СПЕЦИАЛИЗИРАН НАУЧЕН СЪВЕТ ПО ИНФОРМАТИКА
И МАТЕМАТИЧЕСКО МОДЕЛИРАНЕ ПРИ ВАК

КИРИЛ ИВАНОВ СИМОВ

ЛОГИЧЕСКИ СРЕДСТВА ЗА ОБРАБОТКА НА ЛИНГВИСТИЧНИ ЗНАНИЯ В ОПОРНАТА ФРАЗОВА ГРАМАТИКА

АВТОРЕФЕРАТ

на дисертация за присъждане
на образователната и научна степен „Доктор"
по научна специалност 01.01.12 „Информатика"

София
2006

Дисертационният труд е обсъден и насочен за защита от разширено заседание на секция „Математическа логика" при Института по математика и информатика, Българска академия на науките.

Пълният обем на дисертацията е 145 страници. Използваната литература включва 81 заглавия. Списъкът от публикации на автора, отразяващи резултатите от дисертация, съдържа 10 заглавия.

Защитата на дисертационния труд ще се състои на 17.08.06 от 16:00 часа в Милтимедийната зала на ИМИ, БАН на открито заседание на Специализирания научен съвет по информатика и математическо моделиране при ВАК.

СПЕЦИАЛИЗИРАН НАУЧЕН СЪВЕТ ПО ИНФОРМАТИКА
И МАТЕМАТИЧЕСКО МОДЕЛИРАНЕ ПРИ ВАК

КИРИЛ ИВАНОВ СИМОВ

ЛОГИЧЕСКИ СРЕДСТВА ЗА ОБРАБОТКА НА ЛИНГВИСТИЧНИ ЗНАНИЯ В ОПОРНАТА ФРАЗОВА ГРАМАТИКА

АВТОРЕФЕРАТ
на дисертация за присъждане
на образователната и научна степен „Доктор"
по научна специалност 01.01.12 „Информатика"

Научен ръководител:
Ст. н. с. II ст. д-р Радослав Павлов
Рецензенти:
Доц. дмн Георги Тотков
Ст. н. с. II ст. д-р Иван Держански

София
2006

1 Обща характеристика на дисертационния труд

Обект на изследване в настоящата дисертация са логически методи за представяне и обработка на лингвистични знания в рамките на Опорната фразова граматика (ОФГ) (Head-driven Phrase Structure Grammar — [Pollard and Sag 1987] и [Pollard and Sag 1994]). Основният проблем, който се решава, е съвместяването на научната адекватност на избрания логически формализъм и практическата задача за описание на лингвистичните факти, както на теоретично, така и на емпирично равнище. По този начин разработените методи са в духа на съвременните тенденции в областта на автоматичната обработка на естествения език, комбиниращи частични анализи с детайлно лингвистично знание и, също така, базирани на големи емпирични бази данни — корпуси.

Целта, която си поставяме, се изразява в следното: **проучване на задачите, които възникват при емпиричното и теоретичното описание на лингвистичното знание, и разработване на логически методи за тяхното адекватно решение**. Основните задачи, свързани с представянето и обработката на лингвистичните знания, са следните:

О1. Представяне на граматика на естествения език. Лингвистична онтология, граматични принципи и лексикон.

О2. Представяне на структурата на изреченията на естествения език по отношение на граматиката на естествения език. Структура на езиковите обекти.

О3. Създаване на езикови ресурси. Частично и пълно описание на езиковите обекти по отношение на граматиката на естествения език.

О4. Използване на езиковите ресурси. Автоматично извличане на граматики. Адаптиране на езиковите ресурси с оглед на нови приложения и/или нови лингвистични модели.

В нашата разработка за езиков модел приемаме Опорната фразова граматика поради следните нейни преимущества:

- ОФГ е една от основните лингвистични теории, която е базирана на строги формални принципи.
- ОФГ позволява последователно описание на езиковите факти на всички езикови нива: синтактично, семантично, прагматично и т.н.
- ОФГ позволява представяния с различна степен на детайлност по отношение на езиковите факти и следователно — възможността за работа на различни експерти на различните нива на описание.

- Формалните основи на ОФГ позволяват превод в други граматически формализми.

- ОФГ е успешно използвана при моделиране на езици като английския, немския, френския, полския, японския, сръбския, хърватския и други. Опитът, натрупан при тези разработки, може да бъде използван при моделиране на други езици. В нашата разработка обект на изследване е българският език.

Като формална основа за ОФГ приемаме логиката "Speciate Re-entrant Logic" (SRL), представител на логиките с функционални атрибути. SRL е логика, разработена от Paul John King като логически формализъм за ОФГ — виж [King 1989], [King 1994] и [King 1999].

При така избраните езиков модел и логически формализъм и за постигането на формулираната по-горе цел е необходимо да бъдат решени следните конкретни задачи:

К1. Формулиране на единно представяне на граматиките и езиковите обекти с интерпретация в SRL (задачи О1 и О2).

К2. Разработване на алгоритъм за превръщане на всяка крайна теория в SRL в единното представяне (задачи О1 и О2).

К3. Разработване на алгоритъм за извод на базата на така формулираното представяне (задачи О1, О2 и О4).

К4. Дефиниране на корпус в ОФГ на базата на представянето, формулирано по-горе (задачи О2, О3 и О4).

К5. Разработване на алгоритми за класификация на езикови обекти (задачи О3 и О4).

К6. Разработване на алгоритми за извличане на граматики на базата на корпус в ОФГ (задача О4).

К7. Разработване на алгоритъм за адаптиране (рекласификация) на описания на езикови обекти за различни приложения (задача О4).

Езиковите обекти са елементи на така наречената езикова реалност и следователно са външни за формалното моделиране на знанието за езика (задача К1). За да можем да ги описваме емпирично, имаме нужда от формален модел, който да опосредства връзката между езиковата реалност и граматиката или представеното знание. За целта формулираме единно представяне на граматиките и езиковите обекти като множество

от атрибутни графи, които, от една страна, могат да се разглеждат като абстракция над езиковите обекти, а от друга — имат интерпретация в SRL. Това единно представяне позволява директен достъп до емпирично представеното лингвистично знание.

За да представлява наистина база за представяне на ОФГ граматиките и езиковите обекти, предложеният формализъм трябва да позволява представянето на клас от SRL теории, който да включва в себе си езиково релевантните ОФГ граматики. Такъв клас е класът на крайните SRL теории над крайни SRL сигнатури. За този клас е разработен алгоритъм за трансформиране на всяка теория от него в горното представяне (задача К2). По този начин можем да разглеждаме представянето като нормална форма за този клас от теории.

За да може да бъде използвана граматиката, представена в така дефинираната нормална форма, има нужда да се дефинира система за извод на нейната база. По този начин могат да бъдат анализирани нови естественоезикови изречения или те да бъдат генерирани на базата на описанието на техните структури. Проблемът за извод в SRL е неразрешим и следователно методът за извод (задача К3) е непълен и непротиворечив. Предложена е стратегия за контрол над процеса на извод, която в определени случаи го прекратява с интерпретацията, че "не се знае" дали дадената формула за доказване е изводима от граматиката.

За да бъде подходяща за представяне на емпирично важното лингвистично знание, нашата нормална форма трябва да позволява изграждането на ОФГ корпуси. За целта ние дефинираме понятието "корпус, базиран на лингвистична теория" и показваме, че така дефинираната нормална форма е добро представяне на ОФГ базиран корпус (задача К4).

След като сме показали, че така дефинирата нормална форма и методите за извод над нея не ограничават класа от потенциални ОФГ граматики, преминаваме към разработването на алгоритми с практическа насоченост към създаването, поддръжката и използването на езикови ресурси, базирани на ОФГ. Тези алгоритми могат да се използват за: изграждане на лексикони по отношение на дадена ОФГ граматика; изграждане на корпус по отношение на дадена ОФГ граматика; поддържане на ефективно търсене в нормалната форма с цел ускоряване на процеса на извод; извличане на граматики от ОФГ базиран корпус, специализация на ОФГ граматика по отношение на ОФГ корпус, адаптация на ОФГ корпус към нова ОФГ граматика при промяна на лингвистичното знание в оригиналната ОФГ граматика или за ново приложение.

За решаването на тези задачи са разработени следните алгоритми:

- Алгоритми за класификация на езиковите обекти по отношение на

елементите на дадена нормална форма (задача К5).

- Алгоритми за извличане на граматики на базата на корпус в ОФГ. Използването им за специализация на ОФГ граматики по отношение на ОФГ корпус (задача К6).

- Алгоритъм за повторна класификация на вече класифицирани езикови обекти по отношение на нова нормална форма. Използване на този алгоритъм за адаптиране на описания на езикови обекти за различни приложения (задача К7).

Алгоритмите за класификация разглеждат нормалната форма като класификационна схема и използват различни по тип индекси над нея за бързо намиране на елемент от нормалната форма, която описва класифицирация се езиков обект. Знанието за характеристиките на езиковия обект е външно по отношение на алгоритъма и се предоставя от така наречения оракъл, който може да бъде потребителят на системата или друг алгоритъм. Освен за класификация на езикови обекти, представените алгоритми могат да се използват и за класификация на частични описания на езикови обекти. Така на тях може да се разчита в процеса на извод за намиране на подходящи клаузи за удовлетворяване на текущата формула за извод. Тези алгоритми се използват и при изграждане на корпус по отношение на дадена ОФГ граматика.

Един корпус има много приложения — от чисто лингвистични изследвания до приложения за автоматична обработка на текстове на естествен език. Тук е представен алгоритъм за извличане на граматики от ОФГ базиран корпус. Извлечените граматики могат да бъдат използвани за анализиране на нови изречения или като база за прецизиране (специализация) на вече съществуваща граматика. Предложеният алгоритъм позволява по-нататъшно специализиране с оглед на извличане на граматики, които удовлетворяват допълнителни условия. Един отворен проблем е дали граматики, извлечени от корпуси в SRL, правят процедурата за извод да е пълна.

Всеки корпус се изгражда с определена цел, но от друга страна, той е твърде скъп продукт и неговите разработчици трябва да се стремят да го направят многофункционален. За целта той трябва да позволява адаптиране към различни задачи. Адаптацията означава автоматично трансформиране към друга концептуализация на лингвистичното знание, представена като различна ОФГ граматика. Новата ОФГ граматика има нова нормална форма, следователно корпусът би имал нов вид спрямо нея. Процесът на адаптация на корпуса спрямо новата нормална форма

наричаме "повторна класификация" (или рекласификация) на езиковите обекти от корпуса по отношение на новите анализи с оглед на новата нормална форма. Алгоритъмът за повторна класификация е дефиниран с помощта на правила за трансфер на знание от корпуса спрямо старата граматика към корпуса спрямо новата граматика.

Структурата на дисертацията е както следва: глава втора прави обзор на други разработки в областта; глава трета представя основните понятия и принципи на ОФГ; глава четвърта въвежда SRL като логически формализъм за ОФГ; в глава пета се дефинира нормалната форма, която използваме като основно представяне на лингвистичното знание, представя се процедурата за извод, която използва тази нормална форма; глава шеста представя алгоритми за класификация над SRL теория, преобразувана в нормална форма; глава седма разглежда представянето на лингвистична информация в ОФГ; последната глава обобщава резултатите от дисертацията.

2 Глава втора: Представяне и обработка на лингвистично знание

Във втора глава е направен кратък обзор на софтуерните средства за създаване на корпуси. Разгледани са системите CLaRK, GATE и TagLog. Първите две са представители съответно на адитивните и на референтните системи . Първият вид системи съхраняват интерпретацията на текста чрез записване на маркираща информация в самия текст. Вторият вид системи запазват текста без промяна, а интерпретацията е записана отделно и реферира към текста с указатели. TagLog е представител на абстрактните системи. Тези системи представят текста само като част от интерпретацията му. При TagLog интерпретацията и текста се представят като логически теории в едно разширение на Prolog (хорнови клаузи). Идеята, стояща в основата на TagLog, един и същи формализъм да бъде използван както за разработката на граматики на естествения език, така и за интерпретиране на емпиричен материал, е много обещаваща и перспективна. Двете дейности са свързани в много отношения. На първо място всяка интерпретация има нужда от дефиниране на основните елементите на интерпретиране. Най-добре е Това дефиниране да стане в духа на идеите на определена лингвистична теория. Основната критика към подхода на системата TagLog е, че формализмът, стоящ зад нея, не е пряко ориентиран към лингвистичните теории. Друга критика е, че макар и да може определени операции да се дефинират в рамките на

една логическа теория, то много често има други по-ефективни методи за разработка на тези операции. Затова нашата цел е да разработим една система, която поддържа целия цикъл на разработване на един корпус. В тази система бихме искали да има възможност за вграждане на различни съществуващи езикови модули (морфологични анализатори, частични граматики, например), но, от друга страна, резултатът от тази дейност да има логическа интерпретация и да бъде съобразена с дадена лингвистична теория. Обект на нашата работа е дефинирането на логическите средства за това ниво, разработването на изводи в една логика за ОФГ, които да позволят извършването на дейностите, свързани с разработката и използването на един корпус.

3 Глава трета: Опорна фразова граматика

В тази глава представяме накратко основните характеристики и принципи на лингвистична теория **Опорна фразова граматика** (ОФГ) (Head-driven Phrase Structure Grammar — виж [Pollard and Sag 1994]). В основата на ОФГ стои разбирането, че езикът се състои от *езикови обекти*, които оформят езиковата действителност. Езиковата действителност е интерпретация на граматиката на даден език. Езиковата теория се абстрахира от езиковата действителност, разглеждайки езика като система от типове, които отразяват свойствата и взаимовръзките на езиковите обекти. Тези типове се представят чрез атрибутни структури с функционални атрибути. Една ОФГ граматика се състои от лингвистична онтология, представена като *йерархия на сортове* и *граматични принципи* и *лексикон*, представени като *граматична теория*. Йерархията на сортовете определя видовете езикови обекти и основните им свойства. Граматичната теория е множество от ограничения, дефинирани над йерархията на сортове. Те определят структурата на езиковите обекти в езика. Това става, като от обектите, които се допускат от йерархията на сортовете, се избират онези, които удовлетворяват ограниченията на теорията. Един от основните механизми за определяне на структурата на лингвистичните обекти е *структурното съвместяване* (*reentrance* или *structure sharing*). Структурното съвместяване означава, че даден езиков обект е свързан с друг езиков обект чрез две различни характеристики, които се представят чрез последователност от атрибути. ОФГ е лексикалистка лингвистична теория, тъй като основните ограничения се задават към думите на дадения език в частта на граматичната теория, която наричаме *лексикон*.

4 Глава четвърта: Логически формализъм за ОФГ

В тази глава въвеждаме логическия формализъм за Опорната фразова граматика — логиката Speciate Re-entrant Logic (SRL), представител на логиките с функционални атрибути. SRL е логика, разработена от Paul John King като логически формализъм за Head-driven Phrase Structure Grammars (виж [King 1989], [King 1994], [King 1999]). След въвеждане на логиката и дефинирането на основните дедуктивни проблеми показваме, че основният дедуктивен проблем, необходим при формализирането на лингвистично знание — проблемът за цялостен модел на теория, е неразрешим. Това означава, че трябва да се използва система за извод, която е или непълна, или противоречива. В една от следващите глави ние дефинираме такава система, която е непротиворечива, но непълна.

Дефиниция 1 (Крайна сигнатура) $\Sigma = \langle \mathcal{SP}, \mathcal{FE}, \mathcal{AP} \rangle$ *е крайна сигнатура, тогава и само тогава, когато*
$\mathcal{SP}$ *е крайно множество от породи,*
$\mathcal{FE}$ *е крайно множество от атрибути и*
$\mathcal{AP}$ *е навсякъде определена функция на съвместимост от* $\mathcal{SP} \times \mathcal{FE}$ *в* $Pow(\mathcal{SP})$. □

Дефиниция 2 (Интерпретация) *Тройката* $\mathcal{IN} = \langle \textsc{Un}, \textsc{Si}, \textsc{Fi} \rangle$ *е интерпретация, тогава и само тогава, когато*
$\textsc{Un}$ *е множество от обекти на* $\mathcal{IN}$,
$\textsc{Si}$ *е навсякъде определена функция от* $\textsc{Un}$ *в* $\mathcal{SP}$ *(*$\textsc{Si}: \textsc{Un} \rightarrow \mathcal{SP}$*) и*
$\textsc{Fi}$ *е навсякъде определена функция от* $\mathcal{FE}$ *в множеството от частични функции от* $\textsc{Un}$ *в* $\textsc{Un}$ *(*$\textsc{Fi}: \mathcal{FE} \rightarrow \{\textsc{Un} \rightharpoonup \textsc{Un}\}$*),*
така че
за всеки атрибут $\phi \in \mathcal{FE}$, *и всеки обект* $v \in \textsc{Un}$,
$\textsc{Fi}(\phi)(v)$ *е дефинирано тогава и само тогава, когато*
$\mathcal{AP}(\textsc{Si}(v), \phi) \neq \emptyset$ *и*
ако $\textsc{Fi}(\phi)(v)$ *е дефинирано, то* $\textsc{Si}(\textsc{Fi}(\phi)(v)) \in \mathcal{AP}(\textsc{Si}(v), \phi)$. □

Дефиниция 3 (Термове) *Терм е всеки елемент на най-малкото множество* $\mathcal{TE}$, *такова че*
$: \in \mathcal{TE}$ *и*
за всеки атрибут $\phi \in \mathcal{FE}$ *и всеки терм* $\tau \in \mathcal{TE}$, *то* $\tau\phi \in \mathcal{TE}$. □

Термът : се интерпретира като идентитета върху основното множество на интерпретация, а другите термове като композиция на частични функции. Следва дефиницията на интерпретацията на термове.

