Структурно-параметрическая модель перевода естественных языков

Д. Анисимов

24 июля 2004 г.

Содержание

1 Введение

Пусть пока наша система будет называться «Меркурий», — в честь подвига русских моряков принявших неравный бой с десятикратно превосходящим противником, и сумевших в этом бою победить. Почему у меня возникли именно такие ассоциации, надо объяснять?

«Меркурий» — система пофайлового перевода. К сожалению, в настоящее время построение автоматического переводчика, с человеческим качеством перевода невозможно. Я стремлюсь сделать автоматизированную систему — это когда перевод осуществляет в основном человек, а программа более или менее помогает ему, освобождает его от рутинных операций, избавляет от необходимости помнить слова, уменьшает количество набиваемого текста.

Не могу сказать, что это аналог Стилуса. Это другое.

Акцент сделан на возможности пользователя самому настраивать систему на перевод конкретных текстов. Сейчас система умеет переводить с английского и с Эсперанто на русский. Сверх идея состояла в том, чтобы сделать ее настраиваемой на любой язык. Насколько это удалось — покажет будущее.

Система построена по принципу мало кода — много данных.

Данные делятся на две неравнозначные части:

1. описание алгоритма перевода (маленькая и сложная).
2. словари (большие но простые).

Все данные пользователь может модифицировать самостоятельно. Собственно, за это и боролись. В идеале было бы присылать изменения мне, чтобы я их суммировал. Но я пока не придумал «систему суммирования» дополнений.

2 Основополагающие идеи и понятия.

Настоятельно рекомендую прочитать эту секцию. Понятия, которые я использую, несколько отличаются от общепринятых лингвистических. Это нужно, во-первых, чтобы приблизить систему понятий к компьютерным вычислениям, во-вторых, чтобы упростить некоторые без нужды усложненные гуманитариями вещи.

1. Мы занимаемся переводом *фразы* в *фразу.* (Фраза — это то, что от точки до точки.) Этот подход имеет ограничения, но это единственный способ написать такую программу в одиночку. Перевод абзац в абзац, наверное, более правильный, но мне это не по силам.

2. Система делает грамматически связный перевод. Однако выбор смысла многозначных слов оставлена на будущее. В некоторых уродских языках (типа английского) это может представлять значительную проблему. Сейчас выбор смысла таких слов возложен на человека.

3. Фраза состоит из *слов. Слово* — это то, что между двумя пробелами. Лингвисты, наверное, возмутятся этим определением. Но я не лингвист. У меня слово — не смысловое, а структурное понятие. Оно мне нужно, чтобы обьяснить компьютеру что нужно делать.

4. Слова изменяют свою *форму,* — склоняются, спрягаются и т.д. То, что характеризует форму слова (число, падеж и тд.) будем называть *параметром. Форма* бывает не только у слова, но и у словосочетания, и у целой фразы. Параметры разных языков не совпадают.

5. *Параметры* могут быть не только грамматическими, но и еще-какими нибудь. Например я использую «смысловые» параметры, которые мне нужны, например, для выделения различных множеств глаголов (глаголов умственного и физического восприятия, глаголов словесной коммуникации и пр.).

6. *Словосочетание* — группа слов, объединенная по каким-то признакам, и имеющая какой-то общий смысл. Что считать словосочетанием, и где оно начинается и кончается — тоже решает тот, кто описывает грамматику. Это может не совпадать ни с одним учебником.

7. Далее в тексте словосочетания и фразы я буду называть *конструкциями*. Мы транслируем *«конструкции оригинала»* в *«конструкции перевода»*.

8. Процесс (алгоритм) перевода описывается специальным настроечным файлом — *файлом описания алгоритма перевода* (lang.txt). Этот файл описывает следующие вещи:

   (a) Атомы из которых состоит язык оригинала.
   (b) Атомы из которых состоит язык перевода.
   (c) Форматы используемых словарей.
   (d) Соответствие конструкций (фраз и словосочетаний) языка оригинала и языка перевода.

   Описание алгоритма перевода — это почти программа. Так же как программа, оно может быть красивым и уродливым, эффективным или нет, правильным или ошибочным.

3 Идеи и понятия (средний уровень понимания).

Эта секция предназначена для пояснения описанных выше понятий. Также служит в качестве учебника по языку описания грамматики. Формальное описание языка описания грамматики находится в файле "Язык описания алгоритма перевода"(lang.ps).

Чтобы читателю были понятны приводимые в тексте примеры, необходимо дать несколько пояснений. Файл описания грамматики, помимо собственно конструкций, может содержать комментарии. Коментарии такие-же, как в C++ — одно- и многострочные. Например вот такие:

```
/* Это многострочный
   комментарий
*/
// Это однострочный комментарий
```

Служебные слова начинаются с `@` (эт коммерческое). Например, `@Атомы_источника`.

3.1 Параметры

Параметры — это некие характеристики слов. Характеризовать они могут что угодно. У меня они характеризуют две не очень похожие друг на друга вещи: грамматическую форму слов и смысл слов.

Какие будут использоваться параметры — полный произвол составителя грамматики. Например лицо и число русского глагола можно заменить одним параметром — характеристика_подлежащего.

Параметры бывают *постоянные*, то есть которые никогда не меняются у данного слова (например как род и одушевленность существительных), и *переменные* (как падеж и род у прилагательных). Один и тот же параметр у одной части речи может быть постоянным, а у другой переменным. Например у русских существительных род — постоянный параметр, а у русских прилагательных — переменный. В русском языке такая ситуация используется для согласования слов и определения функций слов в предложении. Параметры бывают не только у слов, но и у конструкций.

Параметры влияют на грамматическую форму слов, и могут быть вычислены. О вычислении параметров смотрите секцию, «Вычисление и использование параметров».

Каждый параметр имеет набор допустимых значений. Вот пример описания параметра:

```
@Параметр падеж {  Именительный, Родительный, Дательный,
                   Винительный, Творительный, Передложный  };
```

Эта запись означает, что в русском языке есть параметр падеж, который имеет шесть возможных значений — Именительный, Родительный, Дательный, Винительный, Творительный, Передложный. Очень похоже на enum в языке C++.

3.2 Части речи

Слова в естественном языке по некоторым признакам можно разделить на несколько разновидностей. Это будут *части речи*. Опять таки, это разбиение достаточно произвольно. Тут есть некоторые традиции, но я им не всегда следую. Например, в русском языке, я считаю причастие и деепричастие формой глагола. А в английском, я считаю, что вообще нет герундия — вместо этого есть форма глагола, отглагольное существительное, и прилагательное, совпадающие по внешнему виду.

Слова могут видоизменяться в зависимости от формы слова. Каждая часть речи имеет свой набор форм, и набор параметров, определяющих форму.

Вот пример описания русского существительного:

```
@Часть_речи существительное( род &Род, одушевл &Душа, падеж Падеж, число Число );
```

Эта запись означает, что есть такая часть речи - «существительное», у которого есть четыре грамматических параметра: Род, Душа, Падеж, Число. Параметры Род и Душа — постоянные (отмечены знаком &).

Помимо *элементарных* частей речи, бывают еще и *составные* части речи. Это когда понятие выражается не одним словом, а несколькими словами. Например — «Hot dog»[1]. Такие *составные* части речи описываются с помощью специальных конструкций «Структура1» и «Структура2». Как это делается, смотрите главу «Словосочетания».

3.3 Грамматические словари, и их описание.

Грамматический словарь должен описывать правила изменение части речи по ее формам. Сам словарь представляет собой массив *«строчек»*. В данном случае *«строчка»* — условный термин и потому пишется в кавычках. Это кусок файла от ';' до ';', и может занимать несколько нормальных строк. Каждая «строчка» соответствует одному слову. В «строчке» записываются формы этого слова. Первая запись в «строчке» — содержит слово в *начальной форме* — по ней в словаре перевода можно найти как это слово переводится.

