Indexing of Linguistic Knowledge

Kiril Iv. Simov* & Paul John King†

September 6, 1996

1 Introduction

The classification of linguistic objects is a widespread and important task for not only computational linguistics but also linguistics proper. One example of this (indeed, the particular example that motivates the ongoing research that this paper reports) is the classification of words according to their inflectional paradigms. The benefits of such a classification for natural language processing are apparent even for a language, such as English, with a relatively simple inflectional system. When one considers a more inflectionally complex language, such as German or Bulgarian, in which inflection is a ubiquitous and indispensable part of syntactic processing, the paradigmatic classification of words is a *sine qua non* of any successful natural language processing system.

We take a **classificatory system** to comprise (i) a **classification**, a collection of classes into which linguistic objects can be put, and (ii) an **index**, a mechanism that effectively puts each linguistic object into its correct class on the basis of limited knowledge about the object. Some linguistic theories give classificatory systems implicitly, others explicitly. Linguists usually prefer implicit classificatory systems, since they enable the expression of intuitions about linguistic objects as general theories. However, computer scientists usually prefer explicit classificatory systems, since they enable such tasks as fast look-up of inflectional form, acquisition of new linguistic knowledge, and controlled linguistic inference. Consequently, much work in computational linguistics requires the deduction of a classificatory system from a general linguistic theory. Typically, the deduction of a classificatory system from a theory is done by hand. However, hand-crafting a classificatory system is time consuming, a very significant factor in, say, machine translation, where different classificatory systems for several languages may be required. Moreover, a hand-crafted classificatory system risks unwittingly violating the theory from which it is deduced, with potentially disastrous consequences, as, say, the classification wrongly conflates two distinct classes, or the index puts some linguistic objects into inappropriate classes. For these two reasons alone, a device that effectively generates an accurate classificatory system from a general linguistic theory would be of benefit to computational linguistics. In the present paper, we present a prototypical abstract device that generates classificatory systems from finite HPSG (Head-driven Phrase Structure Grammar) theories.

From our naïve experiences attempting to automatically generate paradigmatic classificatory systems from general theories of inflection, an ideal device for automatically generating a classificatory system from a linguistic theory should possess at least five qualities. (Q1) The classification should provide exactly one class for each linguistic object, in order to reduce the non-determinism in any computational system that uses the classification. (Q2) The classification should not provide classes that cannot be occupied without violating the linguistic theory from which the classification is deduced, in order to reduce the search space in any computational system that uses the classification. (Q3) The classification should provide classes that are neither too narrow nor too broad, in order to ensure that the classification truly captures the intuitions of the theory from which it is deduced. (Q4) The index should classify each linguistic object correctly. (Q5) The index should classify each linguistic object efficiently, in order to facilitate rapid and inexpensive classification of linguistic objects. Qualities (Q3) and (Q5) concern mechanisms by which the device can be controlled so as to output an optimal classificatory system for a given input linguistic theory. In the present paper, we address the much simpler qualities (Q1), (Q2) and (Q4) in order to establish that automatically generating a classificatory system from a linguistic theory is possible, and leave control mechanisms for achieving optimal output to future research.

*Linguistic Modelling Laboratory, Bulgarian Academy of Sciences, Sofia, Bulgaria, kivs@bgcict.acad.bg

†Seminar für Sprachwissenschaft, Eberhard-Karls-Universität, Tübingen, Germany, king@sfs.nphil.uni-tuebingen.de

2 Classification

Our device takes as input a finite HPSG theory written using the SRL (Speciate Re-entrant Logic) of [King 1989]. SRL is a sound, complete and decidable logic (see [King 1989] and [Kepser 1994]) designed specifically for HPSG formalisms. SRL can support formalisms for both the information-based HPSG of [Pollard and Sag 1987] (see [King 1989]) and the object-based HPSG of [Pollard and Sag 1994] (see [King 1994]).

