

Runir's Description Logic Feature Syntax and Semantics

Dominik Drexler
dominik.drexler@liu.se

Version 0.1.0

1 Used Symbols

Symbol	Meaning
$C1, R1, \dots$	concrete-syntax placeholders for concept, role, Boolean, numerical, and query operands
$\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$	relational structure with finite, possibly empty object universe $\Delta = \Delta^{\mathcal{I}}$
s	current planning state, represented as the set of true atoms
γ	goal literal set, containing positive and negative goal atoms
a, b, c, d	object variables; in object constructors and value selection, a, a_i denote individual names interpreted by $\mathcal{I}$
P, G	unary or binary predicate symbols, depending on whether they occur as concepts or roles
p	nullary predicate symbol
H	predicate symbol of any arity $\text{arity}(H) = k \geq 0$, used in relational queries
Q, Q_1, Q_2	query expressions with finite relations as denotations, ordered named columns, and set semantics
x, y, x_i	unquoted column labels
$\bar{u}, \bar{v}, \bar{x}, \bar{y}, \bar{w}$	ordered lists of column labels; $\text{cols}(Q) = \bar{u}$ is the schema of Q
$t, \bar{d}$	tuples of objects from Δ
$t_{\bar{u}}[x], t_{\bar{u}}[\bar{v}]$	the value at label x and the tuple of values at the ordered labels $\bar{v}$, using schema $\bar{u}$
A, C, D	concepts, with denotations in $2^{\Delta^{\mathcal{I}}}$; A is atomic
R, S	roles, with denotations in $2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}}$
B_i, N_i	Boolean and numerical expressions, respectively
e, e_1, e_2	comparison operands of the same category: Boolean or numerical
b	Boolean constant, $b \in \{\text{true}, \text{false}\}$
n, k, m	nonnegative integers in $\mathbb{N} = \{0, 1, 2, \dots\}$, used as bounds, arities, list lengths, or numerical constants
$(E)^{\mathcal{I}}$	denotation of expression E in $\mathcal{I}$ under the current state, goal, and applicable register context

Constructor symbols are defined in the corresponding tables.

2 Preliminaries

Let $\mathcal{I}$ be an interpretation with finite, possibly empty domain $\Delta^{\mathcal{I}}$. Each atomic concept A is interpreted as a set $(A)^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, each atomic role R as a binary relation $(R)^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and each individual name a as an element $(a)^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

Concept descriptions denote subsets of $\Delta^{\mathcal{I}}$. Role descriptions denote binary relations over $\Delta^{\mathcal{I}}$. For planning states, let s denote the set of atoms true in the current state, and let γ denote the set of goal literals.

3 Concept Constructors

Name	Syntax	Semantics
Top	$\top$	$(\top)^{\mathcal{I}} = \Delta^{\mathcal{I}}$
Bottom	$\perp$	$(\perp)^{\mathcal{I}} = \emptyset$
Atomic state concept	P	$(P)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid P(a) \in s\}$
Positive atomic goal concept	G^+	$(G^+)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid G(a) \in \gamma\}$
Negative atomic goal concept	G^-	$(G^-)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \neg G(a) \in \gamma\}$
Intersection	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}} = (C)^{\mathcal{I}} \cap (D)^{\mathcal{I}}$
Union	$C \sqcup D$	$(C \sqcup D)^{\mathcal{I}} = (C)^{\mathcal{I}} \cup (D)^{\mathcal{I}}$
Negation	$\neg C$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus (C)^{\mathcal{I}}$
Value restriction	$\forall R.C$	$(\forall R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b. (a, b) \in (R)^{\mathcal{I}} \Rightarrow b \in (C)^{\mathcal{I}}\}$
Existential quantification	$\exists R.C$	$(\exists R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b. (a, b) \in (R)^{\mathcal{I}} \wedge b \in (C)^{\mathcal{I}}\}$
At-least number restriction	$\geq n R$	$(\geq n R)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}}\} \geq n\}$
At-most number restriction	$\leq n R$	$(\leq n R)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}}\} \leq n\}$
Exact number restriction	$= n R$	$(= n R)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}}\} = n\}$
Qualified at-least number restriction	$\geq n R.C$	$(\geq n R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}} \wedge b \in (C)^{\mathcal{I}}\} \geq n\}$
Qualified at-most number restriction	$\leq n R.C$	$(\leq n R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}} \wedge b \in (C)^{\mathcal{I}}\} \leq n\}$
Qualified exact number restriction	$= n R.C$	$(= n R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}} \wedge b \in (C)^{\mathcal{I}}\} = n\}$
Role-value map inclusion	$R \subseteq S$	$(R \subseteq S)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b. (a, b) \in (R)^{\mathcal{I}} \Rightarrow (a, b) \in (S)^{\mathcal{I}}\}$
Agreement	$R \doteq S$	$(R \doteq S)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b. (a, b) \in (R)^{\mathcal{I}} \Leftrightarrow (a, b) \in (S)^{\mathcal{I}}\}$
Role fillers	$R : \{a_1, \dots, a_k\}$	$(R : \{a_1, \dots, a_k\})^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \{(a_1)^{\mathcal{I}}, \dots, (a_k)^{\mathcal{I}}\} \subseteq \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}}\}\}$
One-of	$\{a_1, \dots, a_k\}$	$(\{a_1, \dots, a_k\})^{\mathcal{I}} = \{(a_1)^{\mathcal{I}}, \dots, (a_k)^{\mathcal{I}}\}$
Nominal	$\{a\}$	$(\{a\})^{\mathcal{I}} = \{(a)^{\mathcal{I}}\}$