Дефиниция 4 (Интерпретационна функция на термове) *За всяка интерпретация* $\mathcal{IN} = \langle \mathrm{Un}, \mathrm{Si}, \mathrm{Fi} \rangle$, $[\![]\!]^{\mathcal{IN}}$ *е навсякъде определена функция от* $\mathcal{TE}$ *в множеството от частични функции от* Un *в* Un, *така че за всеки обект* $v \in \mathrm{Un}$, *за всеки терм* $\tau \in \mathcal{TE}$, *за всеки атрибут* $\phi \in \mathcal{FE}$,

$[\![:]\!]^{\mathcal{IN}}(v)$ *е дефинирано и* $[\![:]\!]^{\mathcal{IN}}(v) = v$,

$[\![\tau\phi]\!]^{\mathcal{IN}}(v)$ *е дефинирано, тогава и само тогава, когато* $[\![\tau]\!]^{\mathcal{IN}}(v)$ *и* $\mathrm{Fi}(\phi)([\![\tau]\!]^{\mathcal{IN}}(v))$ *са дефинирани и ако* $[\![\tau\phi]\!]^{\mathcal{IN}}(v)$ *е дефинирано, то* $[\![\tau\phi]\!]^{\mathcal{IN}}(v) = \mathrm{Fi}(\phi)([\![\tau]\!]^{\mathcal{IN}}(v))$. □

Описание е синтактична конструкция, която е формирана от породи, термове и други символи в съответствие с правилата по-долу и се интерпретира като вярна или невярна за даден обект.

Дефиниция 5 (Описание) *Описание е всеки елемент на най-малкото множество* $\mathcal{DE}$ *такова, че за всяка порода* $\sigma \in \mathcal{SP}$, *за всеки терм* $\tau_1 \in \mathcal{TE}$, *за всеки терм* $\tau_2 \in \mathcal{TE}$, *за всяко описание* $\delta_1 \in \mathcal{DE}$, *за всяко описание* $\delta_2 \in \mathcal{DE}$,

$\tau_1 \sim \sigma \in \mathcal{DE}$, $\tau_1 \approx \tau_2 \in \mathcal{DE}$, $\tau_1 \not\approx \tau_2 \in \mathcal{DE}$[1],

$\neg\delta_1 \in \mathcal{DE}$, $(\delta_1 \wedge \delta_2) \in \mathcal{DE}$, $(\delta_1 \vee \delta_2) \in \mathcal{DE}$ *и* $(\delta_1 \rightarrow \delta_2) \in \mathcal{DE}$. □

Описания от вида $\tau \sim \sigma$, $\tau_1 \approx \tau_2$, $\tau_1 \not\approx \tau_2$ ще наричаме *атоми*. Всяко описание от вида $\tau \sim \sigma$, $\tau_1 \approx \tau_2$, $\tau_1 \not\approx \tau_2$, $\neg\tau \sim \sigma$, $\neg\tau_1 \approx \tau_2$ и $\neg\tau_1 \not\approx \tau_2$ се нарича *литерал*. Интерпретацията на описание се дефинира чрез множествата от обекти, за които даденото описание е вярно.

Дефиниция 6 (Денотационна функция на описания)
За всяка интерпретация $\mathcal{IN} = \langle \mathrm{Un}, \mathrm{Si}, \mathrm{Fi} \rangle$, $[\![.]\!]^{\mathcal{IN}}$ *е навсякъде определена функция от* $\mathcal{DE}$ *в* $Pow(\mathrm{Un})$ *такава, че*

за всяка порода $\sigma \in \mathcal{SP}$, *за всеки терм* $\tau_1 \in \mathcal{TE}$, *за всеки терм* $\tau_2 \in \mathcal{TE}$, *за всяко описание* $\delta_1 \in \mathcal{DE}$, *за всяко описание* $\delta_2 \in \mathcal{DE}$,

$[\![\tau \sim \sigma]\!]^{\mathcal{IN}} = \{v \in \mathrm{Un} \mid [\![\tau]\!]^{\mathcal{IN}}(v)$ *е дефинирано и* $\mathrm{Si}([\![\tau]\!]^{\mathcal{IN}}(v)) = \sigma\}$,

$[\![\tau_1 \approx \tau_2]\!]^{\mathcal{IN}} = \{v \in \mathrm{Un} \mid [\![\tau_1]\!]^{\mathcal{IN}}(v)$ *и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *са дефинирани и* $[\![\tau_1]\!]^{\mathcal{IN}}(v) = [\![\tau_2]\!]^{\mathcal{IN}}(v)\}$,

$[\![\tau_1 \not\approx \tau_2]\!]^{\mathcal{IN}} = \{v \in \mathrm{Un} \mid [\![\tau_1]\!]^{\mathcal{IN}}(v)$ *и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *са дефинирани и* $[\![\tau_1]\!]^{\mathcal{IN}}(v) \neq [\![\tau_2]\!]^{\mathcal{IN}}(v)\}$,

$[\![\neg\delta_1]\!]^{\mathcal{IN}} = (\mathrm{Un} \setminus [\![\delta_1]\!]^{\mathcal{IN}})$,

$[\![(\delta_1 \wedge \delta_2)]\!]^{\mathcal{IN}} = [\![\delta_1]\!]^{\mathcal{IN}} \cap [\![\delta_2]\!]^{\mathcal{IN}}$, $[\![(\delta_1 \vee \delta_2)]\!]^{\mathcal{IN}} = [\![\delta_1]\!]^{\mathcal{IN}} \cup [\![\delta_2]\!]^{\mathcal{IN}}$ *и*

$[\![(\delta_1 \rightarrow \delta_2)]\!]^{\mathcal{IN}} = (\mathrm{Un} \setminus [\![\delta_1]\!]^{\mathcal{IN}}) \cup [\![\delta_2]\!]^{\mathcal{IN}}$. □

За всеки обект от денотацията на дадено описание казваме, че обектът *удовлетворява* описанието. Една интерпретация *удовлетворява* дадено описание, ако описанието има непразна денотация в тази интерпретация.

[1]Описания от вида $\tau_1 \not\approx \tau_2$ не са част от оригиналната дефиниция на логиката, но са удобни за изразяване на нормалните форми в нея.

Всяко множество от описания $\theta \subseteq \mathcal{DE}$ се нарича *SRL теория* или просто теория. Разширяваме действието на денотационната функция над теории. Нека θ е теория, тогава $[\![\theta]\!]^{\mathcal{IN}} = \{v \in \text{Un} \mid \forall \delta \in \theta, v \in [\![\delta]\!]^{\mathcal{IN}}\}$.

Обикновено с оглед на лингвистични приложения интерес представляват следните дедуктивни проблеми:

1. **Непротиворечивост на теория**: За всяка теория $\theta \subseteq \mathcal{DE}$ ще казваме, че θ е *непротиворечива* тогава и само тогава, когато за някоя интерпретация $\mathcal{IN}$, $[\![\theta]\!]^{\mathcal{IN}} \neq \emptyset$.

2. **Непротиворечивост на описание по отношение на теория**: Нека $\theta \subseteq \mathcal{DE}$ бъде теория и $\delta \in \mathcal{DE}$ бъде описание. Ще казваме, че δ е *непротиворечиво по отношение на* θ тогава и само тогава, когато съществува интерпретация $\mathcal{IN}$ такава, че $[\![\theta]\!]^{\mathcal{IN}} \neq \emptyset$ и $[\![\delta]\!]^{\mathcal{IN}} \neq \emptyset$.

3. **Съгласуваност на описание с теория**: Нека $\theta \subseteq \mathcal{DE}$ бъде теория и $\delta \in \mathcal{DE}$ бъде описание. Ще казваме, че δ е *съгласувано с* θ тогава и само тогава, когато за всяка интерпретация $\mathcal{IN}$ такава, че ако $[\![\delta]\!]^{\mathcal{IN}} \neq \emptyset$, то $[\![\theta]\!]^{\mathcal{IN}} \supseteq [\![\delta]\!]^{\mathcal{IN}}$.

4. **Логическо следствие**: Нека $\theta \subseteq \mathcal{DE}$ бъде теория и $\delta \in \mathcal{DE}$ бъде описание. Ще казваме, че δ *логически следва от теорията* θ тогава и само тогава, когато за всяка интерпретация $\mathcal{IN}$ имаме, че $[\![\theta]\!]^{\mathcal{IN}} \subseteq [\![\delta]\!]^{\mathcal{IN}}$.

5. **Пълно описание на област**: Нека $\theta \subseteq \mathcal{DE}$ бъде теория и $\mathcal{IN} = \langle \text{Un}, \text{Si}, \text{Fi} \rangle$ бъде интерпретация. Ще казваме, че θ *пълно описва дадена област* Un тогава и само тогава, когато интерпретацията $\mathcal{IN} = \langle \text{Un}, \text{Si}, \text{Fi} \rangle$ е такава, че $[\![\theta]\!]^{\mathcal{IN}} = \text{Un}$. В такива случаи ще казваме, че интерпретацията е *цялостен модел* на теорията.

4.1 Неразрешимост на проблема за пълно описание на област

Неразрешимостта на проблема за пълно описание на област се доказва чрез свеждане на проблема за пълно описание на област към задача за построение на покритие на равнината с плочки, за която се знае, че е нерешима. Нека имаме n на брой типа плочки, така че страните на плочките от даден тип са маркирани с определен знак. И нека имаме една безкрайна в две посоки шахматна дъска (означена с $\mathbf{N}^2$). Задачата е да се определи дали съществува покритие на шахматната дъска с плочки от

даденитe типове, така че страните на всеки две съседни плочки да бъдат маркирани с един и същ знак.

Нека D е плочка, тогава ние ще пишем $Right(D)$ за знака на десния, $Left(D)$ за знака на левия, $Top(D)$ за знака на горния и $Bottom(D)$ за знака на долния ръб на плочката от тип D.

Плочките D_1 и D_2 са подобни тогава и само тогава, когато $Right(D_1) = Right(D_2)$, $Left(D_1) = Left(D_2)$, $Top(D_1) = Top(D_2)$ и $Bottom(D_1) = Bottom(D_2)$.

По естествен начин горните функции се дефинират и за съответните типове T: $Right(T) = Right(D)$, $Left(T) = Left(D)$, $Top(T) = Top(D)$ и $Bottom(T) = Bottom(D)$, където плочката D е от тип T.

Ако $\mathcal{L} = \langle T_0, \ldots, T_n \rangle$ е крайна непразна последователност от типове на плочки, то $\mathcal{L}$ *покритие на* $\mathbb{N}^2$ е навсякъде определена функция $\psi\colon \mathbb{N}^2 \to \{0, \ldots, n\}$ такава, че за всяка двойка $\langle i, j \rangle \in \mathbb{N}^2$, $Right(T_{\psi(i,j)}) = Left(T_{\psi(i+1,j)})$ и $Top(T_{\psi(i,j)}) = Bottom(T_{\psi(i,j+1)})$.

Една крайна, непразна последователност $\mathcal{L}$ от различни типове плочки *покрива* $\mathbb{N}^2$ тогава и само тогава, когато съществува $\mathcal{L}$ покритие на $\mathbb{N}^2$. Проблемът за покриване на $\mathbb{N}^2$ е *разрешим* тогава и само тогава, когато съществува процедура, която за всяка непразна последователност от типове плочки определя дали съществува покритие на $\mathbb{N}^2$, или не. Доказано е обаче че този проблем е неразрешим ([Harel 1983]).

Следва едно кодиране на този проблем като теория в SRL:

Нека

$\mathcal{SP} = \{node, true, false\}$,

$\mathcal{FE} = \{\mathsf{file}, \mathsf{rank}, \mathsf{type}, \mathsf{next}, \mathsf{earth}\}$,

$\mathcal{AP}\colon \mathcal{SP} \times \mathcal{FE} \to \mathcal{SP}^*$ се дефинира чрез:

$\mathcal{AP}(node, \mathsf{file}) = \mathcal{AP}(node, \mathsf{rank}) = \{node\}$,

$\mathcal{AP}(node, \mathsf{type}) = \{true, false\}$,

$\mathcal{AP}(node, \mathsf{next}) = \mathcal{AP}(node, \mathsf{earth}) = \emptyset$,

$\mathcal{AP}(true, \mathsf{file}) = \mathcal{AP}(true, \mathsf{rank}) = \mathcal{AP}(true, \mathsf{type}) = \emptyset$,

$\mathcal{AP}(true, \mathsf{next}) = \{true, false\}$,

$\mathcal{AP}(true, \mathsf{earth}) = \{node\}$,

$\mathcal{AP}(false, \mathsf{file}) = \mathcal{AP}(false, \mathsf{rank}) = \mathcal{AP}(false, \mathsf{type}) = \emptyset$,

$\mathcal{AP}(false, \mathsf{next}) = \{true, false\}$,

$\mathcal{AP}(false, \mathsf{earth}) = \{node\}$, и

$\Sigma = \langle \mathcal{SP}, \mathcal{FE}, \mathcal{AP} \rangle$.

Σ е крайна SRL сигнатура. За всяка крайна и непразна последователност $\mathcal{L} = \langle T_0, \ldots, T_n \rangle$ от типове на плочки, нека

$\theta^0_{\mathcal{L}} =: \mathsf{file}\ \mathsf{rank} \approx: \mathsf{rank}\ \mathsf{file}$,

$\theta^1_{\mathcal{L}} = \bigvee_{k \le n} : \mathsf{type}\ \mathsf{next}^k \sim true$,

$\theta^2_{\mathcal{L}} = \bigwedge_{k<l\leq n} (: \text{type next}^k \sim \mathit{false} \vee : \text{type next}^l \sim \mathit{false})$,

$\theta^3_{\mathcal{L}} = \bigvee_{\substack{k,l\leq n \\ Right(T_k)=Left(T_l)}} (: \text{type next}^k \sim \mathit{true} \wedge : \text{file type next}^l \sim \mathit{true})$,

$\theta^4_{\mathcal{L}} = \bigvee_{\substack{k,l\leq n \\ Top(T_k)=Bottom(T_l)}} (: \text{type next}^k \sim \mathit{true} \wedge : \text{rank type next}^l \sim \mathit{true})$, и

$\theta_{\mathcal{L}} = (: \sim \mathit{node} \rightarrow (\theta^0_{\mathcal{L}} \wedge \theta^1_{\mathcal{L}} \wedge \theta^2_{\mathcal{L}} \wedge \theta^3_{\mathcal{L}} \wedge \theta^4_{\mathcal{L}}))$.

За всяка крайна и непразна последователност $\mathcal{L}$ от типове на плочки $\theta_{\mathcal{L}}$ е описание над сигнатурата Σ. Доказваме следното твърдение:

Твърдение 7 *За всяка крайна и непразна последователност $\mathcal{L}$ от типове на плочки,*

$\mathcal{L}$ покрива $\mathbf{N}^2$ тогава и само тогава, когато
съществува интерпретация I на Σ, такава че
$\mathcal{IN}$ е цялостен модел на $\theta_{\mathcal{L}}$. □

На базата на това твърдение може да бъде доказана следната теорема.

Теорема 8 *Проблемът за пълно описание на област в SRL е неразрешим.* □

4.2 Нормални форми

Клауза е крайно (възможно е и празно) множество от литерали. *Матрица* е крайно (възможно е и празно) множество от клаузи. Множеството на всички матрици ще отбелязваме с $\mathcal{MA}$. Всяка клауза се интерпретира *конюнктивно*, тоест ако α е клауза, то $[\![\theta]\!]^{\mathcal{IN}} = \bigcap_{l\in\alpha} [\![l]\!]^{\mathcal{IN}}$. Всяка матрица се интерпретира *дизюнктивно*, тоест ако μ е матрица, то $[\![\mu]\!]^{\mathcal{IN}} = \bigcup_{\alpha\in\mu} [\![\alpha]\!]^{\mathcal{IN}}$. *Term* е функция от $\mathcal{MA}$ в $Pow(\mathcal{TE})$, такава че за всяка матрица $\mu \in \mathcal{MA}$,

$Term(\mu) = \{\tau \in \mathcal{TE} \mid$
така че за някоя клауза $\alpha \in \mu$, за някоя последователност $\omega \in \mathcal{FE}^*$,
за някоя порода $\sigma \in \mathcal{SP}$, $(\neg)\tau\omega \sim \sigma \in \alpha$, или
за някой терм $\tau' \in \mathcal{TE}$, $(\neg)\tau\omega \approx \tau' \in \alpha$, или
за някой терм $\tau' \in \mathcal{TE}$, $(\neg)\tau' \approx \tau\omega \in \alpha$, или
за някой терм $\tau' \in \mathcal{TE}$, $(\neg)\tau\omega \not\approx \tau' \in \alpha$, или
за някой терм $\tau' \in \mathcal{TE}$, $(\neg)\tau' \not\approx \tau\omega \in \alpha\}$.

