

LINGUISTICS VS. LANGUAGE
TECHNOLOGY



语言学vs语言技术

——在构建和使用构式库时

北京大学 中文信息处理 唐乾桐

IN CONSTRUCTION BUILDING
AND USE

语言学vs语言技术

语言学 🤝 语言技术

语言学 🤝

语言技术

计算语言学

自然语言处理

语言工程

语言学 🤝

语言技术

计算语言学

自然语言处理

语言工程



Background

的背景

Background

1. **地位的不平等。** 语言学在计算语言学中的弱势地位，与统计方法在计算语言学中的强势地位。
2. **关注的脱节。** 语言技术的研究越来越与语言学的关注脱节，语言技术专家只对系统达成他们所期望的行为感兴趣，对语言本身兴趣不大。
3. **合作领域太窄。** 服务于语言技术早期研究的那些语言学研究工作，其中很多都是在**生成语法**的旗帜下进行的；语言技术仍然没有被**发展成熟的其他语言学分支**所触及——例如类型学和接触语言学，社会语言学，精神语言学或词典学（包括词汇语义学），等等。
4. **单向的交流。** 除了极少数例外，计算语言学方面的工作对一般语言学理论或方法的发展几乎没有影响。

的背景

Background

1. 地位的不平等。 语言学在计算语言学中的弱势地位，与统计方法在计算语言学中的强势地位。
2. 关注的脱节。 语言技术的研究越来越与语言学的关注脱节，语言技术专家只对系统达成他们所期望的行为感兴趣，对语言本身兴趣不大。
 - ① “基于明确的语言知识的应用程序发展得不太好”；
 - ② “投资机构（主要来自美国）受短期实际目标的驱动”；
 - ③ “语言学主要关注语法（并且主要是英语的语法），其理论非常模糊、复杂和自我中心，以至于其他学科的研究者很难理解”
3. 合作领域太窄。 服务于语言技术早期研究的那些语言学研究工作，其中很多都是在生成语法的旗帜下进行的；语言技术仍然没有被发展成熟的其他语言学分支所触及——例如类型学和接触语言学，社会语言学，精神语言学或词典学（包括词汇语义学），等等。
4. 单向的交流。 除了极少数例外，计算语言学方面的工作对一般语言学理论或方法的发展几乎没有影响。

的背景

Background

1. **地位的不平等。** 语言学在计算语言学中的弱势地位，与统计方法在计算语言学中的强势地位。
2. **关注的脱节。** 语言技术的研究越来越与语言学的关注脱节，语言技术专家只对系统达成他们所期望的行为感兴趣，对语言本身兴趣不大。
3. **合作领域太窄。** 服务于语言技术早期研究的那些语言学研究工作，其中很多都是在**生成语法**的旗帜下进行的；语言技术仍然没有被**发展成熟的其他语言学分支**所触及——例如类型学和接触语言学，社会语言学，精神语言学或词典学（包括词汇语义学），等等。
4. **单向的交流。** 除了极少数例外，计算语言学方面的工作对一般语言学理论或方法的发展几乎没有影响。

的背景

Background

4. 单向的交流。除了极少数例外，计算语言学方面的工作对一般语言学理论或方法的发展几乎没有影响。

- 交流不应该是单向的，两个领域会在双向合作中受益匪浅：

① 一方面，数据驱动的LT为进行“**大规模假设驱动的经验语言学调查**” (large-scale hypothesis-driven empirical linguistic investigations) 提供了一种完善而严谨的方法（例如，参见Abney, 2011）。这种方法与通过它开发和完善的分析工具一起，有可能扩大基于语料库的语言学的范围，这将有利于诸如构式构建之类的工作，还将有利于对诸如“**深植 (entrenchment)**” (Schmid 2010) 等构式 (?) 的实证研究，等等。

② 另一方面，由于语言现象的Zipf分布，即使对于一些有非常庞大语料库的语言，数据驱动的LT虽然能揭示语言学家所不了解的一些语言知识，但也注定无法学到语言学家已经确信的一些语言知识。因此，如果将这**两种知识来源以某种方式结合起来**，那将是非常有用的。在这方面，**高度精炼的语言知识——例如在知识网络或构式库中编码的知识——**特别值得关注，但同时也提出了许多方法上的挑战。

似乎是指将数据分为
训练集、测试集
然后进行学习的方法

语言学 🤝 语言技术

语言学 🤝 语言技术

- ① 构建知识库 📌 从知识库中获取知识
- ② 提出概念和任务 📌 完成任务
- ③ 模型可解释性研究 📌 端到端的模型
- ④ 评测模型的方法 📌 端到端的模型
- ⑤ 合作提出和改进模型

LINGUISTICS VS. LANGUAGE
TECHNOLOGY



语言学🤝语言技术

——在构建和使用构式库时

北京大学 中文信息处理 唐乾桐

IN CONSTRUCTION BUILDING
AND USE

本文案例

The Case

本文案例

The Case

① Karp (通用词汇基础设施)

语言学

由于构建Karp时对单个词典资源及其整合的研究的大规模经验化的性质，及其所涉及的词汇资源的多样性，它极有可能对词典学、词汇语义学和词汇类型学做出重大理论贡献。

语言技术

Karp主要应用于LT应用程序。通过在Karp中获得的词汇信息，可以对语料库进行标注。Karp支持创建、策划和集成词汇资源，并且可以与可在线浏览和下载的版本并行维护。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

语言学

提出“构式”这一概念。提出“在真实文本中自动识别构式”这一任务。投入大量精力对多词表达（MWE, multi-word expressions）进行分类和表征。

语言技术

处理这个任务。引入在语言技术领域研究MWE所运用的方法，用来从标注语料库中生成潜在构式。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

in London

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

in [PROPER NOUN]

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

[PREPOSITION] [PROPER NOUN]

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

[PREPOSITION] [NP]

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

in London

in [PROPER NOUN]

[PREPOSITION] London

[PREPOSITION] [PROPER NOUN]

in [NP]

[PREPOSITION] [NP]

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

in London

in [PROPER NOUN]