$\langle \mathbb{S}, \mathbb{F}, \mathbb{A} \rangle$ is a **signature** iff (i) $\mathbb{S}$ is a finite set of **species**, (ii) $\mathbb{F}$ is a finite set of **features**, and (iii) $\mathbb{A}$ is a total function, called **appropriateness**, from $\mathbb{S} \times \mathbb{F}$ to $\mathsf{Pow}(\mathbb{S})$.[1] For notational facility, we work with respect to a fixed signature $\langle \mathbb{S}, \mathbb{F}, \mathbb{A} \rangle$ throughout, and assume that none of the symbols $:$, $\sim$, $\approx$, $\not\approx$, $\neg$, $\wedge$, $\vee$, $\rightarrow$, [and] is a species or an attribute. $I = \langle U, S, F \rangle$ is an **interpretation** iff (i) U is a set of **objects**, (ii) S is a total function from U to S, (iii) F is a total function from $\mathbb{F}$ to the set of partial functions from U to U, (iv) for each $\varphi \in \mathbb{F}$, for each $\upsilon \in U$, $F(\varphi)(\upsilon)$ is defined iff $\mathbb{A}\langle S(\upsilon), \varphi \rangle \neq \emptyset$, and (v) for each $\varphi \in \mathbb{F}$, for each $\upsilon \in U$, if $F(\varphi)(\upsilon)$ is defined then $S(F(\varphi)(\upsilon)) \in \mathbb{A}\langle S(\upsilon), \varphi \rangle$.

The set of **terms** is the smallest set $\mathbb{T}$ such that (i) $: \in \mathbb{T}$, and (ii) for each $\tau \in \mathbb{T}$, for each $\varphi \in \mathbb{F}$, $\tau\varphi \in \mathbb{T}$. For each $:\varphi_1 \ldots \varphi_m \in \mathbb{T}$, for each $:\varphi'_1 \ldots \varphi'_n \in \mathbb{T}$, we write $:\varphi_1 \ldots \varphi_m \circ :\varphi'_1 \ldots \varphi'_n$ for $:\varphi_1 \ldots \varphi_m \varphi'_1 \ldots \varphi'_n$. For each interpretation $I = \langle U, S, F \rangle$, the **term interpretation function** under I is the total function T_I from $\mathbb{T}$ to the set of partial functions from U to U such that (i) for each $\upsilon \in U$, $T_I(:)(\upsilon) = \upsilon$, and (ii) for each $\tau \in \mathbb{T}$, for each $\varphi \in \mathbb{F}$, for each $\upsilon \in U$, $T_I(\tau\varphi)(\upsilon) = F(\varphi)(T_I(\tau)(\upsilon))$.

The set of **descriptions** is the smallest set $\mathbb{D}$ such that (i) for each $\tau \in \mathbb{T}$, for each $\sigma \in \mathbb{S}$, $\tau \sim \sigma \in \mathbb{D}$, (ii) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, $\tau_1 \approx \tau_2 \in \mathbb{D}$ and $\tau_1 \not\approx \tau_2 \in \mathbb{D}$, (iii) for each $\delta \in \mathbb{D}$, $\neg\delta \in \mathbb{D}$, and (iv) for each $\delta_1 \in \mathbb{D}$, for each $\delta_2 \in \mathbb{D}$, $[\delta_1 \wedge \delta_2] \in \mathbb{D}$, $[\delta_1 \vee \delta_2] \in \mathbb{D}$ and $[\delta_1 \rightarrow \delta_2] \in \mathbb{D}$. We call descriptions of the form $\tau \sim \sigma$, $\tau_1 \approx \tau_2$ and $\tau_1 \not\approx \tau_2$ **atomic descriptions**. For each interpretation $I = \langle U, S, F \rangle$, the **description denotation function** under I is the total function D_I from $\mathbb{D}$ to $\mathsf{Pow}(U)$ such that

$$D_I(\tau \sim \sigma) = \{\upsilon \in U \mid T_I(\tau)(\upsilon) \text{ is defined and } S(T_I(\tau)(\upsilon)) = \sigma\},$$
$$D_I(\tau_1 \approx \tau_2) = \{\upsilon \in U \mid T_I(\tau_1)(\upsilon) \text{ is defined, } T_I(\tau_2)(\upsilon) \text{ is defined and } T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)\},$$
$$D_I(\tau_1 \not\approx \tau_2) = \{\upsilon \in U \mid T_I(\tau_1)(\upsilon) \text{ is defined, } T_I(\tau_2)(\upsilon) \text{ is defined and } T_I(\tau_1)(\upsilon) \neq T_I(\tau_2)(\upsilon)\},$$
$$D_I(\neg\delta) = (U \setminus D_I(\delta)),$$
$$D_I([\delta_1 \wedge \delta_2]) = D_I(\delta_1) \cap D_I(\delta_2),$$
$$D_I([\delta_1 \vee \delta_2]) = D_I(\delta_1) \cup D_I(\delta_2), \text{ and}$$
$$D_I([\delta_1 \rightarrow \delta_2]) = (U \setminus D_I(\delta_1)) \cup D_I(\delta_2).$$[2]