Дефиниция 9 (Нормално множество от литерали) *Крайното множество от литерали α е нормално множество от литерали*[2] *тогава и само тогава, когато за него са изпълнени следните условия:*

[2]Без литерали от вида: $\tau_1 \not\approx \tau_2$ и $\neg\tau_1 \not\approx \tau_2$.

1. *ако* $\neg l \in \alpha$, *то* $l \notin \alpha$, *където* l *е атом.*

2. $:\approx: \in \alpha$;

3. *ако* $\tau_1 \approx \tau_2 \in \alpha$, *то* $\tau_2 \approx \tau_1 \in \alpha$;

4. *ако* $\tau_1 \approx \tau_2 \in \alpha$ *и* $\tau_2 \approx \tau_3 \in \alpha$, *то* $\tau_1 \approx \tau_3 \in \alpha$;

5. *ако* $\tau\phi \approx \tau\phi \in \alpha$, *то* $\tau \approx \tau \in \alpha$;

6. *ако* $\tau_1 \approx \tau_2 \in \alpha$, $\tau_1\phi \approx \tau_1\phi \in \alpha$ *и* $\tau_2\phi \approx \tau_2\phi \in \alpha$, *то* $\tau_1\phi \approx \tau_2\phi \in \alpha$;

7. *ако* $\tau \approx \tau \in \alpha$, *то* $\sigma \in \mathcal{SP}$, $\tau \sim \sigma \in \alpha$;

8. *Ако за някоя* $\sigma \in \mathcal{SP}$, $\tau \sim \sigma \in \alpha$, *то* $\tau \approx \tau \in \alpha$;

9. *ако* $\tau_1 \approx \tau_2 \in \alpha$, $\tau_1 \sim \sigma_1 \in \alpha$ *и* $\tau_2 \sim \sigma_2 \in \alpha$, *то* $\sigma_1 = \sigma_2$;

10. *ако* $\tau \sim \sigma_1 \in \alpha$ *и* $\tau\phi \sim \sigma_2 \in \alpha$, *то* $\sigma_2 \in \mathcal{AP}(\sigma_1, \phi)$;

11. *ако* $\tau \sim \sigma \in \alpha$, $\tau\phi \in Term(\alpha)$ *и* $\mathcal{AP}(\sigma, \phi) \neq \emptyset$, *то* $\tau\phi \approx \tau\phi \in \alpha$. □

Дефиниция 10 (Нормална матрица) *Матрицата* μ *е нормална тогава и само тогава, когато всички нейни клаузи са нормални множества от литерали.* □

Дефиниция 11 (Нормална форма) *Матрицата* μ *е нормална форма за описанието* δ *тогава и само тогава, когато* μ *е нормална матрица и* $[\![\mu]\!]^{\mathcal{IN}} = [\![\delta]\!]^{\mathcal{IN}}$ *за всяка интерпретация* $^{\mathcal{IN}}$. □

Следва описание на алгоритъм за построяване на нормалната форма на дадено описание. Първо описанието се преобразува в дизюнктивна нормална форма, която може да бъде представена като матрица, в която клаузите съответстват на дизюнктите в дизюнктивната нормална форма. След това върху така получената матрица се прилагат следните правила ([Götz 1993]):

(R0) $\mu \uplus \{\alpha \uplus \{\neg l, l\}\} \rightarrow \mu$,

(R1) $\mu \uplus \{\alpha\} \rightarrow \mu \cup \{\alpha \uplus \{: \approx :\}\}$,

(R2) $\mu \uplus \{\alpha \uplus \{\tau_1 \approx \tau_2\}\} \rightarrow \mu \cup \{\alpha \uplus \{\tau_1 \approx \tau_2\} \uplus \{\tau_2 \approx \tau_1\}\}$,

(R3) $\mu \uplus \{\alpha \uplus \{\tau_1 \approx \tau_2, \tau_2 \approx \tau_3\}\} \rightarrow \mu \cup \{\alpha \uplus \{\tau_1 \approx \tau_2, \tau_2 \approx \tau_3\} \uplus \{\tau_1 \approx \tau_3\}\}$,

(R4) $\mu \uplus \{\alpha \uplus \{\tau\phi \approx \tau\phi\}\} \rightarrow \mu \cup \{\alpha \uplus \{\tau\phi \approx \tau\phi\} \uplus \{\tau \approx \tau\}\}$,

(R5) $\mu \uplus \{\alpha \uplus \{\tau_1 \approx \tau_2, \tau_1\phi \approx \tau_1\phi, \tau_2\phi \approx \tau_2\phi\}\} \rightarrow$
$\mu \cup \{\alpha \uplus \{\tau_1 \approx \tau_2, \tau_1\phi \approx \tau_1\phi, \tau_2\phi \approx \tau_2\phi\} \uplus \{\tau_1\phi \approx \tau_2\phi\}\}$,

(R6) ако за всяка $\sigma \in \mathcal{SP}$, $\tau \sim \sigma \notin \alpha$, то
$\mu \uplus \{\alpha \uplus \{\tau \approx \tau\}\} \rightarrow \mu \cup \{\alpha \uplus \{\tau \approx \tau, \tau \sim \sigma\} \mid \sigma \in \mathcal{SP}\}$,

(R7) $\mu \uplus \{\alpha \uplus \{\tau \sim \sigma\}\} \rightarrow \mu \cup \{\alpha \uplus \{\tau \sim \sigma\} \uplus \{\tau \approx \tau\}\}$,

(R8) ако $\sigma_1 \neq \sigma_2$, то $\mu \uplus \{\alpha \uplus \{\tau_1 \approx \tau_2, \tau_1 \sim \sigma_1, \tau_2 \sim \sigma_2\}\} \rightarrow \mu$,

(R9) ако $\sigma_2 \notin \mathcal{AP}(\sigma_1, \phi)$, то $\mu \uplus \{\alpha \uplus \{\tau \sim \sigma_1, \tau\phi \sim \sigma_2\}\} \rightarrow \mu$,

(R10) ако $\tau\phi \in \mathit{Term}(\alpha)$ и $\mathcal{AP}(\sigma, \phi) \neq \emptyset$, то
$\mu \uplus \{\alpha \uplus \{\tau \sim \sigma\}\} \rightarrow \mu \cup \{\alpha \uplus \{\tau \sim \sigma\} \uplus \{\tau\phi \approx \tau\phi\}\}$,

Тук μ е матрица, α е клауза, $\tau, \tau_1, \tau_2, \tau_3$ са термове, $\sigma, \sigma_1, \sigma_2$ са породи, ϕ е атрибут, $\uplus$ е обединение на непресичащи се множества.

5 Глава пета: Дедукция в логическия формализъм за ОФГ

В тази глава въвеждаме специална нормална форма на крайна теория, която ще използваме за представяне на една напротиворечива, но непълна система за извод по отношение на проблема за цялостен модел на теория. Първо представяме алгоритъм, който трансформира всяка крайна теория, представена в логиката SRL, в *дизюнктивна нормална форма с изключващо "или"*. Тази нормална форма ще бъде представена като матрица така, че всяка клауза в матрицата да е удовлетворима и всеки две клаузи в матрицата да имат непресичащи се денотации. Всяка такава матрица ще бъде наричана *изключваща матрица*.

Дефиниция 12 (Изключваща матрица) *μ е* изключваща матрица *тогава и само тогава, когато $\mu \in \mathcal{MA}$, и за всяка $\alpha \in \mu$, за всяка $\sigma_1 \in \mathcal{SP}$, за всяка $\sigma_2 \in \mathcal{SP}$, за всеки $\phi \in \mathcal{FE}$, за всеки $\tau_1 \in \mathcal{TE}$, за всеки $\tau_2 \in \mathcal{TE}$, за всеки $\tau_3 \in \mathcal{TE}$, имаме*

1. *ако $l \in \alpha$, то l е атам,*
2. *$: \approx : \in \alpha$,*
3. *ако $\tau_1 \approx \tau_2 \in \alpha$, то $\tau_2 \approx \tau_1 \in \alpha$,*
4. *ако $\tau_1 \approx \tau_2 \in \alpha$ и $\tau_2 \approx \tau_3 \in \alpha$, то $\tau_1 \approx \tau_3 \in \alpha$,*
5. *ако $\tau_1\phi \approx \tau_1\phi \in \alpha$, то $\tau_1 \approx \tau_1 \in \alpha$,*
6. *ако $\tau_1 \approx \tau_2 \in \alpha$, $\tau_1\phi \approx \tau_1\phi \in \alpha$ и $\tau_2\phi \approx \tau_2\phi \in \alpha$, то $\tau_1\phi \approx \tau_2\phi \in \alpha$,*

7. $\tau \approx \tau \in \alpha$ *тогава и само тогава, когато за някоя* $\sigma \in \mathcal{SP}$, $\tau_1 \sim \sigma \in \alpha$,
8. *ако* $\tau_1 \approx \tau_2 \in \alpha$, $\tau_1 \sim \sigma_1 \in \alpha$ *и* $\tau_2 \sim \sigma_2 \in \alpha$, *то* $\sigma_1 = \sigma_2$,
9. *ако* $\tau_1 \sim \sigma_1 \in \alpha$, $\tau_1\phi \in Term(\mu)$ *и* $\mathcal{AP}(\sigma_1, \phi) \neq \emptyset$, *то* $\tau_1\phi \approx \tau_1\phi \in \alpha$,
10. *ако* $\tau_1 \sim \sigma_1 \in \alpha$ *и* $\tau_1\phi \sim \sigma_2 \in \alpha$, *то* $\sigma_2 \in \mathcal{AP}(\sigma_1, \phi)$,
11. *ако* $\tau_1 \not\approx \tau_2 \in \alpha$, *то* $\tau_1 \approx \tau_1 \in \alpha$ *и* $\tau_2 \approx \tau_2 \in \alpha$,
12. *ако* $\tau_1 \approx \tau_1 \in \alpha$ *и* $\tau_2 \approx \tau_2 \in \alpha$, *то* $\tau_1 \approx \tau_2 \in \alpha$ *или* $\tau_1 \not\approx \tau_2 \in \alpha$, *и*
13. $\tau_1 \approx \tau_2 \notin \alpha$ *или* $\tau_1 \not\approx \tau_2 \notin \alpha$. □

Ще представим алгоритъм, който трансформира всяко описание в семантично еквивалентна на него изключваща матрица. Алгоритъмът се състои от три компонента: `Matrix`, `Rewrite` и `Basic`. Първият компонент трансформира всяко описание в съответстваща му матрица, като първо трансформира описанието в дизюнктивна нормална форма. Третият компонент изтрива отрицателни литерали от нормализираната матрица. Основната работа се извършва от втория компонент, който прилага система от преписващи правила към матрицата така, че резултатната матрица е изключваща.

Първо, можем да използваме кой да е алгоритъм за трансформиране на формули на съждителната логика в семантично еквивалентна дизюнктивна нормална форма. И тъй като всяка крайна SRL теория $\{\delta_i \mid 1 \leq i \leq n\}$ е семантично еквивалентна на следното SRL описание $(\delta_1 \wedge \ldots \wedge \delta_n)$ и всяка дизюнктивна нормална форма $\bigvee_{i=1}^{m}(\bigwedge_{j=1}^{n_m} \delta_{i,j})$ е еквивалентна на матрицата $\{\{\delta_{i,j} \mid 1 \leq j \leq n_m\} \mid 1 \leq i \leq m\}$, то `Matrix` изчислява навсякъде определена функция от $FinPow(\mathcal{DE})$ в $\mathcal{MA}$, така че за всяка крайна теория $\theta \subseteq \mathcal{DE}$, за всяка интерпретация $\mathcal{IN}$, $[\theta]^{\mathcal{IN}} = [\texttt{Matrix}(\theta)]^{\mathcal{IN}}$.

Второ, релацията *ExclNorm* е най-малката бинарна релация $\rightarrow$ над $\mathcal{MA} \times \mathcal{MA}$, такава че за всяка $\mu \in \mathcal{MA}$, всяка $\alpha \in \mathcal{C}$, всяка $\sigma_1 \in \mathcal{SP}$, всяка $\sigma_2 \in \mathcal{SP}$, всеки $\phi \in \mathcal{FE}$, всеки $\tau_1 \in \mathcal{TE}$, всеки $\tau_2 \in \mathcal{TE}$, всеки $\tau_3 \in \mathcal{TE}$, и всеки атом δ, имаме че

*R*1. ако $\alpha \notin \mu$, $\delta \in \alpha$ и $\neg\delta \in \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu$,
*R*2. ако $\alpha \notin \mu$ и $: \approx : \notin \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{: \approx :\}\}$,
*R*3. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_2 \in \alpha$ и $\tau_2 \approx \tau_1 \notin \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_2 \approx \tau_1\}\}$,
*R*4. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_2 \in \alpha$, $\tau_2 \approx \tau_3 \in \alpha$ и $\tau_1 \approx \tau_3 \notin \alpha$, то
$\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1 \approx \tau_3\}\}$,
*R*5. ако $\alpha \notin \mu$, $\tau_1\phi \approx \tau_1\phi \in \alpha$ и $\tau_1 \approx \tau_1 \notin \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1 \approx \tau_1\}\}$,
*R*6. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_2 \in \alpha$, $\tau_1\phi \approx \tau_1\phi \in \alpha$, $\tau_2\phi \approx \tau_2\phi \in \alpha$ и $\tau_1\phi \approx \tau_2\phi \notin \alpha$, то
$\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1\phi \approx \tau_2\phi\}\}$,
*R*7. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_1 \in \alpha$ и за всяка $\sigma \in \mathcal{SP}$, $\tau_1 \sim \sigma \notin \alpha$, то
$\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1 \sim \sigma\} \mid \sigma \in \mathcal{SP}\}$,
*R*8. ако $\alpha \notin \mu$, $\tau_1 \sim \sigma_1 \in \alpha$ и $\tau_1 \approx \tau_1 \notin \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1 \approx \tau_1\}\}$,
*R*9. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_2 \in \alpha$, $\tau_1 \sim \sigma_1 \in \alpha$, $\tau_2 \sim \sigma_2 \in \alpha$ и $\sigma_1 \neq \sigma_2$, то
$\mu \cup \{\alpha\} \rightarrow \mu$,

$R10$. ако $\alpha \notin \mu$, $\tau_1 \sim \sigma_1 \in \alpha$, $\tau_1\phi \in Term(\mu \cup \{\alpha\})$,
$\mathcal{AP}(\sigma_1, \phi) \neq \emptyset$ и $\tau_1\phi \approx \tau_1\phi \notin \alpha$, то
$\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1\phi \approx \tau_1\phi\}\}$,
$R11$. ако $\alpha \notin \mu$, $\tau_1 \sim \sigma_1 \in \alpha$, $\tau_1\phi \sim \sigma_2 \in \alpha$ и $\sigma_2 \notin \mathcal{AP}(\sigma_1, \phi)$, то
$\mu \cup \{\alpha\} \rightarrow \mu$,
$R12$. ако $\alpha \notin \mu, \tau_1 \not\approx \tau_2 \in \alpha$ и $\tau_1 \approx \tau_1 \notin \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1 \approx \tau_1\}\}$,
$R13$. ако $\alpha \notin \mu, \tau_1 \not\approx \tau_2 \in \alpha$ и $\tau_2 \approx \tau_2 \notin \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_2 \approx \tau_2\}\}$,
$R14$. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_1 \in \alpha$, $\tau_2 \approx \tau_2 \in \alpha$, $\tau_1 \approx \tau_2 \notin \alpha$ и $\tau_1 \not\approx \tau_2 \notin \alpha$, то
$\mu \cup \{\alpha\} \rightarrow \mu \cup \{\alpha \cup \{\tau_1 \approx \tau_2\}, \alpha \cup \{\tau_1 \not\approx \tau_2\}\}$ и
$R15$. ако $\alpha \notin \mu$, $\tau_1 \approx \tau_2 \in \alpha$ и $\tau_1 \not\approx \tau_2 \in \alpha$, то $\mu \cup \{\alpha\} \rightarrow \mu$.

Тези правила са подобни на правилата за нормализация на матрици. За всяка матрица $\mu \in \mathcal{MA}$, μ е `Terminal` тогава и само тогава, когато за никоя матрица $\mu' \in \mathcal{MA}$, $\mu \rightarrow \mu'$. `Rewrite` изчислява навсякъде определена функция от $\mathcal{MA}$ в $\{\mathsf{Provisional}, \mathtt{Terminal}\} \times \mathcal{MA}$, такава че за всяка матрица $\mu \in \mathcal{MA}$,

ако за някоя матрица $\mu' \in \mathcal{MA}$, $\mu \rightarrow \mu'$,
то за някоя матрица $\mu' \in \mathcal{MA}$, $\mu \rightarrow \mu'$ и $\mathtt{Rewrite}(\mu) = \langle \mathsf{Provisional}, \mu' \rangle$,
иначе $\mathtt{Rewrite}(\mu) = \langle \mathtt{Terminal}, \mu \rangle$.

Трето, `Basic` изчислява навсякъде определена функция от $\mathcal{MA}$ в $\mathcal{MA}$, такава че за всяка матрица $\mu \in \mathcal{MA}$, $\mathtt{Basic}(\mu) = \{\alpha \setminus \{\neg l \mid \neg l \in \alpha\} \mid \alpha \in \mu\}$. `Class` е следният алгоритъм: за всяка крайна теория $\theta \subseteq \mathcal{DE}$, `Class` чете θ, прилага `Rewrite` рекурсивно към $\mathtt{Matrix}(\theta)$, докато `Rewrite` резултатът стане $\langle \mathtt{Terminal}, \mu \rangle$ за някоя матрица $\mu \in \mathcal{MA}$ и връща $\mathtt{Basic}(\mu)$.