[PREPOSITION] London

POS标签由HunPos指派

[PREPOSITION] [PROPER NOUN]

in [NP]

短语结构标签由MaltParser提供的自动句法分析指派

[PREPOSITION] [NP]

(1) 提取和计算在大型语料库中出现的句法模式；

(2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

in London

in [PROPER NOUN]

[PREPOSITION] [PROPER NOUN]

in [NP]

[PREPOSITION] [NP]

[PREPOSITION] London

基于统计搭配度量的排序

- (1) 提取和计算在大型语料库中出现的句法模式；
- (2) 根据相关性度量对这些模式进行排序。

本文案例

The Case

② 挖掘语料库中的潜在构式

目的是捕获那些可以被描述为n个相邻单元的序列（所谓的n-gram）的构式，这些单元可以是特定单词、具有特定词性（POS）标签的单词或短语。

Forsberg (2014)

将以此法发现的1200个排名最高的模式（以及每种模式的至多五个最常见的实例）呈现给三位语言学家，以评估它们作为潜在构式的合适性。

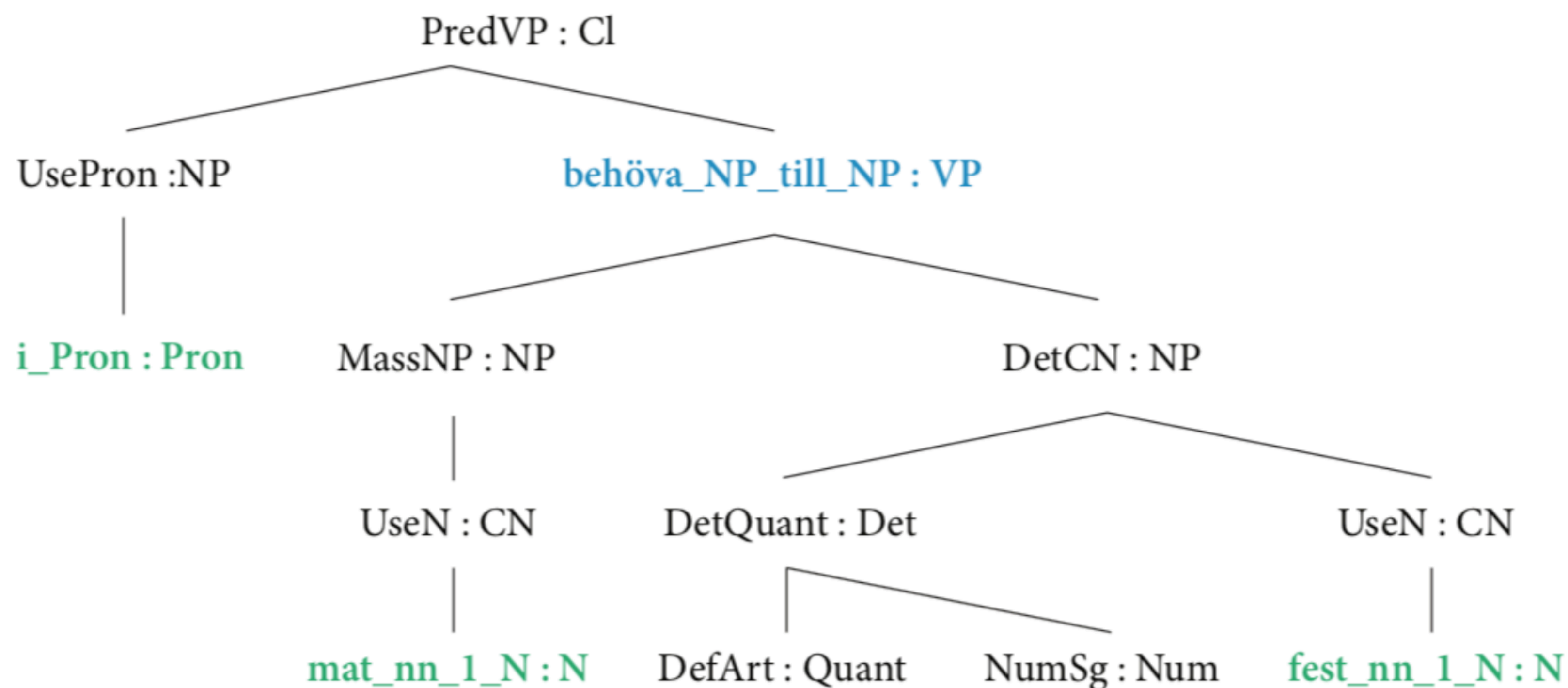
总共有大约200个有前途的潜在构式被发现。多亏了发达的语言技术，语言学家只用手动检查7200个示例（1200 x 6），而不是1900万字的文本。

- （1）提取和计算在大型语料库中出现的句法模式；
- （2）根据相关性度量对这些模式进行排序。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

