Виды модульных систем автоматической обработки текста

Андрей Попов

Кафедра Математической лингвистики СПбГУ,
проект Хурма

hedgeonline@gmail.com

Что за Хурма?

- Проект объединяет сотрудников и студентов кафедры Математической лингвистики СПбГУ;
- Проект стартовал в октябре 2015 года для участия в соревновании FactRuEval-2016 в рамках Dialogue Evaluation;
- Мы участвовали в дорожках:
  - извлечение именованных сущностей;
  - нормализация и кореференция;
  - извлечение фактов;

Что мы сделали

Доработали существующую модульную систему анализа текстов для участия в соревновании:

- в основном усилия были сосредоточены на задачах извлечения и нормализации именованных сущностей;
- времени на привлечение синтаксического анализа не было;

Получили результаты (F1):

- Извлечение ИС: 0.8077 (место 9/13, лучш.: 0.8672)
- Нормализация ИС: 0.7439 (место 5/5, лучш.: 0.7979)
- Извлечение фактов: 0.3246 (место 2/2, лучш.: 0.5100)

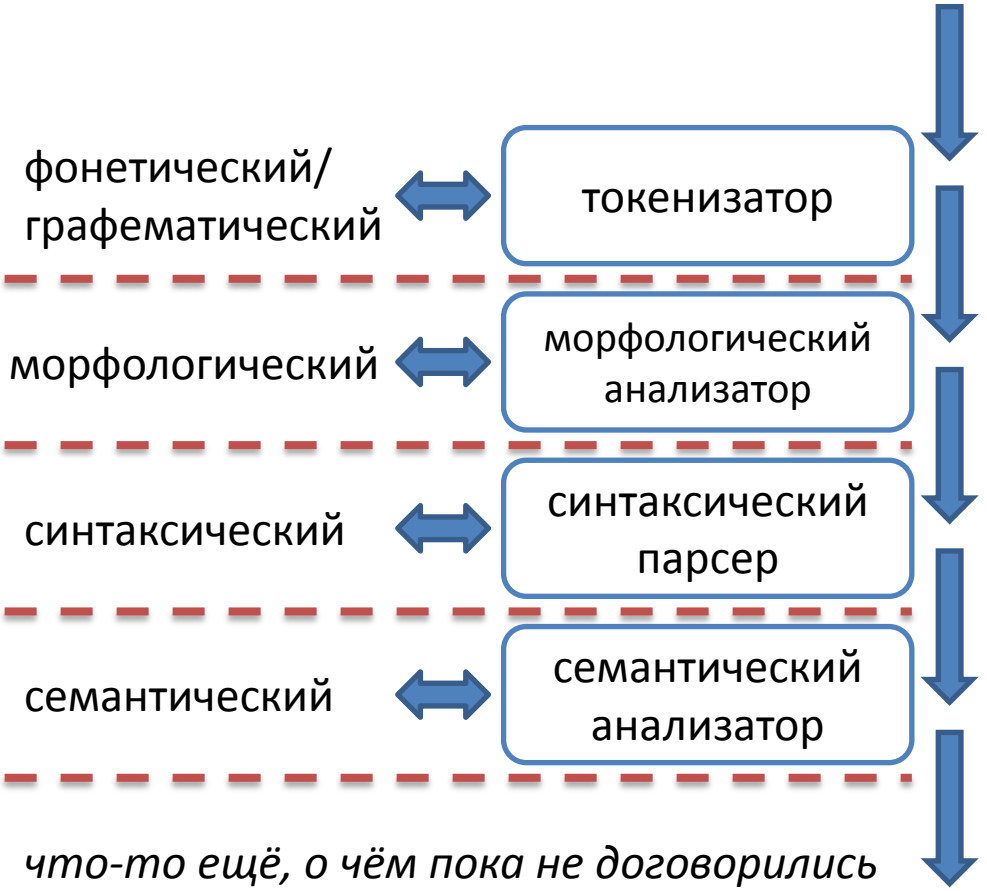
И, проанализировав всё, что мы сделали…

…мы поняли, что всё надо было делать не так!

Почему модульные?

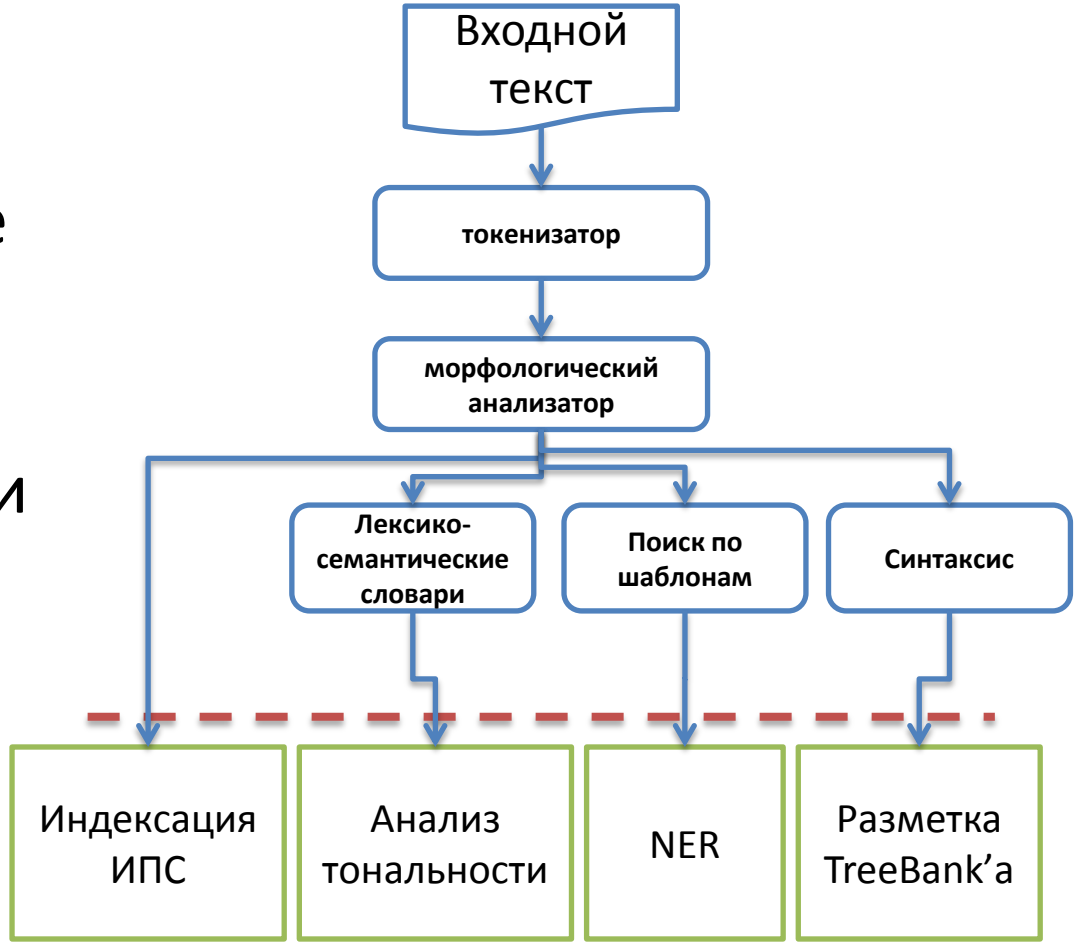
Каждому «уровню» анализа естественного языка хочется сопоставить свой модуль в анализаторе!

…правда, это не всегда получается…



Почему модульные?

Разные прикладные задачи могут быть разбиты на различные подзадачи, решающиеся различными модулями



Вывод

Если мы хотим гибко-настраиваемую систему автоматического анализа текста общего назначения, то разумным выбором будет модульная архитектура.

Основные требования

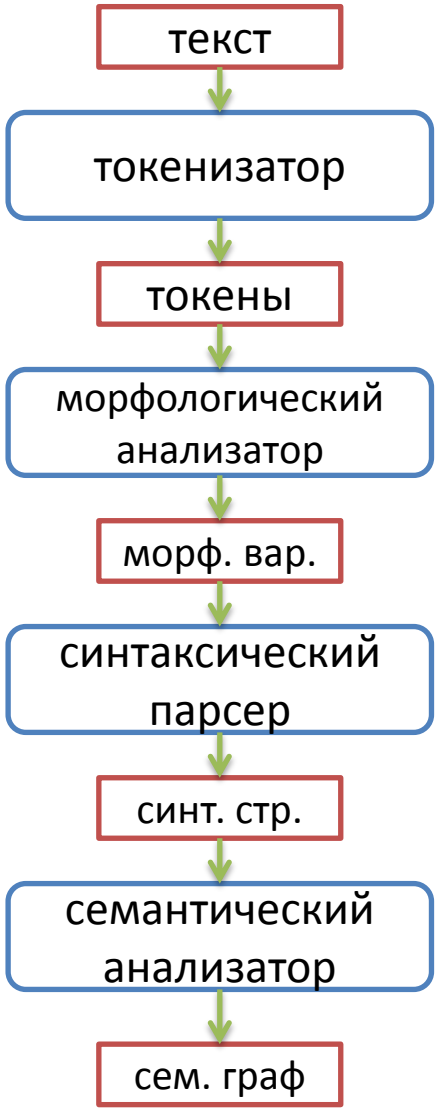
1. Система должна обеспечивать возможность конфигурации пользователем под конкретную задачу без помощи программиста;
2. Пользователь должен иметь возможность сконфигурировать любой конвейер, т.е. создать произвольную последовательность любых доступных модулей.

Ключевой вопрос

Как модули должны между собой общаться?

Или чем они должны обмениваться?

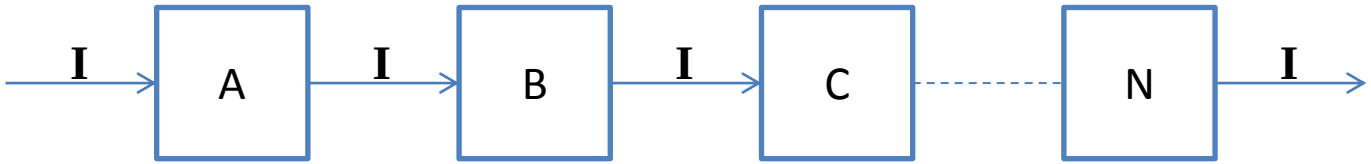
…очевидно, что при такой модели переставить что-либо местами или убрать из конвейера не получится…



Вывод

Каждый модуль должен иметь

стандартный интерфейс ввода и вывода



Утверждение

Именно свойства этого «интерфейса» и правила работы с ним определяют архитектурный тип модульной системы.

«Классические» системы или системы типа «А»

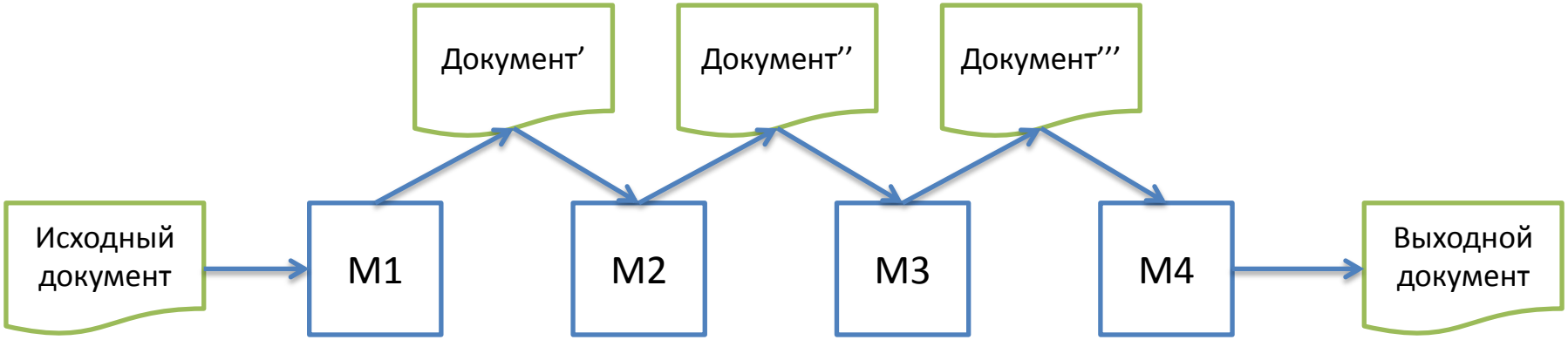
Большинство прикладных задач обработки текста ориентированы на обработку некоторой сущности, которую можно назвать «документ»:

- Новостная статья;
- Сообщение в социальной сети;
- Литературное произведение.

«Документ» удобно представлять физически в виде:

- Файла в файловой системе;
- Записи в базе данных.

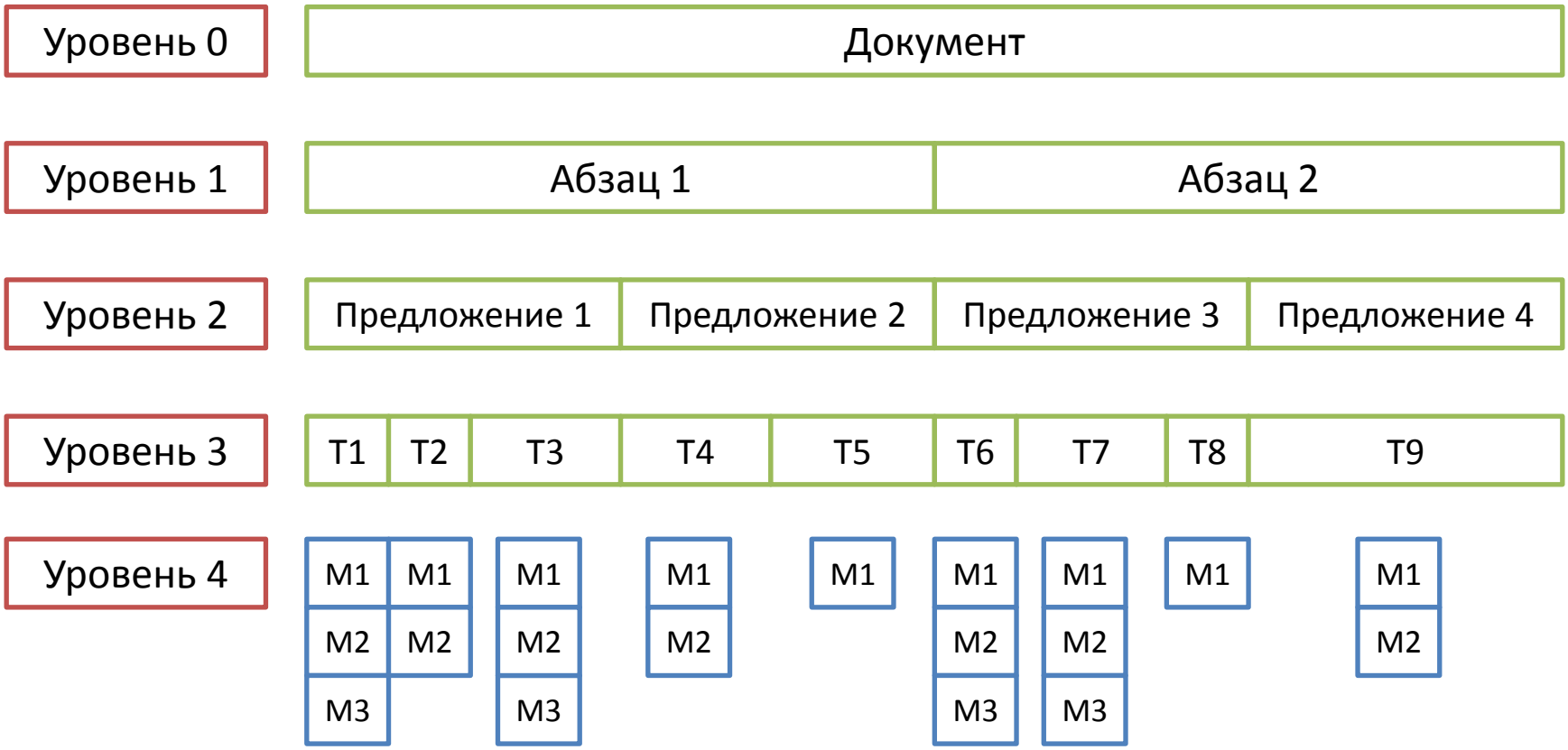
«Классические» системы или системы типа «А»



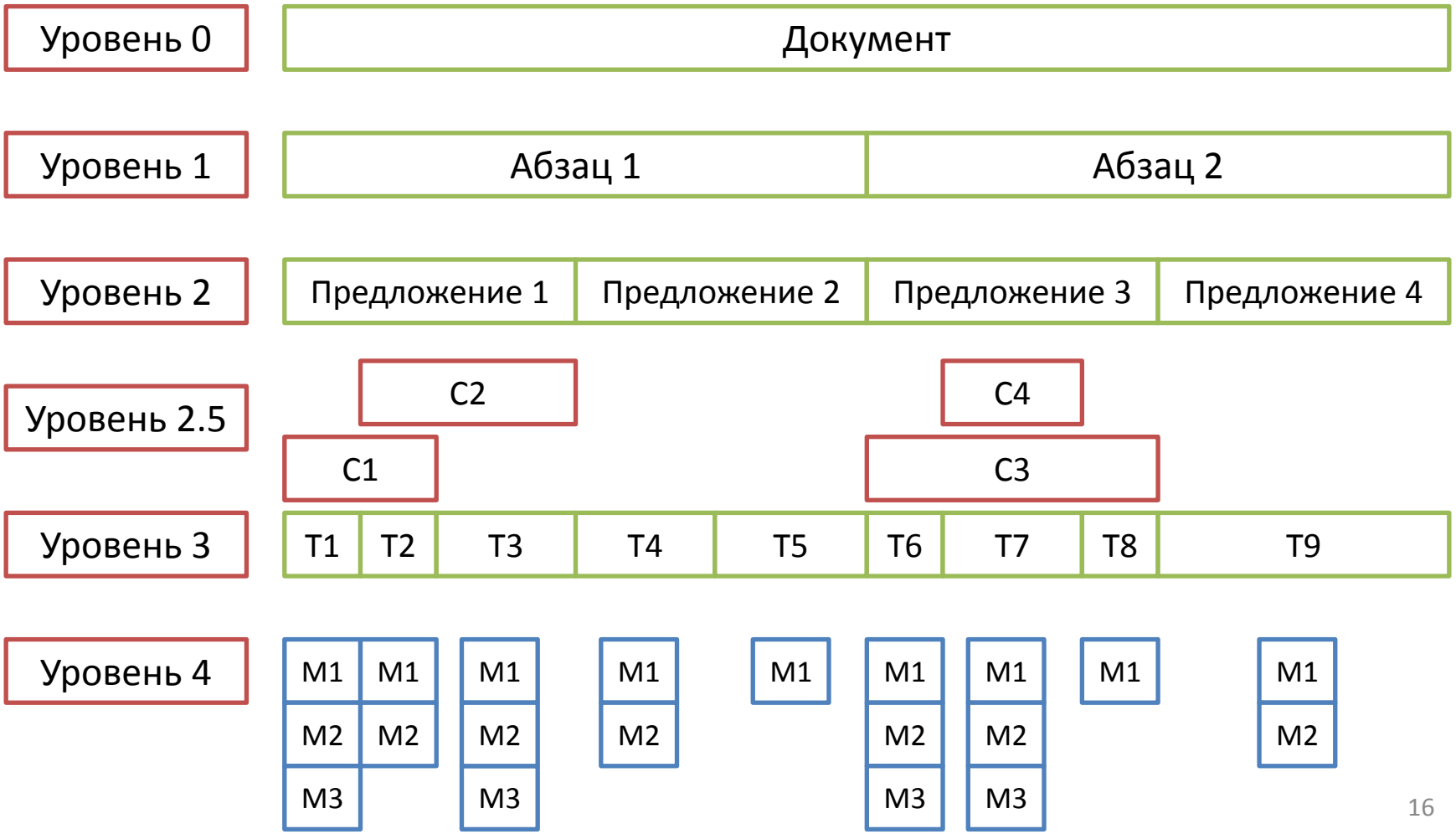
Характеристика систем типа «А»

- Ориентированы на документ;
- Модули работают по одному разу «слева направо»;
- Каждый модуль принимает на вход целый документ, как-то его преобразует и выдаёт на выходе модифицированный документ.

Структура документа



Единицы между словом и предложением



Проблемы извлечения именованных сущностей

...отставкой главы **<u>города Владимира Мамасуева</u>** и добровольным...

город Владимир
[LOC]

Владимир Мамасуев
[PER]

Юрий Никулин. Всё течёт (2013.05.31) // «Новгородские ведомости», 2013

Оставить, нельзя, выбрать

Какие варианты борьбы с «города Владимира Мамасуева»?

- Если система поддерживает пересекающиеся сущности, оставить оба варианта;
- Если не поддерживает – выбрать что-то одно (что и как?!).

Поддержка пересекающихся сущностей

- Если система и алгоритмы извлечения информации изначально проектировались из расчёта на линейно однозначную цепочку токенов, то реализовать поддержку пересекающихся сущностей пост-фактум может оказаться экономически нецелесообразным;
- Если необходимо выгружать результат работы системы в структурированный формат для анализа, то при наличии пересекающихся сущностей специалисту становится труднее читать файл.

Алгоритмическая поддержка, нужна ли она?

Правило 1: [должность][персона]

Примеры: *директор Иванов*

генеральный директор Иванов

генеральный директор Ян Иванов

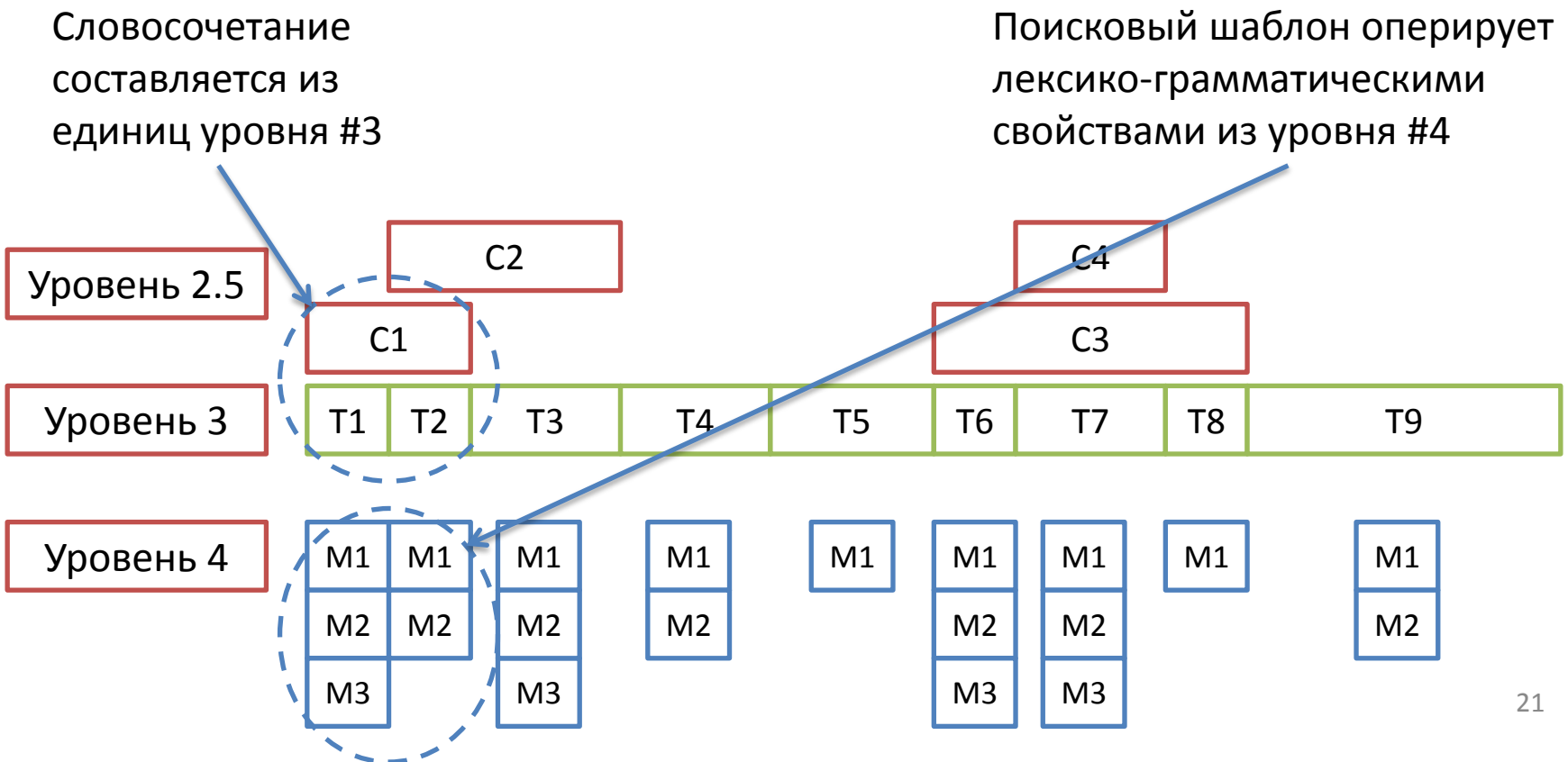
Правило 2: [прилагательное][дескриптор]

Примеры: *Ленинградская область*

Еврейская автономная область

Другая проблема поиска по шаблонам

При выделении словосочетаний шаблон неявно взаимодействует с единицами разных уровней



Это разве проблема?

Правило: [прилагательное][существительное]

Пример:	московская	знать
	Сущ,жр,им,ед,фам	Сущ,жр,им,ед
	Прил,жр,им,ед	Сущ,жр,вн,ед Инф

А это разве проблема?

Правило: [прилагательное][существительное]

Пример:	московская	знать
	~~Сущ,жр,им,ед,фам~~	Сущ,жр,им,ед
	Прил,жр,им,ед	Сущ,жр,вн,ед
		~~Инф~~

И где проблема?

Правило: [прилагательное][существительное]
+ согласование(род, число, падеж)

Пример:	московская	знать
	~~Сущ,жр,им,ед,фам~~	Сущ,жр,им,ед
	Прил,жр,им,ед	Сущ,жр,вн,ед
		~~Инф~~

Покажите, наконец, проблему!

Правило: [прилагательное][существительное]
+ согласование(род, число, падеж)

Пример: московская знать

~~Сущ,жр,им,ед,фам~~ Сущ,жр,им,ед

Прил,жр,им,ед ~~Сущ,жр,вн,ед~~

~~Инф~~

Хьюстон, у нас проблема!

Правило: [имя][фамилия]

+ согласование(род, число, падеж)

Пример: Александра Иванова

Сущ,жр,им,имя  Сущ,жр,им,фам

Сущ,мр,вн,имя  Сущ,мр,вн,фам

Сущ,мр,рд,имя  Сущ,мр,рд,фам

Нормальная форма:

Александр Иванов или Александра Иванова?!

Вывод

Для корректной нормализации необходимо хранить информацию о согласованных морфологических интерпретациях, иначе наряду с верными нормальными формами:

Александр Иванов;

Александра Иванова;

у нас будут появляться некорректные:

~~Александр Иванова;~~

~~Александра Иванов.~~

Функция согласования

- Возвращает истину или ложь, в зависимости от того, согласованы ли слова или нет;
- Для каждого согласуемого слова возвращает множество допустимых морфологических интерпретаций;
- Для всего словосочетания возвращает множество согласованных морфологических интерпретаций.

Основные проблемы систем с архитектурой типа «А»

- Сложность представления языковой омонимии различных уровней;
- Сложность передачи между модулями «нестандартной» информации, не укладывающейся в модель документа;
- Потенциальная возможность комбинаторного взрыва;
- Сложность работы с задачами, ориентированными на поток.

Открытые вопросы по архитектуре типа «А»

- Производительность вообще и в многопоточном режиме?
- Распознавание речи?
- Диалоговые системы?