Вот пример «строчки», описывающей русское слово «победа».

```
победа победы победе  победу победой  победе
победы побед  победам победы победами победах ;
```

[1]Этот несчастный «Hot dog» — одна из основных проблемм машинного перевода. По-этому упоминание «Hot dog» встречается по меньшей мере 10 раз в моих трудах.

В первой строчке параметр «число» имеет значение «единственне», во второй — «множественне». В обеих строчках параметр «падеж» последовательно изменяется от значения «Именительный» до значения «Передложный».

Формат грамматического словаря описывает порядок расположения форм словав «строчке». Чтобы однозначно определить форму слова нужно присвоить значения всем его грамматическим параметрам. Собственно, формат перевода состоит из записей, которые присваивают значения параметрам, и записей, которые говорят о том, что записано на очередной позиции в «строчке» — слово или параметр. Если 2 и более записей слов содержит одинаковое количество параметров, то второй раз параметр присваивать не обязательно.

Ниже дан пример описания словаря русских существительных.

```
@Формат существительное формат2
{
   Число=Ед
   Падеж=И @Слово  // @Слово означает, что очередная запись -
   Падеж=Р @Слово  // форма слова
   Падеж=Д @Слово
   Падеж=В @Слово
   Падеж=Т @Слово
   Падеж=П @Слово

   Число=Мн
   Падеж=И @Слово
   Падеж=Р @Слово
   Падеж=Д @Слово
   Падеж=В @Слово
   Падеж=Т @Слово
   Падеж=П @Слово
}
```

Обратите внимание, что в формате словаря нужно присваивать значения переменным параметрам (у которых не было знака &).

После того, как описаны части речи и их свойства, определены форматы словарей, нужно сказать компьютеру в каких файлах содержатся эти словари. Вот пример задания файлов, содержащих грамматические словри русских существительных:

```
@Файлы существительное формат2 { ru_sub1 ru_sub2 };
```

Эта запись означает, что русские существительные описаны в двух файлах — ru_sub1 и ru_sub2, оба имеют формат2. Если у данной части речи есть несколько грамматических словарей с разными форматами, это нужно описывать несколькими строчками, например вот так:

```
@Файлы существительное формат2  { ru_sub1 };
@Файлы существительное формат2a { ru_sub2 };
```

Несколько словарей для одной части речи бывают нужны, если слова этой части речи изменяются по-разному. Например русские числительные “одна” и “две” изменяются по родам, а все остальные числительные - нет. В этом случае для этих двух слов придется завести отдельный словарь.

3.4 Пример описания атомов языка.

Все описанные выше понятия, называются *атомы*, поскольку это «неделимые» элементы языка, в противоположность конструкциям.

Чтобы понять как связаны друг с другом атомы, привожу пример упрощенного описания атомов русского языка.

```
// *************************************************************
//                Атомы русского языка
// *************************************************************
@Атомы_приемника
{
   // ------- грамматические параметры русского языка ------------
   @Параметр падеж   {  И,Р,Д,В,Т,П  };
   @Параметр число   {  Ед,Мн  };
   @Параметр род     {  М,Ж,С,Мн  };
   @Параметр лицо    {  1е,2е,3е  };
   @Параметр время   {  прошлое,настоящее,будущее  };
   @Параметр залог   {  Актив,Пассив,Повелит  };
   @Параметр совершенность {  Сов,Несов  };
   @Параметр форма   {  Inf,глагол,причастие,деепричастие  };
   @Параметр душа    {  О,Н  };

   // ------- части речи русского языка ------------
   @Часть_речи глагол ( совершенность СС, форма Форма, лицо Лицо,
                        число Число, время Время, род Род, залог Залог ) ;
   @Часть_речи существительное( род &Род, душа &Душа, падеж Падеж, число Число );
   @Часть_речи прилилагательное( род Род, душа  Душа, падеж Падеж, число Число );
   @Часть_речи наречие ;

   @Часть_речи вопрос(  род Род, падеж Падеж, число Число );
   @Часть_речи местоимение( род &Род, лицо &Лицо, число &Число, падеж Падеж );
   @Часть_речи предлог ( падеж &Падеж ) ;
   @Часть_речи союз ;

   // -------- упрощенный формат словаря глаголов ---------------
   // -------- настоящий формат ГОРАЗДО сложнее -----------------
   @Формат глагол формат1
   {
      Форма=Inf @Слово
      Форма=глагол
      {  Залог=Актив
         Время=настоящее
         {  Число=Ед {  Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово  }
            Число=Мн {  Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово  }
         }
         Время=прошлое
         {  Число=Ед {  Род=М @Слово  Род=Ж @Слово  Род=С @Слово  }
            Число=Мн  @Слово
         }
         Залог=Пассив
         {  Число=Ед {  Род=М @Слово  Род=Ж @Слово  Род=С @Слово  }
```

```
            Число=Мн  @Слово
         }
         Залог=Повелит
         {  Число=Ед  @Слово  Число=Мн  @Слово  }
      }
   }
   // -------- формат словаря существительных ---------------
   @Формат существительное формат2
   {
      Число=Ед
      {  Падеж=И @Слово  Падеж=Р @Слово
         Падеж=Д @Слово  Падеж=В @Слово
         Падеж=Т @Слово  Падеж=П @Слово
      }
      Число=Мн
      {  Падеж=И @Слово  Падеж=Р @Слово
         Падеж=Д @Слово  Падеж=В @Слово
         Падеж=Т @Слово  Падеж=П @Слово
      }
   }
   // -------- формат словаря прилагательных ---------------
   //  использование препроцессора чтобы не писать много раз одно и то же
   # define pril_chislo_rod \
      {  Число=Ед        \
         {  Род=М @Слово \
            Род=Ж @Слово \
            Род=С @Слово \
         }               \
         Число=Мн @Слово \
      }
   @Формат прилагательное формат3
   {
      Падеж=И        pril_chislo_rod
      Падеж=Р        pril_chislo_rod
      Падеж=Д        pril_chislo_rod
      Падеж=В Душа=Н pril_chislo_rod
      Падеж=В Душа=О pril_chislo_rod
      Падеж=Т        pril_chislo_rod
      Падеж=П        pril_chislo_rod
   }

   // -------- грамматические словари русского языка ----------------
   @Файлы  глагол                 формат1   { dicts/zform/russkij/ru_verb_r  };
   @Файлы  существительное        формат2   { dicts/zform/russkij/ru_sub_r };
   @Файлы  прилагательное         формат3   { dicts/zform/russkij/ru_adj_r };
   @Файлы  вопрос                 формат3   { dicts/zform/russkij/ru_quest };
}
```

3.5 Конструкции

Конструкция — группа слов, объединенная по каким-то признакам, имеющая какой-то общий смысл, и какую-то роль в предложении (например роль сказуемого).

Конструкции бывают двух основных видов — *структура* и *выбор*. *Структура* — это когда конструкция состоит из нескольких стоящих друг за другом частей. Например, предложение состоит из подлежащего, сказуемого и второстепенных членов предложения.

Выбор — это когда конструкция может быть выражена несколькими способами. Например, подлежащим может быть местоимение или существительное.

Не правда ли, похоже на логические функции “и” и “или”? С помощью этих двух видов конструкций можно описать любое грамматическое явление любого естественного языка.

А чтобы описывать не один язык, а алгоритм перевода с одного языка (языка оригинала) на другой (язык перевода), надо описать эту же конструкцию на языке оригинала и перевода. Таким образом получается *трансляционная пара*, описывающая одно правило перевода. Каждая такая пара содержит конструкцию на языке оригинала, и конструкцию на языке перевода.

