Appendix A

Definition of DGLP Syntax

A.1 BNF Rules for DGLP Syntax

For the following BNF derivation rules, terminal symbols appear surrounded by single quotes and non-terminal symbols are shown in bold face; symbols that may appear zero or one time (optional) are enclosed in square brackets ([...]) and symbols that may appear zero or more times are enclosed in curly brackets ({...}); the vertical bar (|) indicates an alternative choice and an expression of the form $\mathsf{s}_1..\mathsf{s}_n$ denotes choice between the elements of the interval from s_1 to s_n inclusive. For Table A.1 we require that for each **DescriptionGraph** the nodes are provided sequentially w.r.t. the node identifier and that the node identifiers **NI** used in the definition of each **Edge** should occur in the definition of a **Node** of the same **DescriptionGraph** ; similarly, for each **DescriptionGraphNegLabels** expression. Finally, for Table A.1 we require that $B \rightarrow H$ is an expression of the form (2.1).

UppercaseLetter	::=	'A' .. 'Z'
LowercaseLetter	::=	'a' .. 'z'
Letter	::=	**UppercaseLetter** | **LowercaseLetter**
Digit	::=	'0' .. '9'
Alphanumeric	::=	**Letter** | **Digit**
LowercaseString	::=	**LowercaseLetter**{**Alphanumeric**}
NaturalNumber	::=	**Digit**{**Digit**}
ClassName	::=	**LowercaseString**
PropertyName	::=	**LowercaseString**
NI	::=	**NaturalNumber**
Literal	::=	**ClassName** | 'NOT' **ClassName**
Ontology	::=	**DGLP** {, **DGLP** }
DGLP	::=	**Rule** | **Implies** | **ImpliedBy** | **ImpliesAndImpliedBy**
Rule	::=	'NERule' '(' $B \rightarrow H$ ')'
Implies	::=	'**begin**' **ClassName** 'SubClassOf'
		{ **ClassName** 'AND' } **Consequent**
		'**end**'
Consequent	::=	**Property** 'SOME' **DescriptionGraph**
Property	::=	**PropertyName** | 'Inverse' '(' **PropertyName** ')'
DescriptionGraph	::=	'Graph' '(' **NodeSet** **EdgeSet** ')'
NodeSet	::=	'Nodes' '(' **Node** {, **Node** } ')'
Node	::=	**NI** { **ClassName** }
EdgeSet	::=	'Edges' '(' [**Edge** {, **Edge** }] ')'
Edge	::=	**NI** **NI** **PropertyName** { **PropertyName** }

Table A.1: BNF rules for the definition of structured objects

ImpliedBy	::=	**ClassImpliedBy** | **PropertyImpliedBy**
ClassImpliedBy	::=	'begin' **ClassName** 'SuperClassOf'
		Antecedent { 'OR' **Antecedent** }
		'end'
Antecedent	::=	**ClassExpression**
		| [**ClassExpression** 'AND'] **Predicate**
ClassExpression	::=	**ClassName** { 'AND' **Literal** }
Predicate	::=	**Existential** | **CardinalityRestriction** | **Universal**
Existential	::=	**Property** 'SOME' [**Predicative**]
Predicative	::=	**ClassExpression** | **DescriptionGraphNegLabels**
DescriptionGraphNegLabels	::=	'GraphNL' '(' **NodeSetNL** **EdgeSetNL** ')'
NodeSetNL	::=	**NodesMode** '(' **NodeNL** {, **NodeNL** } ')'
NodesMode	::=	'Nodes' | 'DisjointNodes'
NodeNL	::=	**NI** { **Literal** }
EdgeSetNL	::=	'Edges' '(' [**EdgeNL** {, **EdgeNL** }] ')'
EdgeNL	::=	**NI** **NI** **PropertyName**
		{ **PropertyName** } { 'NOT' **PropertyName** }
PropertyImpliedBy	::=	'begin' **PropertyName** 'SuperPropertyOf'
		PropertyExpression { 'OR' **PropertyExpression** }
		'end'
PropertyExpression	::=	**PropertyName** { 'AND' **PropertyName** }
		{ 'AND' 'NOT' **PropertyName** }
		| **PropertyName** 'o' **PropertyName**
		{ 'o' **PropertyName** }
		| 'Inverse' '(' **PropertyName** ')'

