Open source Icelandic resource grammar in GF

Master's thesis in Computer science algorithms, languages and logic

Bjarki Traustason

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis 2017

Open source Icelandic resource garmmar in GF

Bjarki Traustason

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Computer Science
Chalmers University of Technology
Gothenburg, Sweden 2017

Open source Icelandic resource grammar in GF
BJARKI TRAUSTASON

Supervisor: Krasimir Angelov, Computer Science and Engineering Department
Examiner: Aarne Ranta, Computer Science and Engineering Department

Master's Thesis 2017
Computer Science and Engineering Department
Computre Science
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Stock picture of a mushroom.

Typeset in LaTeX
Gothenburg, Sweden 2017

Open source Icelandic resource grammar in GF
BJARKI TRAUSTASON
Computer Science and Engineering Department
Chalmers University of Technology

Abstract

This thesis marks out the implementation of an open source Icelandic resource grammar using the Grammatical Framework. The grammatical framework, GF, is a grammar formalism for multilingual grammars based on using language independent semantics that are represented by abstract syntax trees. The GF Resource Grammar Library is a set of natural languages implemented as resource grammars that all have a shared abstract syntax. Icelandic is the only official language of Iceland. Icelandic is a Germanic language of high morphological complexity. This thesis details some of the more interesting aspects of the grammar from the word forms of single words to how different words react to each other in a set forming phrases and sentences.

Keywords: Language Technology, GF(Grammatical Framework), Natural language processing, Functional programming, Icelandic.

Acknowledgements

I want to thank my supervisors, Krasimir Angelov and Inari Listenmaa, for all their help and guidance in the project. Thanks to my examiner Aarne Ranta for giving me resources and allowing me to attend his lectures and as well as to do this project. Thanks to Eiríkur Rögnvaldsson for giving me linguistic resources and answering my questions.
Special thanks to my girlfriend Stefanía for having my back and being there for me.

Bjarki Traustason, Gothenburg, November 2017

Contents

List of Figures

List of Tables

1
Introduction

1.1 Aim and outline of the project

GF (Grammatical Framework [1]) is a grammar formalism for multilingual grammars and their applications. GF is a typed functional programming language highly influenced by Haskell. The implementation of GF has previously not been conducted for Icelandic grammar in the manner as the following project.

Aim of this project The main goal and aim of this project is implementing an open source Icelandic resource grammar using the Grammatical Framework, and include it in the GF Resource Library. That way it will be freely available for usage in other projects.

Outline of the remainder of this paper In this chapter (1) we will introduce the projects components and give theoretical background for understanding the implementation of the Icelandic resource grammar. First the Grammatical framework (GF) and the GF Resource Library are described. Then short examples are given of how the Grammatical Framework can be used to implement grammars.

Chapter 2 begins by an overview of how a general resource grammar is structured within the Resource Grammar Library. The implementation of the Icelandic resource grammar, by using the Grammatical Framework, is described by a detailed description. Furthermore, when each component is listed a description of the Icelandic syntax and morphology of that component is covered as well.

Chapter 3 is devoted to the testing of the resource grammar. An evaluation is given on the work along with discussion on its coverage.

In chapter 4 a discussion on what future work is needed for the grammar along with some speculations on some ethical considerations that might be related to the project. Finally a conclusion of the project is presented.

1.2 The Grammatical Framework

Abstract and concrete syntax A GF grammar is made up of an abstract syntax and at least one concrete syntax. The abstract syntax of a grammar defines a set

of abstract syntax trees representing the semantically relevant language structure. The concrete syntax defines a relation between abstract syntax trees and concrete structures, i.e. defining how abstract syntax trees are mapped from and to strings. An abstract grammar can be implemented by a set of concrete grammars, each representing a language.
This separation between abstract and concrete syntax is one of the main features of a GF grammar. The separation is based on the idea that type checking and semantics are more relevant on the abstract level but syntax details on the concrete level. Examples of an abstract and a couple of concrete syntaxes given in section 1.3 to explain this separation further.

Parsing and linearization A GF grammar can be used for both parsing and generating. The process of generating a string from an abstract syntax tree is called linearization, and producing an abstract syntax tree from a string is called parsing. If the grammar is ambiguous several abstract syntaxes will be produced.

Resource grammars and the GF Resource Grammar Library A resource grammar is an almost complete linguistic description of a specific language. It describes how to construct phrases and sentences, and how to decline words in the specific language.

The GF Resource Grammar Library [2] is a set of natural language resource grammars in GF. Currently the GF Resource Library covers the fundamental morphology and syntax of about 30 natural languages[1]. All these different languages, implemented as concrete syntaxes, are built upon a common abstract syntax. The grammars are thus in a strong sense parallel to each other. This gives way for opportunities in many language processing tasks, e.g., machine translation, multilingual generation and spoken dialogue systems.
The library can be roughly divided into morphological and syntactical components. The morphological component is different for different languages, since it regards the inflection mechanisms of the different languages. The syntactical component displays a stronger parallelism since all languages in the library have a common representation of syntactic structures and structural words.

Application grammars Application grammars can have the same, or similar, structure as resource grammars but are tailored for a specific applications. Such applications can be written mathematical exercises, or dialogue systems. Each application has a specific domain which makes it easier to guarantee correct translations. A resource grammar, as stated before, is an almost complete description of a specific language. An application grammar can thus be viewed as a resource grammar restricted to some specific domain. Intuitively the components of a resource grammar can be reused in an application grammar where they are restricted.
Both GF and the GF Resource Grammar Library are open-source. GF grammars can be compiled into portable grammar format (PGF), supported by Java [2], JavaScript

[1] http://www.grammaticalframework.org/lib/doc/status.html
[2] https://github.com/GrammaticalFramework/JPGF

and Haskell libraries, and used in software components. Using the GF Resource Grammar Library is thus a very powerful tool for building application grammars.

1.3 Grammars in GF

Let us now look at a small GF grammar. The grammar is centered around plants and is made for making comments about them. For the sake of simplicity the grammar is able to produce only a few phrases on a couple of plants. Since the abstract and concrete syntax are separated, we start with the abstract syntax.

Abstract syntax Like stated before the abstract syntax defines the set of abstract syntax trees that represent the semantically relevant language structure. In our plant based example, we define in the abstract syntax how we want to model semantically the phrases we wish to be able to make about the plants. These definitions are independent of language and therefore of all language dependent features, e.g., number agreement within a phrase is not implemented here but in the concrete syntax. Thus the resulting abstract syntax, shown below, defines what meanings can be expressed about the plants by the grammar.

Listing 1.1: Example abstract syntax

```
abstract Plants = {
    flags
            startcat = Comment ;
    cat
            Comment ; TPlant ; Plant ; Quality ;
    fun
            Pred : TPlant -> Quality -> Comment ;
            This, These : Plant -> TPlant ;
            Very : Quality -> Quality ;
            Pine, Rose : Plant ;
            Big, Fragrant : Quality ;
}
```

Like any module in GF, the abstract syntax above is composed of two main parts:

- The module header that shows the type of module it is along with its name, here abstract and Plants.
- The module body that is a set of judgements.

Judgements in GF are definitions and/or declarations. Furthermore, every judgement introduces a name which is available both within the module it was defined and/or declared and within all modules where its module is extended or opened. The Plants abstract syntax is made of three forms of judgements: *flags*, *cat*, and *fun*.
Flag definitions, *flags*, sets values to flags that are to be used when compiling or using the module. Here the flag definition *startcat* selects the start category for parsing and generation.

Category declarations, *cat*, declare what categories, i.e. the types of trees, there are in the syntax. Here four categories are declared: *Plant*, *Quality* (of a plant), *TPlant*, and *Comment*.
Function declarations, *fun*, declare what tree building functions, i.e. the syntactic constructors, there are in the abstract syntax. Here we declare two kinds of plants, *Pine* and *Rose*, along with two possible qualities they can be described with, *Fragrant* and *Big*. The function *Very* works much like the intensifier very does in English, intensifying qualities of plants. Functions *This* and *That* form a demonstrative, i.e. a specific plant, from a kind of plant. Lastly the function *Pred* forms a comment, i.e. a phrase, given a specific plant and a quality.

Listing 1.2: Example of an abstract syntax tree

```
Pred (This Rose) (Very (Very Fragrant))
```

An example of an abstract syntax tree produced by the Plants abstract syntax is given above and a more "human friendly" visualized version is shown below.