Here $n \in \mathbb{N}$. In role fillers and one-of constructors, the listed object names are interpreted as individual names in $\mathcal{I}$.

4 Concrete Syntax for Concept Constructors

Name	Concrete syntax	Abstract syntax
Top	<code>c_top</code>	$\top$
Bottom	<code>c_bot</code>	$\perp$
Atomic state concept	<code>(c_atomic_state "p")</code>	P
Positive atomic goal concept	<code>(c_atomic_goal "p" true)</code>	G^+
Negative atomic goal concept	<code>(c_atomic_goal "p" false)</code>	G^-
Intersection	<code>(c_and C1 ... Cn)</code>	$C_1 \sqcap \dots \sqcap C_n$
Union	<code>(c_or C1 ... Cn)</code>	$C_1 \sqcup \dots \sqcup C_n$
Negation	<code>(c_not C)</code>	$\neg C$
Value restriction	<code>(c_all R C)</code>	$\forall R.C$
Existential quantification	<code>(c_some R C)</code>	$\exists R.C$
At-least number restriction	<code>(c_at_least n R)</code>	$\geq n R$
At-most number restriction	<code>(c_at_most n R)</code>	$\leq n R$
Exact number restriction	<code>(c_exactly n R)</code>	$= n R$
Qualified at-least restriction	<code>(c_at_least n R C)</code>	$\geq n R.C$
Qualified at-most restriction	<code>(c_at_most n R C)</code>	$\leq n R.C$
Qualified exact restriction	<code>(c_exactly n R C)</code>	$= n R.C$
Same-as / agreement	<code>(c_same_as R1 R2)</code>	$R_1 \doteq R_2$
Role-value map	<code>(c_subset R1 R2)</code>	$R_1 \subseteq R_2$
Role fillers	<code>(c_fillers R a1 ... an)</code>	$R : \{a_1, \dots, a_n\}$
One-of	<code>(c_one_of a1 ... an)</code>	$\{a_1, \dots, a_n\}$
Nominal	<code>(c_nominal a)</code>	$\{a\}$

5 Role Constructors

Name	Syntax	Semantics
Universal role	U	$(U)^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
Atomic state role	P	$(P)^{\mathcal{I}} = \{(a, b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid P(a, b) \in s\}$
Positive atomic goal role	G^+	$(G^+)^{\mathcal{I}} = \{(a, b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid G(a, b) \in \gamma\}$
Negative atomic goal role	G^-	$(G^-)^{\mathcal{I}} = \{(a, b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \neg G(a, b) \in \gamma\}$
Intersection	$R \sqcap S$	$(R \sqcap S)^{\mathcal{I}} = (R)^{\mathcal{I}} \cap (S)^{\mathcal{I}}$