Table A.2: BNF rules for recognition of classes and properties

NonZeroDigit	::=	'1' .. '9'
NonZeroNaturalNumber	::=	**NonZeroDigit** { **Digit** }
N	::=	**NonZeroNaturalNumber**
ZN	::=	**NaturalNumber**
CardinalityRestriction	::=	**Property** 'AT LEAST' **N** [**Predicative**]
		| **ClassName** 'AND' **Property** 'AT MOST'
		ZN [**Predicative**]
		| **Property** 'EXACTLY' **N** [**Predicative**]
Universal	::=	**ClassExpression** 'AND' **Property** 'ONLY' **Disjunction**
Disjunction	::=	'(' **Literal** { 'OR' **Literal** } ')'
ImpliesAndImpliedBy	::=	'**begin**' **ClassName** 'EquivalentTo'
		Consequent { 'AND' **ClassName** }
		'**end**'

Table A.3: BNF rules for cardinality constraints and equivalence axioms

A.2 Mappings of DGLP Axioms to Nonmonotonic Existential Rules

The mapping of the following translation rules is defined recursively, that is in certain cases the mapping of a construct relies on the mapping of its subconstruct(s), where the logical expression generated by the recursive invocation of the mapping is placed in the position of the recursive invocation. Additionally, the auxiliary operator HA is used to specify that the elements upon which HA operates shall be translated into head atoms; the operator BA is analogous for body atoms.

Element E of the BNF grammar	Logical expression NER(E)
DGLP$_1$, . . . , **DGLP**$_n$	$\{\text{NER}(\textbf{DGLP}_i)\}_{i=1}^{n}$
begin **ClassName** SubClassOf **ClassName**$_1$ AND . . . AND **ClassName**$_n$ AND **Consequent** **end**	**ClassName** (x) $\rightarrow \bigwedge_{i=1}^{n}$ **ClassName**$_i$ (x) $\wedge$ HA(**Consequent**)

Element E of the BNF grammar	Logical expression HA(E)
PropertyName SOME Graph (Nodes (**Node**$_1$, . . . , **Node**$_n$) **EdgeSet**)	$\exists_{i=1}^{n} y_i . \bigwedge_{i=1}^{n}$ **PropertyName** (x, y_i) $\wedge \bigwedge_{i=1}^{n}$HA(**Node**$_i$) $\wedge$ HA(**EdgeSet**)
Inverse(**PropertyName**) SOME Graph (Nodes (**Node**$_1$, . . . , **Node**$_n$) **EdgeSet**)	$\exists_{i=1}^{n} y_i . \bigwedge_{i=1}^{n}$ **PropertyName** (y_i, x) $\wedge \bigwedge_{i=1}^{n}$HA(**Node**$_i$) $\wedge$ HA(**EdgeSet**)
(**NI** **ClassName**$_1$ . . . **ClassName**$_n$)	$\bigwedge_{i=1}^{n}$ **ClassName**$_i$ (y_{NI})
Edges (**Edge**$_1$, . . . , **Edge**$_n$)	$\bigwedge_{i=1}^{n}$HA(**Edge**$_i$)
(**NI**1 **NI**2 **PropertyName**$_1$ . . . **PropertyName**$_n$)	$\bigwedge_{i=1}^{n}$ **PropertyName**$_i$ (y_{NI1}, y_{NI2})

Table A.4: Mappings to NER for **Implies** expressions; we have $n \geq 1$ where applicable

