Основы обработки текстов

Лекция 7

Статистические методы синтаксического анализа

Мотивация

- СКС-грамматики позволяют определить лучшее дерево разбора (т.е. устранить многозначность)
- Более точное моделирование языка, по сравнению с n-граммами
  - распознавание речи
  - машинный перевод
  - извлечение информации
  - ...

План

- Стохастические контекстно-свободные грамматики (СКС)
  - –разрешение синтаксической многозначности
  - –моделирование языка
- Вероятностная версия алгоритма CKY
- Обучение СКС
- Проблемы СКС
  - –разделение и слияние нетерминалов
  - –СКС с поддержкой лексики
    - алгоритм Коллинза
- Методы оценки

Стохастические контекстно-свободные грамматики

N	множество **нетерминальных символов**
Σ	множество **терминальных символов** (непересекающееся с *N*)
R	множество правил, каждое вида $A \to \beta[p]$ где A - нетерминал, β - строка символов из множества $(\Sigma \cup N)*$ p - вероятность правила $P(\beta \mid A)$, $\sum_{\beta} P(A \to \beta) = 1$
S	символ начала

Пример

Грамматика	Вероятность			Лексикон
S → NP VP	0.8			Det → the | a | that | this
S → Aux NP VP	0.1	+	1.0	0.6 0.2 0.1 0.1
S → VP	0.1			Noun → book | flight | meal | money
NP → Pronoun	0.2			0.1 0.5 0.2 0.2
NP → Proper-Noun	0.2	+	1.0	Verb → book | include | prefer
NP → Det Nominal	0.6			0.5 0.2 0.3
Nominal → Noun	0.3			Pronoun → I | he | she | me
Nominal → Nominal Noun	0.2	+	1.0	0.5 0.1 0.1 0.3
Nominal → Nominal PP	0.5			Proper-Noun → Houston | NWA
VP → Verb	0.2			0.8 0.2
VP → Verb NP	0.5	+	1.0	Aux → does
VP → VP PP	0.3			1.0
PP → Prep NP	1.0			Prep → from | to | on | near | through
				0.25 0.25 0.1 0.2 0.2

Разрешение многозначности

- Вероятность разбора

$$P(T,S) = \prod_{i=1}^{n} P(RHS_i|LHS_i)$$

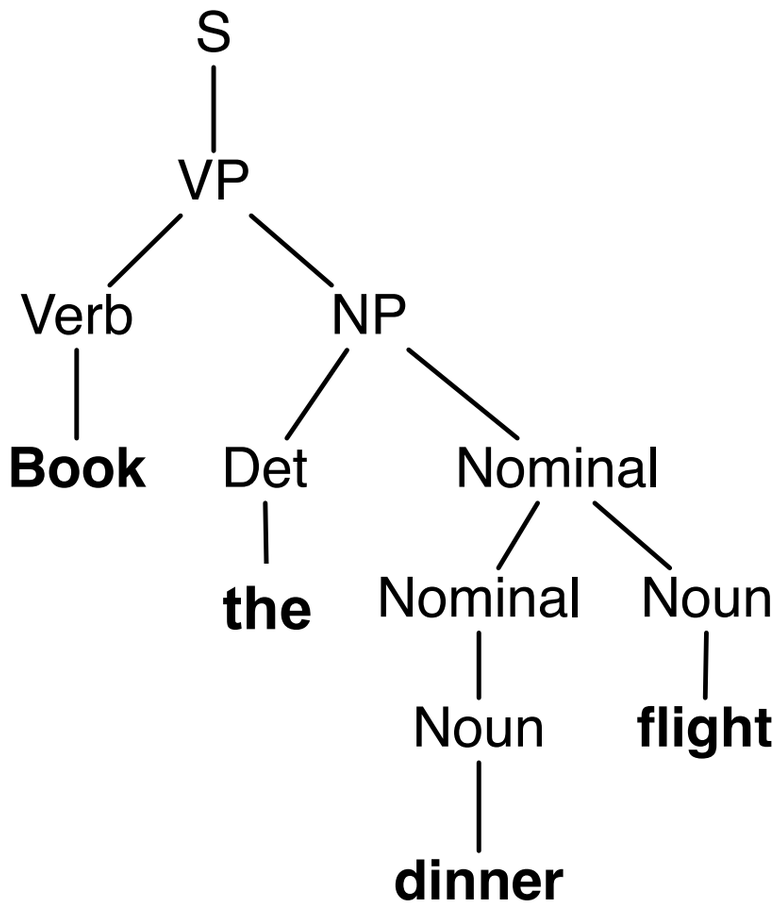
- Вероятность $P(T,S) = P(T)P(S|T) = P(T)$
- Выбор наиболее вероятного дерева разбора $\hat{T}(S) = \arg\max_T P(T|S)$

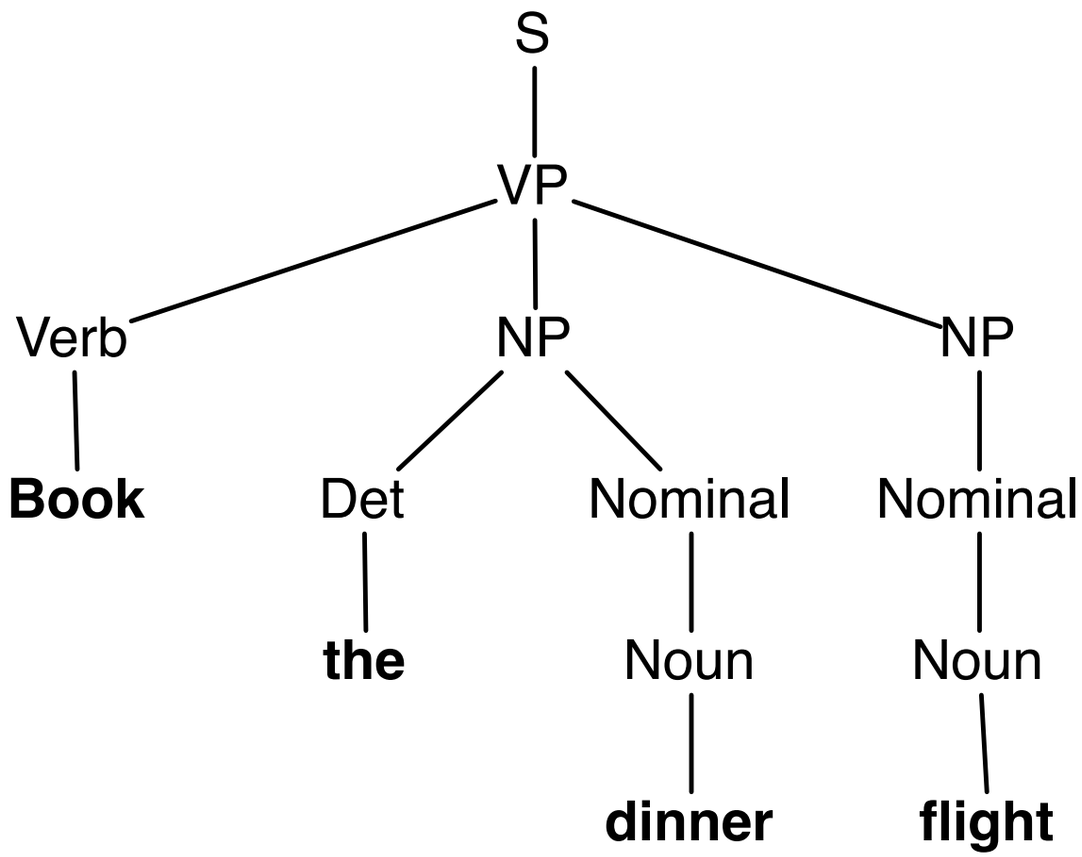
$$\hat{T}(S) = \arg\max_T \frac{P(T,S)}{P(S)}$$

$$\hat{T}(S) = \arg\max_T P(T,S)$$

$$\hat{T}(S) = \arg\max_T P(T)$$

Разрешение многозначности





P(T-left) = .05*.20*.20*.20*75*.30*.60*.10*.40=**2.2*10 -6**

P(T-right) = .05*.10*.20*.15*.75*.75*.30*.60*.10*.40=**6.1*10 -7**

Моделирование языка

$$P(S) = \sum_T P(T, S) = \sum_T P(T)$$

- Вариант 1:
  - –Этап 1: с помощью n-граммной модели получить m лучших предложений
  - –Этап 2: выбрать наиболее вероятное предложение на основе грамматики
- Вариант 2:
  - –Модифицировать парсер для предсказания следующего слова (Xu et. al 2002)

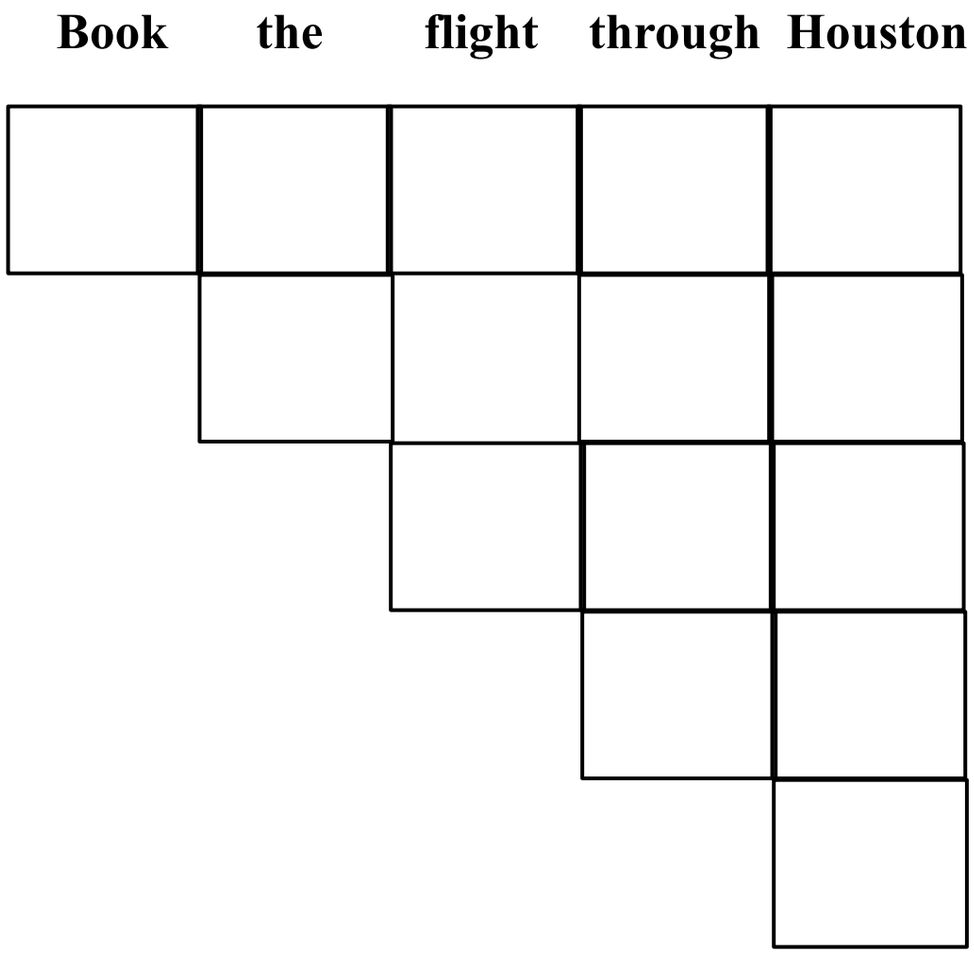
Вероятностная версия алгоритма CKY

- Добавляем в каждую ячейку вероятность нетерминального символа
- Ячейка [i,j] должна содержать наиболее вероятный вывод, покрывающий с i+1 по j слова и содержать их вероятность.
- При трансформации грамматики к нормальной форме необходимо сохранить вероятности правил

Преобразование грамматики

Оригинальная грамматика		Грамматика в нормальной форме Хомского	
S → NP VP	0.8	**S → NP VP**	0.8
S → Aux NP VP	0.1	**S → X1 VP**	0.1
		X1 → Aux NP	1.0
S → VP	0.1	**S → book | include | prefer**	
		0.01 0.004 0.006	
		S → Verb NP	0.05
		S → VP PP	0.03
NP → Pronoun	0.2	**NP → I | he | she | me**	
		0.1 0.02 0.02 0.06	
NP → Proper-Noun	0.2	**NP → Houston | NWA**	
		0.16 .04	
NP → Det Nominal	0.6	**NP → Det Nominal**	0.6
Nominal → Noun	0.3	**Nominal → book | flight | meal | money**	
		0.03 0.15 0.06 0.06	
Nominal → Nominal Noun	0.2	**Nominal → Nominal Noun**	0.2
Nominal → Nominal PP	0.5	**Nominal → Nominal PP**	0.5
VP → Verb	0.2	**VP → book | include | prefer**	
		0.1 0.04 0.06	
VP → Verb NP	0.5	**VP → Verb NP**	0.5
VP → VP PP	0.3	**VP → VP PP**	0.3
PP → Prep NP	1.0	**PP → Prep NP**	1.0

Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY

	Book	the	flight	through	Houston
	S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1				

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1				
	Det:.6			

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None			
	Det:.6			

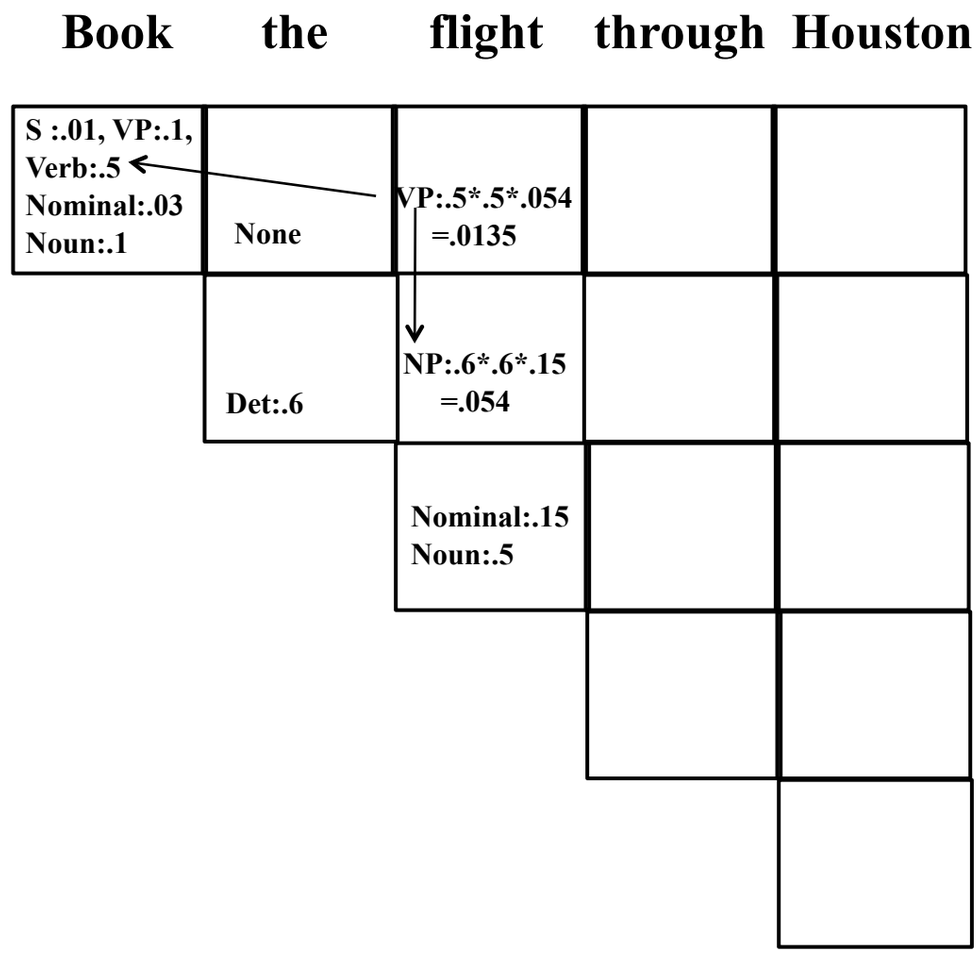
Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None			
	Det:.6			
		Nominal:.15 Noun:.5		

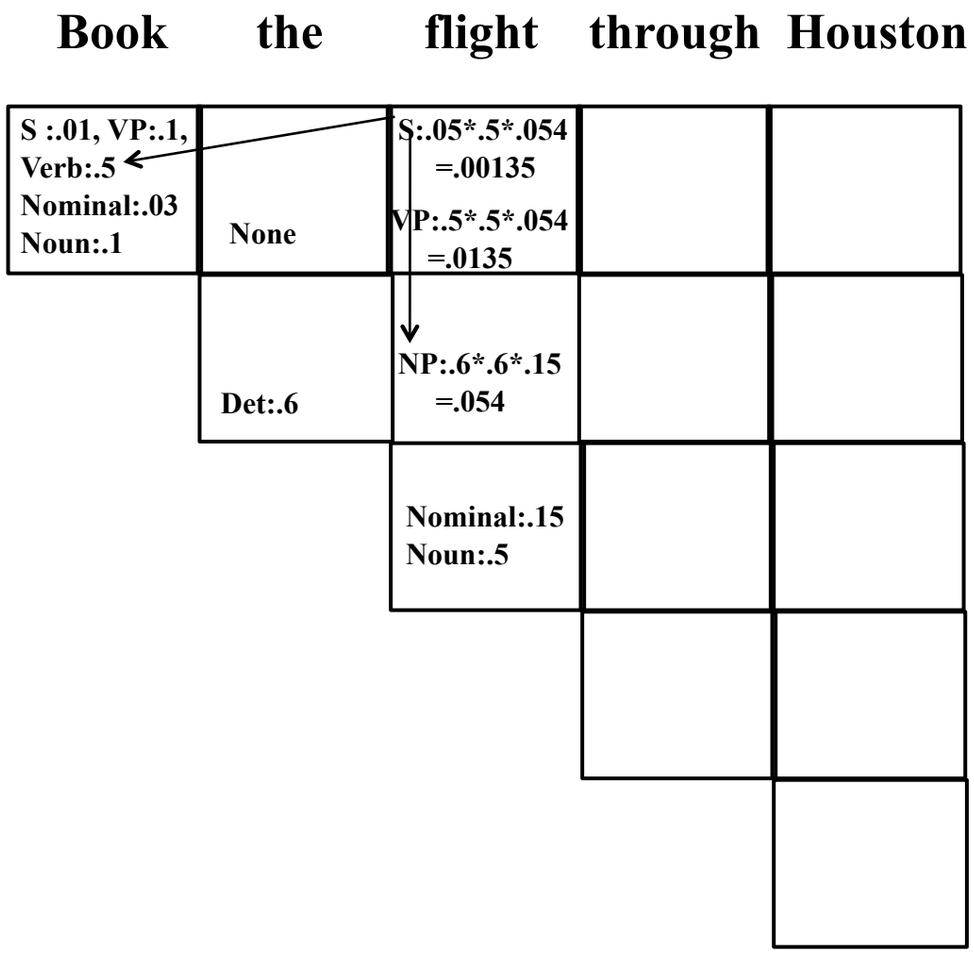
Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None			
	Det:.6	NP:.6*.6*.15 =.054		
		Nominal:.15 Noun:.5		

Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135		
	Det:.6	NP:.6*.6*.15 =.054		
		Nominal:.15 Noun:.5		
			Prep:.2	

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135		
	Det:.6	NP:.6*.6*.15 =.054		
		Nominal:.15 Noun:.5	None	
			Prep:.2	

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135		
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	
			Prep:.2	

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	
			Prep:.2	

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	
			Prep:.2	
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	Nominal: .5*.15*.032 =.0024
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	NP:.6*.6* .0024 =.000864
		Nominal:.15 Noun:.5	None	Nominal: .5*.15*.032 =.0024
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY

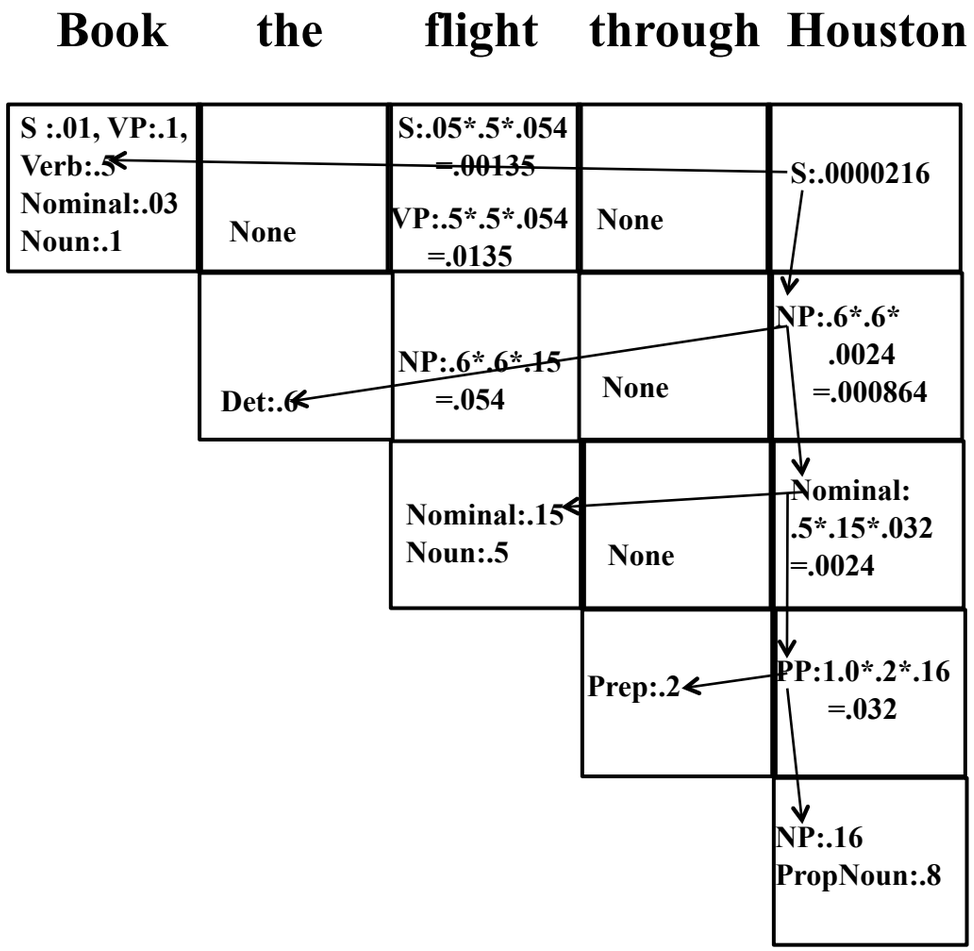
Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	S:.05*.5* .000864 =.0000216
	Det:.6	NP:.6*.6*.15 =.054	None	NP:.6*.6* .0024 =.000864
		Nominal:.15 Noun:.5	None	Nominal: .5*.15*.032 =.0024
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	S:.03*.0135* .032 =.00001296 S:.0000216
	Det:.6	NP:.6*.6*.15 =.054	None	NP:.6*.6* .0024 =.000864
		Nominal:.15 Noun:.5	None	Nominal: .5*.15*.032 =.0024
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY

Выбираем наиболее вероятное дерево разбора



Обучение СКС

- Вычисление вероятности на основе банка деревьев

$$P(\alpha \rightarrow \beta|\alpha) = \frac{Count(\alpha \rightarrow \beta)}{\sum_{\gamma} Count(\alpha \rightarrow \beta)} = \frac{Count(\alpha \rightarrow \beta)}{Count(\alpha)}$$

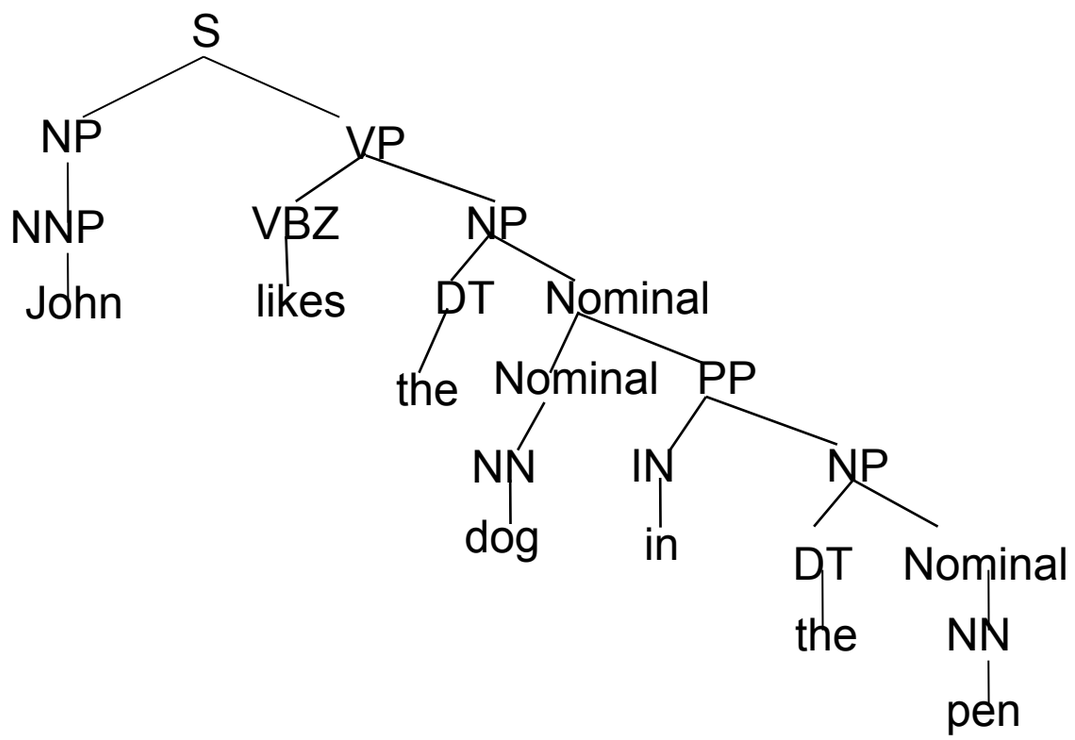
- Вывод без тренировочного множества (EM)
  - На основе множества предложений построить множество наиболее вероятных синтаксических разборов
  - Обновить значения вероятностей на осонве полученных данных
  - (Manning and Schutze 1999)

Проблемы СКС

- Предположение о независимости правил, не позволяет хорошо моделировать структурные зависимости в дереве разбора
- СКС не могут моделировать синтаксические факты о конкретных словах

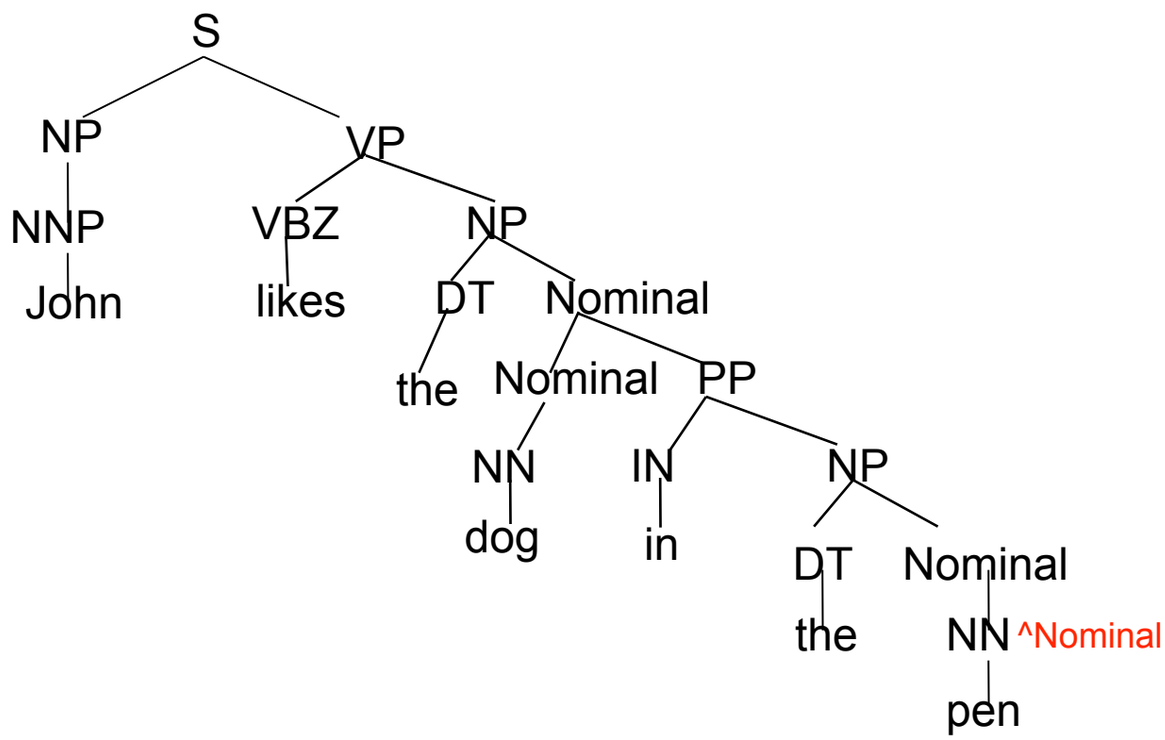
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



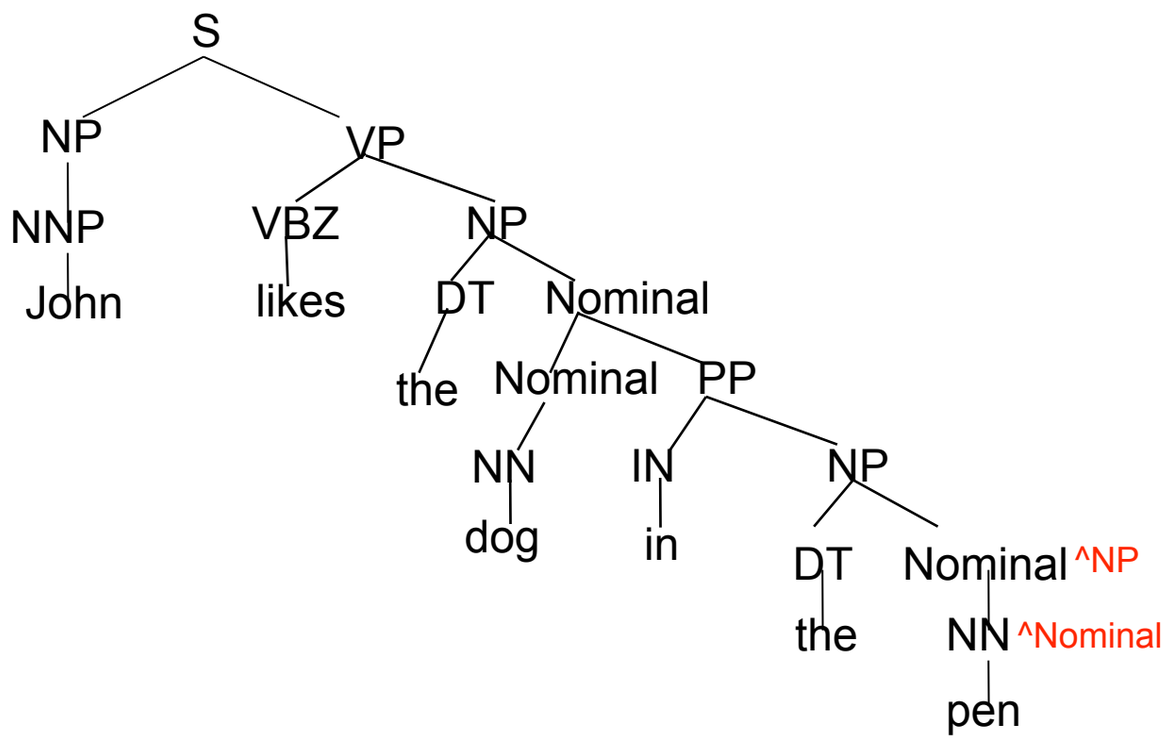
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



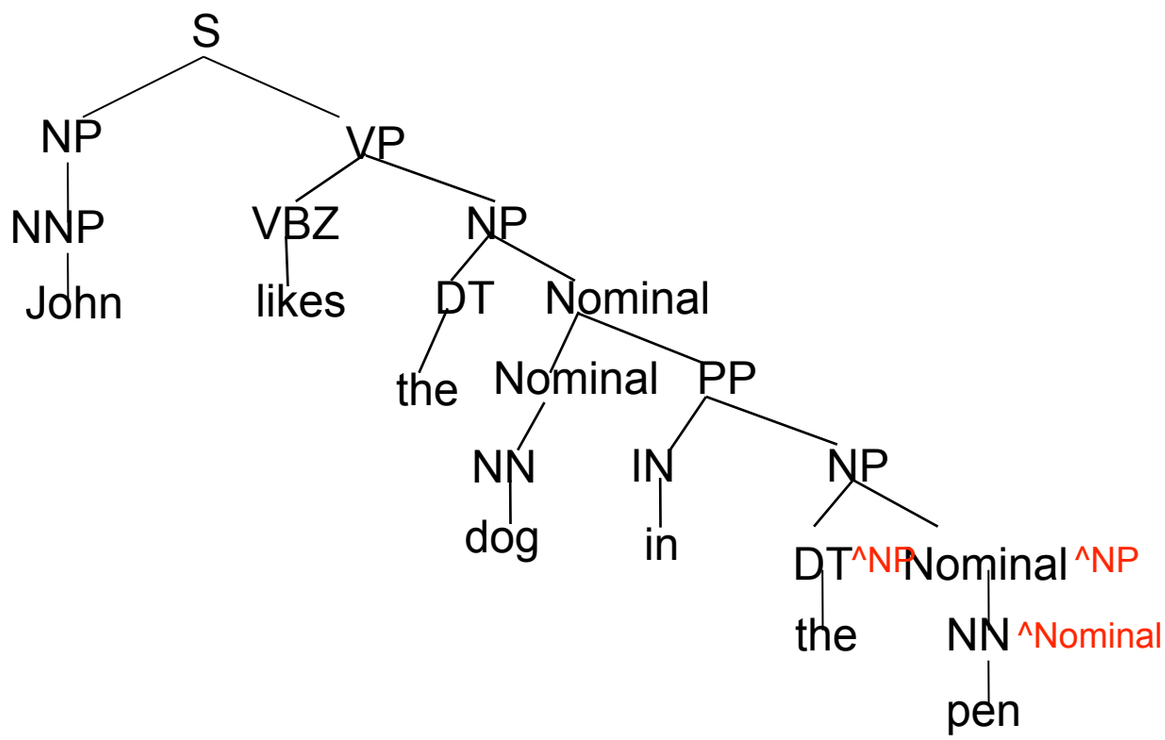
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



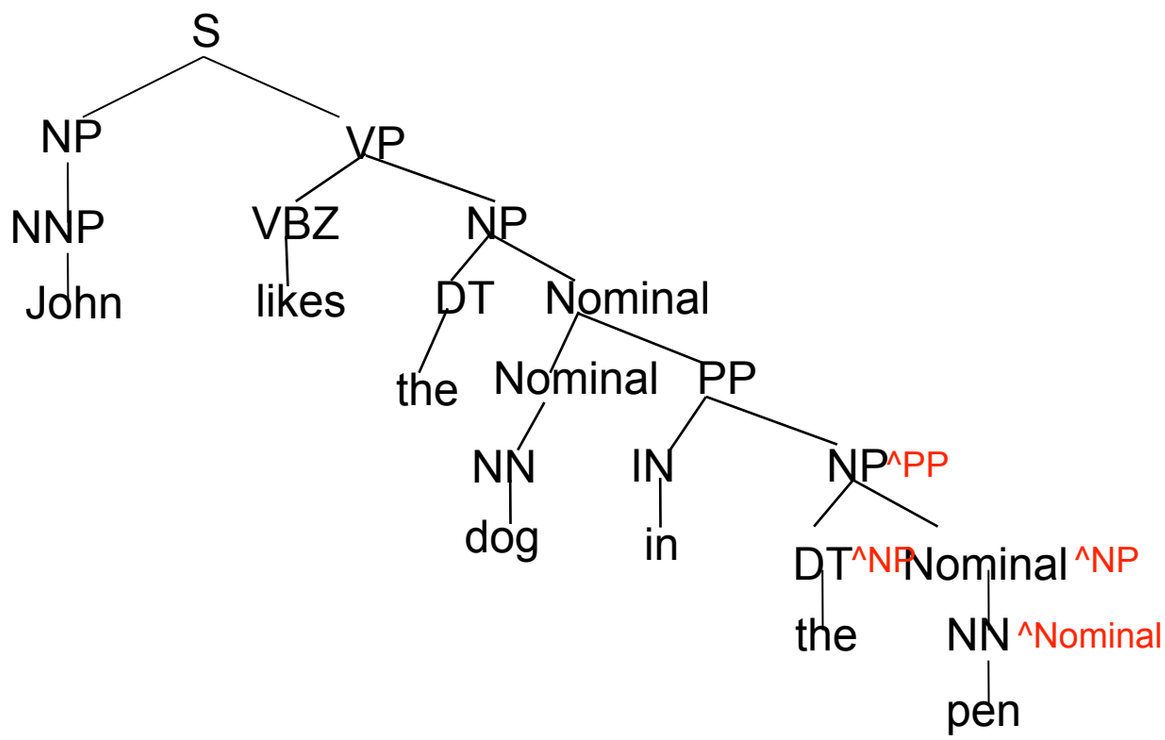
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



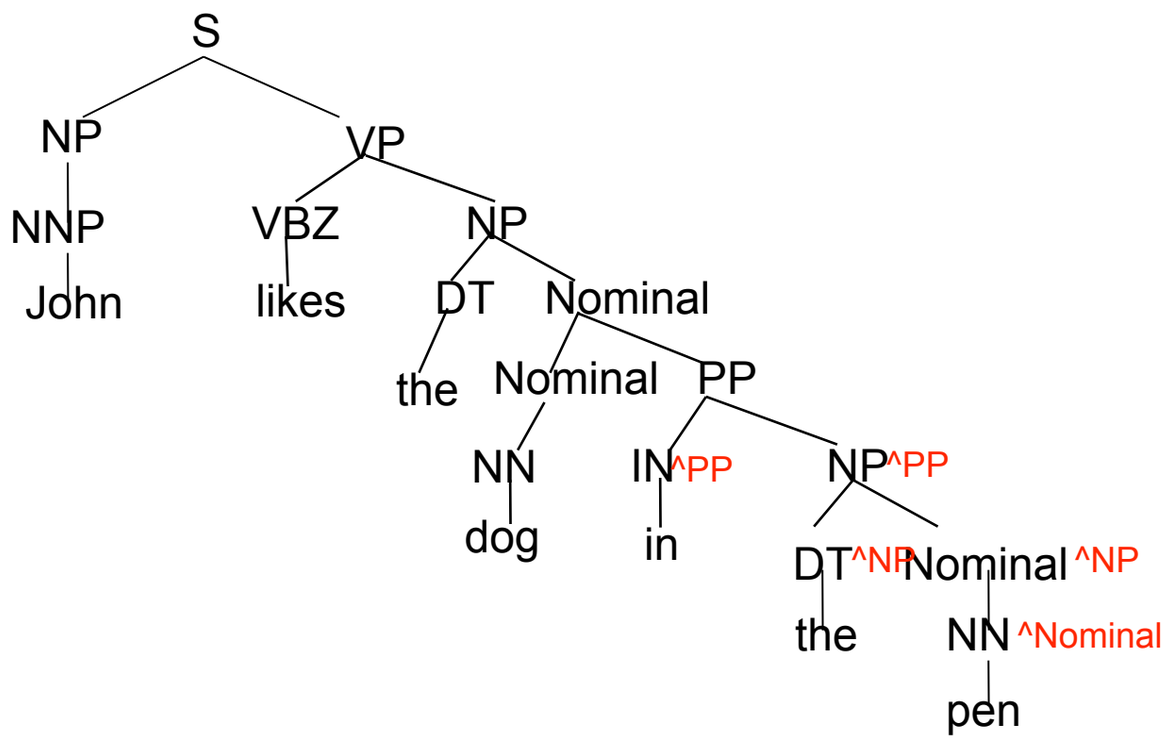
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



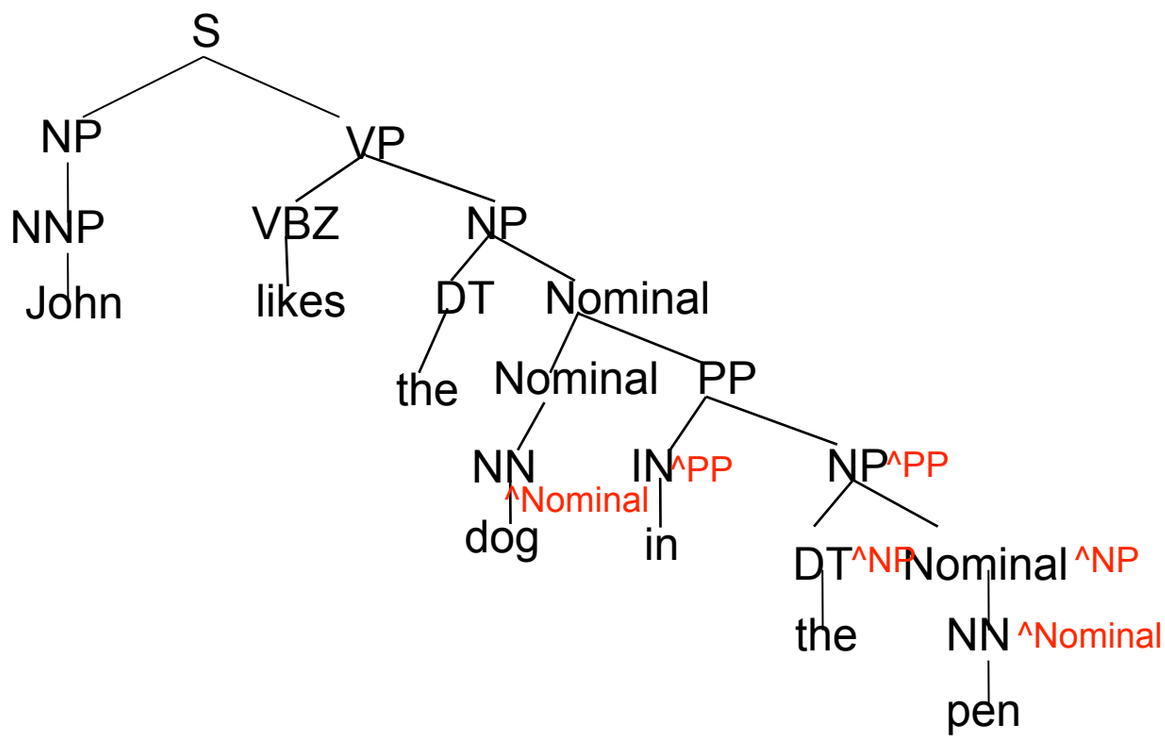
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



