Natural Logic Knowledge Bases and Their Graph Form

Andreasen, Troels; Bulskov, Henrik; Jensen, Per Anker; Nilsson, Jørgen Fischer

Document Version
Accepted author manuscript

Published in:
Data & Knowledge Engineering

DOI:
10.1016/j.datak.2020.101848

Publication date:
2020

License
CC BY-NC-ND

Citation for published version (APA):
Andreasen, T., Bulskov, H., Jensen, P. A., & Nilsson, J. F. (2020). Natural Logic Knowledge Bases and Their Graph Form. *Data & Knowledge Engineering*, *129*, Article 101848. https://doi.org/10.1016/j.datak.2020.101848

Journal Pre-proof

Natural logic knowledge bases and their graph form

Troels Andreasen, Henrik Bulskov, Per Anker Jensen, Jørgen Fischer Nilsson

PII: S0169-023X(18)30616-5
DOI: https://doi.org/10.1016/j.datak.2020.101848
Reference: DATAK 101848

To appear in: *Data & Knowledge Engineering*

Received date : 3 December 2018
Revised date : 2 December 2019
Accepted date : 12 August 2020

Please cite this article as: T. Andreasen, H. Bulskov, P.A. Jensen et al., Natural logic knowledge bases and their graph form, *Data & Knowledge Engineering* (2020), doi: https://doi.org/10.1016/j.datak.2020.101848.

Natural Logic Knowledge Bases and their Graph Form

Troels Andreasen[1], Henrik Bulskov[1], Per Anker Jensen[2], and Jørgen Fischer Nilsson[3]

[1]Computer Science, Roskilde University, Denmark
[2]Management, Society and Communication, Copenhagen Business School, Denmark
[3]Mathematics and Computer Science, Technical University of Denmark, Denmark

December 2, 2019

Abstract

This paper describes how knowledge bases can be represented in and reasoned with in natural logic. Natural logic is a regimented fragment of natural language possessing a well-defined logical semantics. As such, natural logic may be considered an attractive alternative among the various knowledge representation logics such as description logics. Our version of natural logic expands formal ontologies with affirmative propositions expressing a variety of relationships between concepts. It comprises (nested) restrictive relative clauses and prepositional phrases and, as a new construct, adverbial prepositional phrases. The natural logic knowledge base is to be used for deductive query answering applying inference rules. This is facilitated by introduction of DATALOG as an embedding meta-logic. The inference rules are stated in DATALOG and act directly on the natural logic formulations. The knowledge base propositions are decomposed into a graph form enabling path finding between concepts. The examples in the paper are derived from text source life-science descriptions.

Keywords: Natural Logic; Knowledge management applications; Ontologies; Query; Metalogic; Bioinformatics databases.

1 Introduction

This paper describes a version of natural logic designed for use as a knowledge base language. Natural logics are forms of logic which resemble regimented fragments of natural language whilst at the same time constituting a logic with inference rules and a well-defined semantics. Natural logics should not be conceived as merely a natural language wrapping on predicate logic since the reasoning taking place for instance in querying computation is done directly in the natural language-like formulations rather than in predicate logic.

Natural logics originate in subject-predicate logic belonging to the Aristotelian tradition (Klima, 2010). This means that they are generally of limited expressivity, but they possess desirable properties of decidability and tractability similar to description logic dialects. Unlike description logic, however,natural logic renders knowledge bases readable by domain experts and furthermore provide query answers approaching the level of natural language. A crucial aspect of natural logic is the use of transitive verbs for expressing relationships between concepts, as is common in natural language descriptions. This is in contrast to the insistence on the copula form with the verb *is* corresponding to the symbol $\sqsubseteq$ in description logics.

The present paper focusses on a dialect of natural logics called NATURALOG dedicated to ontology-structured knowledge bases. We describe the available form of propositions and their representation in an instructive graph representation called 'concept graphs', not to be confused with conceptual graphs (Sowa, 1979; Sowa, 2000). The concept graphs visualize the applied logical representations and may be seen as generalizations of ontology diagrams along the lines of entity-relationship diagrams (Chen, 1976) and the more ontology focussed OntoUML (Guizzardi, 2005). The representation of complex natural logic propositions as graphs calls for decomposition of the propositions into more elementary constituents. For the various features and characteristics of NATURALOG we refer to the following past papers. In (Andreasen et al., 2014) we discuss the relationship between fragments of a bio-model and a preliminary form of NATURALOG. (Nilsson, 2015) discusses the relationship between syllogistic logic and natural logic. In (Andreasen et al., 2015) we elaborate on the concept path finding and concommitant query evaluation principles. In (Andreasen et al., 2017) we discuss semantic conservative extensions to our natural logic, such as linguistic conjunctions, appositions and parentetic relative clauses. Furthermore, we discuss the problem of extracting natural logic sentences from natural language text sources. In (Andreasen et al., 2019) we briefly introduce our encoding principle using DATALOG as metalogic, and we present a relaxed form of concept querying in addition to pathway querying.

The present natural logic approach to knowledge bases originates in the so-called generative ontologies which we advanced in (Andreasen and Nilsson, 2004). The key idea of generative ontologies is to enable increasingly specialized concepts to be formed "downwards" in an ontology in analogy to the arbitrarily complex phrases formed by recursion in a phrase structure grammar. Our main inspiration sources for extension of the generative ontologies with natural logic forms are (van Benthem, 1986; Moss, 2010; Sánchez Valencia, 1991)

The paper is structured as follows: In sections 2 and 3 we introduce natural logic knowledge bases and their graph form. The grammar of NATURALOG is given in section 4 and NATURALOG explicated in predicate logic in section 5, and its metalogic encoding conforming with the graph form is given in section 6. Sections 7 and 8 introduce reasoning and querying using NATURALOG. Finally, section 9 discusses relationships to description logic and other logics, while section 10 summarizes and concludes the paper.

2 The Gist of Knowledge Base Natural Logic

The foundation for using natural logic in the present context is the conception of the target domain as being constituted by concepts and their relationships. As such the present approach

has affinities to entity-relationship modeling (Chen, 1976). However, it is important to observe that our notion of classes is generative and thus open-ended, reflecting the recursive structure of syntactic phrases. Nevertheless, the applied natural logic has a well-defined logical semantics and inference rules serving querying purposes.

The sentences in NATURALOG has the general form

C R D

where *C* and *D* are concepts and *R* is a transitive verb or the copula isa expressing a binary relation between *C* and *D*, such as

betacell produce insulin

pancreas isa gland

The sentences are called propositions in order to distinguish from sentences in natural language. The present version of the language of NATURALOG propositions are formally defined syntactically in chapter 4 and semantically in chapter 5. As it appears, NATURALOG propositions resemble natural language sentences as seen in:

cell that produce insulin reside-in pancreas

where the subject term comprises the restrictive relative clause: that produce insulin. In NATURALOG, natural language morphology is dispensed with. In linguistic terms, NATURALOG propositions typically consist of a subject term followed by a transitive verb and a linguistic object. The concept graph corresponding to the above example is shown in figure 1.

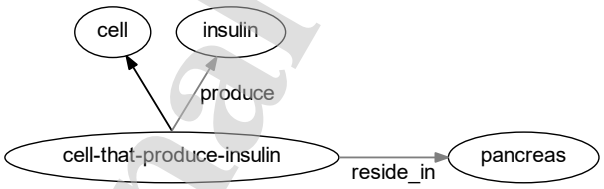


Figure 1: Graph representing the proposition: cell that produce insulin reside-in pancreas

Notice that in this graph the proposition is decomposed into three primitive propositions, namely cell-that-produce-insulin isa cell, cell-that-produce-insulin produce insulin and cell-that-produce-insulin reside-in pancreas, where the term cell-that-produce-insulin names an auxiliary concept created internally. In the graph rendition for isa-relationships, we use upwards-pointing unlabeled arcs.

2.1 The Logic of NATURALOG

From a logical point of view NATURALOG sentences are affirmative predicate-logical sentences in a subset of predicate logic coming in a distinguished variable-free form. The employed particular predicate-logical sublanguage is described in section 5.

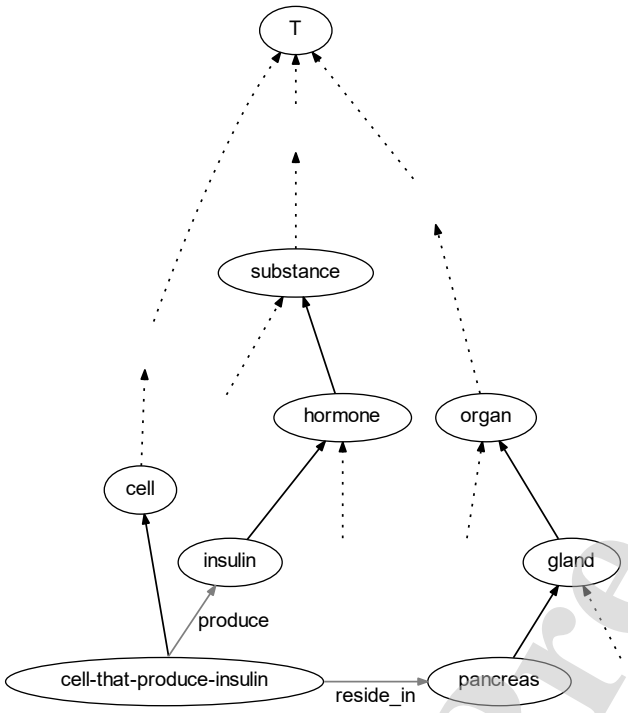


Figure 2: The proposition of figure 1 embedded in a knowledge base graph

The NATURALOG proposition pancreas isa gland, cf. figure 2, is explicated straightforwardly in predicate logic as

$$\forall x(pancreas(x) \rightarrow gland(x))$$

The predicate logical construal of the proposition in figure 1, cell that produce insulin reside-in pancreas, is:

$$\forall x(cell(x) \wedge \exists y(insulin(y) \wedge produce(x,y)) \rightarrow \exists z(pancreas(z) \wedge residein(x,z))$$

This complex proposition is handled by introduction of an auxiliary concept *cell-that-produce-insulin* giving rise to the predicate logical reformulations:

$$\forall x(cell\text{-}that\text{-}produce\text{-}insulin(x) \rightarrow \exists z(pancreas(z) \wedge residein(x,z))$$

$$\forall x(cell\text{-}that\text{-}produce\text{-}insulin(x) \leftrightarrow cell(x) \wedge \exists y(insulin(y) \wedge produce(x,y))$$

as reflected in figure 1.

We stress that the predicate-logical explication of NATURALOG sentences serves the semantical specification, only: NATURALOG propositions are encoded in a metalogic without quantified variables as to be explained chapter 6.

3 Natural Logic Graphs

A natural logic knowledge base can conveniently be comprehended as an annotated directed graph where the nodes represent concepts, and relations are depicted as directed arcs between the concept nodes. This view conforms with and further enriches the common Hasse diagram view of formal ontologies, using the partial order isa inclusion relation. Accordingly, a knowledge base graph contains an ontological skeleton isa-taxonomy augmented with various forms of relationships. This graph form relies on a decomposition of complex propositions into atomic ones. Such a decomposition calls for introduction of auxiliary concepts with accompanying nodes.

We adhere to the following three principles for representing propositions as annotated directed graphs:

- A proposition is represented as a subgraph of the entire coherent knowledge base graph. In the simplest case, a proposition is represented merely as a labeled arc connecting two concept nodes.
- In the case of complex propositions, more arcs and auxiliary concept nodes are introduced, so that all concepts as well as their auxiliary concepts are explicitly represented as concept nodes in the graph.
- Concepts are represented as unique nodes in the knowledge base graph. This means that a node for a given concept is shared by all the propositions in the knowledge base which contain that concept. Thus, usually the knowledge base is represented as one coherent graph with propositions forming intertwined subgraphs.