A **theory** is a set of descriptions. For each interpretation $I = \langle U, S, F \rangle$, the **theory denotation function** under I is the total function Θ_I from $\mathsf{Pow}(\mathbb{D})$ to $\mathsf{Pow}(U)$ such that $\Theta_I(\theta) = \{\upsilon \in U \mid \text{for each } \delta \in \theta, \upsilon \in D_I(\delta)\}$. Theory θ **entails** description δ iff for each interpretation I, $\Theta_I(\theta) \subseteq D_I(\delta)$. [King 1989] provides a Hilbert and Ackermann style inference relation that is sound and complete with respect to entailment. Theory θ is **satisfiable** iff for some interpretation I, $\Theta_I(\theta) \neq \emptyset$. [Kepser 1994] provides an effective algorithm to decide if a finite theory is satisfiable. Consequently, SRL is decidable, since a description δ is a theorem (i.e., $\emptyset$ infers δ) iff $\{\neg\delta\}$ is not satisfiable. See [King 1989] and [King 1994] concerning the formal presentation of HPSG theories as SRL theories.

Moving beyond SRL towards our classification, a **clause** is a finite set of atomic descriptions, and a **matrix** is a finite set of clauses. We write $\mathbb{M}$ for the set of matrices. For each interpretation $I = \langle U, S, F \rangle$, the **matrix denotation function** under I is the total function M_I from $\mathbb{M}$ to $\mathsf{Pow}(U)$ such that $M_I(\mu) = \{\upsilon \in U \mid \text{for some } \theta \in \mu, \upsilon \in \Theta_I(\theta)\}$. Thus, matrix $\{\{\delta_{i,j} \mid j < n_i\} \mid i < m\}$ is semantically equivalent to the negation-free disjunctive normal form description $\bigvee_{i<m}(\bigwedge_{j<n_i} \delta_{i,j})$. Term is the total function from $\mathbb{M}$ to $\mathsf{Pow}(\mathbb{T})$ such that

$$\mathsf{Term}(\mu) = \left\{ \tau \in \mathbb{T} \;\middle|\; \begin{array}{l} \text{for some } \theta \in \mu, \text{ for some } \omega \in \mathbb{F}^*, \\ \qquad \text{for some } \sigma \in \mathbb{S}, \tau\omega \sim \sigma \in \theta, \text{ or} \\ \qquad \text{for some } \tau' \in \mathbb{T}, \tau\omega \approx \tau' \in \theta, \text{ or} \\ \qquad \text{for some } \tau' \in \mathbb{T}, \tau' \approx \tau\omega \in \theta, \text{ or} \\ \qquad \text{for some } \tau' \in \mathbb{T}, \tau\omega \not\approx \tau' \in \theta, \text{ or} \\ \qquad \text{for some } \tau' \in \mathbb{T}, \tau' \not\approx \tau\omega \in \theta \end{array} \right\}.$$

Matrix μ is **exclusive** iff for each $\theta \in \mu$,

(E1) $:\approx: \in \theta$,

(E2) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, if $\tau_1 \approx \tau_2 \in \theta$ then $\tau_2 \approx \tau_1 \in \theta$,

[1] $\mathbb{F}$ can be countably infinite if a recursiveness condition is imposed on $\mathbb{A}$, but we use this simpler formulation for brevity.

[2] A description of the form $\tau_1 \not\approx \tau_2$ is strictly not part of the SRL syntax. However, $\tau_1 \not\approx \tau_2$ is semantically equivalent to $[[\tau_1 \approx \tau_1 \wedge \tau_2 \approx \tau_2] \wedge \neg\tau_1 \approx \tau_2]$, therefore $\tau_1 \not\approx \tau_2$ can be seen as a meta-syntactic addition to the SRL syntax.