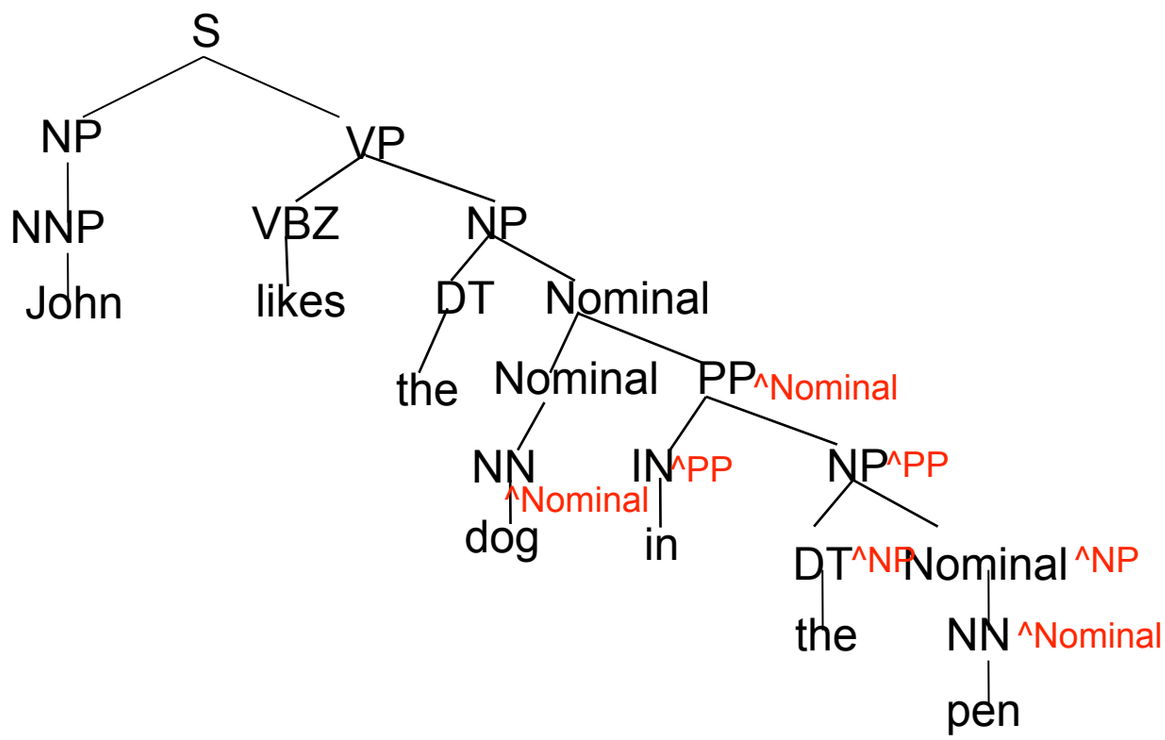
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



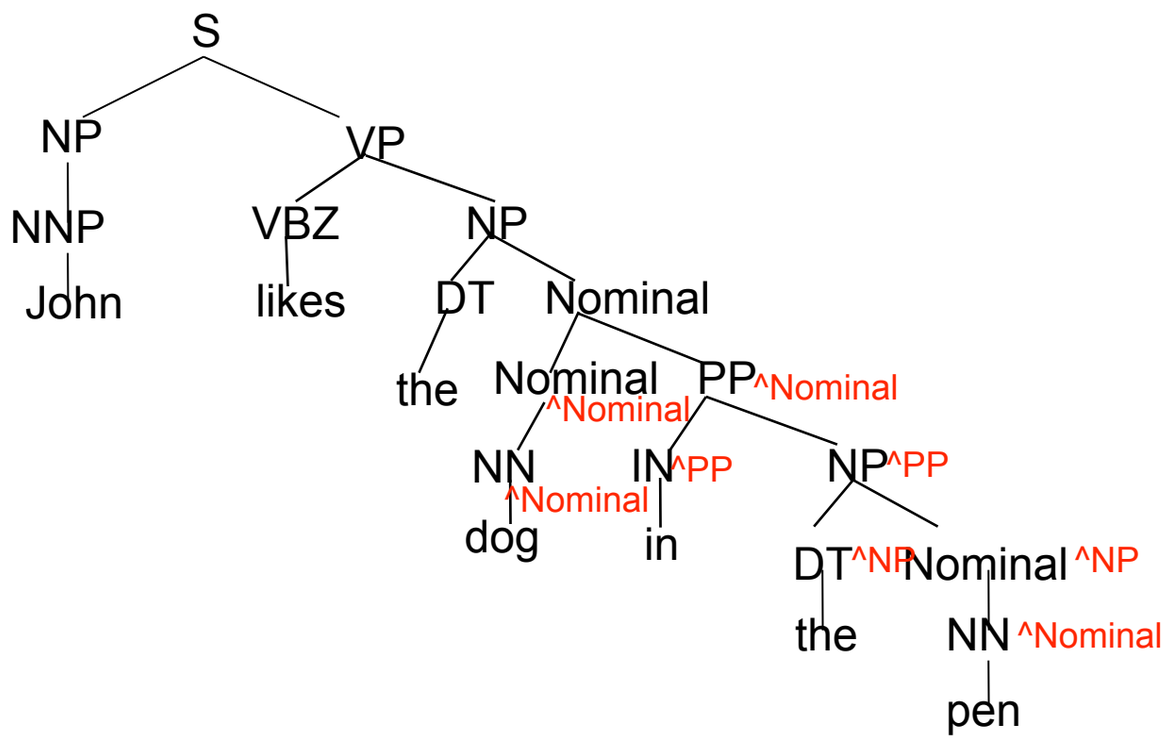
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



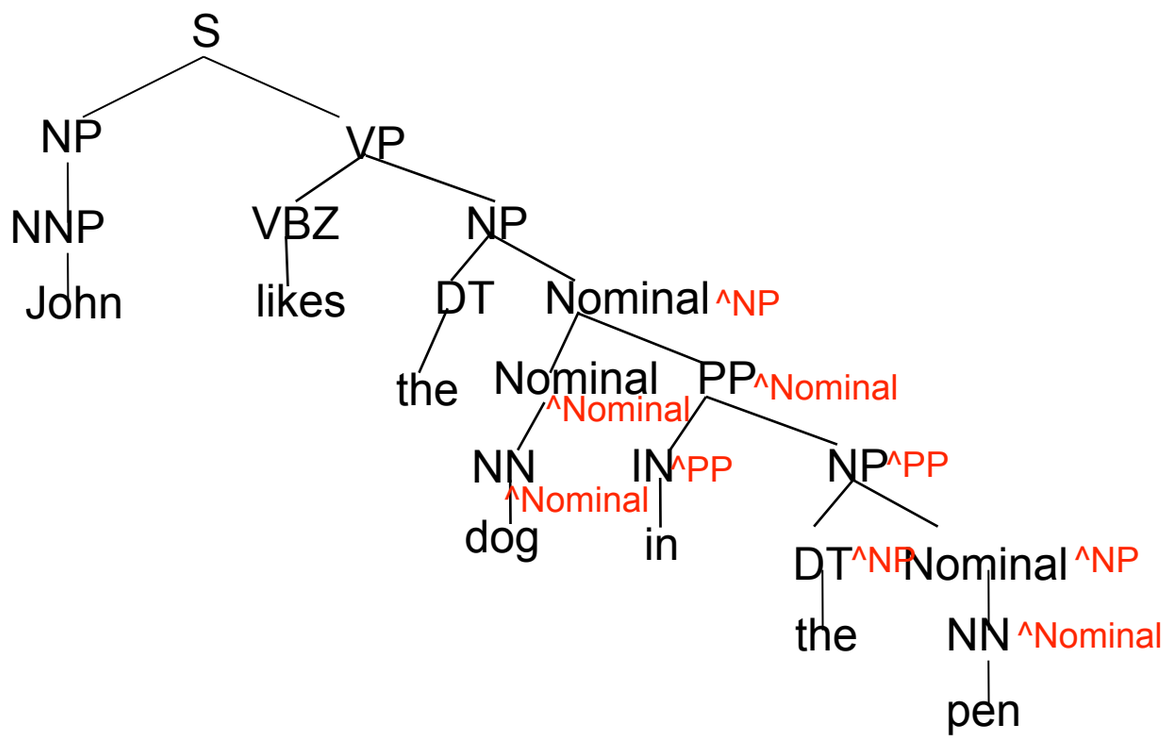
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



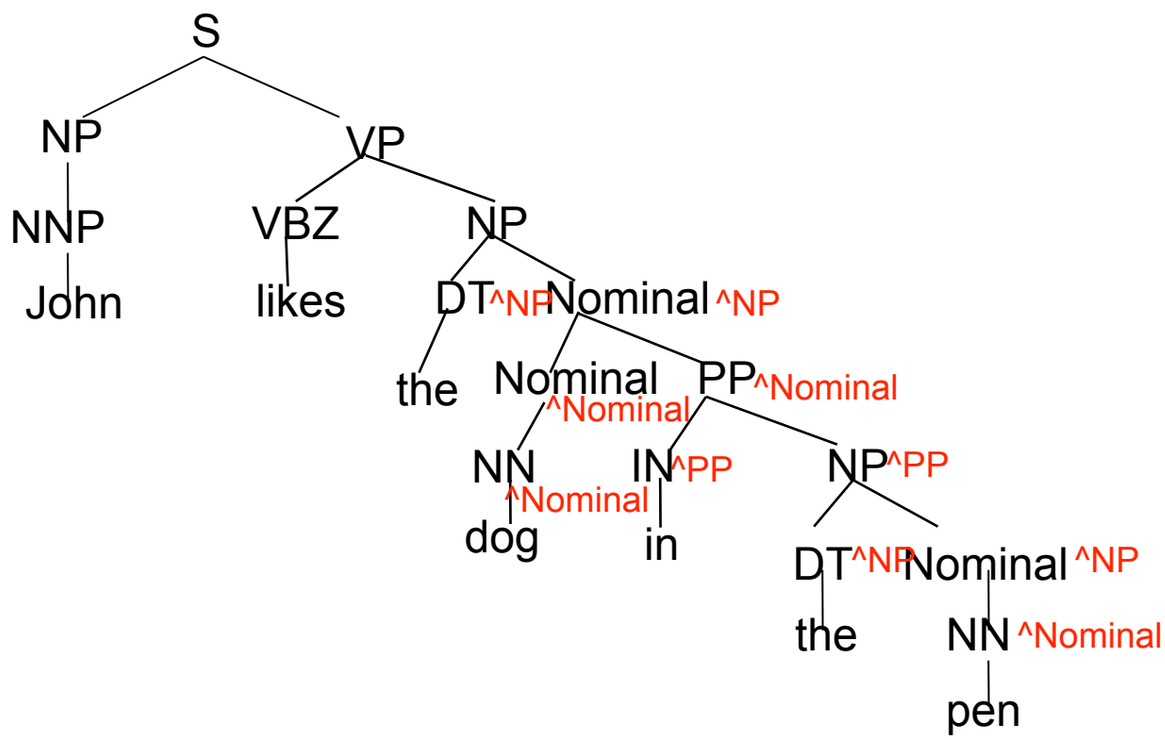
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



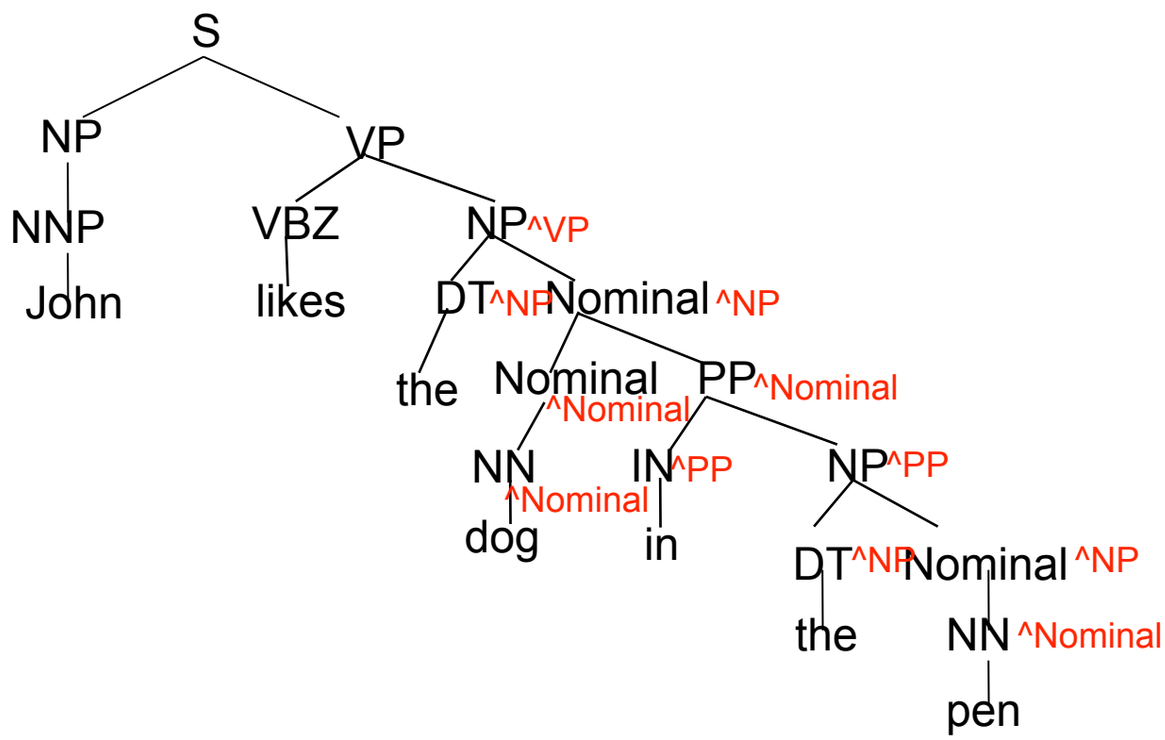
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



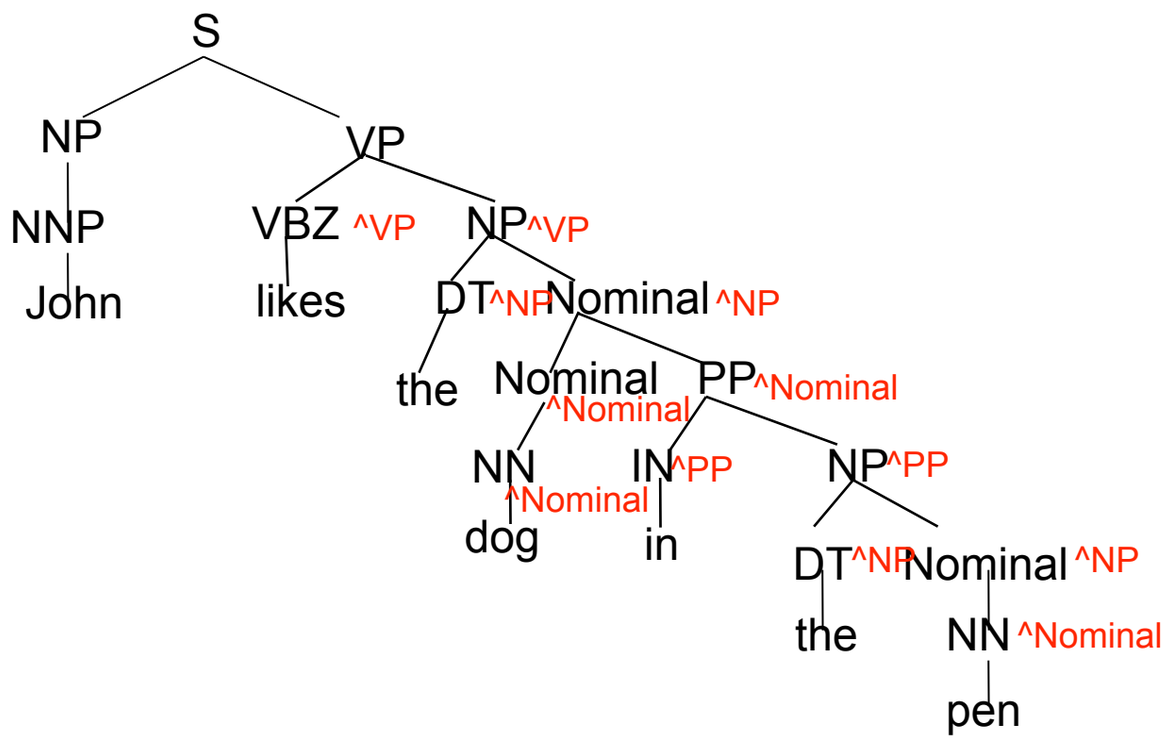
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



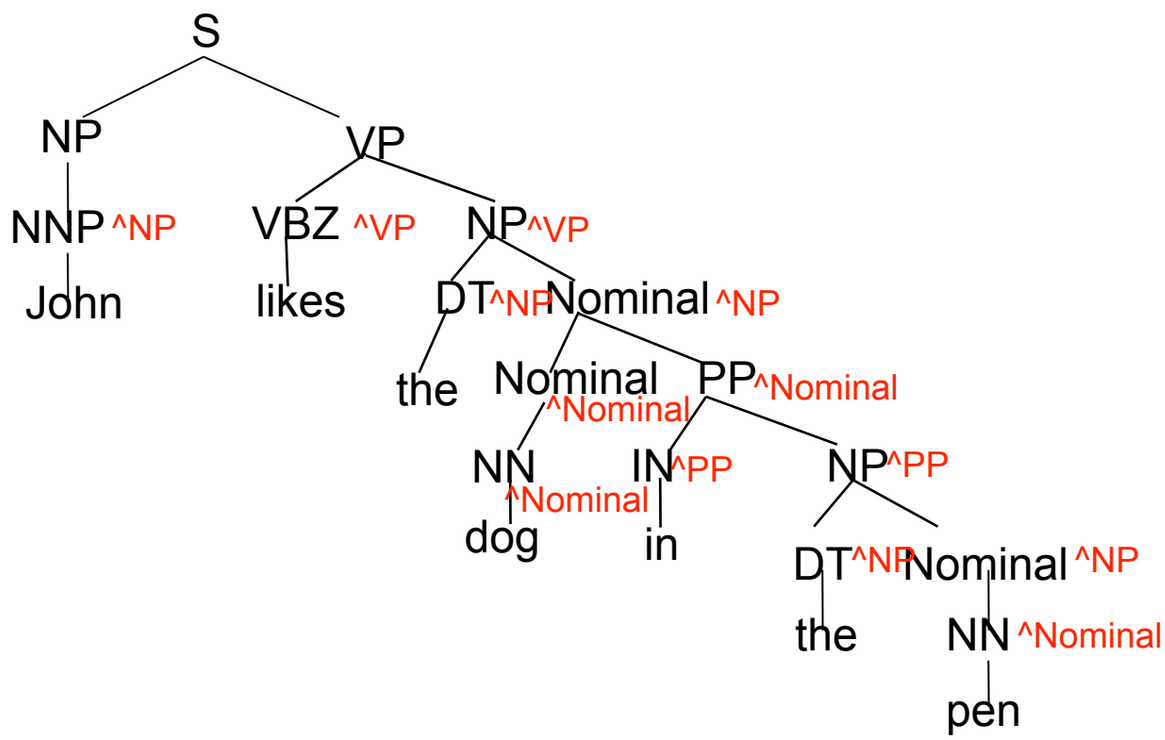
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