From the second principle it follows that all subexpressions are explicitly represented by a node as shown in figure 1. This admits that complex propositions can be reconstructed from their decomposed form of atomic relationships in the knowledge base graph. Furthermore, concept graphs support pathway computation, that is, computation of conceptual connections between two stated query concepts by way of shortest path computation in the knowledge base graph. This notion of reconstruction can also be explained with reference to figure 1, where the natural logic expression cell that produce insulin reside-in pancreas can be reconstructed by combining subexpressions from the outgoing arcs of the auxiliary node cell-that-produce-insulin.

In NATURALOG there is no distinguished empty concept. All concepts are assumed to be non-empty although we do not care about the individual member entities. By convention, then, two concepts are assumed to be disjoint when one is not a subconcept of the other, and they do not explicitly share a common – and hence necessarily non-empty – subconcept. This means that all concepts are initially assumed to be disjoint, and then possibly subsequently made overlapping by inclusion or by introduction of a joint subconcept. This convention conforms with the general implicit norm for classification hierarchies in science. However, it departs from predicate logic, including description logics, as discussed in section 9.1.

In this perspective, then, disjointness of concepts A and B is verified by provable absence of a concept C in the knowledge base such that C isa A and C isa B appealing to the closed-world assumption, cf. 7.3. Notice that this differs from the common extensional Boolean view of

concepts as sets, where there is always such a concept C, that is the intersection set of extensions of A and B. And this latter set, which is bound to exist mathematically, is possibly the empty set.On the other hand, in our setup there are no objections against positing a universal concept U in an ontology, such that all concepts in a knowledge base ontology become subconcepts of U, the class U then forming the "top" of the ontology.

In the following subsections, we consider the various natural logic constructs along with their graph forms.

3.1 Concept Inclusion in the Natural Logic

Concept inclusion is a common and important basic case of NATURALOG propositions and is expressed by the copula form used as building block for ontologies:

C isa *D*

where C and D are concept terms. The explicitly quantified form of this concept inclusion is

every *C* isa some *D*

In this subsection, we consider only simple concept terms (also called classes). In linguistic terms, they basically take the form of common nouns or multi-word names like "*islets of Langerhans*" represented conceptually in NATURALOG as islets_of_langerhans. The copula form declares that the concept C is a subconcept of the concept D as in the propositions

insulin isa hormone

and

hormone isa protein

By the rules of reflexivity, transitivity and antisymmetry, the isa relation becomes a partial order. Accordingly, this form of proposition serves the construction of ontologies in a knowledge base. In our ontology diagrams, the relationships following from reflexivity and transitivity are left implicit. In the general form of NATURALOG, the terms C and D are recursively structured compound concept terms as discussed in section 3.3.

Although hierarchical structures are common in ontologies, not least in life science domains, any partial order is admitted here. Thus, an ontological structure such as the non-hierarchical concept graph in figure 3 given by the four knowledge base propositions:

pancreas isa exocrine_gland
pancreas isa endocrine_gland
exocrine_gland isa gland
endocrine_gland isa gland

provides multiple inheritance to the class pancreas from its two immediate superior classes. Inheritance comes in when concepts are featured with properties by further propositions in the knowledge base. Actually, the above four propositions may be restated in a more succinct form

pancreas isa exocrine_gland and pancreas isa endocrine_gland
exocrine_gland and endocrine_gland isa gland

using an extension of NATURALOG with conjunctions introduced in (Andreasen et al., 2017).

This extension is a purely conservative one facilitating and shortening formulations and is therefore not discussed further in the present context. There may well be other classes sharing the non-hierarchical inheritance. This means that the partial order does not form a (semi)-lattice. However, an if-and-only-if definition of a concept in terms of two immediate superior concepts (akin to lattice infimum) can be achieved in NATURALOG. These aspects are further discussed in section 3.3.

Viewed extensionally, the inclusion relationship *C* isa *D* is understood as the subset relation. However, in this context we do not address underlying concept instances. Rather, we conceive of concepts intensionally, implying that all the properties attributed explicitly or implicitly to entities in the relatum *D* are inherited to entities in the relatum *C*, cf. e.g. (Nilsson, 2014) for a discussion of extension versus intension in modeling. Thus, the deduced knowledge base answers are to take the form of concept terms rather than some underlying extension sets as elaborated in section 7. In the current version of NATURALOG, we dispense with named particulars (signified by proper nouns). Indeed, the member entities of the various concepts remain anonymous. However, particulars may be obtained formally as singleton concepts having no proper subconcepts.

We are now going to enrich the ontological equipment with constructs enabling adornment of the ontological "skeleton" with more general knowledge base propositions.

3.2 Relationships with a Variety of Relations

We now turn from the case of inclusion propositions to propositions that express a quantified relationship, *R*, between two relata concepts *C* and *D*

C R D

where the relation *R* is drawn from a freely chosen repertoire of binary relations as in the proposition

betacell produce insulin

With explicit quantifiers the proposition form is

every *C R* some *D*

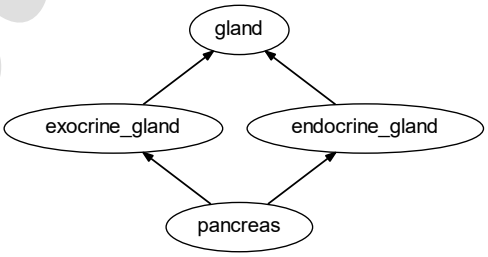


Figure 3: Non-hierarchical concept graph representing four knowledge base propositions

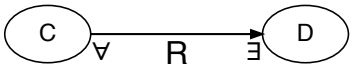


Figure 4: ∀∃ relationship with relation R

and for the sample proposition it becomes

every betacell produce some insulin

Linguistically, the relator R is usually expressed by a transitive verb (in the active or the passive voice) with the copula as a special case. The various forms with explicit quantifiers (linguistically expressed by determiners) are

{every | some} C R {every | some} D

with the quantifier option

every C R some D

being the default form when quantifiers are omitted as in the above example. The corresponding predicate logical form would be $\forall x(C(x) \rightarrow \exists y(R(x,y) \wedge D(y)))$ as elaborated in section 5. This default convention for the quantifiers mirrors the most common interpretation of natural language sentences like *betacells produce insulin.*

In the graph depiction in figure 4, the relationship arc is labelled with a relation variable R and the pair of quantifiers. Whereas the concept nodes are made unique and shared by propositions throughout the entire knowledge base graph, there may be any number of distinctly labelled relationship arcs between two nodes. The quantifier symbols can be omitted from the graph representation except when the interpretation deviates from the default convention every C R some D.

3.2.1 Dual Relationship with Inverse Relation

Mathematically, each relation R posseses an inverse relation R^{-1}. Therefore, for strictly logical reasons and given our pervasive principle of non-empty concepts, for $C\,R\,D$ (i.e. every C R some D) implicitly, as explained in chapter 5, we also have the dual

some D R^{-1} some C

Accordingly, when R is expressed by a transitive verb in the active voice form as in

every betacell produce some insulin

the dual proposition using the passive voice becomes

some insulin is produced by some betacell

Notice that the logically stronger insulin is produced by betacell, i.e. **every** insulin is produced by some betacell (understood as "all amounts of insulin ..."), does not follow from the active voice form. However, the stronger one may of course be claimed if pertinent in the knowledge base,

thereby overruling the weaker one. The general implicit presence of such dual "active/passive voice" pairs for each given proposition as shown in figure 5, besides enforced logically, is crucial for pathway query computations by providing semantically relevant both-way paths, as it were. In particular, the inclusion proposition *C* isa *D* (i.e. every *C* isa some *D*) has as consequence the dual some *D* isa *C*, since *C* is bound to be non-empty. Furthermore, from *C* isa *D* follows directly the proposition some *C* isa *D*.

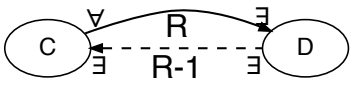


Figure 5: The relationship from figure 4 with inferred inverse ∃∃ relationship

3.3 Compound Concepts

So far we have only considered propositions with simple concepts, that is, concepts taking the form of common nouns or multi-word names like "islets of Langerhans". We now turn to compound concepts, which typically come about by adding restrictive modifiers. Linguistically, restrictive modifiers may assume a number of different forms. In this paper we focus on restrictive relative clauses and prepositional phrases (PPs).

Consider the following compound concept, which in linguistic terms corresponds to a noun phrase,

cell that produce insulin

This is to be understood as the concept cell modified by the restriction produce insulin. The occurrence in the knowledge base of the compound concept cell that produce insulin gives rise to the formation of a concept node named cell-that-produce-insulin, which is defined by two ∀∃ relationships as shown in the concept graph in figure 6. The tails of the two outlet arcs are joined

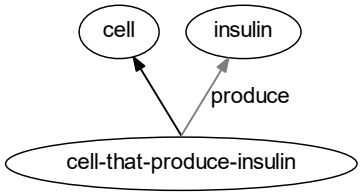


Figure 6: The graph defining the concept: cell-that-produce-insulin

in order to indicate that cell-that-produce-insulin is defined by an if-and-only-if definition. This means that for any concept term *C* such that *C* isa cell and *C* produce insulin we have *C* isa

cell-that-produce-insulin as explained in more detail in section 7.2. Apart from the outlet arcs contributing to the definition, there may of course be additional non-definitional arcs as shown in figure 1. The distinction between definitional and non-definitional outlet arcs of a node admits propositions with compound concept terms to be effectively retained in the concept graph.

In our natural logic, we treat modifiers linguistically expressed by PPs in the same way as relative clauses. For instance, the concept cell in pancreas, where cell is modified by in pancreas, gives rise to the compound concept cell-in-pancreas with the two defining relationships cell-in-pancreas isa cell and cell-in-pancreas in pancreas.

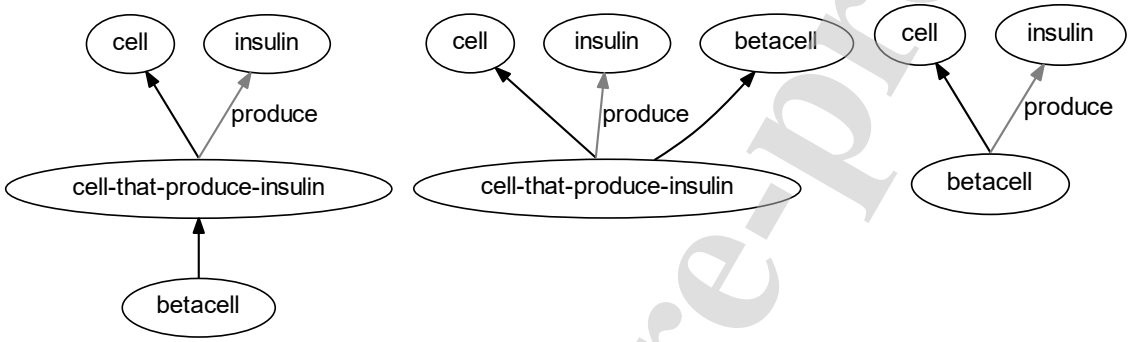


Figure 7: Introducing betacell as a synonym of cell that produce insulin: betacell syn cell that produce insulin

Consider the two propositions:

betacell isa cell that produce insulin
cell that produce insulin isa betacell

In effect, these two propositions introduce betacell as a synonym of the compound concept cell that produce insulin. For synonyms we provide the relation syn as a shorthand for the given propositions:

betacell syn cell that produce insulin

More generally, for two concepts C and D we have the convention that C syn D implies that C isa D and D isa C. Whether to use C or D as the label of the node is an arbitrary choice.