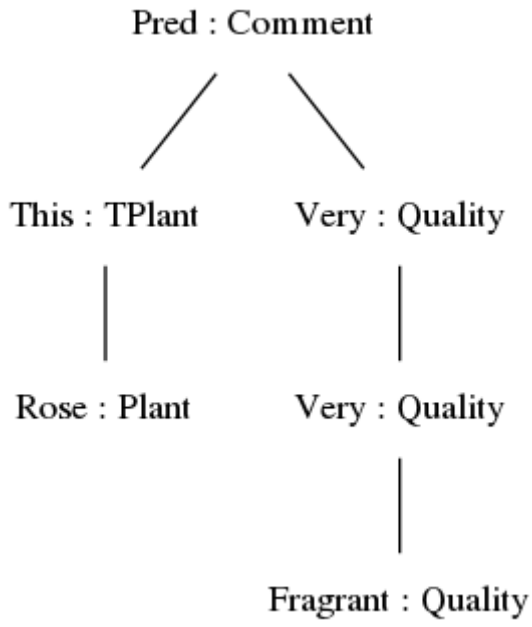

Figure 1.1: A visual representation of an abstract syntax tree.

Concrete syntax We now have a set of abstract syntax trees defined by the abstract syntax. The concrete syntax, as stated before, defines how abstract syntax trees are mapped from an to strings - for a specific language. We can thus implement the abstract syntax above by two distinct concrete syntaxes corresponding to two distinct languages, e.g., English and Icelandic.

Starting with English, shown below, two types of judgements are needed for a concrete syntax that is an implementation of the Plants abstract syntax, namely *lincat*

and *lin*. Linearization type definitions, *lincat*, define the linearization types of trees for each category declaration of the abstract syntax. Linearization rules, *lin*, define the lilnearization functions used for linearizing trees formed by the function declarations of the absract syntax.
Here we have of *lincat*'s: *Kind*, *Quality*, *Plant*, and *Comment*, and of *lin*'s: *Big*, *Fragrant*, *Pine*, *Rose*, *Very*, *This*, *These*, and *Pred*.
Plant's correspond to nouns. Nouns in English inflect in number depending context, singular or plural. We therefore need a parameter for numbers, that indicates if the context is singular or plural so a correct word form is used. Another judgment is needed in addition to the ones defined above for this parameter definition, namely *param*. With this new parameter we can define *Plant* and with it the linearization rules for *Pine* and *Rose* as inflection tables for the words "pine" and "rose" respectively.
Quality (qualities of plants) corresponds to adjectives. In English adjectives do not inflect in number and since no comparison is present in our example, therefore a simple string representation is sufficient for *Quality*.
The implementations for *Fragrant* and *Big* are then straight forward, and linearization rule *Very* is implemented by adding "very" to the beginning of the token list given by the function argument.
Furthermore *TPlants* correspond roughly to noun phrases with *This* being linearized in a similar way as *Very*. Namely by adding "this" to beginning of the token list given by singular form of the function argument. *These* is done in the same way but the context is plural.
A comment then is equivalent of a sentence. But to form a sentence a verb is needed. This is solved here by defining a copula as an operation definition. Operation definitions, *oper*, is a type of judgment in GF that can be viewed as helper functions that have no equivalence in the abstract syntax.

Listing 1.3: English concrete syntax

```
concrete GardenEng of Garden = {
    param
            Number = Sg | Pl ;
    lincat
            Kind = { s : Number => Str } ;

            Quality = { s : Str } ;

            Plant = { s : Str ; n : Number } ;

            Comment = { s : Str } ;
    lin
            Pine = { s = table {
                Sg => "pine" ; Pl => "pines" }
            } ;

            Rose = { s = table {
                Sg => "rose" ; Pl => "roses" }
```

```
    } ;

    Big = { s = "big" } ;

    Fragrant = { s = "fragrant" } ;

    Very quality = { s = "very" ++ quality.s } ;

    This plant = {
        s = "this"
            ++ plant.s ! Sg ;
        n = Sg
    } ;

    These plant = {
        s = "these"
            ++ plant.s ! Pl ;
        n = Pl
    } ;

    Pred plant quality = {
        s = plant.s
            ++ copula ! plant.n
            ++ quality.s
    } ;

    oper copula : Number => Str = table {
        Sg => "is" ; Pl => "are"
    } ;
}
```

With a concrete syntax for our abstract syntax we can now linearize the example abstract syntax tree given in 1.3

```
Plants> l Pred (This Rose) (Very (Very Fragrant))
this rose is very very fragrant
```

We can also parse a comment to form an abstract syntax tree as shown below

```
Plants> p "this pine is very big"
Pred (This Pine) (Very Big)
```

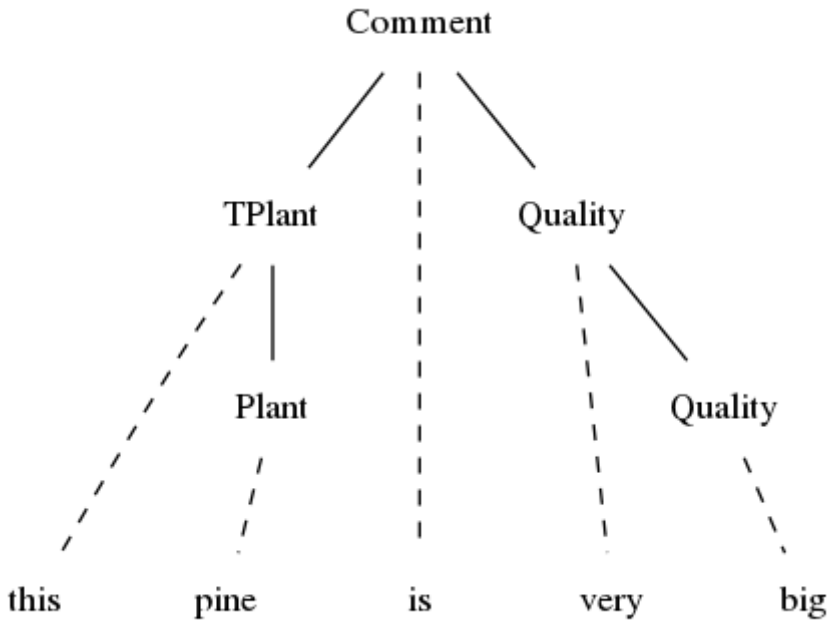

Figure 1.2: A visual representation of a parse tree.

Following a similar procedure to implement the Icelandic concrete syntax as we used for the English concrete syntax.
Icelandic has a much more complex inflection system, both for nouns and adjectives, but for this application the linearization type definitions as done for English are sufficient. Apart from qualities, they must inflect in number as adjectives do in Icelandic. A detailed description of adjectives, nouns, and etc, in Icelandic will be given in chapter 2. The implementation of the Icelandic concrete syntax is given below.

Listing 1.4: Icelandic concrete grammar of the Garden grammar

```
concrete GardenIce of Garden = {
    param
        Number = Sg | Pl ;
    lincat
        Quality = { s : Number => Str } ;

        Kind = { s : Number => Str} ;

        Plant = {s : Str; n : Number} ;

        Comment = { s : Str } ;
    lin
        Pine = {s = table {
            Sg => "fura"; Pl => "furur"}
        };
```

```
    Rose = {s = table {
        Sg => "rós" ; Pl => "rósir"}
    };

    Big = {s = table {
        Sg => "stór"; Pl => "stórar"}
    };

    Fragrant = {s = table {
        Sg => "ilmandi"; Pl => "ilmandi"}
    };

    Very quality = {
        s = \\n => "mjög"
            ++ quality.s ! n
    } ;

    This plant = {
        s = "þessi"
            ++ plant.s ! Sg ;
        n = Sg
    } ;

    These plant = {
        s = "þessar"
            ++ plant.s ! Pl ;
        n = Pl
    } ;

    Pred plant quality = {
        s = plant.s
            ++ copula ! plant.n
            ++ quality.s ! plant.n
    } ;

    oper copula : Number => Str = table {
        Sg => "er" ; Pl => "eru"
    } ;
}
```

Now with two concrete implementations of the example abstract syntax, we can translate comments between the languages. This is done by parsing a comment in one language into an abstract syntax tree. This abstract syntax tree is then used to linearize into a comment in the other language.