Твърдение 13

`Class` *изчислява навсякъде определена функция от* $FinPow(\mathcal{DE})$ *в* $\mathcal{EX}$ *такава, че за всяка теория* $\theta \in FinPow(\mathcal{DE})$, *имаме че*
$[\![\theta]\!]^{\mathcal{IN}} = [\![\mathtt{Class}(\theta)]\!]^{\mathcal{IN}}$. □

Всички дедуктивни проблеми, описани в предишната глава, могат да се решат на базата на изключващите матрици, като се сравняват матриците на дадената теория и на описанието. Проблем остава построяването на цялостен модел на теория. В следващите раздели на тази глава е представен алгоритъм за контролирано построяване на цялостен модел на теория. Този алгоритъм отразява една непротиворечива, но непълна система за извод по отношение на определянето дали една теория има цялостен модел, или не. Този алгоритъм работи с представяне на теорията чрез графи. Тези графи съответстват на клаузите в изключващата матрица, която е построена на базата на теорията, а също така представляват и абстракция над обектите в дадена SRL интерпретация. Това ни позволява да представяме по еднакъв начин както знанието, представено

в дадена SRL теория, така и знанието за конкретни обекти в областта на интерпретация.

Дефиниция 14 (Атрибутен граф) *Нека* $\Sigma = \langle \mathcal{SP}, \mathcal{FE}, \mathcal{AP} \rangle$ *да бъде крайна сигнатура. Нека* $\prec$ *бъде линейна наредба над множеството от атрибути* $\mathcal{FE}$. *Насочен, свързан граф с корен* $\mathcal{GR} = \langle \mathcal{NO}, \mathcal{VE}, \rho, \mathcal{SA} \rangle$ *такъв, че*

$\mathcal{NO}$ *е непразно множество от* възли *(термове),*
$\mathcal{VE} : \mathcal{NO} \times \mathcal{FE} \rightharpoonup \mathcal{NO}$ *е частична* функция на преходите,
ρ *е* коренов възел (корен),
$\mathcal{SA} : \mathcal{NO} \to \mathcal{SP}$ *е навсякъде определена* функция за приписване на породи *такава, че*
за всеки $\nu_1, \nu_2 \in \mathcal{NO}$ *и всеки* $\phi \in \mathcal{FE}$
ако $\mathcal{VE}(\nu_1, \phi)$ *е дефинирано и* $\mathcal{VE}(\nu_1, \phi) = \nu_2$, *то*
$\mathcal{SA}(\nu_2) \in \mathcal{AP}(\mathcal{SA}(\nu_1), \phi)$,

е атрибутен граф *по отношение на сигнатурата* Σ *тогава и само тогава, когато е изпълнено:*

1. $\rho = :$,
2. *за всеки възел* $\nu \in \mathcal{NO}$ *такъв, че* $\nu = : \phi_1 \dots \phi_n$, *то*
$\phi_1 \dots \phi_n$ *е път от корена* ρ *до* ν ($\mathcal{VE}(\rho, \phi_1 \dots \phi_n) = \nu$[3]) *и*
за всеки друг път $\phi'_1 \dots \phi'_m$ *от* ρ *до* ν ($\mathcal{VE}(\rho, \phi'_1 \dots \phi'_m) = \nu$),
за терма $: \phi'_1 \dots \phi'_m$ *е в сила:* $: \phi_1 \dots \phi_n \prec : \phi'_1 \dots \phi'_m$.

С $\mathcal{GRS}$ *ще бележим множеството на всички атрибутни графи.* □

Допълнително дефинираме следните понятия: *пълен атрибутен граф*, за който всички атрибути, подходящи за дадена порода, са представени за възлите с етикет тази порода; *подграф* на даден граф, започващ на даден път или възел в графа; *релацията включване* между два графа, използваща изоморфизъм от възлите на единия граф във възлите на другия; релацията *еквивалентност* на два графа; *унификатор* и *най-общ унификатор* на два графа; *обобщение* и *най-специфично обобщение* на два графа; *разширение на графа* $\mathcal{GR}_1 = \langle \mathcal{NO}_1, \mathcal{VE}_1, \rho_1, \mathcal{SA}_1 \rangle$ *чрез графа* $\mathcal{GR}_2 = \langle \mathcal{NO}_2, \mathcal{VE}_2, \rho_2, \mathcal{SA}_2 \rangle$ *за пътя* π; разширяване на тези дефиниции за повече от два графа.

Всеки краен атрибутен граф може да бъде представен като клауза и да бъде интерпретиран чрез нейната интерпретация: $[\![\mathcal{GR}]\!]^{\mathcal{IN}} = [\alpha_{\mathcal{GR}}]^{\mathcal{IN}}$.

[3] Където $\mathcal{VE}$ е функцията, разширена върху пътища.

За безкрайни графи интерпретацията се дефинира като сечение на интерпретациите на всички крайни графи, които включват даден граф: $[\![\mathcal{GR}]\!]^{\mathcal{IN}} = \cap_{\mathcal{GR} \sqsubset \mathcal{GR}', \mathcal{GR}' < \omega} [\alpha_{\mathcal{GR}'}]^{\mathcal{IN}}$. На базата на тези дефиниции можем да говорим за удовлетворим граф, логическо следствие на граф от множество графи, съгласуване на граф с множество от графи, цялостен модел на множество от графи. Показано е, че всеки граф е удовлетворим.

Клаузите в една изключваща матрица могат да бъдат представени като атрибутни графи. Нека μ да бъде изключваща матрица и нека $\alpha \in \mu$, тогава $\mathcal{GR}_\alpha = \langle \mathcal{NO}_\alpha, \mathcal{VE}_\alpha, \rho_\alpha, \mathcal{SA}_\alpha \rangle$ е атрибутен граф такъв, че

$\mathcal{NO}_\alpha = \{|\tau|_\alpha \mid \tau \approx \tau \in \alpha\}$ е множество от възли[4],

$\mathcal{VE}_\alpha : \mathcal{NO}_\alpha \times \mathcal{FE} \rightharpoonup \mathcal{NO}_\alpha$ е частична *функция на преходите* такава, че

$\mathcal{VE}_\alpha(|\tau_1|_\alpha, \phi)$ е дефинирано и $\mathcal{VE}_\alpha(|\tau_1|_\alpha, \phi) = |\tau_2|_\alpha$

тогава и само тогава, когато

$\tau_1 \approx \tau_1 \in \alpha, \tau_2 \approx \tau_2 \in \alpha$, $\phi \in \mathcal{FE}$, и $\tau_1 \phi \approx \tau_2 \in \alpha$,

ρ_α е коренов възел $|:|_\alpha$ или еквивалентно $:$, и

$\mathcal{SA}_\alpha : \mathcal{NO}_\alpha \rightarrow \mathcal{SP}$ е функция за *свързване с породи* такава, че

$\mathcal{SA}_\alpha(|\tau|_\alpha) = \sigma$ тогава и само тогава, когато $\tau \sim \sigma \in \alpha$.

Всеки граф $\mathcal{GR}_\alpha$ е насочен, свързан и има корен. За всяка клауза $\alpha \in \mu$, графът $\mathcal{GR}_\alpha$ е семантично еквивалентен на клаузата α.

Следващата стъпка е да свържем графите, съответстващи на клаузи в дадена изключваща матрица, с обектите на дадена интерпретация. Нека Σ да бъде крайна сигнатура, нека $\mathcal{IN}$ да бъде интерпретация на Σ и нека $\upsilon \in \mathrm{U_N}^{\mathcal{IN}}$ да бъде един обект от областта на интерпретация. *Компоненти на обекта* υ е следното множество от обекти

$Comp^{\mathcal{IN}}(\upsilon) = \{\upsilon' \in \mathrm{U_N}^{\mathcal{IN}} \mid \exists \tau \in \mathcal{TE}, [\tau]^{\mathcal{IN}}(\upsilon)$ е дефинирано и $\upsilon' = [\tau]^{\mathcal{IN}}(\upsilon)\}$.

На базата на компонентите на обект можем да конструираме представяне на обекта като граф. Нека υ да бъде обект в интерпретацията $\mathcal{IN}$, тогава конструираме следния граф (с евентуално безкраен брой възли)

$\mathcal{GR}_{\upsilon,\mathcal{IN}} = \langle \mathcal{NO}, \mathcal{VE}, \rho, \mathcal{SA} \rangle$ такъв, че

$\mathcal{NO} = \{\tau \mid [\tau]^{\mathcal{IN}}(\upsilon), [\tau]^{\mathcal{IN}}(\upsilon) = \upsilon'$,

термът τ е минималният, за който $[\tau]^{\mathcal{IN}}(\upsilon) = \upsilon'$,

и $\upsilon' \in Comp^{\mathcal{IN}}(\upsilon)$[5],

за всеки възел $\nu_j \in \mathcal{NO}$ имаме, че

$\mathcal{SA}(\nu_i) = \mathrm{SI}^{\mathcal{IN}}(\upsilon_i)$, където $\mathcal{GR}_{\upsilon,\mathcal{IN}}(\nu_i) = \upsilon_i$,

за всеки два възела $\nu_i, \nu_j \in \mathcal{NO}$, за всеки атрибут $\phi \in \mathcal{FE}$,

$\mathcal{VE}(\nu_i, \phi)$ е дефинирано и $\mathcal{VE}(\nu_i, \phi) = \nu_j$

---

[4] $|\tau|_\alpha$ е минималният терм от множеството на всички термове, които са равни на τ в клаузата: $|\tau|_\alpha = min(\{\tau' \mid \tau \approx \tau' \in \alpha\})$.

[5] Отново пишем $\nu, \nu_1, \ldots$ за възлите в графа и $\mathcal{GR}_{\upsilon,\mathcal{IN}}(\nu_i) = \upsilon_i$, за да свържем даден възел със съответния обект.

тогава и само тогава, когато
$\mathcal{GR}_{\upsilon,\mathcal{IN}}(\nu_i) = \upsilon_i$, $\mathcal{GR}_{\upsilon,\mathcal{IN}}(\nu_j) = \upsilon_j$, $\mathrm{Fl}^{\mathcal{IN}}(\phi)(\upsilon_i)$ е дефинирано и $\mathrm{Fl}^{\mathcal{IN}}(\phi)(\upsilon_i) = \upsilon_j$ и
$\rho =: .$

$\mathcal{GR}_{\upsilon,\mathcal{IN}}$ е пълен граф и $\upsilon \in [\mathcal{GR}_{\upsilon,\mathcal{IN}}]^{\mathcal{IN}}$. Имайки представяне както на теорията, така и на обектите от интерпретацията като атрибутни графи, дефинираме връзката между графите в представянето на теорията и графа, съответстващ на обект в цялостен модел на теорията. Използвайки свойството на изключващите матрици, че денотациите на техните клаузи не се непресичат, то на всеки обект от цялостния модел съответства точно една клауза от матрицата, тоест точно един граф от представянето на теорията. Ако разгледаме графа, съответстващ на един обект в цялостния модел, то за всеки възел в графа съществува точно един граф от представянето на теорията, който включва подграфа, започващ от този възел. Вярно е и обратното: ако можем да построим пълен граф с тези свойства, то той може да бъде използван за построяване на интерпретация, която е цялостен модел на теорията. Този факт е използван, за да се дефинира алгоритъм, който по дадена теория и даден граф заявка проверява дали съществува пълен граф с тези свойства, който допълнително се включва в графа заявка. Този алгоритъм представлява система за извод по отношение на цялостния модел на теория като семантика на логиката. Тъй като доказахме, че този проблем е неразрешим, то нашият алгоритъм е една непротиворечива, но непълна система за извод. Липсата на пълнота се изразява във факта, че за определени теории и графи заявки алгоритъмът спира с отговор `unknown`. В този случай не се знае дали съществува цялостен модел с желаните свойства, или не. За да представим алгоритъма, първо дефинираме представяне на теорията като множество от графи, което съдържа цялата информация от сигнатурата и понятието *атрибутен граф от тип гора*, което използваме при дефиниране на ограничението за спиране на извода при потенциално безкрайно изпълнение.

Нека $\Sigma = \langle \mathcal{SP}, \mathcal{FE}, \mathcal{AP} \rangle$ да бъде крайна сигнатура и θ да бъде крайна теория по отношение на тази сигнатура. То тогава нека имаме

$$\theta_\Sigma = (\bigvee_{\sigma \in \mathcal{SP}} (\bigwedge_{\mathcal{AP}(\sigma,\phi) \neq \emptyset, \phi \in \mathcal{FE}} (: \phi \approx: \phi))).$$

Наричаме теорията θ_Σ *сигнатурна теория за* Σ или просто — сигнатурна теория. Всяка интерпретация на Σ е цялостен модел на сигнатурната теория за Σ. Нека имаме $\theta^e = \theta \cup \theta_\Sigma$. Наричаме теорията θ^e *сигнатурно разширение на теорията* θ. От дефиницията на сигнатурната теория имаме, че за всяка интерпретация $\mathcal{IN}$ на Σ е изпълнено

$[\![\theta^e]\!]^{\mathcal{IN}} = [\theta]^{\mathcal{IN}}$. Нека матрицата $\mu = \{\alpha_1, \ldots, \alpha_n\}$ да бъде изключваща матрица, която е семантично еквивалентна на теорията θ^e. Нека множеството $\mathcal{GRS}^\theta = \{\mathcal{GR}_{\alpha_1}, \ldots, \mathcal{GR}_{\alpha_n}\}$ да бъде множеството от атрибутни графи, съответстващи на клаузите от изключващата матрица. Това множество от графи ще наричаме *графово представяне* на теорията θ. По този начин можем да работим само с множество от графи, без да има нужда да се консултираме със сигнатурата, матрицата или теорията. Всяко множество от графи, съответстващи на клаузи от дадена изключваща матрица, ще наричаме *изключващо множество от графи*.

Алгоритъмът започва с граф заявка и изключващо множество от графи и се опитва да разшири графа заявка до пълен граф с помощта на графи от изключващото множество графи. Алгоритъмът следи за разширенията на графа заявка с графи от множеството $\mathcal{GRS}^\theta$ и ако съществува подозрение, че алгоритъмът няма да завърши своята работа (тоест ще работи безкрайно дълго), то тогава алгоритъмът спира своята работа, без да даде отговор на въпроса: дали графът заявка е удовлетворим в цялостен модел на теорията, или не. Управляващата информация за това спиране е свързана с броя на новите възли, получени при прилагането на алгоритъма `Extend`. В случай че алгоритъмът не завърши своята работа, но броят на новите възли не надвишава определено фиксирано число $\mathcal{NL}$, то тогава след краен брой стъпки на алгоритъма ще бъде повторена същата конфигурация от нови възли както тази, от която започва работата на алгоритъма. В този случай ние бихме могли да използваме вече построеното разширение на началната конфигурация, за да разширим текущия граф до пълен атрибутен граф. Тогава новият алгоритъм връща стойност `true` и може да се докаже съществуването на цялостен модел с необходимите свойства, без да се построи експлицитно пълен атрибутен граф. Ако всички възможни разширения на графа заявка са ограничени по този начин и никое не може да бъде разширено до пълен атрибутен граф, то тогава алгоритъмът връща стойност `false`. Ако алгоритъмът разшири графа заявка до пълен атрибутен граф, то алгоритъмът също връща стойност `true` след краен брой стъпки. Ако броят на новите възли надхвърли стойността $\mathcal{NL}$, то тогава алгоритъмът спира, като връща стойност `unknown`. Стойността $\mathcal{NL}$ зависи от приложението и се определя емпирично. Друг важен момент е, че стойността $\mathcal{NL}$ може да е различна за различни входни данни и по този начин да има по-гъвкава стратегия за управление на извода.

Разширението на даден граф за даден възел чрез друг граф зависи преди всичко от породата на възела и съседните му възли. Можем да представим тези съседни възли като специален вид графи, които имат ня-

колко корена и може да не са свързани. Нека $\Sigma = \langle \mathcal{SP}, \mathcal{FE}, \mathcal{AP} \rangle$ да бъде крайна сигнатура. Насоченият граф $\mathcal{GRF} = \langle \mathcal{NO}, \mathcal{VE}, \{\rho_1, \ldots, \rho_k\}, \mathcal{SA} \rangle$ такъв, че

$\mathcal{NO}$ е множество от *възли*, които са индексирани термове,
$\mathcal{VE} : \mathcal{NO} \times \mathcal{FE} \rightharpoonup \mathcal{NO}$ частична *функция на преходите*,
$\{\rho_1, \ldots, \rho_k\}$ е множество от *възли корени*, където $\rho_1 =: 1, \ldots, \rho_k =: k$,
$\mathcal{SA} : \mathcal{NO} \to \mathcal{SP}$ е *функция за приписване на породи*,
за всеки възел ν в $\mathcal{NO}$
  съществува коренов възел ρ в $\{\rho_1, \ldots, \rho_k\}$ такъв, че
    съществува път от кореновия възел ρ до възела ν и
за всеки възел ν в $\mathcal{NO}$ е изпълнено, че
  ν е минималният индексиран терм,
    който води от някой коренов възел до възела ν,

ще наричаме *атрибутен граф от тип гора* спрямо сигнатурата Σ.