Эти два «основных» вида конструкций имеют некоторые важные подвиды, о которых обязательно надо сказать.

Беспорядок — это структура, порядок частей которой строго не зафиксирован, и они могут меняться местами. В английском языке такая конструкция никогда не используется, но для трансляции романских языков без конструкции типа *Беспорядок* обойтись будет нельзя.

Множество — это слово или группа слов, которая может быть той или иной конструкцией из некоторого (простите за тавтологию) множества. При описании естественного языка такая конструкция не применяется, но она нужна, например, если надо переводить текст, в котором спецтермины выделены курсивом, и известно, что эти термины существительные либо глаголы. *Множество* немного похоже на *Выбор*.

О других подвидах конструкций смотрите подраздел «словосочетания и другие сложные конструкции».

Пример описания трансляционной пары типа «структура»:

```
@Перевод
{  // --- структура фразы в языке оригинала ----
   @Структура   фраза_повествовательная =
                подлежащее             сказуемое              дополнение ;
   // --- структура фразы в языке перевода ----
   @Структура   фраза_повествовательная =
                подлежащее( &Число, &Лицо, &Род ) дополнение сказуемое( Число, Лицо, Род ) ;
}
```

Эта запись означает, что в языке оригинала фраза состоит из подлежащего, сказуемого и дополнения, которые стоят друг за другом. В языке перевода эти члены предложения имеют другой порядок - подлежащее, дополнение, сказуемое. Подлежащее и сказуемое согласуются в числе лице и роде.

Пример описания конструкции типа "выбор":

```
@Перевод // --------------- подлежащее ---------------------
{
   // --- варианты реализации подлежащего в языке оригинала ----
   @Выбор     подлежащее( число Число, лицо Лицо ) = // заголовок
              собирательное_местоимение              // варианты подлежащего
              личное_местоимение( &Род, &Лицо ,&Число )
              существительное( &Число )
```

```
            вопросительное_слово ;

   // --- варианты реализации подлежащего в языке перевода ----
   @Выбор     подлежащее( число &Число, лицо &Лицо, // заголовок
                          падеж &Падеж, род Род ) = // заголовок
            собирательное_местоимение             // варианты подлежащего
            личное_местоимение( &Род, Лицо, Число, Падеж )
            существительное( Число, Падеж, &Род )
            вопросительное_слово( Падеж, ) ;

   // --- что такое таблицы передачи параметров, и для чего они нужны,
   // --- будет описано в секции "вычисление и использование параметров"
   @Таблица_передачи( @Выбор, лицо Лицо = лицо Лицо )
   {
      1 @0 = 3е ;
      2 1е = 1е ;
      2 2е = 2е ;
      2 3е = 3е ;
      3 @0 = 3е ;
      4 @0 = 3е ;
   }
}
```

3.5.1 Константы

Некоторые конструкции (чаще всего структуры) могут иметь в своем составе постоянные составляющие. Примером является образование различных временных форм в английском языке. Например в выражении

```
would have been asking
```

только последнее слово (asking) является смысловым глаголом. Все остальное — *константы*, нужные для образования “будущего в прошедшем” времени (Future in the Past Perfect Continuous (Я специально выбрал самое монстрообразное)). Вот как описывается такая конструкция:

```
@Структура сказуемое_предбудущее_в_прошедшем_длительное( смысл_глагола Смысл ) =
           @0[would] @0[have] @0[been] глагол( Ing,&Смысл ) ;
```

Запись типа @0[что_нибудь] обозначает константу. Константы абсолютно неизменны и не являются частью речи.

3.5.2 Константные слова.

По-русски сложная форма будущего времени выражается следующим образом:

```
буду спрашивать
```

В данном случае смысловой глагол — «спрашивать», а «буду» — *константное слово* (глагол) нужное для образования будущего времени. Это слово изменяется по лицам и числам по правилам глагола (и этот факт надо обязательно указать при описании этой структуры вот так: `глагол[быть]`). Сравните это с «would have been» из предыдущего примера. Здесь константное слово, там — настоящие жесткие, неизменяемые константы. Вот как описывается такая конструкция по русски:

```
   @Структура сказуемое_будущее( лицо &Лицо, число &Число ) =
              глагол[быть]( Сов,глагол,Лицо,Число,будущее,@0,Актив,@0 )
              глагол<1>( Несов,Inf,@0,@0,@0,@0,Актив,@0 )
              ;
```

Запись типа глагол[что_нибудь] обозначает слово, присутствующее в конструкции, являющуюся частью речи. Это слово склоняется спрягается и т.д. Иными словами изменяет свою грамматическую форму, сохраняя смысл.

3.5.3 Главное слово конструкции

Иногда хочется задать не константное слово, а целую константную конструкцию. Например, такое желание возникает в случае, когда надо перевести выражение «have breakfast» - «завтракать». «have» - это глагол, значит он будет иметь сложные формы прошедшего и будущего времени. В этом случае хочется сказать программе что если после сказуемого с глаголом «have», будет слово «breakfast», то переводить его надо так-то и так-то.

Проблема решается если знаком `!` обозначить в конструкции главное слово. На языке описания грамматики это выглядит следующим образом:

```
@Перевод
{
   // "главное слово" отмечено восклицательным знаком

   @Структура сказуемое_отр_прошлое( см_глагола S1 ) =
              @0[did] @0[not] !глагол( &S1, %Inf ) ;
   @Структура сказуемое_отр_прошлое( лицо &Лицо, число &Число, род &Род ) =
              @0[не] !глагол( %Сов,%глагол,@0,Число,%прошлое,Род,%Актив,@0 ) ;
}
@Перевод
{
   // передача "главного слова" отмечена восклицательным знаком

   @Выбор сказуемое1 =
              !сказуемое_настоящее
              !сказуемое_отр_настоящее
              !сказуемое_прошлое
              !сказуемое_отр_прошлое
              .... // тут еще много-много вариантов
              ;
   @Выбор сказуемое1( лицо &Лицо, число &Число, род &Род ) =
              !сказуемое_настоящее    ( Лицо, Число )
              !сказуемое_отр_настоящее( Лицо, Число )
              !сказуемое_прошлое      ( Лицо, Число, Род )
              !сказуемое_отр_прошлое  ( Лицо, Число, Род )
      ....
              ;
}
@Перевод
{
   // использование главного слова в обоих частях перевода

   @Структура сказуемое_составное =
```

```
            сказуемое1[have] @0[breakfast];
   @Структура сказуемое_составное( лицо &Лицо, число &Число, род &Род ) =
            сказуемое1[завтракать]( Лицо, Число, Род ) ;
}
```

3.5.4 Индексы передачи смысла

Бывает, что в конструкции встречается несколько составляющих одного типа. Например, несколько существительных. В таком случае нужно задавать соответствие, что во что переводить, чтобы не получались «связи офицера» вместо «офицер связи». Для этого служат *индексы передачи смысла*, которые выглядят вот так: <1> (число в угловых скобках). Если в конструкции не заданы индексы передачи смысла, то первое существительное оригинала переводится в первое существительное перевода, первое прилагательное — в первое прилагательное и т.п. Пример конструкции с индексами смысла:

```
@Перевод
{  @Структура конструкция_генетив =
            существительное<1> @0[de] существительное<2> ;
   @Структура конструкция_генетив =
            существительное<1>( %Имен ) существительное<2>( %Родит ) ;
}
```

3.5.5 Пустой выбор.