2

Implementation

This chapter is devoted to the implementation of the Icelandic resource grammar in the Grammatical Framework.

In the first section we begin by describing the structure of a general resource grammar in the Resource Grammar Library. Main modules are introduced and are given a high-level description of their functionality within a Resource Grammar.

In the remainder sections of the chapter we give descriptions of Icelandic morphology and simultaneously describe the corresponding parts of the resource grammar. The description of the implementation is partition by rules related to noun phrases, verb phrases, and whole sentences and clauses.

2.1 Structure of the resource grammar

The Icelandic resource grammar follows the same module structure as other implemented resource grammars in the Resource Grammar Library. The modular structure, for the main modules and their dependencies, of a GF resource grammar is given in figure 2.1.

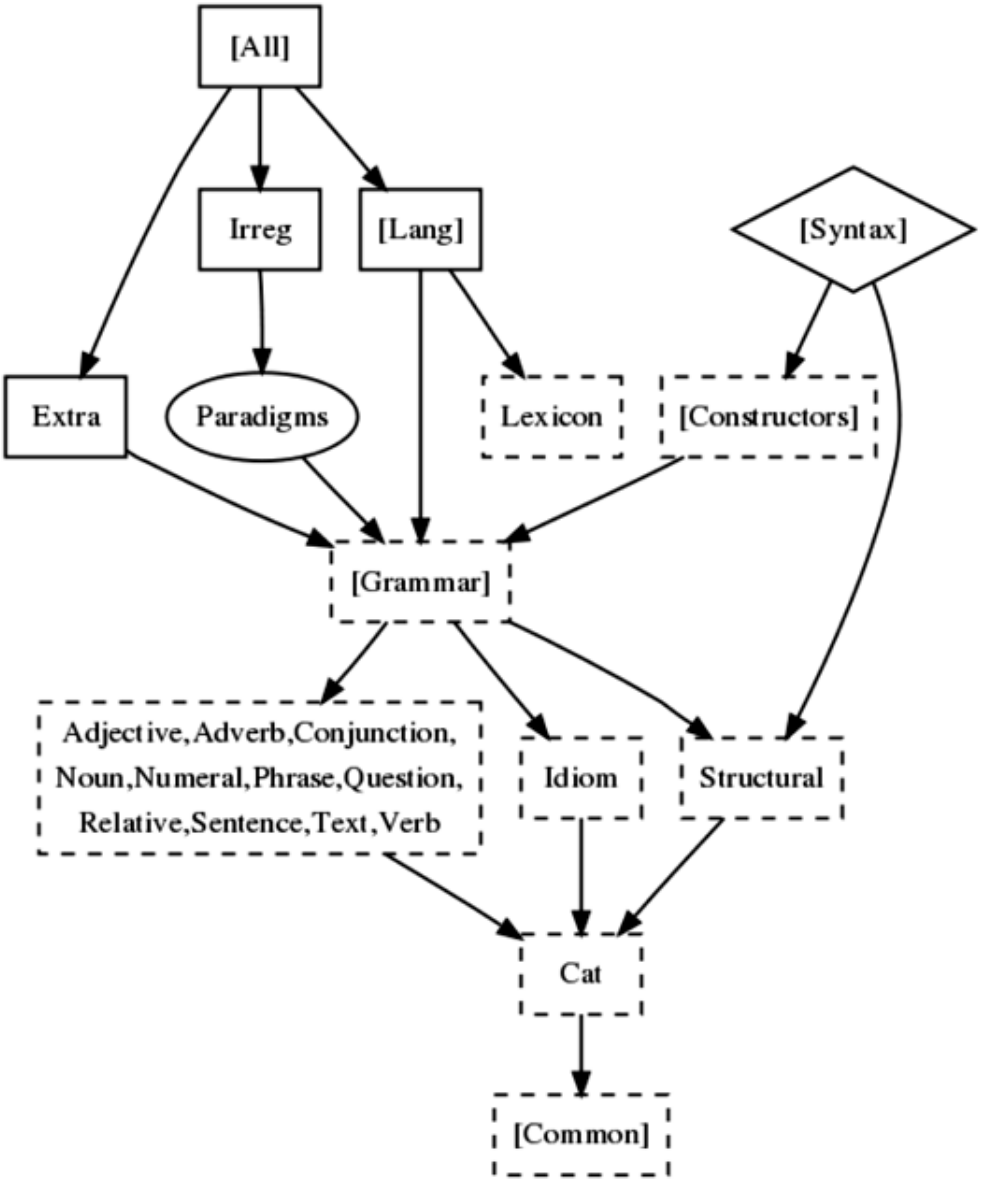

Figure 2.1: The main modules of a resource grammar[1]

In figure 2.1 the following information is contained :

- API modules are denoted with solid contours
- Internal modules are denoted with dashed contours
- Abstract and concrete pairs are denoted with rectangles
- Resources and instances are denoted with ellipses
- Interfaces denoted with diamonds
- Already given and mechanically produced denoted by having the name in brackets

The last group of modules itemized here above are not implemented manually by the resource grammarian. Since some of them are already given and others are produced mechanically. These modules are :

- all abstract modules except Extra and Irreg
- concrete of Common, Grammar, Lang, All
- resources Constructors and Syntax

The modules that have to be implemented manually by the resource grammarian, and thus the main focus of this project, are then :

- Concrete syntaxes of the row from Adjective to Structural
- Concrete syntaxes of Cat and Lexicon
- The resource module Paradigms
- Abstract and concrete of Extra and Irreg

Furthermore, the module Res (Resource) and the auxiliary module Morpho (Morphology), albeit not being shown in the figure 2.1, need to be implemented by the resource grammarian. These modules contain language specific parameter types and morphology.
Below is a summary of some of the module roles :

- Paradigms contains morphological paradigms needed to build a lexicon
- Irreg contains irregularly syntactic inflected verbs
- Extra contains extra syntactical constructs that are specific to the implemented language
- Idiom contains idiomatic expressions
- Structural is lexicon of structural words, e.g., determiners
- Lexicon contains test lexicon of content words, e.g., nouns
- Cat contains the type system common to languages, e.g., type defintion of nouns (N)

The implementation part of the most considerable importance is arguably the implementation of the so called phrase category modules. These are the ten modules in the big box in figure 2.1, i.e. Adjective to Verb. Each of them defines the constructors for one, or more, related part of speech.

Functors In GF a functor is a module-level function that takes instances of interfaces as arguments and outputs modules. An interface is a module itself and similar to a resource, but containing only the *oper* types and not their definitions.
For a group or family of related languages much of the grammar is shared between them, i.e., it is the same partly or wholly for each language member of the group. A functor can be used to take care of the shared parts of the grammar modules. More precisely, it allows the language group to share syntactic constructs which are in common and only write what differs for each language. An example of a language group or family implemented by usage of functors are the Continental Scandinavian languages. This functor is refered to as the scandinavian functor and includes : Danish, Norwegian, and Swedish.

Despite being related to the Continential Scandinavian languages the similiarites where not enough to justify implementation via the Scandinavian functor.

2.2 Noun Phrases

A noun phrase, *NP*, is a set of words that can be a combination of determiners, nouns, pronouns, adjective phrases and relative clauses. A noun phrase can function as a subject, object or complement within a sentence. In the resource grammar the head of this phrase is a noun (or a pronoun) such as "hús" ("house"). This head can then be further modified, e.g., by an adjective phrase, *AP*, such "blár" ("blue") and form:

(1) Blátt hús
Blue house

A set of words like example 1 are referred to as common nouns, *CN*, in the resource grammar. Common nouns, such as example 1, can be considered and used as noun phrases themselves. But it is also possible modify them even further, e.g., by a determiner, *Det*, such as the demonstrative pronoun "þessi" ("this") or by an article to form examples 2 and 3 respectively.

(2) **Þetta** bláa hús
This blue house

(3) Bláa hús**ið**
The blue house

A more detailed description of individual components of the noun phrase such as nouns, common nouns, adjective phrases and determiners, will be carried out in the next subsections, or more precisely in subsections 2.2.1, 2.2.2, 2.2.3, and 2.2.4 respectively.