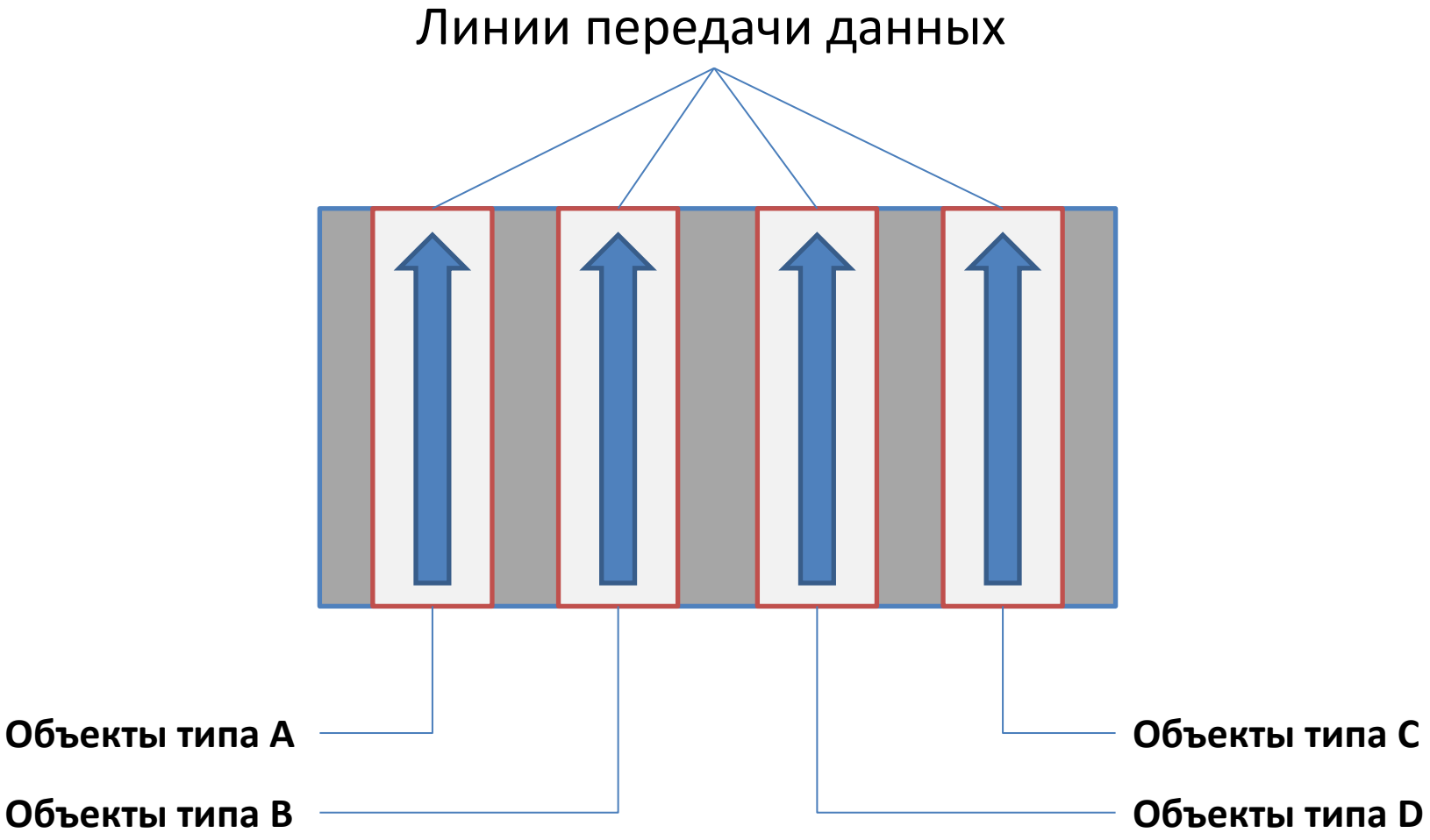
Системы типа «В»

- Система ориентирована не на документ (заранее известной длины), а на поток (произвольной длины);
- В качестве интерфейса взаимодействия модулей выбирается не документ, а «микроинтерфейс» в виде линий передачи лингвистических объектов сравнительно небольшого размера;
- В нормальном режиме работы управление в ходе анализа может передаваться и выше- и нижестоящим модулям.

Библиографическая справка

- Рубашкин В.Ш. в «Онтологической семантике» уделяет внимание проблеме *межуровневого взаимодействия* [с. 253-256] и ссылается при этом на:
- Цейтин Г.С. Программирование на ассоциативных сетях 1985.

Схематичное представление абстрактного модуля



Типы линий передачи данных

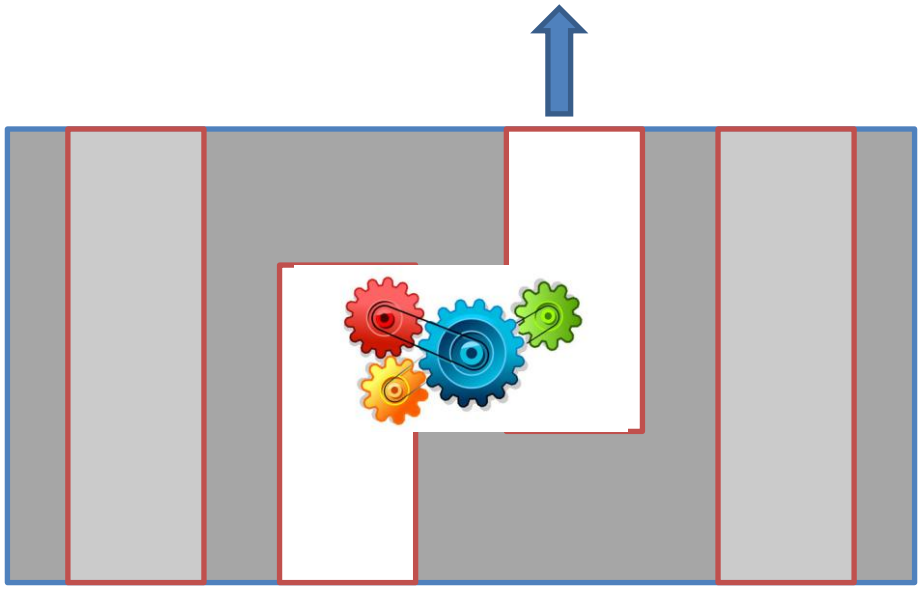
- Входные данные (символы/байты/фреймы);
- Диапазоны (токены/сущности/составляющие);
- Факты (множество заполненных полей);
- Сигналы (начало/конец документа/абзаца/предложения).

Типы модулей

- Кумулятивный:
  - модуль принимает и накапливает объекты какого-либо типа и порождает на их основе объекты того же самого или другого типа;
- Мультипликативный:
  - модуль принимает объекты одного типа и на основе каждого порождает множество новых, отличных по содержанию от исходного;
- Фильтрационный:
  - модуль принимает объекты одного типа и в зависимости от критерия фильтрации или пропускает его дальше вверх по каскаду или отбрасывает.

Кумулятивный модуль (токенизатор)

токены входного текста



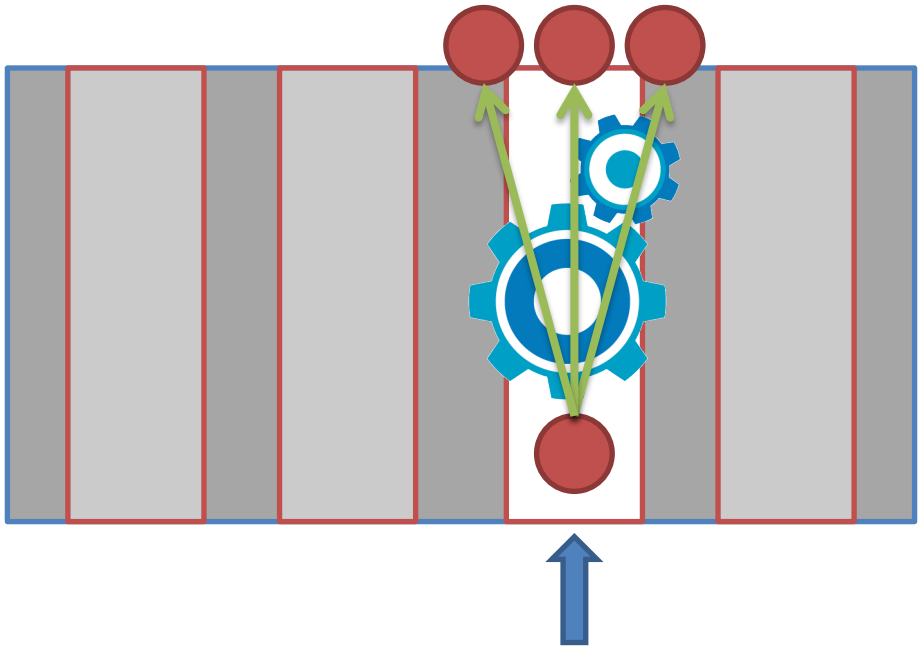
символы входного текста

Типы модулей

- Кумулятивный:
  - модуль принимает и накапливает объекты какого-либо типа и порождает на их основе объекты того же самого или другого типа;
- Мультипликативный:
  - модуль принимает объекты одного типа и на основе каждого порождает множество новых, отличных по содержанию от исходного;
- Фильтрационный:
  - модуль принимает объекты одного типа и в зависимости от критерия фильтрации или пропускает его дальше вверх по каскаду или отбрасывает.

Мультипликативный модуль (морфологический анализатор)

токены+
входного+
текста+

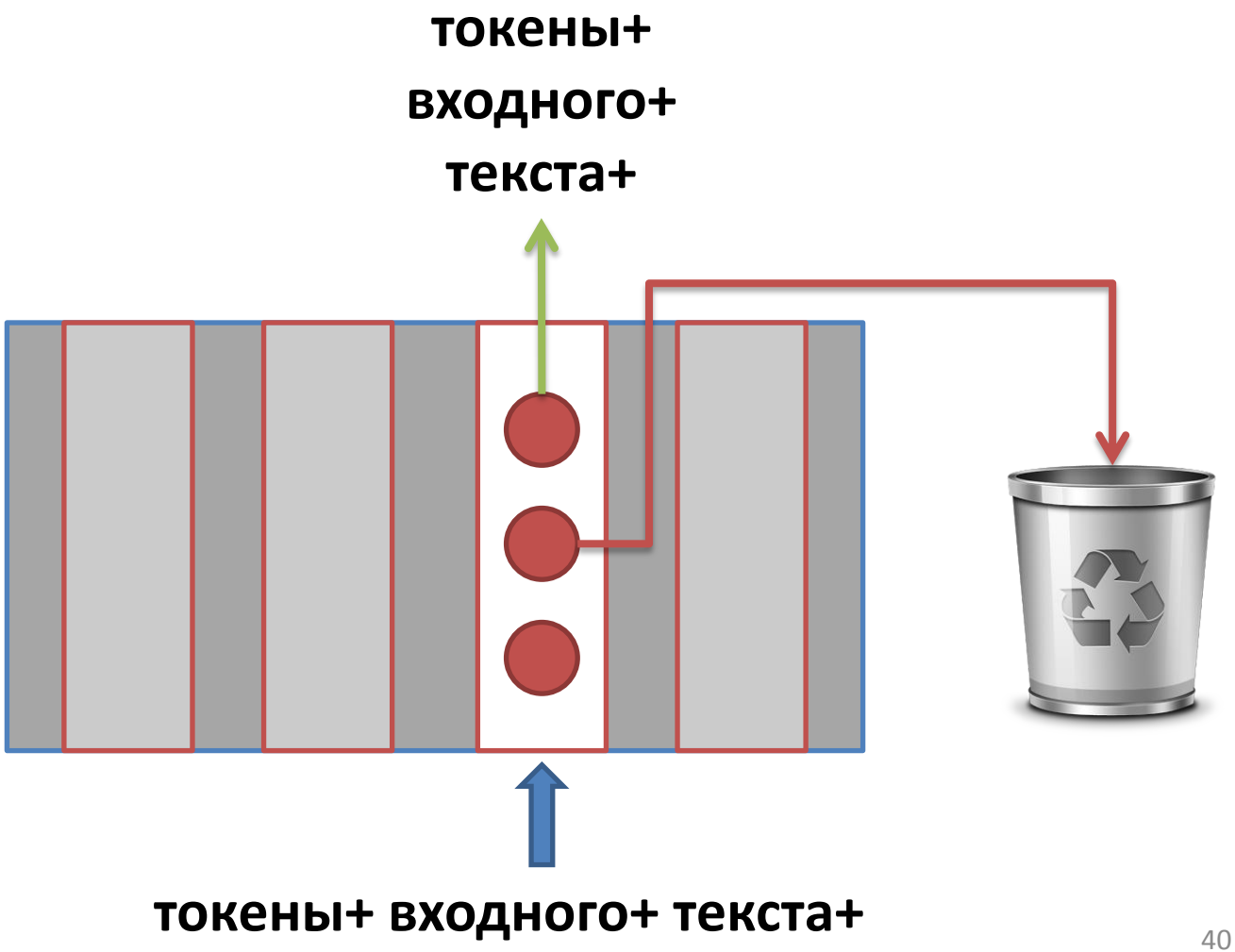


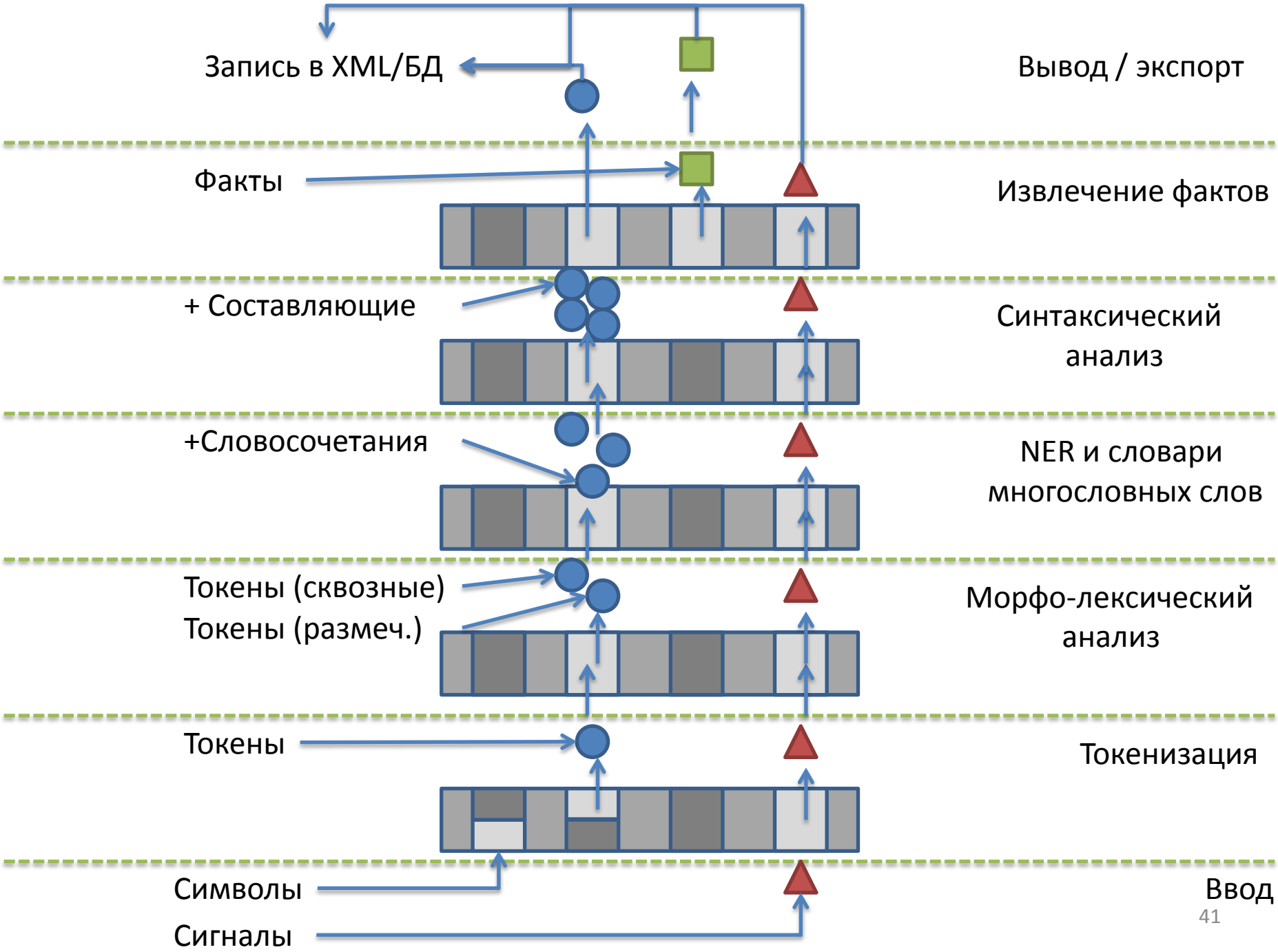
токены входного текста

Типы модулей

- Кумулятивный:
  - модуль принимает и накапливает объекты какого-либо типа и порождает на их основе объекты того же самого или другого типа;
- Мультипликативный:
  - модуль принимает объекты одного типа и на основе каждого порождает множество новых, отличных по содержанию от исходного;
- Фильтрационный:
  - модуль принимает объекты одного типа и в зависимости от критерия фильтрации или пропускает его дальше вверх по каскаду или отбрасывает.

Фильтрационный модуль (морфологический дизамбигуатор)

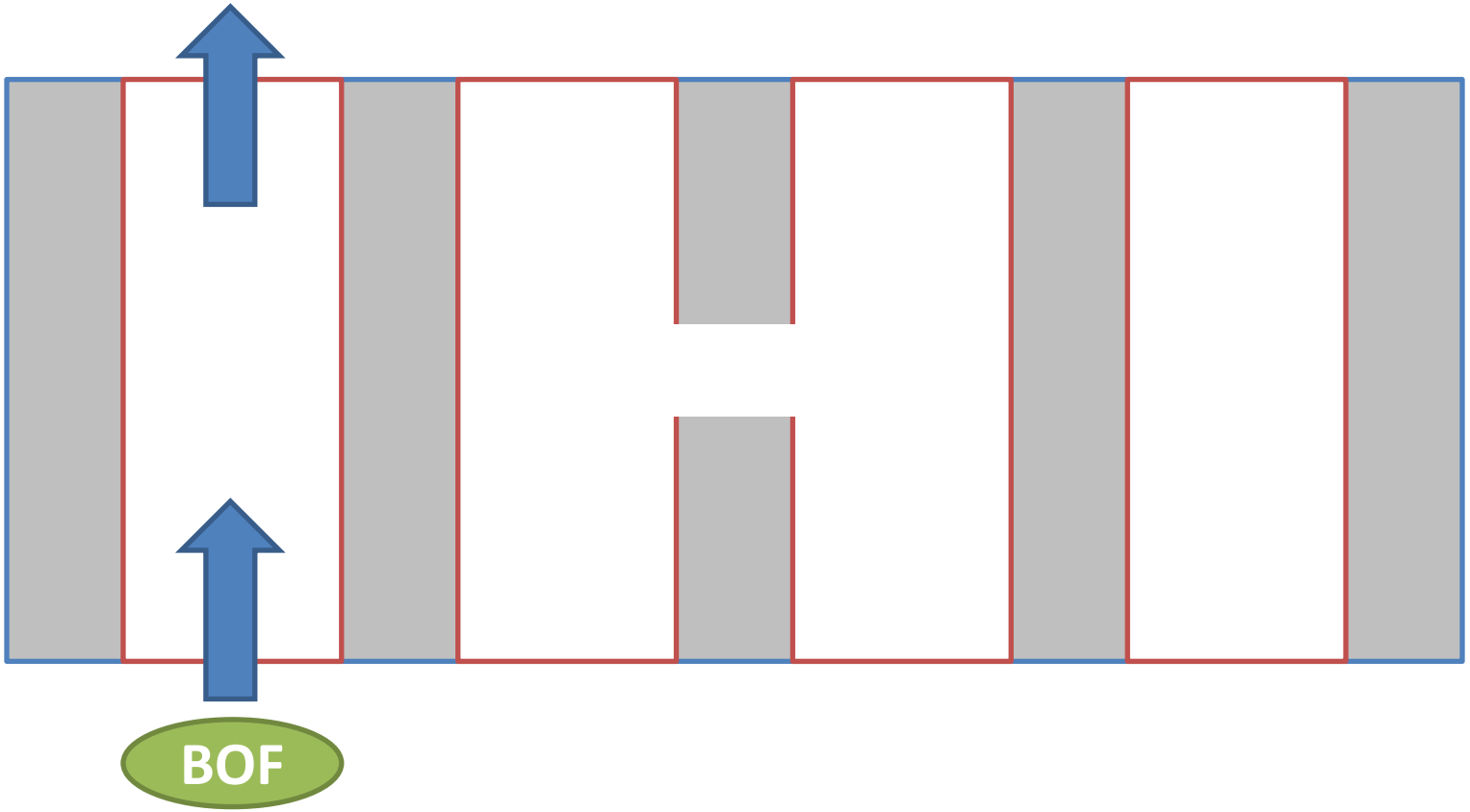



Запись в XML/БД
Вывод / экспорт
Факты
Извлечение фактов
+ Составляющие
Синтаксический анализ
+Словосочетания
NER и словари многословных слов
Токены (сквозные)
Токены (размеч.)
Морфо-лексический анализ
Токены
Токенизация
Символы
Сигналы
Ввод

Особенности реализации модуля синтаксического анализа

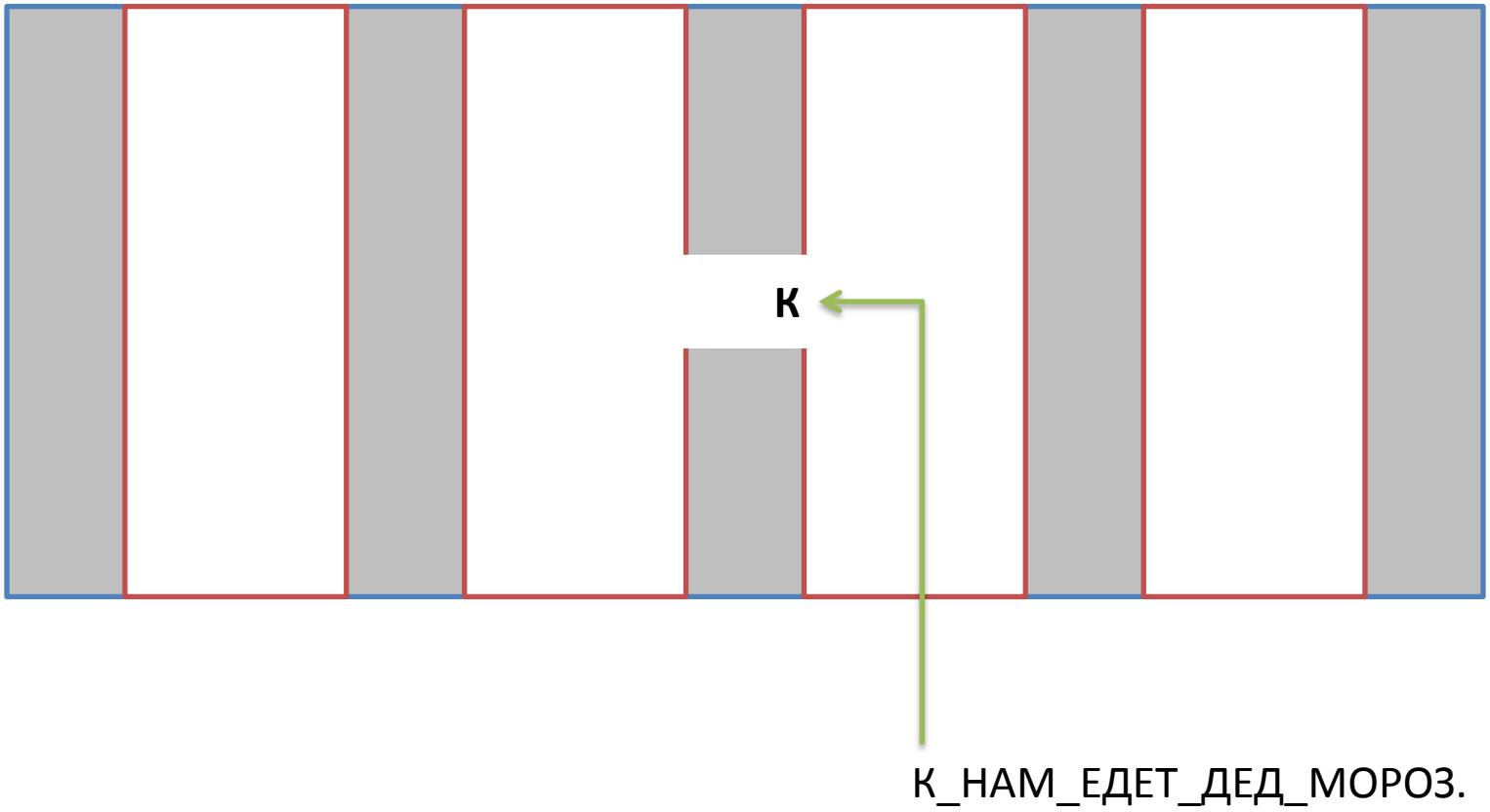
- Обычно, синтаксические анализаторы принимают на вход непересекающиеся атомарные единицы (токены)
  - > на досинтаксическом этапе нужно решить задачу «сборки» многословных сущностей или вообще её не решать (тогда она будет решаться за счёт синтаксиса).
- Наша архитектура позволяет передавать на синтаксический анализ пересекающиеся единицы
  - > мы можем одновременно применять оба способа «сборки» многословных сущностей.

Токенизация (1)

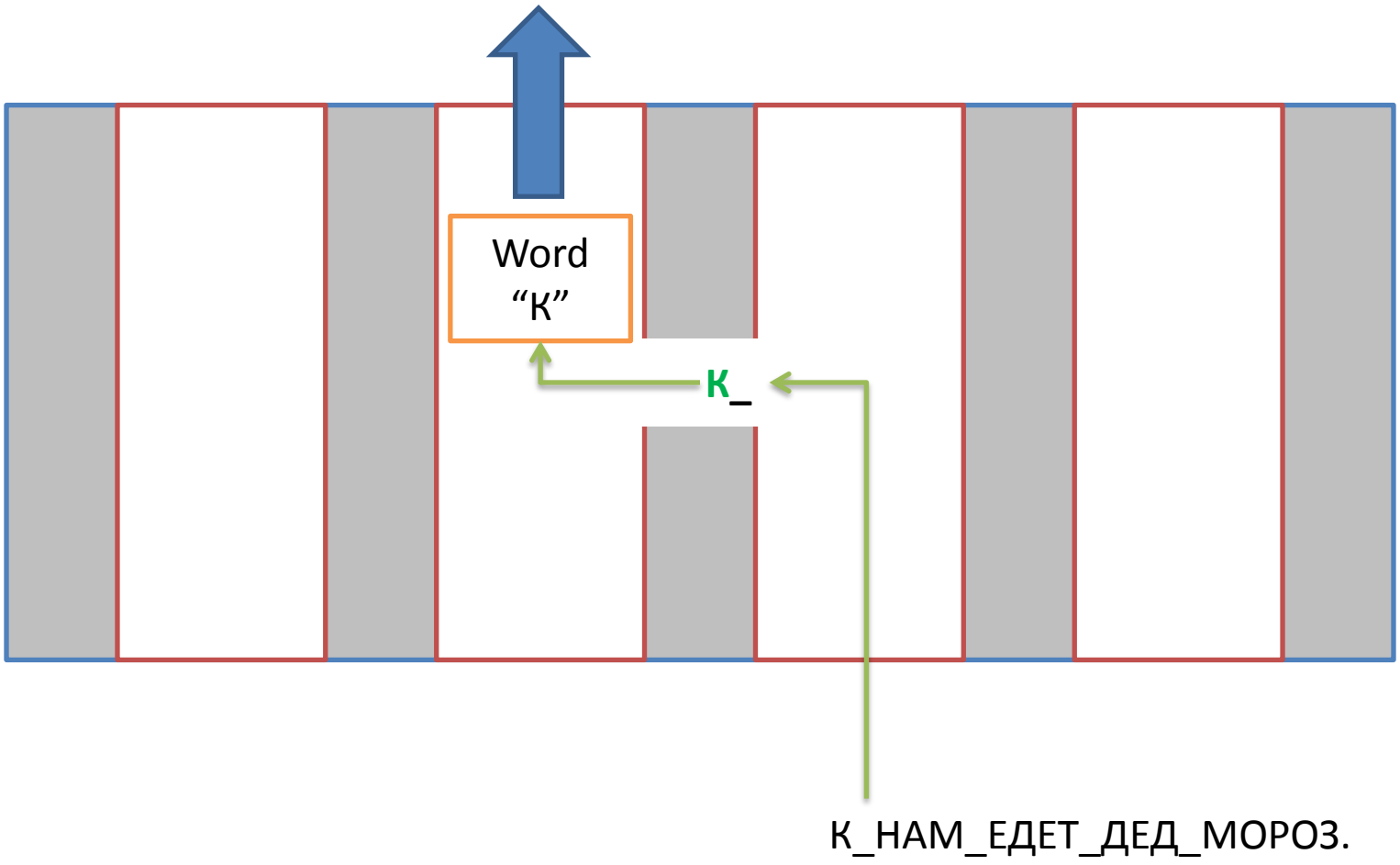


К_НАМ_ЕДЕТ_ДЕД_МОРОЗ.