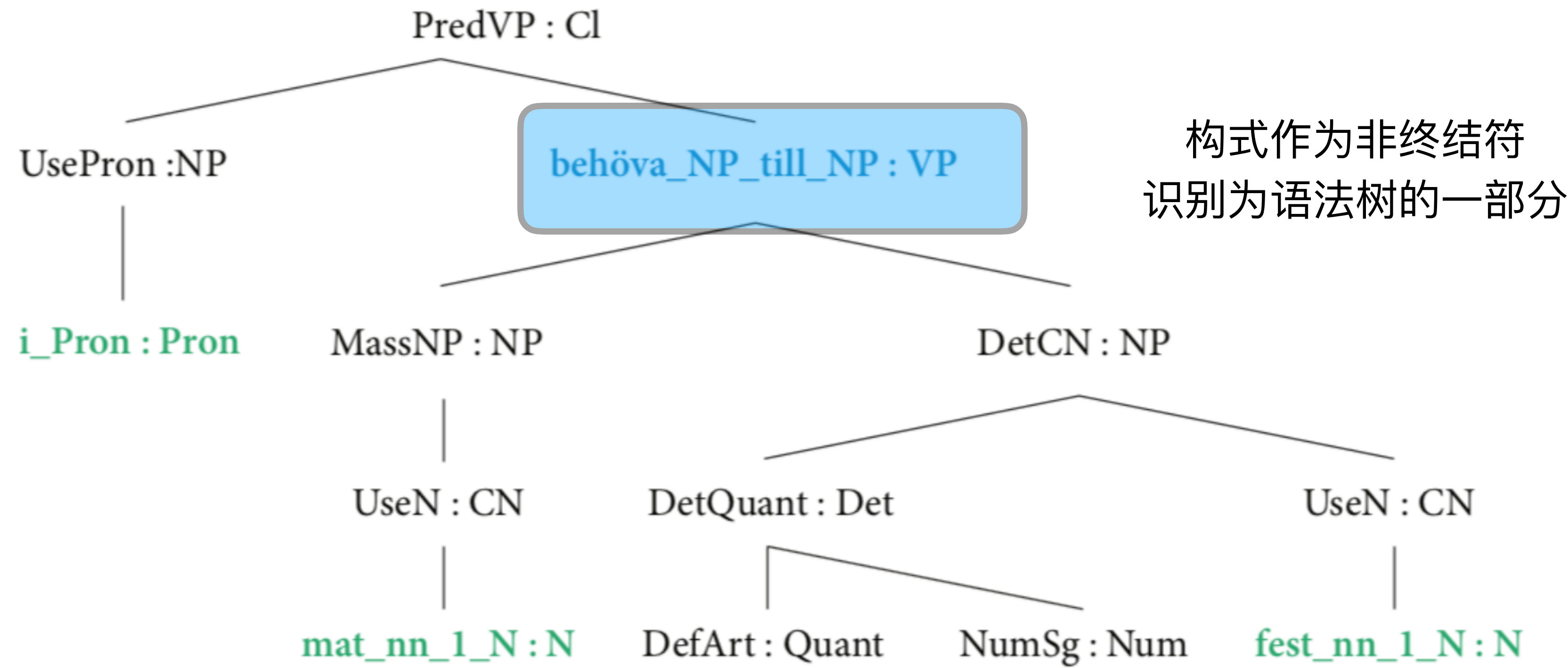


虽然框架语义表示已经在语言技术社区中获得了相当程度的关注，但作为适合在NLP系统中部署的、实用的、轻量级的语义表示，构式语法虽然来自同一知识源却没有得到足够的关注。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

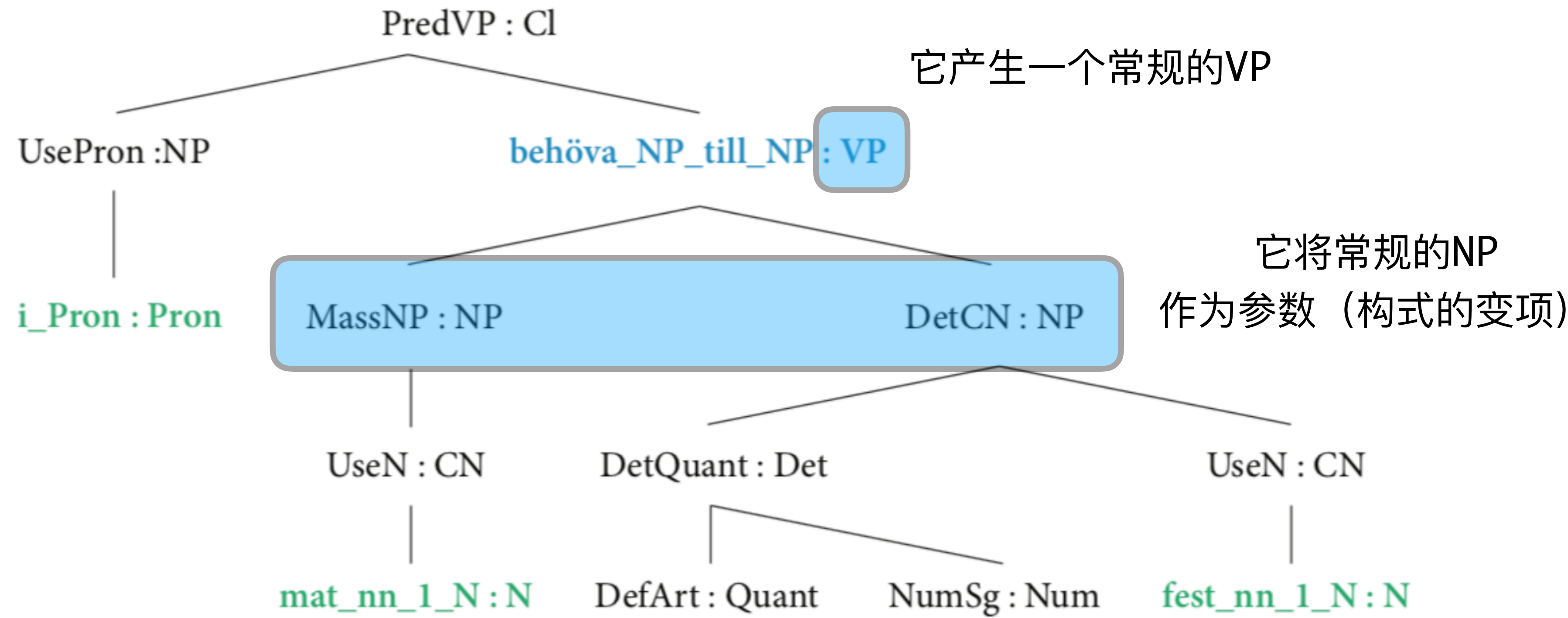


虽然框架语义表示已经在语言技术社区中获得了相当程度的关注，但作为适合在NLP系统中部署的、实用的、轻量级的语义表示，构式语法虽然来自同一知识源却没有得到足够的关注。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