Element E of the BNF grammar	Logical expression NER(E)
begin **ClassName** SuperClassOf **Antecedent**$_1$ OR ... OR **Antecedent**$_\mathsf{n}$ **end**	BA(**Antecedent**$_\mathsf{i}$) $\rightarrow$ **ClassName** (x) $\mathsf{i} = 1, \ldots, \mathsf{n}$
NERule ($B \rightarrow H$)	$B \rightarrow H$

Element E of the BNF grammar	Logical expression BA(E)
ClassName$_1$ AND ... AND **ClassName**$_\mathsf{m}$ AND NOT **ClassName**$_{\mathsf{m}+1}$ ... AND NOT **ClassName**$_\mathsf{n}$	$\bigwedge_{\mathsf{i}=1}^{\mathsf{m}}$ **ClassName**$_\mathsf{i}$ (x) $\wedge$ $\bigwedge_{\mathsf{i}=\mathsf{m}+1}^{\mathsf{n}}$ **not** **ClassName**$_\mathsf{i}$ (x)
ClassExpression AND **Predicate**	BA(**ClassExpression**) $\wedge$ BA(**Predicate**)
PropertyName SOME **ClassName**$_1$ AND ... AND **ClassName**$_\mathsf{k}$ AND NOT **ClassName**$_{\mathsf{k}+1}$ ... AND NOT **ClassName**$_\ell$	**PropertyName** (x, z) $\wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{k}}$ **ClassName**$_\mathsf{i}$ (z) $\wedge$ $\bigwedge_{\mathsf{i}=\mathsf{k}+1}^{\ell}$ **not** **ClassName**$_\mathsf{i}$ (z)
Inverse(**PropertyName**) SOME **ClassName**$_1$ AND ... AND **ClassName**$_\mathsf{k}$ AND NOT **ClassName**$_{\mathsf{k}+1}$ ... AND NOT **ClassName**$_\ell$	**PropertyName** (z, x) $\wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{k}}$ **ClassName**$_\mathsf{i}$ (z) $\wedge$ $\bigwedge_{\mathsf{i}=\mathsf{k}+1}^{\ell}$ **not** **ClassName**$_\mathsf{i}$ (z)

Table A.5: Mappings to NER for **ImpliedBy** expressions; we have $\mathsf{n} \geq 1$, $\mathsf{n} \geq \mathsf{m} \geq 1$ and $\ell \geq \mathsf{k} \geq 0$ where applicable