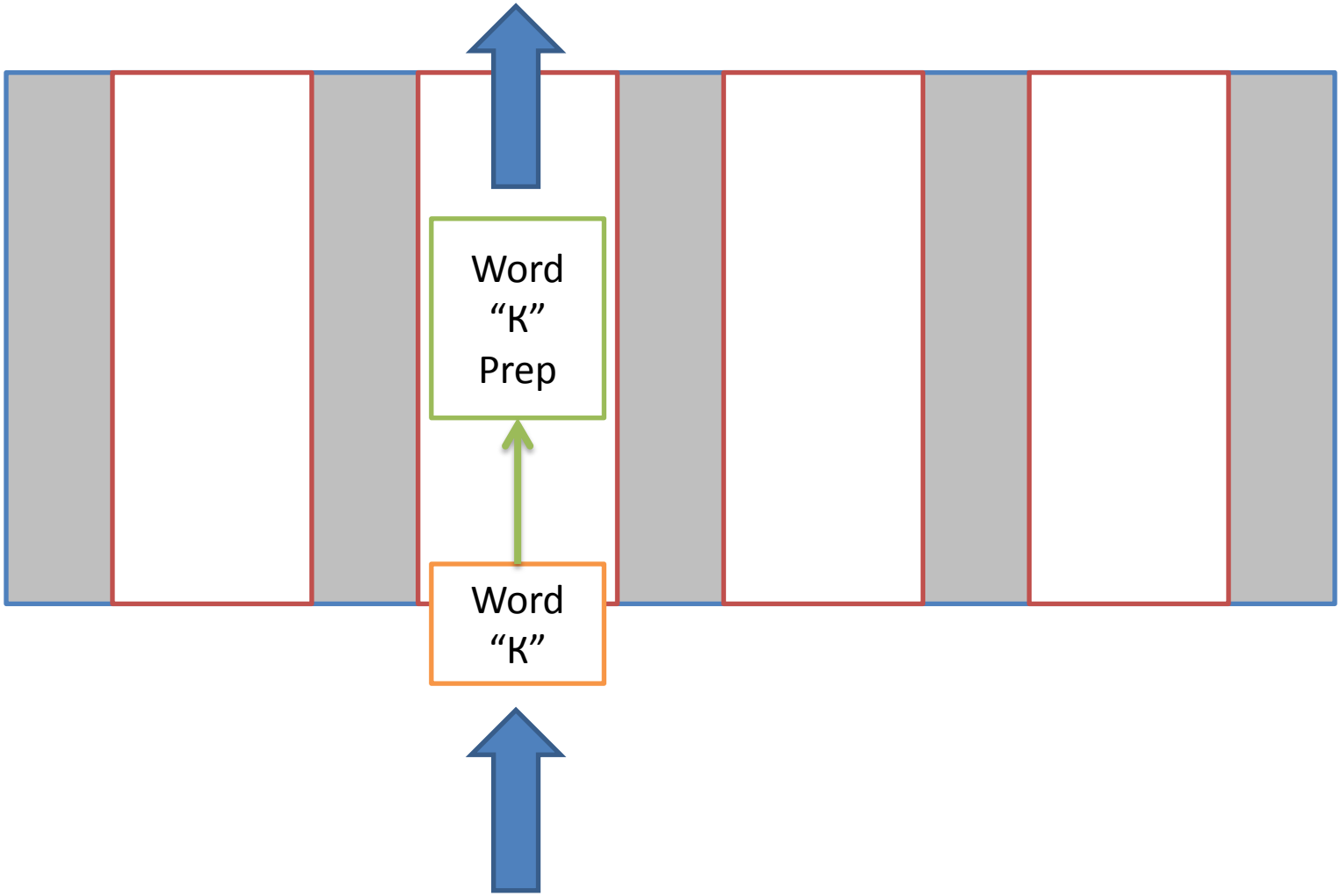
Токенизация (2)



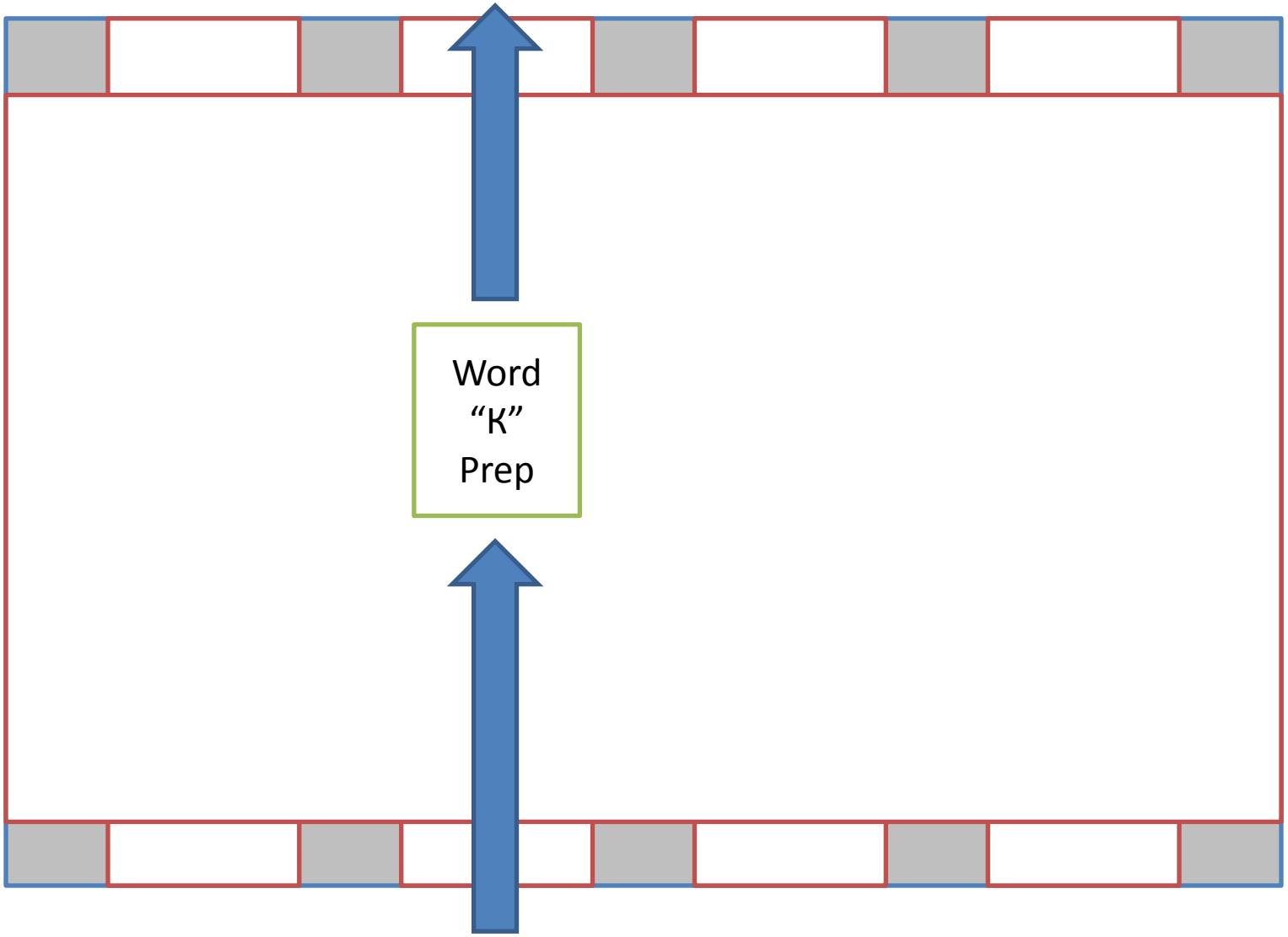
Токенизация (3)



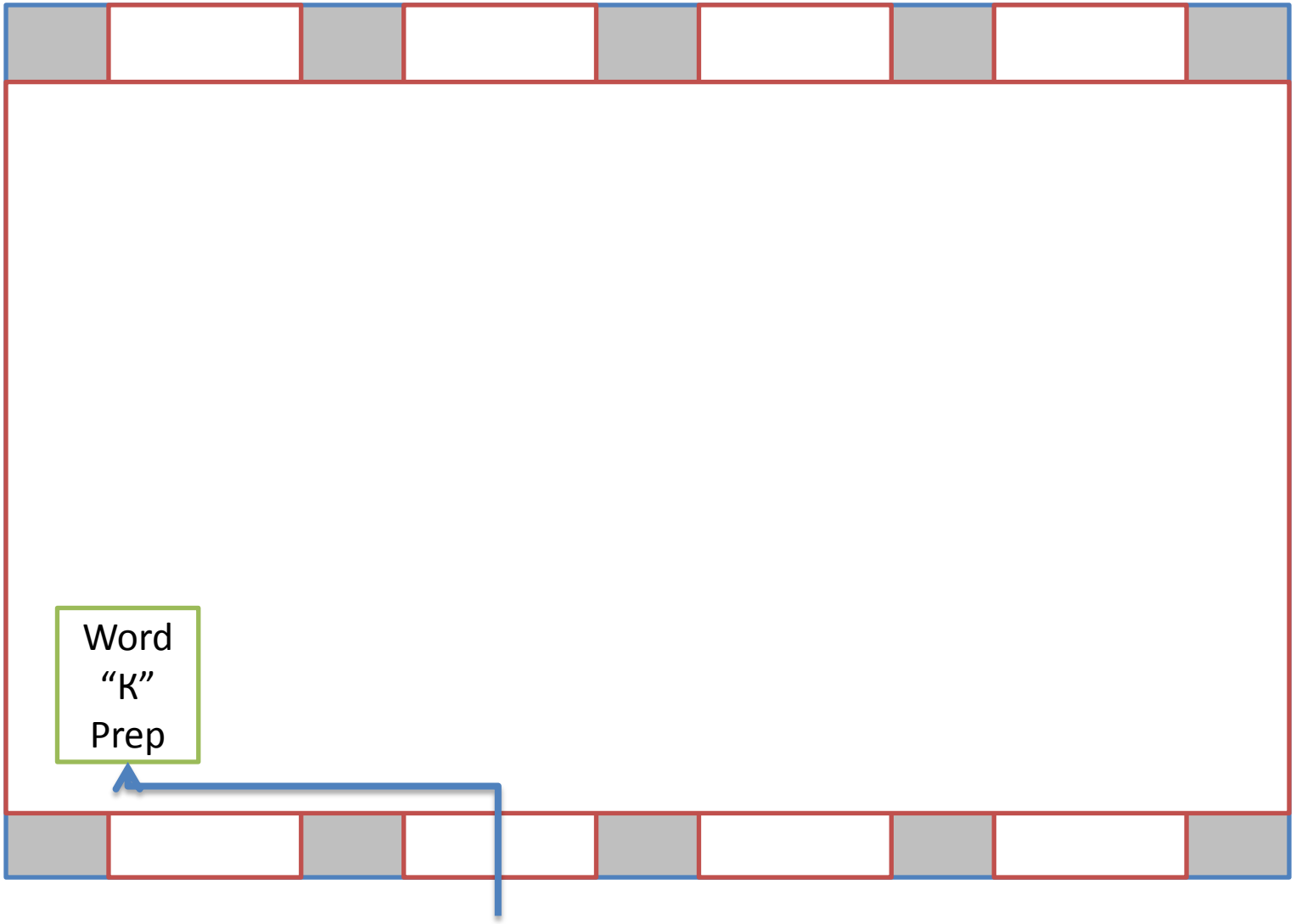
Морфологический анализ (1)



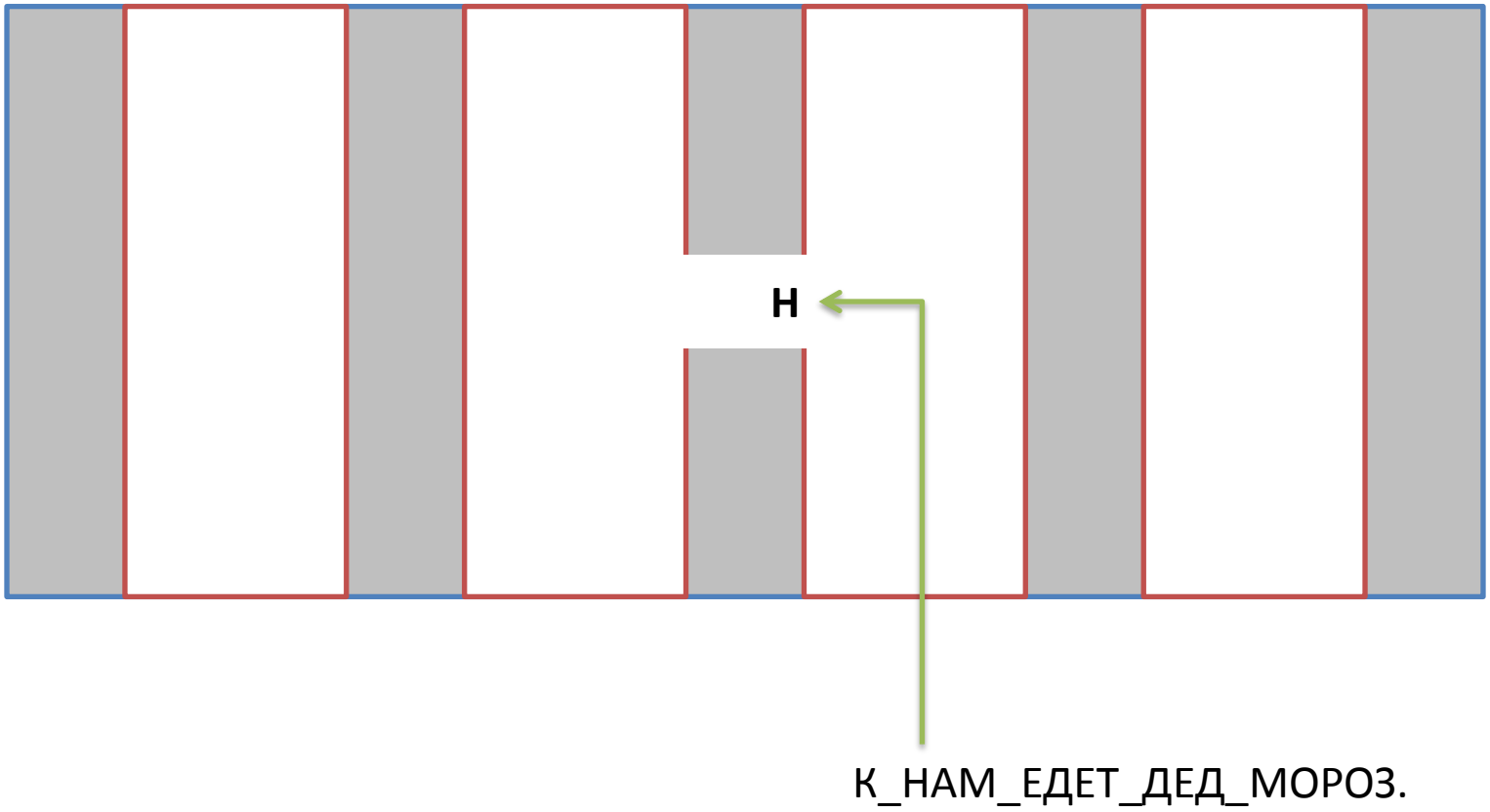
Извлечение именованных сущностей (1)



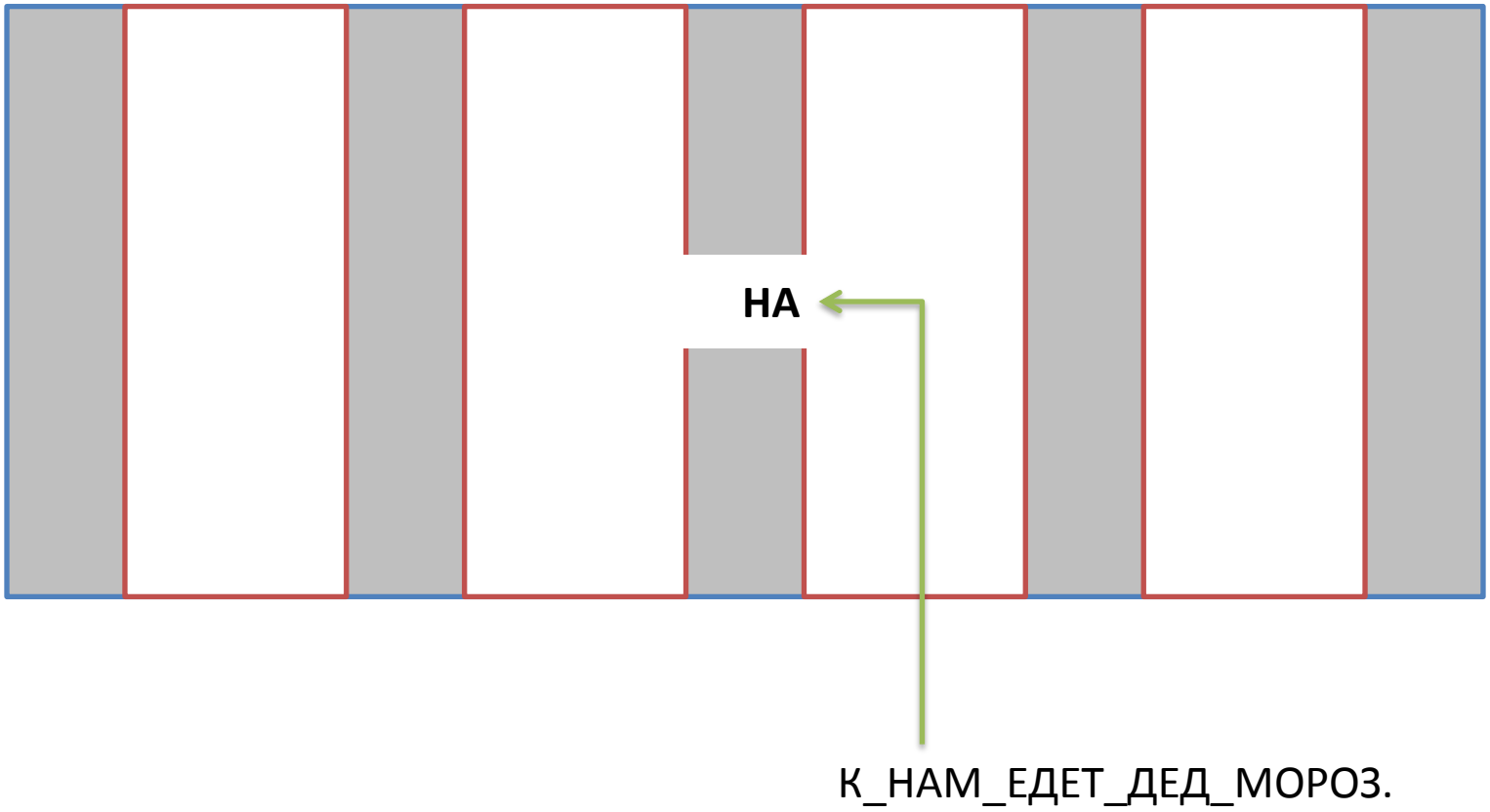
Синтаксический анализ (1)



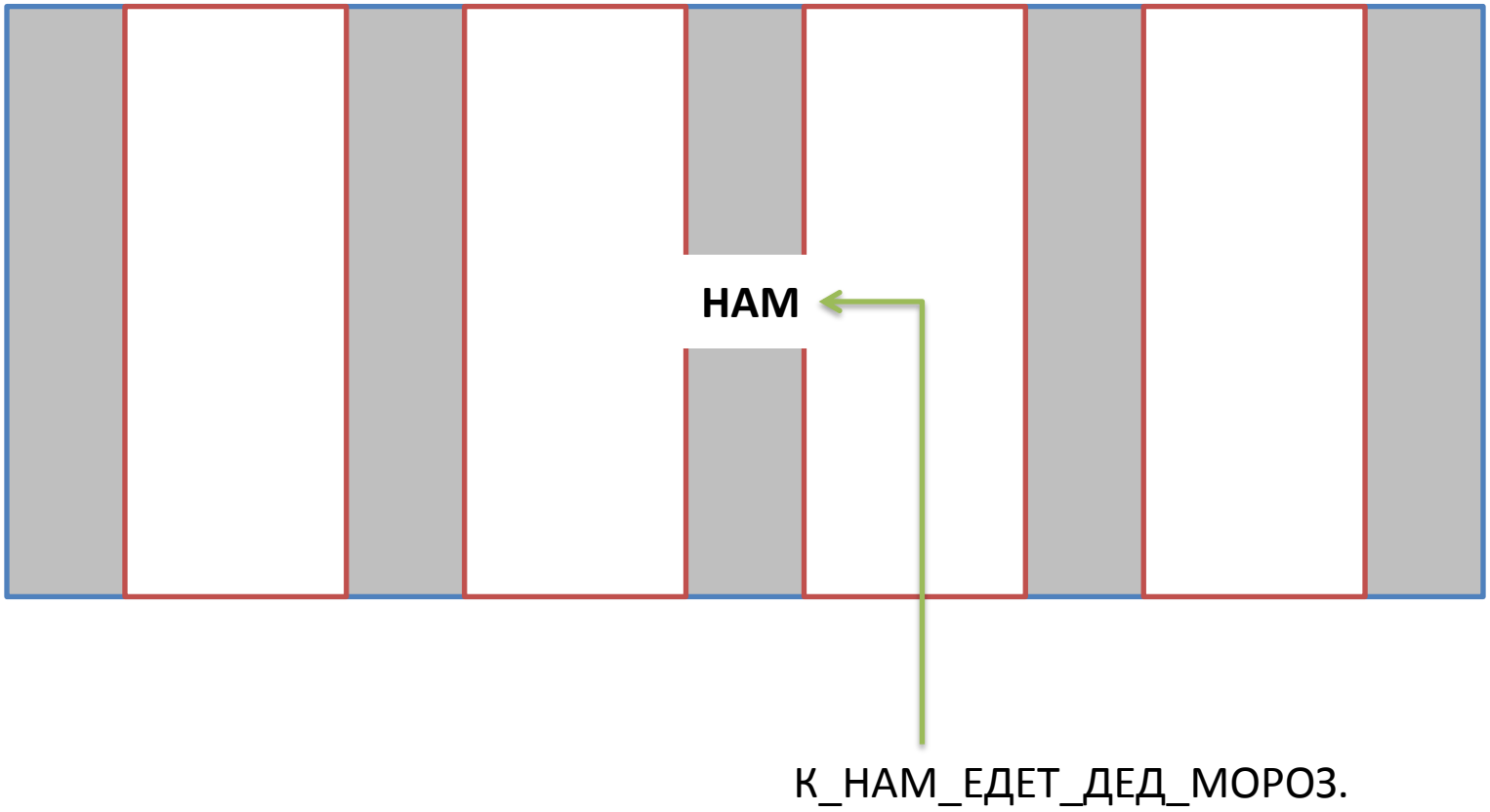
Токенизация (4)



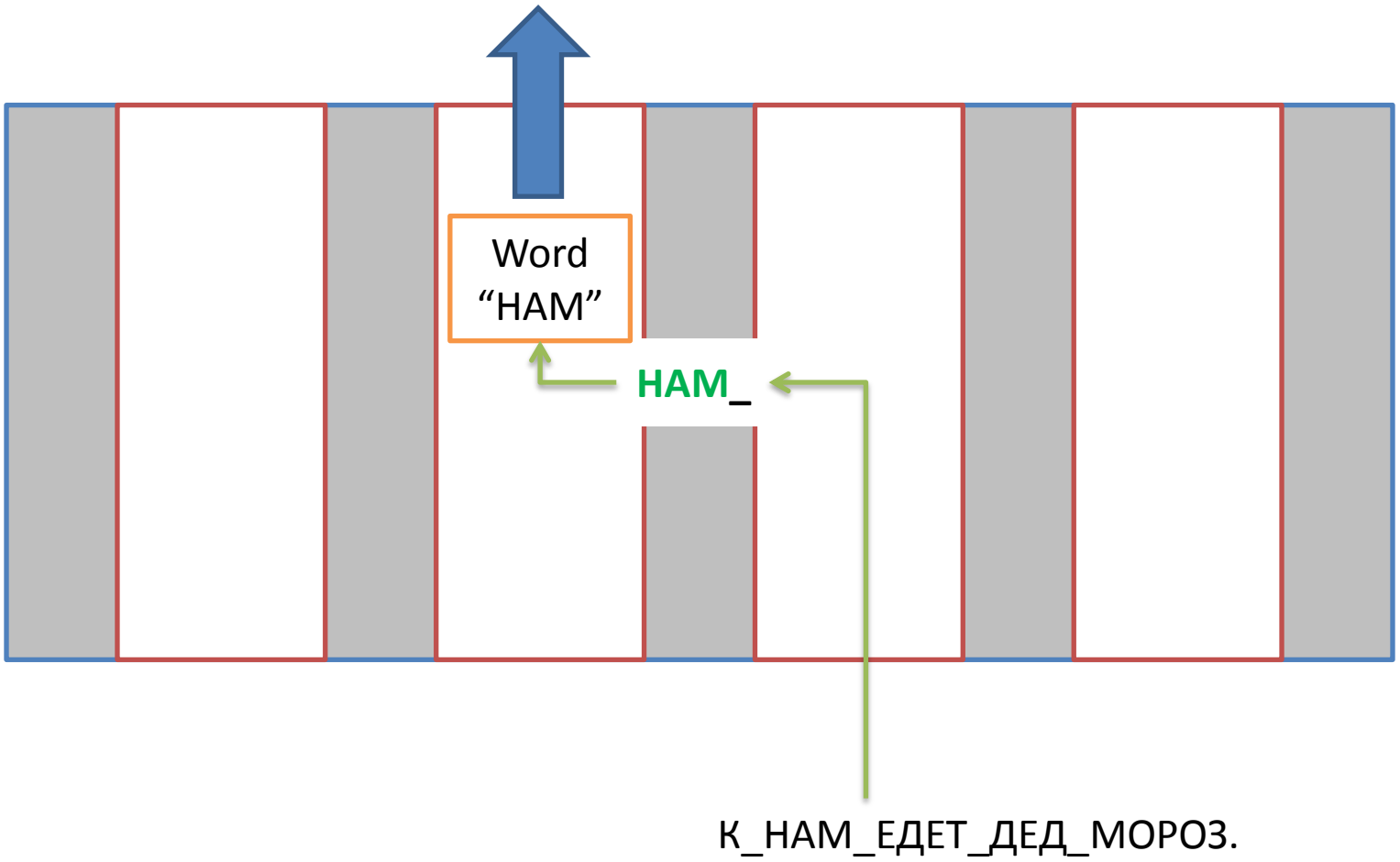
Токенизация (5)