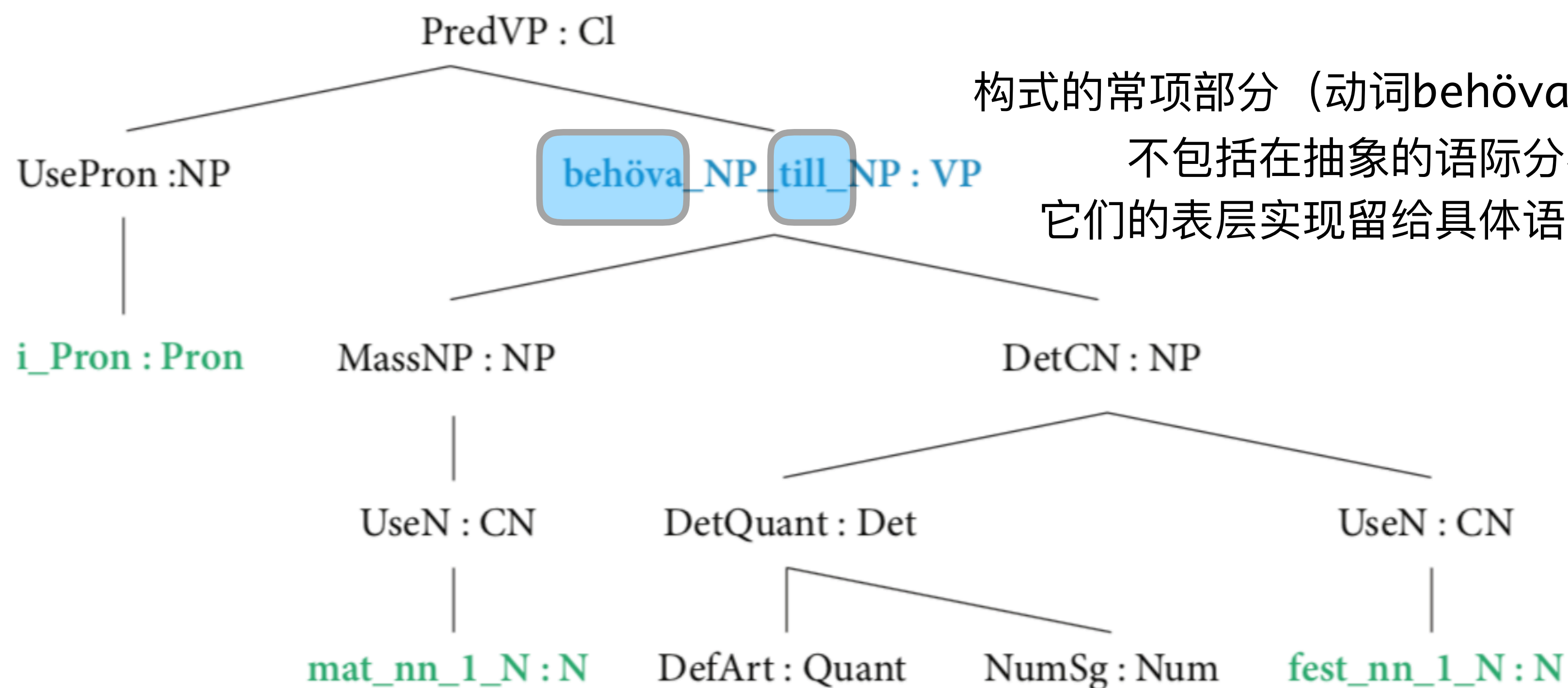


虽然框架语义表示已经在语言技术社区中获得了相当程度的关注，但作为适合在NLP系统中部署的、实用的、轻量级的语义表示，构式语法虽然来自同一知识源却没有得到足够的关注。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析



虽然框架语义表示已经在语言技术社区中获得了相当程度的关注，但作为适合在NLP系统中部署的、实用的、轻量级的语义表示，构式语法虽然来自同一知识源却没有得到足够的关注。

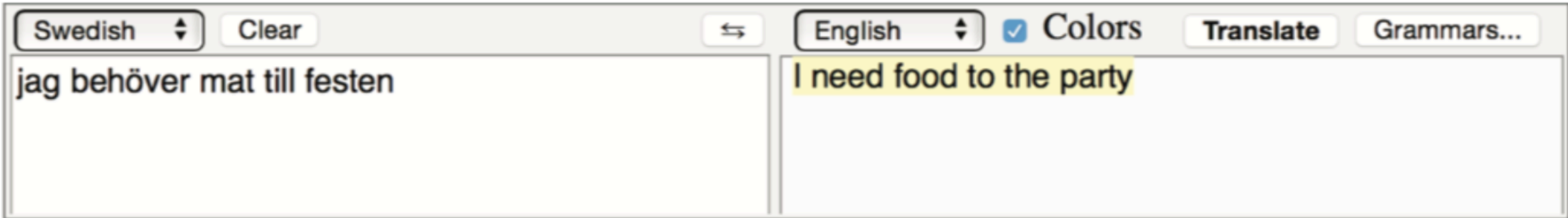
本文案例

The Case

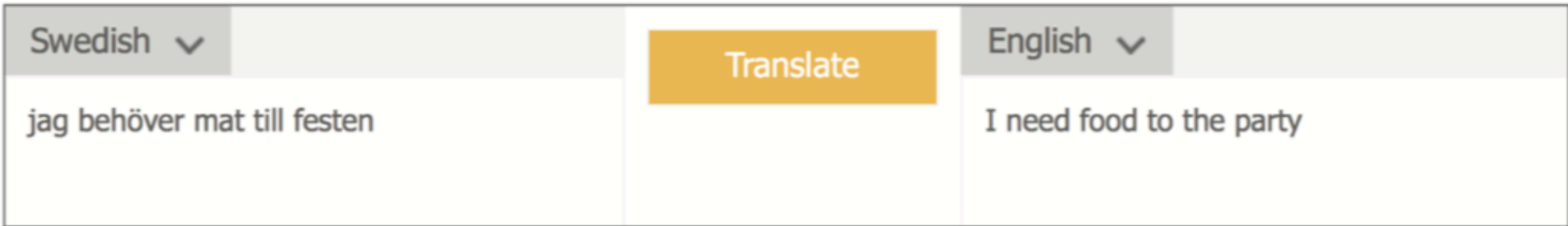
③ 使用瑞典语构式库进行语言分析

计算构式和潜在多语言构式库的作用在机器翻译中变得更加明显该构式的介词‘till’字面翻译是‘to’，但在这个构式中的含义应该是‘for’。我们不可能在词典中列出此构式的所有实例。