3.4 Multiple Inclusion Relations

As seen in figure 6, an auxiliary concept is typically defined by an isa relationship accompanied by a non-isa relationship, where the latter typically stems from the restrictive modifier. However, as an uncommon case, a concept definition may be formed by two isa relationships, as shown in figure 8 for the contrived case of endocrine_exocrine_gland isa endocrine_gland and endocrine_exocrine_gland isa exocrine_gland. Here the concept endocrine_exocrine_gland is defined by two isa relationships achieved by way of the proposition exocrine_gland that isa endocrine_gland isa endocrine_exocrine_gland, which is the reverse of endocrine_gland isa endocrine_exocrine_gland isa exocrine_gland, which together is provided by the relation syn in:

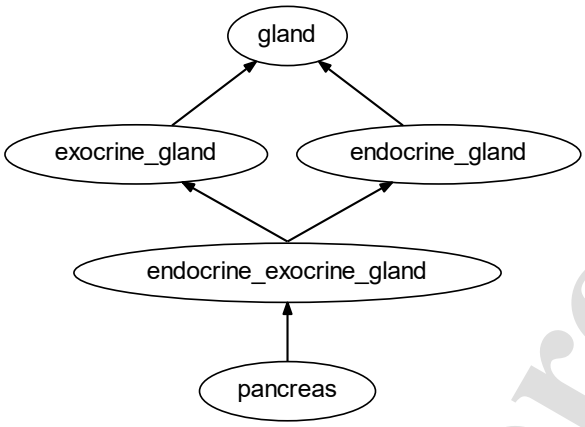


Figure 8: The uncommon case of a concept definition formed from two isa relationships

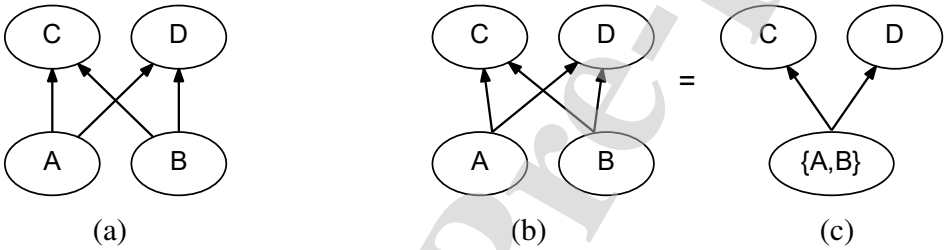


Figure 9: Subordinate concepts

endocrine_exocrine_gland syn exocrine_gland that isa endocrine_gland

following the convention illustrated in figure 7.

In figure 9(a) we have two subordinate concepts which are not defined by their superior concepts, while in figure 9(b) the two subordinate concepts have common definitions and are therefore bound to merge as indicated in figure 9(c). In figure 10(a), on the other hand, only the concept *B* is defined from C and D. This definition implies that any concept that is both a

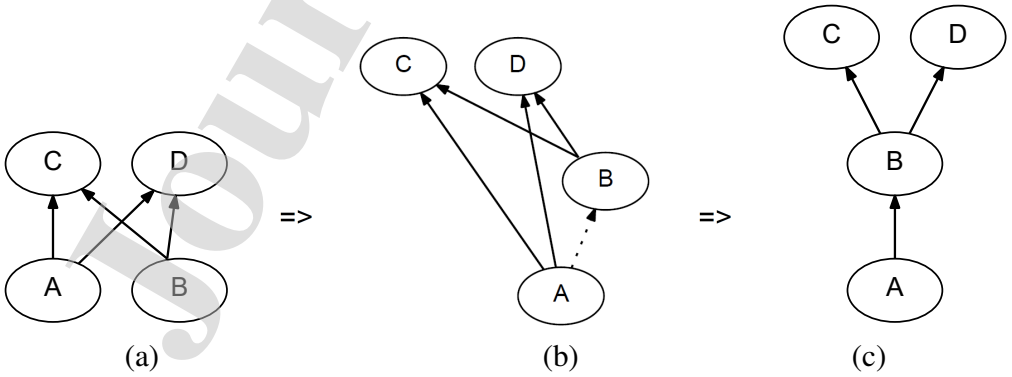


Figure 10: Inclusion inferred by subsumption with subsequent transitivity reduction

C and a D is a B as well. Thus, the dashed isa-arc in figure 10(b) showing that A isa B can be inferred. At the predicate logical level of understanding $\forall x(A(x) \rightarrow C(x) \wedge D(x))$ together with

$\forall x(B(x) \leftrightarrow C(x) \wedge D(x))$ yields $\forall x(A(x) \rightarrow B(x))$, cf. figure 10(c). This case of subsumption is to be discussed further in section 7.2.

3.5 Combination of Modifiers

Restrictive modifiers in compound concepts may be interpreted as either aligned or nested. Consider an example where a PP is followed by a restrictive relative clause, which is represented in natural logic as follows

cell in gland that produce hormone

We propose a default interpretation of this expression as aligned, cf. figure 11a. This graph shows how the compound concept cell in gland that produce hormone is broken down into the two intermediate concepts cell-in-pancreas and cell-that-produce-hormone. The latter two are then joined into cell-in-pancreas-that-produce-hormone. To enforce the nested interpretation of the same expression we use parentheses:

cell in (gland that produce hormone)

as illustrated by the graph in figure 11b. This graph shows how the compound concept cell in (gland that produce hormone) is defined by isa to the concept cell and an in-relation to the nested concept gland that produce hormone.

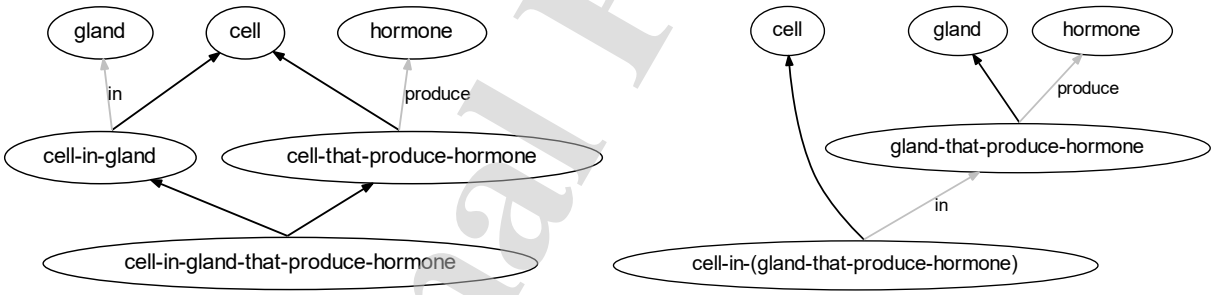


Figure 11: Alignment versus nesting of modifiers.

Recall that, according to the principle of unique representation of concepts, the generated auxiliary concept nodes are common to all the propositions containing these same concepts. In this way, the individual propositions are intertwined in the knowledge base graph, while their meaning is retained. This is illustrated in figure 12, which shows the two subgraphs in figure 11 joined.

3.6 Nominalization and Relationships with Adverbial Modifiers

All transitive verbs have a corresponding nominalized form. For instance, the English verb *produce* may be nominalized by adding *-tion* to the stem *produc-*. In natural logic, this is handled by positing a predicate **nominalization** in addition to the previously introduced predicates **definition** and **proposition**, cf.

nominalization(produce, production)

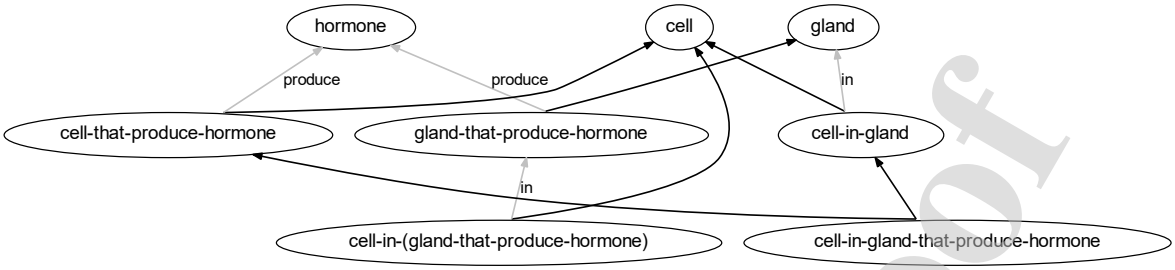


Figure 12: Unique representation of concepts.

or, more generally, **nominalization**(R, nom-R), where nom-R is the concept given by nominalization. This opens the possibility of treating linguistic constructions involving verbs modified by PP-adverbials in natural logic. For instance, assuming that the form in natural logic of a transitive verb like *produce* is R, and of a PP like *in pancreas* is (R1 C), then we can relate the verb phrase *produce in pancreas* to its nominalized counterpart *production in pancreas*

nominalization(produce-in-pancreas, production-in-pancreas),

since by nominalization of the form R(R1 C) we get the concept nom-R(R1 C), which is defined in the knowledge base graph in figure 13 corresponding to the definitions

definition(nom-R-(R1-C), isa, nom-R)
definition(nom-R-(R1-C), R1, C)

The nominalization mapping opens for accommodation of a specialization/generalisation ontology for verbs via their nominalizations. The created concept nom-R-(R1-C) is also used in the inference rules in section 7.1.

So far, the propositions we have considered, have contained simple relations and possibly compound concepts. We now turn to considering propositions containing also compound relations as shown in the English sentence *betacells produce insulin in pancreas*. In our natural logic, in order to avoid attaching the PP *in pancreas* to the noun *insulin*, we represent this sentence as the following proposition:

betacell produce in pancreas insulin

where the relation produce is restrictively modified to become produce in pancreas. By the predicate **nominalization**, we can now relate the construct produce in pancreas to the concept production in pancreas as in figure 14.

This nominalisation/verbalisation maneuvre supports the following reasoning principle for relations with restrictive modifiers: The dropping of a restrictive modifier as well as the conceptual generalization of the concept within the PP modifier weakens the proposition. Thus, the proposition betacell produce insulin in endocrine_gland weakens the given betacell produce insulin in pancreas, given further that pancreas isa endocrine_gland. The predicate **nominalization** is also used in the socalled aboutness querying facility described in section 8.4.

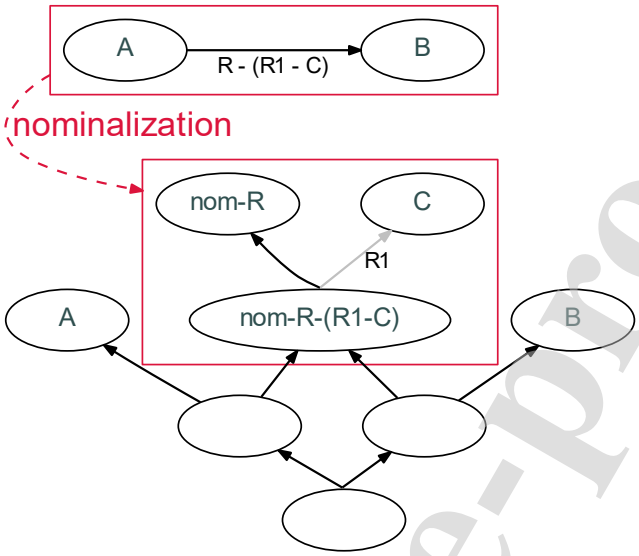


Figure 13: Nominalization of relation with adverbial modifier

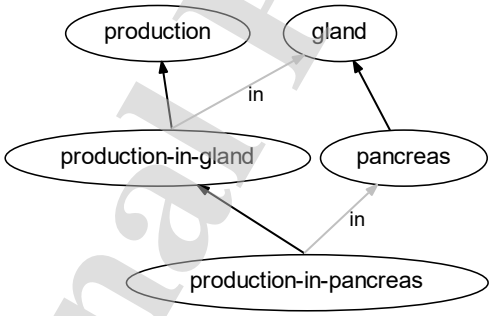


Figure 14: Nominalization contribution from the verbal construct produce in pancreas

4 The NATURALOG Grammar

This section presents the grammar of our natural logic. This is the grammar through which we view natural language text corpora. Obviously, one can only capture fragments of the meaning content of such corpora as discussed in (Andreasen et al., 2017).

