertableau calculus for first order logic [1]: given an HT-clause $\bigwedge_{i=1}^m U_i \rightarrow \bigvee_{j=1}^n V_j$ and an ABox $\mathcal{A}$, the *Hyp*-rule tries to unify the atoms $U_1, \ldots, U_m$ with a subset of the assertions in $\mathcal{A}$; if a unifier σ is found, the rule nondeterministically derives $\sigma(V_j)$ for some $1 \leq j \leq n$. For example, given $R(x, y) \rightarrow \exists R.C(x) \vee D(y)$ and an assertion $R(a, b)$, the *Hyp*-rule derives either $\exists R.C(a)$ or $D(b)$. The $\geq$-rule deals with existential quantifiers: given $\exists R.C(a)$,

Table 1: Derivation Rules of the Tableau Calculus

Hyp-rule	If 1. $U_1 \wedge \ldots \wedge U_m \to V_1 \vee \ldots \vee V_n \in \mathcal{C}$, and 2. a mapping σ of variables to the individuals of $\mathcal{A}$ exists such that 2.1 $\sigma(x)$ is not indirectly blocked for each variable $x \in N_V$, 2.2 $\sigma(U_i) \in \mathcal{A}$ for each $1 \leq i \leq m$, and 2.3 $\sigma(V_j) \notin \mathcal{A}$ for each $1 \leq j \leq n$, then $\mathcal{A}_1 := \mathcal{A} \cup \{\bot\}$ if $n = 0$; $\mathcal{A}_j := \mathcal{A} \cup \{\sigma(V_j)\}$ for $1 \leq j \leq n$ otherwise.
$\geq$-rule	If 1. $\geq n\, R.C(s) \in \mathcal{A}$, 2. s is not blocked in $\mathcal{A}$, and 3. $\mathcal{A}$ does not contain individuals $u_1, \ldots, u_n$ such that 3.1 $\{\mathsf{ar}(R, s, u_i), C(u_i) \mid 1 \leq i \leq n\} \cup \{u_i \not\approx u_j \mid 1 \leq i < j \leq n\} \subseteq \mathcal{A}$, and 3.2 either s is blockable or no u_i, $1 \leq i \leq n$, is indirectly blocked in $\mathcal{A}$ then $\mathcal{A}_1 := \mathcal{A} \cup \{\mathsf{ar}(R, s, t_i),\ C(t_i) \mid 1 \leq i \leq n\} \cup \{t_i \not\approx t_j \mid 1 \leq i < j \leq n\}$ where $t_1, \ldots, t_n$ are fresh distinct successors of s.
$\approx$-rule	If 1. $s \approx t \in \mathcal{A}$ (the equality can possibly be annotated), and 2. $s \neq t$ then $\mathcal{A}_1 := \mathsf{merge}_{\mathcal{A}}(s \to t)$ if t is a named individual, or t is a root individual and s is not a named individual, or s is a descendant of t; $\mathcal{A}_1 := \mathsf{merge}_{\mathcal{A}}(t \to s)$ otherwise.
$\bot$-rule	If $s \not\approx s \in \mathcal{A}$ or $\{A(s), \neg A(s)\} \subseteq \mathcal{A}$ then $\mathcal{A}_1 := \mathcal{A} \cup \{\bot\}$.
NI-rule	If 1. $s \approx t\, @^u_{\leq n\, R.B} \in \mathcal{A}$ or $t \approx s\, @^u_{\leq n\, R.B} \in \mathcal{A}$, 2. u is a root individual, 3. s is a blockable nonsuccessor of u, and 4. t is a blockable individual then $\mathcal{A}_i := \mathsf{merge}_{\mathcal{A}}(s \to \mathsf{can}_{\mathcal{A}}(u.\langle R, B, i\rangle))$ for each $1 \leq i \leq n$.

the rule introduces a fresh individual t and derives $R(a,t)$ and $C(t)$. The $\approx$-rule deals with equality: given $a \approx b$, the rule replaces the individual a in all assertions with the individual b, and it introduces a *renaming* $a \mapsto b$ in order to keep track of the merging. We take $\approx$ to have built-in symmetry; thus, $a \approx b$ should also be read as $b \approx a$. The $\bot$-rule detects obvious contradictions such as $A(a)$ and $\neg A(a)$, or $a \not\approx a$.