基于规则的机器翻译
(GF Wide Coverage Translator)



基于统计的机器翻译
(Bing Translator)

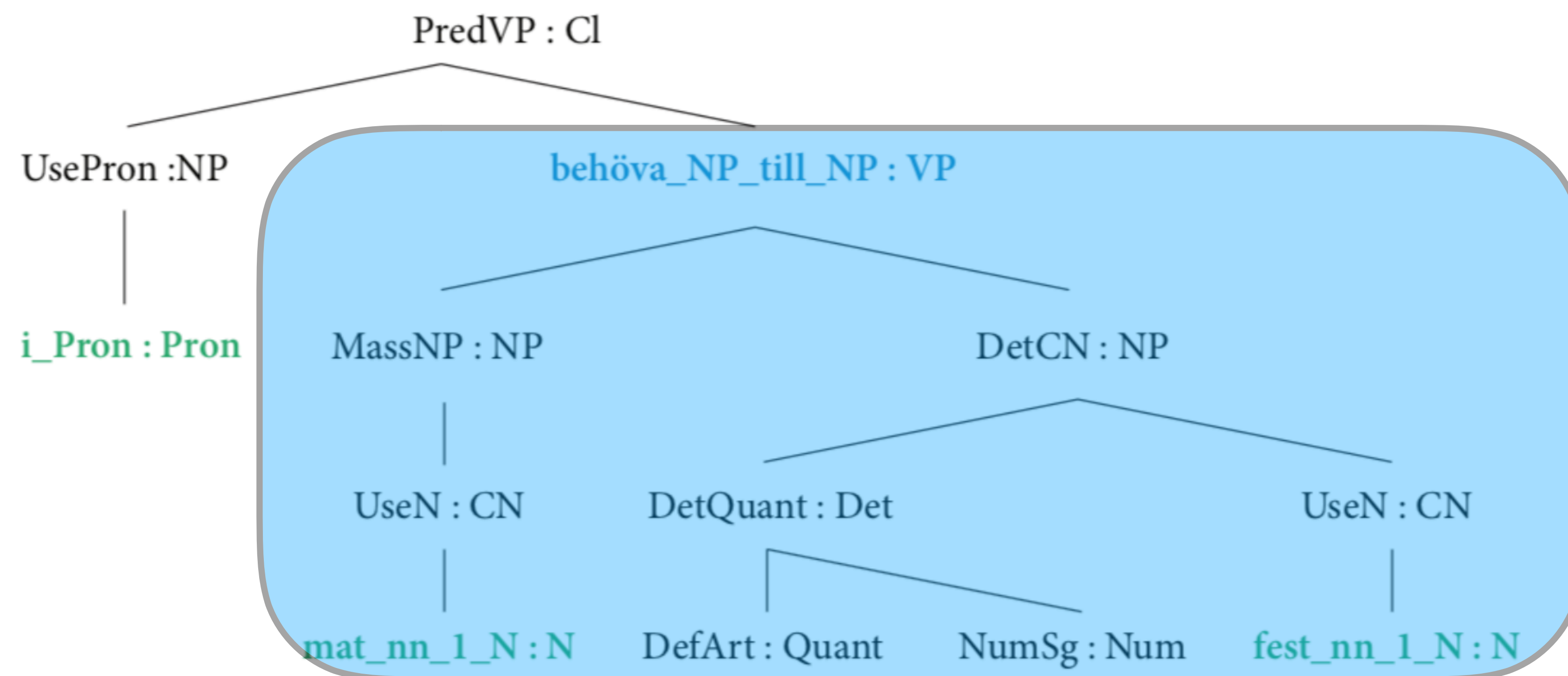


充分说明了对多语言构式的需求

本文案例

The Case

③ 使用瑞典语构式库进行语言分析



为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

behöva_något_till_något

结构式: $[\text{behöva}^1 \text{ NP}_1 \text{ till}^1 \text{ NP}_2 \mid \text{VP}]$

为了得到构式的这类句法树，需要将语言导向的构式描述转换为语法框架（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

behöva_något_till_något

结构式：

[behöva¹NP₁till¹NP₂ | VP]

该结构式结合了四个内部CE

为了得到构式的这类句法树，需要将语言导向的构式描述转换为语法框架（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

behöva_något_till_något

结构式：

[behöva¹ NP₁ till¹ NP₂ | VP]

该结构式结合了四个内部CE

每个CE进一步用包含若干属性的特征矩阵进行描述

{role = “State” name = “State” cat = “V” lu = “behöva..1”}

{role = “Requirement” name = “Requirement” cat = “NP”}

{name = “P” cat = “P” lu = “till..1”}

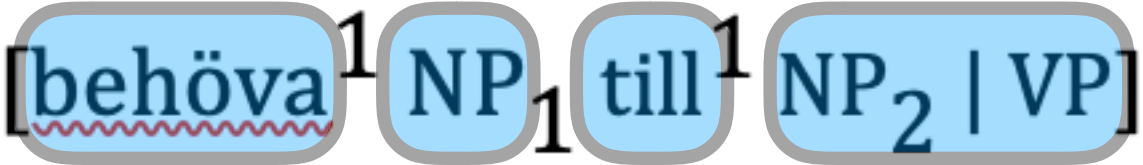
{role = “Purpose” name = “Purpose” cat = “NP|VP”}

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

本文案例


The Case

behöva_något_till_något

结构式： 该结构式结合了四个内部CE

每个CE进一步用包含若干属性的特征矩阵进行描述

语义角色 名称 范畴 常项的词汇单元描述符 (来自SALDO)


{role = "State" name = "State" cat = "V" lu = "behöva..1"}
{role = "Requirement" name = "Requirement" cat = "NP"}
{name = "P" cat = "P" lu = "till..1"}
{role = "Purpose" name = "Purpose" cat = "NP|VP"}