Name	Syntax	Semantics
Union	$R \sqcup S$	$(R \sqcup S)^{\mathcal{I}} = (R)^{\mathcal{I}} \cup (S)^{\mathcal{I}}$
Complement	$\neg R$	$(\neg R)^{\mathcal{I}} = (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus (R)^{\mathcal{I}}$
Inverse	R^{-}	$(R^{-})^{\mathcal{I}} = \{(b, a) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (a, b) \in (R)^{\mathcal{I}}\}$
Composition	$R \circ S$	$(R \circ S)^{\mathcal{I}} = \{(a, c) \mid \exists b. (a, b) \in (R)^{\mathcal{I}} \wedge (b, c) \in (S)^{\mathcal{I}}\}$
Transitive closure	R^{+}	$(R^{+})^{\mathcal{I}} = \bigcup_{n \geq 1} ((R)^{\mathcal{I}})^n$
Reflexive-transitive closure	R^{*}	$(R^{*})^{\mathcal{I}} = \bigcup_{n \geq 0} ((R)^{\mathcal{I}})^n$
Role restriction	$R C$	$(R C)^{\mathcal{I}} = (R)^{\mathcal{I}} \cap (\Delta^{\mathcal{I}} \times (C)^{\mathcal{I}})$
Identity	$\text{id}(C)$	$(\text{id}(C))^{\mathcal{I}} = \{(d, d) \mid d \in (C)^{\mathcal{I}}\}$

The iterated composition $((R)^{\mathcal{I}})^n$ is defined by

$$((R)^{\mathcal{I}})^0 = \{(d, d) \mid d \in \Delta^{\mathcal{I}}\}, \quad ((R)^{\mathcal{I}})^{n+1} = ((R)^{\mathcal{I}})^n \circ (R)^{\mathcal{I}}.$$

6 Concrete Syntax for Role Constructors

Name	Concrete syntax	Abstract syntax
Universal role	<code>r_universal</code>	U
Atomic state role	<code>(r_atomic_state "p")</code>	P
Positive atomic goal role	<code>(r_atomic_goal "p" true)</code>	G^{+}
Negative atomic goal role	<code>(r_atomic_goal "p" false)</code>	G^{-}
Intersection	<code>(r_and R1 ... Rn)</code>	$R_1 \sqcap \dots \sqcap R_n$
Union	<code>(r_or R1 ... Rn)</code>	$R_1 \sqcup \dots \sqcup R_n$
Complement	<code>(r_complement R)</code>	$\neg R$
Inverse	<code>(r_inverse R)</code>	R^{-}
Composition	<code>(r_composition R1 ... Rn)</code>	$R_1 \circ \dots \circ R_n$
Transitive closure	<code>(r_transitive_closure R)</code>	R^{+}
Reflexive-transitive closure	<code>(r_reflexive_transitive_closure R)</code>	R^{*}
Role restriction	<code>(r_restriction R C)</code>	$R C$
Identity	<code>(r_identity C)</code>	$\text{id}(C)$

7 Boolean Constructors

Name	Syntax	Semantics
Positive atomic state predicate	$\text{holds}(p)$	true iff there exists an atom $p() \in s$
Negative atomic state predicate	$\neg\text{holds}(p)$	true iff no current atom with predicate p is true in the state
Positive atomic goal predicate	$\text{goal}(p)$	true iff $p() \in \gamma$
Negative atomic goal predicate	$\neg\text{goal}(p)$	true iff $\neg p() \in \gamma$
Non-empty concept	$C \neq \emptyset$	true iff $(C)^{\mathcal{I}} \neq \emptyset$
Non-empty role	$R \neq \emptyset$	true iff $(R)^{\mathcal{I}} \neq \emptyset$

8 Concrete Syntax for Boolean Constructors

Name	Concrete syntax	Abstract syntax
Positive atomic state predicate	<code>(b_atomic_state "p" true)</code>	$\text{holds}(p)$
Negative atomic state predicate	<code>(b_atomic_state "p" false)</code>	$\neg\text{holds}(p)$
Positive atomic goal predicate	<code>(b_atomic_goal "p" true)</code>	$\text{goal}(p)$
Negative atomic goal predicate	<code>(b_atomic_goal "p" false)</code>	$\neg\text{goal}(p)$
Non-empty concept	<code>(b_nonempty C)</code>	$C \neq \emptyset$
Non-empty role	<code>(b_nonempty R)</code>	$R \neq \emptyset$

9 Numerical Constructors

Name	Syntax	Semantics
Concept count	$ C $	$ (C)^{\mathcal{I}} $
Role count	$ R $	$ (R)^{\mathcal{I}} $
Role distance	$d_R(C, D)$	$\min\{n \in \mathbb{N} \mid (C)^{\mathcal{I}} \times (D)^{\mathcal{I}} \cap ((R)^{\mathcal{I}})^n \neq \emptyset\}$, or ∞ if empty

10 Concrete Syntax for Numerical Constructors

Name	Concrete syntax	Abstract syntax
Concept count	<code>(n_count C)</code>	$ C $
Role count	<code>(n_count R)</code>	$ R $

Name	Concrete syntax	Abstract syntax
Role distance	(n_distance C R D)	$d_R(C, D)$

11 Relational Query Expressions

Queries are operands of features in the base, extended, and unsolvability families.