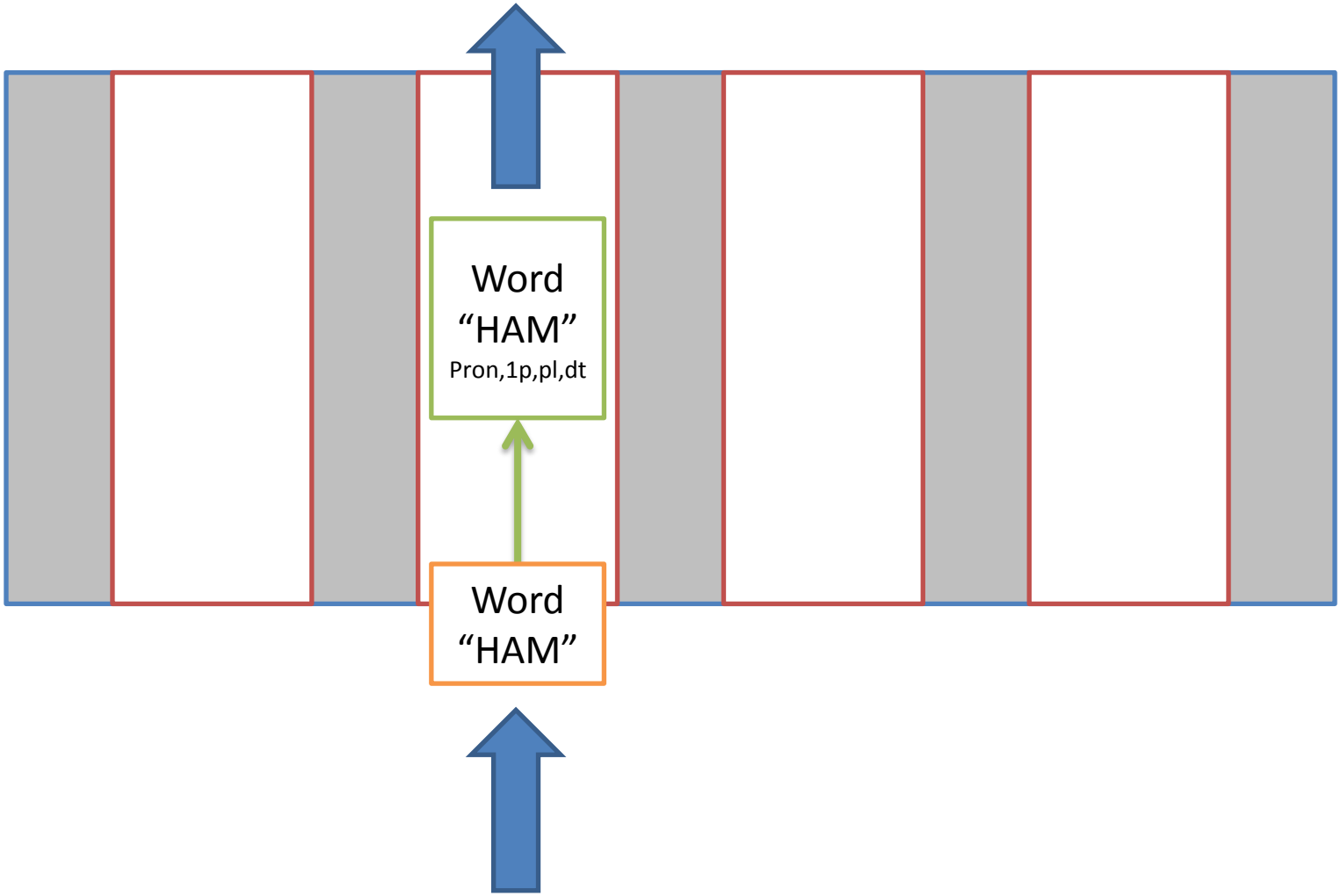
Токенизация (6)



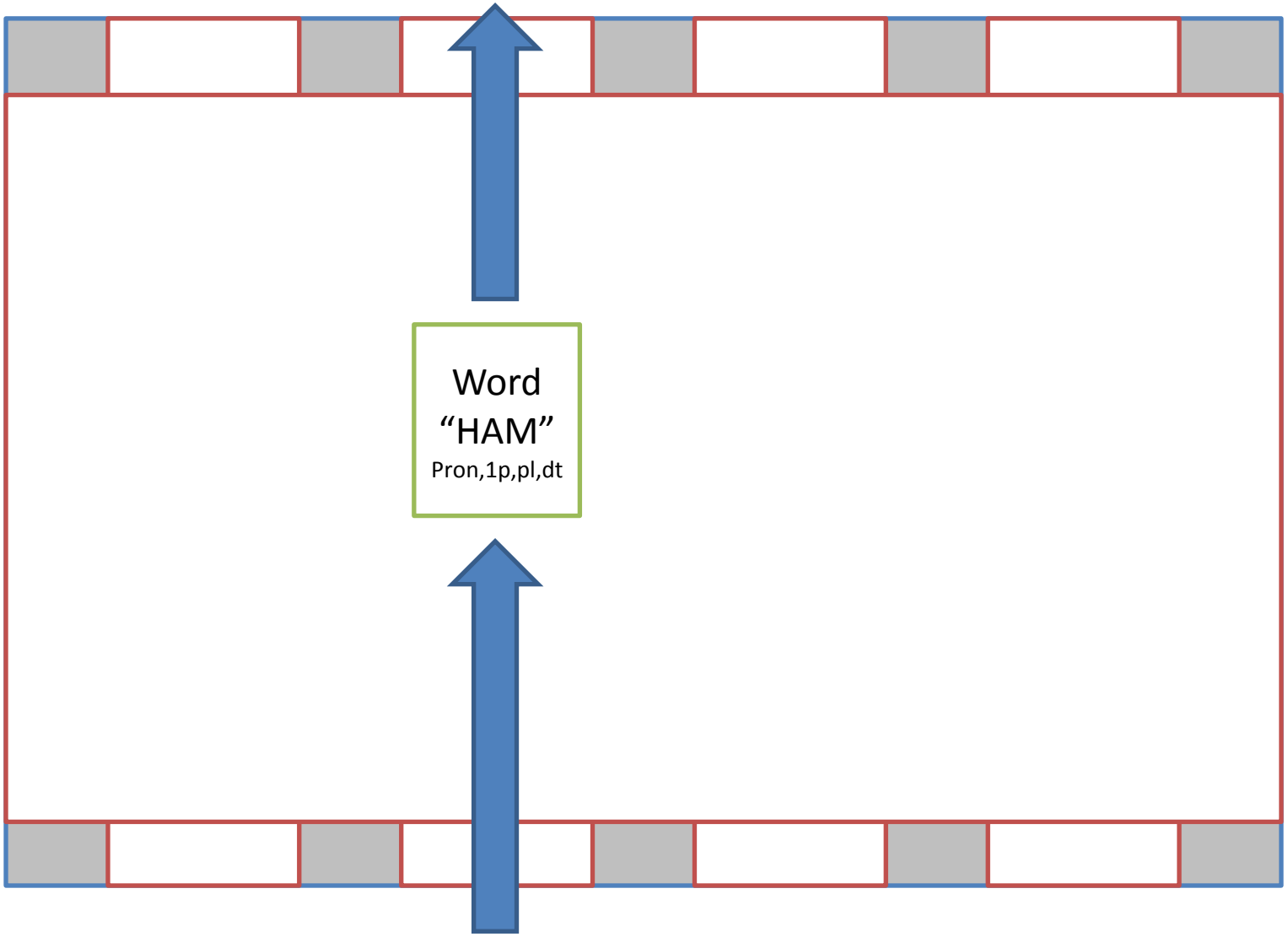
Токенизация (7)



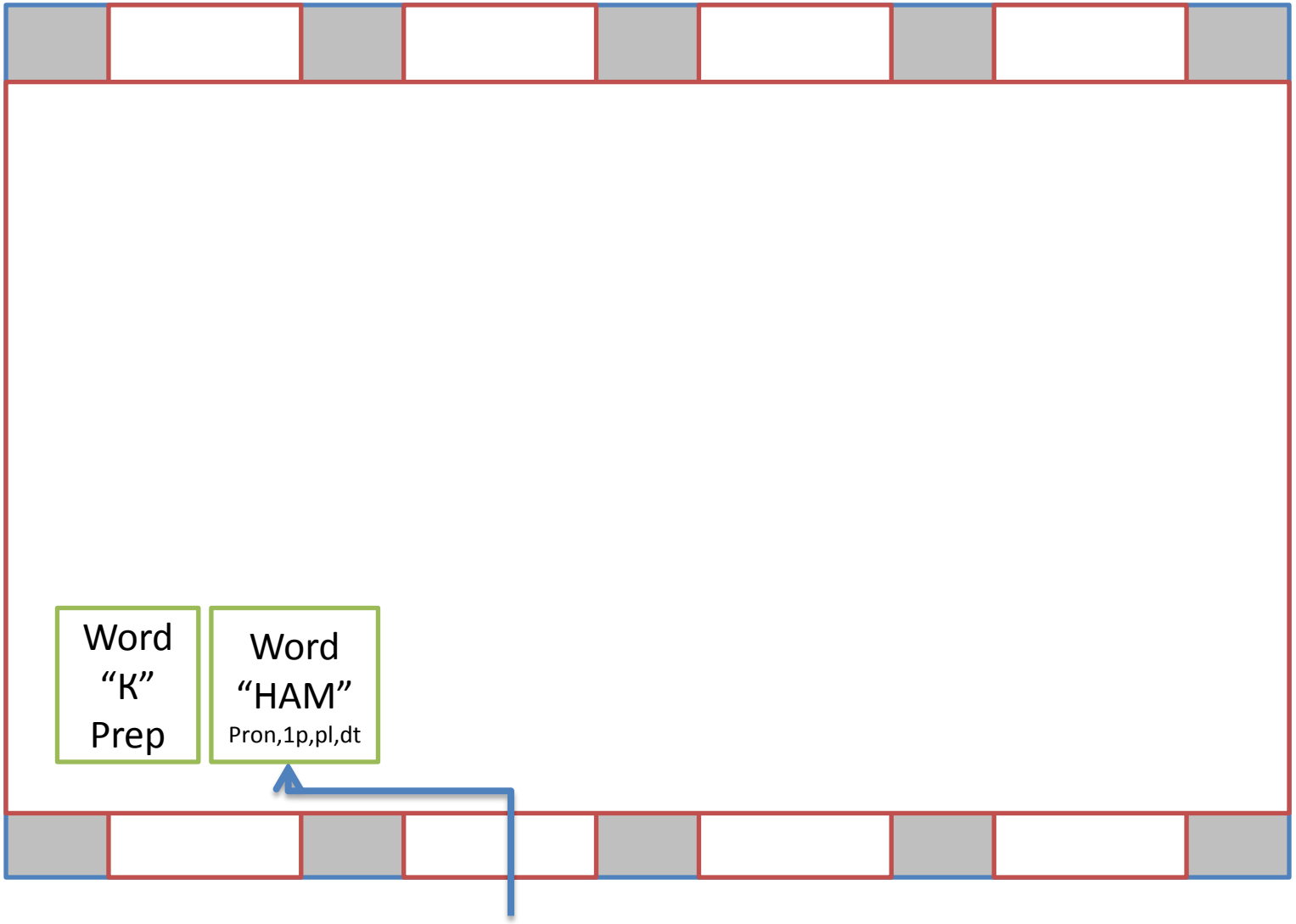
Морфологический анализ (2)



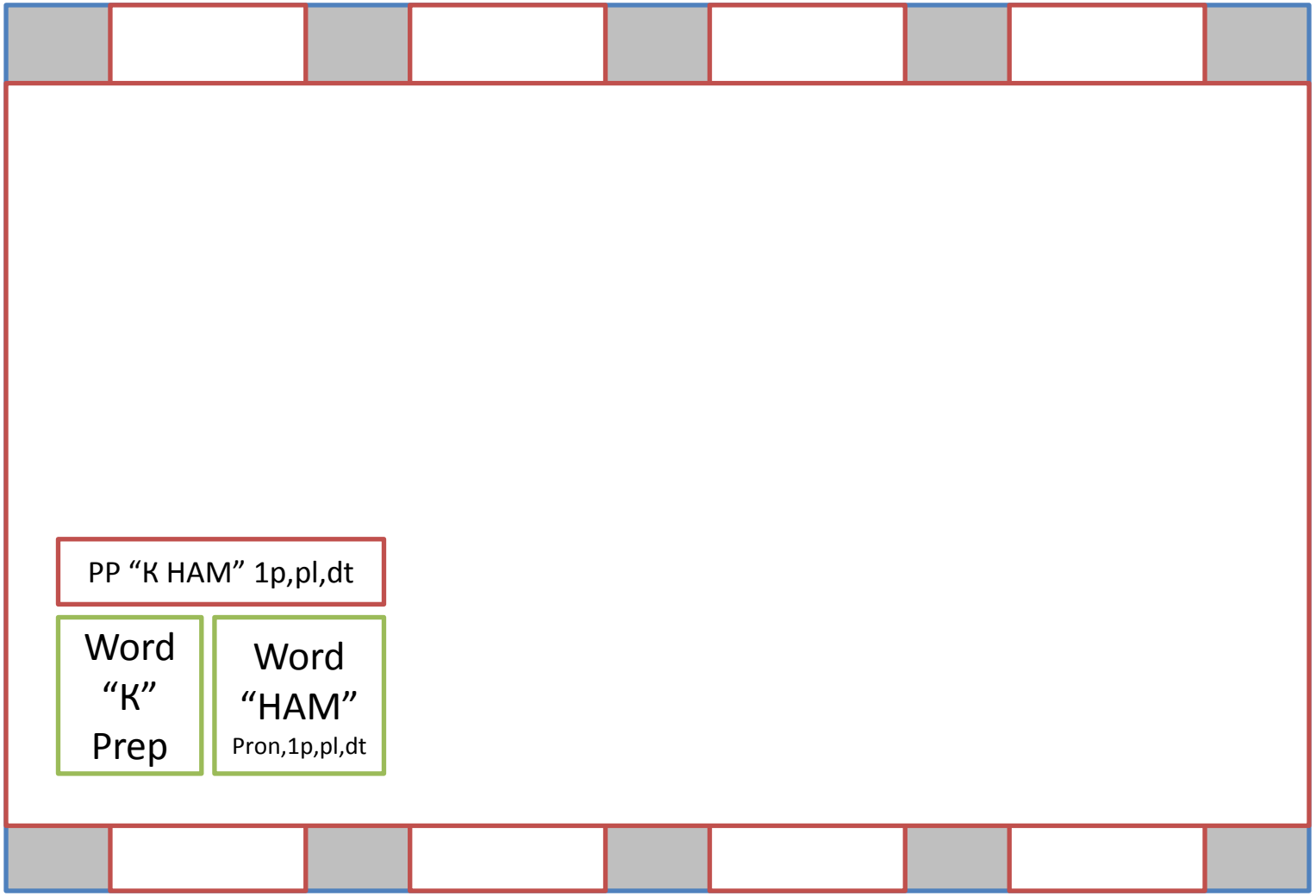
Извлечение именованных сущностей (2)



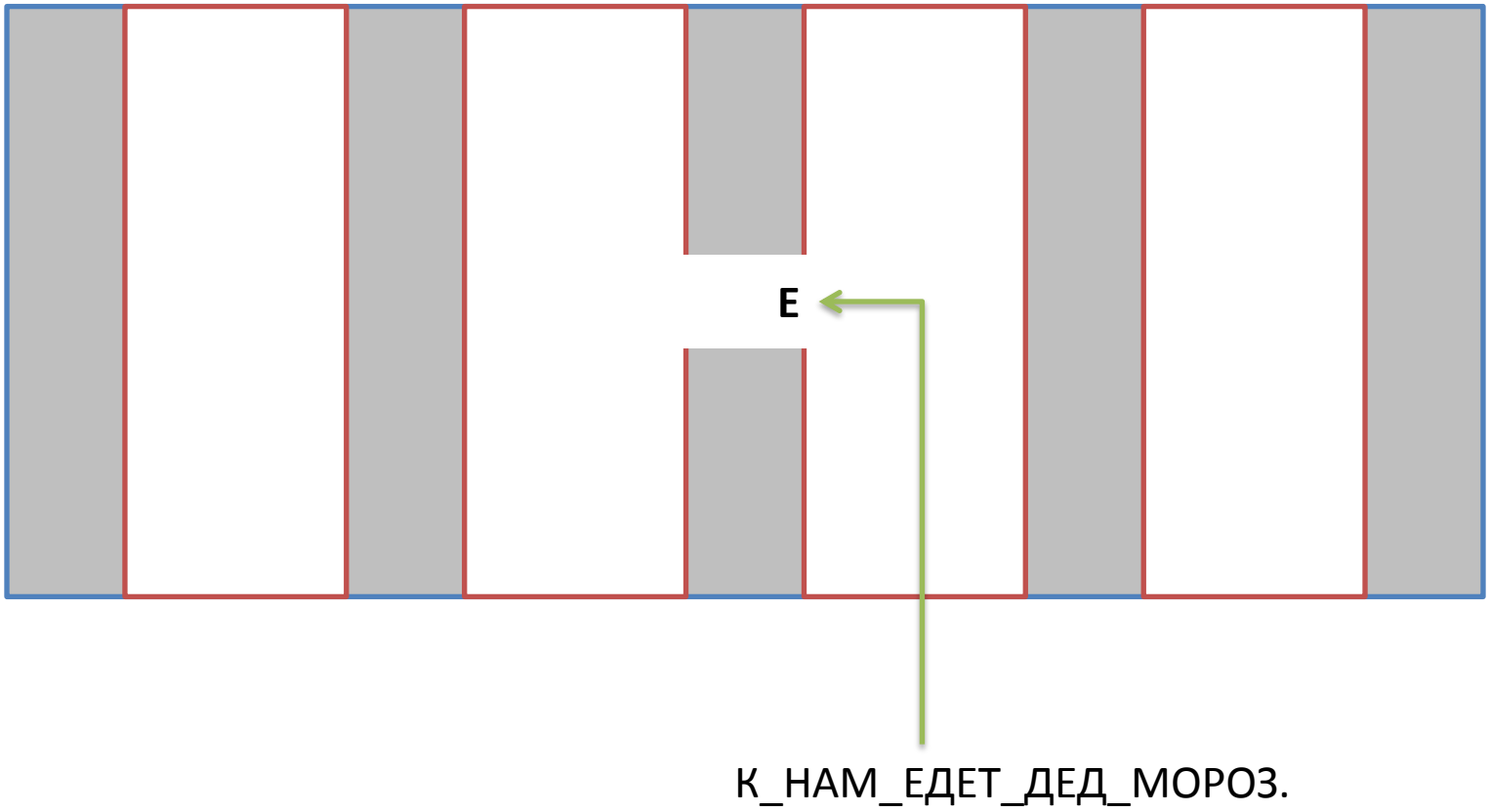
Синтаксический анализ (2)



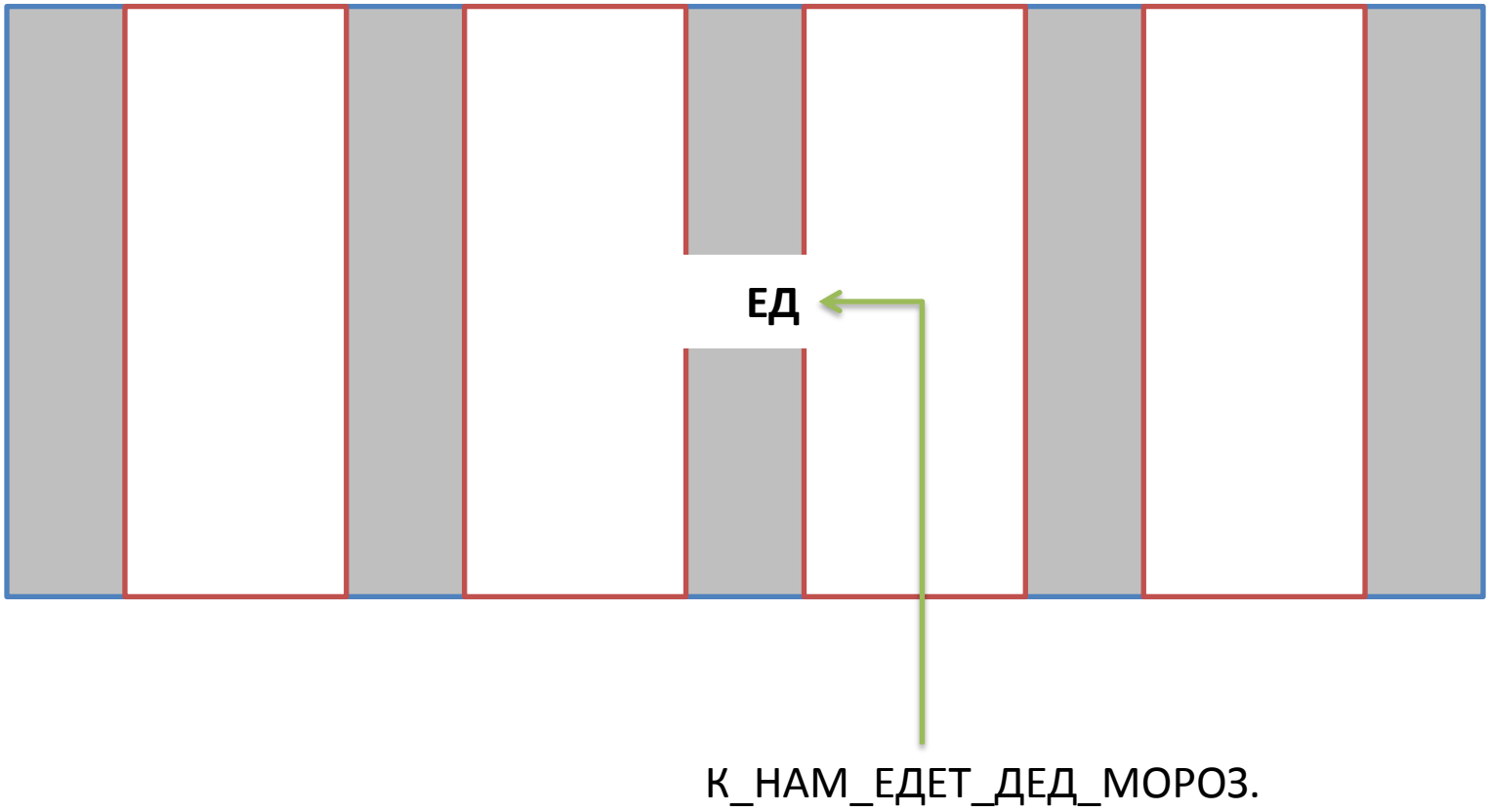
Синтаксический анализ (3)



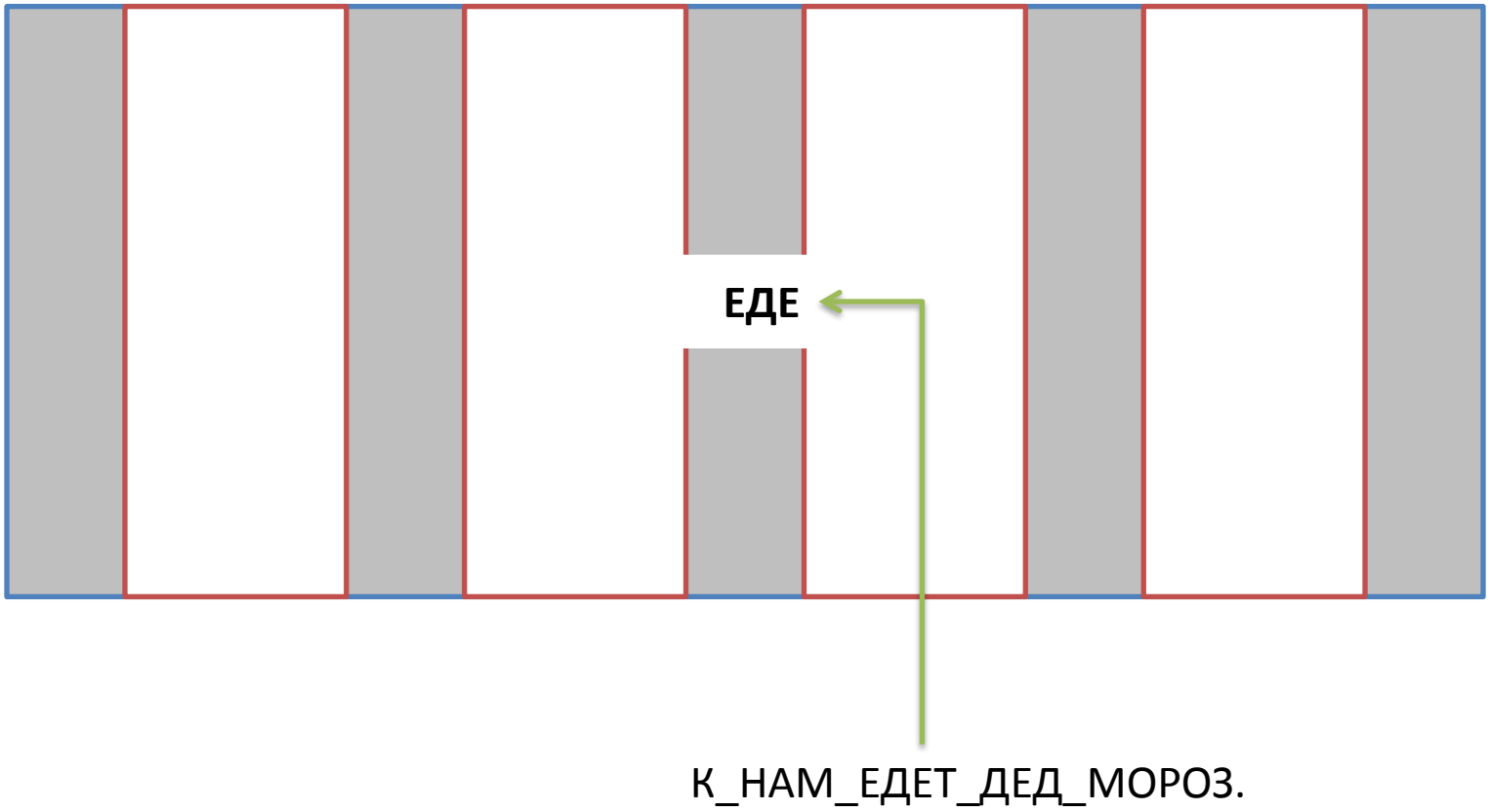
Токенизация (8)



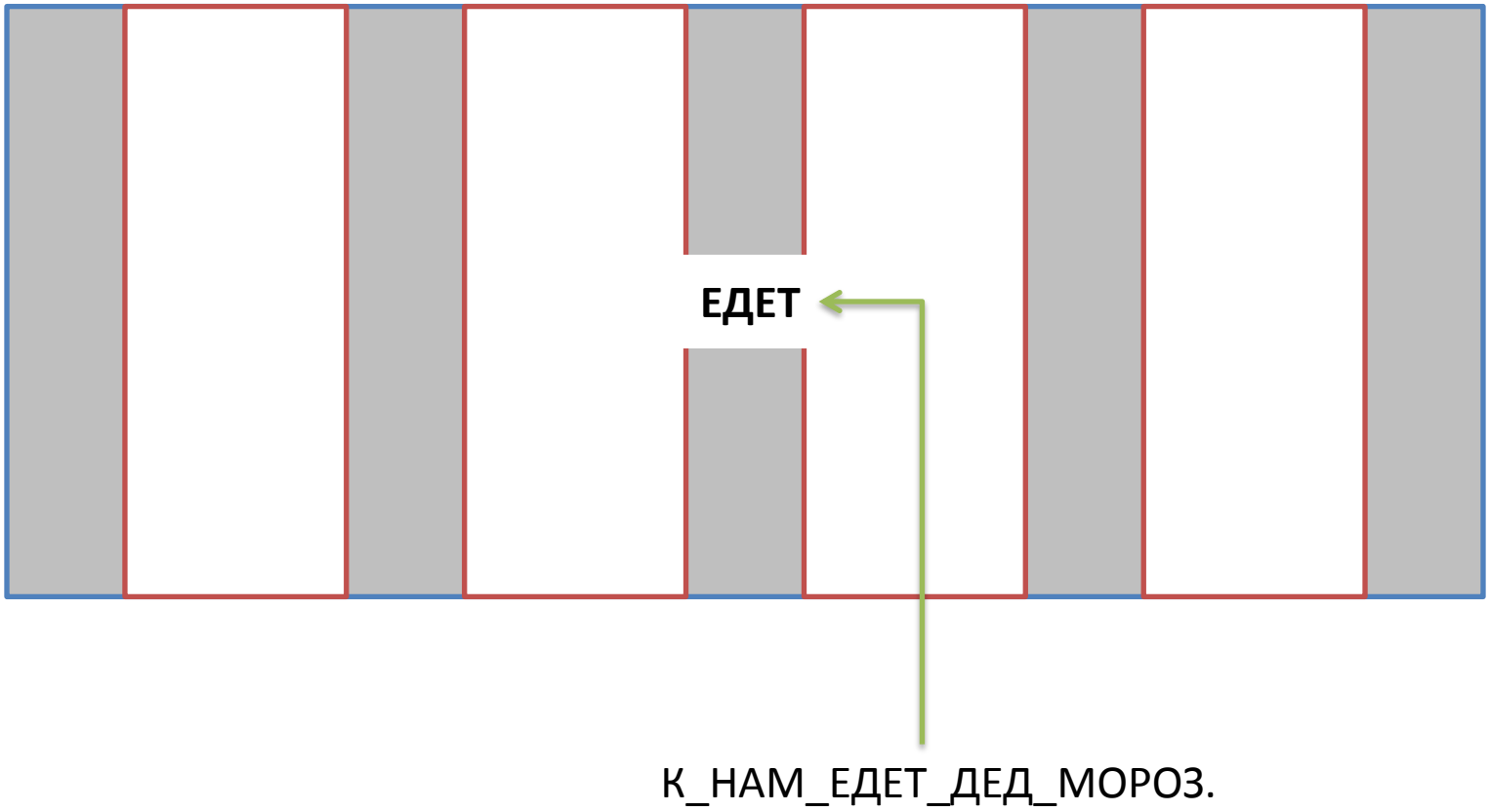
Токенизация (9)