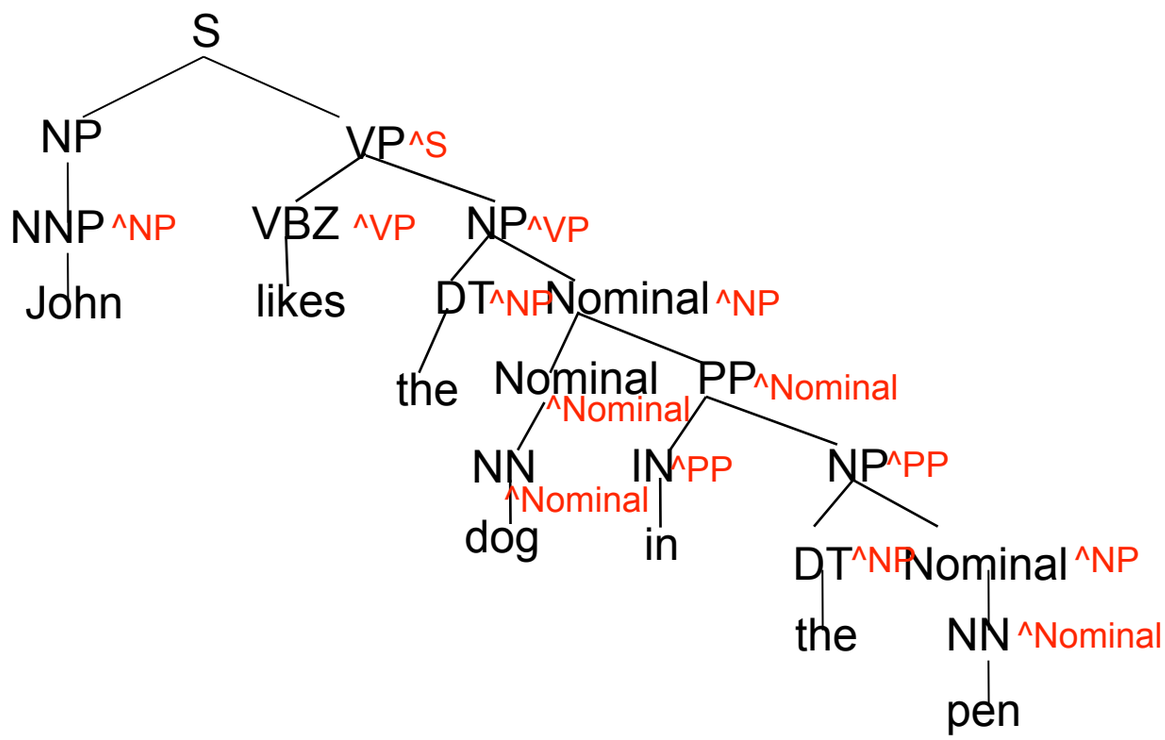
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



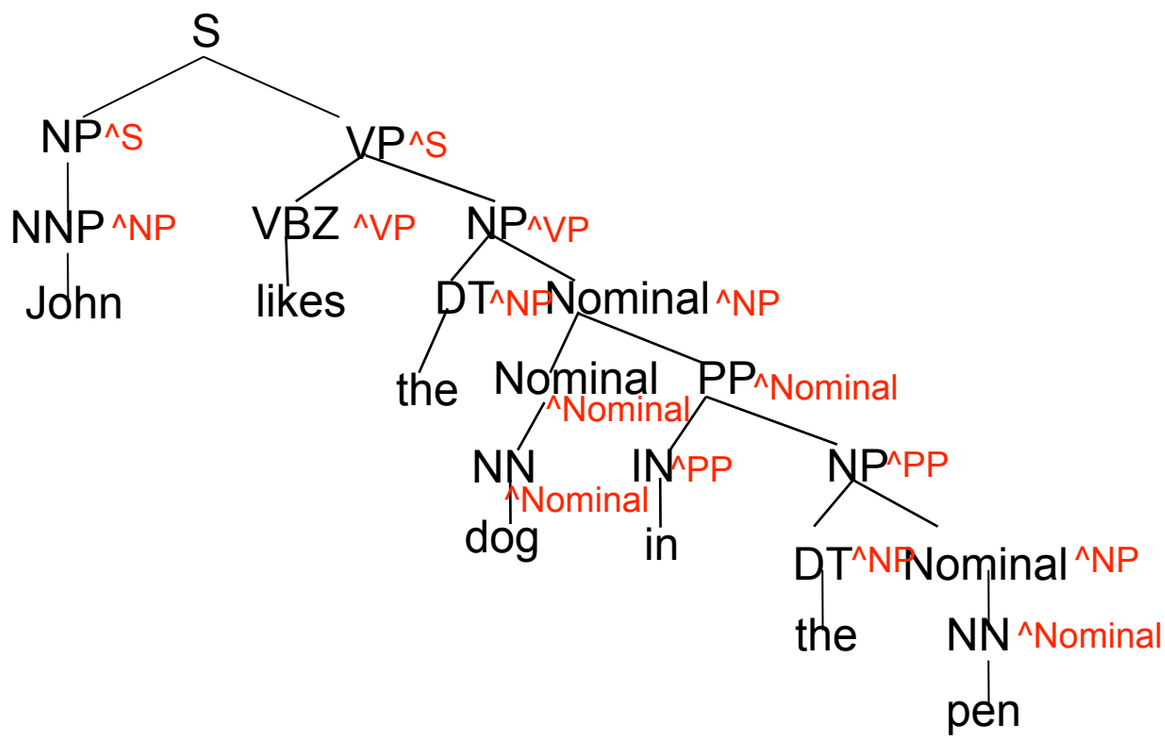
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



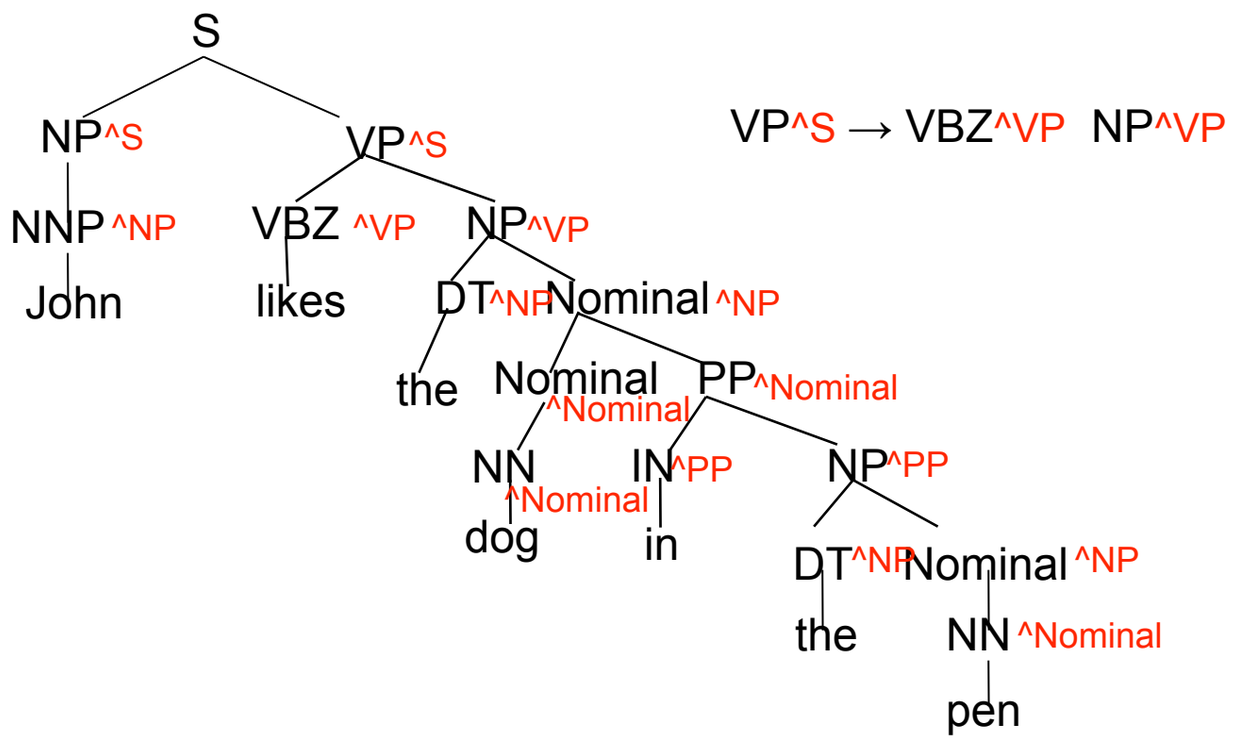
Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)



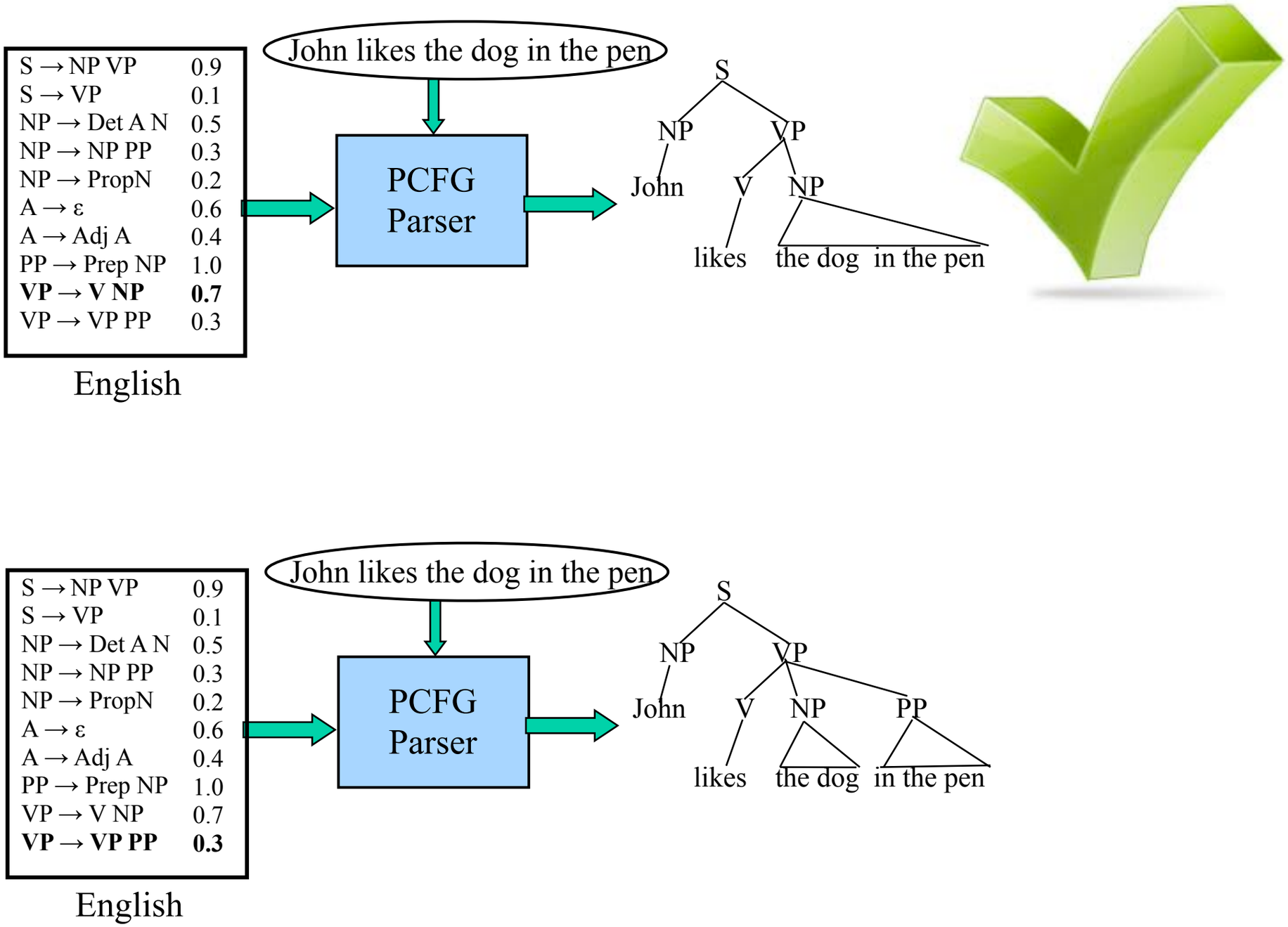
Разделение и слияние

- Разделение нетерминальных символов сильно увеличивает грамматику
- Лучше разделять нетерминалы, только если это приведет к улучшению точности
- Также можно объединять некоторые нетерминальные символы, чтобы достичь болшей точности
- Метод: эвристический поиск наилучшей комбинации разделений и слияний которая будет максимизировать правдоподобие банка деревьев

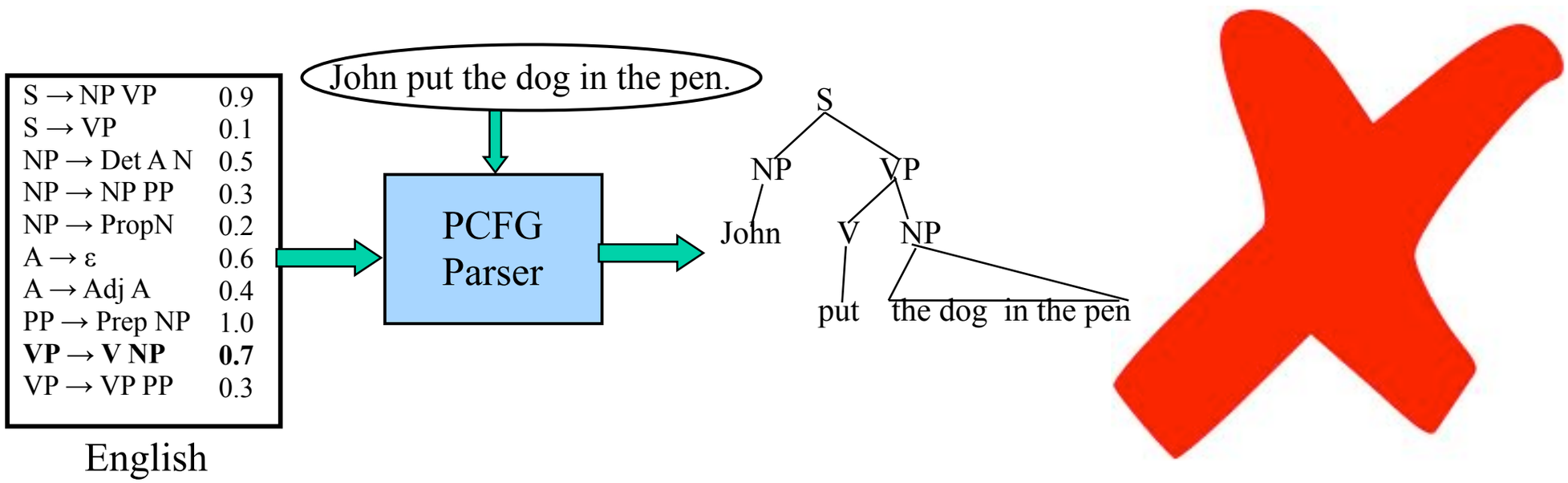
Проблемы СКС

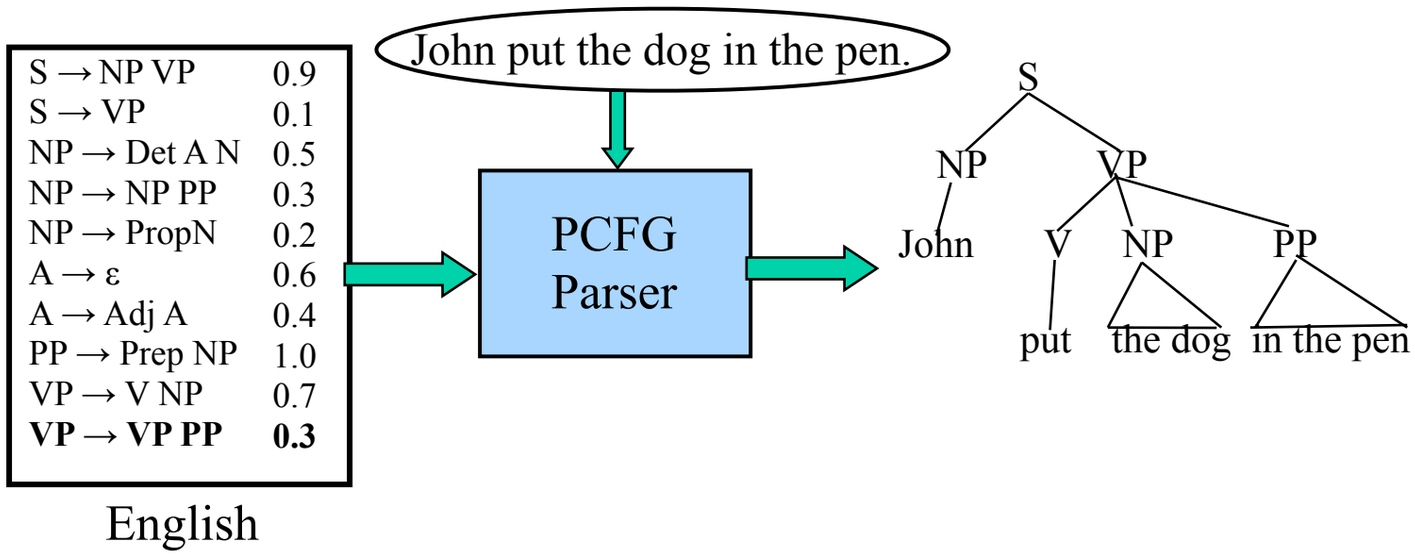
- Предположение о независимости правил, не позволяет хорошо моделировать структурные зависимости в дереве разбора
- СКС не могут моделировать синтаксические факты о конкретных словах

Пример


S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John likes the dog in the pen
PCFG Parser
S
NP VP
John V NP
likes the dog in the pen
S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John likes the dog in the pen
PCFG Parser
S
NP VP
John V NP PP
likes the dog in the pen

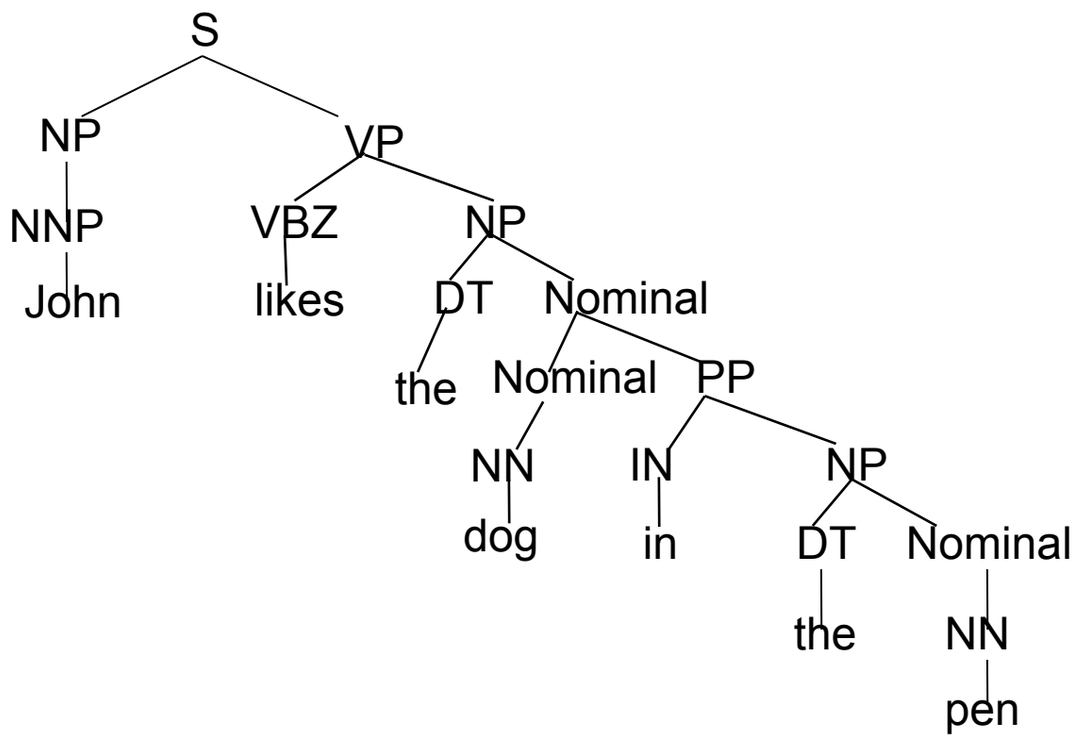
Пример


S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John put the dog in the pen.
PCFG Parser
S
NP
VP
John
V
NP
put
the dog
in the pen


S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John put the dog in the pen.
PCFG Parser
S
NP
VP
John
V
NP
PP
put
the dog
in the pen