Бывает, что в конструкции может быть необязательная часть. Например отрицательная частица «не». Это можно описать с помощью приема *пустой выбор*. Пустой выбор обозначается символом @00 ;

Выглядит это так:

```
@Перевод
{  @Выбор    _отрицание =   //  Эта запись означает, что когда в
              @0[no]        //  языке оригинале присутствует частица "no"
              @00 ;         //  в переводе появляется частица "нет"
   @Выбор    _отрицание =
              @0[не]
              @00 ;
}
@Перевод
{  // --- а вот как используется необязательный член конструкции
   @Структура сказуемое_1( смысл_глагола Смысл ) =
            _отрицание  глагол ;
   @Структура сказуемое_1( лицо &Лицо, число &Число ) =
            _отрицание  глагол( Несов,Inf,@0,@0,@0,@0,Актив,@0 )
}
```

Разумеется можно написать и наоборот — вот так:

```
@Перевод
{  @Выбор    _отрицание =   //  Эта запись означает, что когда в
              @0[no]        //  языке оригинале присутствует частица "no"
              @00 ;         //  в переводе частица "нет" отсутствует
   @Выбор    _отрицание =
```

```
            @00
            @0[не] ;
}
```

Это бывает нужно, когда есть глаголы с противоположным смыслом. Например, есть английское слово «fail» — «не достигнуть».

4 Вычисление и использование параметров.

Для связного перевода нужно не только всем словам подобрать перевод и расставить их в правильном порядке. Нужно также задать правильную форму каждому слову, то есть вычислить их грамматические параметры. Вычисление параметров происходит в нескольких ситуациях.

4.1 Константное задание параметров.

Параметр конструкции может быть просто задан. Перед значением параметра ставится знак `\%`. Например вот так:

```
@Перевод
{  @Структура подлежащее =
            существительное( %Nominativ ) ;
   @Структура подлежащее =
            существительное( %Именительный ) ;
}
```

Это описание подлежащего. Эта конструкция описывает, что подлежащее — это существительное в падеже Nominativ. (То есть, если существительное в каком-то другом падеже, значит это не существительное, а что-то другое). На русский язык это переводится существительным в именительном падеже.

4.2 Незаданный параметр.

Параметр может быть не задан. Если параметр не задан в конструкции языка-оригинала, это значит, что для этой конструкции подходит слово с любым значением этого параметра. Если параметр в конструкции языка-перевода, то значит этот параметр не важен с точки зрения поиска в словаре форм.

Незаданный параметр обозначается знаком `@0`, например вот так:

```
@Перевод
{  @Структура сказуемое_можно =
            мод_глагол[may]( &Форма )
            !глагол( %Inf ) ;
   @Структура сказуемое_можно =
            мод_глагол[можно]( Несов,Глагол,Лицо,Число,&Время )
            !глагол( Несов,Inf,@0,@0,@0,@0 ) ;
}
```

В данном случае смысловой глагол стоит в неопределенной форме и у него отсутствуют значения лица, числа, некоторыех других параметров. Эти параметры не надо, и даже вредно задавать. Если их как нибудь задать, программа будет пытаться найти в словаре глагол в неопределенной форме, например первом лице и единственном числе. Конечно, она его не найдет.

4.3 Передача параметров внутри одного языка.

Внутри конструкций одного языка грамматические параметры просто передаются без всяких вычислений. Направление передачи определяется знаком амперсанда '&'. Параметр, имеющий этот знак является *источником*, не имеющий — *приемником*. Рассмотрим структуру:

```
@Структура группа_существительного( число &Число, падеж &Падеж, род Род ) =
           заголовок( Род, Число, Падеж )
           прилагательное( Род, Падеж, Число )
           существительное( &Род, Число, Падеж ) ;
```

Здесь структура и все ее компоненты имеют одинаковый набор параметров. При этом Род определается существительным и передается снизу вверх, (и также «вбок» от существительного к прилагательному и заголовку)

Число и Падеж в данной конструкции передаются сверху вниз (от конструкции к компонентам).

4.4 Вычисление параметров с помощью таблиц.

Обычно параметры в языке просто передаются - прилагательное имеет тот же род и падеж, что и существительное, глагол - то же лицо и число, что и подлежащее и т.п. Но бывают случаи, когда для параметров нужно делать сравнительно сложные вычисления. Для вычисления параметров служат таблицы.

Таблицы бывают трех видов:

@Таблица_источника - для вычисления параметров в языке оригинала
@Таблица_передачи - для передачи параметров от оригинала к переводу
@Таблица_приемника - для вычисления параметров в языке перевода

Вот пример таблицы, предназначенной для передачи параметров сказуемого:

```
@Перевод
{
   @Структура  сказуемое1 =
               глагол( %Глагол, &Время) ;
   @Структура  сказуемое1( лицо &Лицо, число &Число, род &Род ) =
               глагол( &СС, %глагол, Лицо, Число, &Время, Род, %Актив ) ;

   @Таблица\_передачи( время Время = время Время, совершенность СС )
   {
      прошлое   = прошлое   Сов ;
      настоящее = настоящее Несов ;
      будущее   = будущее   Сов ;
   }
}
```

Заголовок таблицы содержит имена параметров, до знака '=' параметров конструкции языка оригинала, после знака '=' параметров языка перевода. Далее идут строки значений.

Каждая строка является комбинацией входных и выходных значений параметров. Количество значений параметров до и после знака равно должно соответствовать количеству имен параметров в заголовке таблицы. Значения параметров должны соответствовать типам параметров в заголовке таблицы. Работает это так. Осуществляется последовательный просмотр строк таблицы. Если в конструкции оригинала все указанные в заголовке (до знака равно) параметры равны указанным в строке значениям, тогда параметрам конструкции перевода присваиваются значения из этой строки после знака равно.

По смыслу это похоже на таблицы истинности булевых функций. Только возможных значений не два (истина-ложь) а больше (равно числу падежей, например).

4.5 Параметр типа @Выбор.

Если в качестве имени параметра встречается слово «@Выбор», это означает, что строка может быть использована только в том случае, если в оригинале использыется соответствующий вариант реализации конструкции. Варианты выбора считаются с единицы. Слово `@Выбор` может встретиться только в конструкциях типа «Выбор» и только до знака '='.

Пример таблицы с параметром типа @Выбор:

```
@Перевод
{  // ------- подлежащее --------------
   @Выбор     подлежащее( число Число, лицо Лицо ) =
              местоимение( &Род, &Лицо ,&Число, %Именительный )
              существительное( &Число, %Именительный )
              ;
   @Выбор     подлежащее( число Число, лицо Лицо, падеж &Падеж, род Род ) =
              местоимение( &Род, &Лицо, &Число, Падеж )
              существительное( &Род, Падеж, &Число )
              ;
   @Таблица_передачи( @Выбор, лицо Лицо = лицо Лицо )
   {
      1  1е = 1е ;  // Эту таблицу надо понимать так:
      1  2е = 2е ;  // Если подлежащее выражено местоимением, то лицо
      1  3е = 3е ;  // соответствует лицу местоимения - 1е, 2е или 3е,
      2  @0 = 3е ;  // Если существительным, то подлежащее имеет 3е лицо
   }
}
```

Эту таблицу надо понимать так: Если подлежащее выражено местоимением, то лицо соответствует лицу местоимения — 1е, 2е или 3е. Если существительным, то подлежащее имеет 3е лицо. Обратите внимание, что выборы считаются с единицы.

4.6 Незаданный параметр в таблице.

Знак `@0` означает «неважный параметр». В данном случае - если подлежащее выражено существительным, то при любом значении «Лица» оригинала, подлежащее перевода будет иметь 3е лицо.

Если знак "@0"встречается в качестве значения после знака '=', например вот так:

```
   Inf   3е = настоящее @0 ;
```

Это означает, что соответствующий параметр не должен быть присвоен. То есть он останется неопределённым.

5 Словари перевода.