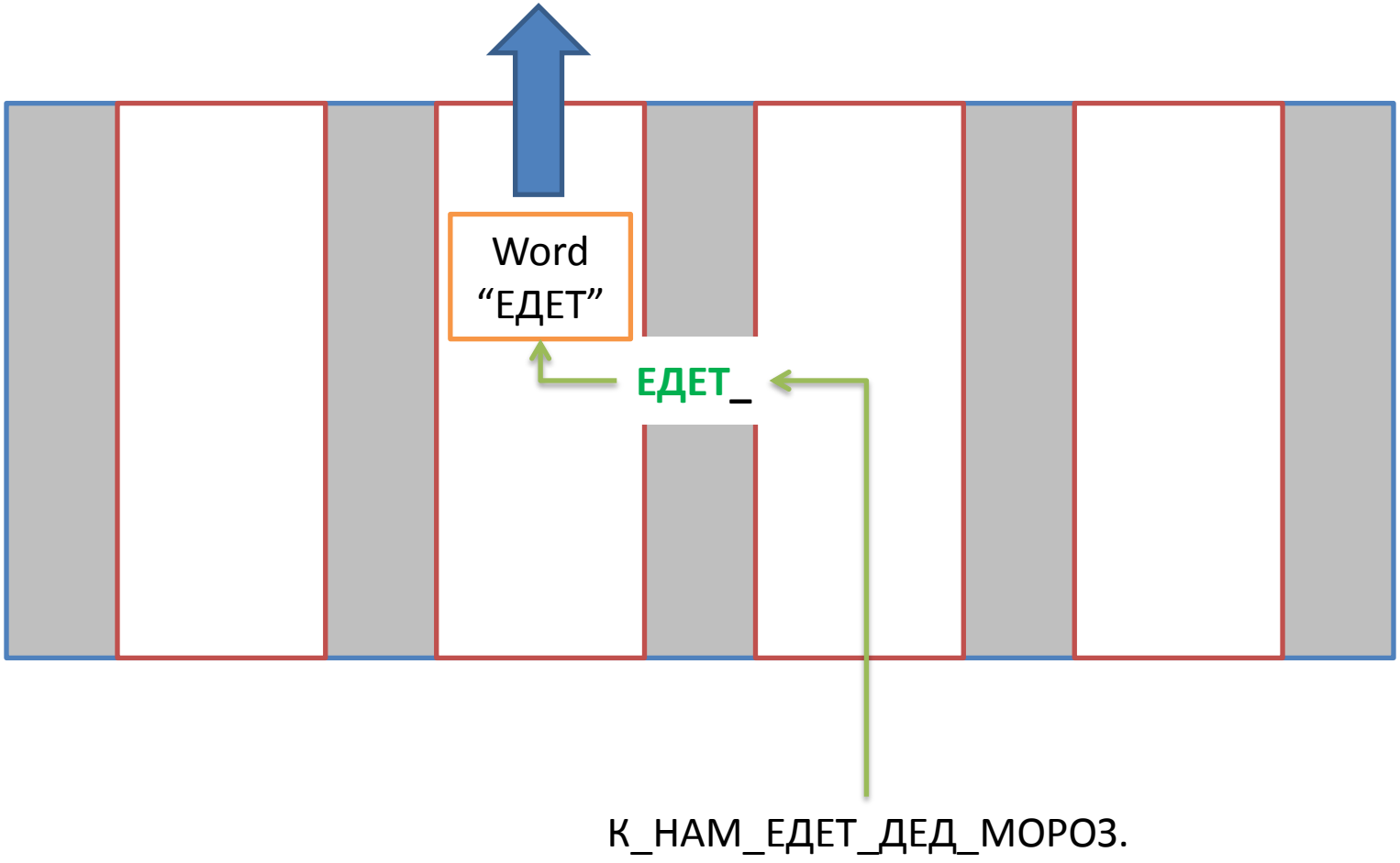
Токенизация (10)



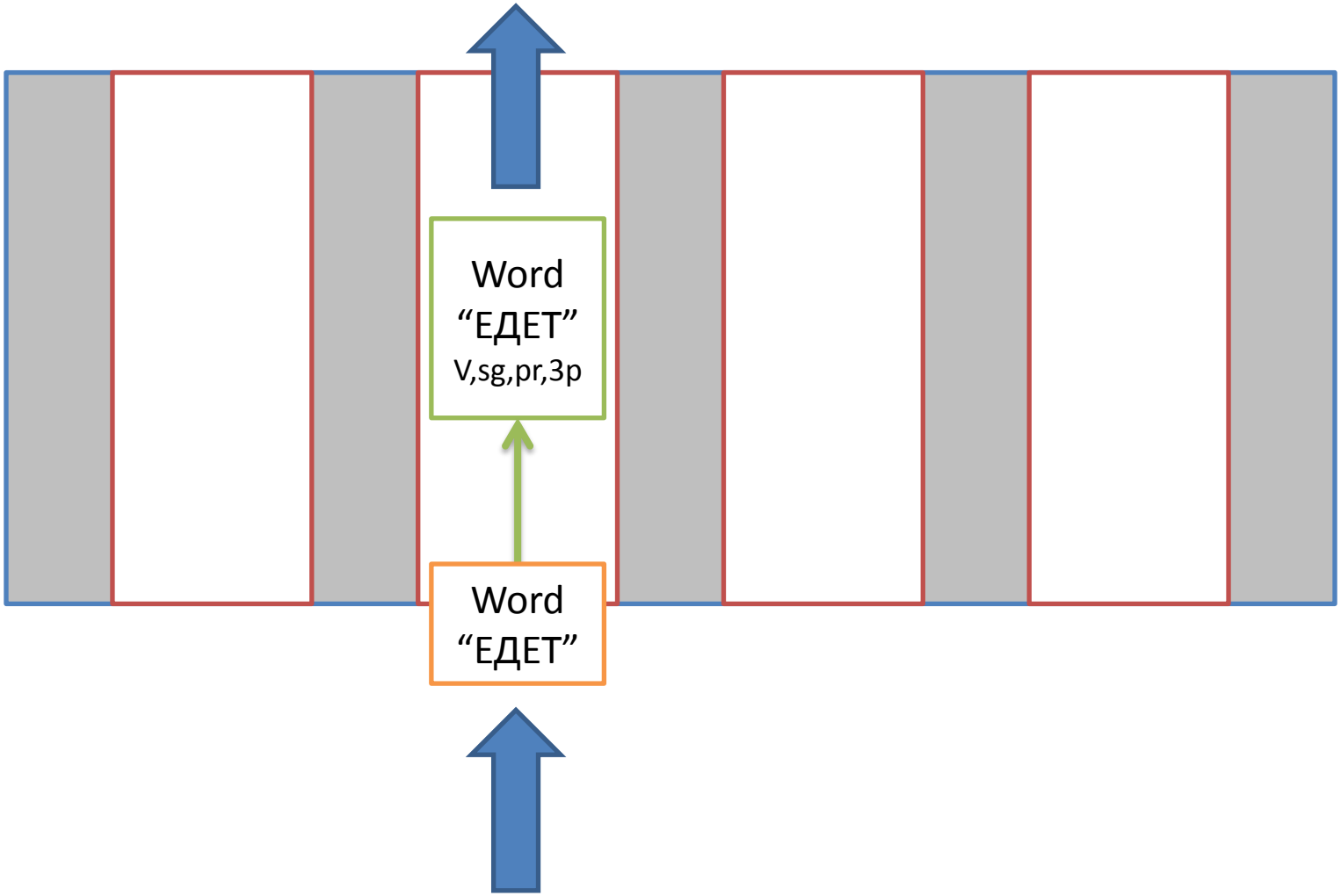
Токенизация (11)



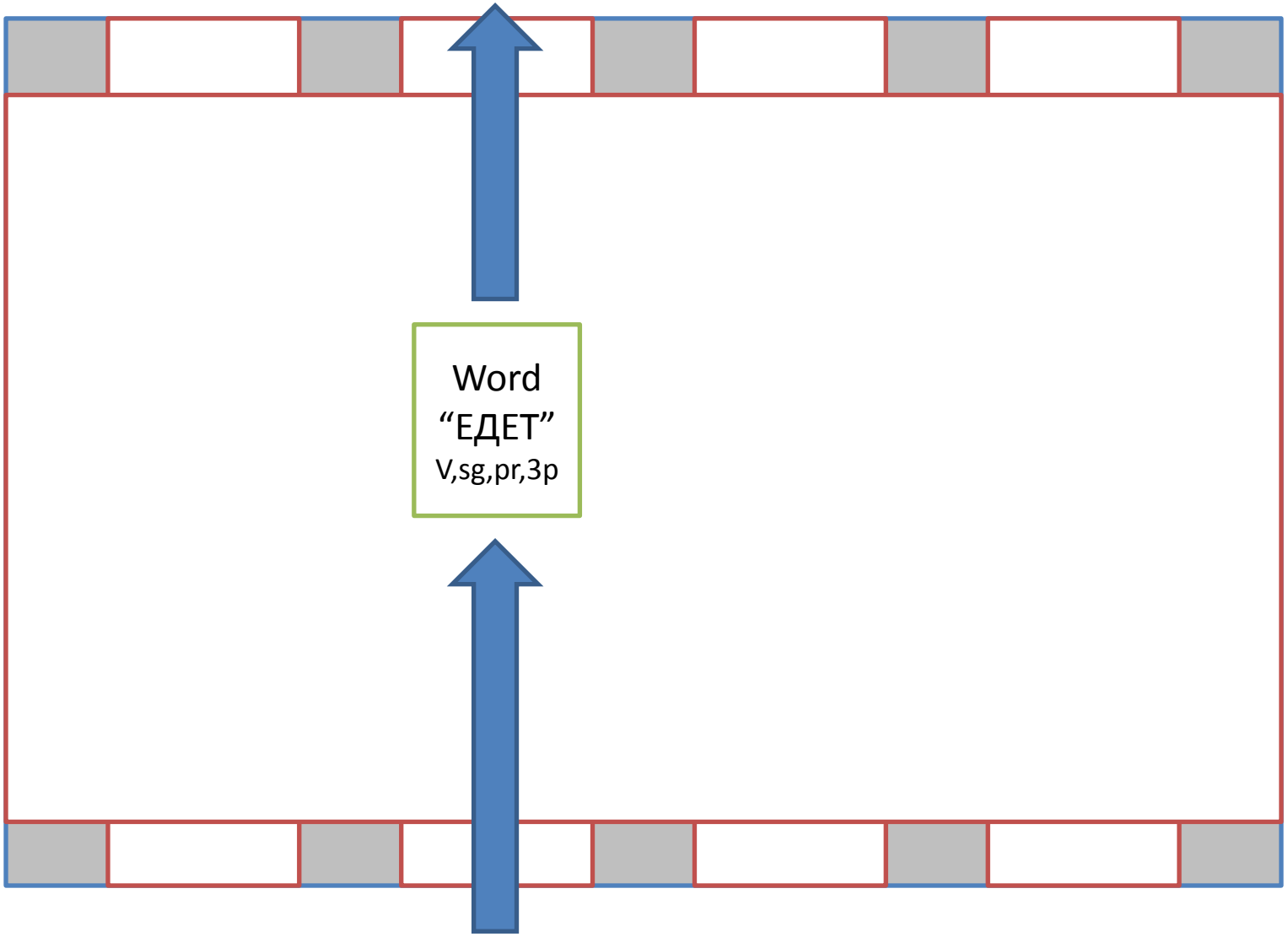
Токенизация (12)



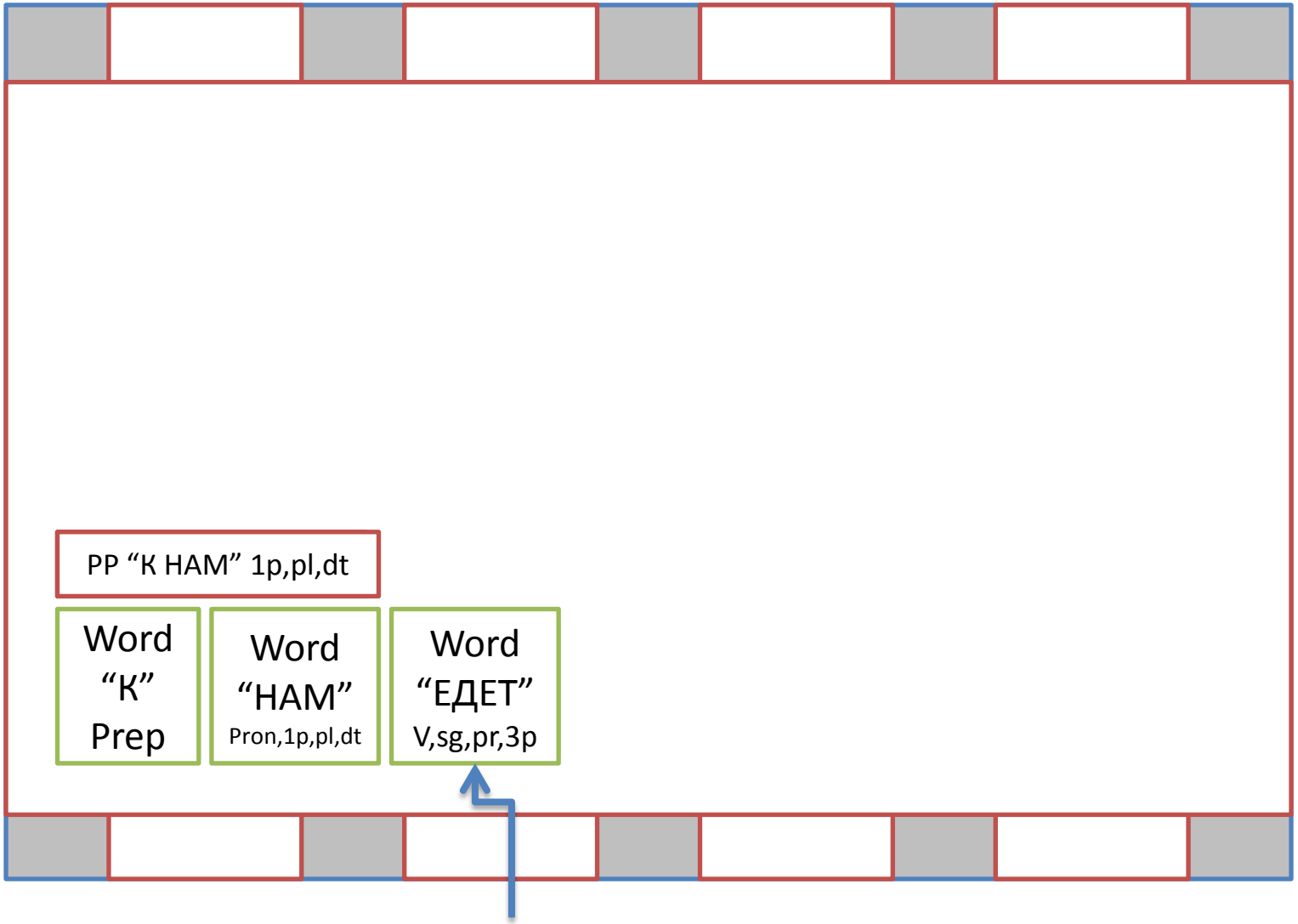
Морфологический анализ (3)



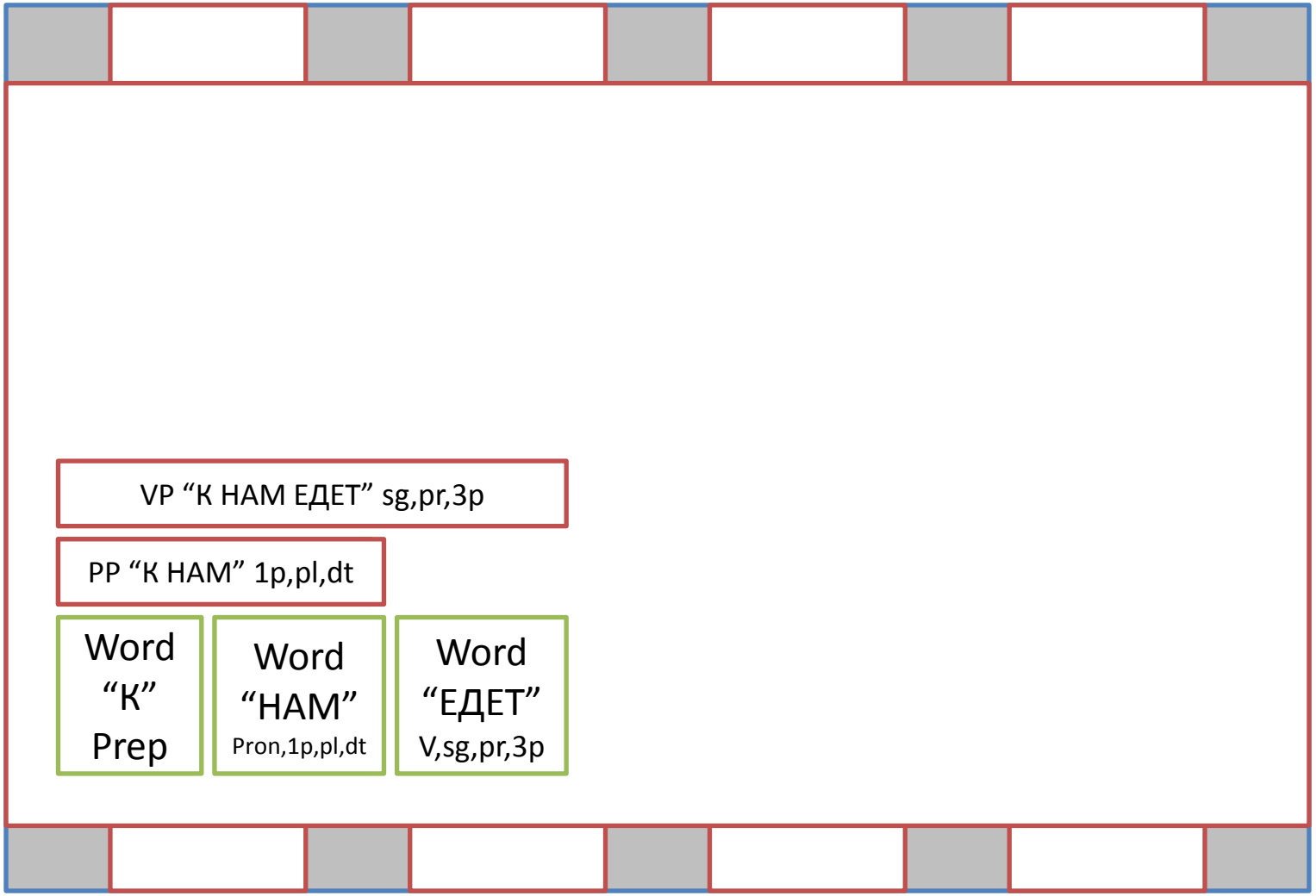
Извлечение именованных сущностей (3)



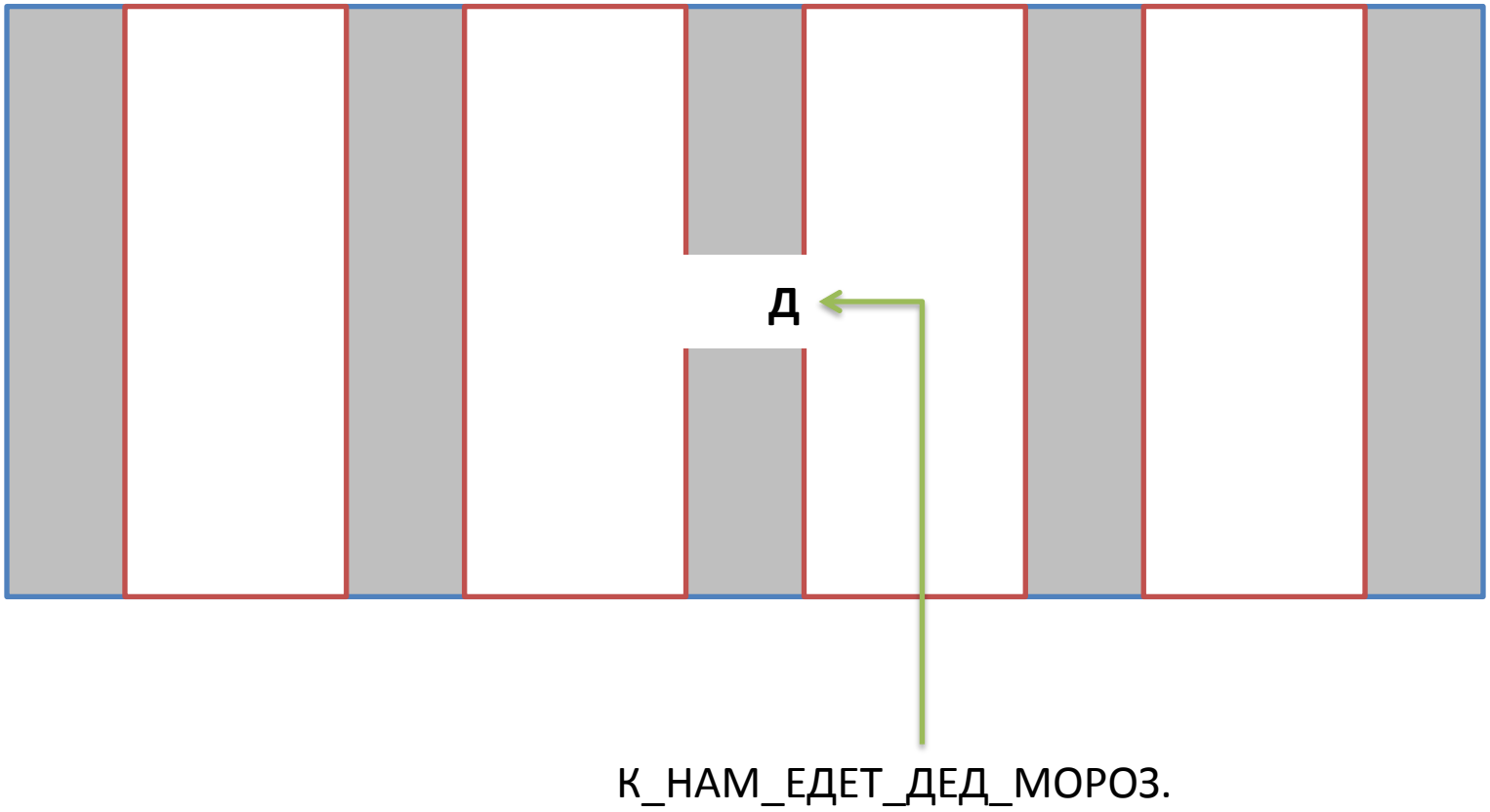
Синтаксический анализ (4)



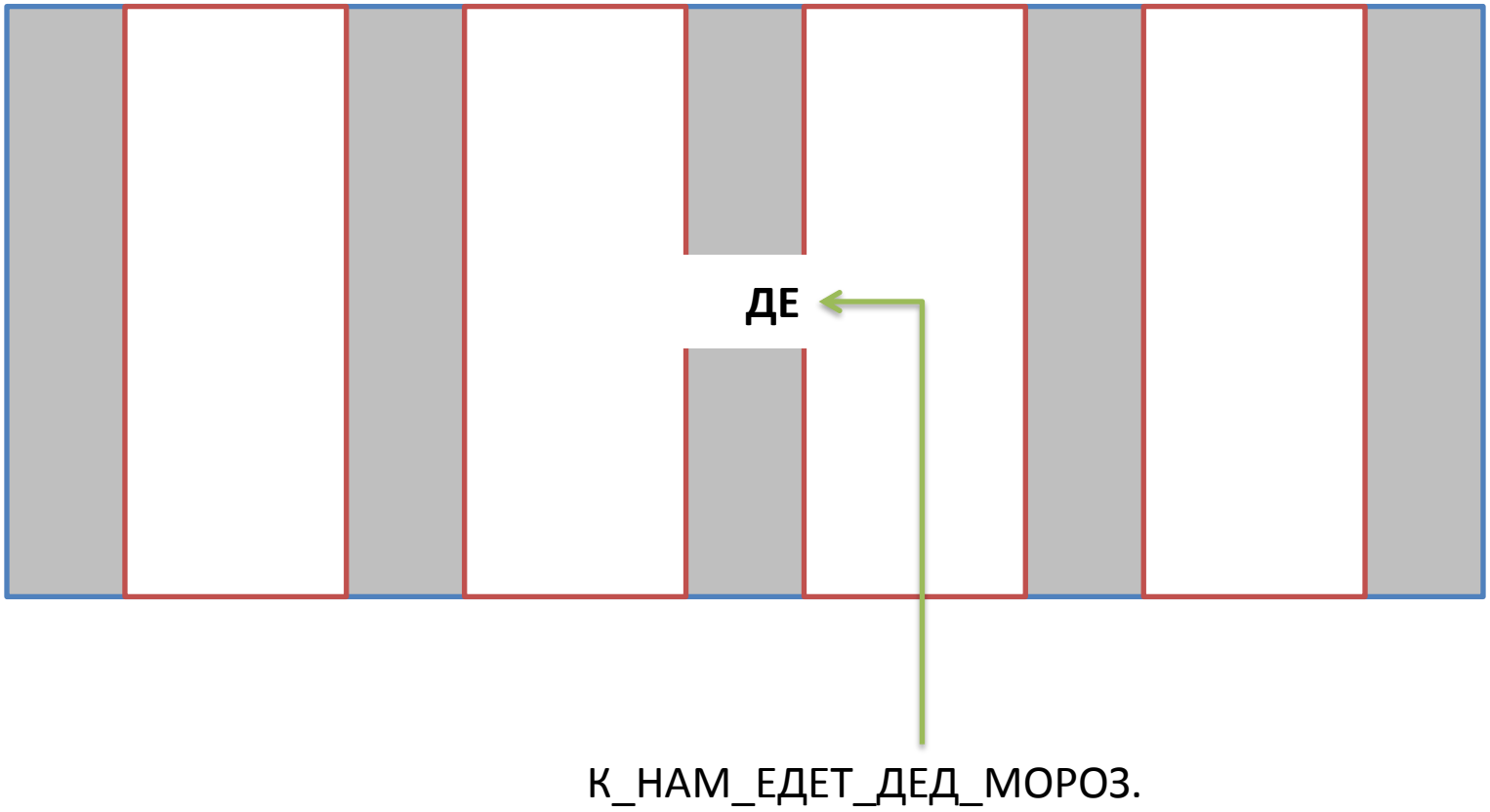
Синтаксический анализ (5)