Listing 2.1: The record type for noun phrases

```
oper
    NP : Type = {
        s : NPCase => Str ;
        a : Agr ;
        isPron : Bool
    } ;
    Agr : PType = {g : Gender ; n : Number ; p : Person} ;
param
    NPCase = NCase Case | NPPoss Number Gender Case ;
```

Most of these components are linearised into strings and kept in the main *s* field when the noun phrase is formed. The *s* field is then a record from a NPCase to String. NPCase can either be Just *Case* or be dependent of *Number* and *Gender* as well. The latter is needed in a possessive context.

The noun phrase also contains information about its Gender, number, and person which verb phrases must agree with when combined together to form a sentence. This information is referred to as agreement, *Agr*, and is kept in the *a* field. Furthermore, it contains a field that indicates whether or not it is a Pronoun. This is because unstressed pronouns as objects in verb phrases undergo the Object Shift [6]. A further discussion on the Object Shift is carried out in section 2.2 on verb phrases.

2.2.1 Nouns

Icelandic nouns inflect in two numbers, singular and plural, and four cases, nominative, accusative, dative, and genitive. They inherit a grammatical gender, masculine, feminine and neuter. [3] An example of an Icelandic noun the word forms of the masculine noun "maður" are shown in table 3.1 below.

Table 2.1: Inflectional table for the masculine noun "maður" ("man").

	Singular		Plural	
Case	Without article	With article	Without article	With article
Nominative	maður	maðurinn	menn	mennirnir
Accusative	mann	manninn	menn	mennina
Dative	manni	manninum	mönnum	mönnunum
Genitive	manns	mannsins	manna	mannanna

The implementation of nouns in the resource grammar is rather straight forward, as seen in the record type below compared to the inflection table above. But word forms of nouns formed by the definite article must be taken in to account. The definite article is suffixed on the noun and inflects with it. In subsection 7 a more detailed discussion on the definite article is carried out.

Listing 2.2: The record type for nouns and necessary parameters.

```
oper
    N : Type = {
        s : Number => Species => Case => Str ;
        g : Gender
    } ;
param
    Case = Nom | Acc | Dat | Gen ;
    Gender = Masc | Fem | Neutr ;
    Species = Free | Suffix ;
```

2.2.2 Common nouns

The group *N* for nouns is not really used in the resource grammar for more than being an inflection table for nouns. *N* can be viewed as a group for simple nouns

that are turned into common nouns, CN, for usage within a noun phrase.

Listing 2.3: The record type for common nouns.

```
lincat
    CN = {
        s : Number => Species => Declension => Case => Str ;
        comp : Number => Case => Str ;
        g : Gender
    } ;
param
    Declension = Weak | Strong ;
```

CN, as defined above, can be viewed as an extension of a simple noun, and is thus similarly defined in GF. CN can be further modified by other CN, i.e. conjoining more than one CN's together, or adjective phrases. Since common nouns can contain adjective phrases who depend on declension then information on the *Declension* must be available to the common noun. A more detailed discussion on adjective phrases and their structure is carried out in section 2.2.3.

(4) Góður maður
A good man

An example of a noun turned into a common noun and then modified by an adjective phrase is given in example 4.

Listing 2.4: Functions for constructing and modifying common nouns.

```
    UseN : N -> CN ;

    AdjCN    : AP -> CN  -> CN ;
```

CN's also contains the field *comp* that contains any additions that follow the noun. This is because of the word order in possessive constructions. The possessor, e.g. "stelpa" ("girl"), generally follows its possession, such as "bók" ("book") [5].

(5) Bók stelpu**nnar**
The girls book

(6) Bók**in** mín
*__The__ book my
My book

In example 6 a construction with a personal pronoun is shown. The same word order is used, but the possession must take the suffixed definite article [5] [6].

2.2.3 Adjective phrase

An adjective phrase is a group of words describing a noun or a pronoun within a noun phrase. In the resource grammar an adjective, A, such as "blár" ("blue"), is the head of an adjective phrase, AP. This head can be further modified, e.g., by an

ad-adjective, *AdA*, such as "**mjög** blár" ("very blue"), and by an adverb(ial), such as "alltaf blár" ("always blue").

Listing 2.5: An Example of functions to construct and modify an *AP*.

```
PositA   : A   -> AP ;

AdAP     : AdA -> AP -> AP ;
```

Icelandic adjectives have a great number of word forms. This results from three factors combined. Firstly an adjective must agree with its noun, i.e. the noun which the adjective is describing. A noun, as stated before, inherits one of three genders, and has four cases in both singular and plural. Therefore an adjective must exist in three genders and for each gender it must be in four cases in both singular and plural. Then there are the *strong* and *weak* declensions that adjectives exist in all genders, numbers and cases. Lastly, there is the comparison of adjectives. Icelandic adjectives have three degrees of comparison, positive, comparative and superlative. Comparative and superlative have different suffixes that distinct them from each other and the positive degree. The positive and superlative contain both the *weak* and *strong* declension, while the comparative has only the *weak* declension.

The implementation of adjectives, or *A* as shown below, in the resource grammar thus depends on comparison, declension, number, gender and case. Since the comparative only contains the weak declension, the parameter *AForm* is used to prevent unnecessary word forms to be kept for the comparative.

Listing 2.6: The record type for adjectives and its AForm parameter and linearization category for adjective phrases.

```
oper
    A : Type = {
        s : AForm => Str ;
        adv : Str
    } ;
param
    AForm =
          APosit Declension Number Gender Case
        | ACompar Number Gender Case
        | ASuperl Declension Number Gender Case
        ;
lincat
    AP = {s : Number => Gender => Declension => Case => Str} ;
```

Adjective phrases, *AP* as shown above, are dependent on the same variables as adjectives. Its number, gender and case agree with its noun but its declension depends on context. When the adjective modifies indefinite nouns or is predicative, the strong declension is used, and when the adjective modifies a noun that is determined the weak declension is used [9]. The declension is thus governed by the quantifiers or the determiners of the noun phrase.

2.2.4 Quantifiers and determiners

In the resource grammar there is a difference between quantifiers (*Quant*), determiners (*Det*), and predeterminers (*Predet*). Quantifiers and determiners are used to modify common nouns within noun phrases, while predeterminers are used to modify whole noun phrases.
A Quantifier inflects in number, gender and case, and inherits definiteness. That is, it has a predetermined information on whether or not the common noun will be defined from its modification, and therefore governs the declension, *Strong* or *Weak*, of the adjective phrase in the common noun. It furthermore specifies if the simple noun of the common noun should have the suffixed definite article or not.

Listing 2.7: Linearization categories for determiners and quantifiers.

```
lincat
    Det = {
            s : Gender => Case => Str ;
            pron : Gender => Case => Str ;
            n : Number ;
            b : ResIce.Species ;
            d : ResIce.Declension ;
    } ;
    Quant = {
            s : Number => Gender => Case => Str ;
            b : ResIce.Species ;
            d : ResIce.Declension ;
            isPron : Bool
    } ;
```

Of quantifiers in the GF grammar are, e.g., demonstrative pronouns, possessive pronouns, and the definite and indefinite articles.
A Determiner is defined like a quantifier, as seen above, except it does not inflect in number, but rather inherits it. Furthermore, quantifiers can be viewed as the kernels of the determiners since they are only used via conversion to determiners in the resource grammar.

(7) **Þessir** góðu menn
These good men

An a example of a quantifier converted to a determiner, and then used to modify a common noun is given in example 7 above.

Listing 2.8: Functions to construct determiners from quantifiers and to modify a common noun.

```
    DetQuant : Quant -> Num -> Det ;

    DetQuantOrd : Quant -> Num -> Ord -> Det ;

    DetCN    : Det -> CN -> NP ;
```

Since a constructed noun phrase already has a number and definiteness, modifying predeterminer only agrees with the noun phrase in number, gender and case. Therefore a predeterminer inherits neither number nor definiteness.