Element E of the BNF grammar	Logical expression $\mathsf{BA}(E)$
PropertyName SOME GraphNL (Nodes (**NodeNL**$_1$, . . . , **NodeNL**$_\mathsf{n}$) **EdgeSetNL**)	$\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}$ **PropertyName** $(\mathsf{x}, \mathsf{z_i}) \wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}\mathsf{BA}($ **NodeNL$_\mathsf{i}$** $) \wedge$ $\mathsf{BA}($ **EdgeSet** $)$
Inverse(**PropertyName**) SOME GraphNL (Nodes (**NodeNL**$_1$, . . . , **NodeNL**$_\mathsf{n}$) **EdgeSetNL**)	$\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}$ **PropertyName** $(\mathsf{z_i}, \mathsf{x}) \wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}\mathsf{BA}($ **NodeNL$_\mathsf{i}$** $) \wedge$ $\mathsf{BA}($ **EdgeSetNL** $)$
PropertyName SOME GraphNL (DisjointNodes (**NodeNL**$_1$, . . . , **NodeNL**$_\mathsf{n}$) **EdgeSetNL**)	$\bigwedge_{1\leq \mathsf{i}<\mathsf{j}\leq \mathsf{n}}\ \mathsf{z_i} \neq \mathsf{z_j} \wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}$ **PropertyName** $(\mathsf{x}, \mathsf{z_i}) \wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}\mathsf{BA}($ **NodeNL$_\mathsf{i}$** $) \wedge$ $\mathsf{BA}($ **EdgeSetNL** $)$
Inverse(**PropertyName**) SOME GraphNL (DisjointNodes (**NodeNL**$_1$, . . . , **NodeNL**$_\mathsf{n}$) **EdgeSetNL**)	$\bigwedge_{1\leq \mathsf{i}<\mathsf{j}\leq \mathsf{n}}\ \mathsf{z_i} \neq \mathsf{z_j} \wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}$ **PropertyName** $(\mathsf{z_i}, \mathsf{x}) \wedge$ $\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}\mathsf{BA}($ **NodeNL$_\mathsf{i}$** $) \wedge$ $\mathsf{BA}($ **EdgeSetNL** $)$
(**NI** **ClassName**$_1$ . . . **ClassName**$_\mathsf{m}$ NOT **ClassName**$_{\mathsf{m}+1}$ . . . NOT **ClassName**$_\mathsf{n}$)	$\bigwedge_{\mathsf{i}=1}^{\mathsf{m}}$ **ClassName$_\mathsf{i}$** $(\mathsf{z_{NI}}) \wedge$ $\bigwedge_{\mathsf{i}=\mathsf{m}+1}^{\mathsf{n}}$ **not** **ClassName$_\mathsf{i}$** $(\mathsf{z_{NI}})$
Edges (**EdgeNL**$_1$, . . . , **EdgeNL**$_\mathsf{n}$)	$\bigwedge_{\mathsf{i}=1}^{\mathsf{n}}\mathsf{BA}($ **EdgeNL$_\mathsf{i}$** $)$
(**NI**1 **NI**2 **PropertyName**$_1$ . . . **PropertyName**$_\mathsf{m}$ NOT **PropertyName**$_{\mathsf{m}+1}$ . . . NOT **PropertyName**$_\mathsf{n}$)	$\bigwedge_{\mathsf{i}=1}^{\mathsf{m}}$ **PropertyName$_\mathsf{i}$** $(\mathsf{z_{NI1}}, \mathsf{z_{NI2}}) \wedge$ $\bigwedge_{\mathsf{i}=\mathsf{m}+1}^{\mathsf{n}}$ **not** **PropertyName$_\mathsf{i}$** $(\mathsf{z_{NI1}}, \mathsf{z_{NI2}})$

Table A.6: Mappings to NER for **DescriptionGraphNegLabels** expressions; we have $\mathsf{n} \geq 1$ and $\mathsf{n} \geq \mathsf{m} \geq 1$ where applicable

Element E of the BNF grammar	Logical expression NER(E)
begin **PropertyName** SuperPropertyOf **PropertyExpression**$_1$ OR ... OR **PropertyExpression**$_n$ **end**	BA(**PropertyExpression**$_i$) $\rightarrow$ **PropertyName** (x_0, x_n) $i = 1, \ldots, n$
Element E of the BNF grammar	Logical expression BA(E)
PropertyName$_1$ AND ... AND **PropertyName**$_m$ AND NOT **PropertyName**$_{m+1}$ ... AND NOT **PropertyName**$_n$	$\bigwedge_{i=1}^{m}$ **PropertyName**$_i$ $(x_0, x_n) \wedge$ $\bigwedge_{i=m+1}^{n}$ **not** **PropertyName**$_i$ (x_0, x_n)
PropertyName$_1$ o ... o **PropertyName**$_n$	$\bigwedge_{i=1}^{n}$ **PropertyName**$_i$ (x_{i-1}, x_i)
'Inverse' '(' **PropertyName** ')'	**PropertyName** (x_n, x_0)

Table A.7: Mappings to NER for **PropertyImpliedBy** expressions; we have $n \geq 1$ and $n \geq m \geq 1$ where applicable

Element E of the BNF grammar	Logical expression NER(E)
ClassExpression AND **PropertyName** ONLY (**ClassName**$_1$ OR ... **ClassName**$_m$ OR NOT **ClassName**$_{m+1}$ OR ... NOT **ClassName**$_n$)	BA (**ClassExpression**) $\wedge$ **not** **AuxClassName** (x) and also add the following rule **PropertyName** $(x, z) \wedge$ **not** **ClassName**$_1$ $(z) \wedge \ldots \wedge$ **not** **ClassName**$_m$ $(z) \wedge$ **ClassName**$_{m+1}$ $(z) \wedge \ldots \wedge$ **ClassName**$_n$ (z) $\rightarrow$ **AuxClassName** (x)