特征矩阵集非常详细，但没有与结构式中给出的对应元素建立明确的指示。在许多矩阵中，每个元素的说明顺序往往是多样的；它不一定遵循结构式中给出的元素说明规范。当我们为了语言技术分析开始分析数据库时，系统地依据不同的规范及其相应元素对于改进自动分析非常重要。

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

用GF来描述VP构式

语法框架 (Grammatical framework, GF) 是一种范畴语法, 也是一种实现计算语法的框架。GF的特点是它采用两级方法来表示自然语言:

1. 抽象语法, 定义了与具体语言无关的结构;
2. 具体语法, 定义了与具体语言相关的特定语法以及抽象语法的词法实现。

为了得到构式的这类句法树, 需要将语言导向的构式描述转换为语法框架 (Grammatical Framework, GF), 一种计算形式文法流派。

本文案例

The Case

用GF来描述VP构式

语法框架 (Grammatical framework, GF) 是一种范畴语法，也是一种实现计算语法的框架。

GF的特点是它采用两级方法来表示自然语言：

1. 抽象语法，定义了与具体语言无关的结构；

相同的抽象语法可以对应于许多（多语言的）具体语法——从抽象语法树映射到特征结构和字符串。

GF提供了一个**通用的资源语法库RGL** (Ranta, 2009)，目前有30种语言运用了相同的抽象语法。RGL有一个高级接口，提供像 $\text{mkVP: } V \rightarrow \text{NP} \rightarrow \text{VP}$ 这样的构造函数，用于根据一个动词和一个名词短语构建一个动词短语，但不需要指定低级的细节，如**屈折范式，句法论元和单词顺序**。

2. 具体语法，定义了与具体语言相关的特定语法以及抽象语法的词法实现。

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架** (Grammatical Framework, GF)，一种计算形式文法流派。

本文案例

The Case

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

1. 预处理：结构式的自动标准化，一致性检查和重写；
2. 自动生成GF语法的抽象和具体语法；
3. 对获得的语法进行半自动验证。

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

1. 预处理：结构式的自动标准化，一致性检查和重写；

- (2) a. *behöva_något_till_något*: [*behöva*¹ NP₁ *till*¹ NP₂ | VP]
e.g. *behöva kvällen till att plugga* ‘need the evening to study’
- (3) a. *verba_av_sig.transitiv*: [V *av*¹ Pn_{refl} (NP)]
e.g. *ta av mig skorna* ‘take off myself the shoes’
- (4) a. *snacka_NP*: [*snacka*¹ | *prata*¹ | *tala*¹ NP_{indef}]
e.g. *prata skolminnen* ‘talk school memories’
- (5) a. *få_resultativ.agentiv*: [*få*¹ NP PcP]
e.g. *få gräsmattan klippt* ‘get the lawn trimmed’
- (6) a. *x-städa*: [N | Adj + *städa*¹]
e.g. *storstäda* ‘bigclean (V)’

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

1. 预处理：结构式的自动标准化，一致性检查和重写；

预处理前：

- (2) a. behöva_något_till_något: [behöva¹ NP₁ till¹ NP₂|VP]
e.g. *behöva kvällen till att plugga* ‘need the evening to study’
- (3) a. verba_av_sig.transitiv: [V av¹ Pn_{refl} (NP)]
e.g. *ta av mig skorna* ‘take off myself the shoes’
- (4) a. snacka_NP: [snacka¹|prata¹|tala¹ NP_{indef}]
e.g. *prata skolminnen* ‘talk school memories’
- (5) a. få_resultativ.agentiv: [få¹ NP PcP]
e.g. *få gräsmattan klippt* ‘get the lawn trimmed’
- (6) a. x-städa: [N|Adj+städa¹]
e.g. *storstäda* ‘bigclean (V)’

预处理后：

- (2) b. behöva_V NP₁ till_{Prep} NP₂ | behöva_V NP till_{Prep} VP
- (3) b. V av_{Prep} Pron_{refl} NP | V av_{Prep} Pron_{refl}
- (4) b. snacka|prata|tala_V aSg_Det CN | snacka|prata|tala_V aPl_Det CN |
snacka|prata|tala_V CN
- (5) b. få_V NP PcP_{perf}
- (6) b. N + städa_V | A + städa_V

为了得到构式的这类句法树，需要将语言导向的构式描述转换为语法框架（Grammatical Framework, GF），一种计算形式文法流派。

预处理前：

- (2) a. behöva_något_till_något: [behöva¹ NP₁ till¹ NP₂|VP]
e.g. *behöva kvällen till att plugga* ‘need the evening to study’
- (3) a. verba_av_sig.transitiv: [V av¹ Pn_{refl} (NP)]
e.g. *ta av mig skorna* ‘take off myself the shoes’
- (4) a. snacka_NP: [snacka¹|prata¹|tala¹ NP_{indef}]
e.g. *prata skolminnen* ‘talk school memories’
- (5) a. få_resultativ.agentiv: [få¹ NP PcP]
e.g. *få gräsmattan klippt* ‘get the lawn trimmed’
- (6) a. x-städa: [N|Adj+städa¹]
e.g. *storstäda* ‘bigclean (V)’

预处理后：

- (2) b. behöva_V NP₁ till_{Prep} NP₂ | behöva_V NP till_{Prep} VP
- (3) b. V av_{Prep} Pron_{refl} NP | V av_{Prep} Pron_{refl}
- (4) b. snacka|prata|tala_V aSg_Det CN | snacka|prata|tala_V aPl_Det CN |
snacka|prata|tala_V CN
- (5) b. få_V NP PcP_{perf}
- (6) b. N + städa_V | A + städa_V