Fix the state, goal, and register context, with finite universe $\Delta = \Delta^{\mathcal{I}}$. A query Q has an ordered schema and denotation

$$\text{cols}(Q) = \bar{u} = (u_1, \dots, u_k), \quad k \geq 0, \quad u_i \neq u_j \ (1 \leq i < j \leq k), \quad (Q)^{\mathcal{I}} \subseteq \Delta^k.$$

For $t = (d_1, \dots, d_k) \in \Delta^k$ and a list $\bar{v} = (v_1, \dots, v_m)$ of labels from $\bar{u}$, $m \geq 0$, define

$$t_{\bar{u}}[u_i] = d_i \ (1 \leq i \leq k), \quad t_{\bar{u}}[\bar{v}] = (t_{\bar{u}}[v_1], \dots, t_{\bar{u}}[v_m]).$$

Write $\bar{x} = (x_1, \dots, x_m)$, $\bar{y} = (y_1, \dots, y_k)$, $\bar{u}_i = \text{cols}(Q_i)$, and $\bar{w} = \bar{u}_1 \cdot (\bar{u}_2 \setminus \bar{u}_1)$. Here $\cdot$ concatenates lists; list difference deletes matching labels without reordering. The semantics column gives the denotation of each expression; $\text{arity}(H) = k$.

Name	Syntax	Semantics
Atomic state query	$\rho_{\bar{x}}(H_s)$	$\{\bar{d} \in \Delta^k \mid H(\bar{d}) \in s\}$
Positive atomic goal query	$\rho_{\bar{x}}(H_g^+)$	$\{\bar{d} \in \Delta^k \mid H(\bar{d}) \in \gamma\}$
Negative atomic goal query	$\rho_{\bar{x}}(H_g^-)$	$\{\bar{d} \in \Delta^k \mid \neg H(\bar{d}) \in \gamma\}$
Concept lift	$\rho_{(x)}(C)$	$\{(d) \mid d \in (C)^{\mathcal{I}}\}$
Role lift	$\rho_{(x,y)}(R)$	$(R)^{\mathcal{I}}$
Natural join	$Q_1 \bowtie Q_2$	$\{t \in \Delta^{ \bar{w} } \mid t_{\bar{w}}[\bar{u}_1] \in (Q_1)^{\mathcal{I}} \wedge t_{\bar{w}}[\bar{u}_2] \in (Q_2)^{\mathcal{I}}\}$
Projection	$\pi_{\bar{x}}(Q)$	$\{t_{\bar{u}}[\bar{x}] \mid t \in (Q)^{\mathcal{I}}\}$
Column renaming	$\rho_{\bar{y}}(Q)$	$(Q)^{\mathcal{I}}$
Equality selection	$\sigma_{x=y}(Q)$	$\{t \in (Q)^{\mathcal{I}} \mid t_{\bar{u}}[x] = t_{\bar{u}}[y]\}$
Value selection	$\sigma_{x=a}(Q)$	$\{t \in (Q)^{\mathcal{I}} \mid t_{\bar{u}}[x] = (a)^{\mathcal{I}}\}$
Union	$Q_1 \cup Q_2$	$(Q_1)^{\mathcal{I}} \cup (Q_2)^{\mathcal{I}}$
Difference	$Q_1 \setminus Q_2$	$(Q_1)^{\mathcal{I}} \setminus (Q_2)^{\mathcal{I}}$
Concept projection	$\pi_x^C(Q)$	$\{t_{\bar{u}}[x] \mid t \in (Q)^{\mathcal{I}}\}, \quad x \in \{u_1, \dots, u_k\}$
Role projection	$\pi_{x,y}^R(Q)$	$\{(t_{\bar{u}}[x], t_{\bar{u}}[y]) \mid t \in (Q)^{\mathcal{I}}\}, \quad x, y \in \{u_1, \dots, u_k\}, \ x \neq y$
Non-empty query	$Q \neq \emptyset$	true iff $(Q)^{\mathcal{I}} \neq \emptyset$
Query count	$ Q $	$ (Q)^{\mathcal{I}} $