Listing 2.9: Linearization category for predeterminers.

```
lincat
    Predet = {
            s : Number => Gender => Case => Str
    } ;
```

Of predeterminers in the GF grammars are, e.g., some indefinite pronouns.
An example of a predeterminer modifying a already formed noun phrase could be "allir þessir góðu menn/all these good men". Where the predeterminer "allir/all" modifies the noun phrase "þessir góðu menn/these good men"

Listing 2.10: Function to modify a noun phrase with a predeterminer.

```
        PredetNP : Predet -> NP -> NP ;
```

The definite and indefinite articles in Icelandic

There is no indefinite article in Icelandic, thus the absence of an article indicates its indefiniteness [9]. The definite article on the other hand exists and can either be freestanding or as a suffix. The freestanding article is rare and can only be used when an adjective intervenes[9][6]. Both the freestanding and the suffix articles have their own inflections, and inflect like nouns in number, gender and case. The freestanding and the suffix articles cannot be used to define the same noun, furthermore, double definiteness is generally not found in Icelandic [6].The articles and their usage are displayed in the following examples:

(8) Indefinite
Hérna er hestur
Here is (a) horse

(9) Definite (suffix)
Hérna er hestur**inn**
Here is **the** horse

(10) Definite (free)
Hérna er **hinn** föli hestur
Here is **the** pale horse

The abstract syntax doesn't assume the existence of more than one form of the definite article. Therefore, using two, like is done in Icelandic, is not assumed.
To solve this situation, as described above, we introduced the parameter *Species* for quantifiers (including determiners) and nouns. *Species* then specifies if the noun affected by the quantifier has the suffixed article or is free standing. Thus *Species* can have the value $Free$ or $Suffix$. Nouns are then presented both with and without the suffixed article in their inflection tables. This is also how inflection tables for Icelandic nouns are presented in most grammar books as seen in table 3.1.

Listing 2.11: The functional implementation for the definite and indefinite articles.

```
    DefArt = {
            s = table {
                    Sg => table {
                            Masc    => caseList "hinn" ... ;
                            Fem     => caseList "hin" ... ;
t                                    Neutr   => caseList "hið" ...
                    } ;
                    Pl => table {
                            Masc    => caseList "hinir" ... ;
                            Fem     => caseList "hinar" ... ;
                            Neutr   => caseList "hin" ... ;
                    }
            } ;
            b = Suffix;
            d = Weak ;
            isPron = False
    } ;

    IndefArt = {
            s = \\_,_,_ => [] ;
            b = Free ;
            d = Strong ;
            isPron = False
    } ;
```

But this introduction of *Species* means that quantifiers still need to be assigned the value $Free$ or $Suffix$. This assignment is mutually exclusive. Therefore, only one form of the definite article can be generally used in the resource grammar, albeit both forms existing in it. Since the suffixed definite article can be used in most, if not all, situations where the freestanding definite article is used, it is the default choice in the resource grammar. The freestanding definite article is left as an extra feature, and its usage then within applications made for situations where it must occur.

2.2.5 Pronouns

Pronouns in Icelandic are usually grouped into: personal pronouns, reflexive pronouns, possessive pronouns, demonstrative, indefinite and interrogative. But in the resource grammar only the personal pronouns, and its possessive equivalences, make up the GF category for pronouns PN as defined below. This is because of the syntax oriented analysis in GF.

Listing 2.12: The record type for pronouns.

```
    Pron : Type = {
        s : NPCase => Str ;
        a : Agr
```

```
    } ;
```

Demonstrative and indefinite pronouns are classified as determiners, quantifiers or predeterminers in the resource grammar since the can determine noun phrases, e.g., "sérhver" ("every") in example 11.

(11) Sérhver fölur hestur...
Every pale horse...

Interrogative pronouns do get a category of their own, IP for their role in the module QuestionIce where the constructions of interrogative clauses is governed. Interrogative pronouns inflect like (most) other pronouns in Icelandic, in number, gender and case.
There is only one reflexive pronoun in Icelandic, namely "sig" [9]. It is the same in all gender and numbers. It is not a part of any GF category but rather has a function definition for its inflection table as defined below. But besides such conveniences of being the same for all numbers and genders, it does not technically exist in the nominative case. To solve this the personal pronoun of the subject is instead used (in the nominative) along with the indefinite pronoun "sjálfur" ("himself"). But this reflective pronoun is only applicable for 3rd person context. In the case of 1st or 2nd person, the possessive pronoun of the subject is used.

```
reflPron : Person -> Number -> Gender -> Case -> Str ;
```

2.2.6 Numerals

Icelandic numerals are, like in other Germanic languages, split into two groups cardinals and ordinals. Cardinals denote definite numbers while ordinals indicate a position within a series. Both cardinals and ordinals can be viewed as limiting adjectives, except "hundrað" ("hundred") and "þúsund" ("thousand") which are neuter nouns, and "milljón" ("million") and "billjón" ("billion") which are feminine nouns. Only the first four cardinals inflect and of them only "einn" ("one") inflects in number as well in gender and case. All other cardinals have only one word form, and all cardinals (except "einn") are inherently plural. Ordinals on the other hand inflect in number, gender and case.
Digits not being word do not inflect. A period "." is suffixed on ordinal digits to distinguish them from cardinals, e.g., "1." ("1st") and "2." ("2nd"). The definition of numerals and digits is shown below.

Listing 2.13: Type definition of numerals and digits.

```
oper
    Numeral : Type = {s : CardOrd => Str ; n : Number} ;
    Digits : Type = {s : CardOrd => Str ; n : Number} ;
param
    CardOrd = NOrd Number Gender Case
            | NCard Number Gender Case
            ;
```

Di

2.3 Verb Phrases

A verb phrase, VP, is a set of words that contains (at least one) verb and its dependants, e.g., an object (noun phrase). In the resource grammar the head of the verb phrase is a verb. Verb phrases consisting of just one verb, such as "deyja" ("die"), can be considered and used as verb phrases themselves. Then the verb is simply used to form a verb phrase:

```
UseV     : V -> VP ;
```

But it is also possible to form more complex phrases from, e.g., a transitive verb and an object such as "sjá" ("see") and "rauði svifnökkvinn" ("the red hovercraft") in example 12 below.

(12) (Ég) sé rauða svifnökkvann
(I) see the red hovercraft

In the resource grammar verb categories that can take objects, e.g., transitive verbs ($V2$ in the resource grammar) or ditransitive verbs ($V3$ in the resource grammar), form verb phrases by using $VPSlash$. That is, a $VPSlash$ is constructed from the verb and then the object is added in a separate step to form a verb phrase. $VPSlash$ is a reference to $VP\backslash NP$, i.e. a verb phrase missing a noun phrase (object), from categorical grammar. Example 12 could thus be constructed by functions listed below.

Listing 2.14: Example functions for constructing verb phrases.

```
SlashV2a : V2          -> VPSlash ;

ComplSlash : VPSlash -> NP -> VP ;
```

Verb phrases in Icelandic are in the most essential respect verb initial, i.e. they begin with a verb (an auxiliary or the main verb)[6]. The basic order within an Icelandic verb phrase is given in table 2.2 below [1].

Table 2.2: Basic order within the verb phrase

X	Main verb	indirect object	direct object	bound adverbials or predicative complements
Ég ætla að I intend to	gefa give	henni her	penna pen	í jólagjöf for christmas-present
Bjarki Bjarki	keypti bought		bók (a) book	í gær yesterday

The verb and its complements are stored in the verb phrase category, VP, as shown below.

[1] http://www.lunduniversity.lu.se/lup/publication/9d883cb9-82e2-4e88-9d55-b9c2bcc64ac3

Listing 2.15: The record type for verb phrases and its depending parameter VPForm.

```
oper
    VP : Type = {
            s        : VPForm => Polarity => Agr => {
                       fin      : Str ;
                       inf      : Str ;
                       a1       : Str * Str
            } ;
            p        : PForm => Str ;
            indObj   : Agr => Str ;
            dirObj   : Agr => Str ;
            a2       : Str ;
            indShift : Bool ;
            dirShift : Bool
    } ;

param
    VPForm = VPInf
           | VPImp
           | VPMood Tense Anteriority
           ;
```

As can be seen above the verb phrase has many components of different types. Unlike the noun phrase each component of the verb phrase is put in its place when a clause or a sentence is formed. The verb is kept in the s field. The s field includes the verbs auxiliary verb(s) (fin), the main verb itself (in fin if standing alone otherwise in inf), and the sentence adverb ($a1$). In Icelandic sentence adverb (including negation) has to follow the last finite verb of the verb phrase [6]. This is described with examples in table 2.3 below.

Table 2.3: Positions of the main and its auxiliary verbs with respect to the sentence adverb.