За атрибутните графи от тип гора дефинираме аналогични понятия: *включване на графи*, *унификатор*, *обобщение*. Разширяването на даден граф с множество от графи за множество от възли зависи от възлите, за които се извършва разширението и техните околности, тоест части от графа, до които стигат пътищата в графите, чрез които се разширява даденият граф. Възлите и техните околности представяме като атрибутен граф от тип гора, наречен *ограничителен граф от тип гора*. При своята работа алгоритъмът запазва тези ограничителни графи. Ако на даден етап от работата си алгоритъмът открие повторение на ограничителен граф, то той може да прекъсне своята работа с отговор `true`.

Алгоритъмът работи по следния начин. На всяка стъпка той избира един граф за разширяване от множеството графи, което в началото съдържа само графа заявка. Ако графът може да бъде разширен, се проверява дали разширенията съдържат пълен граф. Ако да, то такъв пълен граф съответства на цялостен модел на теорията, който удовлетворява графа заявка. Ако не, в разширението се оставят само онези графи, които спазват условието за брой на нови възли. Ако това множество не е празно, се повтаря същата процедура. Ако множеството е празно, се избира нов граф за разширение. Ако няма такъв нов граф, то цялостен модел не съществува. Ако на даден етап от работата си алгоритъмът открие повторение на ограничителен граф, то той може да прекъсне своята работа с отговор, че цялостен модел на теорията, който удовлетворява графа заявка съществува, но е потенциално безкраен и не се построява. Тук ще дадем алгоритъма. Основен компонент в него е функцията `Extend`. `Extend` съпоставя всяка наредена последователност $\langle \mathcal{GR}, \mathcal{NO}, \mathcal{LGRF}, \mathcal{GRS}^{\mu} \rangle$ в последователност *Exts* от наредени

тройки, където $\mathcal{GR}$ е граф, $\mathcal{NO}$ е непразна наредена последователност от възли в графа $\mathcal{GR}$, $\mathcal{LGRF}$ е списък от графи от тип гора и $\mathcal{GRS}^{\mu}$ е множество от графи. Всяка тройка $\langle \mathcal{GR}_e, \mathcal{NO}_n, \mathcal{LGRF}_n \rangle$ в $Exts$ е такава, че $\mathcal{GR}_e \in \texttt{MGE}(\mathcal{GR}, \mathcal{NO}, \mathcal{GRS}^{\mu}_{sub})$, $\mathcal{NO}_n$ са новите възли в графа $\mathcal{GR}_e$ и $\mathcal{LGRF}_n$ е последователност от графи от тип гора, която се формира чрез добавяне на ограничителните графи от тип гора за $\mathcal{GR}$, $\mathcal{NO}$, и $\mathcal{GRS}^{\mu}_{sub}$ към началото на списъка $\mathcal{LGRF}$.

Вторият компонент на алгоритъма е функция $\texttt{ControlMod}(\mathcal{GRS}^{\theta}, \mathcal{GR}_q)$, която съпоставя всяко изключващо множество от графи $\mathcal{GRS}^{\theta}$ и всеки граф $\mathcal{GR}_q$ с (1) `true`, ако съществува цялостен модел на множеството $\mathcal{GRS}^{\theta}$, в който графът $\mathcal{GR}_q$ е удовлетворим; (2) `false`, ако такъв цялостен модел не съществува; и (3) `unknown`, ако алгоритъмът не може да каже дали такъв цялостен модел съществува, или не.

```
ControlMod(GRS^θ, ⟨NO_q, VE_q, ρ_q, SA_q⟩):
  begin
   Exts := Extend(⟨NO_q, VE_q, ρ_q, SA_q⟩, NO_q, GRS^θ)
   if Exts = ⟨⟩
    then return false
    else return ConCover(Exts, GRS^θ, false)
   endif
  end
```

Третият елемент на алгоритъма е функцията $\texttt{ConCover}(Exts, \mathcal{GRS}^{\theta})$, която съпоставя всяка наредена тройка $\langle Exts, \mathcal{GRS}^{\theta}, TruthVal \rangle$ с (1) `true`, ако някой граф в някоя наредена тройка в $Exts$ може да бъде допълнен до цялостен модел на множеството $\mathcal{GRS}^{\theta}$; (2) `false`, ако за всяка наредена тройка в $Exts$ графът в тройката не може да бъде разширен до цялостен модел на множеството $\mathcal{GRS}^{\theta}$; и (3) `unknown`, ако броят на новите възли в разширението на даден граф е по-голям от определената граница $\mathcal{NL}$. За всяка наредена тройка $\langle Exts, \mathcal{GRS}^{\theta}, TruthVal \rangle$, $Exts$ е последователност от наредени тройки, върнати като стойност от функцията `Extend`, $\mathcal{GRS}^{\theta}$ е изключващо множество от графи и $TruthVal$ е `false` или `unknown`. Целта на аргумента $TruthVal$ е да се запомни дали за поне едно разширение на някой граф броят на новите възли е надвишил ограничението $\mathcal{NL}$.

```
ConCover(Exts, GRS^θ, TruthVal):
  begin
   if Terminal⟨Exts⟩
    then return true
    else
```

```
begin
 Exts := ⟨(GR,NO⁻, LGRF) | RestExts⟩
 if ‖NO⁻‖ > NL
  then
   if RestExts = ⟨⟩
    then return unknown
    else return ConCover(RestExts, GRS^θ, unknown)
   endif
  else
   begin
    Exts^GR := Extend(GR, NO⁻, LGRF, GRS^θ)
    if Exts^GR = ⟨⟩
     then
      if RestExts = ⟨⟩
       then return TruthVal
       else return ConCover(RestExts, GRS^θ, TruthVal)
      endif
     else
      if ConCover(Exts^GR, GRS^θ, TruthVal) = true
       then return true
       else return ConCover(RestExts, GRS^θ, TruthVal)
      endif
    endif
   end
  endif
 end
endif
end
```

Алгоритъмът `ControlMod` спира след краен брой стъпки по една от следните причини: (1) множеството на новите възли в графа е празно; или (2) новият ограничителен граф от тип гора е вече включен в списъка с такива графи за обработвания граф; или (3) не съществуват повече разширения на графа. В случаите (1) и (2) стойността е `true`, в случая (3) стойността е `false`, ако всички наредени тройки във всички проверени разширения съдържат по-малко от $\mathcal{NL}$ възела. Стойността е `unknown`, ако поне за една наредена тройка, броят на новите възли е по-голям от $\mathcal{NL}$. Ако стойността е `true`, то тогава съществува цялостен модел с желаните качества. Ако стойността е `false`, то такъв цялостен модел не съществува. И ако стойността е `unknown`, то тогава алгоритъмът не може

да вземе решение дали такъв цялостен модел съществува, или не.

Могат да се разработят и други алгоритми с други ограничения върху изпълнението с подобни свойства. Например, връщане на стойност unknown след определен брой стъпки или ограничаване на броя само на свързаните нови възли, тоест със свързани ограничителни графи.

6 Глава шеста: Класификационни схеми в SRL

В тази глава за класификационна схема използваме представянето на крайни SRL теории като изключващи множества от графи. Класификацията ще разбираме като процес на избирането на един или повече класове в една класификационна схема на базата на (непълно) описание на даден обект или обекти. Предлагаме специфичен механизъм за класификация над такива множества, който се основава на механизъм за индексиране над графите в множеството. Индексът кодира атомични описания, които се оценяват като верни или неверни за графите в множеството и избират съответно подмножество от графи. Следователно, използването на индекса е на основата на истинността на тези елементарни описания по отношение на класифицирания обект.

В приложения за обработка на естествен език класификацията може да се използва за поддържане на създаването на езикови ресурси, където потребителят класифицира лингвистични обекти по отношение на дадена класификационна схема. Друго типично приложение на класификацията е намирането на граф или графи при автоматично извършване на извод в SRL. Примерно, една от стъпките при алгоритмите за построяване на цялостен модел, описани в предишната глава, е да се изберат подходящи графи при разширението на съответния граф. Графите, измежду които трябва да бъдат избрани подходящите за разширение, могат да бъдат много голямо количество (за една реалистична ОФГ граматика само сортовете в сорт йерархията могат да бъдат с хиляди). Следователно, последователното търсене на графи е нереалистично. Едно решение на проблема е класификация на съответния подграф от ограничителния граф над множеството с графи.

Разгледани са два типа индекси над графите в дадена класификационна схема: частични и пълни. Общата идея и в двата случая е, че индексът е ориентиран граф, в който възлите са маркирани с въпроси или графи, а дъгите с отговори. Въпросите и отговорите отговарят на елементарни описания за обекта, който се класифицира, а графите съответстват на

класовете в дадена класификационна схема. При отговор на даден въпрос се редуцира броят на потенциалните класове, които описват обекта.

Нека $\Sigma = \langle \mathcal{SP}, \mathcal{FE}, \mathcal{AP} \rangle$ да бъде крайна сигнатура и θ да бъде крайна теория по отношение на тази сигнатура. Нека изключващото множество от графи $\mathcal{GRS}^\theta = \{\mathcal{GR}_{\alpha_1}, \ldots, \mathcal{GR}_{\alpha_n}\}$ да бъде графово представяне на теория θ. Графовото представяне на теорията θ ще наричаме *класификационна схема*, генерирана от теорията. Множеството на всички класификационни схеми ще отбелязваме с $\mathcal{CS}$.

Въпросите и отговорите представляват онези въпроси, които могат да бъдат задавани за обектите в една интерпретация, и техните отговори.

Дефиниция 15 (Въпроси и отговори)

$\mathcal{QU} = \{\boxed{\tau} \mid \tau \in \mathcal{TE}\} \cup \{\boxed{\tau_1, \tau_2} \mid \tau_1 \in \mathcal{TE} \text{ и } \tau_2 \in \mathcal{TE}\}$, *и*

$\mathcal{RE} = \{\boxed{\sigma} \mid \sigma \in \mathcal{SP}\} \cup \{\boxed{-}, \boxed{\approx}, \boxed{\not\approx}, \boxed{\pm}, \boxed{\mp}, \boxed{=}\}$.

Всеки елемент на множеството $\mathcal{QU}$ *наричаме* въпрос *и всеки елемент на множеството* $\mathcal{RE}$ *наричаме* отговор. *За всяка интерпретация* $\mathcal{IN} = \langle \mathrm{Un}, \mathrm{Si}, \mathrm{Fi} \rangle$, *за всеки обект* $v \in \mathrm{Un}$ *имаме, че*

за всеки терм $\tau \in \mathcal{TE}$,

$\boxed{\tau}$ *е въпросът "Коя е породата* $\mathrm{Si}([\![\tau]\!]^{\mathcal{IN}}(v))$*?" с отговори:*

- $\boxed{\sigma}$, *който за всяка порода* $\sigma \in \mathcal{SP}$ *казва, че отговорът е "*$[\![\tau]\!]^{\mathcal{IN}}(v)$ *е дефинирано и* $\mathrm{Si}([\![\tau]\!]^{\mathcal{IN}}(v)) = \sigma$*", и*
- $\boxed{-}$, *който казва, че отговорът е "*$[\![\tau]\!]^{\mathcal{IN}}(v)$ *не е дефинирано"; и*

за всеки два терма $\tau_1 \in \mathcal{TE}$ *и* $\tau_2 \in \mathcal{TE}$,

$\boxed{\tau_1, \tau_2}$ *е въпросът "Дали* $[\![\tau_1]\!]^{\mathcal{IN}}(v) = [\![\tau_2]\!]^{\mathcal{IN}}(v)$*?" с отговори:*

- $\boxed{\approx}$, *който казва, че отговорът е "*$[\![\tau_1]\!]^{\mathcal{IN}}(v)$ *и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *са дефинирани и* $[\![\tau_1]\!]^{\mathcal{IN}}(v) = [\![\tau_2]\!]^{\mathcal{IN}}(v)$*";*
- $\boxed{\not\approx}$, *който казва, че отговорът е "*$[\![\tau_1]\!]^{\mathcal{IN}}(v)$ *и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *са дефинирани и* $[\![\tau_1]\!]^{\mathcal{IN}}(v) \neq [\![\tau_2]\!]^{\mathcal{IN}}(v)$*";*
- $\boxed{\pm}$, *който казва, че отговорът е "*$[\![\tau_1]\!]^{\mathcal{IN}}(v)$ *е дефинирано и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *не е дефинирано",*
- $\boxed{\mp}$, *който казва, че отговорът е "*$[\![\tau_1]\!]^{\mathcal{IN}}(v)$ *не е дефинирано и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *е дефинирано", и*
- $\boxed{=}$, *който казва, че отговорът е "*$[\![\tau_1]\!]^{\mathcal{IN}}(v)$ *и* $[\![\tau_2]\!]^{\mathcal{IN}}(v)$ *не са дефинирани".* □

6.1 Частично индексиране

Индекс е крайно дърво такова, че всеки нетерминален възел е маркиран с въпрос, всяка дъга е маркирана с отговор и всеки терминален възел е маркиран с граф. Наредената четворка $\langle \mathcal{NI}, \mathcal{IE}, \mathcal{OU}, \mathcal{LA} \rangle$ е *индекс* тогава и само тогава, когато

$\mathcal{NI} \in FinPow(\mathcal{RE}^*)$,
  $\varepsilon \in \mathcal{NI}$,
  за всяка последователност $\iota \in \mathcal{RE}^*$, за всеки отговор $\rho \in \mathcal{RE}$,
    ако $\iota\rho \in \mathcal{NI}$, то и $\iota \in \mathcal{NI}$,
$\mathcal{IE} = \{\iota \in \mathcal{NI} \mid$ за някой отговор $\rho \in \mathcal{RE}$, $\iota\rho \in \mathcal{NI}\}$,
$\mathcal{OU} = \{\iota \in \mathcal{NI} \mid$ за всеки отговор $\rho \in \mathcal{RE}$, $\iota\rho \notin \mathcal{NI}\}$,
$\mathcal{LA}$ е навсякъде определена функция от $\mathcal{NI}$ в $\mathcal{QU} \cup \mathcal{GRS}$[6],
  за всеки възел $\iota \in \mathcal{IE}$, то имаме, че $\mathcal{LA}(\iota) \in \mathcal{QU}$ и
  за всеки възел $\iota \in \mathcal{OU}$, то имаме, че $\mathcal{LA}(\iota) \in \mathcal{GRS}$.

Ще наричаме всеки елемент на $\mathcal{NI}$ *възел*, всеки елемент на $\mathcal{IE}$ *вътрешен възел*, всеки елемент на $\mathcal{OU}$ *външен възел* и $\mathcal{LA}$ *етикиране*. За всеки възел $\iota \in \mathcal{NI}$, ще казваме, че $\mathcal{LA}(\iota)$ *етикира* ι.

За да е възможно даден индекс да се използва за класификация на обекти над дадена класификационна схема $\mathcal{GRS}^\theta$, налагаме допълнителни условия над индекса. Маркираме всеки възел на индекса с множество от графи, което е подмножество на класификационната схема. Всеки въпрос, свързан с вътрешен възел в индекса, е такъв, че от него излизат поне две дъги, етикирани с отговори. Ако възелът е маркиран с множеството $\mathcal{GRS}_0^\theta$, то всяка дъга води до възел, който е маркиран с множеството $\mathcal{GRS}_i^\theta$ такова, че $\mathcal{GRS}_i^\theta \subset \mathcal{GRS}_0^\theta$ и елементарните описания, които са свързани с въпроса и отговора, са логически следствия от графите в множеството $\mathcal{GRS}_i^\theta$. С други думи, ако отговорът е верен за въпроса по отношение на даден обект в денотацията на граф в $\mathcal{GRS}_0^\theta$, то този граф е в множеството $\mathcal{GRS}_i^\theta$. Това свойство осигурява, че при движение по индекса при всеки правилен отговор, броят на потенциалните графи, описващи обекта, намалява. Допълнителни условия са: (1) коренът на индексът е маркиран с първоначалната класификационна схема $\mathcal{GRS}^\theta$; (2) от всеки вътрешен възел излизат толкова дъги, колкото правилни отговори има съответният въпрос по отношение на съответното множество от графи; и (3) външните възли са етикирани с графите, ко-

[6] $\mathcal{GRS}$ е множеството от всички атрибутни графи.

ито са единствени елементи на множеството, което ги маркира. Когато индекс отговаря на тези условия, ще казваме, е той *анализира* класификационната схема $\mathcal{GRS}^{\theta}$.

Ако се навигира по възлите на индекса $\mathcal{IX}$ от кореновия възел към даден външен възел чрез поставяне на въпроси за обекта υ и преминаване по дъгите, чиито етикети са отговори на тези въпроси, то тогава се стига до уникалния граф в $\mathcal{GRS}^{\theta}$, който е верен за обекта υ. Това е идеята зад алгоритъма `Graph`. Две неща са важни. Първо, алгоритъмът `Graph` не използва маркирането на индекса $\mathcal{IX}$ с множества от графи. Важното е, че такова маркиране на индекса $\mathcal{IX}$ съществува и са удовлетворени условията, зададени по-горе. Второ, алгоритъмът `Graph` може да класифицира обекта υ само, ако `Graph` има възможност да поставя въпросите на точен и ефективен оракъл за обекта υ. *Оракъл* е всеки алгоритъм, който изчислява навсякъде определена функция от множеството на въпросите $\mathcal{QU}$ в множеството на отговорите $\mathcal{RE}$. За всеки оракъл `Oracle`, за всяка интерпретация $\mathcal{IN} = \langle \textsc{Un}, \textsc{Si}, \textsc{Fi} \rangle$, за всеки обект $\upsilon \in \textsc{Un}$, `Oracle` *разкрива* обекта υ по отношение на $\mathcal{IN}$ тогава и само тогава, когато `Oracle` връща правилния отговор за всеки въпрос по отношение на обекта в интерпретацията. Примери на оракли, които разкриват езикови обекти може да включват хора, които знаят дадения език и използват своето знание или програми за анализ на корпуси за дадения език.