Below we present the NATURALOG grammar using the following standard metasymbols:

|	means choice between alternatives
{ }	means 1 occurrence
{ }*	means 0, 1 or more occurrences
[ ]	means optional.

The NATURALOG grammar:

Proposition	::=	*Cterm R Cterm*
Cterm	::=	[*Quant*] *NOUN* [*PostModifier*]
Quant	::=	{ every | some }
PostModifier	::=	{ *Prepterm* }* [*RelClauseterm* { and *RelClauseterm* }*]
Prepterm	::=	*SimplePrepterm* | *ComplexPrepterm*
SimplePrepterm	::=	*Prep NOUN*
ComplexPrepterm	::=	*Prep* (*Cterm*)
RelClauseterm	::=	{ that | which } *R Cterm*
R	::=	R_{act} | R_{pas} | R_{adv} | BE_{cop}
R_{act}	::=	*VERB* [*Prepterm*]
R_{pas}	::=	BE_{aux} $VERB_{ppp}$ [*Prepterm*] by
R_{adv}	::=	BE_{aux} $VERB_{ppp}$ R_{prep}
R_{prep}	::=	*PREPOSITION*
BE_{aux}	::=	is
BE_{cop}	::=	isa

The parenthesis symbols ( ) are used as proper symbols and not as metasymbols. In the non-terminal class *ComplexPrepterm* they are intended to prevent structural ambiguity in the grammar. We illustrate these production rules with sample phrases for selected non-terminal symbols:

Proposition	betacell produce insulin / pancreas isa gland
Cterm	betacell / every betacell / cell in pancreas / cell that produce insulin / cell in (gland that produce hormone)
PostModifier	in pancreas / that produce insulin
SimplePrepterm	in pancreas
ComplexPrepterm	in (gland that produce hormone)
RelClauseterm	which produce insulin / that affect gland that produce hormone / that affect gland and that produce hormone / which reside_in gland that produce insulin / which is located in gland that produce insulin
R_{Act}	produce / produce in pancreas
R_{Pas}	is produced by
R_{Adv}	is located in

In the examples concerning the *RelClauseterm*, we illustrate the possibilities of having alignment or nesting of relative clauses, cf. that affect gland that produce hormone versus that

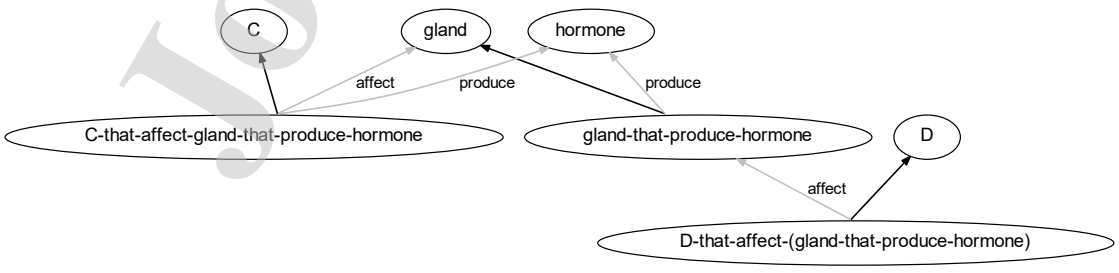


Figure 15: Alignment versus nesting of relative clauses

affect gland and that produce hormone corresponding to the graph in figure 15. Moreover, we insist on mandatory presence of "by" in the passive voice production for R_{pas} in order to avoid mix-up with the adverbial production for R_{adv}.

Using the grammar above for proposition every betacell produce some insulin, we get the key syntactic derivation steps

$$\begin{array}{c} Proposition \\ \Downarrow \\ Cterm\ R\ Cterm \\ \Downarrow \\ \text{every } NOUN\ R_{act} \text{ some } NOUN \\ \Downarrow \\ \text{every betacell produce some insulin} \end{array}$$

As a more complex example illustrating passive voice as well as adverbial restrictive modification consider some insulin is produced in pancreas by (some) betacell with key grammatical derivation steps

$$\begin{array}{c} Proposition \\ \Downarrow \\ Cterm\ R\ Cterm \\ \Downarrow \\ \text{some } NOUN\ R_{pas} \text{ some } NOUN \\ \Downarrow \\ \text{some } NOUN\ BE_{aux}\ VERB_{ppp}\ Prepterm \text{ by some } NOUN \\ \Downarrow \\ \text{some insulin is produced in pancreas by some betacell} \end{array}$$

using the participle produced and an adverbial modifier in the form of a prepositional phrase in pancreas.

Clearly, this grammar in insufficient with respect to premodifiers such as adjectives and genitive forms as well as noun-noun compounds. We posit that premodifiers may be treated on a par with postmodifiers recognizing, however, that the essential problem is that the relation is not explicitly available unlike the case for postmodifiers. Conservative extensions with conjunctions, that is, extensions that can be mapped into this language by paraphrazation, are treated in (Andreasen et al., 2017). Obviously, numerous syntactic as well as semantic phenomena met in scientific corpora fall outside the present grammar and logic framework.

5 Explication of NATURALOG in Predicate Logic

In this section we explain the correlation between NATURALOG and predicate logical sentences. Recall that this logical explication is not part of the NATURALOG system per se, since the computational reasoning applies NATURALOG itself, adhering to the principles of natural

$\forall\exists$	every C R some D	$\forall x(C(x) \rightarrow \exists y(R(x,y) \wedge D(y)))$
$\exists\exists$	some C R some D	$\exists x(C(x) \wedge \exists y(R(x,y) \wedge D(y)))$
$\forall\forall$	every C R every D	$\forall x(C(x) \rightarrow \forall y(D(y) \rightarrow R(x,y)))$
$\exists\forall$	some C R every D	$\exists x(C(x) \wedge \forall y(D(y) \rightarrow R(x,y)))$

Table 1: Quantifier configurations

logics. The computational reasoning rules devised in 7 and 8 refer directly to NATURALOG being decomposed and encoded in a metalogic as explained in section 6.

As indicated in the graphs in the previous sections, the prevailing quantifier configuration is $\forall\exists$, as in

every C R some D

In predicate logic this is explicated as

$$\forall x(C(x) \rightarrow \exists y(R(x,y) \wedge D(y)))$$

and similarly for the other quantifier configurations as shown in table 1, cf. (Nilsson, 2013; Nilsson, 2011).

An important special case of $\forall x(C(x) \rightarrow \exists y(R(x,y) \wedge D(y)))$ is obtained with R being equality, giving $\forall x(C(x) \rightarrow \exists y(x = y \wedge D(y)))$. This is logically equivalent to $\forall x(C(x) \rightarrow D(x))$, which is the NATURALOG sentence form C isa D, cf. section 3.1. Thus, for the case of every C R some D, we get the copula constructions of the form C isa D, which is shorthand for every C isa some D. According to section 3.2.1, we also obtain some D isa some C. The remaining constructions $\forall\forall$ and $\exists\forall$ are not relevant for the case of R being equality and seem less relevant for the general case.

The explication of adverbial modifiers in predicate logic invites higher order notions. Alternatively, transitive verbs may be explicated within first order predicate logic by ternary instead of binary relations, where the third argument contains an appropriate nominalized form of the modifier and with an absent modifier formed as the ontological top concept. Instead of these two approaches we introduce and define auxiliary relations as exemplified below.

As mentioned in chapter 2 all classes are assumed non-empty. This principle, known as existential import, means that there is for each class C present in some NATURALOG sentence implicitly in the knowledge base the declaration $\exists x C(x)$ at the predicate-logical level of explication. In NATURALOG the intended presence of a specific entity or individual k in C may be specified with the proposition k isa C, where k is then re-conceived of as a singleton class. If necessary, the status of k and C may be made explicit at the metalogic level introduced in chapter 6 with clauses individual(k) and class(C). However, as a principle, at the general level of description we conceive of everything as being at the level of classes. This applies also to the most specific "leaf" level in ontologies.

In order to elucidate the definition of NATURALOG in predicate logic let us consider the two derivation examples from chapter 4. The sample every betacell produce some insulin

has the predicate logical corelate

$$\forall x(betacell(x) \rightarrow \exists y(produce(x,y) \wedge insulin(y)))$$

The more complex example some insulin is produced in pancreas by (some) betacell becomes

$$\exists x(\text{insulin}(x) \wedge \exists y(\text{produce-in-pancreas}(\underline{y,x}) \wedge \text{betacell}(y)))$$

appealing to the $\exists\exists$ form in table 1 and inversing the binary predicate by swopping of its arguments in passive voice.

As an additional example, for cell that insulin reside_in pancreas from figure 1 we obtain the rather incomprehensible $\forall x(cell(x) \wedge (\exists z(produce(x,z) \wedge insulin(z)) \rightarrow \exists y(reside\_in(x,y) \wedge pancreas(y))$. As for substances like insulin in the modelling generally speaking they are conceived ontologically as not further specified collections of portions.

In the below listing NATURALOG components are correlated (using $\rightsquigarrow$) with their lambda-abstracted predicate logical terms for the purpose of explaining the semantics. The lambda abstractions are intermediate auxiliaries, only, being bound to vanish in the composition of complete predicate logical sentences using the forms in table 1 substituting for C, R and D.

Noun (Cterms):

betacell $\rightsquigarrow \lambda x.betacell(x)$

Cterm with restrictive modifier:

cell that produce insulin $\rightsquigarrow \lambda x.cell(x) \wedge \exists y(insulin(y) \wedge produce(x,y))$

Prepositional phrase modifier :

in pancreas $\rightsquigarrow \lambda x.\exists y(pancreas(y) \wedge in(x,y))$

Restrictive clausal modifier:

that produce insulin $\rightsquigarrow \lambda x.\exists y(insulin(y) \wedge produce(x,y))$

Nested modifiers:

which reside-in (gland that produce insulin) $\rightsquigarrow$

$$\lambda x.\exists y(gland(y) \wedge \exists z(insulin(z) \wedge produce(x,z)) \wedge residein(x,y))$$

Copula:

isa $\rightsquigarrow \lambda x,y.x = y$

Verb:

produce $\rightsquigarrow \lambda x,y.produce(x,y))$

Verb in passive voice:

is produced by $\rightsquigarrow \lambda x,y.produce(\underline{y,x}))$

Verb with adverbial prepositional phrase:

produce in pancreas $\rightsquigarrow \lambda x,y.produce\text{-}in\text{-}pancreas(x,y))$

Adverbial prepositional phrases are to be supported by nominalization and added supplementary propositions, cf. chapter 3.6. In this example these are: production-in-pancreas isa production and production-in-pancreas in pancreas, where the preposition in acts synonymously with reside-in. Moreover, the adverbial modification giving the relation produce-in-pancreas applies nominalization at the metalogic level as shown for this example in section 3.6.

We now turn to the problem of conducting reasoning directly at the NATURALOG level using the graph decomposition representation in the metalogic.

6 Encoding NATURALOG Propositions in Metalogic

The NATURALOG propositions in a knowledge base are encoded as terms in a metalogic. The inference rules for NATURALOG are stated in the metalogic for computation of query answers. As metalogic we use DATALOG that ensures decidability and tractability in computations (Grosof et al., 2003). The variable-free form of NATURALOG prevents clashes with DATALOG variables unlike the case for a prospective embedded predicate logic. As a simple example the NATURALOG proposition insulin isa hormone is encoded in the knowledge base as a ground atomic DATALOG clause

proposition(insulin, isa, hormone)

where **proposition** is a DATALOG predicate.