Для того, чтобы решать задачи связного перевода, словари перевода должны содержать не только слова, но и конструкции (называемые лингвистами речевыми оборотами). Соответственно есть два вида записей - перевод слова и перевод конструкции. Сейчас будет описан только перевод слов. О переводе выражений написано в главе «Словосочетания».

Словарь программы машинного перевода должен описывать не только перевод слова, но и некоторую грамматическую информацию о слове. Сейчас это:

1. какой частью речи является это слово
2. какие параметры (например род и одушевленность) имеет это слово

Перевод слова в словаре выглядит примерно так:

```
s[abbreviation]=s[сокращение](С,Н);s[аббревиатура](Ж,Н)
```

Здесь 's' перед квадратными скобками, означает, что слово «abbreviation» является существительным. В квадратных скобках пишется строковая константа (как в языке описания грамматики). Точка с запятой разделяет варианты перевода слова. В круглых скобках пишутся параметры. В данном случае они обозначают 'С' - средний род, 'Н' - неодушевленность.

Чтобы программа могла воспользоваться таким словарем, нужно описать следующее:

1. Псевдонимы частей речи.
2. Псевдонимы значений параметров[2]
3. Имена файлов, в которых находятся словари.

Пример описания формата и файлов перевода (мне кажется тут не нужно особых пояснений что зачем):

```
@Словари_перевода
{
   @Формат_перевод формат1
   {  // ---- псевдонимы грамматических параметров (наречия)
      @Псевдоним  нс степени
      @Псевдоним  нв времени
      @Псевдоним  нп пространства
      @Псевдоним  нч частоты
      @Псевдоним  .  прочее

      @Части_речи_источника
      {  // ----- псевдонимы конструкций оригинала
         v глагол ;
         s существительное ;
         a прилагательное ;
         d наречие ;
         i числительное ;
         p предлог ;
         c союз ;
         q вопрос ;
         o местоимение ;
```

[2]Параметры можно писать и без псевдонимов, если сами параметры достаточно короткие. Например у меня падежи и грамматический род обозначаются одной буквой. В таких случаях использование псевдонимов не дает выигрыша.

```
      }
      @Части_речи_приемника
      {  // ----- псевдонимы конструкций перевода
         v глагол ;
         s существительное ;
         a прилагательное ;
         d наречие ;
         i числительное ;
         p предлог ;
         c союз ;
         q вопрос ;
         o местоимение ;
      }
   }
   // ----- файлы перевода ---------
   @Файлы_переводов формат1 { base_enru };
}
```

6 Словосочетания и устойчивые выражения

6.1 Постановка задачи.

В любом естественном языке есть понятия, которые выражаются не одним словом, а несколькими словами, причем их нельзя переводить просто как эти самые слова сложенные в конструкцию. Пример такого понятия в английском — «hot dog», — ну нельзя его переводить как «горячая собака» . Такие словосочетания должны присутствовать в словаре и переводиться как единое целое.

В естественных языках любое словосочетание заменяет собой какую-нибудь часть речи. Чаще всего это «составное существительное» (например, «hot dog») или «составной глагол» (например, «have breakfast»). Бывают, и другие случаи. Давайте пока будем называть это «сложной частью речи».

По внутреннему устройству «сложная часть речи» всегда соответствует какой-нибудь нормальной конструкции языка, просто смысл у него не является суммой составляющих. В приведенных выше примерах это существительное с определением («hot dog») и сказуемое с дополнением («have breakfast»).

Внутри «сложной части речи» отдельные части изменяются по-падежам, числам и вообще, ведут себя как отдельные слова.

Составляющие «сложной части речи» словосочетания также могут иметь сложную структуру. Например, «have breakfast» в одном из сложных прошедших времен может иметь форму «have had having breakfast» [3].

С учетом этого всего чтобы описать перевод такого словосочетания надо определить следующие вещи:

1. Роль какой конструкции оно играет.
2. Из каких частей состоит (например «hot dog» - это прилагательное и существительное) Это нужно для того, чтобы программа знала, как может изменяться данное выражение. Например в данном случае во множественном числе - «hot dogs».
3. Передача параметров в этой конструкции.

[3] Чего Вы смеетесь? Я сам не знаю, что это означает. Я же про грамматику говорю!

Вот как это можно было бы описать на языке описания грамматики:

```
@Перевод
{  // как выглядел бы перевод выражения "hot dog" на языке описания грамматики

   @Структура существительное_составное =
              прилагательное[hot] существительное[dog] ;
   @Структура существительное_составное( род Род, душа Душа, число &Число, падеж &Падеж ) =
              существительное[сосиска]( &Род, &Душа, Число, Падеж ) ;

   @Таблица_передачи( число Число = число Число )
   {
      Ед = Ед ;
      Мн = Мн ;
   }
}
@Перевод
{  // как выглядел бы перевод выражения "have breakfast" на языке описания грамматики

   @Структура сказуемое_составное =
              сказуемое1[have] @0[breakfast];
   @Структура сказуемое_составное( лицо &Лицо, число &Число, род &Род ) =
              сказуемое1[завтракать]( Лицо, Число, Род ) ;
}
```

Описывать перевод устойчивых выражений таким способом можно, но очень затратно. Хочется записать их в словарь. Разуметтся, при этом словарь должен быть «достаточно умным» чтобы понимать перевод таких сложных конструкций. По сути словарь должен являться продолжением файла описания грамматики и хранить внутри себя аналогичную информацию. Только хранить ее надо в более компактном виде, чтобы человек мог легко читать и редактировать словарь.

Внимательно посмотрев на естественные языки можно заметить некоторые закономерности:

1. Устойчивое выражение соответствует какой-либо части речи, или какой-нибудь конструкции.

2. Внутренняя структура устойчивых выражений часто повторяется. Например, очень часто устойчивым выражением является прилагательное + существительное.

3. Помимо типичной структуры выражений может быть и нетипичная, то есть иметь уникальный членов.

4. Внутренняя структура выражения на языке источника и приемника обычно не совпадает.

5. Выражение внутри себя может содержать не только атомарные части речи, но и сложные конструкции (например сложное сказуемое).

Соответственно, чтобы описать перевод выражений нужно обеспечить следующие вещи:

1. Описать на языке грамматики внутреннюю структуру устойчивых выражений, отдельно для языка источника и языка приемника. Причем это описание должно быть похоже на описание части речи.

2. Ввести специальную конструкцию, похожую на @Выбор, которая сообщала бы программе, о том, что такие-то конструкции языка оригинала могут переводиться такими-то конструкциями языка перевода. Причем число типов конструкций оригинала и перевода может не совпадать.

3. Сделать так, чтобы словарь мог бы содержать в себе описание выражений. Это описание должно (как и конструкция) иметь заголовок и члены конструкции. Членом конструкции может быть не только элементарная часть речи, но и любая другая конструкция.

6.2 Стандартное словосочетание.

Структура типичного устойчивого выражения описывается следующим образом:

```
// -------- прилагательное + существительное
@Структура1 группа_существительного_с1( род Род1, одушевленность Душа1,
                                       число &Число, падеж &Падеж ) =
            прилагательное (  Род,  Душа, Число, Падеж )
            существительное( &Род, &Душа, Число, Падеж ) ;
```

То есть очень похоже на описание обычной структуры. Только слово @Структура заменено на @Структура1. В словаре перевод выражения этого типа будет выглядеть следующим образом:

```
S1:a[hot]s[dog]=S1(Ж,Н):a[невкусная]s[сосиска]
```

В этой строчке используются следующие обозначения:

S1 - псевдоним заголовка структуры (группа_существительного_с1)
a - псевдоним прилагательного
s - псевдоним существительного
Ж - параметр грамматического рода для структуры в целом (Женский)
Н - параметр одушевленность для структуры в целом (Неодушевленный)

Обратите внимание, что во всех структурах одного типа передача параметров будет происходить одним и тем же способом. Например, в данном случае род и одушевленность существительного будет передаваться в прилагательного, а число и падеж обеих составляющих будет передаваться из заголовка. Поэтому, передачу параметров для стандартных структур можно не описывать каждый раз в словаре, а описать один раз в файле описания грамматики. Также обратите внимание, что можно задавать параметры (род и одушевлённость) для структуры в целом.