- 1.对特征结构中的结构式和属性值进行标准化，修复由于手动标注而导致的各种不一致性。
- 2.在可选的CE、有选项类型的CE的情况下，正式地将几个构式压缩成一个。重写原始构式，以便对于每个组合，存在单独的替代结构。然而，这不适用于有选项的LU。如果CE由一组固定的LU表示，我们假设它们是可互换的（同义词）。否则，它们应该被拆分成可互相替代的构式（单独的条目），或者CE应该更通用（更多变）。
- 3.重写的结构式富含来自特征矩阵的附加形态句法信息，因此可以获得完整的描述。
- 4.瑞典语构式库中使用的语法类别被转换为GF类别。在特定情况下，转换可能会导致更一般或更具体的描述，并且可能包括形态句法标记，并且可能取决于上下文CE。这需要随后重写整个结构。不过，有几个类别在这一步中并没有被转换，他们的转换被推迟到GF语法生成的环节中去了。例如，Pc（分词）和PcP（分词短语）不分别转换为V（动词）和VP，因为它们必须在具体语法中区别对待：PcP是一个VP，它最终被转换成AP（形容词短语）或Adv（修饰vp的副词），如få_resultativ.agentiv（5）所示。

本文案例

The Case

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

1. 预处理：结构式的自动标准化，一致性检查和重写；

预处理后：

(2) b. $\text{behöva}_V \text{NP}_1 \text{till}_{\text{Prep}} \text{NP}_2 \mid \text{behöva}_V \text{NP} \text{till}_{\text{Prep}} \text{VP}$

(3) b. $\text{V} \text{av}_{\text{Prep}} \text{Pron}_{\text{refl}} \text{NP} \mid \text{V} \text{av}_{\text{Prep}} \text{Pron}_{\text{refl}}$

(4) b. $\text{snacka|prata|tala}_V \text{aSg_Det CN} \mid \text{snacka|prata|tala}_V \text{aPl_Det CN} \mid$
 $\text{snacka|prata|tala}_V \text{CN}$

(5) b. $\text{få}_V \text{NP PcP}_{\text{perf}}$

(6) b. $\text{N} + \text{städa}_V \mid \text{A} + \text{städa}_V$

这些被重写的构式结构性描述提供了足够的信息来生成基于构式的抽象的和具体的句法

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

2. 自动生成GF语法的抽象和具体语法；

每个构式由一个或多个语法规则（函数）表示，
具体取决于在预处理阶段生成（重写）了多少替代性的结构描述。
每个函数采用一个或多个参数，这些参数对应于相应构式描述的变项CE。

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

2. 自动生成GF语法的抽象和具体语法；

每个构式由一个或多个语法规则（函数）表示，
具体取决于在预处理阶段生成（重写）了多少替代性的结构描述。
每个函数采用一个或多个参数，这些参数对应于相应构式描述的变项CE。

预处理后：

- (2) b. $\text{behöva}_V \text{NP}_1 \text{till}_{\text{Prep}} \text{NP}_2 \mid \text{behöva}_V \text{NP} \text{till}_{\text{Prep}} \text{VP}$
- (3) b. $\text{V} \text{av}_{\text{Prep}} \text{Pron}_{\text{refl}} \text{NP} \mid \text{V} \text{av}_{\text{Prep}} \text{Pron}_{\text{refl}}$
- (4) b. $\text{snacka|prata|tala}_V \text{aSg_Det CN} \mid \text{snacka|prata|tala}_V \text{aPl_Det CN} \mid \text{snacka|prata|tala}_V \text{CN}$
- (5) b. $\text{få}_V \text{NP PcP}_{\text{perf}}$
- (6) b. $\text{N} + \text{städa}_V \mid \text{A} + \text{städa}_V$

抽象语法的重写结构描述：

- (2) c. $\text{behöva_något_till_något}_1: \text{NP} \rightarrow \text{NP} \rightarrow \text{VP}$
 $\text{behöva_något_till_något}_2: \text{NP} \rightarrow \text{VP} \rightarrow \text{VP}$
- (3) c. $\text{verba_av_sig_transitiv}_1: \text{V} \rightarrow \text{NP} \rightarrow \text{VP}$
 $\text{verba_av_sig_transitiv}_2: \text{V} \rightarrow \text{VP}$
- (4) c. $\text{snacka_NP}_1: \text{CN} \rightarrow \text{VP}$
 $\text{snacka_NP}_2: \text{CN} \rightarrow \text{VP}$
 $\text{snacka_NP}_3: \text{CN} \rightarrow \text{VP}$
- (5) c. $\text{få_resultativ_agentiv}: \text{NP} \rightarrow \text{VP} \rightarrow \text{VP}$
- (6) c. $\text{x_städa}_1: \text{N} \rightarrow \text{VP}$
 $\text{x_städa}_2: \text{A} \rightarrow \text{VP}$

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架**（Grammatical Framework, GF），一种计算形式文法流派。

预处理后：

- (2) b. behöva_V NP₁ till_{Prep} NP₂ | behöva_V NP till_{Prep} VP
- (3) b. V av_{Prep} Pron_{refl} NP | V av_{Prep} Pron_{refl}
- (4) b. snacka|prata|tala_V aSg_Det CN | snacka|prata|tala_V aPl_Det CN | snacka|prata|tala_V CN
- (5) b. få_V NP PcP_{perf}
- (6) b. N + städa_V | A + städa_V

至于具体的语法，通过系统地应用GF RGL提供的高级构造函数，可以在GF中实现许多构式：

- mkVP: VP → Adv → VP
- mkVP: V2 → NP → VP
- mkV2: V → V2
- mkV: Str → V
- mkAdv: Prep → NP → Adv
- mkPerp: Str → Prep
- etc.

抽象语法的重写结构描述：

- (2) c. behöva_något_till_något₁: NP → NP → VP
behöva_något_till_något₂: NP → VP → VP
- (3) c. verba_av_sig_transitiv₁: V → NP → VP
verba_av_sig_transitiv₂: V → VP
- (4) c. snacka_NP₁: CN → VP
snacka_NP₂: CN → VP
snacka_NP₃: CN → VP
- (5) c. få_resultativ_agentiv: NP → VP → VP
- (6) c. x_städa₁: N → VP
x_städa₂: A → VP

预处理后：

- (2) b. behöva_V NP₁ till_{Prep} NP₂ | behöva_V NP till_{Prep} VP
- (3) b. V av_{Prep} Pron_{refl} NP | V av_{Prep} Pron_{refl}
- (4) b. snacka|prata|tala_V aSg_Det CN | snacka|prata|tala_V aPl_Det CN | snacka|prata|tala_V CN
- (5) b. få_V NP PcP_{perf}
- (6) b. N + städa_V | A + städa_V

至于具体的语法，通过系统地应用GF RGL提供的高级构造函数，可以在GF中实现许多构式：

- mkVP: VP → Adv → VP
- mkVP: V2 → NP → VP
- mkV2: V → V2
- mkV: Str → V
- mkAdv: Prep → NP → Adv
- mkPerp: Str → Prep
- etc.