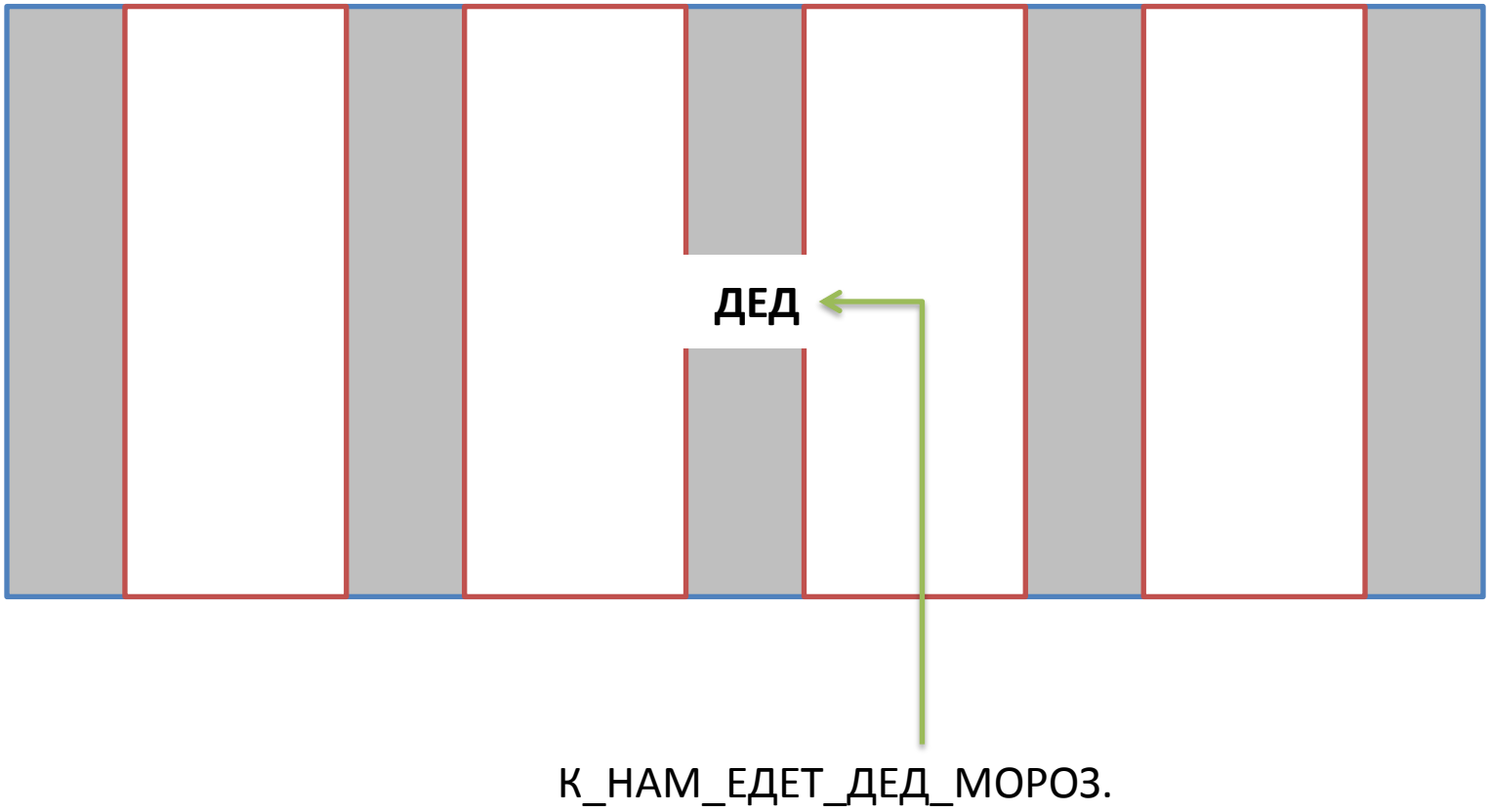
Токенизация (13)



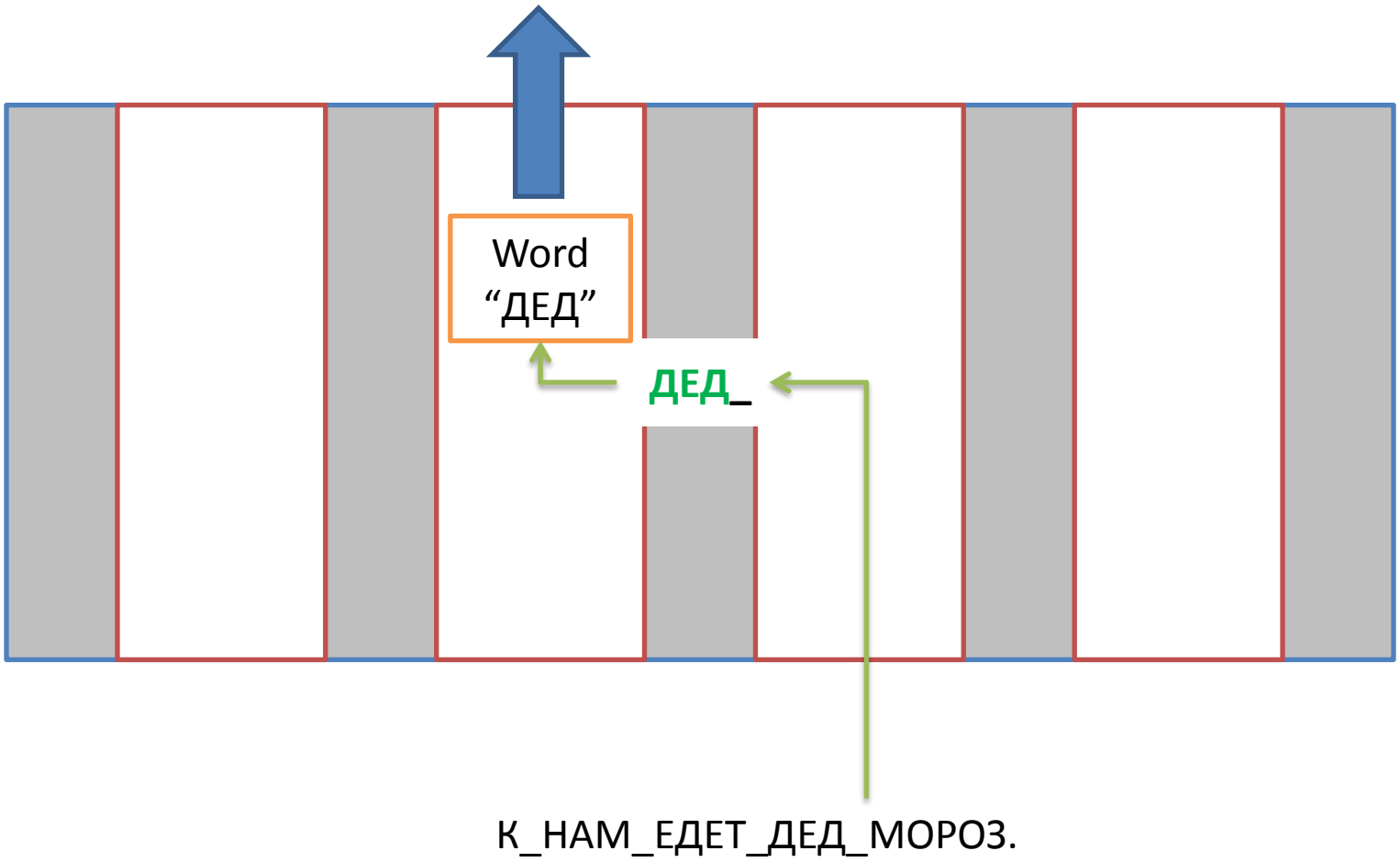
Токенизация (14)



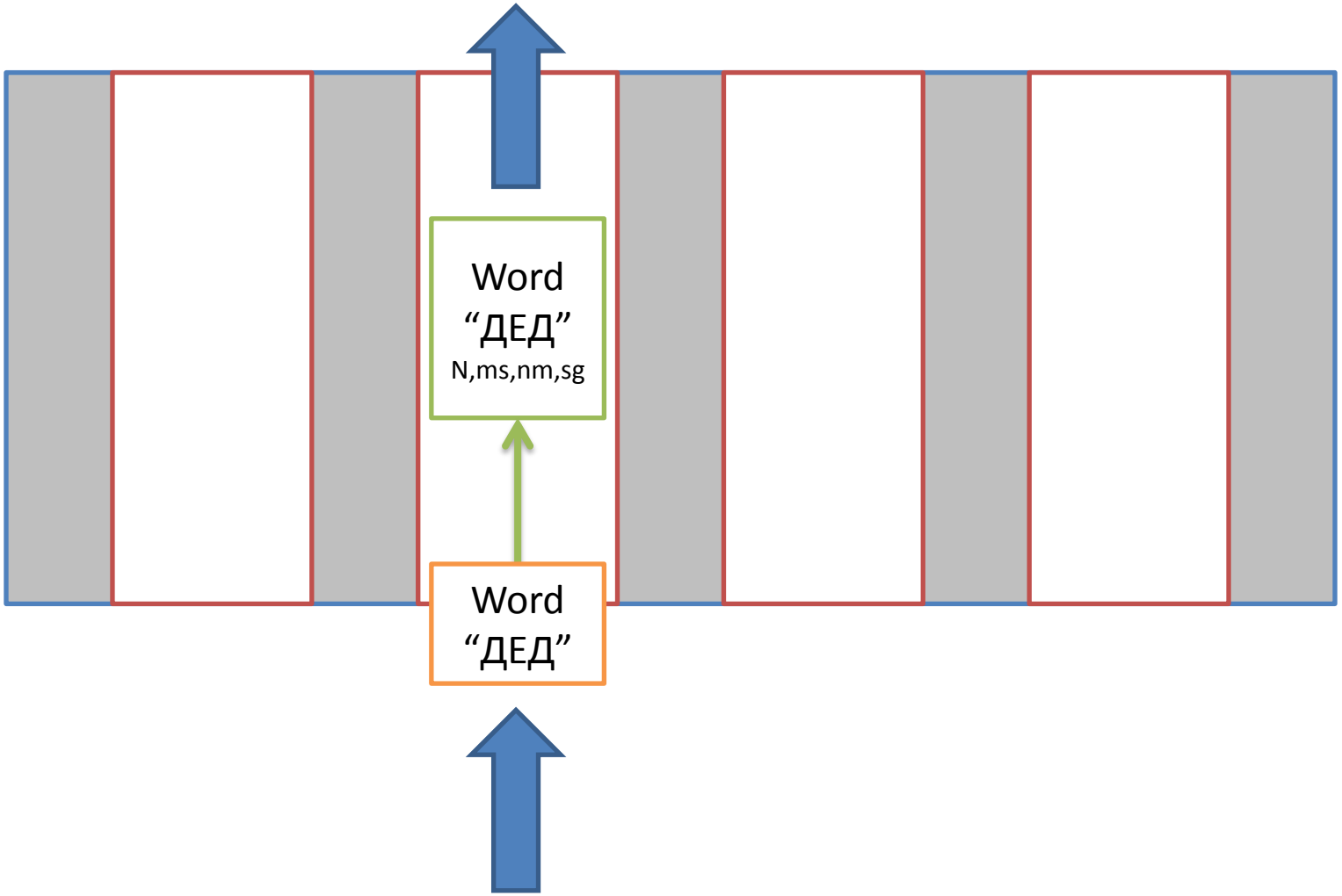
Токенизация (15)



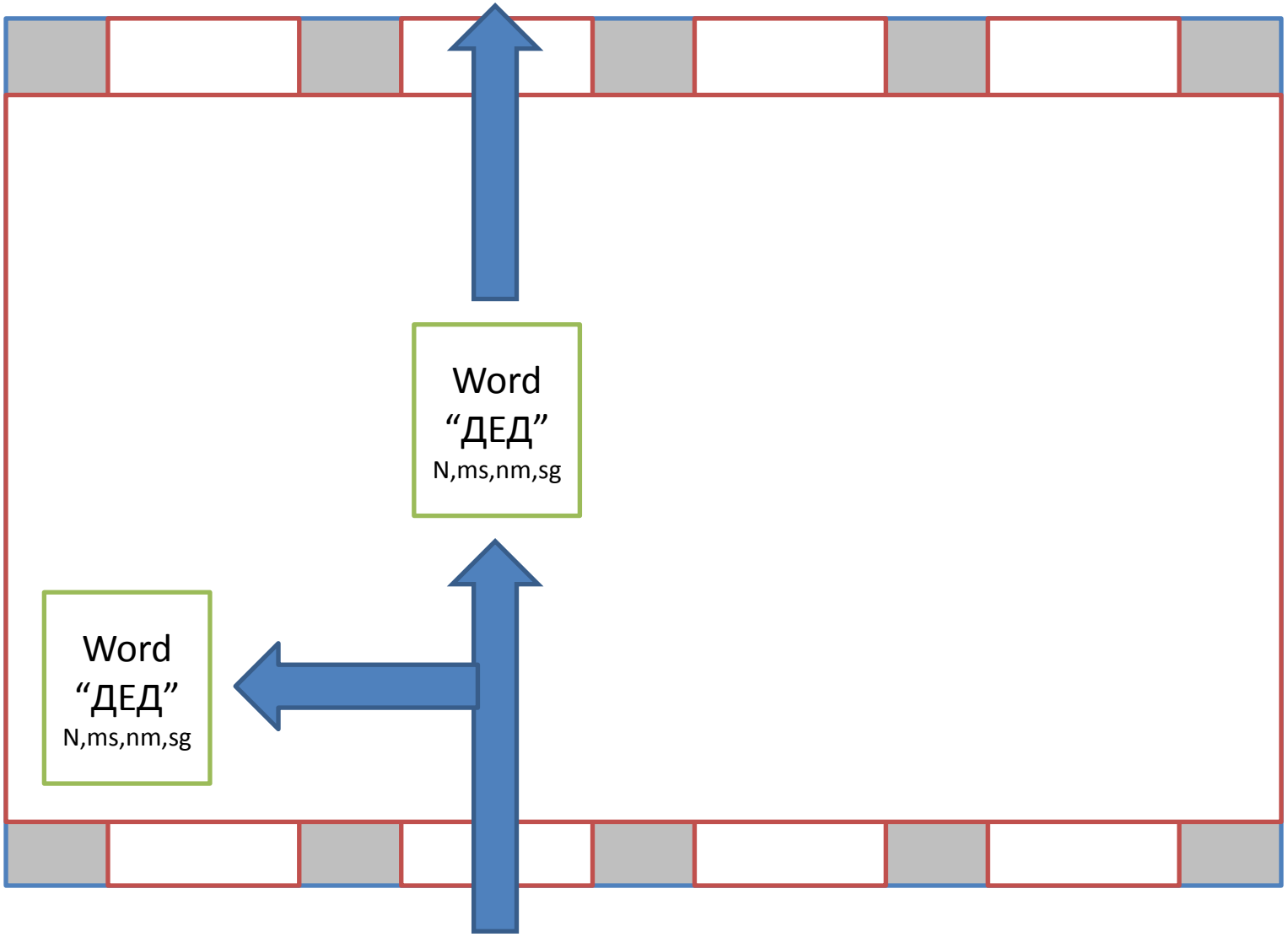
Токенизация (16)



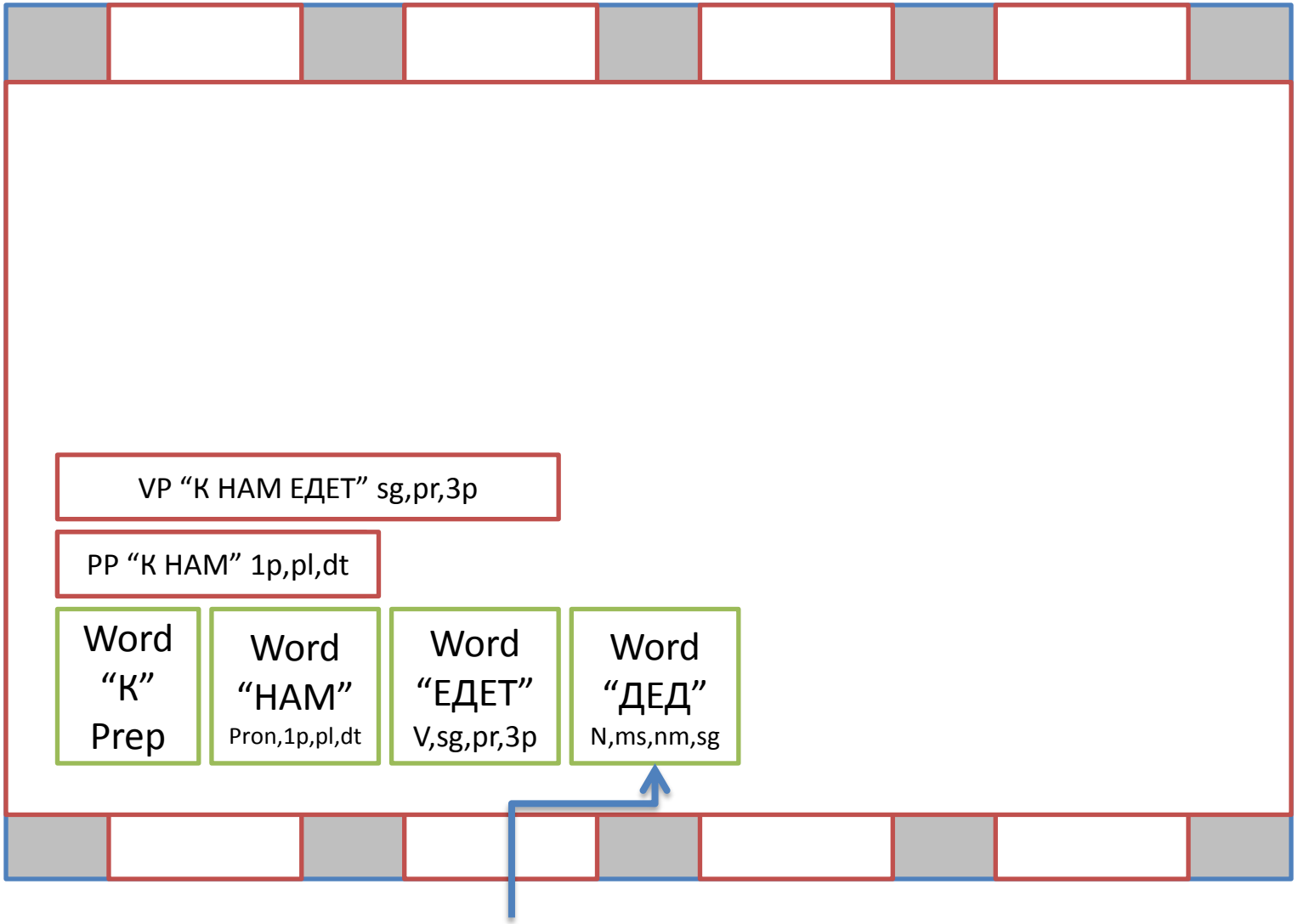
Морфологический анализ (4)



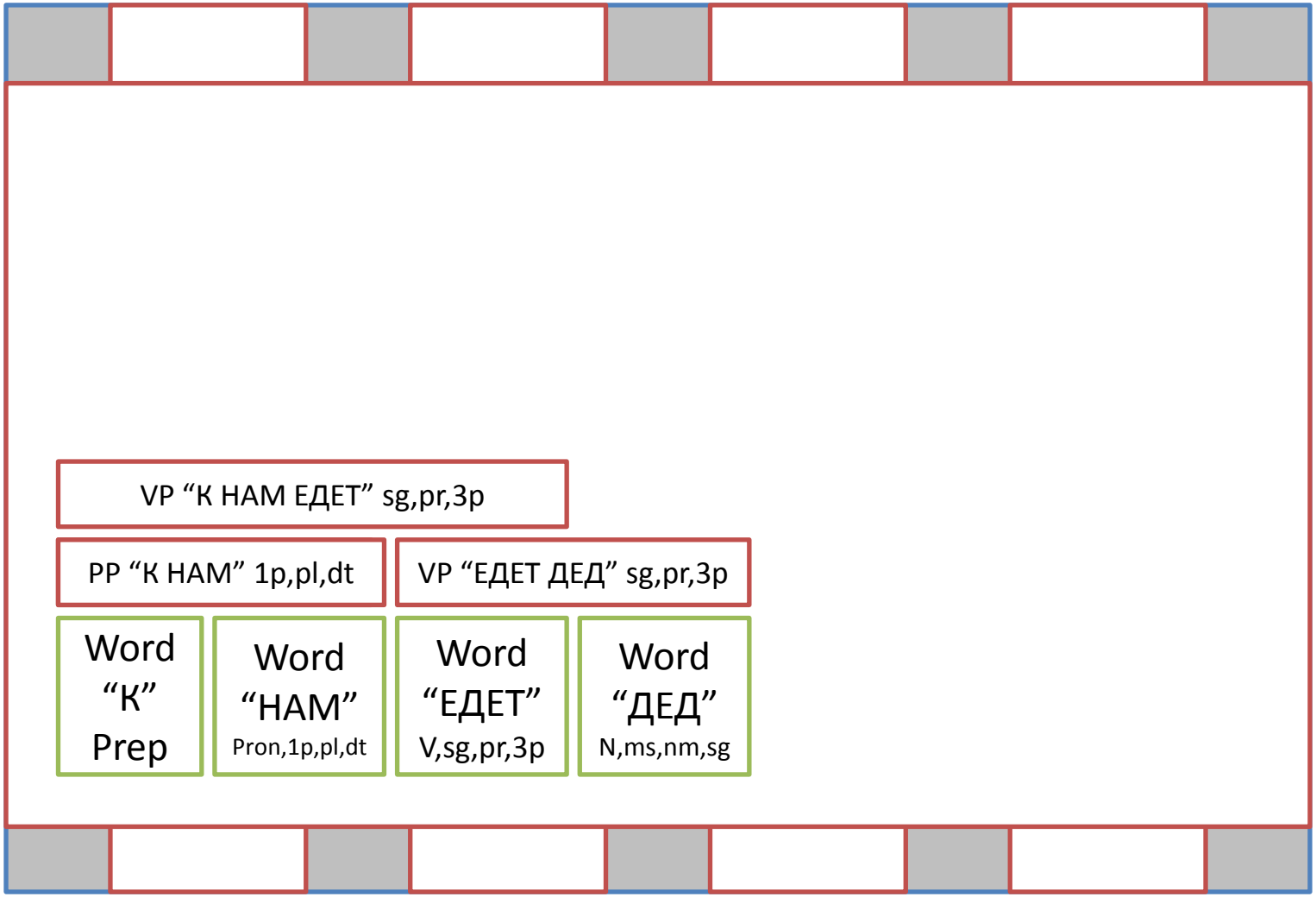
Извлечение именованных сущностей (4)



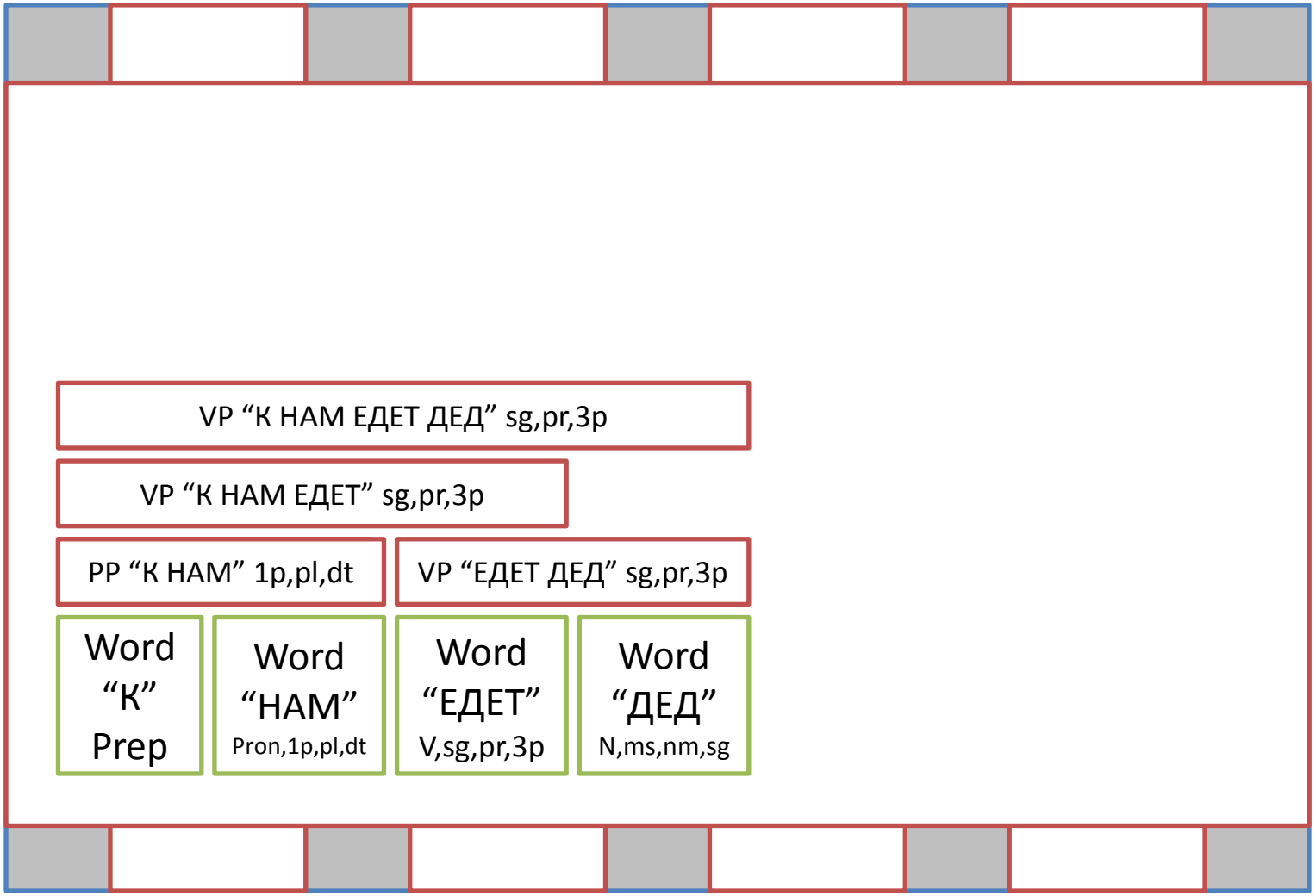
Синтаксический анализ (6)