DATALOG, unlike say PROLOG, does not endorse compound terms. Thus in general, NATURALOG compound terms are to be decomposed and represented by simple constant terms in DATALOG. As an example, the NATURALOG proposition cell that produce insulin reside-in pancreas from section 2, becomes

proposition(cell-that-produce-insulin, reside_in, pancreas)
definition(cell-that-produce-insulin, isa, cell)
definition(cell-that-produce-insulin, produce, insulin)

reflecting the graph in figure 1. As shown, the predicate arguments including cell-that-produce-insulin are constants at this logical level. Thus, this logical level forms a metalogic in which NATURALOG is term-encoded and embedded and where the inference rules are specified, cf. section 7. The meta-predicate **definition** is intended to convey that the first argument is defined by an if-and-only-if definition. Accordingly, the predicate-logical explication for the concept cell-that-produce-insulin would be

$$\forall x\ (\text{cell-that-produce-insulin}(x)\ \leftrightarrow\ \text{cell}(x) \wedge \exists y(\ \text{produce}\ (x,y) \wedge \text{insulin}\ (y)))$$

As stated earlier the predicate-logical explication of NATURALOG sentences serves the semantical specification, only: The NATURALOG encoding evades predicate logic forms, thereby avoiding an ensuing clash of predicate logical variables and DATALOG variables.

The predicates **proposition** and **definition** at the metalogic level are reflected in an obvious manner in the devised graphs form with each atomic DATALOG sentence corresponding to an arc. The definitional contributions of a concept stated by the predicate **definition** are rendered in the graph by joined outlet arcs, while the arcs of a **proposition** are separated. This notation and the logical difference between **proposition** and **definition** are explicated in sections 3.3 and 7, respectively. The devised decomposition leads in general to creation of multiple DATALOG clauses as shown in this example, which gives rise to three atomic DATALOG sentences.

Description logics would also candidate as a variable-free knowledge base language. However, we advance NATURALOG for a number of reasons compared with description logics as discussed in section 9.1. The main reason is that we consider NATURALOG to be more natu-

ral for knowledge base users in that it offers the linguistically fundamental subject-verb-object form, whereas description logics force all sentences dealing with concept relations into the unnatural and occasionally awkward copula form, that is subject-isa-object.

7 Inference Rules for Deductive Querying

Rather than resorting to the predicate logical explication of NATURALOG, reasoning is carried out at the encoded level. This means that deductive querying can be conducted within the DATALOG logic, that is, the function free sub-language of definite clauses well-known from logic programming, and supported by resolution proving. At this encoded level, universally quantified variables are conceived to range over the concept terms (simple or compound) and relation terms present in the knowledge base. However, from the point of view of DATALOG all of these terms are simply constants. With deductive querying being realized within DATALOG, termination and tractability can be ensured, cf. (Grosof et al., 2003).

As a convenient default, the metalogic predicate **proposition** comes in two forms distinguished by their arity:

proposition(∀∃, C, R, D) ← **proposition**(C, R, D)
proposition(C, R, D) ← **proposition**(∀∃, C, R, D)

The argument tag ∀∃ is a DATALOG constant. Furthermore, we stipulate that definitions are propositions

proposition(C, R, D) ← **definition**(C, R, D)

We also introduce active to passive voice switching, cf. section 3.2.1:

proposition(∃∃, D, Rinv, C) ← **proposition**(∀∃, C, R, D) ∧ **inverse**(R, Rinv)

appealing to an active/passive vocabulary **inverse**(_,_), exemplified by **inverse**(produce, is-produced-by). For the copula we have the metalogic atomic clause **inverse**(isa, isa), cf. again section 3.2.1. Still, from *C* isa *D*, that is every *C* isa *D*, only follows some *D* isa *C*.

In the simplest case, querying of the knowledge base takes place by stating a query predicate with variables (uppercase) as a goal clause as in

proposition(X, produce, insulin)

yielding the answer X = betacell given the knowledge base proposition **proposition**(betacell, produce, insulin).

Here we consider mainly ∀∃ propositions with their dual ∃∃ propositions without undue loss of generality. For the sake of simplicity, we ignore the fact that often defined concepts (concepts appearing as first arguments of the predicate **definition**) such as cell-that-produce-insulin are uninformative from a query point of view and should be omitted in answers.

Of particular importance are copula queries such as

proposition(X, isa, hormone)

intended to explore the various of subclasses of hormones “extensionally”, and conversely

proposition(hormone, isa, X)

intended to provide recorded properties, i.e. superclasses, of hormones in an "intensional" style of querying.

7.1 Monotonicity Deduction Rules

A query task appeals implicitly to appropriate deduction rules. This is because a query normally involves propositions that are deducible from the ones given explicitly in the knowledge base. As an example, given the knowledge base propositions **proposition**(insulin, isa, hormone) and **proposition**(betacell, produce, insulin), the query

proposition(X, produce, hormone)

would intuitively yield X = betacell (with multiple answers to be expected in a more comprehensive knowledge base). This is achieved by means of a pair of logical deduction rules known as monotonicity rules in natural logic (van Benthem, 1986), which can be stated in the embedding DATALOG as

proposition(Csub, R, D) ← **proposition**(Csub, isa, C) ∧ **proposition**(C, R, D)
proposition(C, R, Dsuper) ← **proposition**(C, R, D) ∧ **proposition**(D, isa, Dsuper)

The first rule, illustrated by the two graphs in figure 16(a), provides inheritance to all subconcepts of a concept C, and the second rule, illustrated by the two graphs in figure 16(b), admits generalization of an ascribed property. One may observe that, as a special case, the monotonicity rules provide transitivity of isa with the relation R being isa. These reasoning rules are easily verifiable formally by reduction to the underlying predicate logic.

For the sake of logical completeness we also need the following ∃∃ rules:

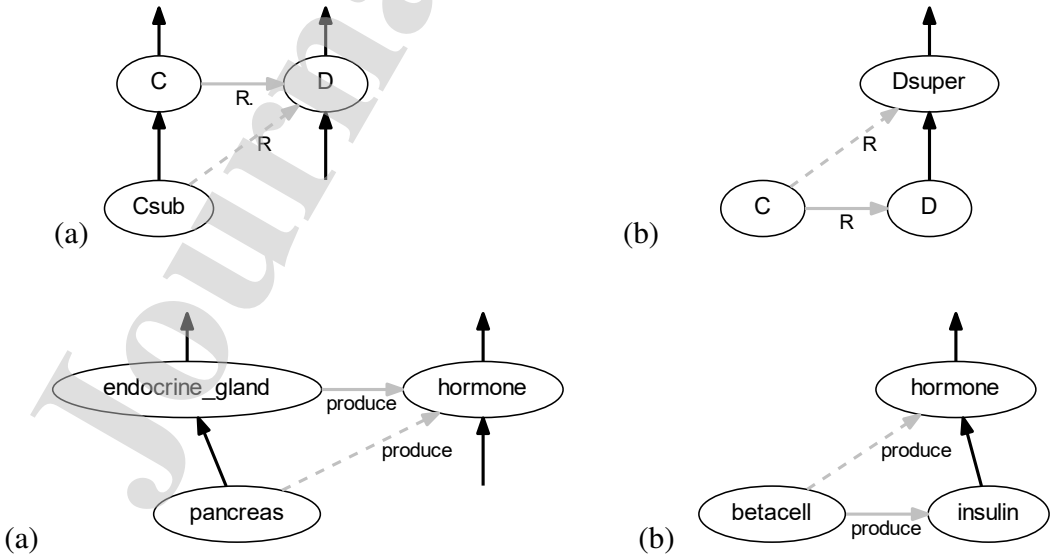


Figure 16: Monotonicity rules: (a) inheritance and (b) generalization

proposition(∃∃, C', R, D) ← **proposition**(∃∃, C, R, D) ∧ **proposition**(C, isa, C')
proposition(∃∃, C, R, D') ← **proposition**(∃∃, C, R, D) ∧ **proposition** (D, isa, D')
proposition(∃∃, D1, isa, D2) ← **proposition**(C, isa, D1) ∧ **proposition**(∃∃, C, isa, D2)
proposition(∃∃, D, isa, C) ← **proposition**(∃∃, C, isa, D)

The former two rules are monotonicity rules. The latter two rules, which define partial overlap and inversion, are illustrated in figure 17. All these ∃∃ rules seem to be less relevant from the point of view of query functionality. Altogether with the given rules, from the NATURALOG proposition *C* isa *D* we get the proposition some *C* isa *D*, from which we get the proposition some *D* isa *C*, recalling insistence on existential import, so that *C* and *D* are non-empty.

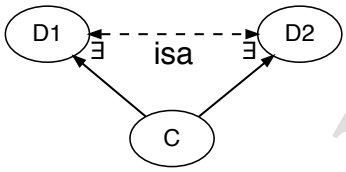


Figure 17: Graph illustrating partial overlap and inversion

Adding adverbial modifiers to relations calls for yet another monotonicity rule saying that dropping as well as relaxing a modifier generalize a relation analogous to generalization of a concept. Thus, by relaxation betacell produce in pancreas insulin entails betacell produce in gland insulin, which in turn entails betacell produce insulin by dropping of the adverbial PP. This adverbial monotonicity principle is achieved by a monotonicity rule for the relational part:

proposition(C, R', D) ← **proposition**(C, R, D) ∧
nominalization(R', R'nom) ∧ **nominalization**(R, Rnom) ∧
proposition(Rnom,isa,R'nom)

This rule admits derivation of, for instance, **proposition**(C, produce, D) given that **proposition**(C, produce-in-gland, D) appealing to **proposition**(production-in-gland, isa, production), cf. figures 14(a) and (b).

7.2 Subsumption

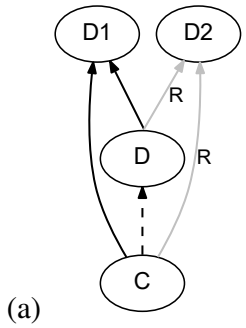
The presence of if-and-only-if definitions in the knowledge base, indicated by the use of the predicate **definition**, calls for the following subsumption rule:

proposition(C, isa, D) ← **definition**(D, isa, D1) ∧ **definition**(D, R, D2) ∧
proposition(C, isa, D1) ∧ **proposition**(C, R, D2)

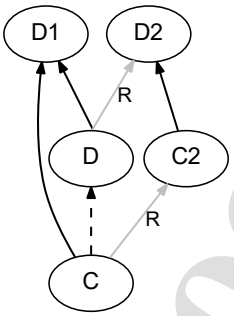
This subsumption rule yields a derived concept inclusion proposition depicted as a dashed arc in figure 18(a) and the example in figure 19(a). Due to the if-and-only-if definition of D, it holds that for any concept X such that X isa D1 and X is R related to D2 we must have that X isa D.

The following alternative version does not appeal to the monotonicity rules, except for transitivity for isa:

proposition(C, isa, D) ← **definition**(D, isa, D1) ∧ **definition**(D, R, D2) ∧
proposition(C, isa, D1) ∧ **proposition**(C2, isa, D2) ∧
proposition(C, R, C2)



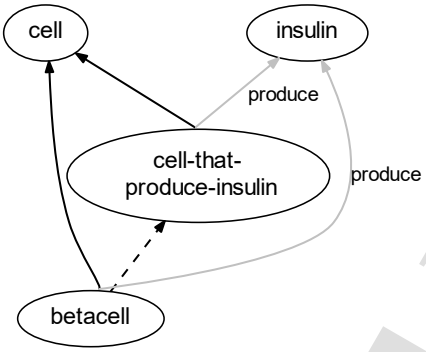
(a)



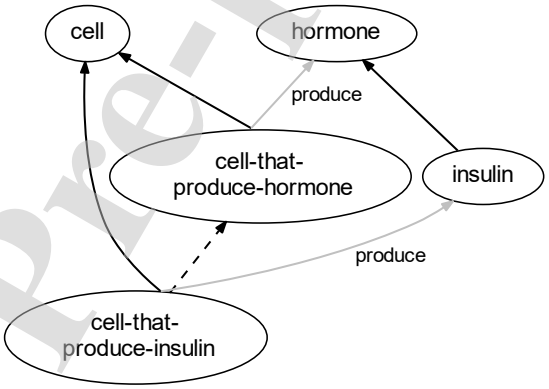
(b)

Figure 18: The subsumption rule with implicit (a) and explicit (b) monotonicity

This is illustrated in figure 18(b) and exemplified in figure 19(b).