Сега можем да представим алгоритъма `Graph` формално: за всеки оракъл `Oracle`, за всеки индекс $\langle \mathcal{NI}, \mathcal{IE}, \mathcal{OU}, \mathcal{LA} \rangle \in \mathcal{INX}$,

```
Graph(⟨NI, IE, OU, LA⟩) :
  begin
    I := IE
    L := LA
    P := ε
    while P ∈ I;
       begin
          Q := L(P)
          R := Oracle(Q)
          P := suffix(P, R)
       end
    C := L(P)
    return C
  end
```

Следващата стъпка е да дадем алгоритъма `Index`, който по класификационна схема $\mathcal{GRS}^{\theta}$ построява индекс, който я анализира. В действител-

ност, алгоритъмът `Index` конструира както индекса, така и маркиране на индекса, като свързва класификационната схема $\mathcal{GRS}^{\theta}$ с корена, но алгоритъмът връща само индекса. В процеса на описание на алгоритъма `Index` има нужда да работим с възли на индекса, които са маркирани с класификационна схема, но все още нямат прикачен етикет (въпрос или граф). Този вид възли ще наричаме *висящи* възли. Алгоритъмът `Index` започва с единствен висящ възел ε, който е маркиран с класификационната схема $\mathcal{GRS}^{\theta}$. Започвайки от този единствен висящ възел, алгоритъмът `Index` построява индекс и маркиране с множества от графи едновременно, като превръща всеки висящ възел в маркиран вътрешен или външен възел. При това преобразуване на възли може да бъдат добавени нови висящи възли. При обработката на даден висящ възел ι и неговата маркировка $\mathcal{GRS}^{\theta\prime}$ алгоритъмът `Index` първо разглежда $\mathcal{GRS}^{\theta\prime}$, за да реши дали ι трябва да се преобразува във вътрешен или външен възел. Ако класификационната схема $\mathcal{GRS}^{\theta\prime}$ съдържа само един граф $\mathcal{GR}$, то тогава алгоритъмът `Index` преобразува ι във външен възел и добавя като етикет на ι графа $\mathcal{GR}$. В случая, когато схемата $\mathcal{GRS}^{\theta\prime}$ съдържа повече от един граф, то тогава алгоритъмът `Index` преобразува ι във вътрешен възел и прилага два алгоритъма към него: `Query` и `Grow`. Алгоритъмът `Query` определя въпрос κ, който се отнася само за термовете, които се срещат в литерали на графите в $\mathcal{GRS}^{\theta\prime}$, и алгоритъмът `Index` приписва този въпрос като етикет на възела ι. Алгоритъмът `Grow` се използва от `Index`, за да бъдат добавени дъги (със съответни етикети), които излизат от възела ι към нови висящи възли. Когато алгоритъмът `Index` спре, тоест `Index` е превърнал всички висящи възли във вътрешни или външни, то резултатът е индекс с маркиране, което маркира корена на индекса с класификационната схема $\mathcal{GRS}^{\theta}$. Следва алгоритъмът `Index`: за всяка класификационна схема $\mathcal{GRS}^{\theta} \in \mathcal{CS} \setminus \{\emptyset\}$,

```
Index(GRS^θ):
  begin
    E := GRS^θ
    I := ∅
    O := ∅
    L := ∅
    P := ⟨(ε, E)⟩
    while P е непразна последователност
       begin
          H := head(P)
          T := tail(P)
```

```
      N := left(H)
      E := right(H)
      if E е едноелементно множество
        then
          begin
            C := member(E)
            O := O ∪ {N}
            L := L ∪ {⟨N,C⟩}
            P := T
          end
        else
          begin
            Q := Query(E)
            G := Grow(E, Q)
            G := insert(N,G)
            I := I ∪ {N}
            L := L ∪ {⟨N,Q⟩}
            P := concatenate⟨T, G⟩
          end
    end
  N := I ∪ O
  return N
  return I
  return O
  return L
end
```

Твърдение 16 *Алгоритъмът* Index *изчислява навсякъде определена функция от* $\mathcal{CS} \setminus \{\emptyset\}$ *в* $\mathcal{INX}$ *такава, че за всяка класификационната схема* $\mathcal{GRS}^\theta \in \mathcal{CS} \setminus \{\emptyset\}$ *е изпълнено, че*

Index$(\mathcal{GRS}^\theta)$ *анализира* $\mathcal{GRS}^\theta$. □

6.2 Пълно индексиране

Следващият тип индекс, предложен в дисертацията, е индекс, съдържащ цялата информация от графите в класификационната схема. Този вид индексиране ще наричаме *пълно индексиране*. Идеята зад пълното индексиране е да построим всички частични индекси над класификационната схема, като ги обединим, където това е възможно. За целта дефинираме

еквивалентност на въпроси и еквивалентни класове въпроси в зависимост от начина на разделяне на класификационната схема на непресичащи се подмножества от класове. Тогава построяваме индекс, в който възлите са маркирани с класове еквивалентни въпроси или графи, а дъгите са маркирани с отговори на съответните въпроси. Навигацията над индекса е подобна на тази при частичните индекси, но сега оракълът може да отговори на който и да е от въпросите в индекса, тоест може да си избира по-лесни въпроси или такива, за които знае отговора.

Трябва да се отбележи, че пълните индекси в общия случай са много големи и може да се окажат неефективни за някои задачи. Затова трябва да се използват внимателно.

Също е възможно да се използват индекси, построени само на базата на част от въпросите в $\mathcal{QU}_{\mathcal{GRS}^g}$. По този начин полученият индекс ще е по-малък.

6.3 Дискусия

Използването на индекси за класификация на обекти спрямо класификационни схеми може да се разглежда като специфична система за извод в SRL. В тази своя същност бихме могли да я класифицираме като пълна, но не непротиворечива. Пълнотата идва от факта, че ако даден обект принадлежи на даден клас, то при използването на някой от индексите този обект ще бъде класифициран по подходящ начин. Но може да се случи обект, който не принадлежи на никой клас от класификационната схема, да бъде класифициран като принадлежащ на определен клас. Това може да се случи, ако информацията в индекса е вярна за обекта, но той се отличава от другите обекти в класа по информация, която не е в индекса. Следователно, след като даден обект е класифициран към даден клас, трябва да се провери дали удовлетворява цялата информация от този клас. Разбира се, тази проверка е много по-лесна, когато вече знаем за кой клас да проверяваме.

7 Глава седма: Представяне и обработка на лингвистично знание в ОФГ

В тази глава са представени приложения на формалния апарат за обработка на лингвистични знания в ОФГ, разработен в предишните глави. Ще разгледаме следните задачи, които са типични при обработката на

естествен език и създаването на езикови ресурси: (1) Анализ на изречения; (2) Дефиниране и изграждане на ОФГ корпус; (3) Извличане на граматики от ОФГ корпус; (4) Преструктуриране на ОФГ корпус.

7.1 Анализ на изречения

Това е една от основните задачи на компютърната лингвистика. Ако е дадена формална граматика на даден естествен език в определена лингвистична теория и определен лингвистичен формализъм, то да се разработи алгоритъм, който за всяко изречение решава дали изречението е граматично по отношение на граматиката. Ако изречението е граматично, то алгоритъмът връща съответната структура, която граматиката приема за изречението. Ако изречението е неграматично, то такава структура не съществува. В някои случаи е желателно на неграматичните изречения да се приписват частични структури, които отразяват анализа на отделни части на изречението. Нека $\Gamma = \langle \Sigma, \theta \rangle$ да бъде ОФГ граматика. Нека $\mathcal{ST} = \chi_1 \cdots \chi_n$ да бъде изречение, където χ_i $1 \leq i \leq n$ са словоформите, които съставят изречението. Теорията θ^e е преобразувана в изключващо множество от атрибутни графи $\mathcal{GRS}^{\theta^e}$. Представяме изречението като атрибутен граф $\mathcal{GR}_q^{\mathcal{ST}}$, който представлява граф заявка към граматиката. За да бъде изречението граматично спрямо граматиката, то графът заявка трябва да има непразна денотация в цялостен модел на граматиката. Използваме алгоритъма `ControlMod`, за да проверим този факт. Ако алгоритъмът върне стойност `true`, то изречението е граматично. Ако алгоритъмът върне стойност `false`, то изречението е неграматично. Ако алгоритъмът върне стойност `unknown`, то интерпретацията е неясна и може да зависи от състоянието на самата граматика.

7.2 Дефиниране и изграждане на ОФГ корпус

Нека припомним, че корпус е съвкупност от написани текстове или транскрибирана реч, които служат за база на лингвистични анализи и описания. От гледна точка на лингвистичните теории корпус е съвкупност от текстове или транскрибирана реч, анализирани с оглед на дадена лингвистична теория, тоест текст плюс интерпретация.

Разгледан като компютърен продукт, един корпус преминава през следните етапи на развитие: *проектиране*, *имплементиране*, *валидиране*, *документиране* и *експлоатация*. В нашия конкретен случай ОФГ корпус е набор от изречения (самостоятелни или образуващи свързан текст), ко-

ито са анализирани като пълни атрибутни графи. Тук се разглеждат етапите на развитие на един ОФГ корпус и как разработените средства могат да подпомогнат всеки един етап. Дефинираме формално ОФГ корпуса, като първо дефинираме корпус по отношение на граматически формализъм и след това прилагаме тази дефиниция към ОФГ.

Дефиниция 17 (Корпус в граматически формализъм) *Един корпус C в граматическия формализъм G е последователност от анализирани изречения, където всяко анализирано изречение е член на множеството от структури, дефинирани като* силен генеративен капацитет *(strong generative capacity (ще го бележим като* SGC*)) на дадена граматика Γ в този граматически формализъм:*

$\forall S.S \in C \rightarrow S \in \mathtt{SGC}(\Gamma)$,

където Γ е граматика във формализма G и

ако $\sigma(S)$ е фонологичният низ на S и $\Gamma(\sigma(S))$ е множеството от всички анализи, приписани от граматиката Γ на фонологичния низ $\sigma(S)$, тогава

$\forall S'.S' \in \Gamma(\sigma(S)) \rightarrow S' \in C$. □

За да приложим горната дефиниция, трябва да дефинираме силен генеративен капацитет за граматики, изразени в SRL. Такава дефиниция е разработена в много близко сътрудничество от Paul John King ([King 1999]) и Carl Pollard ([Pollard 1999]). Така дефинираме ОФГ корпус, както следва:

Дефиниция 18 (Корпус в ОФГ) *C е* корпус в ОФГ, *ако G е последователност от пълни атрибутни графи, такива че за дадена граматика SRL граматика Γ е изпълнено, че:*

$\forall S.S \in C \rightarrow S \in \mathtt{SGC}(\Gamma)$ *и*

ако $\sigma(S)$ е фонологичният низ на S и $\Gamma(\sigma(S))$ е множеството от всички анализи, приписани от граматиката Γ на фонологичния низ $\sigma(S)$, тогава

$\forall S'.S' \in \Gamma(\sigma(S)) \rightarrow S' \in C$. □

Можем да заключим, че атрибутните графи са приложими и за двете неща: (1) *Представяне на ОФГ корпус.* Всяко изречение в корпуса е представено като пълен атрибутен граф; (2)*Представяне на ОФГ граматика като множество от атрибутни графи.* Можем да покажем

формална връзка между една граматика и един корпус, като използваме характеристиките на атрибутните графи:

Дефиниция 19 (Корпусна граматика) *Нека C да бъде ОФГ корпус и Γ да бъде една ОФГ граматика с графично представяне $\mathcal{GR}$. Казваме, че Γ е* граматика на корпуса *C тогава и само тогава, когато за всеки граф $\mathcal{GR}_C$ в C и всеки възел $\nu \in \mathcal{GR}_C$ съществува граф $\mathcal{GR}_\Gamma$ в $\mathcal{GR}$ такъв, че $\mathcal{GR}_C|_\nu \sqsubseteq \mathcal{GR}_\Gamma$.* □

Ако Γ е корпусна граматика на C, то Γ приема всички анализи в C.

Проектиране на ОФГ корпус. В процеса на проектиране на един корпус се решават следните проблеми: (1) Определяне на езиковите явления, които ще бъдат интерпретирани в корпуса; (2) Формализиране на тези явления и определяне на формата за тяхното кодиране; (3) Написване на ръководство на анотатора, което определя механизма на създаване на корпуса и процеса на взимане на решения по отношение на проблематични случаи на описания. Разработеният от нас формален инвентар може да се приложи към втората задача. Формализацията на явленията изисква написването на подходяща сигнатура и съответна теория към нея. След като е написана, дадената теория може да се преобразува в изключваща матрица. Този процес ще позволи да се провери непротиворечивостта на теорията и дали получените клаузи съответстват на лингвистичната интуиция на експертите, които формализират явлението. Форматът на кодиране на явленията в корпуса е, както казахме по-горе, пълни атрибутни графи.

Анотиране на ОФГ корпус. Анотирането на ОФГ корпуса съответства на етапа имплементиране на корпус. Анотирането на ОФГ корпуса, представено тук, се базира на идеята за избирането на (синтактичен) анализ от множество автоматично построени изреченски синтактични анализи. Тези синтактични анализи са направени чрез механизъм за извод, който използва графовото представяне на анотационната схема и графовото кодиране на изречението. Анотационният процес се организира, както следва:

1. Избраното изречение се обработва частично. Тази обработка е съвместима с йерархията от сортове на ОФГ и съдържа: морфологичен анализ, снемане на многозначност и нерекурсивен частичен анализ.

2. Резултатът от предишната стъпка се кодира като атрибутен граф (графи) и се обработва по-нататък чрез алгоритъма `ControlMod` с помощта на описаната анотационна схема. Резултатът е множество от пълни атрибутни графи.

3. Избирането на правилния анализ се разглежда като *класификация* на частичните описания на верния изреченски анализ.

Валидиране на ОФГ корпус. В процеса на валидация се проверяват две условия: (1) Непротиворечивост на корпуса и анотационната схема; (2) Съгласуване на решенията на анотаторите при повече от една вярна възможност. Първото условие се проверява автоматично по време на анотирането на корпуса. Второто условие се удовлетворява по-трудно и изисква намесата на експерт. Тук разработеният апарат се използва за налагане на допълнителни условия над корпуса и извличане на анализи, които не ги удовлетворяват.

7.3 Извличане и специалиране на граматики от ОФГ корпус

В този раздел е разработен механизъм за извличане на граматики и прецизиране на ОФГ граматики от ОФГ корпус.

Парсинг, ориентиран към данните (DOP) Опираме се на модела на Rens Bod – Парсинг, ориентиран към данните (Data-Oriented Parsing Model). Основните принципи на модела може да се намерят в [Bod 1998]. Авторът представя модел за научаване на граматика от корпуси, който включва следните елементи: (1) Дефиниране на граматически формализъм за граматиката-цел (target grammar); (2) Въвеждане на процедура за построяване на изреченски анализ в избрания граматически формализъм; (3) Въвеждане на процедура за декомпозиране (фрагментиране), която извлича граматика в граматическия формализъм-цел от структурите в корпуса; (4) Модел за изпълнение, който ръководи анализа на новите изречения спрямо някои желателни условия. Към тях добавяме още две предположения, които не са били споменавани в работите на Rens Bod: (5) Структурите в корпуса са декомпозируеми в граматическия формализъм; (6) Извлечената граматика не трябва нито да генерира повече от необходимите анализи (overgenerate), нито да генерира по-малко от необходимите анализи спрямо обучаващия корпус.

Дефинираме множеството от фрагменти, извлечени от ОФГ корпуса C. Конструираме множество $\mathcal{GRF}$ от атрибутни графи такова, че:

1. За всеки граф $\mathcal{GR} \in \mathcal{GRF}$, $\mathcal{VE}\langle\rho,\phi\rangle$ е дефинирано тогава и само тогава, когато $\mathcal{AP}\langle \mathrm{SI}\langle\rho\rangle,\phi\rangle$ е дефинирано, и

2. За всеки граф $\mathcal{GR} \in \mathcal{GRF}$, съществува граф $\mathcal{GR}_C$ в C и съществува възел $\nu \in \mathcal{GR}_C$ такъв, че $\mathcal{GR}_C \mid \nu \sqsubseteq \mathcal{GR}$.

Първото условие осигурява всички атрибути, които са подходящи за дадена порода, да бъдат представени на кореновия възел за всеки атрибутен граф в $\mathcal{GRF}$, чийто корен е маркиран с тази порода. Второто условие осигурява всеки атрибутен граф в $\mathcal{GRF}$ наистина да е фрагмент на поне един атрибутен граф в корпуса. Множеството $\mathcal{GRF}$ е наредено по отношение на релацията включване на графи. Нека Γ да бъде множество от атрибутни графи такова, че за всеки минимален атрибутен граф M в $\mathcal{GRF}$ съществува поне един атрибутен граф в Γ, който включва M и Γ съдържа само атрибутни графи от $\mathcal{GRF}$. Всяка граматика Γ, конструирана по този начин, е граматика на корпуса C.