6.3 Члены словосочетания.

Членом структуры может быть неизменная константа. Это обозначается знаком `@1`. В описании грамматики это будет выглядеть так:

```
// -------- прилагательное + существительное
@Структура1 группа_существительного_с2( род Род1, одушевленность Душа1,
                                       число &Число, падеж &Падеж ) =
            существительное( &Род, &Душа, Число, Падеж )
            @1 ;
```

А в словаре так:

```
... =S2:s[мясо] @0[по французски]
```

Членом структуры может быть слово, у которого задана только часть речи, но не заданно само слово. Тогда в словаре квадратные скобочки \[\] заменяются на тильдочку ~. Например английское выражение «a lot of существительное» может быть обозначено вот так:

```
S3:@0[a lot of] s~ =S3:[много] s~(P)
```

Также можно задавать параметры составляющим структуры. Например в выражении:

```
S3:@0[a lot of] s~ =S3:[много] s~(P)
```

буква 'Р' в круглых скобочках означает, что существительное, после слова «много» должно стоять в родительном падеже.

Точно так-же как при описании структур, в переводимом выражении могут встретиться одинаковые члены. В этом случае нужно использовать смысловые индексы в угловых скобочках. Например[4] :

```
VV:@0[prefere] S~<1> @0[ol] S~<2> =VV:@0[Лучше] S~<1>(И) @0[чем] S~<2>(И)
```

6.4 Нестандартное словосочетания.

Кроме стандартных структур, бывают еще и нестандартные. Разница в том, что в нестандартных структурах число и расположение их частей уникальное и почти не повторяется. Точно так-же уникальным является и вычисление параметров внутри таких структур. Нестандартные структуры описываются с помощью конструкции `@Структура2`. Например в описании грамматики:

```
@Структура2 гр_сущ_пп( род &Р, падеж &Падеж, число Число ) = ;
```

Обратите внимание, что описан только заголовок структуры. Тело ее находится в словаре. А вот как нестандартная структура описывается в словаре:

```
Sb:@0[most of] SS~=Sb(C):s[большинство](С,Падеж,Число) SS~(P)
```

Обратите внимание, что в словаре для слова «большинство» указаны параметры (а не значения) Падеж и Число. Это нужно, чтобы программа знала, как передаются параметры в этой нестандартной структуре. В стандартных структурах такое указание является лишним.

6.5 Конструкция типа @Выбор1.

Как уже говорилось, внутренняя структура устойчивых выражений на языке-оригинале и языке-переводе может не совпадать. Например структура «сложное существительное» может в языке оригинале выражаться двумя словами, а в языке-приемнике четырьмя. (То есть структурами1 разного типа) Например:

```
S1:a[main] s[manager]=S4(M,O):a[старший] s[начальник] @0[младшего дворника]
```

Чтобы указать программе варианты такого перевода, служит конструкция `@Выбор1`. Вариантами являются структуры типа @Структура1 и @Структура2. Любой вариант оригинала может соответствовать любому варианту перевода. Какому именно варианту соответствует перевод, определяется словарем. Число вариантов в оригинале и переводе у такого выбора может быть различным.

Вот как описывается конструкция типа @Выбор1:

[4]Пример взят из Эсперанто, но я думаю он понятен даже тем, кто не знает Эсперанто

```
// ------ словосочетания, которые могут играть роль существительного
@Перевод
{  // ------ язык оригинала
             // заголовок
   @Выбор1    группа_существительного_с ( число Число ) =
             // варианты реализации (внутренняя структура описана ниже)
             группа_существительного_с0( &Число )
             группа_существительного_с1( &Число )
             группа_существительного_с2( &Число )
             группа_существительного_сс( &Число ) ;
   // ------ язык перевода
             // заголовок
   @Выбор1    группа_существительного_с ( число &Число, падеж &Падеж, род Род ) =
             // варианты реализации (в русском языке их несколько больше)
             группа_существительного_с0( Число, Падеж, &Род )
             группа_существительного_с1( Число, Падеж, &Род )
             группа_существительного_с2( Число, Падеж, &Род )
             группа_существительного_с3( Число, Падеж, &Род )
             группа_существительного_с4( Число, Падеж, &Род )
             группа_существительного_сс( Число, Падеж, &Род ) ;
}
```

7 Особые конструкции.

В языке описания грамматики есть конструкции, которые играют особую роль.

7.1 @все_все

Это самая главная конструкция. Своего рода аналог функции main в языке C++. Когда программа переводит, она пытается построить именно эту конструкцию.

7.2 @догадка

Все, что я писал выше хорошо и здорово, но программа далеко не всегда может полностью перевести фразу. Часто бывает, что переводится только половина фразы. Происходит это по разным причинам - неизвестные слова, заковыристая грамматика, банальные ошибки авторов текста, для которых язык оригинала не родной. Оставшуюся часть фразы тоже хочется переводить связно. Для этого существует специальная конструкция `@догадка`. Вот как она может быть описана:

```
   @Выбор     @догадка =
              гр_существительного
              предлог_существительное
              гр_наречия
              сказуемое
              фраза_что
              фраза_обстоятельство
              фраза_однородный_член ;
```

Если фраза переведена не полностью программа пытается построить структуру @догадка начиная с первого непереведенного слова. Если с этого слова не получается, то начиная со следующего и так далее. Если связный перевод еще раз обрывается, то делается следующая попытка возобновить перевод.

Таким образом часто удается привести к приемлемому виду сложноподчиненные фразы, в которых проблеммы возникли в одной из частей.

7.3 @междометие

В естественных языках имеются слова, которые не являются членами предложения, а служат для оформления мысли, высказывания личного отношения, степени вероятности, междометия, слова-паразиты и т.п. Такие слова могут стоять в любом месте предложения. Для таких слов я завел специальную "особую"часть речи:

```
@междометие
```

В принципе, для таких слов надо задавать свои правила трансляции. Ведь эти слова могут относиться к фразе в целом, к какой-то части фразы, какому-то конкретному слову... Я пока этого не делаю, а просто перевожу "междометия"по словарю и вставляю их в русскую фразу приблизительно в то место, где они стояли в языке оригинала. Поскольку русский язык имеет почти свободный порядок слов, он мне прощает эту вольность.