The *NI*-rule uses the provenance information on annotated equalities introduced by applications of the *Hyp*-rule to identify blockable individuals which must be replaced with root individuals due to at-most restrictions. Note that the *NI*-rule is never applied to an annotated equality of the form $s \approx t\, @^u_{\leq n\, R.B}$ if u is not a root individual, so such an equality can be eagerly simplified into $s \approx t$. The *NI*-rule, however, must be applied to $s \approx t\, @^u_{\leq n\, R.B}$ even if $s = t$; hence, such an equality must be derived even though it is a logical tautology. Finally, if $\mathcal{C}$ is the translation of a DL knowledge base that does not use nominals, inverse roles, and number restrictions, then the precondition of the *NI*-rule will never be satisfied, so we need not keep track of annotations at all.

Rule Priority The *NI*-rule is given higher precedence than other rules: the $\approx$-rule can be applied to a (possibly annotated) equality $s \approx t$ in an ABox $\mathcal{A}$ only if $\mathcal{A}$ does not contain an equality $s \approx t\, @^u_{\leq n\, R.B}$ to which the *NI*-rule is applicable.

Derivations For a set of HT-clauses $\mathcal{C}$ and an ABox $\mathcal{A}$, a *derivation* is a pair (T, λ) where T is a finitely branching tree and λ is a function that labels the nodes of T with ABoxes such that, for each node $t \in T$,

- $\lambda(t) = \mathcal{A}$ if t is the root of T,
- t is a leaf of T if $\bot \in \lambda(t)$ or no derivation rule is applicable to $\lambda(t)$ and $\mathcal{C}$, and
- t has children $t_1, \ldots, t_n$ such that $\lambda(t_1), \ldots, \lambda(t_n)$ are exactly the results of applying one (arbitrarily chosen, but respecting the rule precedence) applicable rule to $\lambda(t)$ and $\mathcal{C}$ otherwise.

Theorem 1. *Checking whether a $\mathcal{SHOIQ}$ knowledge base $\mathcal{K}$ is satisfiable can be performed by computing the translation of $\mathcal{K}$ into a set of HT-clauses $\mathcal{C}$ and an ABox $\mathcal{A}$, and then checking whether some derivation for $\mathcal{C} \cup \mathcal{A}$ contains a leaf node labeled with a clash-free ABox. Furthermore, such an algorithm can be implemented in* 2NExpTime *in* $|\mathcal{K}|$.

5 Conclusion

In this paper, we analyzed the problems arising in tableau calculi due to an interaction between nominals, inverse roles, and number restrictions. We also presented a new hypertableau calculus for $\mathcal{SHOIQ}$, which we implemented in our reasoner HermiT. As we report in [8], our reasoner seems to perform well on many practical problems.

References

1. P. Baumgartner, U. Furbach, and I. Niemelä. Hyper Tableaux. In *Proc. JELIA '96*, pages 1–17, Évora, Portugal, September 30–October 3 1996.
2. V. Haarslev and R. Möller. RACER System Description. In *Proc. IJCAR 2001*, pages 701–706, Siena, Italy, June 18–23 2001.
3. I. Horrocks and U. Sattler. Ontology Reasoning in the $\mathcal{SHOQ}(\mathbf{D})$ Description Logic. In *Proc. IJCAI 2001*, pages 199–204, 2001.
4. I. Horrocks and U. Sattler. A Tableaux Decision Procedure for $\mathcal{SHOIQ}$. In *Proc. IJCAI 2005*, pages 448–453, Edinburgh, UK, July 30–August 5 2005.
5. I. Horrocks, U. Sattler, and S. Tobies. Reasoning with Individuals for the Description Logic $\mathcal{SHIQ}$. In *Proc. CADE-17*, pages 482–496, Pittsburgh, USA, 2000.
6. B. Motik, R. Shearer, and I. Horrocks. A Hypertableau Calculus for SHIQ. In *Proc. of the 2007 Description Logic Workshop (DL 2007)*, pages 419–426, 2007.
7. B. Motik, R. Shearer, and I. Horrocks. Optimized Reasoning in Description Logics using Hypertableaux. In *Proc. CADE-21*, volume 4603 of *LNAI*, pages 67–83, 2007.
8. B. Motik, R. Shearer, and I. Horrocks. Hypertableau Reasoning for Description Logics. Technical report, University of Oxford, 2008. Submitted to an international journal. `http://web.comlab.ox.ac.uk/oucl/work/rob.shearer/HtShoiq.pdf`.
9. B. Parsia and E. Sirin. Pellet: An OWL-DL Reasoner. Poster, In Proc. ISWC 2004, Hiroshima, Japan, November 7–11, 2004.
10. D. Tsarkov and I. Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Proc. IJCAR 2006*, pages 292–297, Seattle, WA, USA, 2006.