(a)



(b)

Figure 19: Inferred arcs: (a) by subsumption (b) by inheritance and subsumption

Adverbial modifiers on relations call for an enhanced version of the subsumption rule appealing to the nominalization of the relations. Recall that the predicate **nominalization**(R,C) says that the nominalization of relation R is the concept C, where C is intended to take into account any modifier on the relation R. Thus, when nominalizing a relation modified by an adverbial PP, the adverbial PP is turned into a postmodifying adnominal PP. For example, we have **nominalization**(produce,production), and further with an adverbial PP we have **nominalization**(produce-in-pancreas,production-in-pancreas). Here the compound concept production-in-pancreas is defined ontologically by **definition**(production-in-pancreas,isa,production) and **definition**(production-in-pancreas,in,pancreas).

proposition(C, isa, D) ← **definition**(D, R1, D1) ∧ **definition**(D, R2, D2) ∧
proposition(C, R1', D1) ∧ **proposition**(C, R2', D2) ∧
isarelation(R1',R1) ∧ **isarelation**(R2',R2)

where this enhanced version of subsumption takes into account comparison of relations carried out by the predicate **isarelation**:

isarelation(Ra, Rb) ← **nominalization**(Ra,Ca) ∧ **nominalization**(Rb,Cb) ∧
proposition(Ca,isa,Cb)

7.3 Special Inference Rules

The above described inference rules enable computing of logical consequences of the given knowledge base sentences for the purpose of query answer computation. In addition, *ad hoc* inference rules may be introduced at the DATALOG metalogical level in order to capture properties of relations. An example is the property of transitivity of a relation. Since the NATURALOG logic has no means *per se* of expressing transitivity of a relation, special purpose inference rules, for expressing transitivity of dedicated relations may be added at the discretion of the knowledge engineer. In addition to the "hardwired" transitivity of isa, some relations, such as relations expressing parthood and causality, may be declared as transitive through an additional inference rule of the outlined form, assuming that appeal can be made to the monotonicity rules:

proposition(C,R,D) ← **istransitive**(R) ∧ **proposition**(C,R,CD) ∧ **proposition**(CD,R,D)

Selected relations are then equipped with the property of transitivity e.g. by stating

istransitive(cause) **istransitive**(part_for) **istransitive**(has_part)

The parthood relations part_for and has_part are introduced and described in (Smith and Rosse, 2004). The NATURALOGsentences C part_for D and D has_part C can be explicated at the level of predicate logic by respectively $\forall x(C(x)) \rightarrow \exists y(part(y,x) \wedge D(y))$ and $\forall x(D(x)) \rightarrow \exists y(part(x,y) \wedge C(y))$ conforming with the principles in section 5. One should notice that these two partonomic relations are not each other's inverse.

Another issue is the handling of negative information in the knowledge base. In the version presented here NATURALOG does not cover negative sentences. However, negative sentences of the form no C R some D may be amended to the language and supported by an inference rule appealing to negation-by-failure $\nvdash$ via adoption of the closed-world assumption for the knowledge base as known from logic programming:

proposition(no,C,R,D) ← $\nvdash$ **proposition**(∃,C,R,D)

In particular, a confirmation of **proposition**(no,C,isa,D), for given C and D, verifies that these two classes are disjoint. Confer also figure 17.

7.4 Materialization of relationships and concepts

We institute a Completion principle saying that all concepts producible within NATURALOG that subsume concepts already present in the graph, are to be materialized as nodes in the graph. This principle ensures that all concepts potentially contributing to the answer of a query are made explicit in the graph. Furthermore, the pathway querying described in section 8.2 also calls for explicit presence of inferred relationships and concepts initially being only implicitly present in the knowledge base.

7.4.1 Materialization of inferred relationships

As a consequence of the Completion principle, additional relations have to be materialized. Exempt from the Completion principle are those isa relations that are inferred by transitivity. This insistence on the explicit presence of isa is adopted to ensure that implicit propositions

following logically should be derivable by applying one of the monotonicity rules. The challenge is that the subsumption rule in the process of materializing an isa-proposition may refer to other isa-propositions pending materialization, thereby initiating recursive invocation of the subsumption rule throughout the knowledge base graph. To this end, we outline an algorithm that materializes isa relationships inferable by subsumption in (Andreasen et al., 2015). In effect, this algorithm applies the subsumption rule in a preprocessing forward-reasoning mode rather than in a call-by-need top-down mode.

By way of example, the materialization of subsumption-derivable isa-propositions ensures that

proposition(betacell, reside-in, pancreas)

becomes directly derivable by means of one of the monotonicity rules from the following propositions

proposition(betacell, isa, cell)
proposition(betacell, produce, insulin)

together with

definition(cell-that-produce-insulin, isa, cell)
definition(cell-that-produce-insulin, produce, insulin)
proposition(cell-that-produce-insulin, reside-in, pancreas)

The above inference rules are crucial to query-answering in that the computed answers in general are only implicitly present in the knowledge base as logical consequences.

7.4.2 Materialization of inferred concepts

As mentioned, all concepts potentially contributing to the answer of a query are to be made explicit in the graph. To achieve this, we now introduce inference rules for integrating new concepts by positing an auxiliary predicate **newconcept**, which composes a new concept from already given concepts. For instance, from cell and produce and hormone we can construct the constant cell-that-produce-hormone. This new concept takes the form of a new constant in the metalogic, thereby, strictly speaking, transcending the confines of DATALOG. The total number of such generated constants is, however, bound to be finite.

We distinguish two cases for a concept, namely pairs of definitional arcs given by the predicate **definition** and pairs of non-definitional arcs given by the predicate **proposition**. The following three rules sharing the same right-hand side take care of a defined concept A.

$$\left.\begin{array}{l}\textbf{definition}(\text{Cnew, isa, B})\\ \textbf{definition}(\text{Cnew, R, C'})\\ \textbf{proposition}(\text{A, isa, Cnew})\end{array}\right\} \leftarrow \begin{array}{l}\textbf{definition}(\text{A, isa, B}) \wedge\\ \textbf{definition}(\text{A, R, C}) \wedge\\ \textbf{proposition}(\text{C, isa, C'}) \wedge\\ \textbf{newconcept}(\text{B, R, C', Cnew})\end{array}$$

As illustrated in figure 20 the three rules create a new concept as the value of Cnew, namely B-that-R-C'. In the case of figure 21 (a) the rules create two new concepts B-that-R-C' and B'-that-R-C, and then, in turn, due to the presence of these, B'-that-R-C'. As it appears, the rules initiate a cascading effect all the way to the top of the ontology.

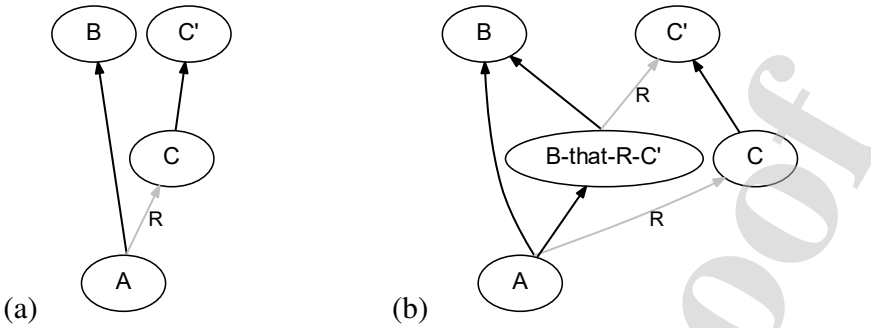


Figure 20: Materialization of a new concept B-that-R-C' from a defined concept A

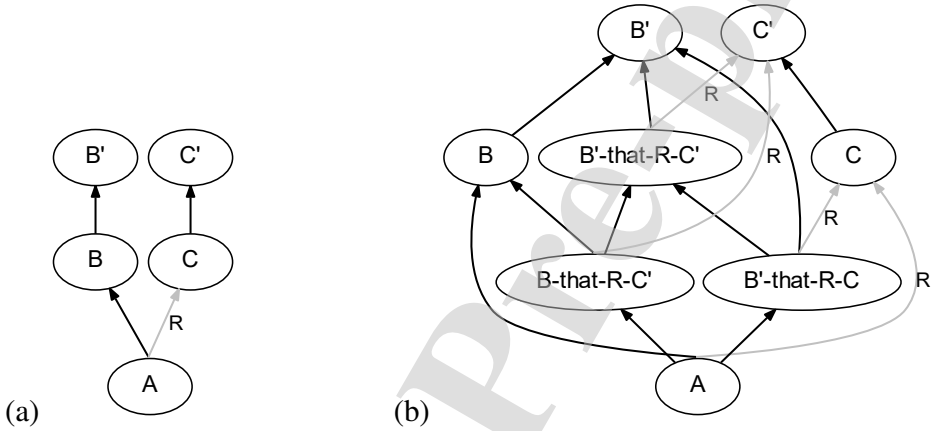


Figure 21: Materialization of three new concepts from a defined concept A

We now turn to the case of non-definitional arcs of concepts. Figure 22 shows how pairs of non-definitional arcs give rise to new defined concepts such as D-that-R-E and cell-that-produce-insulin. Consider every concept C that has a pair of non-definitional outlet arcs such as the

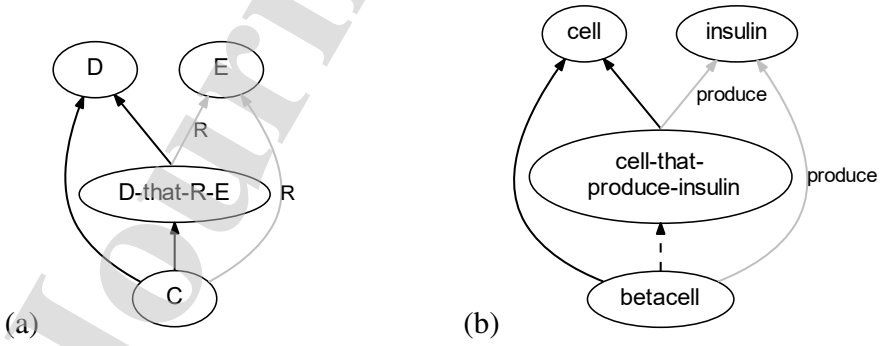


Figure 22: Concepts C in (a) and betacell in (b) with non-definitional arcs introducing new defined concepts

concept betacell in figure 22(b). These concept arcs give rise to additional defined concepts by means of the following three rules sharing the same right-hand side:

definition(Cnew, isa, D)
definition(Cnew, R, E)
proposition(C, isa, Cnew)
$\leftarrow$
proposition(C, isa, D) $\wedge$
proposition(C, R, E) $\wedge$
newconcept(D, R, E, Cnew)

where the auxiliary predicate **newconcept** forms a new concept name such as D-that-R-E and cell-that-produce-insulin in figure 22. The new defined concepts materialized by the stated rules may in turn by regress give rise to formation of further defined concepts as exemplified in figure 23. Assume given betacell produce insulin, betacell isa cell and betacell reside-in pancreas. In a first step, the two former propositions give rise to the concept cell-that-produce-insulin and the two latter ones give rise to cell-that-reside_in-pancreas. In a second step, these two new concepts give rise to cell-that-produce-insulin-and-that-reside_in-pancreas.