(E3) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, for each $\tau_3 \in \mathbb{T}$, if $\tau_1 \approx \tau_2 \in \theta$ and $\tau_2 \approx \tau_3 \in \theta$ then $\tau_1 \approx \tau_3 \in \theta$,

(E4) for each $\tau \in \mathbb{T}$, for each $\varphi \in \mathbb{F}$, if $\tau\varphi \approx \tau\varphi \in \theta$ then $\tau \approx \tau \in \theta$,

(E5) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, for each $\varphi \in \mathbb{F}$,
if $\tau_1 \approx \tau_2 \in \theta$, $\tau_1\varphi \approx \tau_1\varphi \in \theta$ and $\tau_2\varphi \approx \tau_2\varphi \in \theta$ then $\tau_1\varphi \approx \tau_2\varphi \in \theta$,

(E6) for each $\tau \in \mathbb{T}$, $\tau \approx \tau \in \theta$ iff for some $\sigma \in \mathbb{S}$, $\tau \sim \sigma \in \theta$,

(E7) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, for each $\sigma_1 \in \mathbb{S}$, for each $\sigma_2 \in \mathbb{S}$,
if $\tau_1 \approx \tau_2 \in \theta$, $\tau_1 \sim \sigma_1 \in \theta$ and $\tau_2 \sim \sigma_2 \in \theta$ then $\sigma_1 = \sigma_2$,

(E8) for each $\tau \in \mathbb{T}$, for each $\varphi \in \mathbb{F}$, for each $\sigma_1 \in \mathbb{S}$, for each $\sigma_2 \in \mathbb{S}$,
if $\tau \sim \sigma_1 \in \theta$ and $\tau\varphi \sim \sigma_2 \in \theta$ then $\sigma_2 \in \mathbb{A}\langle\sigma_1, \varphi\rangle$,

(E9) for each $\tau \in \mathbb{T}$, for each $\varphi \in \mathbb{F}$, for each $\sigma \in \mathbb{S}$,
if $\tau \sim \sigma \in \theta$, $\tau\varphi \in \mathsf{Term}(\mu)$ and $\mathbb{A}\langle\sigma, \varphi\rangle \neq \emptyset$ then $\tau\varphi \approx \tau\varphi \in \theta$,

(E10) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, if $\tau_1 \not\approx \tau_2 \in \theta$ then $\tau_1 \approx \tau_1 \in \theta$ and $\tau_2 \approx \tau_2 \in \theta$,

(E11) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$,
if $\tau_1 \approx \tau_1 \in \theta$ and $\tau_2 \approx \tau_2 \in \theta$ then $\tau_1 \approx \tau_2 \in \theta$ or $\tau_1 \not\approx \tau_2 \in \theta$, and

(E12) for each $\tau_1 \in \mathbb{T}$, for each $\tau_2 \in \mathbb{T}$, $\tau_1 \approx \tau_2 \notin \theta$ or $\tau_1 \not\approx \tau_2 \notin \theta$.

By modifying an algorithm given in [Götz 1993] and used in [Kepser 1994], we arrive at a computable function Excl from the set of finite theories to the matrices, such that for each finite $\theta \subseteq \mathbb{D}$, for each interpretation I, $\Theta_I(\theta) = M_I(\mathsf{Excl}(\theta))$. Moreover, for each exclusive matrix μ, (i) for each $\theta_1 \in \mu$, for each $\theta_2 \in \mu$, if $\theta_1 \neq \theta_2$ then $\theta_1 \cup \theta_2$ is not satisfiable, and (ii) for each $\theta \in \mu$, θ is satisfiable. Thus,

1 Theorem. *For each finite $\theta \subseteq \mathbb{D}$, for each interpretation I, for each $\upsilon \in \Theta_I(\theta)$,*
for some unique $\theta' \in \mathsf{Excl}(\theta)$, $\upsilon \in \Theta_I(\theta')$.

In short, Excl effectively generates an exclusive matrix from an arbitrary finite SRL theory such that the denotations of the clauses of the exclusive matrix constitute the classes of a classification that has qualities (Q1) and (Q2).

3 Index

Having established that a classification can be effectively generated from a finite theory, we now turn to the problem of effectively generating an index for that classification. In a trivial sense, each exclusive matrix *is* an index, since one can classify an object by asking of each description in each clause "Is the object in the denotation of the description?" until one finds a clause all of whose descriptions denote the object. However, while creating such an index is extremely efficient (it is essentially free), using it is ludicrously inefficient. Here we present a better – albeit still extremely naïve – index generating device.