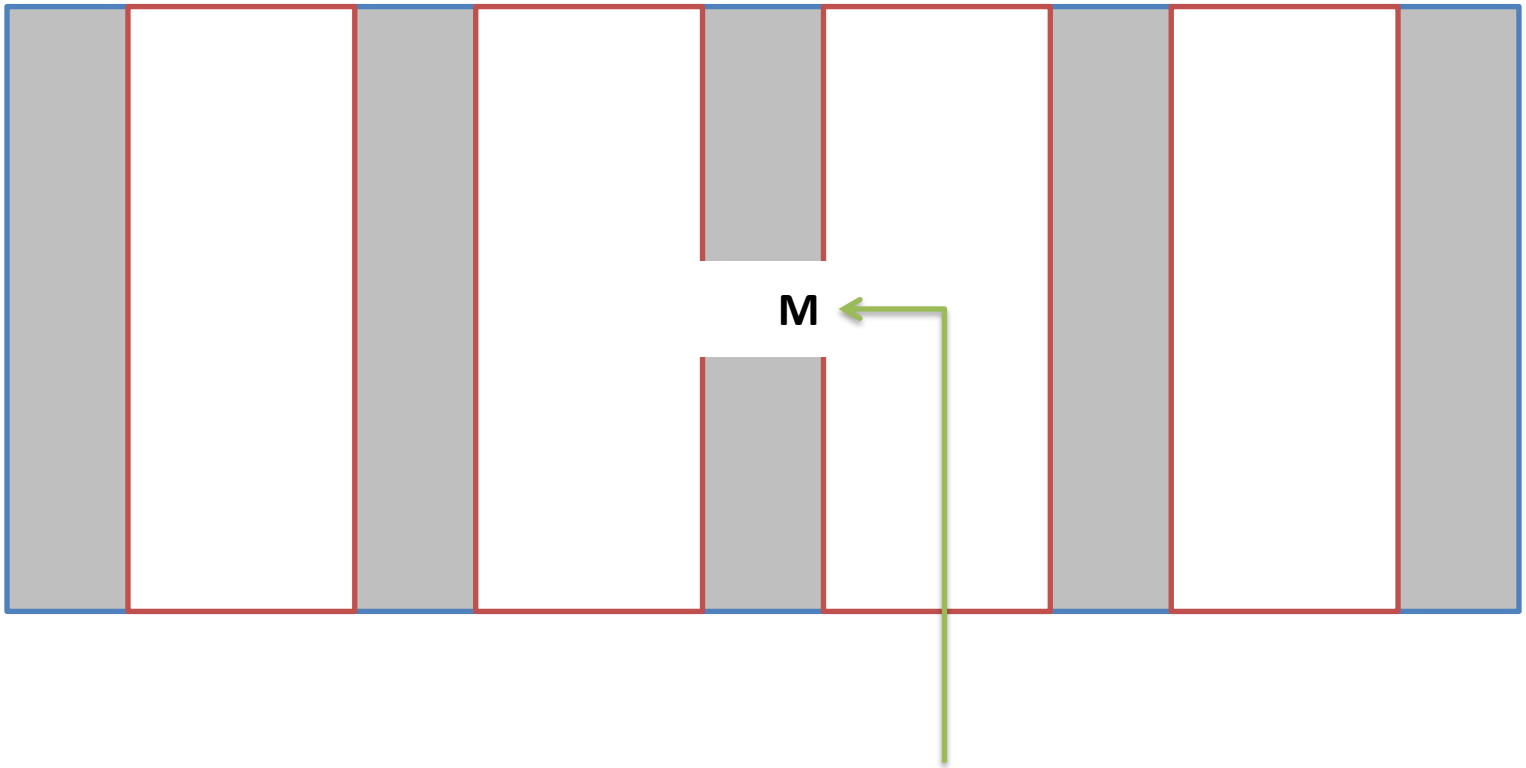
Синтаксический анализ (7)



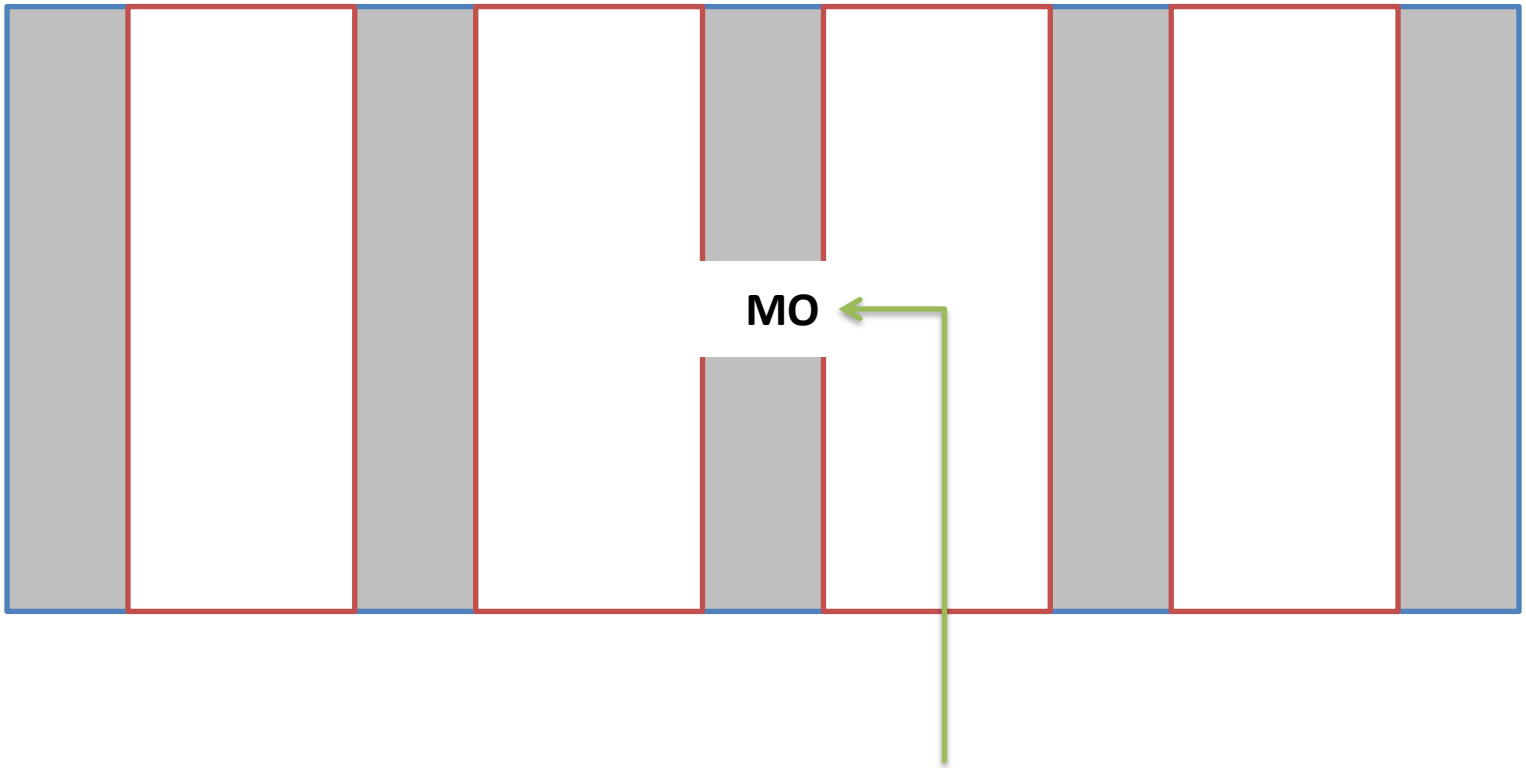
Синтаксический анализ (8)



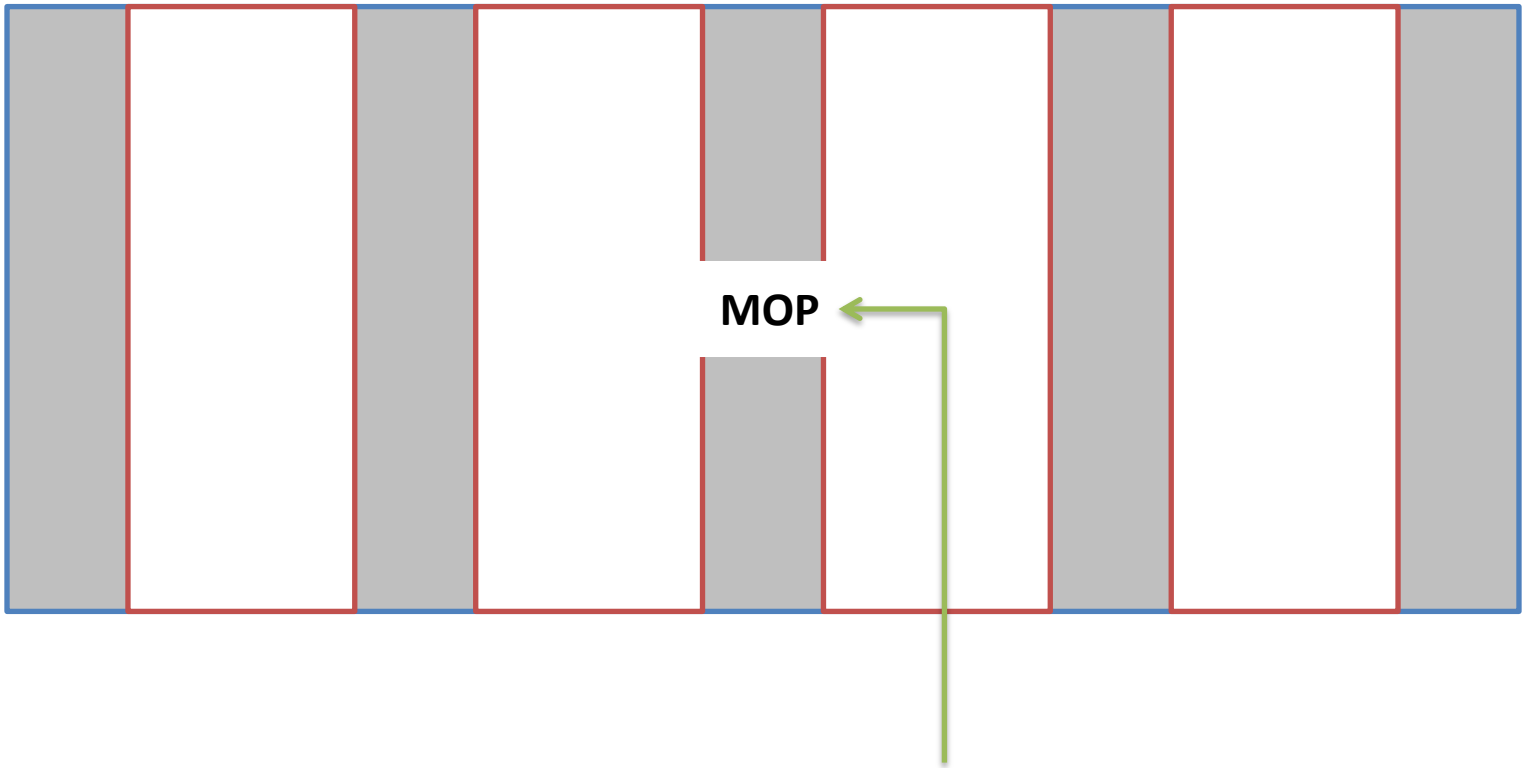
Токенизация (17)



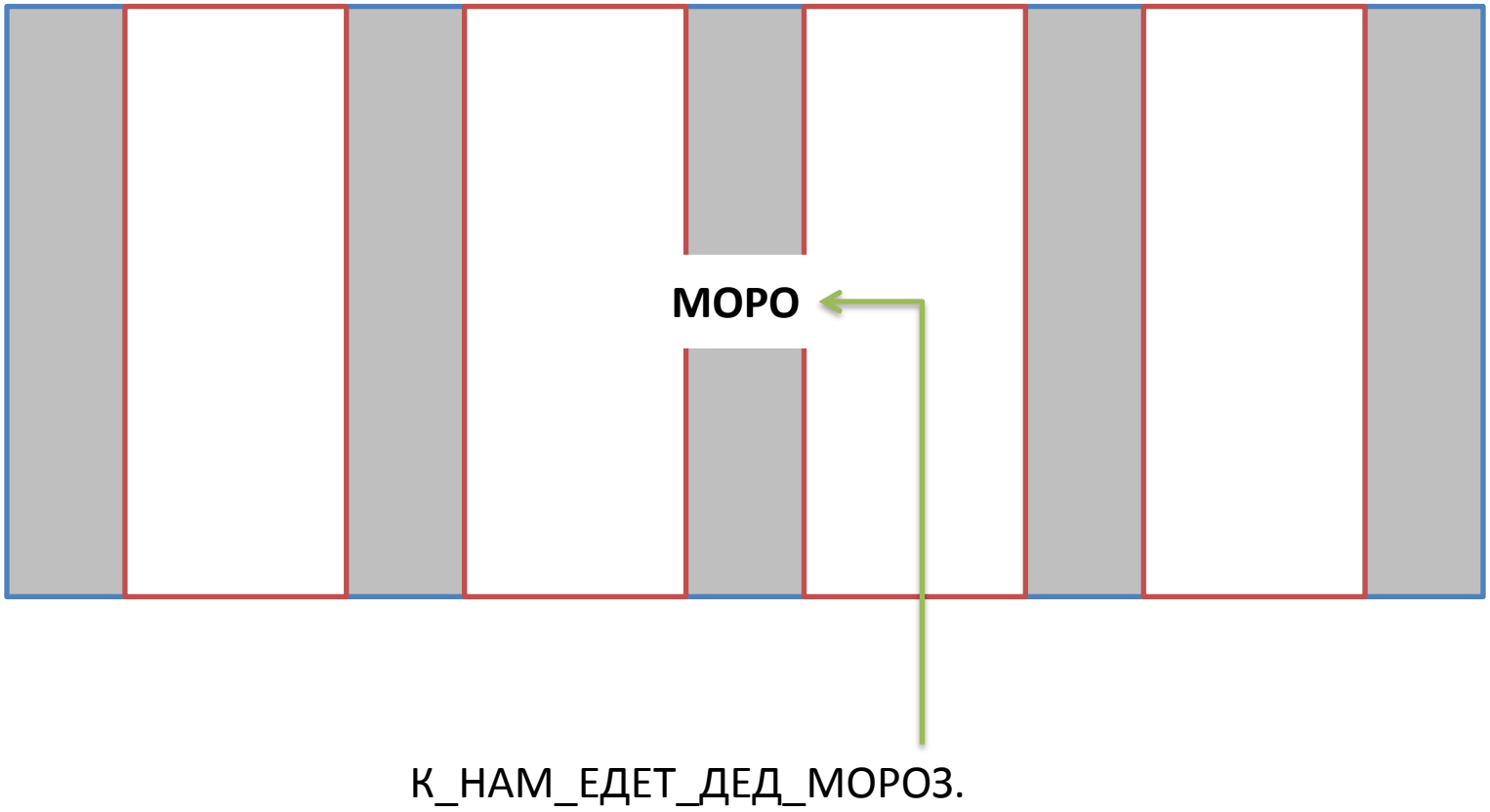
Токенизация (18)



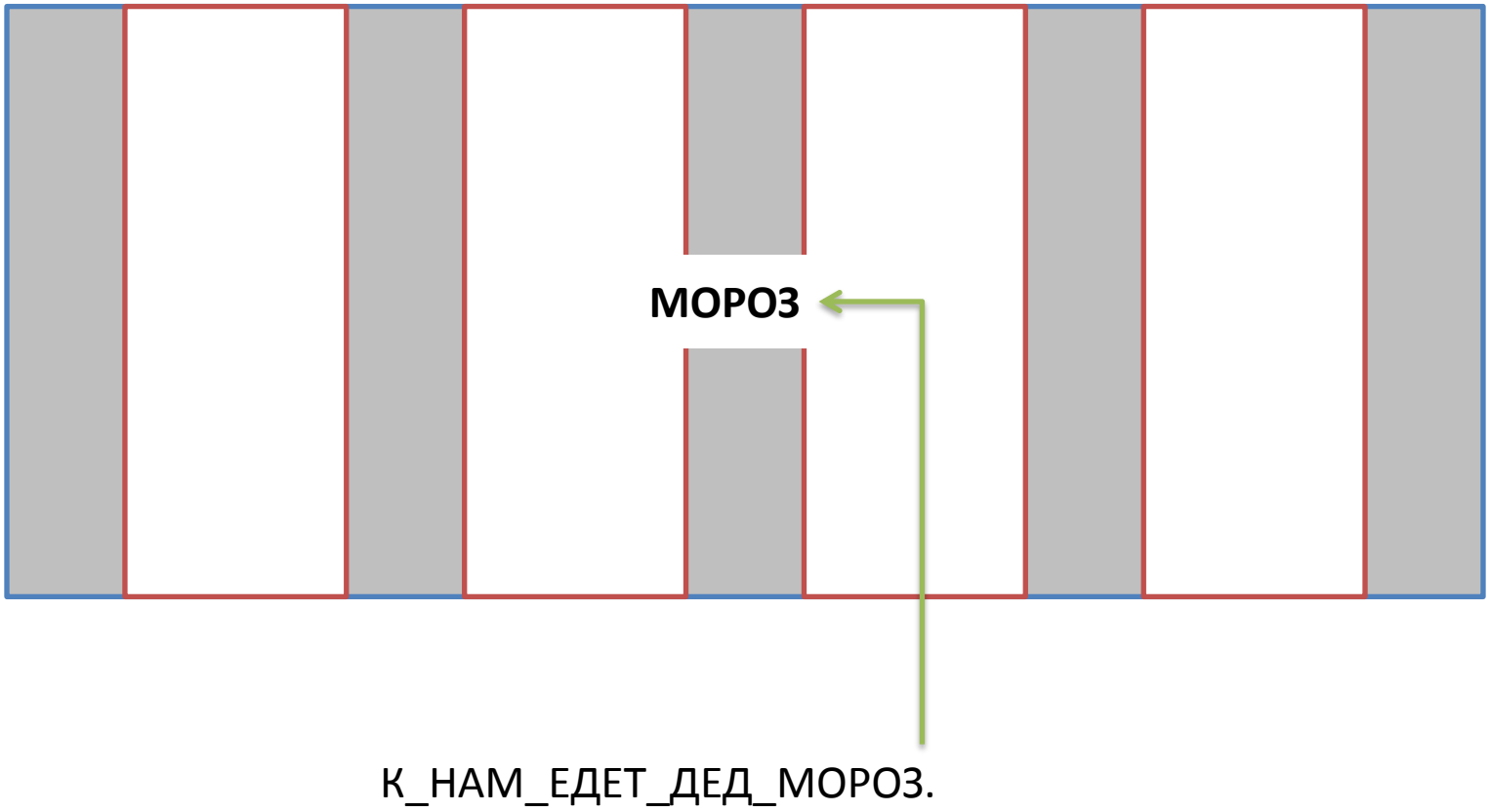
Токенизация (19)



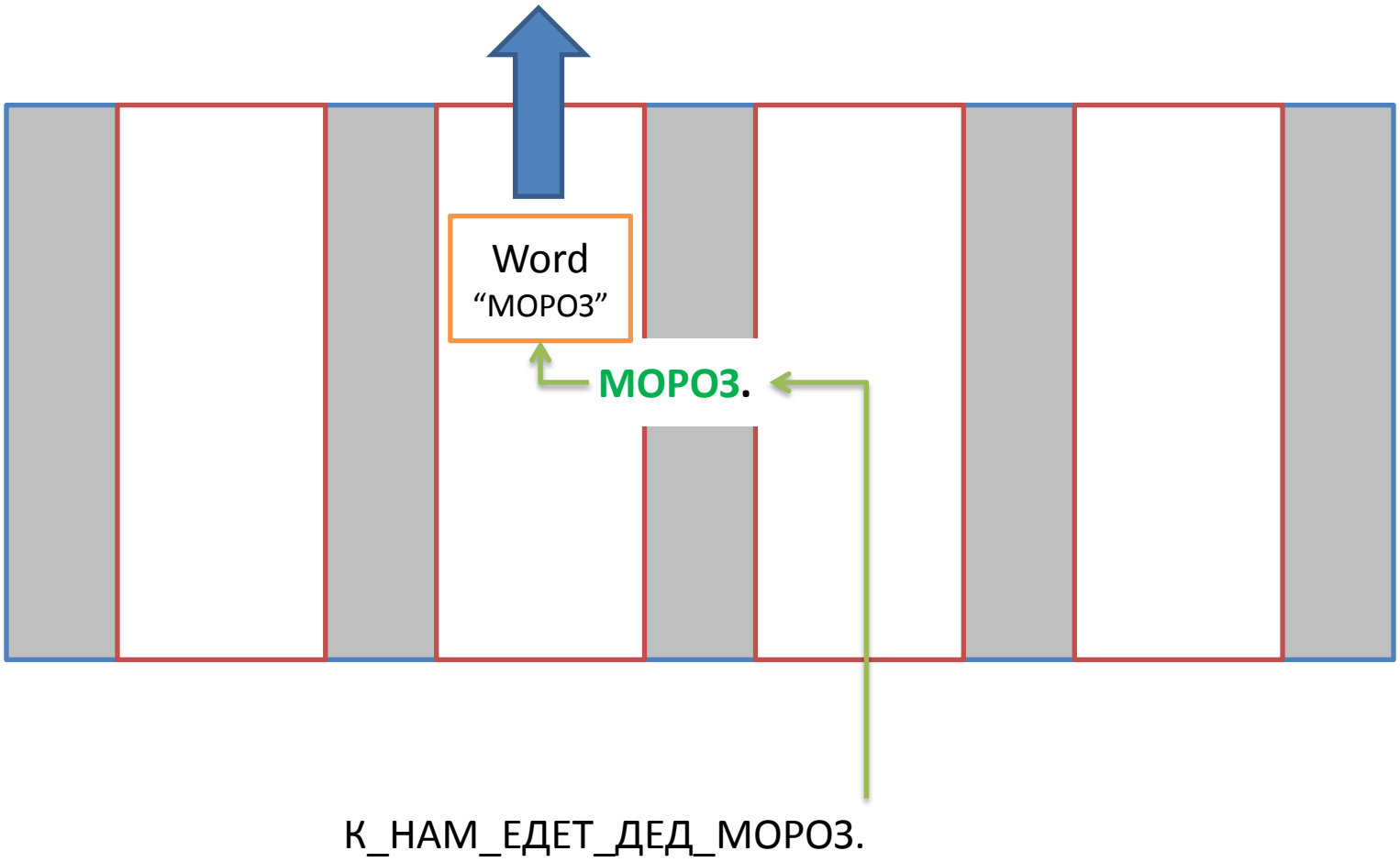
Токенизация (20)



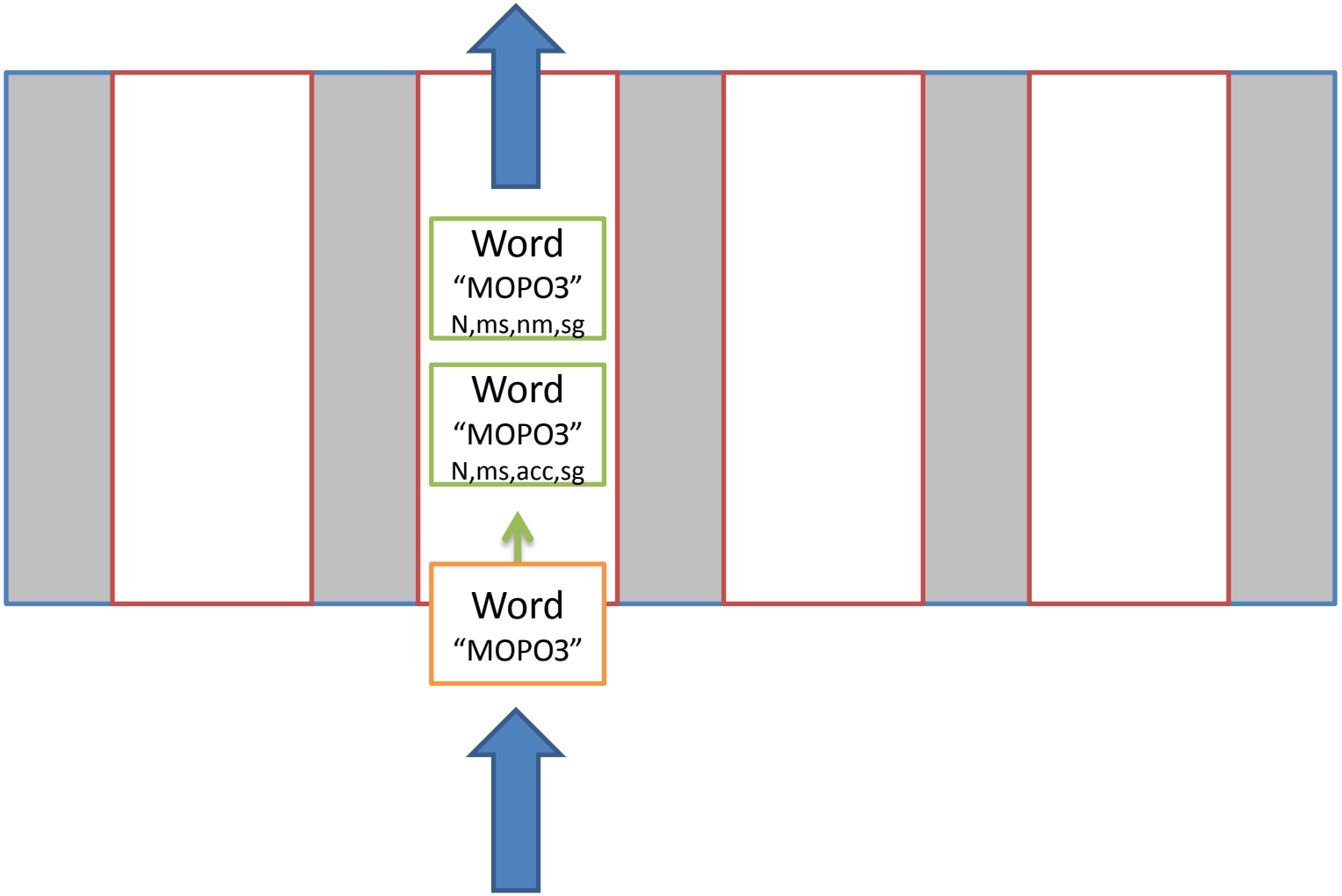
Токенизация (21)



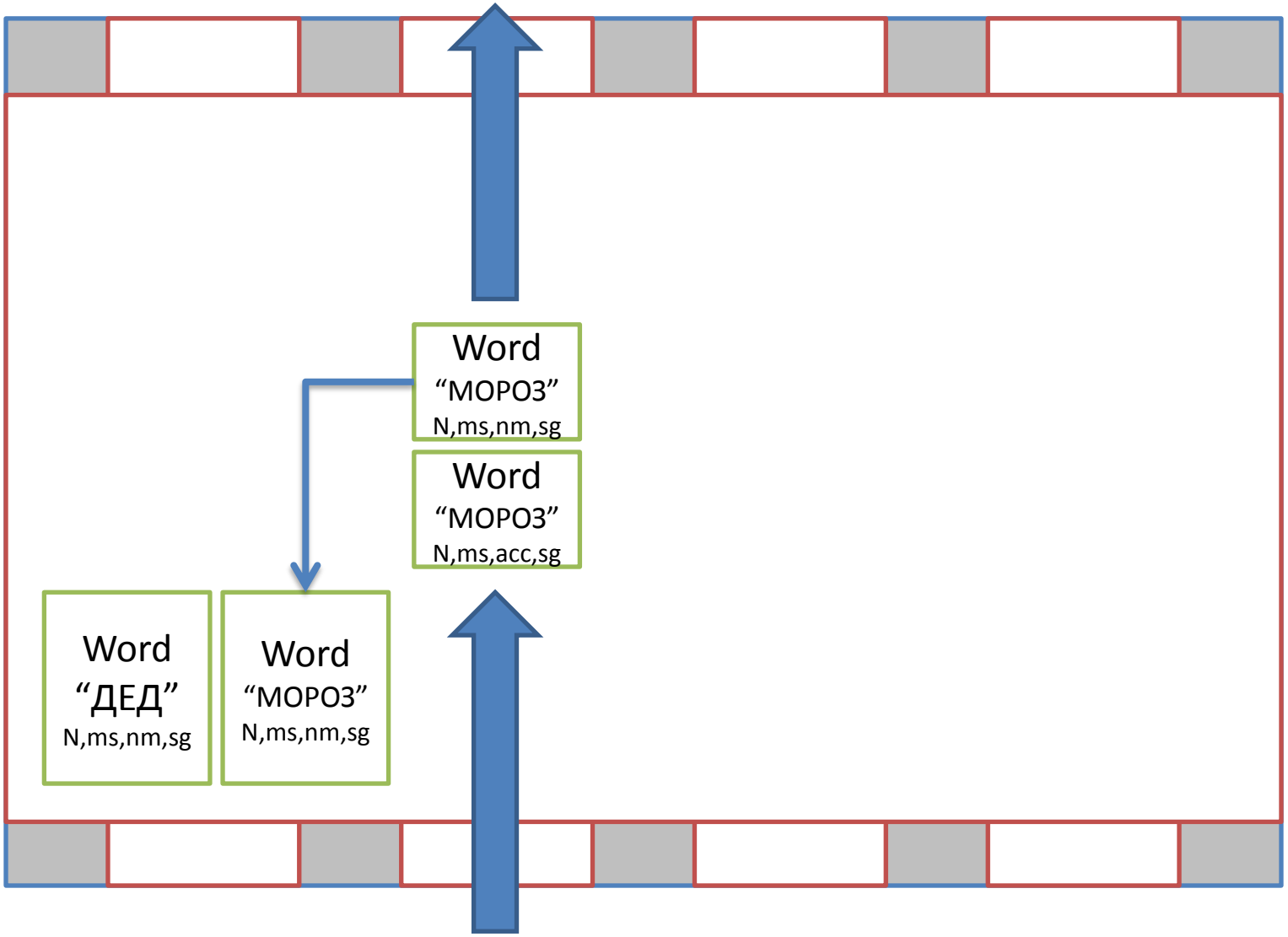
Токенизация (22)