Нека Γ_i и Γ_j да бъдат две граматики. Казваме, че граматката Γ_i е по-обща от граматиката Γ_j тогава и само тогава, когато $\mathtt{SGC}(\Gamma_j) \subseteq \mathtt{SGC}(\Gamma_i)$. В този случай наричаме Γ_i *генерализация* на Γ_j и наричаме Γ_j *специализация* на Γ_i.

Прецизиране на граматика на базата на ОФГ корпус. Развитието на един ОФГ корпус се базира на използването на граматика. Бихме могли да приемем, че заедно с корпуса имаме на разположение и първоначалната ОФГ граматика. Преди да опишем процеса на прецизиране, е необходимо да въведем две понятия. Нека наричаме граматиката Γ *изключваща граматика*, ако тя е изключващо множество от атрибутни графи. В противен случай граматиката е *неизключваща граматика*.

Първо, построяваме графово представяне на първоначалната ОФГ граматика. Нека Γ_0 да бъде множеството от графи в това представяне. Γ_0 е граматика на корпуса, защото графите в Γ_0 се използват по време на анотацията на корпуса и следователно, те включват всички пълни графи, представени в корпуса. Като цяло, Γ_0 ще приписва повече от необходимите анализи над корпуса.

Граматиката Γ_0 е изключваща граматика, защото тя се построява чрез алгоритъм за правене на изключващи матрици. За *прецизиране* смятаме

всяка граматика Γ' такава, че Γ' е специализация на Γ_0 и Γ' е изключваща граматика.

Няма гаранция за съществуването на точно едно най-общо прецизиране на дадената граматика. Така нашата задача е да намерим най-общото(ите) прецизиране(ия) на Γ_0, което не приписва повече от необходимите анализи над корпуса. Нека $R = \{\Gamma_1, \ldots, \Gamma_n\}$ да бъде множество от всички прецизирания на първоначалната граматика Γ_0 . След построяването на R има различни начини да се продължи нататък. Първо, можем да изберем една от граматиките на базата на други критерии.

Друга възможност е да използваме всички графи в тези граматики като една граматика - $\Gamma_r = \bigcup_{\Gamma_i \in R} \Gamma_i$. Тази граматика може да бъде неизключваща и в общия случай тя ще приписва повече от необходимите анализи над корпуса. Ако искаме да съхраним това свойство, трябва да индексираме графите в Γ_r с граматиките, към които принадлежат. Тогава при извода ще използваме само графите, които принадлежат на същата граматика.

Прецизиране на граматика чрез негативна информация. В процеса на анотация на изречения анализите на всяко изречение по отношение на анотационната схема Γ_0 се разделят на две: множество на верните анализи, които се записват в корпуса, и множество на неверните анализи, които се отхвърлят. Прецизирането на анотационната схема (граматика) може да се извърши на базата на тези отхвърлени анализи. Тъй като части от отхвърлените анализи може да са части от приетите анализи, то ние разглеждаме множеството от всички пълни графи, които са подграфи на отхвърлените анализи и от тях избираме онези графи, които не се срещат при верните анализи. След това избираме специализация на анотационната схема, която отхвърля само тези пълни графи, които се използват единствено при отхвърлените графи. Отново съществуват повече от едно прецизиране.

Ако алгоритъмът пропадне и не произведе ново множество от графи, тогава съществува непоследователност в приемането и/или в отричането на графите. Това би могло да се случи, ако анотаторът маркира като грешен анализ, който е бил посочен като правилен преди.

7.4 Преструктуриране на ОФГ корпус

Нуждата от преструктуриране на ОФГ корпус, или *рекласификация* на вече класифицирани лингвистични обекти, възниква във връзка със следните проблеми и задачи: (1) Промени в лингвистичното описание-цел на елементите в корпуса; (2) Нови задачи, за които е разработен корпусът; (3) Нови развития в лингвистичната теория; (4)Подвеждащи решения, взети по време на етапа на проектиране на корпуса. Във всеки от тези случаи е необходимо развиването на нова анотационна схема. Проблемите, които съпътстват подобна стъпка, са добре известни: Какво се случва с корпуса, който е бил изграждан досега? Как да го използваме при новите обстоятелства и с минимален разход?

Нека Σ_{old} и Σ_{new} да бъдат две сигнатури и нека Γ_{old} да бъде анотационна схема, конструирана на базата на Σ_{old}, а Γ_{new} да бъде анотационната схема, конструирана на базата на Σ_{new}. Идеята за рекласификация се базира на понятието за правила на съответствия между описанията по отношение на старата и новата анотационни схеми. Общият формат на тези правила е: $\delta_{old} \Rightarrow \delta_{new}$, където δ_{old} е описание по отношение на Σ_{old}, а δ_{new} е описание по отношение на Σ_{new}. Смисълът на такива правила е: за всеки модел $\mathcal{IN}_{old}$ на Γ_{old} такъв, че описанието δ_{old} е удовлетворимо в него, съществува модел $\mathcal{IN}_{new}$ на Γ_{new} такъв, че описанието δ_{new} е удовлетворимо в него. Смятаме тези правила като правила за пренасяне на знание между две анотационни схеми. Алгоритъмът за рекласификация работи по следния начин:

1. Нека Σ_{old} и Σ_{new} да бъдат две сигнатури и нека Γ_{old} да бъде анотационна схема, построена на базата на Σ_{old}, а Γ_{new} да бъде анотационна схема, построена на базата на Σ_{new}. Нека CR е множество от правила за съответствия.

2. Нека $\mathcal{GR}_{old}$ да бъде граф по отношение на Γ_{old} за изречението S. Нека $\{\mathcal{GR}^1_{new}, \ldots, \mathcal{GR}^n_{new}\}$ са кандидат-анализите за изречението S по отношение на новата анотационна схема Γ_{new}.

3. Алгоритъмът конструира множеството ED_{old} на всички описания δ_{old} такива, че съществува правило на съответствие $\delta_{old} \Rightarrow \delta_{new} \in CR$, а $\mathcal{GR}_{old}$ е в денотацията на δ_{old} за всяка интерпретация на Γ_{old}, която удовлетворява $\mathcal{GR}_{old}$. Следователно, ED_{old} съдържа всички описания на лявата страна на правилата за съответствие, които са верни за графа.

4. След това алгоритъмът конструира множеството ED_{new} на описания δ_{new} такива, че съществува правило на съответствие $\delta_{old} \Rightarrow \delta_{new} \in CR$ и $\delta_{old} \in ED_{old}$. Ние смятаме множеството ED_{new} за пренесено знание от старата анотация на изречението S към новата анотация.

5. Накрая алгоритъмът използва множеството ED_{new} и индекса за новите потенциални анализи за изречението S, за да намери минималния брой графи от множеството $\{\mathcal{GR}^1_{new}, \ldots, \mathcal{GR}^n_{new}\}$, които удовлетворяват всички описания в ED_{new}.

Резултатът от този алгоритъм е множество от графи. Ако множеството е празно, това означава, че пренесеното знание е несъвместимо с новата анотационна схема и не може да бъде реално използвано. В този случай създателите на корпуса трябва да преразгледат правилата за съответствия. Ако множеството е с един елемент, тогава то е равно на изреченския анализ по отношение на новата анотационна схема. Ако множеството съдържа повече от един елементи, тогава старият анализ не съдържа достатъчно информация за уникална класификация на изречението спрямо новата анотационна схема и е необходима намесата на човек.

Рекласификацията играе много важна роля при създаването на така наречените динамични езикови ресурси. Тези ресурси търпят промяна в зависимост от промените в съответната лингвистична теория. В останалата част от раздела е разгледано едно приложение на рекласификацията при оценка на системи за синтактичен анализ над корпус в ОФГ.

8 Глава осма: Заключение

В дисертационния труд са разработени логически методи за представяне и обработка на лингвистични знания в рамките на Опорната фразова граматика. За логически формализъм беше избрана логиката SRL, която е специално разработена като граматически формализъм за ОФГ. Беше показано, че основният дедуктивен проблем за съществуване на цялостен модел на теория е неразрешим. За да бъдат дефинирани системи за извод, които са непротиворечиви, но непълни, или пълни, но противоречиви, първо предложихме нова нормална форма за крайни теории в SRL — множество от атрибутни графи. Тази нормална форма е добра както за представянето на теориите в SRL, така също и като абстракция над обектите в дадена интерпретация на SRL теория. На базата на

това представяне са разработени две системи за извод. Първата система е непротиворечива, но непълна. Тази система се базира на контролирано изграждане на атрибутен граф, който да е основа за построяване на интерпретация с необходимите свойства. Втората система за извод е предназначена за класификация на обекти по отношение на класификационна схема, построена на базата на крайна теория в SRL. Класификацията се извършва с помощта на индекс над класовете в класификационната схема. Индексът съдържа елементарни твърдения за обектите, които се класифицират, под формата на въпроси и отговори. Получавайки информация за верността на съответните елементарни твърдения, алгоритъмът за класификация навигира по индекса и намира (след краен брой стъпки) класа, който описва (е верен за) даденият обект. Класификацията е пълна система за извод, тоест, ако даденият обект се описва от даден клас на класификационната система, то той ще бъде класифициран правилно. Но класификацията е противоречива, тоест обект, който не е описан от никой клас в класификационната схема, се класифицира като такъв. Това се дължи на факта, че се проверява само част от информацията за обекта.

На базата на тези два начина за извод в SRL и нормалната форма, дефинирана преди това, показваме как може да се поддържат дейности, свързани с автоматичния анализ на изречения по отношение на ОФГ граматика, построяването на ОФГ корпус, извличане на ОФГ граматика от такъв корпус, специализация на ОФГ граматика на базата на корпус и преобразуване на ОФГ корпус по отношение на нова ОФГ граматика. По този начин показваме, че разработените алгоритми подпомагат основните етапи на жизнения цикъл на един корпус, което беше основната цел на разработката.

Резултати, които са директен продукт на работата

R1. Доказана е неразрешимостта на проблема за цялостен модел на теория в SRL.

R2. Дефинирана е нормална форма за крайни теории в SRL. Тя има две представяния: като изключваща матрица от клаузи и като изключващо множество от атрибутни графи.

R3. Разработен е алгоритъм за непротиворечив, но непълен извод по отношение на цялостните модели на SRL теории.

R4. Разработен е механизъм за класификация на крайни теории в SRL.

R5. Дефиниран е корпус по отношение на лингвистична теория и тази дефиниция е приложена към корпус в рамките на ОФГ. Предложена е методология за изграждане на такъв корпус с помощта на класификация на анализи на изречения.

R6. Дефинирана е процедура за извличане на граматики от ОФГ корпус.

R7. Разработени са алгоритми за специализация на ОФГ граматики на базата на ОФГ корпус. Те работят на базата на положителна информация от корпуса или на базата на отрицателна информация от анализите, отхвърлени по време на създаването на корпуса.

R8. Разработен е алгоритъм за повторна класификация (рекласификация) на базата на правила за трансфер на знание между две ОФГ граматики.

Публикации, свързани с приносите на дисертацията

1. Kiril Simov and Paul J. King. 1996. *Indexing of linguistic knowledge.* In: *Proceedings Logical Aspects of Computational Linguistics* (conference abstracts). Nancy, France. pages 81–84.

2. Paul John King and Kiril Simov. 1997. *The Automatic Deduction of Classificatory Systems from Finite Linguistic Theories. (Abridged)* In: *Proceedings Logical Aspects of Computational Linguistics.* Lecture Notes in Artificial Intelligence 1328, Springer-Verlag, Berlin, Germany. pages 248-273.

3. Paul John King and Kiril Simov. 1998. *The Automatic Deduction of Classificatory Systems from Linguistic Theories.* In: *Grammars,* volume 1, number 2. Kluwer Academic Publishers, The Netherlands. pages 103-153.

4. Paul J. King, Kiril Simov and Bjørn Aldag. 1999. *The complexity of modelability in finite and computable signatures of a constraint logic for head-driven phrase structure grammar.* In: *The Journal of Logic, Language and Information,* volume 8, number 1. Kluwer Academic Publishers, The Netherlands. pages 83-110.

5. Kiril Simov. 2001. *Grammar Extraction from an HPSG Corpus.* In: *Proceedings of the RANLP 2001 Conference.* Tzigov chark, Bulgaria. pages 285–287.

6. Kiril Simov, Gergana Popova and Petya Osenova. 2002. *HPSG-based syntactic treebank of Bulgarian (BulTreeBank).* In: *"A Rainbow of Corpora: Corpus Linguistics and the Languages of the World",* edited by Andrew Wilson, Paul Rayson, and Tony McEnery; Lincom-Europa, Munich, pp. 135-142.

7. Kiril Simov, Milen Kouylekov, Alexander Simov. 2002. *Incremental Specialization of an HPSG-Based Annotation Scheme.* In: *Proceedings of the Workshop on "Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data".* The LREC Conference. Canary Islands, Spain. pages 16–23.

8. Kiril Simov. 2002. *Grammar Extraction and Refinement from an HPSG Corpus.* In: *Proceedings of the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics.* Trento, Italy. pages 38–55.

9. Kiril Simov. 2004. *HPSG-based annotation scheme for corpora development and parsing evaluation*. In: Nicolas Nicolov and Kalina Botcheva and Galia Angelova and Ruslan Mitkov (eds.) *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*. John Benjamins, Amsterdam/Philadelphia. Current Issues in Linguistic Theory (CILT). volume 260. pages 327–336.

10. Kiril Simov, Petya Osenova, Alexander Simov and Milen Kouylekov. 2004. *Design and Implementation of the Bulgarian HPSG-based Treebank*. In: Erhard Hinrichs and Kiril Simov eds. *Special Issue on Treebanks and Linguistic Theories*. Research on Language & Computation. Springer Science+Business Media B.V., Formerly Kluwer Academic Publishers B.V. Volume 2, Number 4. pages 495–522.

Някои от резултатите от дисертацията са приложени при проектирането и имплементацията на системата CLaRK ([Simov et. al. 2001]), като са видоизменени по подходящ начин, за да бъдат приложени върху XML документи. Също така те са приложени при разработването на анотационната схема на българската ОФГ синтактична база и нейното анотиране в рамките на международния проект BulTreeBank (виж [Simov et. al. 2002c], [Simov and Osenova 2003] и [Simov et. al. 2004b]).

Използвана литература

[Abney 1991] Steven Abney. 1991. *Parsing By Chunks.* In: Berwick R., Abney St., Tenny C. (eds), *Principle-Based Parsing.* Kluwer Academic Publishers, Dordrecht.

[Abney, 1996] Steve Abney. 1996. *Partial Parsing via Finite-State Cascades.* In: *Proceedings of the ESSLLI'96 Robust Parsing Workshop.* Prague, Czech Republic.

[Blackburn and Spaan 1993] Patric Blackburn and Edith Spaan. 1993. *A modal perspective on computational complxity of attribute value grammar.* In: *The Journal of Logic, Language and Information,* volume 2, pages 129-169. Kluwer Academic Publishers, The Netherlands.

[Blaheta 2002] Don Blaheta. 2002. *Handling noisy training and testing data.* In: *Proceedings of the 7th conference on Empirical Methods in Natural Language Processing.*

[Bod 1998] Rens Bod. 1998. *Beyond Grammar: An Experience-Based Theory of Language.* CSLI Publications, CSLI, California, USA.

[Bod 2000a] Rens Bod. *Parsing with the Shortest Derivation.* 2000. In: *Proceedings COLING'2000.* http://arxiv.org/abs/cs.CL/0011040.

[Bod 2000b] Rens Bod. 2000. *Do All Fragments Count?.* Technical Report COMP-11-12, http://arxiv.org/abs/cs.CL/0011040.

[Bod 2003] Rens Bod, Remko Scha and Khalil Sima'an (eds.). 2003. *Data-Oriented Parsing.* CSLI Publications, University of Chicago Press.

[Carpener 1992] Bob Carpenter. 1992. *The Logic of Typed Feature Structures.* Cambridge Tracts in Theoretical Computer Science 32. Cambridge University Press.

[Carpenter and Penn 2005] Bob Carpenter and Gerald Penn. 2005. *ALE: Attribute-Logic Engine.* System Home Page: `http://www.cs.toronto.edu/ gpenn/ale.html`

[Carroll et al. 2003] J. Carroll, G. Minnen, T. Briscoe. 2003. *Parser Evaluation Using a Grammatical Relation Annotation Scheme.* In: A. Abeillé (ed.), *Treebanks. Building and Using Parsed Corpora.* Dordrecht: Kluwer

[Carpener 1992] Bob Carpenter. 1992. *The Logic of Typed Feature Structures.* Cambridge Tracts in Theoretical Computer Science 32. Cambridge University Press.

[Copestake 2002] Ann Copestake. 2002. *Implementing Typed Feature Structure Grammars.* CSLI Publications. USA.

[Cotton and Bird 2002] Scott Cotton and Steven Bird. 2002. *An Integrated Framework for Treebanks and Multilayer Annotations.* In: *Proceedings from the LREC conference.* Canary Islands, Spain. pp 1670-1677.