Suppose that $\mathbb{S} = \{\sigma_1, \ldots, \sigma_p\}$, μ is an exclusive matrix, $I = \langle U, S, F\rangle$ is an interpretation, and $\upsilon \in M_I(\mu)$. Call a clause that could possibly denote υ **active**. Informally, our device generates an index that first assumes all clauses in μ are active, then poses a series of queries, the responses to which properly decrease the number of active clauses in μ until a single clause remains active. This last active clause is the unique clause that denotes υ. In order to reduce the number of queries the index must pose before arriving at a single active clause, each query is chosen with careful regard to the currently active clauses so as to minimise the greatest number of clauses that remain active after a response to the query has been made. The choice of query is determined automatically.

The index poses two kinds of queries about υ: (i) **species queries** of the form "What is $S(T_I(\tau)(\upsilon))$?", where $\tau \in \mathsf{Term}(\mu)$, and (ii) **equation queries** of the form "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?", where $\tau_1 \in \mathsf{Term}(\mu)$ and $\tau_2 \in \mathsf{Term}(\mu)$. There are $p+1$ possible responses to query "What is $S(T_I(\tau)(\upsilon))$?", namely "σ_1", …, "σ_p" and "$T_I(\tau)(\upsilon)$ is undefined", and there are four possible responses to query "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?", namely "yes", "no"[3], "$T_I(\tau_1)(\upsilon)$ is undefined" and "$T_I(\tau_2)(\upsilon)$ is undefined". Notice that since both $\mathbb{S}$ and $\mathsf{Term}(\mu)$ are finite, there are only finitely many queries.

2 Lemma. *For each exclusive matrix μ, for each $\theta \in \mu$, for each $\tau \in \mathsf{Term}(\mu)$, for each interpretation I, for each $\upsilon \in \Theta_I(\theta)$,*
$T_I(\tau)(\upsilon)$ is defined iff $\tau \approx \tau \in \theta$.

Given lemma 2, each response to a query renders certain clauses in μ inactive. Informally,
if the response to query "What is $S(T_I(\tau)(\upsilon))$?" is "σ"
then each active clause contains $\tau \sim \sigma$,
if the response to query "What is $S(T_I(\tau)(\upsilon))$?" is "$T_I(\tau)(\upsilon)$ is undefined"
then no active clause contains $\tau \approx \tau$,

[3] with both $T_I(\tau_1)(\upsilon)$ defined and $T_I(\tau_2)(\upsilon)$ defined

if the response to query "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "yes"
then each active clause contains $\tau_1 \approx \tau_2$,
if the response to query "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "no"
then each active clause contains $\tau_1 \not\approx \tau_2$,
if the response to query "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "$T_I(\tau_1)(\upsilon)$ is undefined"
then no active clause contains $\tau_1 \approx \tau_1$, and
if the response to species query "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "$T_I(\tau_2)(\upsilon)$ is undefined"
then no active clause contains $\tau_2 \approx \tau_2$.

Formally, for each $\mu' \subseteq \mu$, for each $\tau \in \mathsf{Term}(\mu)$, let $\mathsf{Yield}\langle \mu', \text{"What is } S(T_I(\tau)(\upsilon))\text{?"}\rangle = \langle \mu_1, \ldots, \mu_{p+1} \rangle$, where (i) for each $1 \leq i \leq p$, $\mu_i = \{\theta \in \mu' \mid \tau \sim \sigma_i \in \theta\}$, and (ii) $\mu_{p+1} = \{\theta \in \mu' \mid \tau \approx \tau \notin \theta\}$. Then,
if $\upsilon \in M_I(\mu')$ and the response to "What is $S(T_I(\tau)(\upsilon))$?" is "σ_i"
then $\upsilon \in M(\mu_i)$, and
if $\upsilon \in M_I(\mu')$ and the response to "What is $S(T_I(\tau)(\upsilon))$?" is "$T_I(\tau)(\upsilon)$ is undefined"
then $\upsilon \in M(\mu_{p+1})$.