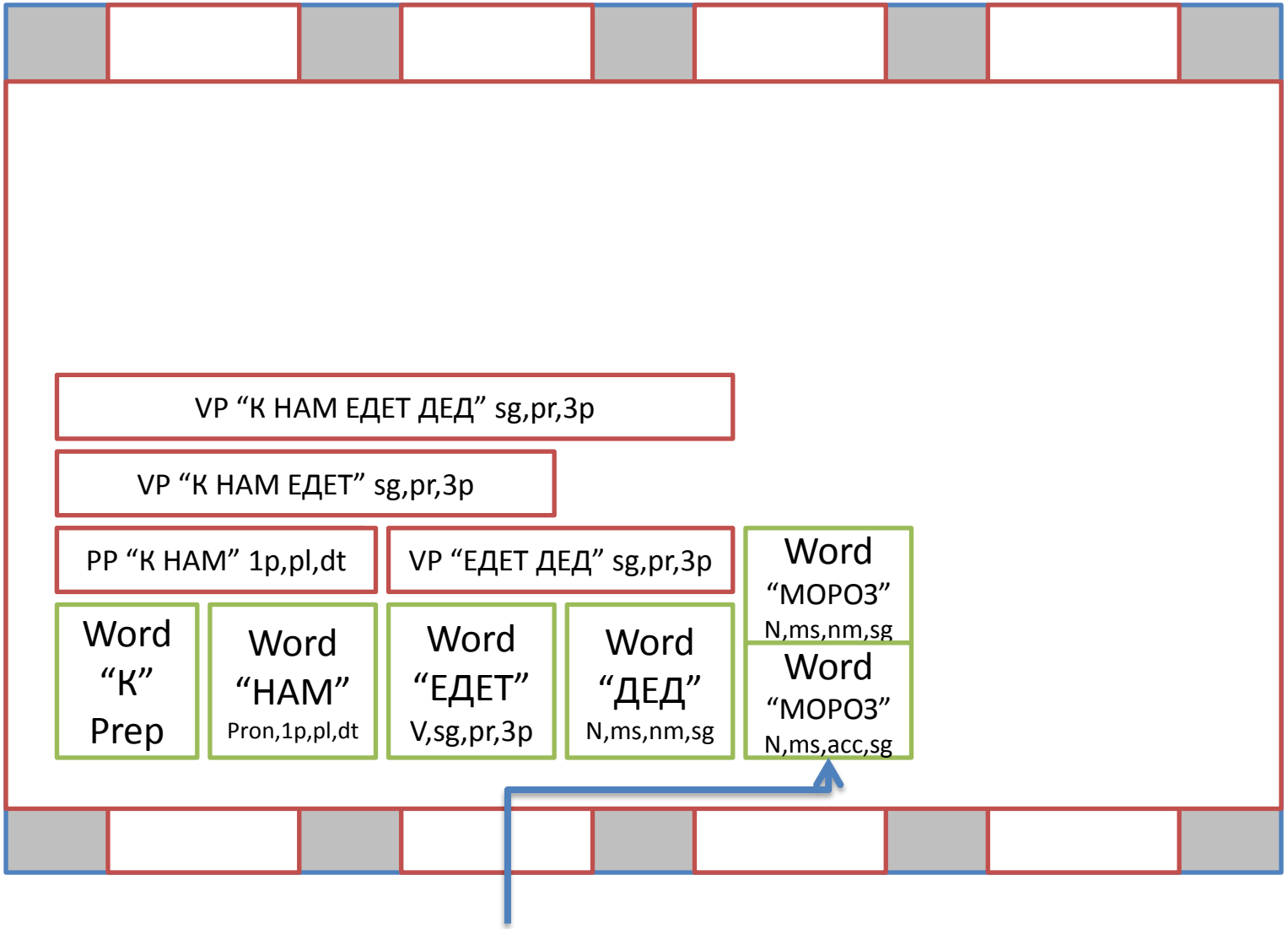
Морфологический анализ (5)



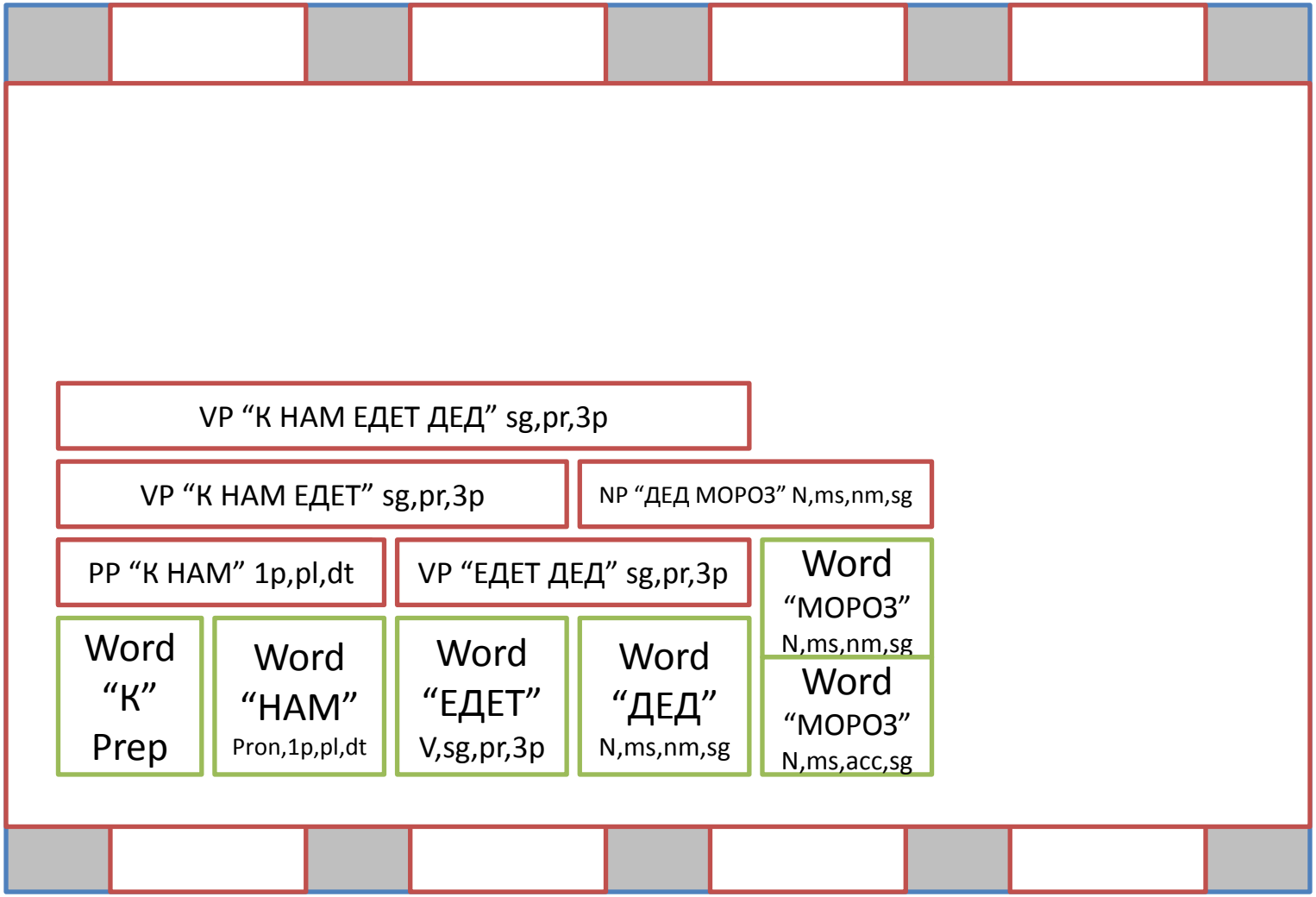
Извлечение именованных сущностей (5)



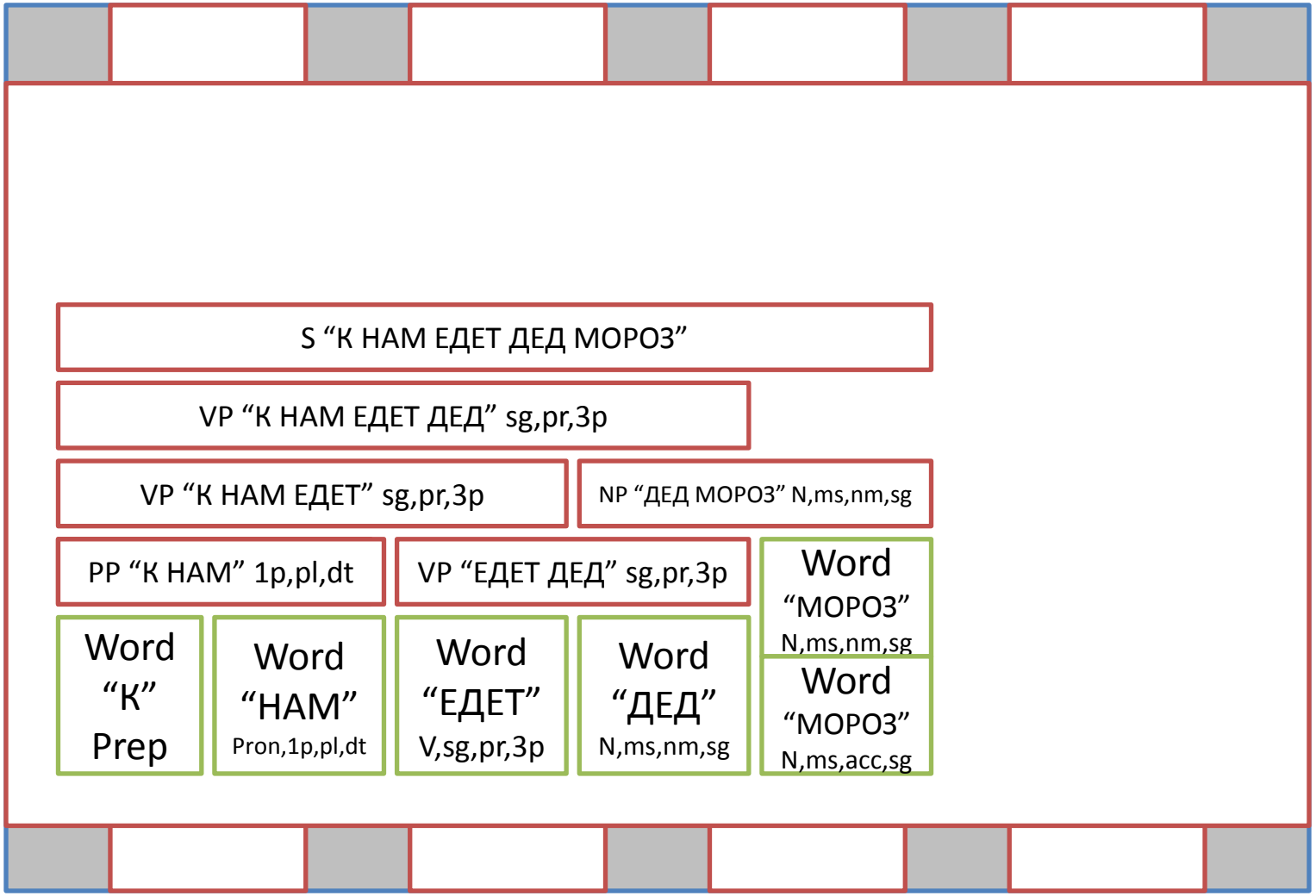
Синтаксический анализ (9)



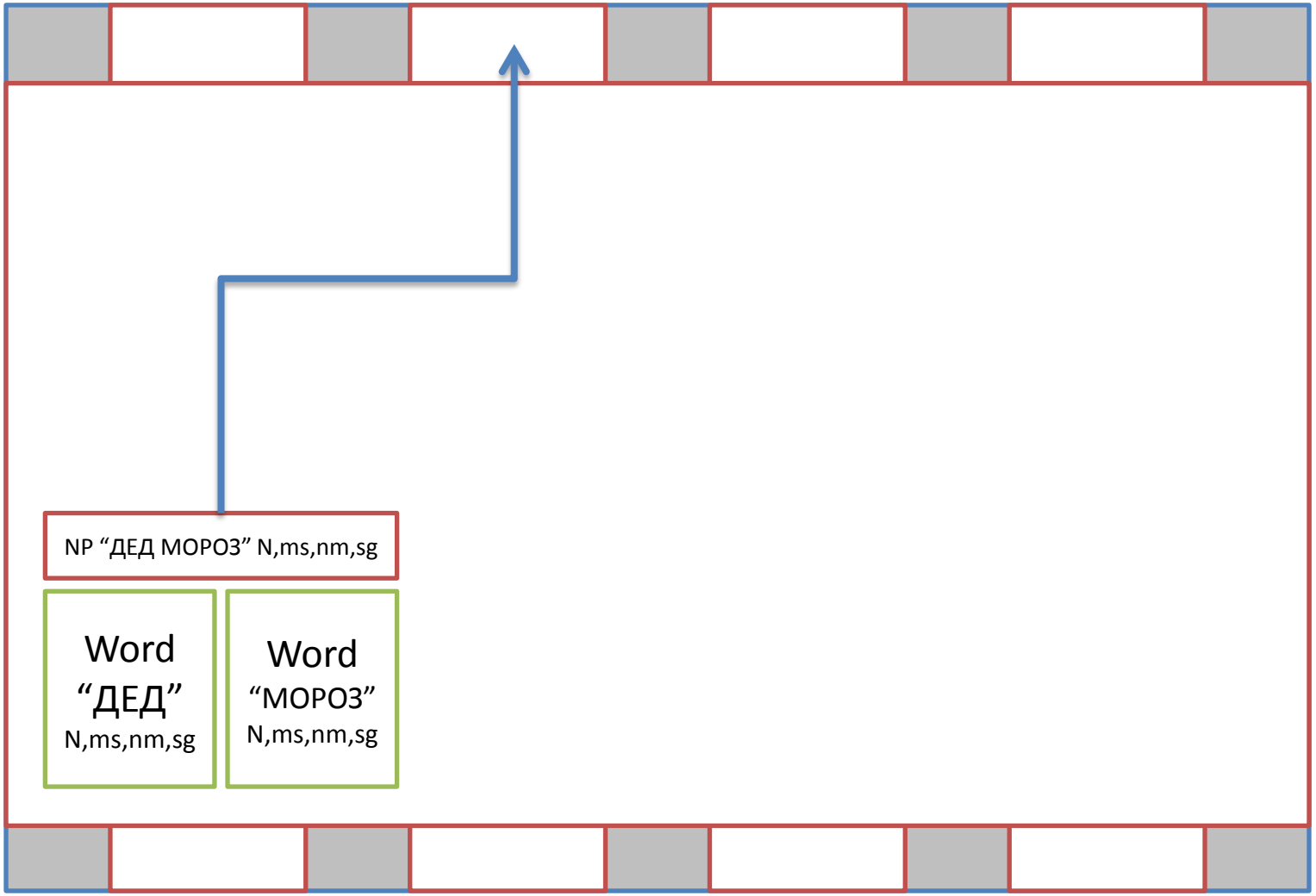
Синтаксический анализ (10)



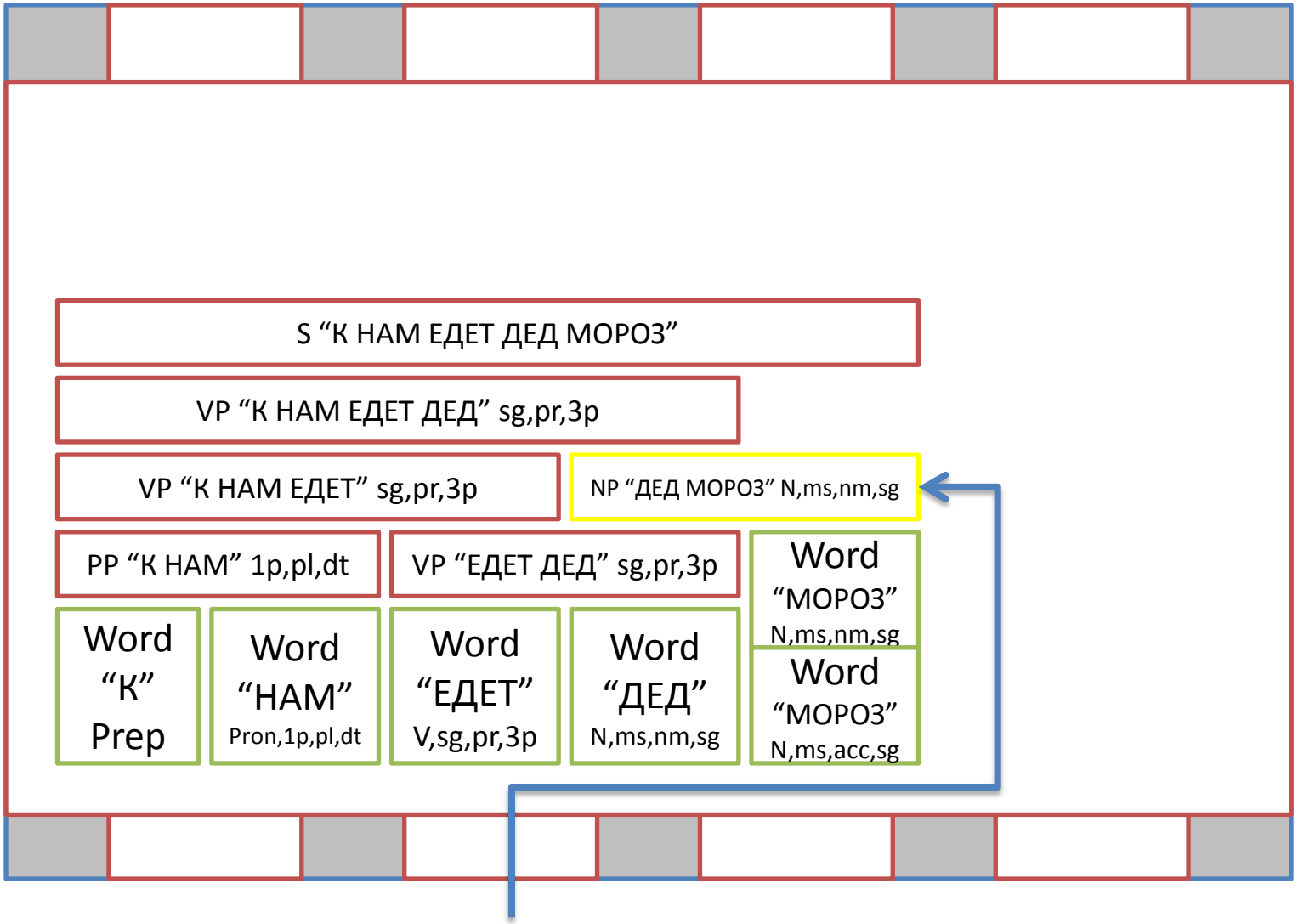
Синтаксический анализ (11)



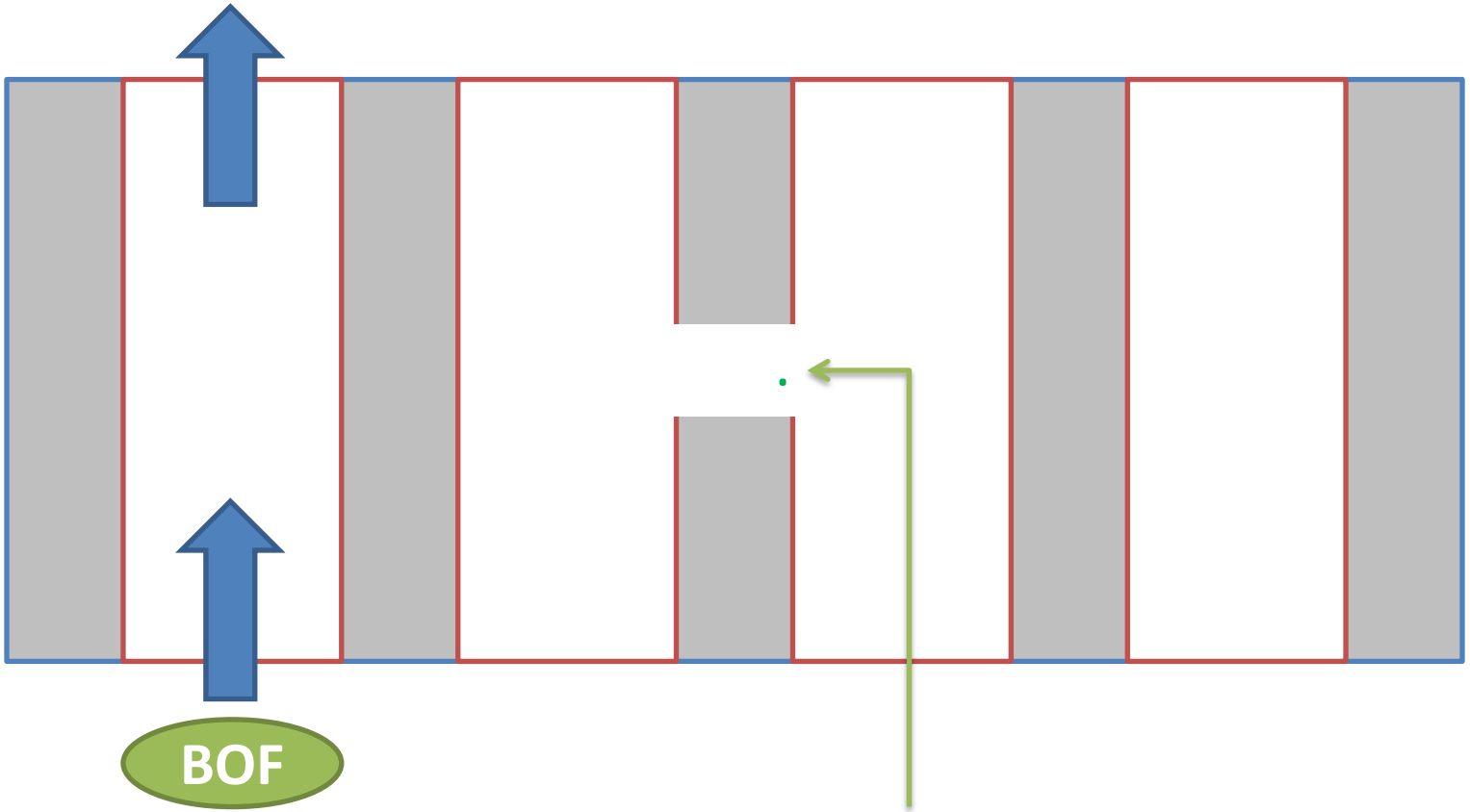
Извлечение именованных сущностей (6)



Синтаксический анализ (12)

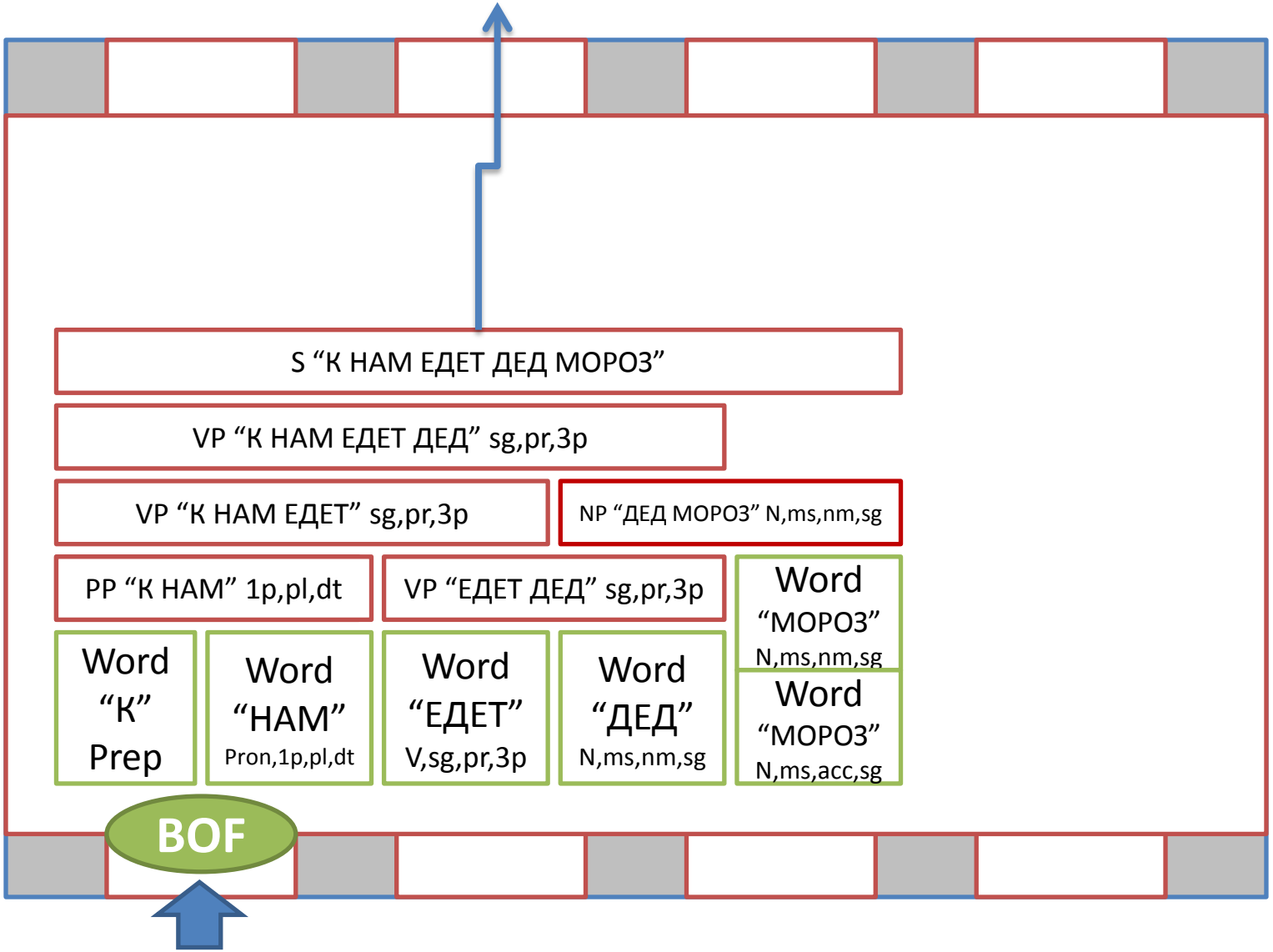


Токенизация (23)



К_НАМ_ЕДЕТ_ДЕД_МОРОЗ.

Синтаксический анализ (13)



Что даёт такая архитектура? (1)

- Нет проблем с представлением языковой омонимии: каждый модуль хранит только то, что ему нужно в удобной форме;
- В полной мере можно использовать более высокие уровни анализа для снятия омонимии более низких уровней (начиная с омонимии токенизации);
- Потребление памяти не зависит от объёма обрабатываемых данных;

Что даёт такая архитектура? (2)

- Можно получать результаты анализа не дожидаясь окончания данных в потоке;
- Предположительно, более высокая производительность в многопоточном режиме;
- Нет ограничений для задач, ориентированных на поток (распознавание речи, диалоговые системы).

Потенциальные проблемы архитектуры

- Более сложная логика работы системы в целом (труднее осуществлять отладку и логгирование);
- Понятие линейного порядка как таковое отсутствует (усложняется задача определения контактности и расстояния между языковыми единицами);

Спасибо за внимание!

http://hurmining.com

hedgeonline@gmail.com