抽象语法的重写结构描述：

- (2) c. behöva_något_till_något₁: NP → NP → VP
behöva_något_till_något₂: NP → VP → VP
- (3) c. verba_av_sig_transitiv₁: V → NP → VP
verba_av_sig_transitiv₂: V → VP
- (4) c. snacka_NP₁: CN → VP
snacka_NP₂: CN → VP
snacka_NP₃: CN → VP
- (5) c. få_resultativ_agentiv: NP → VP → VP
- (6) c. x_städa₁: N → VP
x_städa₂: A → VP

例如：

- (2) d. behöva_något_till_något₁ np₁ np₂ =
mkVP
(mkVP (mkV2 (mkV “behöver”)) np₁)
(mkAdv (mkPrep “till”) np₂)

预处理后：

- (2) b. behöva_V NP₁ till_{Prep} NP₂ | behöva_V NP till_{Prep} VP
- (3) b. V av_{Prep} Pron_{refl} NP | V av_{Prep} Pron_{refl}
- (4) b. snacka|prata|tala_V aSg_Det CN | snacka|prata|tala_V aPl_Det CN | snacka|prata|tala_V CN
- (5) b. få_V NP PcP_{perf}
- (6) b. N + städa_V | A + städa_V

至于具体的语法，通过系统地应用GF RGL提供的高级构造函数，可以在GF中实现许多构式：

- mkVP: VP → Adv → VP
- mkVP: V2 → NP → VP
- mkV2: V → V2
- mkV: Str → V
- mkAdv: Prep → NP → Adv
- mkPerp: Str → Prep
- etc.

抽象语法的重写结构描述：

- (2) c. behöva_något_till_något₁: NP → NP → VP
behöva_något_till_något₂: NP → VP → VP
- (3) c. verba_av_sig_transitiv₁: V → NP → VP
verba_av_sig_transitiv₂: V → VP
- (4) c. snacka_NP₁: CN → VP
snacka_NP₂: CN → VP
snacka_NP₃: CN → VP
- (5) c. få_resultativ_agentiv: NP → VP → VP
- (6) c. x_städa₁: N → VP
x_städa₂: A → VP

例如：

- (2) d. behöva_något_till_något₁ np₁ np₂ =
mkVP
(mkVP (mkV2 (mkV “behöver”)) np₁)
(mkAdv (mkPrep “till”) np₂)

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

1. 预处理：结构式的自动标准化，一致性检查和重写；
2. 自动生成GF语法的抽象和具体语法；

(2) d. behöva_något_till_något₁ np₁ np₂ =
mkVP

(mkVP (mkV2 (mkV “behöver”)) np₁)
(mkAdv (mkPrep “till”) np₂)

3. 对获得的语法进行半自动验证。

为了得到构式的这类句法树，需要将语言导向的构式描述转换为**语法框架** (Grammatical Framework, GF)，一种计算形式文法流派。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

用GF来描述VP构式

将词典式的SweCcn条目转换为计算GF式的构式语法的方法包括以下步骤：

1. 预处理：结构式的自动标准化，一致性检查和重写；
2. 自动生成GF语法的抽象和具体语法；
3. 对获得的语法进行半自动验证。

对于VP构式，得到的计算构式语法模块产生了127个GF函数

为了得到构式的这类句法树，需要将语言导向的构式描述转换为语法框架（Grammatical Framework, GF），一种计算形式文法流派。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

如何评测？

一种可能的评测方式，是用生成的语法解析一些语料并检查结果。

第一步，我们从SweCcn中VP构式的已标注句子中编译了337个例句的基准集合。（评测集）

第二步，使用生成的语法解析每个句子。

本文案例

The Case

③ 使用瑞典语构式库进行语言分析

如何评测？

一种可能的评测方式，是用生成的语法解析一些语料并检查结果。

第一步，我们从SweCcn中VP构式的已标注句子中编译了337个例句的基准集合。（评测集）

第二步，使用生成的语法解析每个句子。

在337个已标注的例句中，281个在构式语法中具有相应的具体功能。句法分析器成功解析了80%的句子。句法分析器有可能不返回任何句法树，这种情况主要是由于SweCcn数据库中的标注不一致或标注错误，例如，特征矩阵需要NP的单数形式，而在标注的示例中存在的是复数形式。这些实例对于编写SweCcn构式的构式描述的语言学家来说是很有用的反馈。

GF句法分析器解析失败的大多数单词都是专有的名词和复合词。关于语法，句法分析器在解析存在形式错误或语法复杂的句子时会失败，这些句子通常包含并列、连词和长句，包含不相关的短语和不符合构造的标点符号。将来，语法分析可以通过“逆向测试”来补充完善，即，我们可以使用GF内置支持来自动生成树库，并测试我们的语法是否能够为每个构式提供正确的句法分析。SweCcn开发人员（语言学家）可以反过来进一步检查和验证生成的句法树。

语言学 🤝 语言技术

- ① 构建知识库 📌 从知识库中获取知识
- ② 提出概念和任务 📌 完成任务
- ③ 模型可解释性研究 📌 端到端的模型
- ④ 评测模型的方法 📌 端到端的模型
- ⑤ 合作提出和改进模型

语言学 🤝 语言技术

① 构建知识库 📌 从知识库中获取知识

② 提出概念和任务 📌 完成任务

③ 模型可解释性研究 📌 端到端的模型

④ 评测模型的方法 📌 端到端的模型

⑤ 合作提出和改进模型

LINGUISTICS VS. LANGUAGE
TECHNOLOGY



语言学vs语言技术

——在构建和使用构式库时

北京大学 中文信息处理 唐乾桐

IN CONSTRUCTION BUILDING
AND USE