[Cucerzan and Yarowsky 1999] Silviu Cucerzan and David Yarowsky. 1999. *Language Independent Named Entity Recognition. Combining Morphological and Contextual Evidence.* In: *Proceedings, Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora.* College Park, USA. pp. 90-99.

[Cunningham et. al. 1997] Hamish Cunningham, Kevin Humphteys, Robert Gaizaukas and Yorick Wilks. 1997. *Software Infrastructure for Natual Language Processing.* In: *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97).* Washington, DC, USA. pp. 237–244.

[Dickinson and Meurers 2003] Markus Dickinson and W. Detmar Meurers, 2003. *Detecting inconsistencies in treebanks.* In: *Proceedings of the Second Workshop on Treebanks and Linguistic Theories (TLT 2003).* Växjö, Sweden.

[Dipper 2000] Stefanie Dipper. 2000. *Grammar-based Corpus Annotation.* In: *Proceedings of the Workshop on Linguistically Interpreted Corpora.* Luxembourg.

[Harel 1983] David Harel. 1983. *Recurring dominoes: making the highly undecidable highly understandable.* In: *Proceedings of the Conference on Foundations of Computing Theory.* Springer Lecture Notes in Computer Science 158. pp. 177–194.

[Harrison et al. 1991] P. Harrison, St. Abney, E. Black, D. Flickinger, C. Gdaniec, R. Grishman, D. Hindle, B. Ingria, M. Marcus, B. Santorini, and T. Stralkowski. 1991. *Evaluating syntax performance of parser/grammars of English.* In: *Proceedings of the Workshop on Evaluating Natual Language Processing Systems.* pages 71-77. Berkeley, Ca, USA.

[Herzog and Rollinger 1991] Otthein Herzog, Claus-Rainer Rollinger (Eds.). 1991. *Text Understanding in LILOG, Integrating Computational Linguistics and Artificial Intelligence, Final Report on the IBM Germany LILOG-Project.* Lecture Notes in Computer Science 546. Springer. Germany.

[Götz 1993] Thilo Götz. 1993. *A normal form algorithm for King's descriptive formalism.* Term paper. Eberhard-Karls-Universität, Tübingen, Germany.

[Götz and Meurers 1997] Thilo Götz and W. Detmar Meurers. 1997. *The ConTroll system as large grammar development platform.* In *Proceedings of the ACL/EACL post-conference workshop on Computational Environments for Grammar Development and Linguistic Engineering.* Madrid, Spain.

[Ide Véronis 1998] Nancy Ide and Jean Véronis. 1998. (Eds.) *Word Sense Disambiguation. Special issue of Computational Linguistics.* In: *Computational Linguistics Vol. 24 No. 1.* ACL. USA. pp 1–40.

[Ivanova and Doikoff 2002] Krassimira Ivanova and Dimitar Doikoff. 2002. *Cascaded Regular Grammars and Constraints over Morphologically Annotated Data for Ambiguity Resolution.* In: *Proceedings of The TLT Workshop.* Sozopol, Bulgaria. pp. 96–113.

[Johnson 1988] Mark Johnson. 1988. *Attribute-Value Logic and the Theory of Grammar.* Stanford, CA: CSLI.

[Kasper and Rounds 1986] Robert Kasper and William Rounds. 1986. *A Logical Semantics for Feature Structures.* In: *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics.* Association for Computational Linguistics. pp. 235–242.

[Kay 1979] Martin Kay. 1979. *Functional Grammar.* In: *Proceedings of the Fifth Annual Meeting of the Berkeley Linguistic Society.* Berkeley Linguistic Society, Berkeley, California, USA. pp. 142–158.

[Kepser 1994] Stephan Kepser. 1994. *A Satisfiability Algorithm for a Logic for Typed Feature Structures.* Unpublished manuscript. Universität Tübingen, Tübingen, Germany.

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

[King 1994] Paul John King. *An expanded logical formalism for head-driven phrase structure grammar.* Unpublished manuscript. Universität Tübingen, Tübingen, Germany. 1994.

[King 1999] Paul J. King. *Towards Thruth in Head-Driven Phrase Structure Grammar.* In V. Kordoni (Ed.), *Tübingen Studies in HPSG,* Number 132 in Arbeitspapiere des SFB 340, pp 301-352. Germany. 1999.

[King and Simov 1997] Paul John King and Kiril Iv. Simov. 1997. *The Automatic Deduction of Classificatory Systems from Finite Linguistic Theories. (Abridged)* In: *Proceedings Logical Aspects of Computational Linguistics.* Lecture Notes in Artificial Intelligence 1328, Springer-Verlag, Berlin, Germany. pages 248-273.

[King and Simov 1998] Paul John King and Kiril Iv. Simov. 1998. *The Automatic Deduction of Classificatory Systems from Linguistic Theories.* In: *Grammars,* volume 1, number 2, pages 103-153. Kluwer Academic Publishers, The Netherlands.

[King, Simov and Aldag 1999] Paul J. King, Kiril Iv. Simov and Bjørn Aldag. 1999. *The complexity of modelability in finite and computable signatures of a constraint logic for head-driven phrase structure grammar.* In: *The Journal of Logic, Language and Information,* volume 8, number 1, pages 83-110. Kluwer Academic Publishers, The Netherlands.

[Kinyon and Rambow 2003] Alexandra Kinyon, Owen Rambow. 2003. *The Meta-Grammar: a cross-framework and cross-language test-suite generation tool.* In: *Proceedings of The 4th International Workshop on Linguistically Interpreted Corpora.* Budapest, Hungary.

[Krushkov 2001] Hristo Krushkov. 2001. *Automatic Morphological Processing of Bulgarian Proper Nouns.* In: *Journal TAL (Traitement Automatique des Langues), Vol. 41, No. 3.* France. pp 709–726.

[Крушков и Тачев 2002] Христо Крушков и Георги Тачев. 2002. *Стохастичен маркировчик на частите на речта.* In: *Proceedings of the 27th International Conference ICT&P'2002.* Приморско, България, стр. 147–151.

[Крушков, Ганчев и Крушкова 2000] Христо Крушков, Д. Ганчев и М. Крушкова. 2000. *Автоматичен анализ и синтез на сложни глаголни форми.* В *Юбилейна научна сесия – 30 години Факултет по математика и информатика на ПУ "Паисий Хилендарски".* Пловдив, България. стр. 241-246.

[Kübler and Hinrichs 2001] S. Kübler and E. Hinrichs. 2001. *From Chunks to Function-Argument Structure: A Similarity-Based Approach.* In: *Proceedings of ACL-EACL 2001.* Toulouse, France.

[Kveton and Oliva 2002] Pavel Kveton and Karel Oliva, 2002. *(Semi-)automatic detection of errors in pos-tagged corpora.* In: *Proceedings of the 17th International Conference on Computational Linguistics (COLING 2002).* Taiwan.

[Lager 1996] Torbjörn Lager. 1996. *A Logical Approach to Computational Corpus Linguistics.* Doctoral thesis. Department of Linguistics, University of Goteborg, Goteborg, Sweden. `http://www.ling.gu.se/ lager/taglog.html`

[Lin 2003] Dekang Lin. 2003. *Dependency-based Evaluation of Minipar.* In *Proc. of the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics.* In: A. Abeillé (ed.), *Treebanks. Building and Using Parsed Corpora.* Dordrecht: Kluwer

[Marciniak et al. 2003] Margorzata Marciniak, Agnieszka Mykowiecka, Adam Przepiórkowski, Anna Kupść. 2003. *An HPSG-Annotated Test Suite for Polish.* In: Anne Abeillé (editor). *Treebanks. Building and Using Parsed Corpora.* Kluwer Academic Publishers. pp 129-146.

[McDonald 1996] David D. McDonald. 1996. *Internal and External Evidence in the Identification and Semantic Categorization of Proper Names.* In: *Corpus Processing for Lexical Acquisition.* (ed. by Bran Boguraev and James Pustejovsky). chapter 2, The MIT Press, Cambridge, MA, USA. p. 21-39.

[Meurers et al. 2002] Detmar Meurers, Gerald Penn, and Frank Richter. 2002. *A Web-based Instructional Platform for Constraint-Based Grammar Formalisms and*

Parsing. In *Proc. of the Effective Tools and Methodologies for Teaching NLP and CL.* ACL. Philadelphia, PA, USA. pp 18–25.

[Mikheev et. al. 1999] Andrei Mikheev, Marc Moens and Claire Grover. 1999. *Named Entity Recognition without Gazetteers.* In: *Proceedings of EACL'99.* Bergen, Norway. pp. 1–8.

[Mikheev 1999] Andrei Mikheev. 1999. *Periods, Capitalized Words, etc.* In: *Computational Linguistics, Volume 28, Number 3.* ACL. USA. pp 289–318.

[Mikheev 2000a] Andrei Mikheev. 2000. *Tagging Sentence Boundaries.* In: *Proceedings of NACL'2000.* ACL. Seattle. USA. pp. 264–271.

[Mikheev 2000b] Andrei Mikheev. 2000. *Document Centered Approach to Text Normalization.* In: *Proceedings ofSIGIR'2000 (Athens).* ACM. pp. 136-143.

[Oepen et. al. 2002] Stephan Oepen, Ezra Callahan, Dan Flickinger and Christopher D. Manning. 2002. *LinGO Redwoods. A Rich and Dynamic Treebank for HPSG,* In: *Proc. of The Workshop Beyond PARSEVAL. The Third LREC Conference.* Las Palmas, Spain.

[Osenova 2002] Petya Osenova. 2002. *Bulgarian Nominal Chunks and Mapping Strategies for Deeper Syntactic Analyses.* In: *Proceedings of The First TLT Workshop.* Sozopol, Bulgaria. pp. 150–166.

[Osenova and Simov 2002] Petya Osenova and Kiril Simov. 2002. *Learning a Token Classification from a Large Corpus. (A case study in abbreviations).* In: *Proceedings of the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics.* Trento, Italy.

[Osenova and Kolkovska 2002] Petya Osenova and Sia Kolkovska. 2002. *Combining the Named-Entity Tecognition Task and NP Chunking Strategy for Robust Preprocessing.* In: *Proceedings of The First TLT Workshop.* Sozopol, Bulgaria. pp. 167–182.

[Pollard 1999] Carl Pollard. 1999. *Strong Generative Capacity in HPSG.* In: Webelhuth, G., Koenig, J.-P., and Kathol, A., editors, *Lexical and Constructional Aspect of Linguistic Explanation,* pp 281-297. CSLI, Stanford, California, USA.

[Pollard and Sag 1987] Carl J. Pollard and Ivan A. Sag. 1987. *Information-Based Syntax and Semantics,* vol. 1. CSLI Lecture Notes 13. Center for the Study of Language and Information, Stanford, California, USA.

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

[Попов, Симов и Видинска 1998] Димитър Попов, Кирил Симов, Светломира Видинска. 1998. *Речник за правоговор, правопис и пунктуация.* Атлантис КЛ, София, България.

[Richter and Sailer 1995] Frank Richter and Manfred Sailer. 1995 *Remarks on linearization: reflections on the treatment of LP-rules in HPSG in a typed feature logic.* Master's thesis. Seminar für Sprachwissenschaft, Eberhard-Karls-Universität, Tübingen, Germany.

[Rounds and Kasper 1986] William Rounds and Robert Kasper. 1986. *A Complete Logical Calculus for Record Strucutres Representing Linguistic Information.* In: *Proceedings of the 15th IEEE Symposium on Logic in Computer Science.* Cambridge, Massachusetts, USA. pp. 38–43.

[Shieber et. al. 1983] Stuart Shieber, Hans Uszkoreit, Fernando Pereira, Jane Robinson, and Mabry Tyson. 1983. *The formalism and implementation of PATR-II.* In: Barbara J. Grosz and Mark E. Stickel, editors, *Research on Interactive Acquisition and Use of Knowledge.* AI Center, SKI International, Menlo Park, California, USA. pp. 39–79.

[Shieber 1986] Stuart Shieber. 1986. *Introduction to Unification-Based Approaches to Grammar.* CSLI Lectur Notes Number 4. Center for the Study of Language and Information, Stanford, California, USA.

[Simov 2001] Kiril Simov. 2001. *Grammar Extraction from an HPSG Corpus.* In: *Proceedings of the RANLP 2001 Conference.* Bulgaria. pp. 285–287.

[Simov 2002] Kiril Simov. 2002. *Grammar Extraction and Refinement from an HPSG Corpus.* In: *Proceedings of the ESSLLI Workshop on Machine Learning Approaches in Computational Linguistics.* Trento, Italy. pp 38–55.

[Simov 2004] Kiril Iv. Simov. 2004 *HPSG-based annotation scheme for corpora development and parsing evaluation.* In: Nicolas Nicolov and Kalina Botcheva and Galia Angelova and Ruslan Mitkov (eds.) *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003.* John Benjamins, Amsterdam/Philadelphia. Current Issues in Linguistic Theory. volume 260. pp. 327–336.

[Simov et. al. 1990] Kiril Simov, Galia Angelova and Elena Paskaleva. 1990. *MORPHO-ASSISTANT: The proper treatment of morphological knowledge.* In: *Proceedings of COLING'90,* volume 3. Helsinki, Finland. pages 453–457.

[Simov et. al. 1992] Kiril Simov, Elena Paskaleva, Mariana Damova and Milena Slavcheva. 1992. *MORPHO-ASSISTANT – a knowledge based system for Bulgarian morphology.* Demo description in *Proceeding of Demo Descriptions of Third conference on Natural Language Application.* Trento, Italy.

[Simov and King 1996] Kiril Simov and Paul J. King. 1996. *Indexing of linguistic knowledge.* In: *Proceedings Logical Aspects of Computational Linguistics* (conference abstracts). Nancy, France. pages 81–84.

[Simov et. al. 2001] Kiril Simov, Zdravko Peev, Milen Kouylekov, Alexander Simov, Marin Dimitrov, Atanas Kiryakov. 2001. *CLaRK - an XML-based System for Corpora Development.* In: *Proceedings of the Corpus Linguistics 2001 Conference.* UK. pp. 558–560.

[Simov and Osenova 2001] Kiril Simov and Petya Osenova. 2001. *A Hybrid System for MorphoSyntactic Disambiguation in Bulgarian.* In: *Proceedings of the RANLP 2001 Conference.* Bulgaria. pp. 288–290.

[Simov et. al. 2002a] Kiril Simov, Milen Kouylekov, Alexander Simov. 2002. *Incremental Specialization of an HPSG-Based Annotation Scheme.* In: *Proceedings of the Workshop on "Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data".* The LREC Conference. Canary Islands, Spain. Pages: 16–23.

[Simov et. al. 2002b] Kiril Simov, Milen Kouylekov, Alexander Simov. 2002. *Cascaded Regular Grammars over XML Documents.* In: *Proceedings of the 2nd Workshop on NLP and XML (NLPXML-2002).* Taipei, Taiwan. pp. 51–58.

[Simov et. al. 2002c] Kiril Simov, Gergana Popova and Petya Osenova. 2002. *HPSG-based syntactic treebank of Bulgarian (BulTreeBank).* In: *"A Rainbow of Corpora: Corpus Linguistics and the Languages of the World"*, edited by Andrew Wilson, Paul Rayson, and Tony McEnery; Lincom-Europa, Munich, pp. 135-142.

[Simov et. al. 2003] Kiril Simov, Alexander Simov, Milen Kouylekov. 2002. *Constraints for Corpora Development and Validation.* In: *Proceedings of the Corpus Linguistics 2003 Conference.* Lancaster, UK. pp. 698–705.

[Simov and Osenova 2003] Kiril Simov and Petya Osenova. 2003. *Practical Annotation Scheme for an HPSG Treebank of Bulgarian.* In: *Proceedings of the 4th Workshop on Linguistically Interpreteted Corpora.* EACL. Budapest, Hungary.

[Simov et. al. 2004a] Kiril Simov, Alexander Simov and Petya Osenova. 2004. *An XML Architecture for Shallow and Deep Processing.* In: *Proceedings of the ESSLLI 2004 Workshop on Combining Shallow and Deep Processing for NLP.* Nancy, France. pp. 51–60.

[Simov et. al. 2004b] Kiril Simov, Petya Osenova, Alexander Simov and Milen Kouylekov. 2004. *Design and Implementation of the Bulgarian HPSG-based Treebank.* In: Erhard Hinrichs and Kiril Simov eds. *Special Issue on Treebanks and Linguistic Theories.* Research on Language & Computation. Springer Science+Business Media B.V., Formerly Kluwer Academic Publishers B.V. Volume 2, Number 4. Pages: 495–522.

[Slavcheva 2002] Milena Slavcheva. 2002. *Segmentation Layers in the Group of the Predicate: a Case Study of Bulgarian within the BulTreeBank Framework.* In: *Proceedings of The First TLT Workshop.* Sozopol, Bulgaria. PP. 199–210.

[Smolka 1988] Gert Smolka. 1988. *A Feature Logic with Subsorts.* LILOG-Report 33. IBM Germany.

[Тотков и Крушков 1989] Георги Тотков и Христо Крушков. 1989. *Приближения морфологический анализ болгарского текста.* In: *Proceedings of the Conference Intelligent Management Systems.* България. стр. 141-147.

[Ule and Simov 2004] Tylman Ule and Kiril Simov. 2004. *Unexpected Productions May Well be Errors.* In: *Proceedings of Fourth International Conference on Language Resources and Evaluation (LREC 2004).* Lisbon, Portugal.