Subj	s.fin	s.a.p1	s.inf	s.a.p2	Obj
Hann	les				bókina
Hann	les			ekki	bókina
Hann	hefur		lesið		bókina
Hann	hefur	ekki	lesið		bókina

This separation is then necessary since the verb phrase has been given neither polarity nor tense. A more detailed discussion about tense in Icelandic and the tense system used in the GF Resource Grammar Library is carried out in subsection 2.3.1.

The Object Shift In Icelandic verb phrases the object can precede the sentence adverb in what is known as the Object Shift [6]. The Object Shift applies to pro-

nouns and full noun phrases, but only applies obligatory to unstressed pronouns [6]. The shift generally only takes place when there is only one verb form in the verb phrase, i.e. the main verb has no auxiliary verbs [6]. Examples 13 and 14 show this in its simplest form with unstressed pronouns.

(13) Ég sá hana **ekki**
I **didn't** see her

(14) Ég **hef ekki** séð hana
I **haven't** seen her

Furthermore the Object Shift also applies to conjoined pronouns [6] as is shown in example 15 below.

(15) Hún sá mig og þig **ekki**
She **didn't** see me and you

Some verbs allow two object within a verb phrase, e.g. ditransitive verbs ($V3$ in the resource grammar). The objects are then generally referred to as the indirect object and the direct object within the verb phrase. In such verb phrases the indirect object can be shifted or both the indirect and direct object can be shifted[6]. This is depicted in examples 16, 17, and 18 below.

(16) Both indirect and direct objects are unstressed pronouns
Ég gaf henni það **ekki**
I **didn't** give it to her

(17) Only the indirect object is an unstressed pronoun
Ég sendi honum **ekki** bókina
I **didn't** send him the book

(18) Only the direct object is an unstressed pronoun
Ég sagði börnunum það **ekki**
I **didn't** tell the children this

To account for this in the resource grammar the object is separated into two fields in the definition of verb phrases (listing 12 above). Namely $indObj$ and $dirObj$, representing the indirecte object and the direct object respectively. The fields $indShift$ and $dirShift$ then govern both if a shifting takes place and which objects do shift.

2.3.1 Verbs

Verbs in Icelandic, like in other Germanic languages, inflect in tenses, numbers and persons, and have voices, moods and non-finite forms (infinitive, participles). The tenses are two that can be differentiated by inflexion, the present and the past. The other tenses are constructed with auxiliary verbs. A further discussion on tense is carried out in subsection 2.3.1. Icelandic verbs, like Icelandic nouns and adjectives,

have two numbers, singular and plural. Verbs have these numbers in all moods and tenses. The moods are three, indicative, subjunctive, and imperative. The persons are three, first, second and third persons, in all tenses of the indicative, subjunctive, and partly of the imperative. There are three non-finite forms, the infinitive and the present and past participles. The past participle inflects like adjectives in the positive degree while the present participle has only one distinct word form. The Voices are three, active, middle and passive. The active and middle are distinct by different inflexional endings. The passive is formed with the auxiliary verb "að vera" (e. "to be") and the past participle of the verb in question.

Listing 2.16: The record type for verbs along with necessary parameter definitions.

```
oper
    V : Type = {
            s : VForm => Str ;
            pp : PForm => Str
    } ;
param
    Mood = Indicative | Subjunctive ;
    Voice = Active | Middle ;
    PForm =
            PWeak Number Gender Case
            | PStrong Number Gender Case ;
    VForm =
            VInf
            | VPres Voice Mood Number Person
            | VPast Voice Mood Number Person
            | VImp Voice Number
            | VPresPart
            | VSup Voice
            ;
```

The *s* field contains all the word forms apart from the past participles that are kept in the *pp* field. These fields are then records from a $VForm$ and $PForm$, respectevly, to *String*.
The passive voice in Icelandic is formed, as stated above, with the auxiliary verb "að vera" ("to be") and the past participle of the verb to be used. Therefore it is not kept in the inflection table, V, but rather constructed when needed. That is by using the word form of the present participlethat agrees with the context and the auxiliary verb function *verbBe*. The passivisation for transitive verb is shown below:

```
PassV2 V2 =
    let
            vp = predV verbBe
    in
            {
                    s = \\ten,ant,pol,agr =>
                        vf (vp.s ! ten ! ant ! pol ! agr).fin
```

```
                    (v2.pp ! PStrong agr.n agr.g Nom)
                    (negation pol) ;
                ...
        } ;
```

Only word forms in the *Indicative* and *Subjunctive* moods inflect in tense. $VPres$ and $VPast$ indicate the present and past tense respectively. The Imperative mood, $VImp$ only inflects in number and is only found in the second person.

Tense

Traditionally tenses in Icelandic have been described as eight. Of these are six originally based on Latin morphology, i.e., the six tenses that Latin is traditionally described with (present, past, perfect, pluperfect, future, and perfect future). [7] In Icelandic only two are simple tenses,past and present as stated before, and the others are constructed with the auxiliary verbs "hafa" ("have") and "munu" ("will")[3]. The remaining two collective tenses are results of taking the past tense of auxiliary verb "munu" in the future and perfect future. An overview of these tenses is given in table 2.4 below.

Table 2.4: Morphological and collective tenses of the Icelandic verb "berja" ("beat").

Present	ég ber	Past	ég barði
Perfect	ég hef barið	Pluperfect	ég hafði barið
Future	ég mun berja	Perfect Future	ég mun hafa barið
Present Conditional	ég myndi berja	Past Conditional	Ég myndi hafa barið

The GF Resource Grammar Library uses a combination of anteriority (simultaneous and anterior) and temporal order (present, past, future, and conditional) to describe tense. This, along with polarity (positive and negative), gives a total of 16 tense forms that are provided by the GF Resource Grammar Library. An overview of the 16 possible tense forms is given in table 2.5 here below.

Table 2.5: Tense system in the Resoure Grammar Library along with Icelandic equivalences.

Tense	Anteriority	Polarity	Example	Description
Present	Simultaneous	Positive	ég sef	Present
Present	Simultaneous	Negative	ég sef ekki	
Present	Anterior	Positive	ég hef sofið	Perfect
Present	Anterior	Negative	ég hef ekki sofið	
Past	Simultaneous	Positive	ég svaf	Past
Past	Simultaneous	Negative	ég svaf ekki	
Past	Anterior	Positive	ég hafði sofið	Pluperfect
Past	Anterior	Negative	ég hafði ekki sofið	
Future	Simultaneous	Positive	ég mun sofa	Future
Future	Simultaneous	Negative	ég mun ekki sofa	
Future	Anterior	Postive	ég mun hafa sofið	Perfect Future
Future	Anterior	Negative	ég mun ekki hafa sofið	
Conditional	Simultaneous	Positive	ég myndi sofa	Present Conditional
Conditional	Simultaneous	Negative	ég myndi ekki sofa	
Conditional	Anterior	Positive	ég myndi hafa sofið	Past Conditional
Conditional	Anterior	Negative	ég myndi ekki hafa sofið	

Verb categories The resource grammar distinguishes between verbs based on their transitivity. Of different transitivity groups there are:

- Intransitive verbs or one-place verbs, V. These are verbs that relate no object to a subject, e.g., "deyja" ("die").
- Transitive verbs or two-place verbs, $V2$. These are verbs that relates one object to a subject, e.g., "taka" ("take").
- Ditransitive verbs or three-place verbs, $V3$. These are verbs that relate two objects to a subject, e.g., "gefa" ("give").

There is also a distinction made on what kind of complement a verb relates to a subject, e.g., verbs that take sentences and adjectival complements have the type VS and VA respectively. Information on verbs transitivity and the type of complement it can relate to a subject can be very important in functions that construct verb phrases. This information has to be defined when the verb it self is defined in the Lexicon. The Lexicon therefore plays a role of considerable importance within the resource grammar.
Auxiliary verbs, on the other hand, do not have a special group within the resource grammar. Similarly, Icelandic auxiliary verbs do not form a special group that is distinctive from other verbs[6]. Verbs that are most frequently listed and used as auxiliaries in Icelandic grammar, such as "hafa" ("have"), "vera" ("be"), and "munu" ("will"), have agreement like other verbs and inflect for tense. They are therefore not considered to be separate inflectional class of verbs.