Table A.8: Mappings to NER for **Universal** expressions; we have $m \geq 0$ and $n \geq 0$ where applicable

Element E of the BNF grammar	Logical expression BA(E)
Property AT LEAST **N** **CN**$_1$ AND ... **CN**$_\mathbf{k}$ AND NOT **CN**$_{\mathbf{k}+1}$ AND ... NOT **CN**$_\ell$	BA (**Property** SOME GraphNL (DisjointNodes (1 **CN**$_1$ ... **CN**$_\mathbf{k}$ NOT **CN**$_{\mathbf{k}+1}$ ... NOT **CN**$_\ell$, ..., **N** **CN**$_1$ ... **CN**$_\mathbf{k}$ NOT **CN**$_{\mathbf{k}+1}$ ... NOT **CN**$_\ell$) Edges ( )))
Property AT LEAST **N** GraphNL (**NodesMode** (NI$_1$ **NodeLabel**$_1$, ..., NI$_\mathbf{m}$ **NodeLabel**$_\mathbf{m}$) Edges (S$_1$ D$_1$ **EdgeLabel**$_1$, ..., S$_\ell$ D$_\ell$ **EdgeLabel**$_\ell$))	BA (**Property** SOME GraphNL (DisjointNodes (**NI**$^1_1$ **NodeLabel**$_1$,..., **NI**$^1_\mathbf{m}$ **NodeLabel**$_\mathbf{m}$,..., **NI**$^\mathbf{N}_\mathbf{m}$ **NodeLabel**$_\mathbf{m}$) Edges (S$^1_1$ D$^1_1$ **EdgeLabel**$_1$,..., S$^1_\ell$ D$^1_\ell$ **EdgeLabel**$_\ell$,..., S$^\mathbf{N}_\ell$ D$^\mathbf{N}_\ell$ **EdgeLabel**$_\ell$)))
ClassName AND **Property** AT MOST **ZN** **Predicative**	BA (**ClassName** AND NOT **AuxClassName**) and also add the following rule NER (begin **AuxClassName** SuperClassOf **Property** AT LEAST **ZN** + 1 **Predicative** end)
Property EXACTLY N **Predicative**	BA (**Property** AT LEAST N **Predicative** AND NOT **AuxClassName**) and also add the following rule NER (begin **AuxClassName** SuperClassOf **Property** AT LEAST **ZN** + 1 **Predicative** end)

Table A.9: Mappings to NER for **CardinalityRestriction** expressions; we have $\mathsf{k} \geq 0$, $\mathsf{m} \geq 1$ and $\ell \geq 0$ where applicable, **AuxClassName** is a fresh **ClassName** and **ZN** + 1 is **ZN** increased by 1

Element E_{def} of the BNF grammar
begin **ClassName** EquivalentTo **PropertyName** SOME Graph (Nodes (NI_1 $\mathbf{NodeLabel}^1_1$ ... $\mathbf{NodeLabel}^1_{\mathsf{k}_1}$, ..., NI_ℓ $\mathbf{NodeLabel}^\ell_1$ ... $\mathbf{NodeLabel}^\ell_{\mathsf{k}_\ell}$) Edges (S_1 D_1 $\mathbf{EdgeLabel}^1_1$ ... $\mathbf{EdgeLabel}^1_{\mathsf{m}_1}$, ..., S_n D_n $\mathbf{EdgeLabel}^\mathsf{n}_1$ ... $\mathbf{EdgeLabel}^\mathsf{n}_{\mathsf{m}_\mathsf{n}}$)) AND $\mathbf{ClassName}_1$ AND ... AND $\mathbf{ClassName}_\mathsf{h}$ **end**