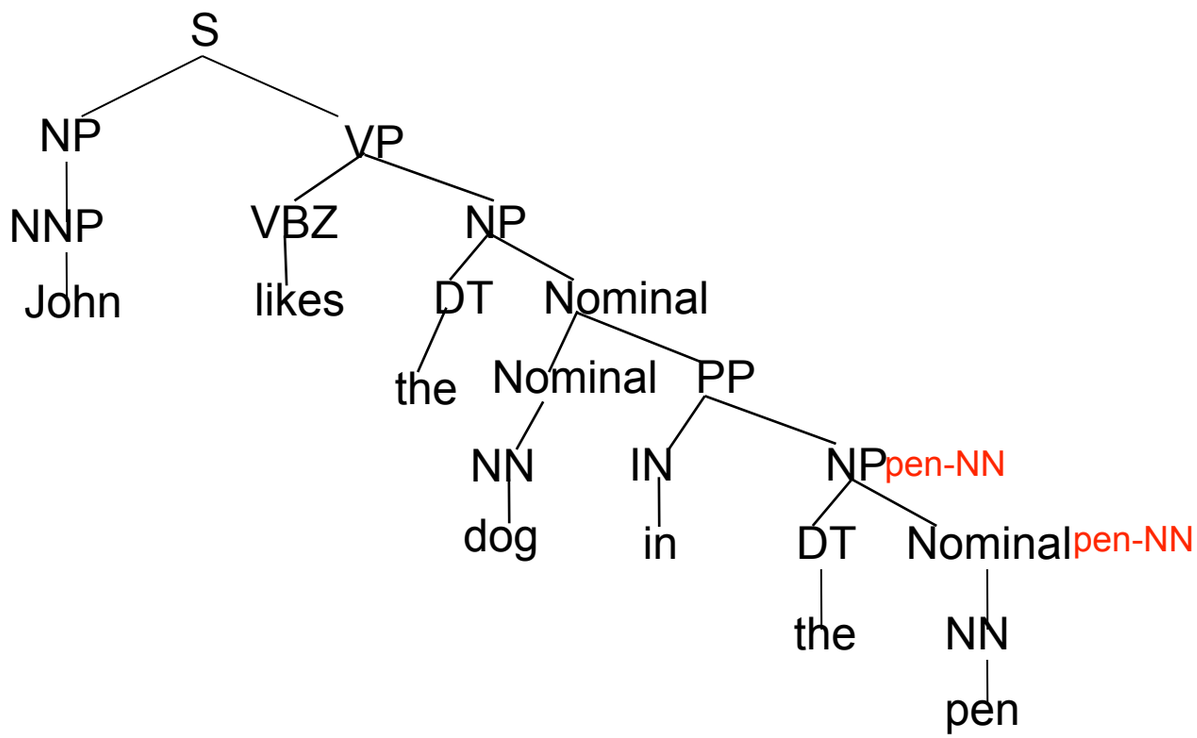
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



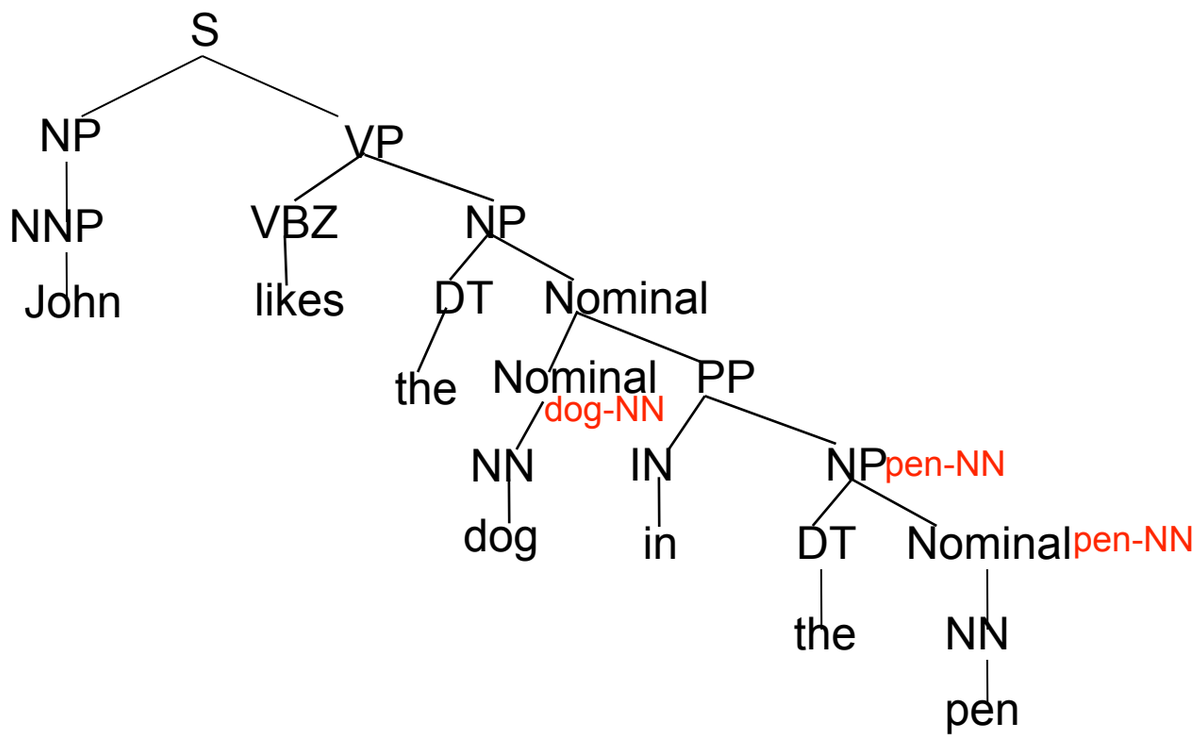
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



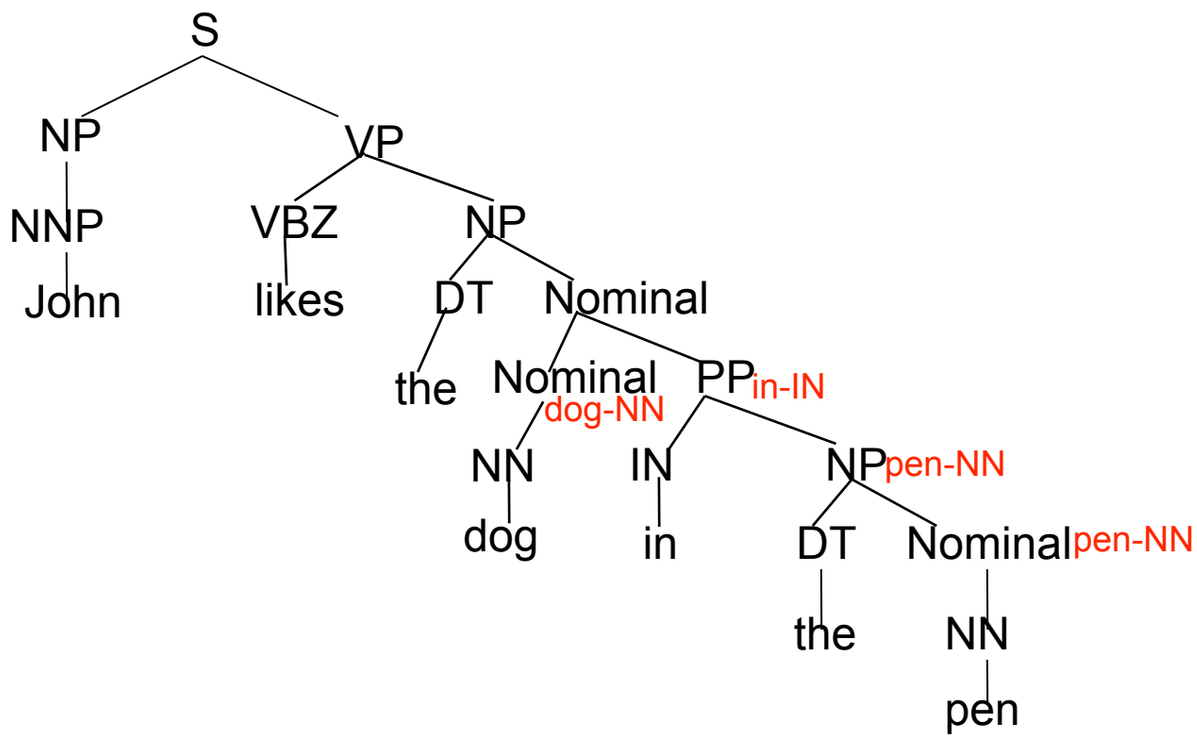
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



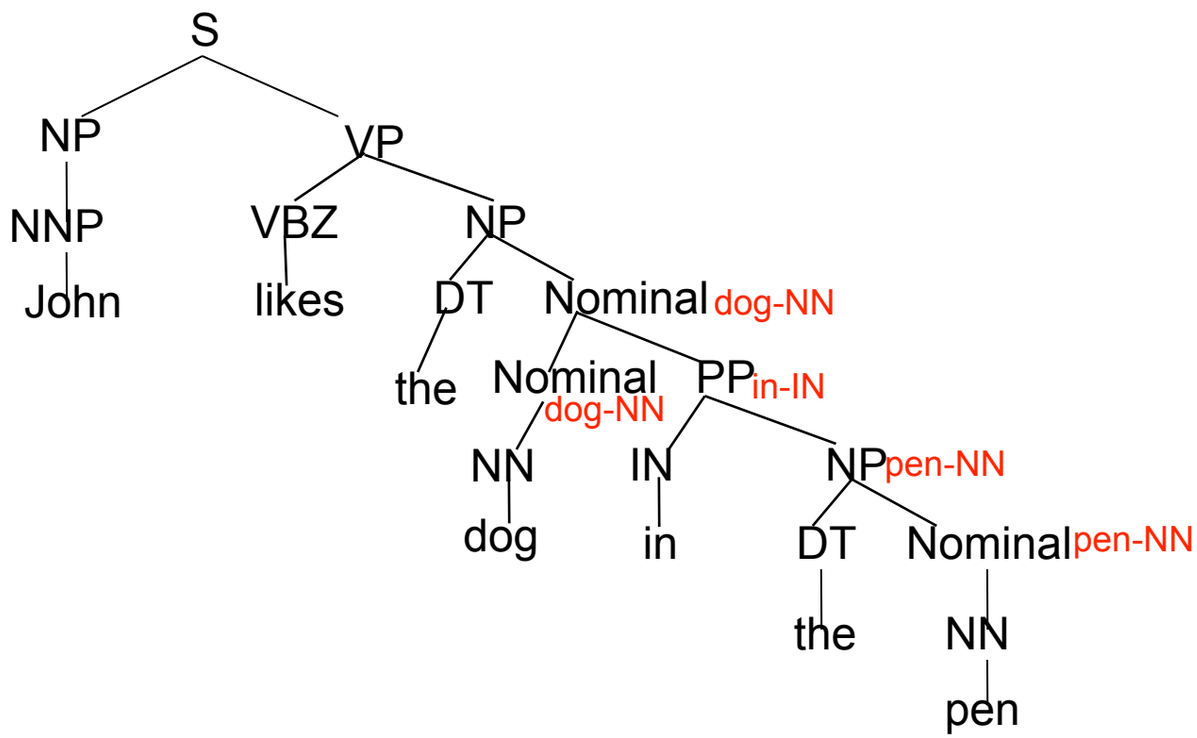
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



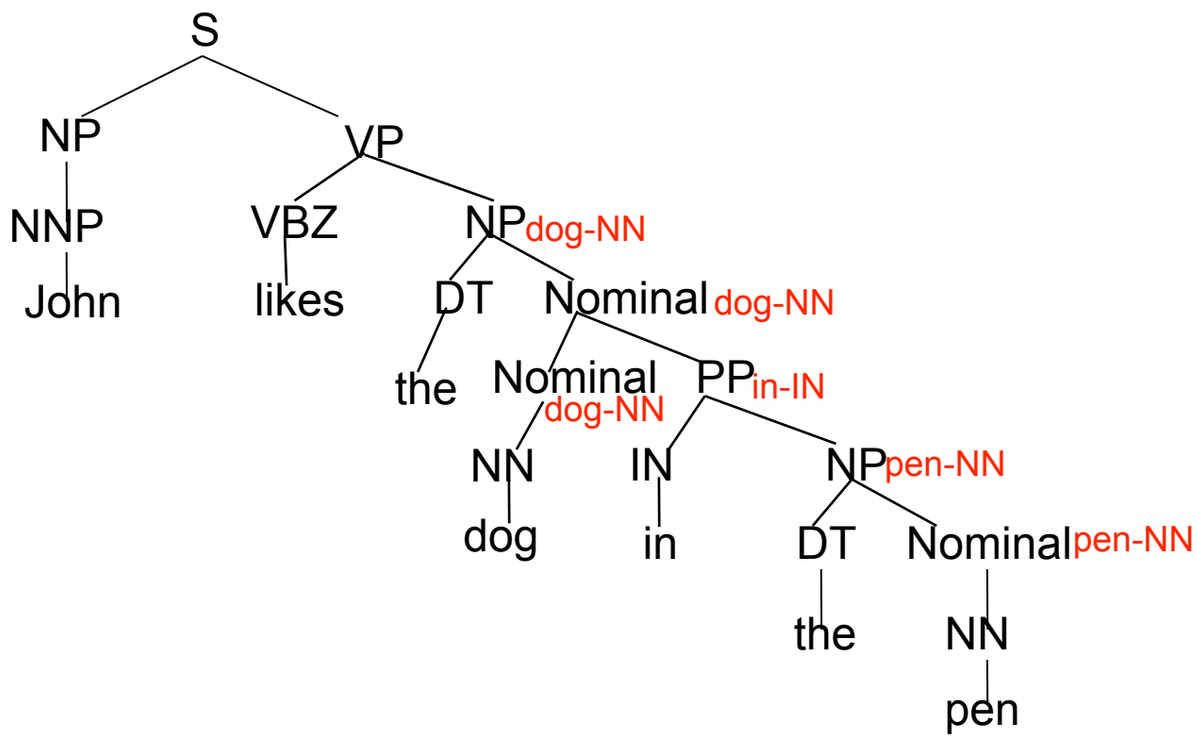
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



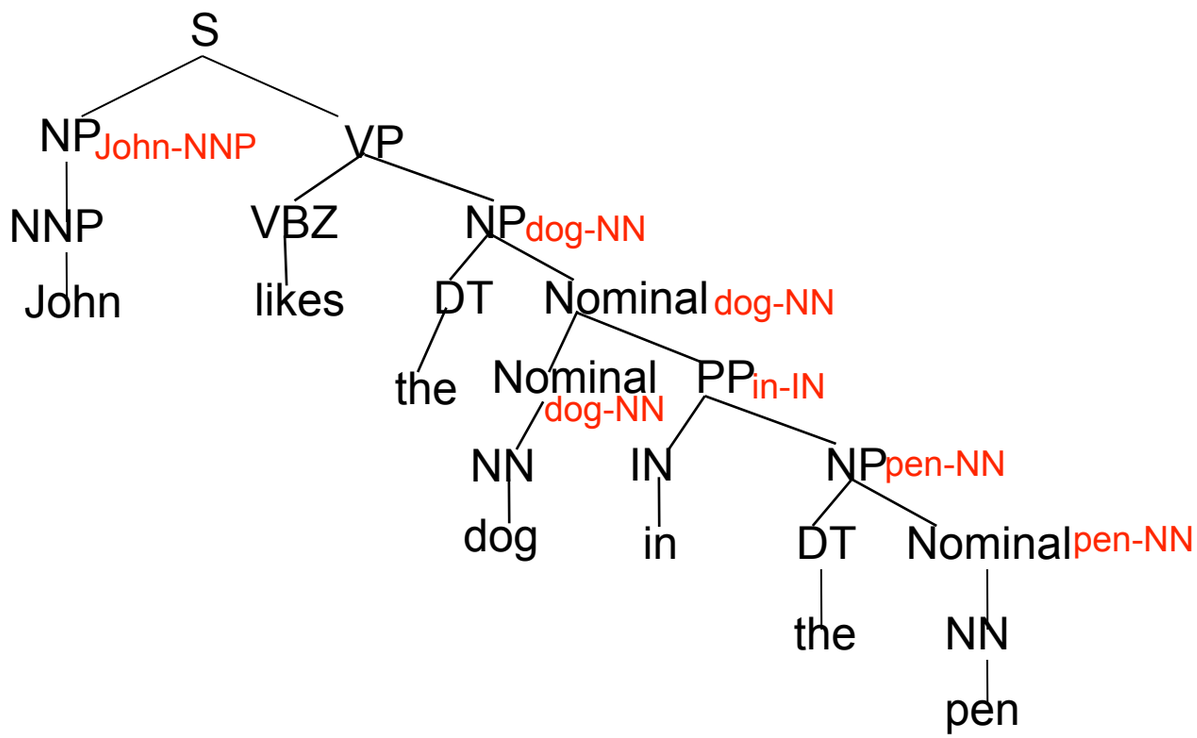
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



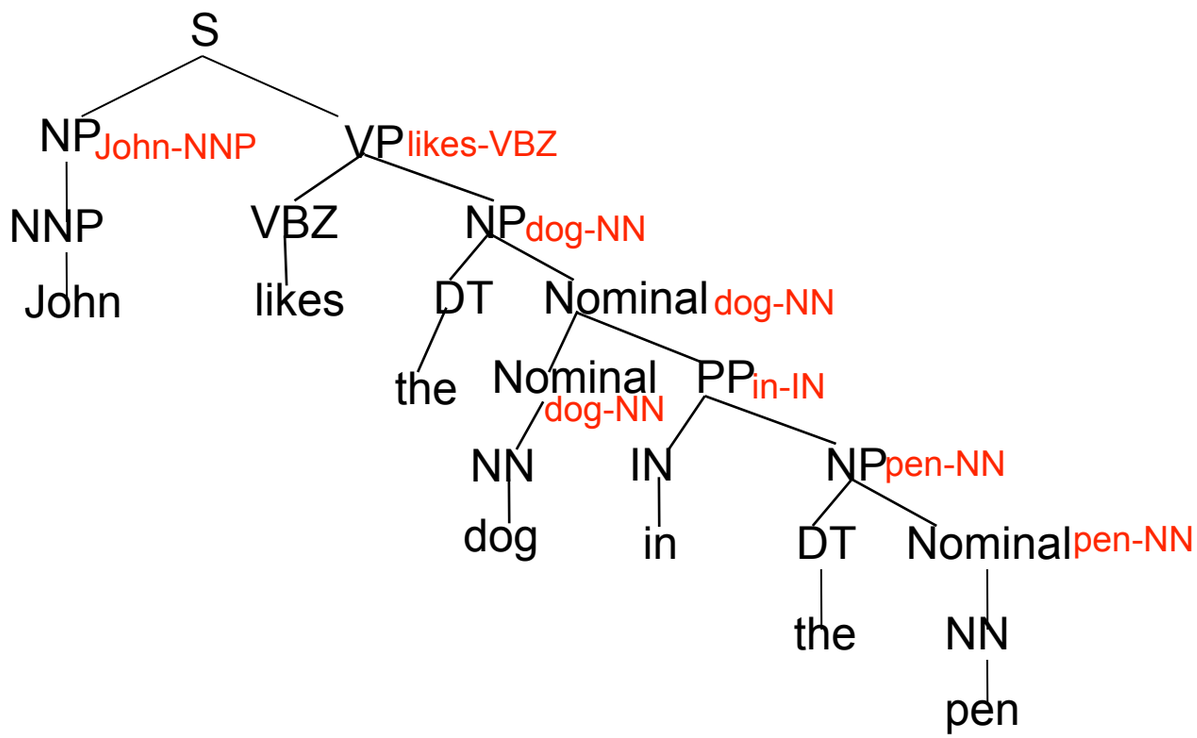
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



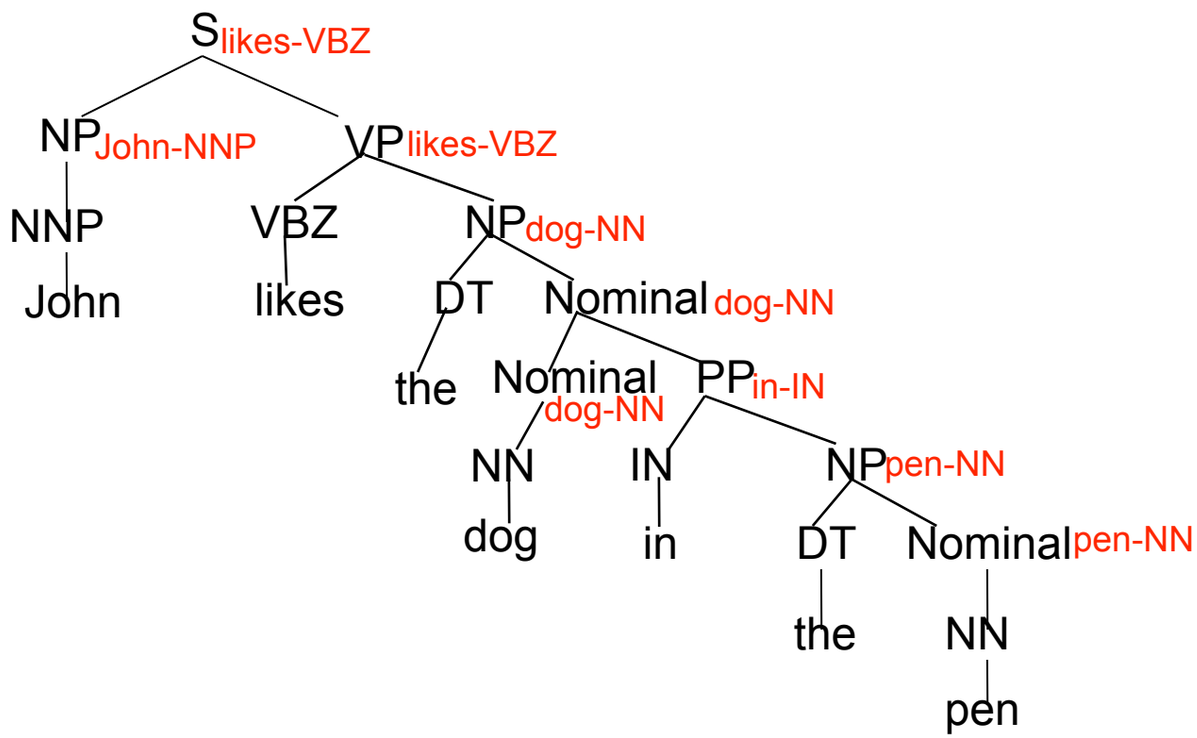
СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