Schemas and side conditions. Column labels are unquoted identifiers matching `[A-Za-z_][A-Za-z0-9_-]*`; every schema has distinct labels.

$$\begin{array}{ll}
\text{cols}(\rho_{\bar{x}}(H_s)) = \text{cols}(\rho_{\bar{x}}(H_g^\pm)) = \bar{x}, & |\bar{x}| = \text{arity}(H), \\
\text{cols}(\rho_{(x)}(C)) = (x), & \text{cols}(\rho_{(x,y)}(R)) = (x, y), \quad x \neq y, \\
\text{cols}(Q_1 \bowtie Q_2) = \bar{w}, & \\
\text{cols}(\pi_{\bar{x}}(Q)) = \bar{x}, & \{x_1, \dots, x_m\} \subseteq \{u_1, \dots, u_k\}, \\
\text{cols}(\rho_{\bar{y}}(Q)) = \bar{y}, & |\bar{y}| = |\bar{u}|, \\
\text{cols}(\sigma_{x=y}(Q)) = \bar{u}, & x, y \in \{u_1, \dots, u_k\}, \\
\text{cols}(\sigma_{x=a}(Q)) = \bar{u}, & x \in \{u_1, \dots, u_k\}, \quad a \text{ a declared domain constant}, \\
\text{cols}(Q_1 \cup Q_2) = \text{cols}(Q_1 \setminus Q_2) = \bar{u}_1, & \bar{u}_1 = \bar{u}_2.
\end{array}$$

For the empty schema,

$$\Delta^0 = \{()\}, \quad t_{\bar{u}}[()] = (), \quad (\pi_{()}(Q))^{\mathcal{I}} = \begin{cases} \{()\}, & (Q)^{\mathcal{I}} \neq \emptyset, \\ \emptyset, & (Q)^{\mathcal{I}} = \emptyset. \end{cases}$$

12 Concrete Syntax for Relational Query Constructors

Name	Concrete syntax	Abstract Syntax
Atomic state query	(q_atomic_state "H" (x1 ... xk))	$\rho_{\bar{x}}(H_s)$
Positive atomic goal query	(q_atomic_goal "H" true (x1 ... xk))	$\rho_{\bar{x}}(H_g^+)$
Negative atomic goal query	(q_atomic_goal "H" false (x1 ... xk))	$\rho_{\bar{x}}(H_g^-)$
Concept lift	(q_concept x C)	$\rho_{(x)}(C)$
Role lift	(q_role (x y) R)	$\rho_{(x,y)}(R)$
Natural join	(q_join Q1 Q2)	$Q_1 \bowtie Q_2$
Projection	(q_project (x1 ... xm) Q)	$\pi_{\bar{x}}(Q)$
Column renaming	(q_rename (y1 ... yk) Q)	$\rho_{\bar{y}}(Q)$
Equality selection	(q_select_equal x y Q)	$\sigma_{x=y}(Q)$
Value selection	(q_select_value x "a" Q)	$\sigma_{x=a}(Q)$
Union	(q_union Q1 Q2)	$Q_1 \cup Q_2$
Difference	(q_difference Q1 Q2)	$Q_1 \setminus Q_2$
Concept projection	(c_project x Q)	$\pi_x^C(Q)$
Role projection	(r_project x y Q)	$\pi_{x,y}^R(Q)$
Non-empty query	(b_nonempty Q)	$Q \neq \emptyset$
Query count	(n_count Q)	$ Q $

For the displayed constructors, syntactic complexity is

$$\text{size}(\text{op}(E_1, \dots, E_m; \alpha)) = 1 + \sum_{i=1}^m \text{size}(E_i),$$

where E_i are expression operands and α collects non-expression arguments.

13 Unsolvability Family Constructors

The unsolvability family (**uns**) extends the base family with comparison operators and constants. A comparison takes two operands of the same category—either two Booleans or two Numericals—and yields a Boolean. Boolean operands are interpreted numerically as 0 (false) or 1 (true), so the same relational operators apply uniformly to both operand categories. We write $(e)^I$ for the numeric value of an operand e (a count/distance for Numericals, or 0/1 for Booleans).