Table A.10: **ImpliesAndImpliedBy** expression; we have $\ell \geq 1$, $\mathsf{n} \geq 0$, $\mathsf{k_i} \geq 0$ for $1 \leq \mathsf{i} \leq \ell$, $\mathsf{m_i} \geq 1$ for $1 \leq \mathsf{i} \leq \mathsf{n}$ and $\mathsf{h} \geq 0$

Logical expression $\mathsf{NER}(E_{\mathsf{def}})$
$\mathbf{ClassName}\ (\mathsf{x}) \wedge \ \mathbf{not}\ \ \mathbf{RecClassName}\ (\mathsf{x})$ $\rightarrow \exists^\ell_{\mathsf{i}=1}\mathsf{y_i}.\ \bigwedge^\ell_{\mathsf{i}=1} \mathbf{PropertyName}\ (\mathsf{x}, \mathsf{y_i}) \wedge \bigwedge^\ell_{\mathsf{i}=1} \left[\bigwedge^{\mathsf{k_i}}_{\mathsf{j}=1} \mathbf{NodeLabel}^\mathsf{i}_\mathsf{j}\ (\mathsf{y_i})\right] \wedge$ $\bigwedge^\mathsf{n}_{\mathsf{i}=1} \left[\bigwedge^{\mathsf{m_i}}_{\mathsf{j}=1} \mathbf{EdgeLabel}^\mathsf{i}_\mathsf{j}\ (\mathsf{y}_{\mathsf{S_i}}, \mathsf{y}_{\mathsf{D_i}})\right] \wedge \bigwedge^\mathsf{h}_{\mathsf{i}=1} \mathbf{ClassName_i}\ (\mathsf{x}) \wedge$ $\bigwedge^\ell_{\mathsf{i}=1} \mathbf{NewClassName}\ (\mathsf{y_i})$ $\bigwedge^\ell_{i=1}\ \mathbf{PropertyName}\ (\mathsf{x}, \mathsf{z_i}) \wedge \bigwedge^\ell_{\mathsf{i}=1} \left[\bigwedge^{\mathsf{k_i}}_{\mathsf{j}=1} \mathbf{NodeLabel}^\mathsf{i}_\mathsf{j}\ (\mathsf{z_i})\right] \wedge$ $\bigwedge^\mathsf{n}_{\mathsf{i}=1} \left[\bigwedge^{\mathsf{m_i}}_{\mathsf{j}=1} \mathbf{EdgeLabel}^\mathsf{i}_\mathsf{j}\ (\mathsf{z}_{\mathsf{S_i}}, \mathsf{z}_{\mathsf{D_i}})\right] \wedge \bigwedge^\mathsf{h}_{\mathsf{i}=1} \mathbf{ClassName_i}\ (\mathsf{x}) \wedge$ $\bigwedge^\ell_{\mathsf{i}=1}\ \mathbf{not}\ \ \mathbf{NewClassName}\ (\mathsf{z_i})$ $\rightarrow\ \mathbf{ClassName}\ (\mathsf{x}) \wedge\ \mathbf{RecClassName}\ (\mathsf{x})$

Table A.11: Mappings to NER for **ImpliesAndImpliedBy** expressions; we have $\ell \geq 1$, $\mathsf{n} \geq 0$, $\mathsf{k_i} \geq 0$ for $1 \leq \mathsf{i} \leq \ell$, $\mathsf{m_i} \geq 1$ for $1 \leq \mathsf{i} \leq \mathsf{n}$ and $\mathsf{h} \geq\ 0$; **RecClassName** and **NewClassName** are fresh unary predicates unique for each **ClassName**