7.4 @знаковые_части_речи

Иногда в текстах встречаются слова, которые не содержатся даже в самых полных словарях. В этом случае программа должна сделать какие-то предположения о том, какой частью речи является это слово. Самое простое - предположить, что слово может быть любой частью речи. Правда, это приводит к тому, что программе при переводе надо просмотреть огромное вариантов. Естественно предположить, что служебные части речи полностью присутствуют в словаре, а незнакомое слово является знаковой частью речи. Чтобы сказать программе, какие части речи у нас знаковые, служит конструкция `@знаковые_части_речи`:

```
@Множество  @знаковые_части_речи =
            существительное( &Ч, &П )
            прилагательное( &Ч, &П )
            глагол( &Ф, &В )
            наречие( &S )
            междометие ;
```

8 Всякие хитрости

8.1 Пре- и пост- процессор

В естественных языках встречаются всякие жаргонные гадости. Носители языка ужасающе коверкают свой язык. Например, в английском есть такие жаргонные выражения: «he've», «aren't», «it's». На грамотном языке эти выражения должны выглядеть так: «he have», «are not», «it is». Для перевода таких выражений с английского надо их преобразовать с жаргона, а при переводе на английский — на жаргон. Для этого существуют словари препроцессора и постпроцессора. По формату эти словари аналогичны словарю перевода. В файле описания грамматики задание пре- и постпроцессора выглядит следующим образом:

```
@Файлы_препроцессоров  формат1 {  dicts/english_russian/base_pre  };
@Файлы_постпроцессоров формат1 {  dicts/english_russian/base_post  };
```

Тело словаря выглядит вот так:

```
@0[aren't]=@0[are not]
@0[can't]=@0[can not]
@0[couldn't]=@0[could not]
@0[didn't]=@0[did not]
```