Name	Syntax	Semantics
Equal	$e_1 = e_2$	true iff $(e_1)^I = (e_2)^I$
Not equal	$e_1 \neq e_2$	true iff $(e_1)^I \neq (e_2)^I$
Less than	$e_1 < e_2$	true iff $(e_1)^I < (e_2)^I$
Less or equal	$e_1 \leq e_2$	true iff $(e_1)^I \leq (e_2)^I$
Greater than	$e_1 > e_2$	true iff $(e_1)^I > (e_2)^I$
Greater or equal	$e_1 \geq e_2$	true iff $(e_1)^I \geq (e_2)^I$
Boolean constant	$b \in \{\text{true}, \text{false}\}$	the constant truth value b
Numerical constant	$k \in \mathbb{N}$	the constant nonnegative integer k

14 Concrete Syntax for Unsolvability Family Constructors

Name	Concrete syntax	Abstract syntax
Boolean equal	(b_eq B1 B2)	$B_1 = B_2$
Boolean not equal	(b_neq B1 B2)	$B_1 \neq B_2$
Boolean less than	(b_lt B1 B2)	$B_1 < B_2$
Boolean less or equal	(b_le B1 B2)	$B_1 \leq B_2$
Boolean greater than	(b_gt B1 B2)	$B_1 > B_2$
Boolean greater or equal	(b_ge B1 B2)	$B_1 \geq B_2$
Numerical equal	(n_eq N1 N2)	$N_1 = N_2$
Numerical not equal	(n_neq N1 N2)	$N_1 \neq N_2$
Numerical less than	(n_lt N1 N2)	$N_1 < N_2$
Numerical less or equal	(n_le N1 N2)	$N_1 \leq N_2$
Numerical greater than	(n_gt N1 N2)	$N_1 > N_2$
Numerical greater or equal	(n_ge N1 N2)	$N_1 \geq N_2$
Boolean constant	(b_const true) / (b_const false)	true / false
Numerical constant	(n_const k)	k

Source

This document summarizes Appendix 1, Section A1.2 of *The Description Logic Handbook*, edited by Franz Baader, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider.

15 Numerical Arithmetic

The Ext and Uns families support the following numerical constructors. Operands take values in $\{0, \dots, M - 1\} \cup \{\infty\}$, where $M = \text{max}(\text{ygg}::\text{uint_t})$ represents infinity. Finite addition and multiplication results at least M are mapped to infinity. Subtraction saturates at zero.

Syntax	Value for operands a, b
<code>(n_const k)</code>	k
<code>(n_add A B)</code>	$a + b$; infinity if either operand is infinite
<code>(n_sub A B)</code>	$\max(a - b, 0)$; infinity if $a = \infty$
<code>(n_mul A B)</code>	ab ; zero if either operand is zero, even with infinity
<code>(n_div A B)</code>	$\lfloor a/b \rfloor$ for finite a and positive finite b
<code>(n_min A B)</code>	$\min(a, b)$
<code>(n_max A B)</code>	$\max(a, b)$

For subtraction, finite a minus infinity is zero. For multiplication, a nonzero infinite operand gives infinity. Division by zero gives infinity; otherwise an infinite numerator gives infinity, and a finite numerator divided by infinity gives zero. Thus $\infty - \infty = \infty$ and $\infty/\infty = \infty$. Boolean comparisons and Boolean arithmetic are not introduced into Ext by these constructors.

16 Choose Binding Order

A Choose body may end with `(:order (min f1) (max f2) ...)`, after its optional effects. Each term references a declared Boolean or numerical feature in the enclosing module. Load and other rule kinds do not accept this section. Let D_E be the admitted bindings after effect filtering, r the target register, and $(d_i, f_i)_{i=1}^k$ the ordering terms. Define

$$v_i(o) = f_i(s, \rho[r \leftarrow o]), \quad o \in D_E.$$

The planning state, module arguments, and other registers remain unchanged. For $o, p \in D_E$, let $j = \min\{i : v_i(o) \neq v_i(p)\}$ when this set is nonempty. Binding o precedes p exactly when

$$(d_j = \min \wedge v_j(o) < v_j(p)) \vee (d_j = \max \wedge v_j(o) > v_j(p)).$$

Booleans are ordered `false` < `true`; numerical infinity is greater than every finite value. Equal keys retain the original denotation iteration order. An omitted or empty order preserves that order for all bindings. Features are evaluated once per admitted candidate and term before sorting; empty and singleton choices need no scoring.

Ordering changes only which binding is tried first. It removes no candidate, imposes no condition on successful continuations, and leaves universal obligations and backtracking unchanged. Scoring does not count as successor generation. Time limits cover scoring and sorting as well as search.