Element $E_{\mathsf{def-inv}}$ of the BNF grammar
begin **ClassName** EquivalentTo Inverse (**PropertyName**) SOME Graph (Nodes (NI_1 **NodeLabel**$^1_1$ ... **NodeLabel**$^1_{\mathsf{k}_1}$, ..., NI_ℓ **NodeLabel**$^\ell_1$ ... **NodeLabel**$^\ell_{\mathsf{k}_\ell}$) Edges (S_1 D_1 **EdgeLabel**$^1_1$ ... **EdgeLabel**$^1_{\mathsf{m}_1}$, ..., S_n D_n **EdgeLabel**$^\mathsf{n}_1$ ... **EdgeLabel**$^\mathsf{n}_{\mathsf{m}_\mathsf{n}}$)) AND **ClassName**$_1$ AND ... AND **ClassName**$_\mathsf{h}$ **end**

Table A.12: **ImpliesAndImpliedBy** expression with inverse property; we have $\ell \geq 1$, $\mathsf{n} \geq 0$, $\mathsf{k_i} \geq 0$ for $1 \leq \mathsf{i} \leq \ell$, $\mathsf{m_i} \geq 1$ for $1 \leq \mathsf{i} \leq \mathsf{n}$ and $\mathsf{h} \geq 0$

Logical expression $\mathsf{NER}(E_{\mathsf{def-inv}})$
ClassName (x) $\wedge$ **not** **RecClassName** (x) $\rightarrow \exists^\ell_{\mathsf{i}=1}\mathsf{y_i}.\ \bigwedge^\ell_{\mathsf{i}=1}$ **PropertyName** $(\mathsf{y_i}, \mathsf{x}) \wedge \bigwedge^\ell_{\mathsf{i}=1}\left[\bigwedge^{\mathsf{k_i}}_{\mathsf{j}=1}\right.$ **NodeLabel**$^\mathsf{i}_\mathsf{j}$ $\left.(\mathsf{y_i})\right] \wedge$ $\bigwedge^\mathsf{n}_{\mathsf{i}=1}\left[\bigwedge^{\mathsf{m_i}}_{\mathsf{j}=1}\right.$ **EdgeLabel**$^\mathsf{i}_\mathsf{j}$ $\left.(\mathsf{y_{S_i}}, \mathsf{y_{D_i}})\right] \wedge \bigwedge^\mathsf{h}_{\mathsf{i}=1}$ **ClassName**$_\mathsf{i}$ (x) $\wedge$ $\bigwedge^\ell_{\mathsf{i}=1}$ **NewClassName** $(\mathsf{y_i})$ $\bigwedge^\ell_{i=1}$ **PropertyName** $(\mathsf{z_i}, \mathsf{x}) \wedge \bigwedge^\ell_{\mathsf{i}=1}\left[\bigwedge^{\mathsf{k_i}}_{\mathsf{j}=1}\right.$ **NodeLabel**$^\mathsf{i}_\mathsf{j}$ $\left.(\mathsf{z_i})\right] \wedge$ $\bigwedge^\mathsf{n}_{\mathsf{i}=1}\left[\bigwedge^{\mathsf{m_i}}_{\mathsf{j}=1}\right.$ **EdgeLabel**$^\mathsf{i}_\mathsf{j}$ $\left.(\mathsf{z_{S_i}}, \mathsf{z_{D_i}})\right] \wedge \bigwedge^\mathsf{h}_{\mathsf{i}=1}$ **ClassName**$_\mathsf{i}$ (x) $\wedge$ $\bigwedge^\ell_{\mathsf{i}=1}$ **not** **NewClassName** $(\mathsf{z_i})$ $\rightarrow$ **ClassName** (x) $\wedge$ **RecClassName** (x)

Table A.13: Mappings to NER for **ImpliesAndImpliedBy** expressions with inverse property; we have $\ell \geq 1$, $\mathsf{n} \geq 0$, $\mathsf{k_i} \geq 0$ for $1 \leq \mathsf{i} \leq \ell$, $\mathsf{m_i} \geq 1$ for $1 \leq \mathsf{i} \leq \mathsf{n}$ and $\mathsf{h} \geq 0$; where **RecClassName** and **NewClassName** are fresh unary predicates unique for each **ClassName**