Some of these auxiliary verbs have however a limited number of verb forms, e.g., "munu" ("will") and "vera" ("be") do not exist in the middle nor the passive voice and "munu" ("will") does exist in the past tense of the indicative mood.
Icelandic auxiliaries are thus only defined by their usage, i.e. a group of words that are used to systematically express grammatical categories. Examples of such categories are the passive and perfect, such as shown in examples 19 and 20 below.

(19) Hurðin **var opnuð**
The door **was opened**

(20) Strákurinn **hefur lesið** þessa bók
The boy **has read** this book

The auxiliary verbs are implemented as helper functions within the Icelandic resource grammar. They have have the same type and functionality as regular verbs, V. The auxiliary verbs that are implemented as functions in the Icelandic resource grammar are:

- "vera" ("be") as $verbBe$
- "verða" ("become") as $verbBecome$
- "mun" ("will") as $verbWill$
- "hafa" ("have") as $verbHave$

Middle voice

As stated in section 2.2.1 there is in addition to the active and the passive a middle voice. The middle voice is said to be in the middle between the active and passive voices because the subject can often be categorized as both agent and patient. Verbs in the middle voice are identifiable by the inflexional suffix -st.
Verbs in the middle voice are often used in the following situations :

(21) Reflexive
Bjarni klæðist
Bjarni gets dressed

(22) Reciprocal
Bjarni og Gunnar heilsast
Bjarni and Gunnar greet each other

(23) Passive
Fjallið sést ekki
The mountain cannot be seen

(24) Anticausative
Glugginn opnaðist af sjálfu sér
The window opened by itself

The middle voice can also be used to construct verbs from nouns, e.g., "djöflast" (to do some thing aggressively) from "djöfull" ("demon").
Now the middle voice is currently implemented only as a verb forms in the resource grammar. That is, it is not used anywhere outside of the inflection tables within the resource grammar. Since the abstract syntax does not include a middle voice, but only the active an passive voices, the implementation is not trivial and needs special care. A further discussion on what remains to be done regarding the middle voice is carried out in section 4.2.

2.4 Paradigms

In linguistics a morphological paradigm is the complete description of word forms associated with a word. Examples of paradigms are the declensions of nouns and adjectives. Traditionally the word forms of a word are arranged into a inflection table. Such tables are then classified by shared inflectional categories. Inflection tables of Nouns, for an example, would be categorized by number (singular and plural) and case (nominative, accusative, dative and genitive). Furthermore, a noun would be needing two such tables, with and without the suffixed definite article.

Table 2.6: Inflectional table for the masculine noun "armur" ("arm").

	Singular		Plural	
Case	Without article	With article	Without article	With article
Nominative	armur	armurinn	armar	armarnir
Accusative	arm	arminn	arma	armana
Dative	armi	arminum	örmum	örmunum
Genitive	arms	armsins	arma	armanna

In GF the paradigms are functions that produce inflection tables. Such a function has word strings as arguments, i.e. the word forms of a word, and outputs a n-tuple of word strings. This n-tuple then corresponds to the full inflection table of a word.

Listing 2.17: The inflectional output for "armur" with the masculine noun paradigm dArmur.

```
s Sg Free Nom : armur
s Sg Free Acc : arm
s Sg Free Dat : armi
s Sg Free Gen : arms
s Sg Suffix Nom : armurinn
s Sg Suffix Acc : arminn
s Sg Suffix Dat : arminum
s Sg Suffix Gen : armsins
s Pl Free Nom : armar
s Pl Free Acc : arma
s Pl Free Dat : örmum
s Pl Free Gen : arma
```

```
s Pl Suffix Nom : armarnir
s Pl Suffix Acc : armana
s Pl Suffix Dat : örmunum
s Pl Suffix Gen : armanna
```

Most natural languages have many paradigms. Pairing a word and a paradigm for every lexeme of a lexicon is extremely time consuming for large lexicons. Furthermore, this gives way for a lot of human error as the lexicographer has to choose manually among many paradigms for each word.
In GF this is solved by using a smart paradigm[4]. A smart paradigm is a meta-paradigm, which inspects a given base form and tries to infer which low-level paradigm applies. If the results are uncertain or the given form simply is indeterminable, more forms are given for discrimination. This reduces the number of paradigms to just one smart paradigm with a varying number of input variables. The average number of input variables needed is then used as a measurement of the predictability of the languages morphology.

2.5 Clauses and sentences

In the GF resource Grammar Library clauses, *Cl*, are a representation of sentences that do not yet have any tense, polarity or word order set. There is furthermore made distinction between three kinds of clauses. Namely declarative, interrogative, and relative, and they are represented within the GF Resource Grammar Library by the category names *Cl*, *QCl* and *RCl* respectively. Their definitions are very similar as shown below.

Listing 2.18: The definition of declerative

```
oper
    Cl : Type = {
        s : Tense => Anteriority => Polarity => Order => Str
    } ;

    QCl : Type = {
        s : Tense => Anteriority => Polarity => QForm => Str
    } ;

    RCl : Type = {
        s : Tense => Anteriority => Polarity => Agr => Str
    } ;
param
    Order = ODir | OQuestion ;
    QForm = QDir | QIndir ;
```

Clauses in the Icelandic resource grammar are generally made from a noun phrase (the subject) and a verb phrase (verb and object). The word order, of a declarative clause, in Icelandic is generally SVO [6], i.e., subject - verb - object. Other orders are possible such as OVS [6], i.e., object - verb - subject, as shown in examples

25 and 26. Nevertheless, the SVO is arguably the default word order of Icelandic and used by most modern speakers. The simpler approach of only implementing the SVO in the resource grammar is thus taken, other word orders are left to application grammars if needed.

(25) (OVS) Harald elskar María
(SVO) María elskar Harald
Mary loves Harold

(26) (OVS) Harald hefur María elskað
(SVO) María hefur elskað Harald
Mary has loved Harold

Since both interrogative and relative clauses can be formed from a declarative clause, it must contain the necessary word orders for such constructions. This is solved with the *Order* parameter that contains two different orders, *ODir* that represents a direct declarative order (SVO) and *OQuestion* that represents an interrogative order. In Icelandic this is done very much like in English, the subject is moved in front of the last finite verb form of the verb phrase. An overview of these different orders is given in the table 2.7 below where a clause is linearized into different sentences. Interrogative clauses can also be further linearized in different forms depending on whether they are direct or indirect questions. The parameter *QDir* then governs which form is used.

Table 2.7: Comparison of the clauses "ég lesa bókina" ("I read the book") and its possible sentence linearizations.

Tense	Anteriority	Polarity	Order	Sentence
Present	Simultaneous	Positive	ODir	ég les bókina
Present	Anterior	Positive	ODir	ég hef lesið bókina
Past	Simultaneous	Positive	ODir	ég las bókina
Past	Anteriour	Positive	ODir	ég hafði lesið bókina
Present	Simultaneous	Positive	OQuestion	les ég bókina (?)
Present	Anterior	Positive	OQuestion	hef ég lesið bókina (?)
Past	Simultaneous	Positive	OQuestion	las ég bókina (?)
Past	Anteriour	Positive	OQuestion	hafði ég lesið bókina (?)

To form a sentence a clause must then contain all combinations of tense, polarity and order needed to represent it. A sentence, S, in the resource grammar will then simply be a string - albeit with a complicated history.

3

Evaluation

3.1 Testing

To evaluate the correctness of the Icelandic resource grammar a modest sized test set was used. The test set [1] consisted of 172 abstract syntax trees that where used to evaluate the Icelandic resource grammar. The test set modified so that the trees where of top category, i.e. *Utt*, *Phr*, or *Text*. This was done to prevent discontinuities in linearization of the trees which otherwise would happen in many cases, e.g., some adjective phrases who would otherwise not be given any gender.

When evaluating linearizations of abstract syntax trees it can be of great benefit to have more languages linearized than just the one that is under evaluation. That is, a language that is already existing in the Resource Grammar Library and been thoroughly tested itself is used for comparision. Naturally it is of importance that the language chosen for comparision is familiar, thus English was the most natural choice. The linearizations where then automaticllay lineariezed into Icelandic and English for evaluation. The results from these evaluations are presented in section 3.2.

3.2 Results

An overview of the tests components along with total number of trees and correct linearizations of those trees is given in table 3.1 below.