Оценка вероятности

- Точная оценка невозможна, так как не существует достаточного количества деревьев
- Необходимо сделать еще предположения (о независимости), которые помогут оценить эти вероятности
- Алгоритмы (Collins, 1999), (Charniak, 1997)

Алгоритм Коллинза

- Использует простую производящую модель
- LHS → $L_n L_{n-1} \ldots L_1 H\ R_1 \ldots R_{m-1} R_m$
  - H-вершина группы
  - L - символы слева
  - R - символы справа
  - По краям символы STOP
- Вероятности левых и правых символов зависят только от вершины группы и нетерминала в левой части правила

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP

L_1 H R_1 R_2 R_3

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP
L_1 H R_1 R_2 R_3

VPput-VBD

VBDput-VBD

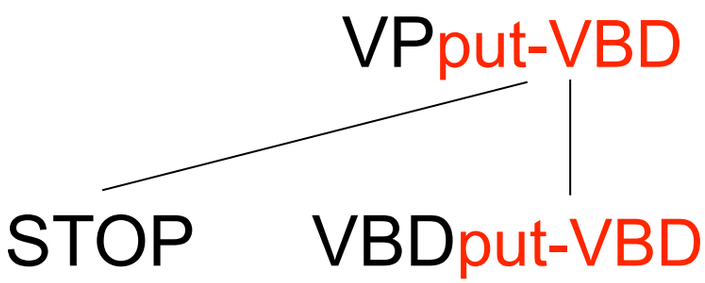
P(VPput-VBD → VBDput-VBD NPdog-NN PPin-IN) =

= P_H(VBDput-VBD| VPput-VBD) *

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP
L_1 H R_1 R_2 R_3



P(VPput-VBD → VBDput-VBD NPdog-NN PPin-IN) =

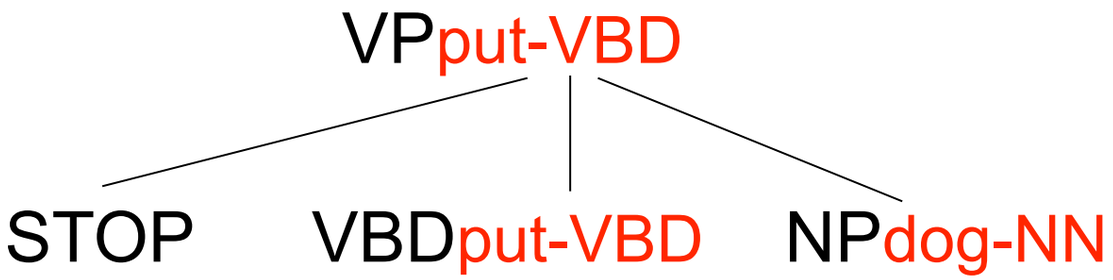
= P_H(VBDput-VBD| VPput-VBD) *

* P_L(STOP | VBDput-VBD, VPput-VBD) *

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP
L_1 H R_1 R_2 R_3



P(VPput-VBD → VBDput-VBD NPdog-NN PPin-IN) =

$= P_H$(VBDput-VBD| VPput-VBD) *

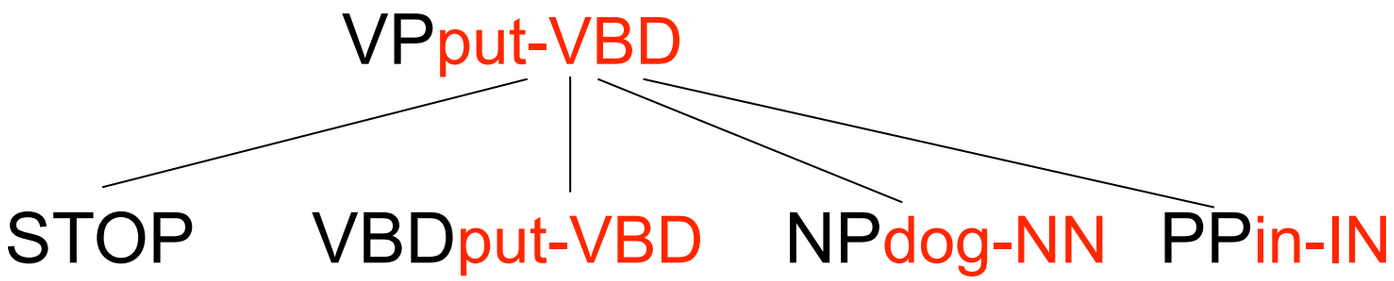
$* P_L$(STOP | VBDput-VBD, VPput-VBD) *

$* P_R$(NPdog-NN | VBDput-VBD, VPput-VBD) *

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP
L_1 H R_1 R_2 R_3



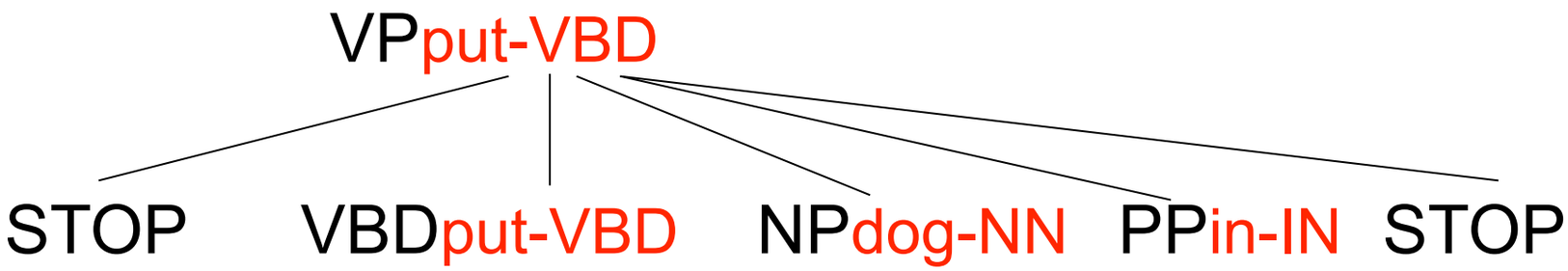
P(VPput-VBD → VBDput-VBD NPdog-NN PPin-IN) =

= P_H(VBDput-VBD| VPput-VBD) *
* P_L(STOP | VBDput-VBD, VPput-VBD) *
* P_R(NPdog-NN | VBDput-VBD, VPput-VBD) *
* P_R(PPin-IN | VBDput-VBD, VPput-VBD) *

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP
L_1 H R_1 R_2 R_3



P(VPput-VBD → VBDput-VBD NPdog-NN PPin-IN) =

= P_H(VBDput-VBD| VPput-VBD) *
* P_L(STOP | VBDput-VBD, VPput-VBD) *
* P_R(NPdog-NN | VBDput-VBD, VPput-VBD) *
* P_R(PPin-IN | VBDput-VBD, VPput-VBD) *
* P_R(STOP | VBDput-VBD, VPput-VBD)

Оценка вероятностей

- Вероятности можно оценить на основе банка деревьев

$$P_R(PP_{in\text{-}IN} \mid VP_{put\text{-}VBD}) = \frac{\text{Count}(PP_{in\text{-}IN} \text{ справа от вершины в правиле для } VP_{put\text{-}VBD})}{\text{Count}(\text{символы справа от вершины в правиле для } VP_{put\text{-}VBD})}$$

- Сглаживание можно осуществлять через откат или линейную интерполяцию

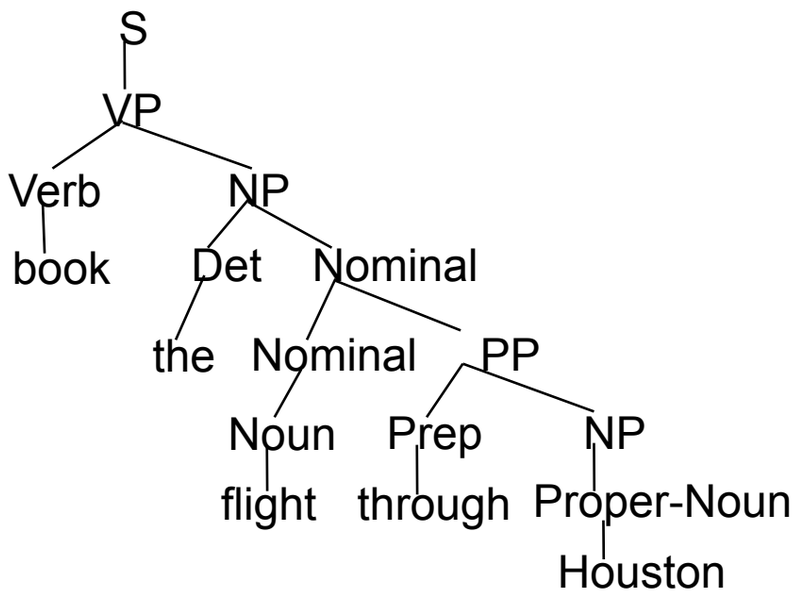
$$\begin{aligned} smP_R(PP_{in\text{-}IN} \mid VP_{put\text{-}VBD}) = {} & \lambda_1 P_R(PP_{in\text{-}IN} \mid VP_{put\text{-}VBD}) \\ & + (1-\lambda_1)\,(\lambda_2 P_R(PP_{in\text{-}IN} \mid VP_{VBD}) + \\ & (1-\lambda_2) P_R(PP_{in\text{-}IN} \mid VP)) \end{aligned}$$

Оценка качества алгоритма

- Метрика PARSEVAL: пусть P - дерево разбора, созданное алгоритмом, T - дерево разбора, созданное экспертами
  - Точность = (# правильных компонент в *P*) / (# компонент в *T*)
  - Полнота = (# правильных компонент в *P*) / (# компонент в *P*)
  - F-мера = 2PR / (P + R)
- Современные алгоритмы показывают точность и полноту более 90%

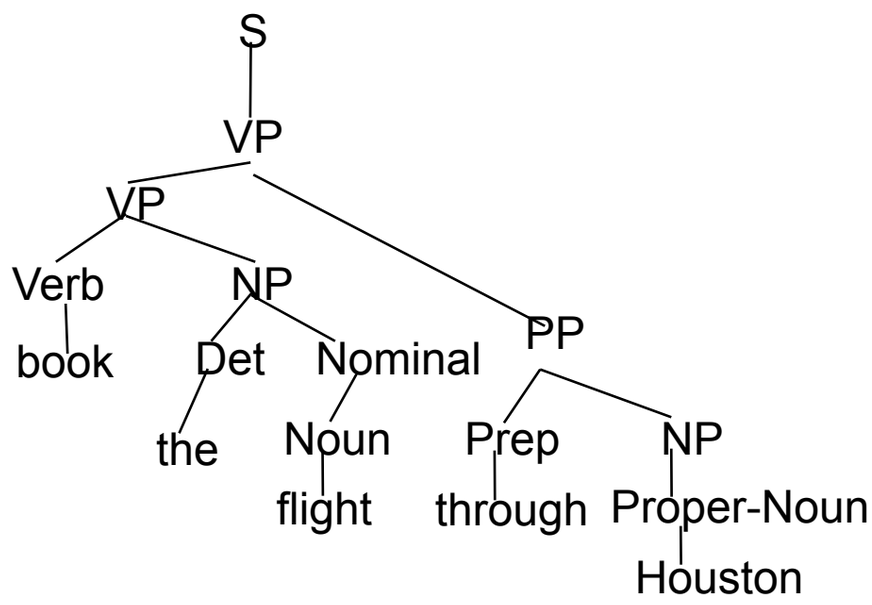
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

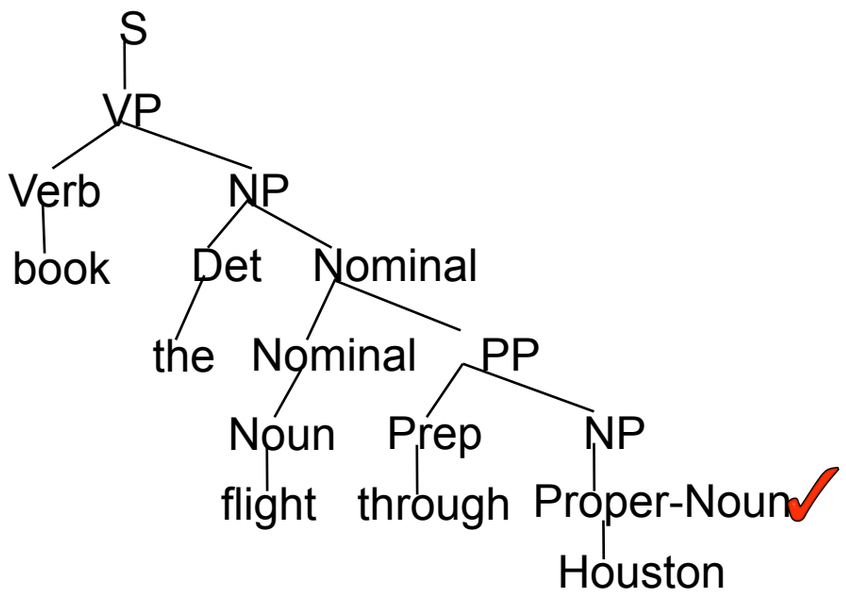
P - вычисленное дерево



компонент: 12

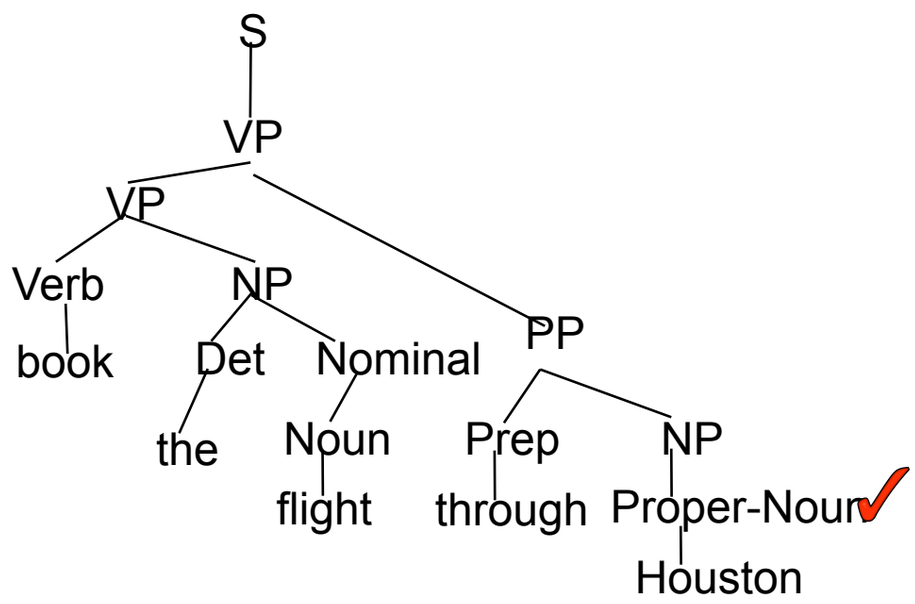
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

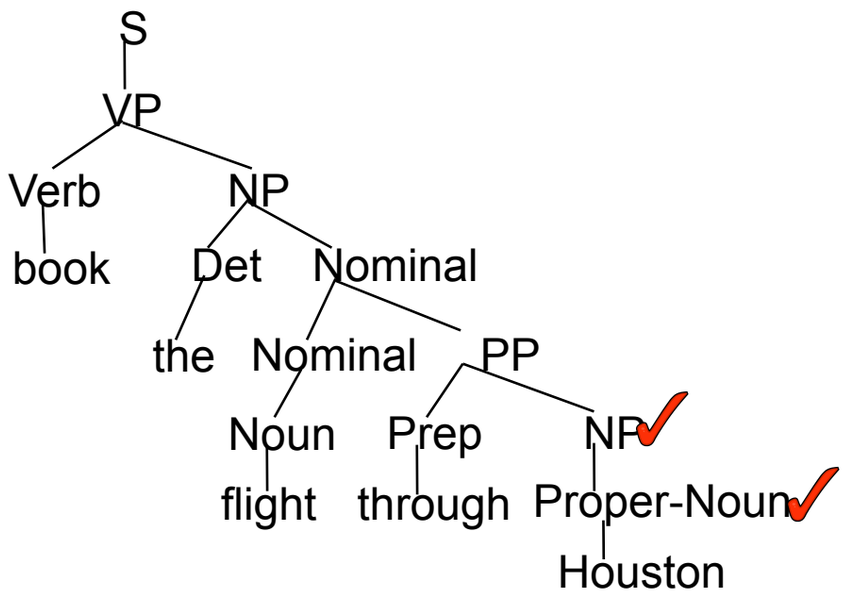
P - вычисленное дерево



компонент: 12

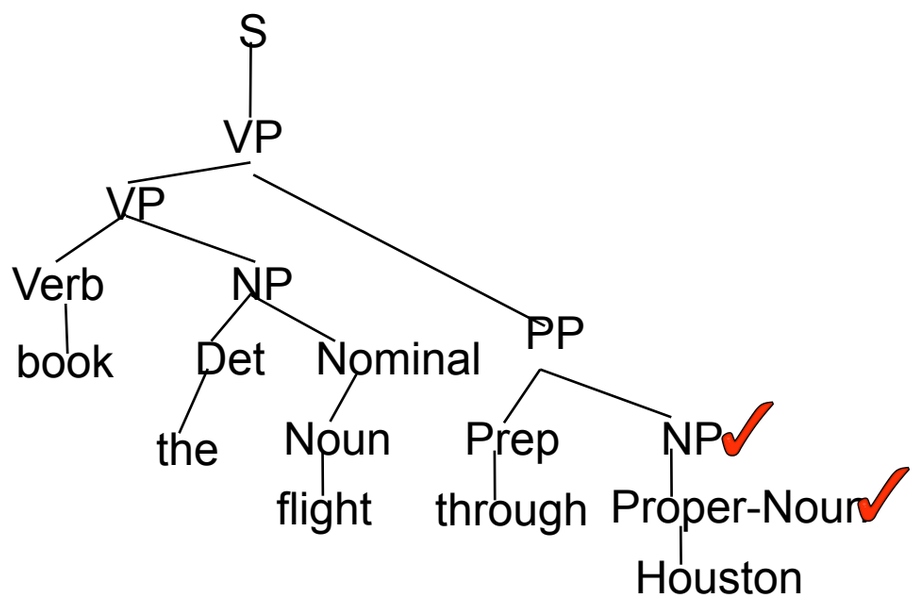
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

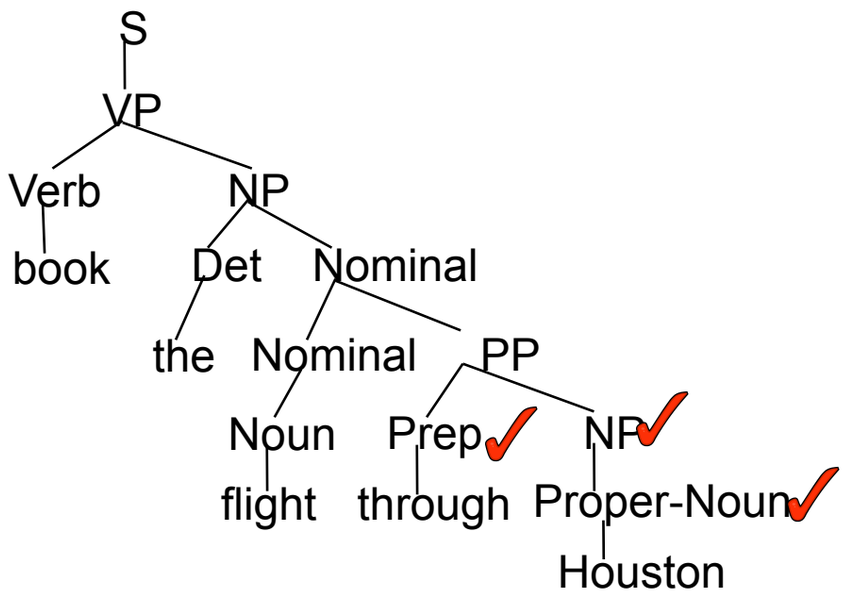
P - вычисленное дерево



компонент: 12

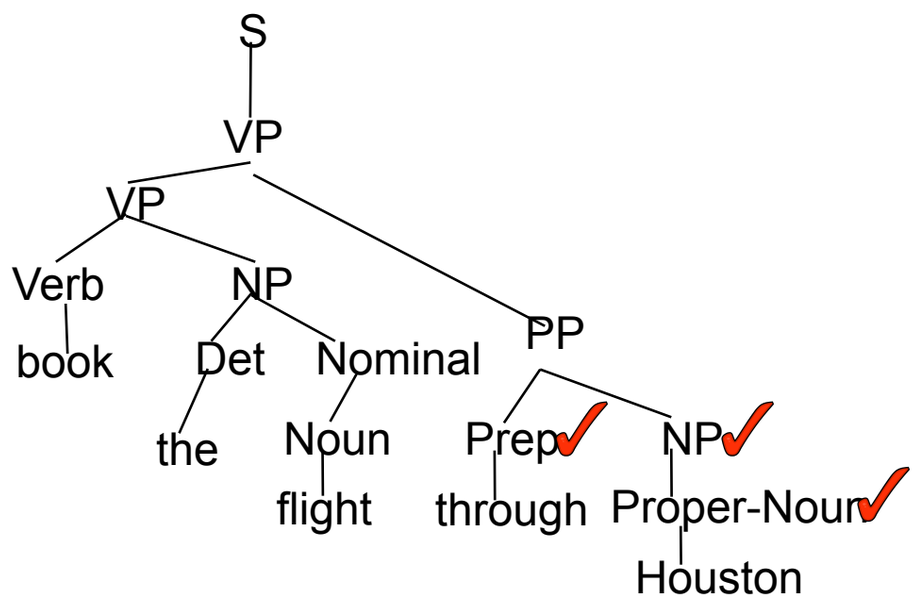
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