9 Пример файла описания алгоритма перевода

```
/****************************************************************************/
//             Пример описания перевода с идеального аналитического
//                      языка на идеальный синтетический
/****************************************************************************/

/****************************************************************************/
//                           язык оригинала
/****************************************************************************/
@Атомы_источника
{
   // ------- параметры ---------------------
   @Параметр число {  Ед,Мн  };
   @Параметр лицо  {  1е,2е,3е  };

   // ------- части речи --------------------
   @Часть_речи глагол ;
   @Часть_речи существительное( число &Число ) ;
   @Часть_речи прилагательное ;
   @Часть_речи наречие ;
   @Часть_речи местоимение( число &Число, лицо &Лицо ) ;

   // ------- структуры из словаря ----------
   @Структура1 существительное1 = прилагательное существительное ;
   @Структура1 существительное2 = существительное @1 ;

   // ------- форматы и файлы ---------------
   @Формат существительное формат1
   {  Число = Ед @Слово ;
      Число = Мн @Слово ;
   }

   @Файлы существительное формат1 { dicts_form/s_sub };
}
/****************************************************************************/
//                           язык перевода
/****************************************************************************/
@Атомы_приемника
{
   // ------- параметры ---------------------
   @Параметр число   {  Ед,Мн  };
   @Параметр лицо    {  1е,2е,3е  };
   @Параметр падеж   {  И,Р,Д,В,Т,П  };
```

```
@Параметр род      {  М,Ж,С  };
@Параметр время    {  прошлое,настоящее,будущее  };
@Параметр залог    {  Неопредел,Актив,Повелит  };

// ------- части речи ---------------------
@Часть_речи глагол          ( число Число, лицо Лицо, время Время, залог Залог );
@Часть_речи существительное ( число Число, падеж Падеж, род &Род );
@Часть_речи прилагательное  ( число Число, падеж Падеж, род  Род );
@Часть_речи наречие         ( степень С );
@Часть_речи местоимение     ( число Число, лицо Лицо, падеж Падеж, род &Род );

// ------- структуры из словаря ----------
@Структура1 существительное1( число &Число, падеж &Падеж, род  Род ) =
            прилагательное  ( Число, Падеж,  Род )
            существительное ( Число, Падеж, &Род );

@Структура1 существительное2( число &Число, падеж &Падеж, род  Род ) =
            существительное ( Число, Падеж, &Род )
            @1 ;
@Структура1 существительное3( число &Число, падеж &Падеж, род  Род ) =
            прилагательное  ( Число, Падеж,  Род )
            существительное( Число, Падеж, &Род )
            @1 ;

// ------- форматы и файлы ----------------
@Формат глагол формат1
{
   залог=Неопредел @Слово
   залог=Актив
   Время=настоящее
   Число=Ед   Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово
   Число=Мн   Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово

   Время=прошлое
   Число=Ед   Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово
   Число=Мн   Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово

   Время=будущее
   Число=Ед   Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово
   Число=Мн   Лицо=1е @Слово  Лицо=2е @Слово  Лицо=3е @Слово

   залог=Повелит @Слово
}
@Формат существительное формат2
{
   Число=Ед
   Падеж=И @Слово  Падеж=Р @Слово
   Падеж=Д @Слово  Падеж=В @Слово
   Падеж=Т @Слово  Падеж=П @Слово

   Число=Мн
```

```
      Падеж=И @Слово  Падеж=Р @Слово
      Падеж=Д @Слово  Падеж=В @Слово
      Падеж=Т @Слово  Падеж=П @Слово
   }
   // ---- пример использования препроцесора C++
   // ---- повторяющуюся последовательность
   // ---- не обязятельно писать несколько раз
   # define pril_chislo_rod \
         Число=Ед        \
         Род=М @Слово    \
         Род=Ж @Слово    \
         Род=С @Слово    \
         Число=Мн @Слово \

   @Формат прилагательное формат3
   {
      Падеж=И pril_chislo_rod
      Падеж=Р pril_chislo_rod
      Падеж=Д pril_chislo_rod
      Падеж=В pril_chislo_rod
      Падеж=Т pril_chislo_rod
      Падеж=П pril_chislo_rod
   }
   @Формат местоимение формат4
   {
      Падеж=И @Слово
      Падеж=Р @Слово
      Падеж=Д @Слово
      Падеж=В @Слово
      Падеж=Т @Слово
      Падеж=П @Слово
   }

   @Файлы глагол           формат1 { dicts_form/d_verb } ;
   @Файлы существительное  формат2 { dicts_form/d_substantive } ;
   @Файлы прилагательное   формат3 { dicts_form/d_adjective };
   @Файлы местоимение      формат4 { dicts_form/d_pronoun } ;
}
/*****************************************************************************/
//                         словарь перевода
/*****************************************************************************/
@Словари_перевода
{
   @Формат_перевод формат1
   {
      @Части_речи_источника
      {
         v  глагол ;
         s  существительное ;
         S1 существительное1 ;  // структура из словаря
         S2 существительное2 ;  // структура из словаря
```

```
      SS гр_сущ1 ;           // произвольная структура из словаря
      s  прилагательное ;
      d  наречие ;
      O  местоимение ;
   }
   @Части_речи_приемника
   {
      v  глагол ;
      m  модальный_глагол ;
      s  существительное ;
      S1 существительное1 ;  // структура из словаря
      S2 существительное2 ;  // структура из словаря
      S3 существительное3 ;  // структура из словаря
      SS гр_сущ1 ;           // произвольная структура из словаря
      a  прилагательное ;
      d  наречие ;
      O  местоимение ;
   }
 }
 // ------ файлы, содержащие словари переводов ------
 @Файлы_препроцессоров  формат1 {  dicts/base_pre  };
 @Файлы_переводов       формат1 {  dicts/base_dict };
 @Файлы_постпроцессоров формат1 {  dicts/base_post };
}
/*****************************************************************************/
//                       правила переводов
/*****************************************************************************/
@Переводы
{
// -------------------------------------------------------------
//             Вот как делается необязательный член
// -------------------------------------------------------------
@Перевод
{  @Структура уточняющий_оборот =
             @O[wich]        // "жесткая" константа
             неполная_фраза ;
   @Структура уточняющий_оборот( число &Число, падеж &Падеж ) =
             существительное[который]( Число, Падеж ) // мягкая константа
             неполная_фраза ;
}
@Перевод
{  @Выбор    _уточняющий_оборот =
             уточняющий_оборот
             @00 ;          // пустой выбор
   @Выбор    _уточняющий_оборот( число &Число, падеж &Падеж ) =
             уточняющий_оборот( Число, Падеж )
             @00 ;
}
// -------------------------------------------------------------
//             Вот как делается описание однородных членов
// -------------------------------------------------------------
```

```
@Перевод
{  @Структура  определение1 =
               прилагательное
              _определение ;
   @Структура  определение1( число &Число, падеж &Падеж, род  &Род ) =
               прилагательное( Число, Падеж, Род )
              _определение   ( Число, Падеж, Род ) ;
}
@Перевод
{  @Выбор     _определение =
               определение1
               @00 ;
   @Выбор     _определение( число &Число, падеж &Падеж, род  &Род ) =
               определение1( Число, Падеж, Род )
               @00 ;
}
// -------------------------------------------------------------------
//                      группа существительного
// -------------------------------------------------------------------
@Перевод
{  @Выбор     артикль  = @0[a] @0[an] @00 ;
   @Выбор     артикль  = @00   @00    @00 ;
}
@Перевод
{  // ------- существительное, с его определениями --------
   @Структура гр_сущ1( число Число ) =
              артикль
              определение
              существительное
              _уточняющий_оборот ;
   @Структура гр_сущ1( число &Число, падеж &Падеж ) =
              определение( Число, Падеж, Род )
              существительное( Число, Падеж, &Род )
              _уточняющий_оборот( Число, Падеж ) ;
}
@Перевод
{  // ------- устойчивее выражения из словаря -------------
   @@Выбор1   составное_существительное =
              существительное1
              существительное2 ;
   @@Выбор1   составное_существительное( число &Число, падеж &Падеж ) =
              существительное1( Число, Падеж, &Род )
              существительное2( Число, Падеж, &Род )
              существительное3( Число, Падеж, &Род );
}
@Перевод
{  // ------- все, что может играть роль существительного --------
   @Выбор     гр_сущ( число Число, лицо Лицо ) =
              гр_сущ1( &Число )
              местоимение( &Число, &Лицо )
              составное_существительное( &Число ) ;
```

```
   @Выбор     гр_сущ( число &Число, лицо &Лицо, падеж &Падеж ) =
              гр_сущ1( Число, Падеж )
              местоимение( Число, Лицо, Падеж, &Род )
              составное_существительное( Число, Падеж );
   @Таблица( @Выбор лицо Лицо = лицо Лицо )
   {  1 @0 = 3е ;
      2 1е = 1е ;
      2 2е = 2е ;
      2 3е = 3е ;
   }
   @Таблица( число Число = число Число )
   {  Ед  = Ед ;
      Мн = Мн ;
   }
}
@Перевод
{  // ------- просто обертка, чтобы сделать удобные параметры ---------
   @Структура группа_существительного =
              гр_сущ   ( &Число ,&Лицо ) ;
   @Структура группа_существительного( падеж &Падеж ) =
              гр_сущ   ( &Число1, 3е, Падеж );
}
// -----------------------------------------------------------------------
//                            сказуемое
//     Как правила сказуемое делается таким образом:
//     Много-много вариантов, которые обьединяются одним выбором
// -----------------------------------------------------------------------
@Перевод
{  // ------- пример использования констант ------
   @Выбор     _Not = @0[not] @00 ;
   @Выбор     _Не =  @0[не]  @00 ;
}
@Перевод
{  @Структура сказуемое_настоящее =
              глагол _Not ;
   @Структура сказуемое_настоящее( лицо &Лицо, число &Число ) =
              _Не глагол( Лицо, Число, настоящее, Актив ) ;
}
@Перевод
{  @Структура сказуемое_прошлое =
              @0[have] _Not глагол ;
   @Структура сказуемое_прошлое( лицо &Лицо, число &Число ) =
              _Не глагол( Лицо, Число, прошлое, Актив );
}
@Перевод
{  @Структура сказуемое_будущее =
              @0[will] _Not глагол ;
   @Структура сказуемое_будущее( лицо &Лицо, число &Число ) =
              _Не глагол( Лицо, Число, будущее, Актив ) ;
}
@Перевод
```

```
{  @Выбор     сказуемое =
              сказуемое_настоящее
              сказуемое_прошлое
              сказуемое_будущее ;
   @Выбор     сказуемое( лицо &Лицо, число &Число ) =
              сказуемое_настоящее( Лицо, Число )
              сказуемое_прошлое  ( Лицо, Число )
              сказуемое_будущее  ( Лицо, Число ) ;
}
// ----------------------------------------------------------------------------
//         структуры, нужные для второстепенных членов предложения
// ----------------------------------------------------------------------------
@Перевод
{  @Структура предложный_оборот1 = @0[from] группа_существительного ;
   @Структура предложный_оборот1 =          группа_существительного( Р );
}
@Перевод
{  @Структура предложный_оборот2 = @[to] группа_существительного ;
   @Структура предложный_оборот2 =       группа_существительного( Д );
}
@Перевод
{  @Структура предложный_оборот3 = @0[by] группа_существительного ;
   @Структура предложный_оборот3 =        группа_существительного( Т );
}
@Перевод
{  @Выбор     предложный_оборот =
              предложный_оборот1
              предложный_оборот2
              предложный_оборот3 ;
   @Выбор     предложный_оборот =
              предложный_оборот1
              предложный_оборот2
              предложный_оборот3 ;
}
// ----------------------------------------------------------------------------
//           Всякие разные типы фраз. Делаются так-же как сказуеме:
//           то есть много-много вариантов, и обьединение в конце.
// ----------------------------------------------------------------------------
@Перевод
{  @Структура фраза_повествовательная =
              подлежащее
       сказуемое
       предложный_оборот<1>
       предложный_оборот<2> ;
   @Структура фраза_повествовательная =
              подлежащее( &Лицо, &Число )
              сказуемое( Лицо, Число )
              предложный_оборот<2>
              предложный_оборот<1> ;
}
@Перевод
```

```
{  // ------- общий вопрос ------------------
   @Структура фраза_вопросительная =
              сказуемое
              подлежащее
              предложный_оборот<1>
              предложный_оборот<2> ;
   @Структура фраза_вопросительная =
              подлежащее( &Лицо, &Число )
              сказуемое( Лицо, Число )
              @0[ли]
              предложный_оборот<2>
              предложный_оборот<1> ;
}
@Перевод
{  // ------- вопрос к подлежащему ---------
   @Структура фраза_вопросительная1 =
              вопрос_слово
              сказуемое( %3е, %Ед )
              предложный_оборот<1>
              предложный_оборот<2> ;
   @Структура фраза_вопросительная1 =
              вопрос_слово
              сказуемое( 3е,Ед )
              предложный_оборот<2>
              предложный_оборот<1> ;
}
@Перевод
{  // ------- вопрос к второстепенному члену предложения
   @Структура фраза_вопросительная2 =
              вопрос_слово
              сказуемое
              подлежащее
              предложный_оборот ;
   @Структура фраза_вопросительная2 =
              вопрос_слово
              подлежащее( &Лицо, &Число )
              сказуемое( Лицо, Число )
              предложный_оборот ;
}
@Перевод
{  @Выбор     фраза
              фраза_повествовательная
              фраза_вопросительная
              фраза_вопросительная1
              фраза_вопросительная2 ;
   @Выбор     фраза
              фраза_повествовательная
              фраза_вопросительная
              фраза_вопросительная1
              фраза_вопросительная2 ;
}
```

```
// ------------------------------------------------------------------
//            сложноподчиненные фразы
// ------------------------------------------------------------------
@Перевод
{  @Структура сложная_фраза_времени =
              @0[When] фраза_повествовательная @0[then] фраза ;
   @Структура сложная_фраза_времени =
              фраза @0[когда] фраза_повествовательная ;
}
@Перевод
{  @Структура сложная_фраза_условия =
              @0[If] фраза_повествовательная @0[then] фраза ;
   @Структура сложная_фраза_условия =
              фраза @0[если] фраза_повествовательная ;
}
// ---- все_все - это обязательно присутствующая конструкция
// ---- также, как функция main в языке C++
@Перевод
{
   @Выбор все_все =
          фраза
          сложная_фраза_времени
          сложная_фраза_условия ;
   @Выбор все_все =
          фраза
          сложная_фраза_времени
          сложная_фраза_условия ;
}
}
```