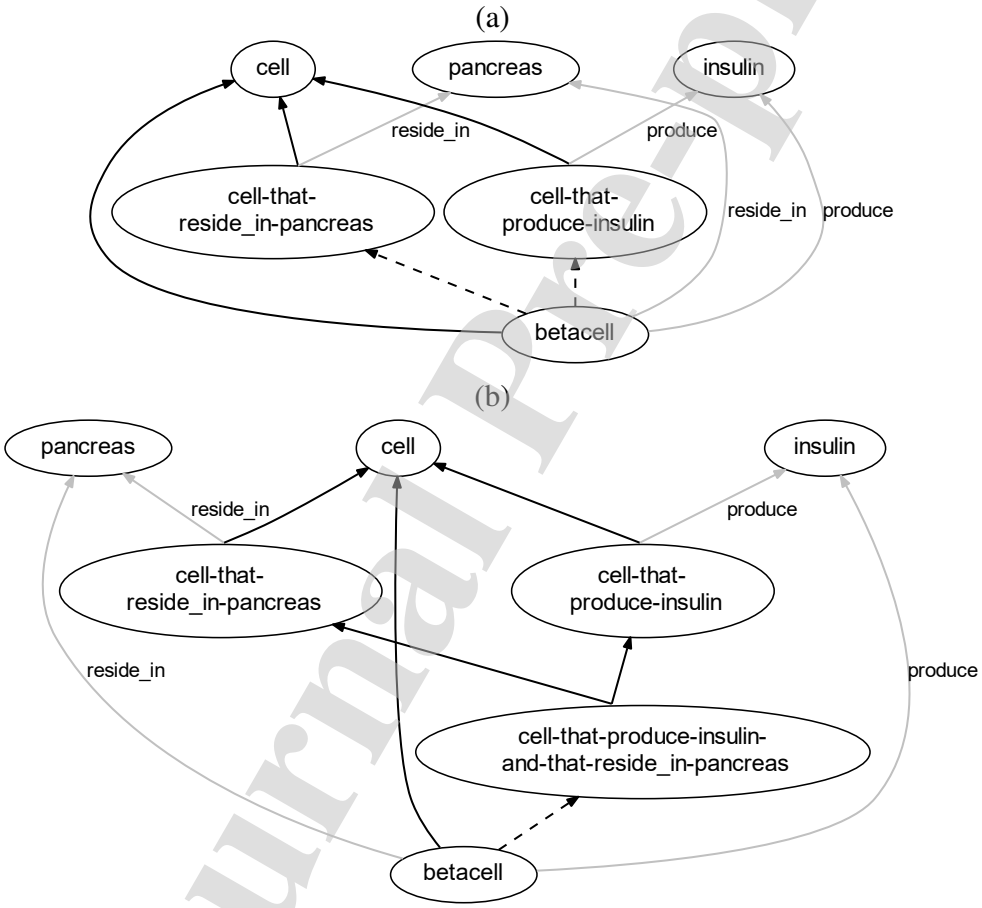


Figure 23: Creating three new concepts in two steps, two in (a) and one in (b)

8 Querying

Having explained the graph form of NATURALOG knowledge base with concepts and relations, we now turn to the question of how this representation can be used for various forms of querying. As already indicated, the key query principle is to form query goals with variables in DATALOG, where these variables range over concepts and relations appearing as labels in the

graph. The query computation is deductive in the sense that it may appeal to the stated inference rules. However, most queries can be computed without using inference rules (less transitivity of isa) these rules having already been applied in the materialization explained in section 7.4. We describe three main forms of querying, namely concept querying, where answers take the form of concepts, pathway querying, where answers take the form of paths between two given concepts, and aboutness querying, where answers take the form of concepts stemming the from nominalization of verbs.

8.1 Concept querying

Concept queries take the form of NATURALOG sentences where terms are replaced by variables indicated by capital letters as in the sample sentence:

X isa hormone

The anticipated answers come about as instantiations of the variables yielding NATURALOG sentences that follow logically from the knowledge base. In the example, the answer would comprise X = insulin assuming that insulin isa hormone is present in the knowledge base. In principle, the computation of concept queries is carried out by formation of goal clauses as described in section 7.

Now consider the knowledge base $\mathcal{K}$:

betacell isa cell
insulin isa hormone
betacell produce hormone

In addition quite generally, we assume that the graph has a top concept $\top$ such that for all concepts C we have that C isa $\top$ as illustrated in figure 25, that is:

cell isa $\top$
hormone isa $\top$

This knowledge base gives rise to the following concept graph following the materialization principles given in section 7.4

Consider next the query:

X produce hormone

Given the knowledge base $\mathcal{K}$ (including materialized concepts), we get the answer set {cell-produce-hormone, cell-produce-insulin, betacell} as successive instantiations of X, appealing to the transitivity of isa.

Query variables can also range over relations as R in:

betacell R hormone

yielding the instantiation answer R = produce given $\mathcal{K}$.

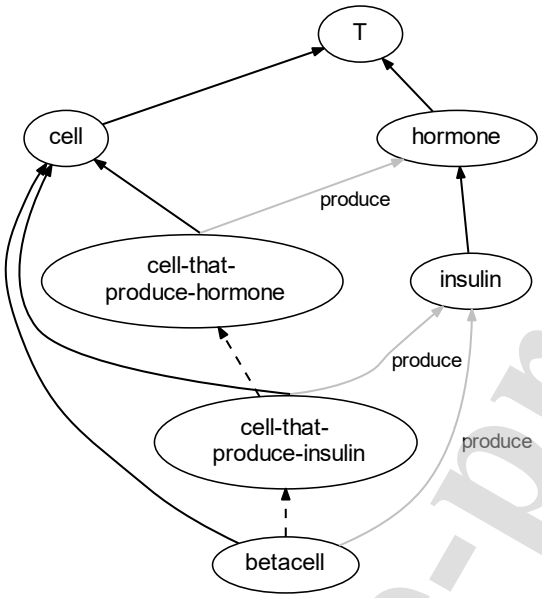


Figure 24: Graph for the knowledge base $\mathcal{K}$ including materialized concepts

8.2 Pathway Querying

The entire knowledge base graph forms a road map between all the applied concepts. The introduction of a universal concept at the top of the ontology ensures that all concepts are connected. This concept map can be queried by means of rules searching pathways in the graph between two stated concepts as sketched here:

path(C, D) ← **proposition**(Q, C, R, CD) ∧ **path**(CD, D)
path(C, D) ← **proposition**(Q, C, R, D)

The predicate **path** may exploit the inverse relation paths, by virtue of the quantifier labels Q being left unspecified, thus exploiting ∃∃ as well as ∀∃ arcs in the pathway, as explained in section ref. The sketched predicate **path** should be extended with an argument that is to be instantiated to the obtained path consisting of a sequence of relations and concepts. The interesting pathways are obviously the shortest ones employing appropriate distance weights to the various relationship forms. This calls for application of efficient standard search algorithms.

Consider the following simple definition extending the predicate **path** with arguments to instantiate the computed pathway:

path2(C, R1, D, R2, E) ← **proposition**(Q1, C, R1, D) ∧ **proposition**(Q2, D, R2, E)

Assume that the knowledge base contains

proposition(∀∃, pancreas, isa, endocrine_gland)
proposition(∀∃, hypothalamus, isa, endocrine_gland)

From the latter follows

proposition(∃∃, endocrine_gland, isa, hypothalamus)

according to the active to passive voice switching in section 7.

The sample query **path2**(pancreas, R1, D, R2, hypothalamus) now yields the intermediate concept endocrine_gland with the isa relations in the form **path2**(pancreas, isa, endocrine_gland, isa, hypothalamus). This result is construed as telling that the two concepts pancreas and hypothalamus have the common property of being endocrine glands. Along these lines we introduce the following definition:

path3(C, R1, D, R2, E, R3, F) ←
proposition(Q1, C, R1, D) ∧ **proposition**(Q2, D, R2, E) ∧ **proposition**(Q3, E, R3, F)

Assume that the knowledge base also contains:

proposition(∀∃, parathyroid_gland, isa, endocrine_gland)
proposition(∀∃, parathyroid_gland, secrete, parathyroid_hormone)
proposition(∀∃, parathyroid_hormone, stimulate, production-of-calcitonin)

The sample query **path3**(endocrine_gland, R1, D, R2, E, R3, production-of-calcitonin) now yields the pathway **path3**(endocrine_gland, isa, parathyroid_gland, secrete, parathyroid_hormone, stimulate, production-of-calcitonin). Obviously, these definitions can be extended to cover ever longer pathways. A realistic implementation may take resort to a standard shortest-path algorithm, such as A*, taking into account also weights on arcs.

8.3 Advanced deductive query forms

The embedding metalogic of NATURALOG enables sophisticated query forms. As an example consider a query setup for computing the for a pair of stated concepts C and D the properties they have in common:

commonality(C, D, R, X) ← **proposition**(C, R, X) ∧ **proposition**(D, R, X)

where the DATALOG variables R and X are to deliver deduced answers for given concepts C and D.

The highly intensional nature of NATURALOG according to which all relevant concept terms are pre-materialized (recalling figure 22 in section 7.4.2) suggests the alternative definition

commonality(C, D, CD) ← **proposition**(C, isa, CD) ∧ **proposition**(D, isa, CD)

Given the NATURALOG propositions: alphacell isa cell, alphacell produce glucagon, beta-cell isa cell that produce insulin, glucagon isa hormone, insulin isa hormone, then a commonality concept as instantiation of CD obtains as the compound term

cell that produce hormone

by engaging of appropriate inference rules in the query computations. One observes that the three concepts in figure 25, cell that produce insulin, cell that produce glucagon and cell that produce hormone are materialized following the principles in section 7.4.

As another example, alluding to the supplementation principle in mereology albeit here at the level of concepts rather than individuals, and using DATALOG$^{\forall}$ (DATALOG extended with negation-as-failure), one may introduce

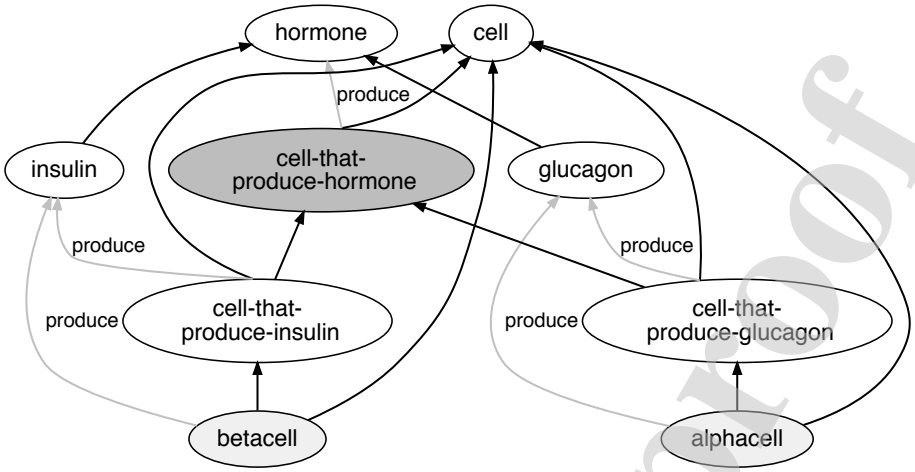


Figure 25: The commonality of the concepts alphacell and betacell is cell that produce hormone

hasmultiparts(D) ←
proposition(C1, partfor, D) ∧ **proposition**(C2, partfor, D) ∧ ∀ **identical**(C1, C2)

with the auxiliary clause

identical(C, C)

assuming here for the sake of simplicity that the partonomic relation partfor is not declared transitive.