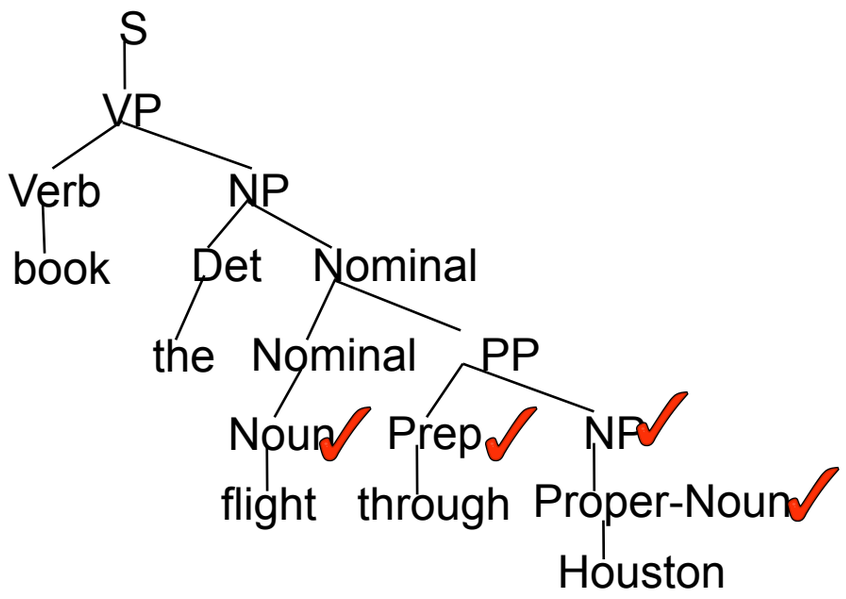
P - вычисленное дерево



компонент: 12

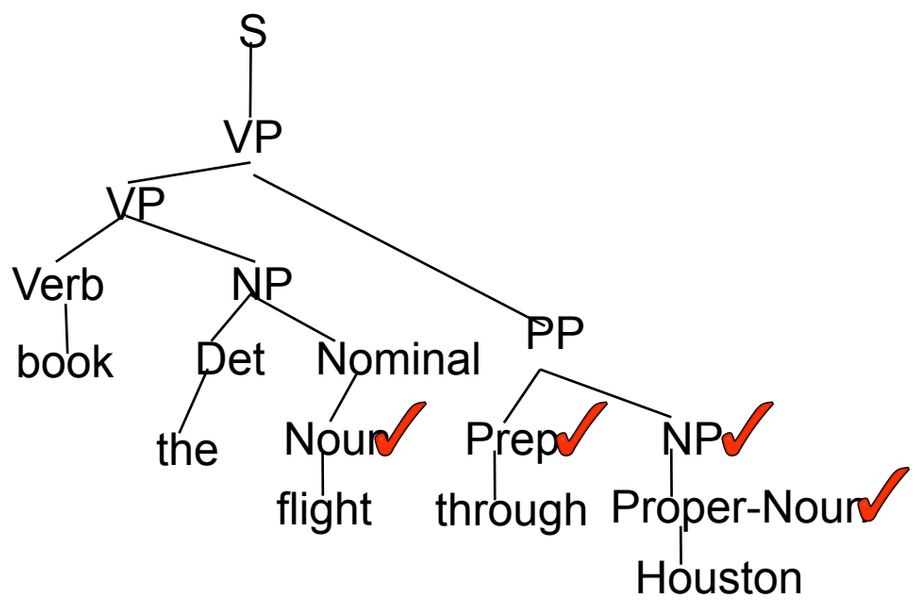
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

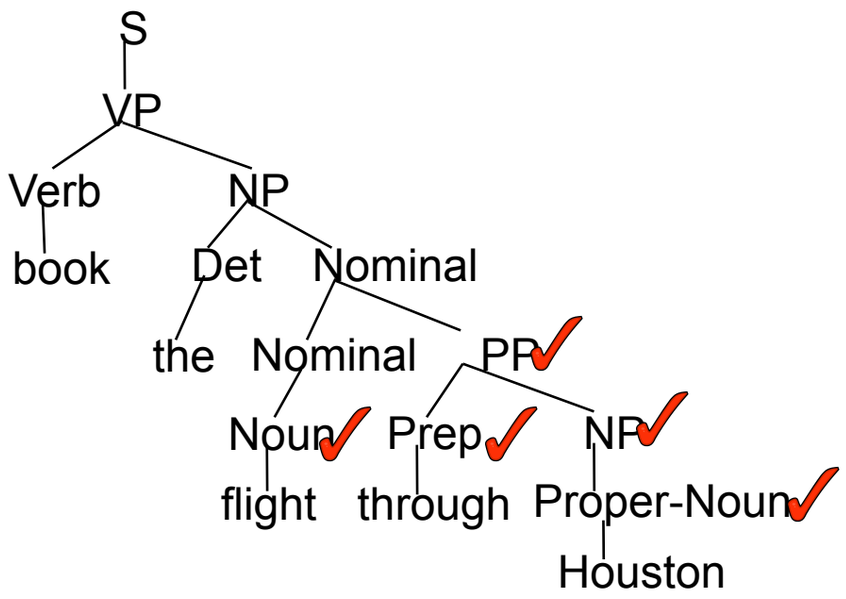
P - вычисленное дерево



компонент: 12

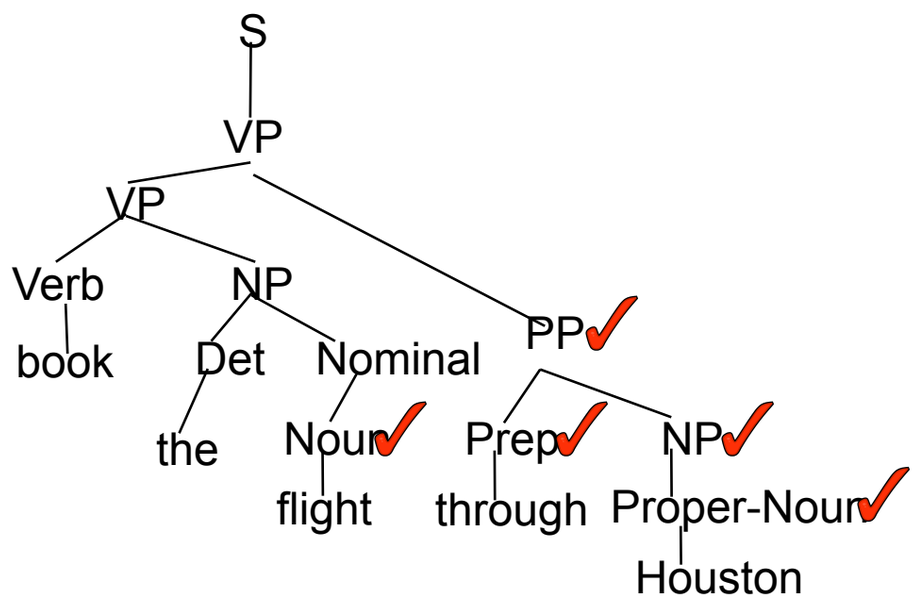
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

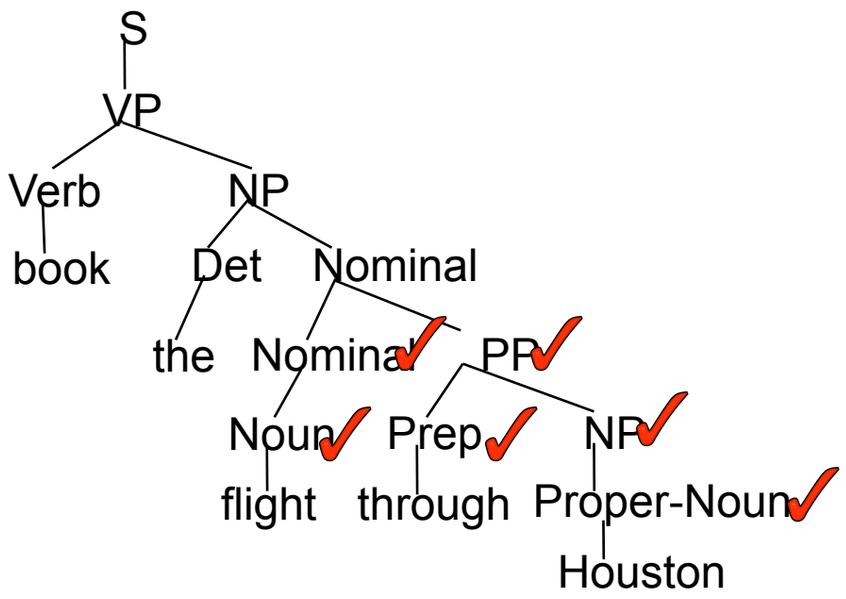
P - вычисленное дерево



компонент: 12

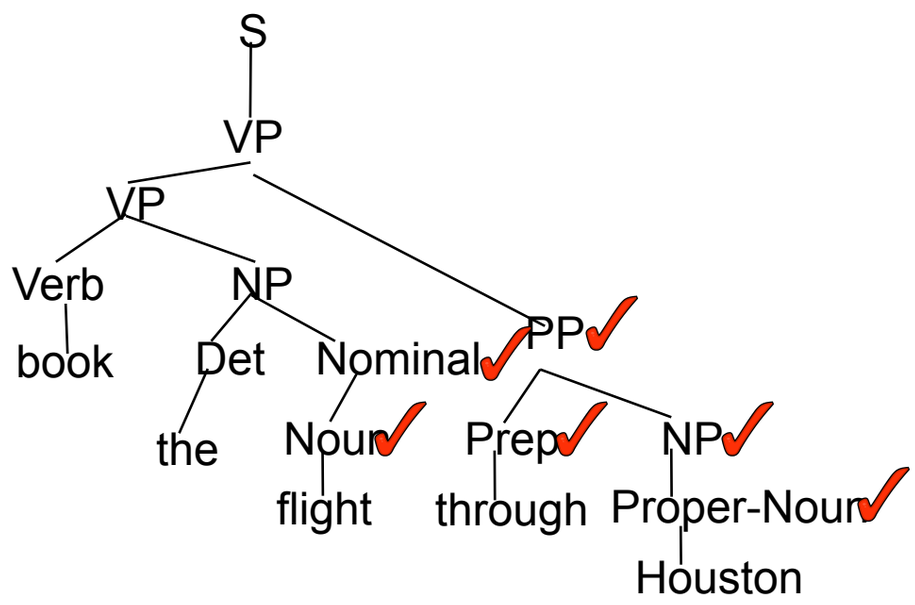
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

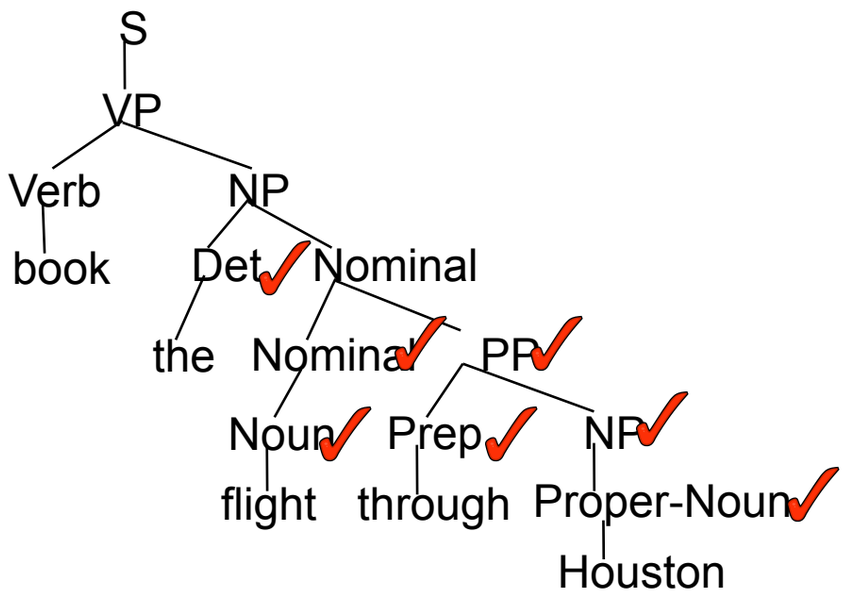
P - вычисленное дерево



компонент: 12

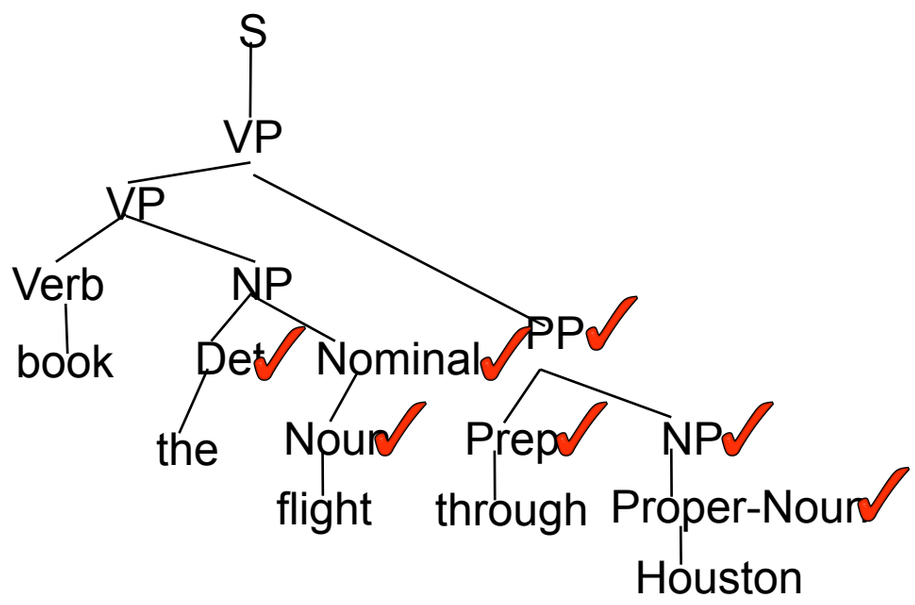
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

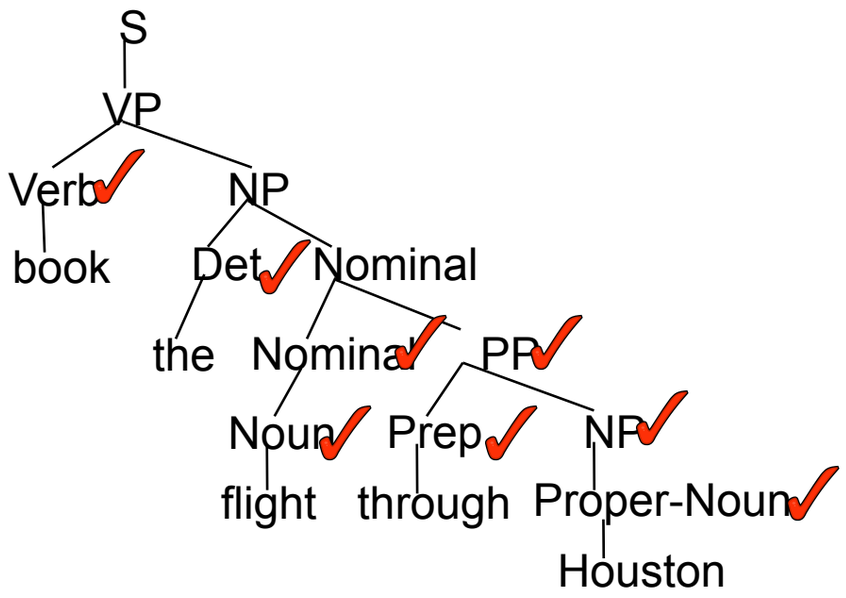
P - вычисленное дерево



компонент: 12

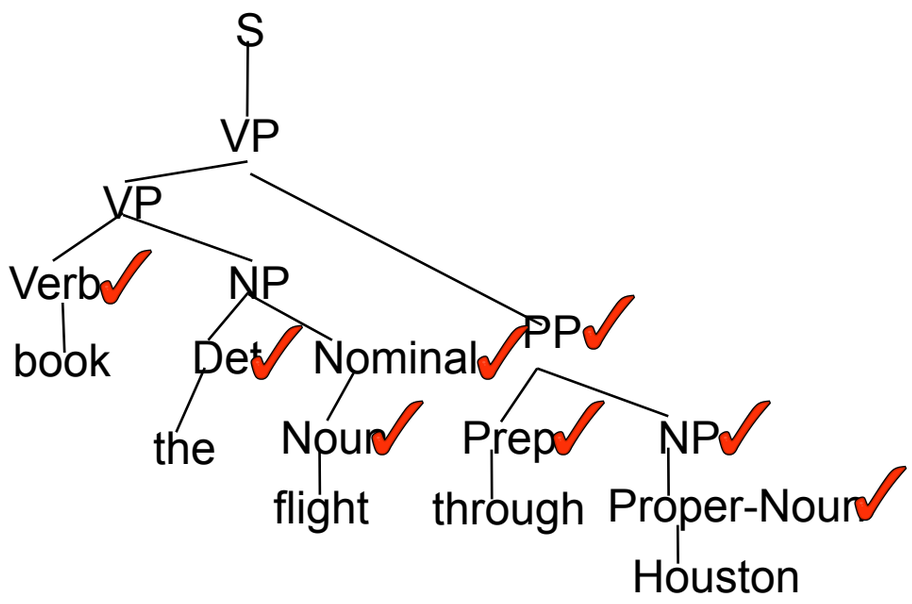
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

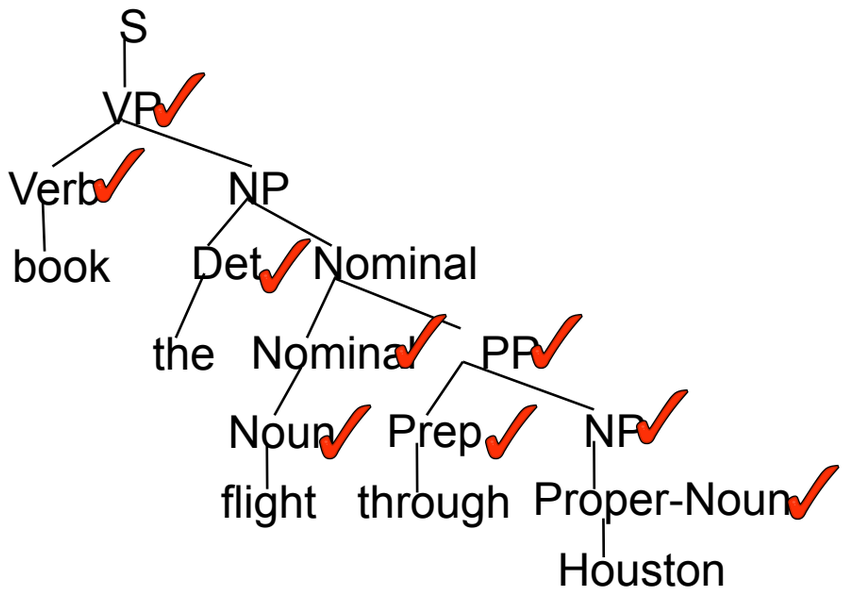
P - вычисленное дерево



компонент: 12

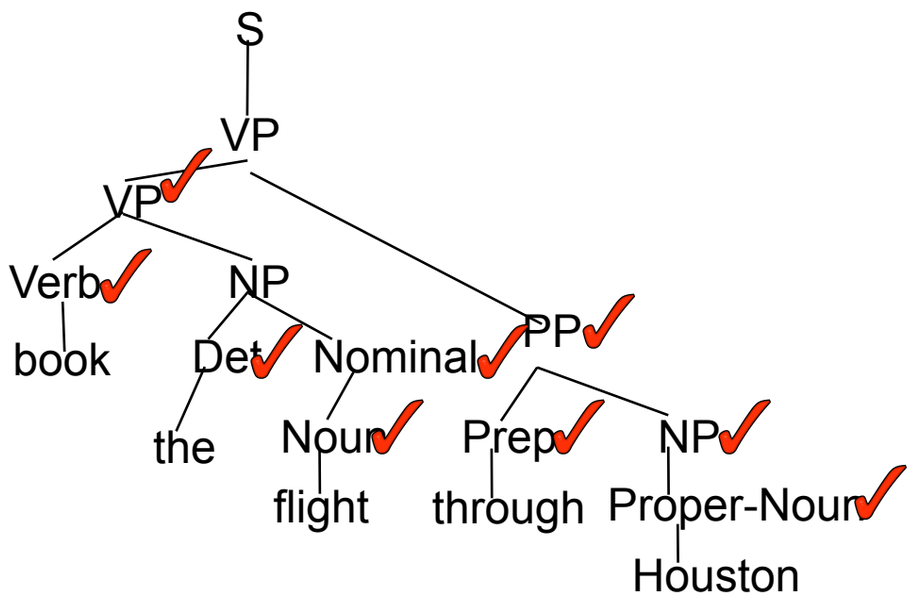
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