Similarly, for each $\mu' \subseteq \mu$, for each $\tau_1 \in \mathsf{Term}(\mu)$, for each $\tau_2 \in \mathsf{Term}(\mu)$, let $\mathsf{Yield}\langle \mu', \text{"Does } T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)\text{?"}\rangle = \langle \mu_1, \mu_2, \mu_3, \mu_4 \rangle$, where $\mu_1 = \{\theta \in \mu' \mid \tau_1 \approx \tau_2 \in \theta\}$, $\mu_2 = \{\theta \in \mu' \mid \tau_1 \not\approx \tau_2 \in \theta\}$, $\mu_3 = \{\theta \in \mu' \mid \tau_1 \approx \tau_1 \notin \theta\}$, and $\mu_4 = \{\theta \in \mu' \mid \tau_2 \approx \tau_2 \notin \theta\}$. Then,
if $\upsilon \in M_I(\mu')$ and the response to "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "yes"
then $\upsilon \in M(\mu_1)$,
if $\upsilon \in M_I(\mu')$ and the response to "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "no"
then $\upsilon \in M(\mu_2)$,
if $\upsilon \in M_I(\mu')$ and the response to "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "$T_I(\tau_1)(\upsilon)$ is undefined"
then $\upsilon \in M(\mu_3)$, and
if $\upsilon \in M_I(\mu')$ and the response to "Does $T_I(\tau_1)(\upsilon) = T_I(\tau_2)(\upsilon)$?" is "$T_I(\tau_2)(\upsilon)$ is undefined"
then $\upsilon \in M(\mu_4)$.

For each $\mu' \subseteq \mu$, for each query q, if $\mathsf{Yield}\langle \mu', q \rangle = \langle \mu_1, \ldots, \mu_n \rangle$ then let $\mathsf{Weight}\langle \mu', q \rangle = \mathsf{Max}(\{|\mu_i| \mid 1 \leq i \leq n\})$. Thus, $\mathsf{Weight}\langle \mu', q \rangle$ is the greatest number of clauses in μ' that can remain active after q is responded to. For each $\mu' \subseteq \mu$, let $\mathsf{Query}(\mu')$ be a query q such that $\mathsf{Weight}\langle \mu', q \rangle = \mathsf{Min}(\{\mathsf{Weight}\langle \mu', q' \rangle \mid q' \text{ is a query}\}$. Thus, $\mathsf{Query}(\mu')$ is a query with the least greatest number of clauses in μ' that can remain active after the query is responded to. Moreover, Yield, Weight and Query can be determined effectively. Our index generation device exploits this last fact to automatically generate the following index.

1 Set $\mu' = \mu$
2 if μ' is a singleton $\{\theta\}$ then output θ and halt else go to step 3
3 pose $\mathsf{Query}(\mu')$
4 from the response to $\mathsf{Query}(\mu')$, set μ' to the appropriate member of $\mathsf{Yield}\langle \mu', \mathsf{Query}(\mu') \rangle$
5 go to step 2

A technical proof shows that step 4 of the index always reduces the size of μ'. Therefore, since μ is finite, the index always outputs the unique clause that denotes υ after finite time and posing at most $|\mu| - 1$ queries.

References

[Götz 1993] Thilo Götz. *A Normal Form Algorithm for King's Descriptive Formalism.* Term paper. Eberhard-Karls-Universität, Tübingen, Germany. 1993.

[Kepser 1994] Stephan Kepser. *A Satisfiability Algorithm for a Typed Feature Logic.* Masters thesis. Eberhard-Karls-Universität, Tübingen, Germany. 1994.

[King 1989] Paul John King. *A Logical Formalism for Head-Driven Phrase Structure Grammar.* Doctoral thesis. The University of Manchester, Manchester, England. 1989.

[King 1994] Paul John King. *An Expanded Logical Formalism for Head-Driven Phrase Structure Grammar.* Arbeitspapiere des SFB 340, number 59. Eberhard-Karls-Universität, Tübingen, Germany. 1994.

[Pollard and Sag 1987] Carl J. Pollard and Ivan A. Sag. *Information-Based Syntax and Semantics.* CSLI Lecture Notes, number 13. CSLI, Stanford, California, USA. 1987.

[Pollard and Sag 1994] Carl J. Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar.* University of Chicago Press, Chicago, Illinois, USA. 1994.