[1] https://github.com/GrammaticalFramework/gf-contrib/blob/master/testsuite/resource.gfs

Table 3.1: Overview of the test set components and results.

Component	Number of trees	Number of correct trees
Adjective Phrase	9	9
Adverbs	6	6
Conjunctions	8	7
Idiom	8	8
Noun Phrases	40	37
Numerals	14	14
Phrase	13	10
Question	12	12
Relative	4	3
Sentence	15	13
Text	4	4
Verb Phrases	20	18
Other long examples	19	11
Total	172	152

As we can see from the table above the total number of correctly linearized trees are 152 out of 172, which calculates to a correctness of just about 88 %.
Of the incorrectly linearized trees many where because of exceptions from general rules which are hard to catch. An example of this is a possessive construction where a pronoun is is the possessor and possession is a noun depicting a kinship. In such cases the possession does not take the suffixed article[6]. This applies also to a few other relational nouns such as "vinur" ("friend")[**?**].

(27) *Móðirin/faðirinn mín/minn
Móðir/faðir mín/minn
My mother/father

(28) *Vinurinn/vinkonan minn/mín
Vinur/vinkona minn/mín
My friend

Other examples incorrect linearizations are because of limited word forms of some words in the lexicon, i.e., a word not containing some word forms that other words of the same category generally have.
It should be noted that the total percentage of correctly linearized trees is a measurement on how well the resource grammar covers this particular set of trees. No measurements have been made on how well the resource grammar covers the Icelandic Language in general. Furthermore, no measurement or test result currently exist on the parsing ability of the Icelandic resource grammar, which might be of considerable interest in some language processing tasks.

4
Discussion

4.1 Conclusion

The first part of the main goal was and is to implement the Icelandic resource grammar in GF, as is stated before in this project. This goal has been, on the whole, achieved in this project.
Evaluation showed good results on a modest sized test set. There are, however, some limitations known in the resource grammar that effect its coverage of the Icelandic language, of which most notably are lexical resources. Because of these limitation, and the size of this project, further evaluation on linearization of larger tree sets and on parsing text have not been made.
The grammar covers all of the constructs provided by the abstract syntax of the GF Resource Grammar Library. The Icelandic resource grammar stands therefore fully parallel to other languages implemented in the GF Resource Grammar Library, e.g., English and Swedish.

4.2 Future Work

Large scale Lexicon The resource grammar includes a small lexicon of common words, around 300 words, which is common to all the languages implemented in the Resource Grammar Library. For better usage of the Icelandic resource grammar, a bigger lexicon is needed. More serious machine translation work a lexicon should have a coverage of a 100 times larger order of magnitute, c.a. 30.000 words.
Such an extension would not only strengthen the usability of the Icelandic resource grammar within machine translations and other language processing tasks, it would be the optimal task to test thoroughly test the smart paradigms that have been implemented. Furthermore a measurement of the predictability of the languages morphology, as described in section 2.4, would be obtained.
There are already available and free of use sources online. Most notably is the Apertium dictionary, existing for both pairs of Icelandic and English, and Icelandic and Faroese [1].
Other sources do also exist online, such as the Database of Modern Icelandic Inflection that is a collection of Icelandic paradigms[2]. Such a collection could be used for more testing and comparison of this projects smart paradigms. It must be noted,

[1] http://wiki.apertium.org/wiki/Icelandic
[2] http://bin.arnastofnun.is/DMII/

however, that the Database of Modern Icelandic Inflection is copyrighted (when this project was done).
Another source of interest is the ISLEX project, a multilingual translation project between the Nordic languages[3].

MiddleVoice The middle voice, or voices in general, is not a construct of the Resource Grammar Library. But it is a quite frequently used functionality of the language, and thus it could be of value to implement it, at least within the Icelandic resource grammar.
Currently, in the Icelandic resource grammar, the middle voice exists only as word forms in the inflection table of Verbs. The functionality of the middle voice, as discussed in section 20, is not implemented. Since there is no standard equivalence of the middle voice within the Resource Grammar Library it should, like all extra features of a language, be implemented in the *Extra* module. Implementing the common functionalities of the middle voice, as listed in section 20, could be an interesting task.

4.3 Ethics

The project itself does not immediately raise ethical questions, but its implementation opens many opportunities in language processing tasks and other implementations that do raise ethical questions.
The implementation of the Icelandic Language as a GF grammar and its addition to the GF Resource Library would undoubtedly strengthen linguistic research, but what about elementary teaching of the language? This project can give way to a grammar checking programs that could potentially ensure the user always uses the grammar when constructing sentences. Would such an implementation be the beginning of the end of human grammar knowledge? We humans are in nature very lazy, i.e. when retrieval of information is much easier and quicker than learning it we tend to exploit such "short-cuts". In addition with automatic spell checking in various programs being as good as it already is today, combining such powerful language tools might really weaken the general need for humans to learn correct text writing. One might fear that it would lead to a scenario where the native Icelandic speaker doesn't bother learning the grammar anymore. Such considerations are not defined to the Icelandic language of course.
I personally do not agree, and on the contrary think it might even strengthen the language skill of its users. We humans after all our laziness tend to also learn from repetition. Having such tools would, in my opinion, give rapid feedback on errors people tend to make everyday regardless of having had considerable educational background in the language. Having a firmly defined grammar implementation for such tasks could thus increase consistency in written text; such as reducing jumping between tenses and wrong declensions of nouns, pronouns, and adjectives. So I think it would generally strengthen the language skill of its speakers with the language

[3]http://www.islex.is/islex?um=1

increased strength in digital applications and not make grammatical knowledge a relic of the past.
Languages are always evolving, some are rapidly changing while others seem to remain the same (in written form at least) over many centuries. The Icelandic languages falls historically under the latter, having little changed in written form over the last 1000 years or so. But changing nonetheless, and with its change the grammar changes as well. Will the implementation of such powerful text tools as described above lead to the current grammar being carved in stone, i.e. will it delay or stop all together the language's evolution? Such a scenario would undoubtedly please a number of speakers, but would that justify it? Again such considerations are not necessarily defined to the Icelandic language. I think, considering the above, it would in a sense delay the evolution of the language, but not to a great extent. Language are generally tools of speech before they are used for writing. Language evolve subtly anyways, mostly with added vocabulary or by semantic change. Also, the GF grammar implementation can be changed and improved later on if needed.
But this raises further questions regarding the chose of grammar definition and its implementation as a GF grammar: what is considered the correct grammar of the Icelandic language? The Icelandic Ministry of Education has a policy regarding the teaching of Icelandic language for both elementary schools and high schools (ages 6 - 16 and 16 - 20 respectively) that is highly or almost exclusively formed by the Árni Magnússon Institute for Icelandic studies and other linguistics related to the institute.

Bibliography

[1] Aarne Ranta
Grammatical Framework: programming with multilingual grammars. CSLI Publications, Stanford. (2011)

[2] Aarne Ranta
The GF Resource Grammar Library. Linguistic Issues in Language Technology volume 2(2). (2009)

[3] Eiríkur Rögnvaldsson
Hljóðkerfi og orðhlutafræði íslensku. Reykjavík. (2013)

[4] Grégorie Détrez and Arne Ranta
Smart paradigms and the predictability and complexity of inflectional morphology. In Proceedings of the 13th Conference of the EACL. Avignon, France: Association for Computational Linguistics. (2012)

[5] Halldór Ármann Sigurðsson
The Icelandic Noun Phrase: Central Traits. Arkiv för nordisk filologi 121. (2006)

[6] Höskuldur Þráinsson
The Syntax of Icelandic. Cambridge University Press. (2007)

[7] Höskuldur Þráinsson
Hvað eru margar tíðir í íslensku og hvernig vitum við það? Íslenskt mál 21:181-224. (1999)

[8] Per Martin-Löf
Intuitionistic Type Theory. Bibliopolis. (1984)

[9] Stefán Einarsson
Icelandic: Grammar, Texts, Glossary. Baltimore: Johns Hopkins. (1945)

A

Appendix 1

The implementation of the Icelandic Resource Grammar is, currently, available at https://github.com/bjarkit/GF-Icelandic . The code is licenced under GNU LESSER GENERAL PUBLIC LICENSE as is the Resource Grammar Library[1] which is available at http://www.grammaticalframework.org/lib/src/ .

[1] http://www.grammaticalframework.org/LICENSE