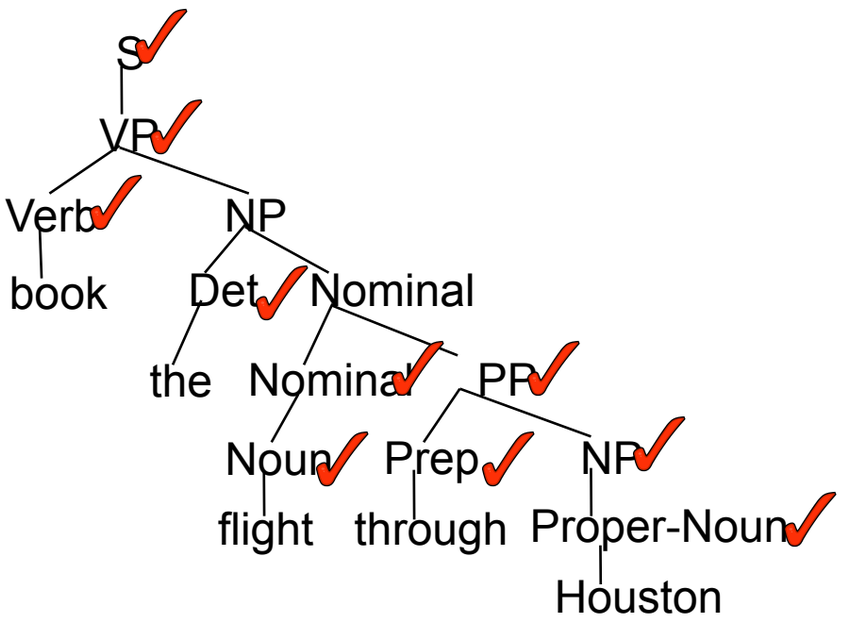
P - вычисленное дерево



компонент: 12

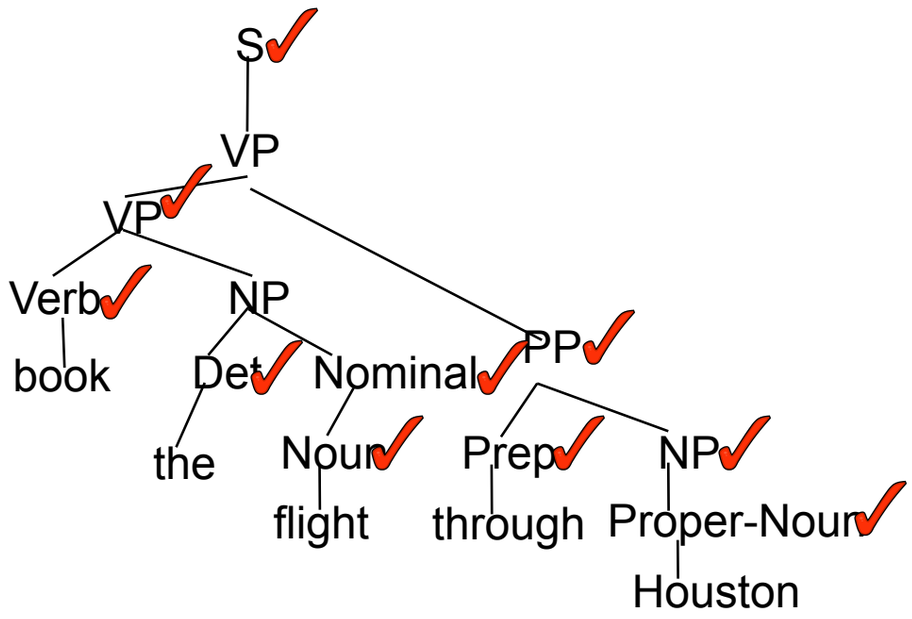
Оценка качества алгоритма

T - дерево, размеченное вручную



компонент: 12

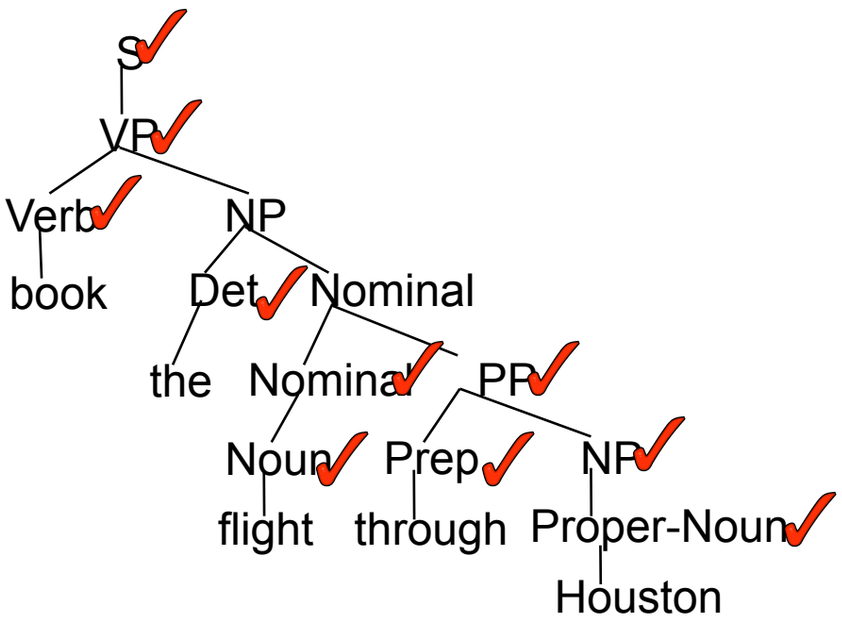
P - вычисленное дерево



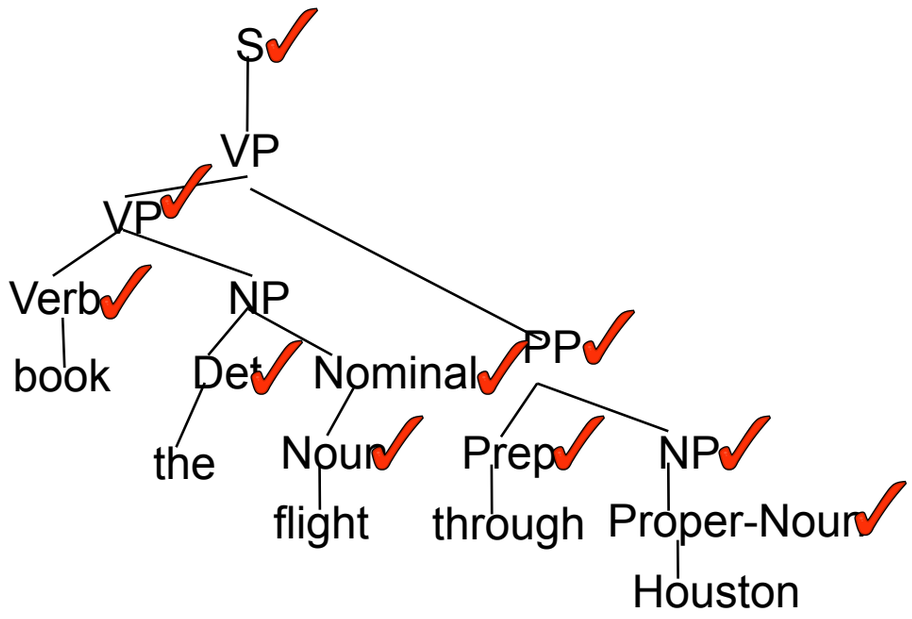
компонент: 12

Оценка качества алгоритма

T - дерево, размеченное вручную



P - вычисленное дерево



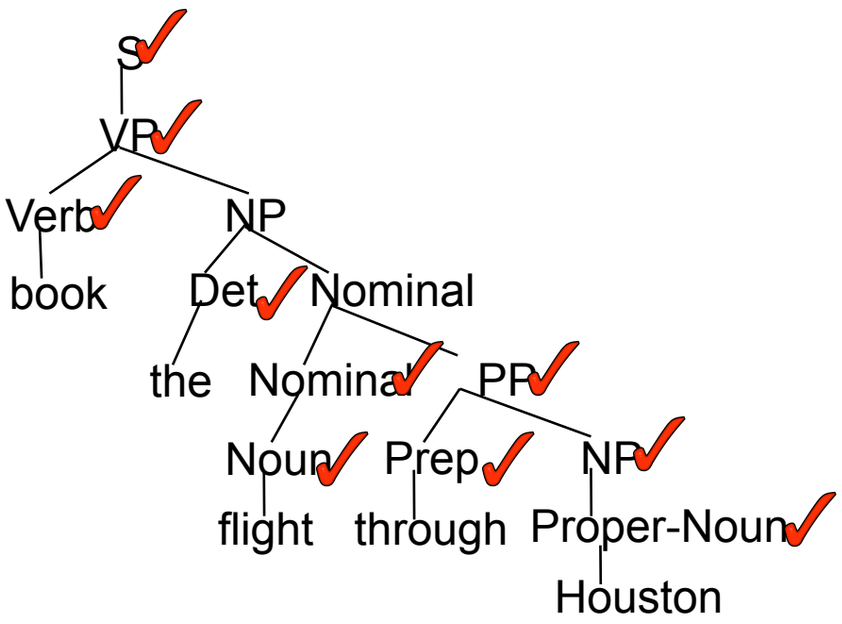
компонент: 12

компонент: 12

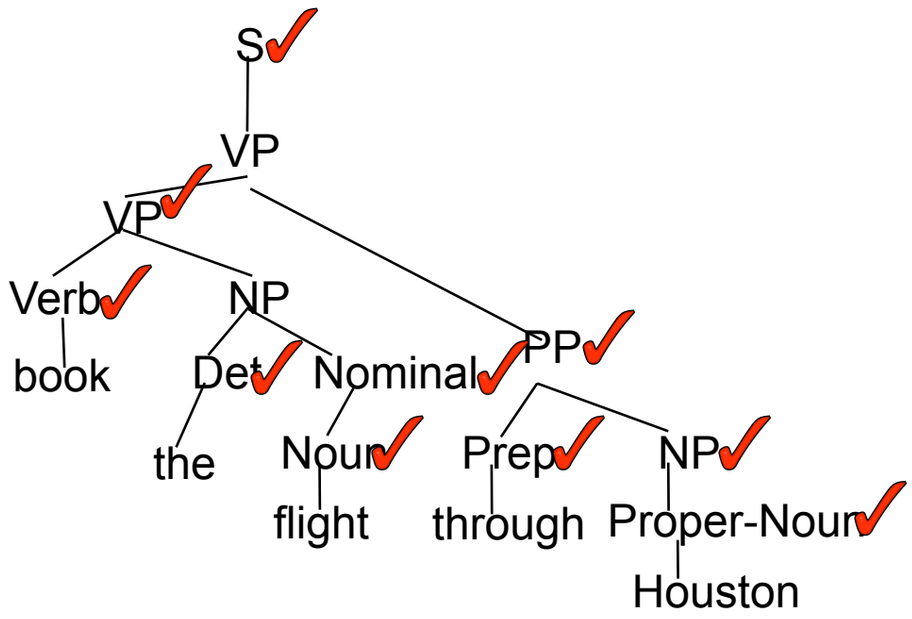
правильных компонент: 10

Оценка качества алгоритма

T - дерево, размеченное вручную



P - вычисленное дерево



компонент: 12

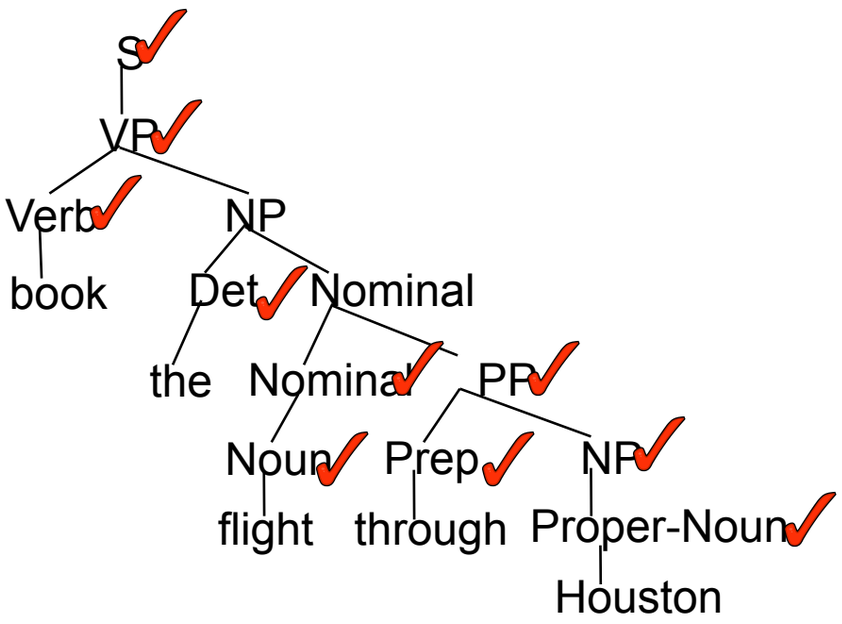
компонент: 12

правильных компонент: 10

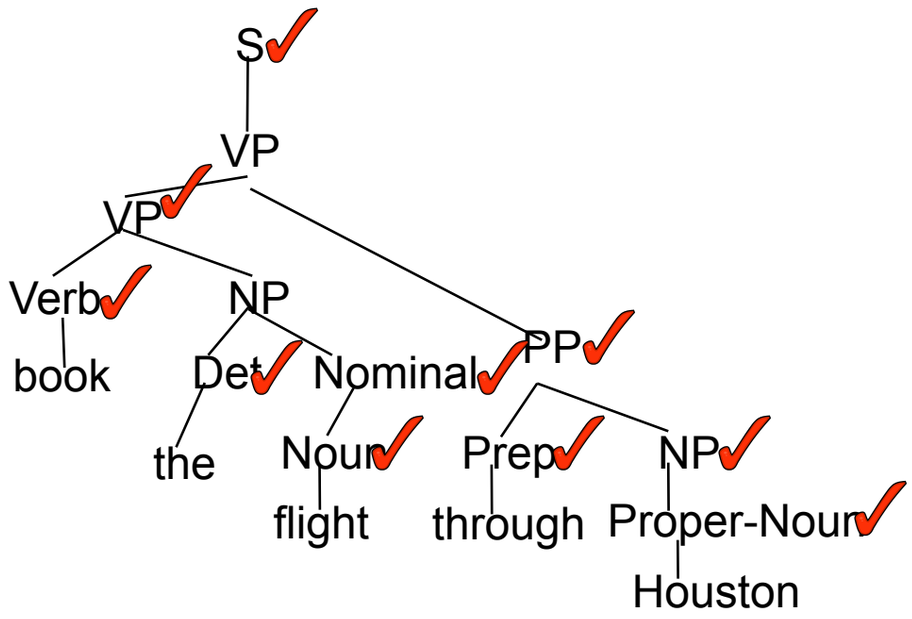
Точность = 10/12= 83.3%

Оценка качества алгоритма

T - дерево, размеченное вручную



P - вычисленное дерево



компонент: 12

компонент: 12

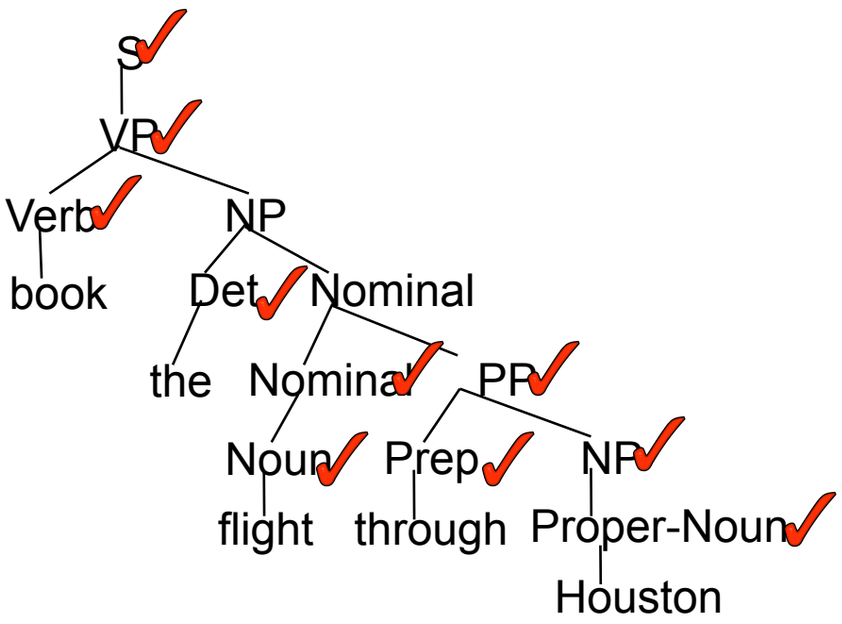
правильных компонент: 10

Точность = 10/12= 83.3%

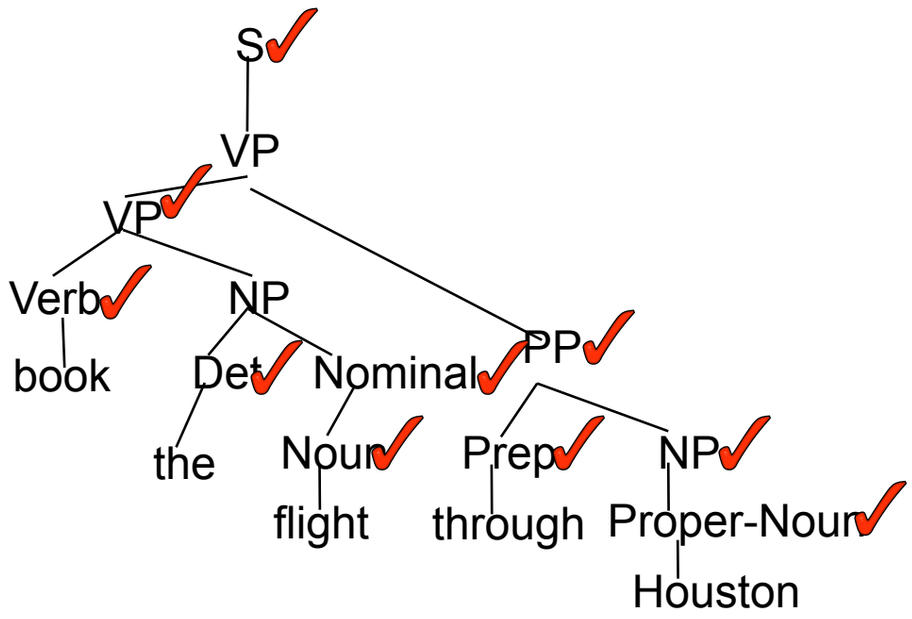
Полнота = 10/12= 83.3%

Оценка качества алгоритма

T - дерево, размеченное вручную



P - вычисленное дерево



компонент: 12

компонент: 12

правильных компонент: 10

Точность = 10/12= 83.3%

Полнота = 10/12= 83.3%

F_1 = 83.3%

Делают ли люди синтаксический разбор?

- Психолингвистика
- Алгоритмы синтаксического разбора могут быть использованы для предсказания времени, которое потребуется человеку для прочтения каждого слова в предложении
- Чем выше вероятность слова, тем скорость чтения больше
- Для моделирования этого эффекта требуется инкрементальный алгоритм

Предложения с временной неоднозначностью

- Garden path sentence
  - –Complex houses married students
  - –The horse raced past the barn fell
- Инкрементальные парсеры могут найти и объяснить сложность таких предложений

Сложность языка

- Является ли естественный язык регулярным?
  - –контр-пример был на прошлой лекции
- Является ли естественный язык контекстно-свободным?
  - –Диалект немецкого языка в Швейцарии содержит контекстно-зависимые конструкции вида $a^n b^m c^n d^m$
- Сложность понимания людьми
  - –чем проще конструкция, тем легче понимание смысла текста

Заключение

- Статистические модели, такие как СКС позволяют разрешать многозначность
- СКС может можно выучить на основе банка деревьев
- Учет лексики и разделение нетерминальных символов позволяет разрешить дополнительные неоднозначности
- Точность современных алгоритмов синтаксического разбора высока, но не достигает уровня экспертного разбора

Следующая лекция

- Лексическая семантика и разрешение лексической многозначности