

# Runir's Description Logic Feature Syntax and Semantics

Dominik Drexler  
dominik.drexler@liu.se

## 1 Preliminaries

Let  $\mathcal{I}$  be an interpretation with non-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  $\gamma$  denote the set of goal literals.

## 2 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	$= n R.C$	$(= n R.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid  \{b \in \Delta^{\mathcal{I}} \mid (a, b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}  = n\}$

Name	Syntax	Semantics
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  $\mathcal{I}$ .

### 3 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>	$\{a\}$

**Polarity of atomic state vs. goal constructors.** Atomic *state* concepts and roles take no polarity argument: the concrete syntax is (`c_atomic_state "p"`) and (`r_atomic_state "p"`), and a negated state concept/role is built compositionally with `c_not/r_complement`. Atomic *goal* concepts and roles carry an explicit polarity, (`c_atomic_goal "p" true`) / (`c_atomic_goal "p" false`) (and likewise for `r_atomic_goal`), because the goal condition may assert a literal positively or negatively. The boolean atomic constructors below also carry a polarity argument and require a nullary (0-arity) predicate.

## 4 Restrictions on Role Interpretations

**Functional roles.** A feature  $f$  is interpreted as a functional binary relation:

$$\forall a, b, c. (a, b) \in f^{\mathcal{I}} \wedge (a, c) \in f^{\mathcal{I}} \Rightarrow b = c.$$

Equivalently, features may be viewed as partial functions.

**Transitive roles.** A transitive role  $R$  is interpreted as a transitive binary relation:

$$\forall a, b, c. (a, b) \in R^{\mathcal{I}} \wedge (b, c) \in R^{\mathcal{I}} \Rightarrow (a, c) \in R^{\mathcal{I}}.$$

## 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	$R \sqcap S$	$R \sqcap S^{\mathcal{I}} = R^{\mathcal{I}} \cap S^{\mathcal{I}}$
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	<code>holds(<math>p</math>)</code>	true iff there exists an atom $p() \in s$
Negative atomic state predicate	<code><math>\neg</math>holds(<math>p</math>)</code>	true iff no current atom with predicate $p$ is true in the state
Positive atomic goal predicate	<code>goal(<math>p</math>)</code>	true iff $p() \in \gamma$
Negative atomic goal predicate	<code><math>\neg</math>goal(<math>p</math>)</code>	true iff $\neg p() \in \gamma$
Non-empty concept	<code><math>C \neq \emptyset</math></code>	true iff $C^{\mathcal{I}} \neq \emptyset$
Non-empty role	<code><math>R \neq \emptyset</math></code>	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>	<code>holds(<math>p</math>)</code>
Negative atomic state predicate	<code>(b_atomic_state "p" false)</code>	<code><math>\neg</math>holds(<math>p</math>)</code>

Name	Concrete syntax	Abstract syntax
Positive atomic goal predicate	(b_atomic_goal "p" true)	goal( $p$ )
Negative atomic goal predicate	(b_atomic_goal "p" false)	$\neg$ goal( $p$ )
Non-empty concept	(b_nonempty C)	$C \neq \emptyset$
Non-empty role	(b_nonempty R)	$R \neq \emptyset$

## 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 $\infty$ if empty

## 10 Concrete Syntax for Numerical Constructors

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

## 11 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  $e^{\mathcal{I}}$  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 $e_1^{\mathcal{I}} = e_2^{\mathcal{I}}$
Not equal	$e_1 \neq e_2$	true iff $e_1^{\mathcal{I}} \neq e_2^{\mathcal{I}}$
Less than	$e_1 < e_2$	true iff $e_1^{\mathcal{I}} < e_2^{\mathcal{I}}$
Less or equal	$e_1 \leq e_2$	true iff $e_1^{\mathcal{I}} \leq e_2^{\mathcal{I}}$

Name	Syntax	Semantics
Greater than	$e_1 > e_2$	true iff $e_1^{\mathcal{I}} > e_2^{\mathcal{I}}$
Greater or equal	$e_1 \geq e_2$	true iff $e_1^{\mathcal{I}} \geq e_2^{\mathcal{I}}$
Boolean constant	$b \in \{\text{true}, \text{false}\}$	the constant truth value $b$
Numerical constant	$k \in \mathbb{N}$	the constant nonnegative integer $k$

## 12 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)	$k$

## 13 Module Programs

A module program state is an extended state

$$x = (\bar{s}, m, \mu, \rho_C, \rho_R)$$

where  $\bar{s}$  is an annotated planning state,  $m$  is the current module,  $\mu$  is a typed memory state, and  $\rho_C, \rho_R$  are the concept and role register valuations. The implementation keeps memory as  $\mu \in M_{int} \uplus M_{ext}$  so internal control states and externally visible module states are not conflated.

A proof edge records the transition that changed the planning state and the rule that justified the transition. For sketch proofs the edge label is  $(e, r)$  with a state-graph edge  $e$  and a sketch rule  $r$ . For module-program proofs the corresponding label is  $(e, r_v)$  where  $r_v$  is a rule variant (load, do, sketch, or call). Edges that only expose control failure or an open state may omit  $e$  or  $r_v$  until the parser/executor rejects the corresponding ill-formed program earlier.

The intended small-step relation has the form

$$(\bar{s}, m, \mu, \rho_C, \rho_R) \xrightarrow{r_v, e} (\bar{s}', m', \mu', \rho'_C, \rho'_R).$$

Load rules update registers and internal memory, do and sketch rules may change  $\bar{s}$  through a planning transition, and call rules push the caller frame before entering the callee module. A proof succeeds when the initial extended state reaches a goal-annotated state and the constructed proof graph has no open states and no directed cycle.

## 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.