8.4 Aboutness querying

When querying the knowledge base, one may be interested in retrieving all the propositions that are "about" a certain concept. This may be achieved in the present setup by appealing to the meta-relation between relations R for transitive verbs and their corresponding nominalized concepts, if such ones exist, as explained in 3.6. As an example, one may query with the concept production-of-insulin and get a proposition such as betacell produce insulin. This functionality relies on availability of vocabularies providing thematic roles for the verbs and is therefore not elaborated in this context.

Various knowledge base integrity constraints can readily be specified at the metalogic level. As an example, an accidentally erroneous parthood specification in the KB violating asymmetry can be discovered with the general metalevel clause:

error(partfor, C, D) ← **proposition**(C, partfor, D) ∧ **proposition**(D, partfor, C)

9 Relationship to other Languages and Logics

As discussed thoroughly in the previous sections, our natural logic is closely has an accompanying graph representation, the concept graphs, where complex concepts are decomposed and where concepts are supposed to have a unique node representation. This means that concept

graphs bear some affinity to the semantic network tradition (Brachman and Schmolze, 1985; Woods and Schmolze, 1992), where the conceptual-graph proposal (Sowa, 1979; Sowa, 2000) is a prominent example. Sowa's conceptual graphs are based on Peirce's existential graphs, see e.g. (Shin, 2002) for a contemporary description. Existential graphs afford diagrams for propositions in first order predicate logic without function symbols. In the graph diagram *n*-ary predicate symbols are represented as nodes with arcs (possibly multiple, that is connecting more than two nodes) representing co-occurrence of a variable in predicate arguments. The universal quantifier $\forall$ becomes the existential quantifier embraced in negations, $\neg\exists\neg$, where the negation (here twice) is depicted by an enclosure in the graph diagram. This means that a simple copula sentence such as '*every betacell isa cell*' becomes what corresponds to the rather awkward '*there does not exist a betacell which is not a cell*'. Moreover, in existential graphs each proposition is to be represented separately, whereas in our concept graphs the knowledge base sentences form a connected graph with unique node representations. For these reasons, there is no tight relationship between the our concept graphs and conceptual graphs.

There are affinities between our NATURALOG approach and modelling languages such as Entity-Relationship (Chen, 1976) and UML (Rumbaugh et al., 2004) as well as more ontology focussed languages like OntoUML (Guizzardi, 2005). Modelling languages are commonly supported by tools ranging from simple editors to more complex tools supporting validation by logic reasoning such as the OntoUML editor (Guerson et al., 2015) and ICOM (Fillottrani et al., 2012). NATURALOG is rather to be seen as an attempt to provide a stylised fragment of natural language coming with a formal semantics and a collection of inference rules facilitating deductive querying and validation. This means that NATURALOG at the same time is a modelling language and a knowledge base logic.

Finally, let us mention class relationship diagrams, which are a diagrammatic form of NATURALOG without compound terms (Nilsson, 2013). Class relationship diagrams employ the usual notion of Euler diagrams for reasoning with class inclusion extended with cogent diagrammatic symbols for relational reasoning with the monotonicity rules and with pathway reasoning.

9.1 Natural Logics and Description Logics

Let us now turn to description logics (DL), cf. (Krötzsch et al., 2012; Motik et al., 2009). At first sight, there might seem to be a close relationship to DL, with NATURALOG appearing as a syntactically ameliorated version of DL. The so-called terminological forms in DL at stake here follow the pattern subject-*copula*-object, where the copula is given by the operator symbol $\sqsubseteq$. As an example, betacell produce insulin in DL becomes betacell $\sqsubseteq \exists$ produce.insulin, which may be understood as betacell isa [thing] that produce insulin.

By contrast, the salient scheme in NATURALOG is the more general subject-*verb*-object with copula provided as merely a special case. Thus, we recognize the key role of (transitive) verbs in assertoric sentences and find the insistence on copula forms far from common use in natural language texts. Moreover, as described in section 3.2.1, NATURALOG supports the active/passive dual sentence pairs. These latter are problematic in DL, if not plainly missing, in that the multiple quantifier constellation *some-some* unlike *every-some* is not directly available in DL. On the other hand, NATURALOG in the present version affords only affirmative propo-

sitions and preclude the empty concept as well as numerical quantifiers. However, one may bear in mind that the case of disjointness of two concepts, say, alphacell and betacell (stated in DL either as alphacell $\sqsubseteq \neg$ betacell or by alphacell $\sqcap$ betacell $\sqsubseteq \bot$) is achieved by default in NATURALOG, appealing to the closed-world assumption as explained in section 3. In addition, as also described in section 3.6, NATURALOG affords relational restrictions in the form of adverbial PPs, which are not uncommon in the considered domain.

Besides these differences there is a deeper, fundamental logical difference in that DL at the outset relies on an extensional understanding of concept terms as denoting sets. Accordingly, two concept terms become inter-substitutable if they comprise the same individuals, similarly to predicates in predicate logic. By contrast, NATURALOG takes a more comprehensive intensional view on concepts and relationships as manifest by the applied encoding into an embedding logic as elaborated above in section 6 and 7. This implies that query answers may take form of compound concept terms and relational terms. The intensional view further affords pathway deductive computation between two stated concepts from the knowledge base as described in section 8.2.

10 Summary and Conclusion

In this paper we have demonstrated how sentences in natural logic knowledge bases can be decomposed into simpler sentences that can be encoded into in DATALOG clauses. The relevant logical inference rules are expressed in DATALOG clauses with variables ranging over constituents of natural logic sentences, ensuring computational tractability and decidability. The encoded sentences with the accompanying inference rules provide a coherent graph conception of the knowledge base, generalising the usual graph conception of formal ontologies, insisting on unique representations of concepts as nodes throughout the graph. We have specified syntactically and logically a form of natural logic called NATURALOG, a regimented fragment of natural language, intended for logical knowledge bases, that can be queried deductively. Furthermore, the decomposition of sentences into labelled subgraphs which are integrated in the overall coherent knowledge base graph, enables pathway querying. Our natural logic applies the closed world assumption in accord with common implicit conventions in scientific ontologies and taxonomies and goes beyond copula forms by admitting transitive verbs, thereby enabling specifications of relationships between stated classes in augmented ontologies.

We consider this work a modest but elaborate contribution meeting the challenge of providing knowledge bases with reasoning capabilities expressed in a natural logic approaching scientific use of natural language.

10.1 Acknowledgement

We would like to thank the anonymous reviewers for their detailed comments and suggestions leading to clarifications and improvements on the first version of this paper. Allow us to dedicate this paper to our dear colleague and coauthor Per Anker Jensen, who sadly passed away before the finalizing of our paper.

References

Andreasen, T., Bulskov, H., Jensen, P. A., and Nilsson, J. F. (2014). Computing pathways in bio-models derived from bio-science text sources. In *Proceedings of the IWBBIO International Work-Conference on Bioinformatics and Biomedical Engineering, Granada, April*, pages 217–226.

Andreasen, T., Bulskov, H., Jensen, P. A., and Nilsson, J. F. (2015). A system for conceptual pathway finding and deductive querying. In *Flexible Query Answering Systems 2015*, pages 461–472. Springer.

Andreasen, T., Bulskov, H., Jensen, P. A., and Nilsson, J. F. (2017). *Partiality, Underspecification, and Natural Language Processing*, chapter A Natural Logic for Natural-Language Knowledge Bases. Cambridge Scholars.

Andreasen, T., Bulskov, H., Jensen, P. A., and Nilsson, J. F. (2019). Deductive querying of natural logic bases. In Cuzzocrea, A., Greco, S., Larsen, H. L., Saccà, D., Andreasen, T., and Christiansen, H., editors, *Flexible Query Answering Systems*, pages 231–241, Cham. Springer International Publishing.

Andreasen, T. and Nilsson, J. F. (2004). Grammatical specification of domain ontologies. *Data Knowl. Eng.*, 48(2):221–230.

Brachman, R. J. and Schmolze, J. G. (1985). An overview of the kl-one knowledge representation system. *Cognitive Science*, 9(2):171–216. http://www.cogsci.rpi.edu/CSJarchive/1985v09/i02/p0171p0216/MAIN.PDF.

Chen, P. P.-S. (1976). The entity-relationship model—toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36.

Fillottrani, P. R., Franconi, E., and Tessaris, S. (2012). The icom 3.0 intelligent conceptual modelling tool and methodology. *Semant. web*, 3(3):293–306.

Grosof, B. N., Horrocks, I., Volz, R., and Decker, S. (2003). Description logic programs: Combining logic programs with description logic. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 48–57, New York, NY, USA. ACM.

Guerson, J., Prince Sales, T., Guizzardi, G., and Almeida, J. (2015). Ontouml lightweight editor: A model-based environment to build, evaluate and implement reference ontologies.

Guizzardi, G. (2005). *Ontological foundations for structural conceptual models*. PhD thesis, University of Twente.

Klima, G. (2010). Natural logic, medieval logic and formal semantics. *MAGYAR FILOZFIAI SZEMLE*, 54 (4):58–75.

Krötzsch, M., Simančík, F., and Horrocks, I. (2012). A description logic primer. *CoRR*, abs/1201.4089.

Moss, L. S. (2010). Syllogistic logics with verbs. *J. Log. Comput.*, 20(4):947–967.

Motik, B., Grau, B. C., Horrocks, I., and Sattler, U. (2009). Representing ontologies using description logics, description graphs, and rules. *Artificial Intelligence*, 173(14):1275 – 1309.

Nilsson, J. F. (2011). Querying class-relationship logic in a metalogic framework. In Christiansen, H., De Tré, G., Yazici, A., Zadrozny, S., Andreasen, T., and Larsen, H. L., editors, *Flexible Query Answering Systems*, pages 96–107, Berlin, Heidelberg. Springer Berlin Heidelberg.

Nilsson, J. F. (2013). Diagrammatic reasoning with classes and relationships. In Moktefi, A. and Shin, S., editors, *Visual Reasoning with Diagrams*, Studies in Universal Logic, pages 83–100. Springer.

Nilsson, J. F. (2014). *IS-A Diversified*, pages 17–24. IOS Press.

Nilsson, J. F. (2015). In pursuit of natural logics for ontology-structured knowledge bases. In *The Seventh International Conference on Advanced Cognitive Technologies and Applications*.

Rumbaugh, J., Jacobson, I., and Booch, G. (2004). *Unified Modeling Language Reference Manual, The (2nd Edition)*. Pearson Higher Education.

Sánchez Valencia, V. M. (1991). *Studies on Natural Logic and Categorial Grammar*. Categorial grammar. Universiteit van Amsterdam, Amsterdam, Holland.

Shin, S.-J. (2002). *The Iconic Logic of Peirce's Graphs*. Bradford Book. From the Commens Bibliography — http://www.commens.org/bibliography/monograph/shin-sun-joo-2002-iconic-logic-peirces-graphs.

Smith, B. and Rosse, C. (2004). The role of foundational relations in the alignment of biomedical ontologies. *Medinfo*, 11(Pt 1):444–448.

Sowa, J. F. (1979). Semantics of conceptual graphs. In *Proceedings of the 17th Annual Meeting on Association for Computational Linguistics*, ACL '79, pages 39–44, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sowa, J. F. (2000). *Knowledge Representation: Logical, Philosophical and Computational Foundations*. Brooks/Cole Publishing Co., Pacific Grove, CA, USA.

van Benthem, J. (1986). *Essays in Logical Semantics, Volume 29 of Studies in Linguistics and Philosophy*. D. Reidel, Dordrecht, Holland.

Woods, W. A. and Schmolze, J. G. (1992). The kl-one family. *Computers & Mathematics with Applications*, 23(2):133 – 177.

- Natural logic for knowledge bases
- The predicate logic semantics for natural logic
- Graph representation of natural logic sentences
- Deductive querying of natural logic knowledge bases
- A natural logic for extended ontologies

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: