Machine Translation: Green, Yellow, and Red

Aarne Ranta

University of Gothenburg and Digital Grammars AB

BAULT Seminar, University of Helsinki
3 June 2015

CLT REMU digital Grammars
Language technology to rely on.

Versions also given at

University of Gothenburg, April 2014

NLCS/NLSR, Vienna Summer of Logic, July 2014

CNL, Galway, August 2014

WoLLIC, Valparaiso, September 2014

University of Stockholm, September 2014

Shanghai University of Finance and Economics, Nov 2014

Hong Kong Polytechnic University, Nov 2014

University of Malta, Mar 2015

Contributors

Krasimir Angelov, Björn Bringert, Grégoire Détrez, Ramona Enache, Erik de Graaf, Thomas Hallgren, Qiao Haiyan, Chotiros Kairoje, Prasanth Kolachina, Inari Listenmaa, Peter Ljunnglöf, K.V.S. Prasad, Scharolta Siencnik, Daniel Vidal, Shafqat Virk, Liza Zimina

50+ GF Resource Grammar Library contributors

5 March 2014 -

REMU

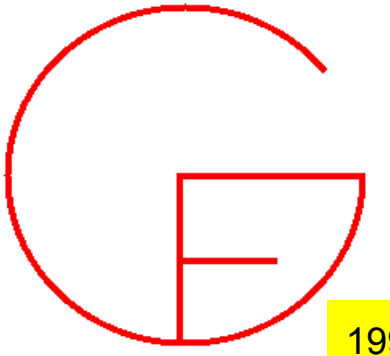
VR 2013 - 2017

MOLTO

EU 2010 - 2013

CLT

2009 -



1998 -

Translation: producer vs. consumer

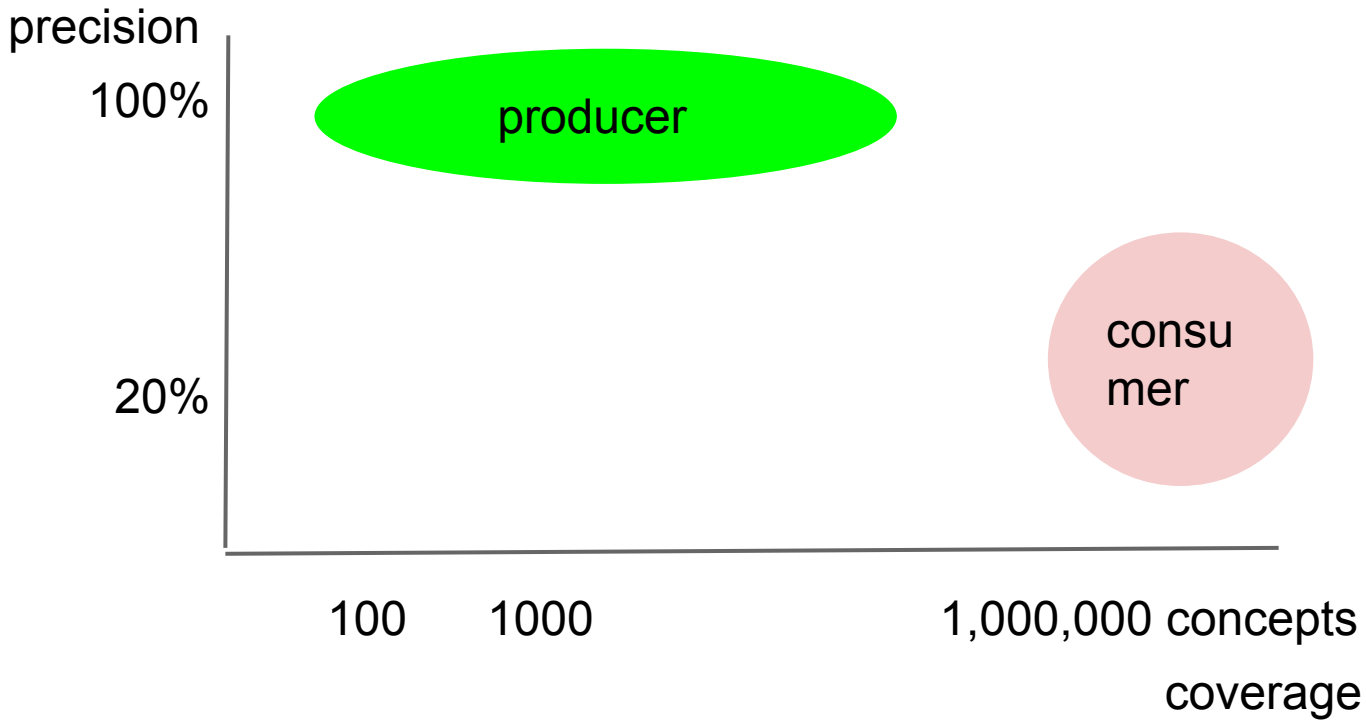
Consumer (MT mainstream):

- must translate anything
- browsing quality enough

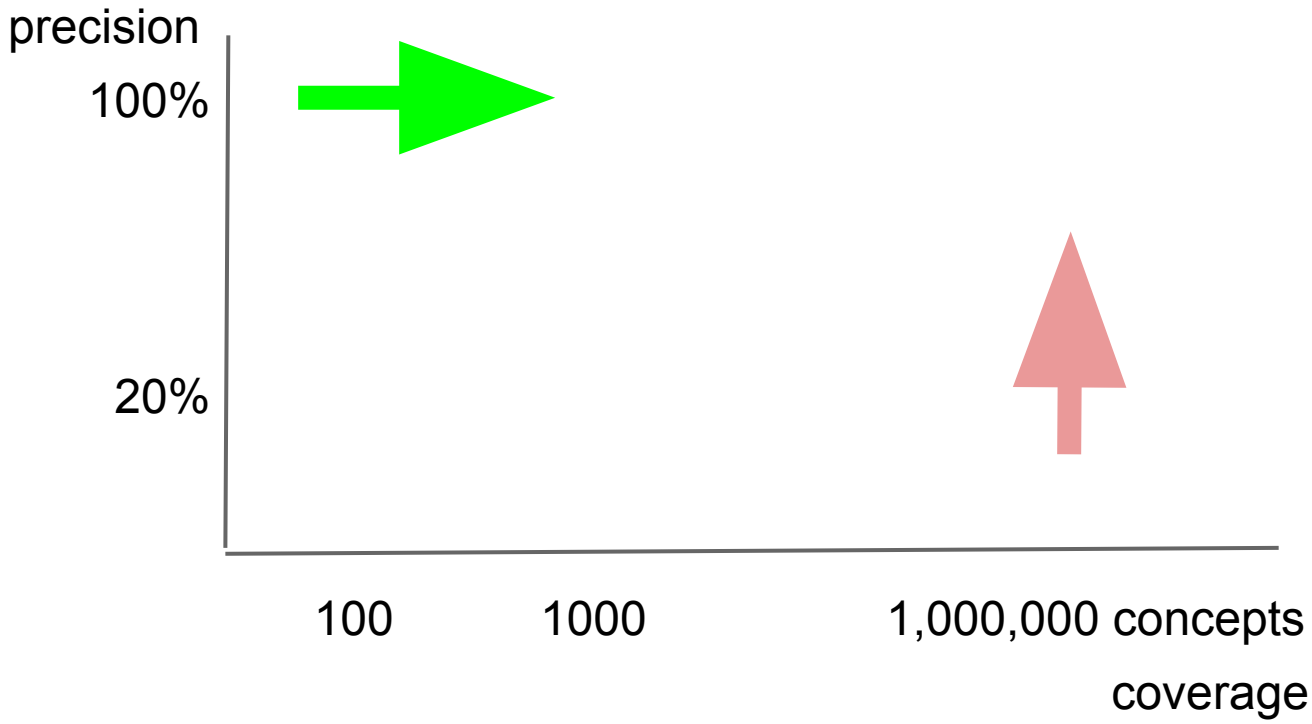
Producer:

- must translate my content
- publication quality required

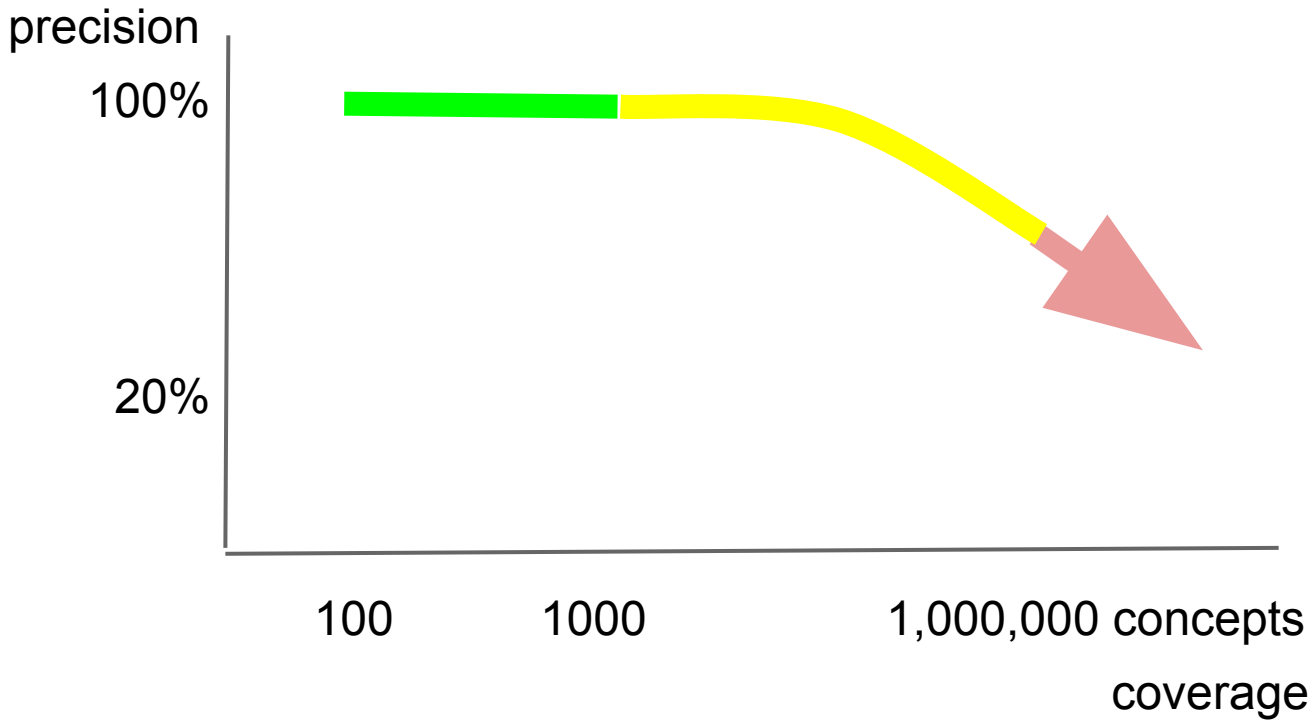
Orthogonal concepts



Two ways of developing a system



The best scenario?



An example

My son is hungry.

Pojallani on nälkä.

The vice dean kicked the bucket.

Pahedekaani potkaisi sankoa.

Little boy eat big snake.

Pieni poika syökää suuri käärme.

An example

My son is hungry.

Pojallani on nälkä. **meaning**

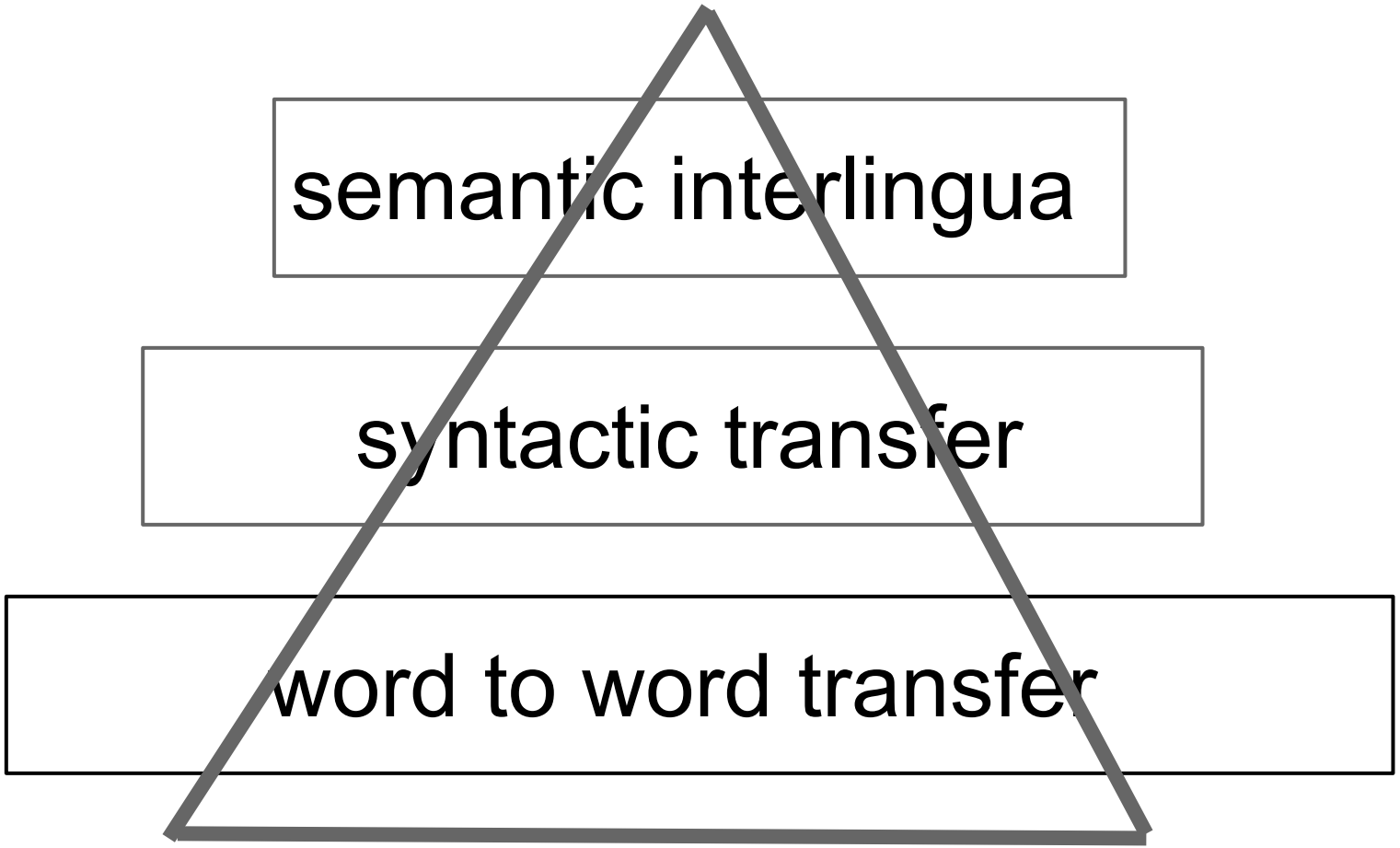
The vice dean kicked the bucket.

Pahedekaani potkaisi sankoa. **syntax**

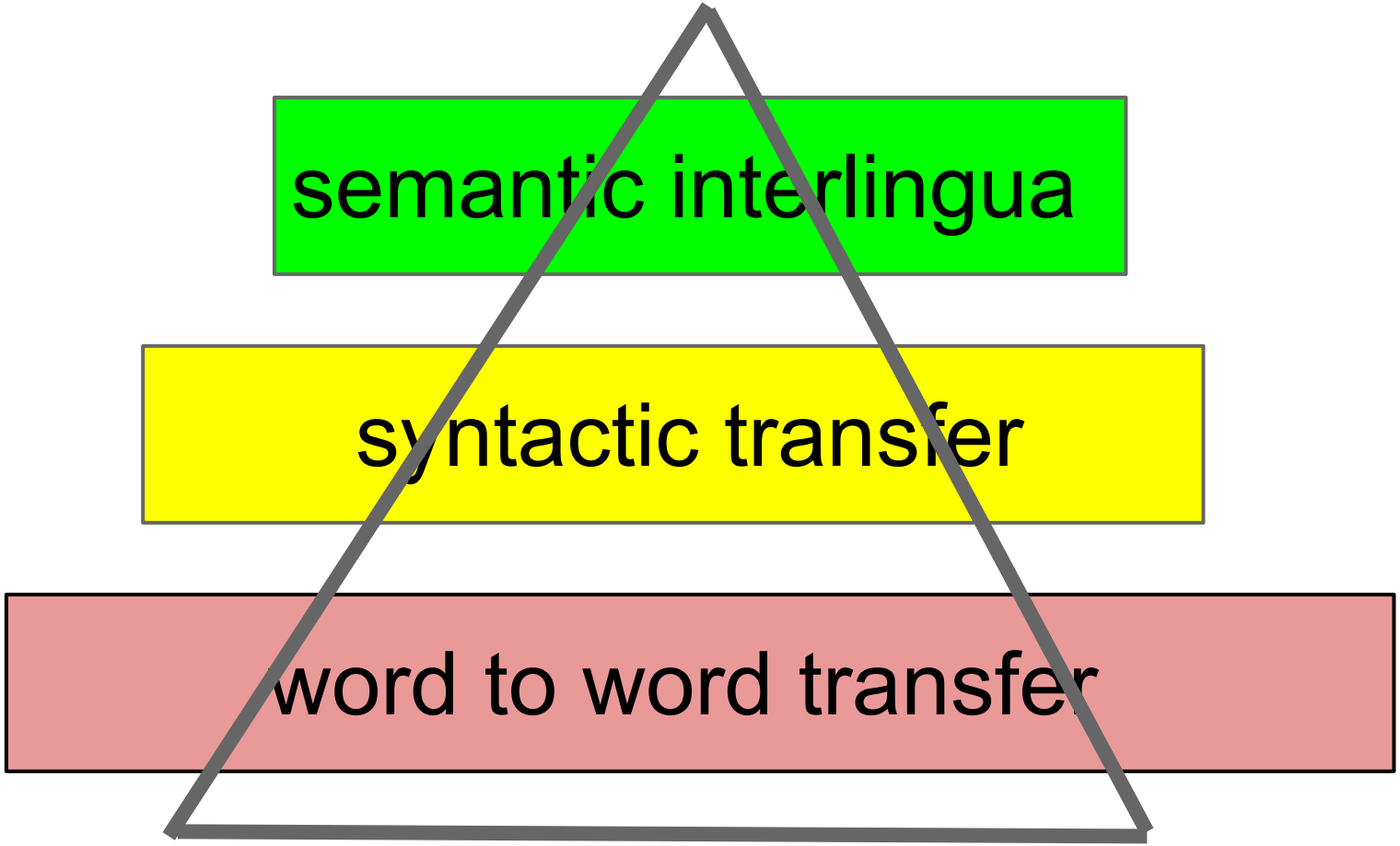
Little boy eat big snake.

Pieni poika syökää suuri käärme. **chunks**

The Vauquois triangle



The Vauquois triangle



What is it good for?

publish the content

get the grammar right

get an idea

Who is doing it?

GF in MOLTO

GF the last 24 months

Google, Bing, Apertium

GF = Grammatical Framework

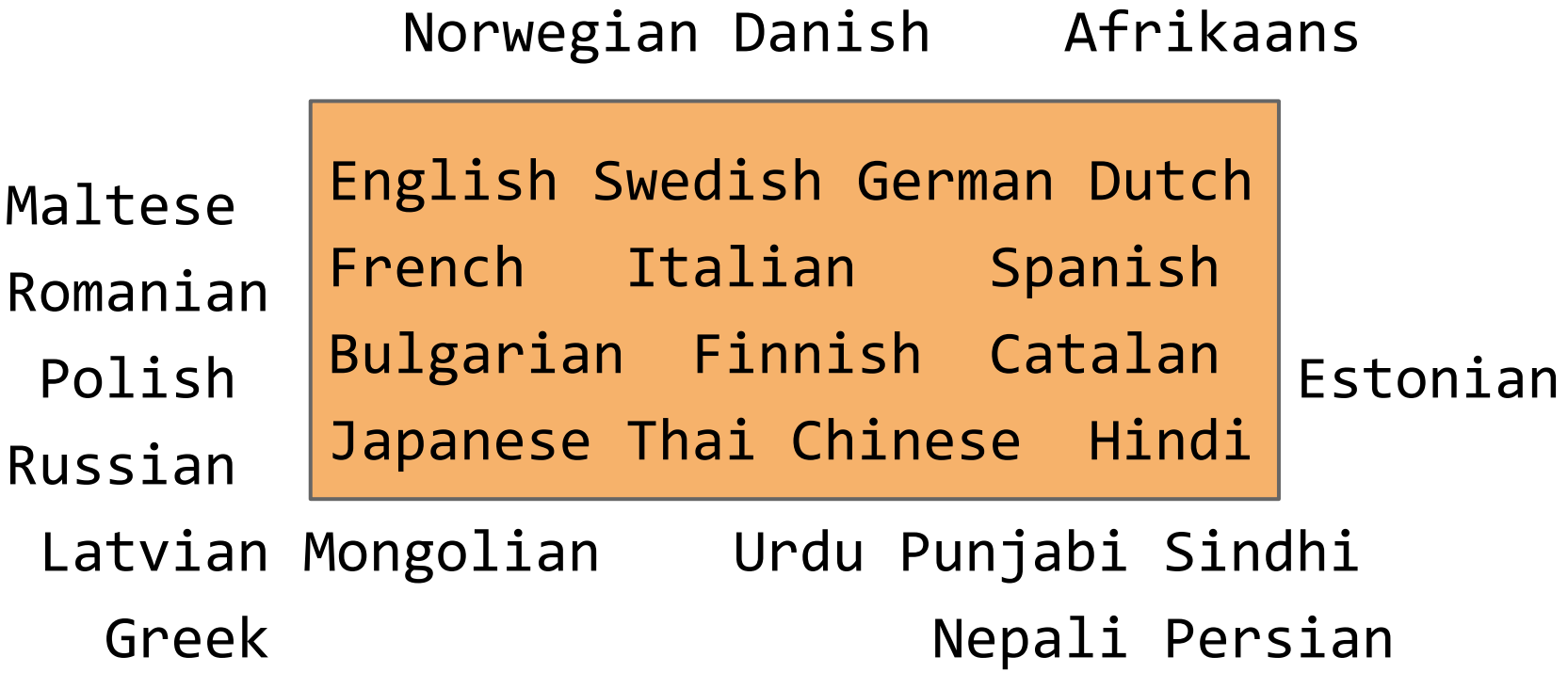
Based on **type theory** and **functional programming.**

Xerox 1998: **multilingual controlled language** translation.

Closest prior work: Montague grammar, Rosetta (Philips).

Today: open source, 150+ people, 30+ languages

Resources: basic and bigger



my new house is very big

मेरा अजनबी शाला बहुत महत्वपूर्ण है

你爱我吗

est-ce que tu m'aimes

ich wohne in einem gelben Haus

io risiedo in una casa gialla

jag är inte en älg

minä en ole hirvi

What should we work on?

All!

semantics for full quality

syntax for grammaticality

chunks for robustness

We want a system that

- can reach perfect quality
- has robustness as back-up
- tells the user which is which

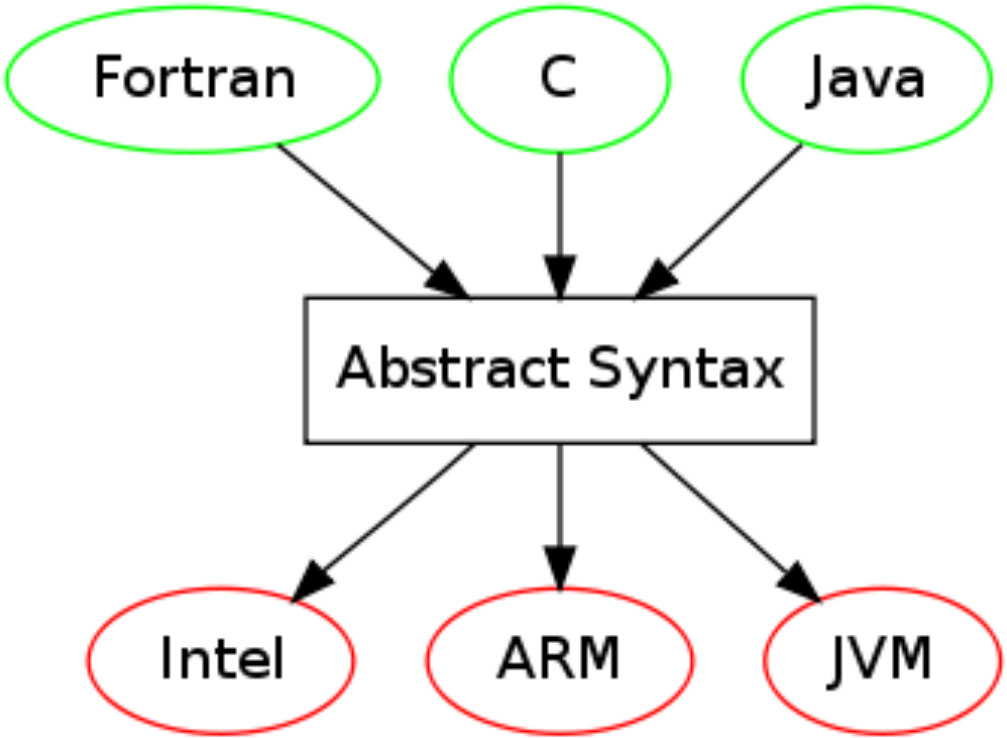
We “combine GF, Apertium, and Google”

But we do it all in GF!

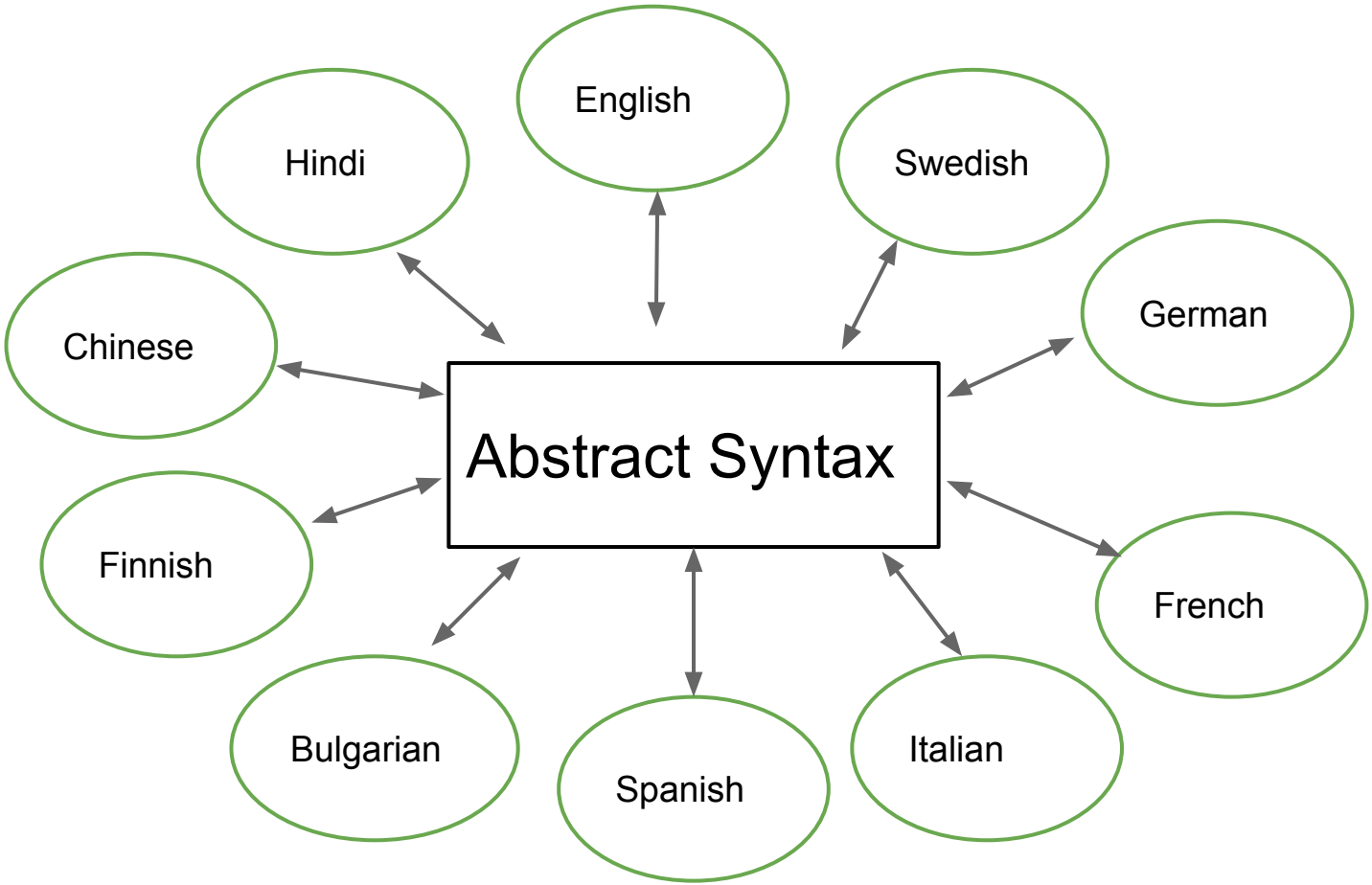
How to do it in GF?

a brief summary

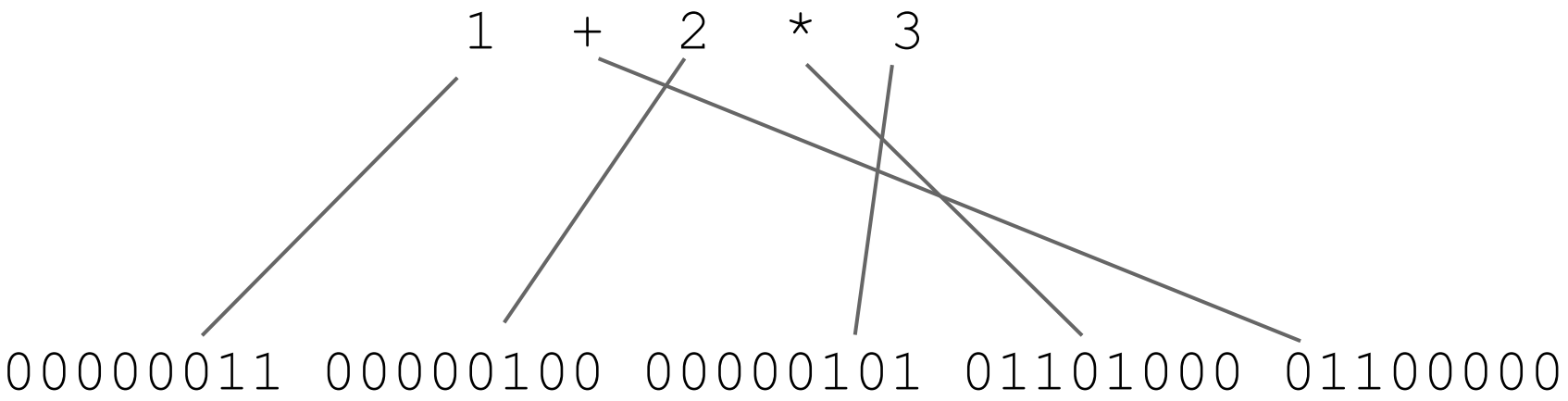
Translation model: multi-source multi-target compiler



Translation model: multi-source multi-target compiler-**decompiler**



Word alignment: compiler



Abstract syntax

Add : Exp -> Exp -> Exp

Mul : Exp -> Exp -> Exp

E1, E2, E3 : Exp

Add E1 (Mul E2 E3)

Concrete syntax

abstrakt	**Java**	**JVM**
Add x y	*x* “+” *y*	*x y* “01100000”
Mul x y	*x* “*” *y*	*x y* “01101000”
E1	“1”	“00000011”
E2	“2”	“00000100”
E3	“3”	“00000101”

Compiling natural language

Abstract syntax

Pred : NP -> V2 -> NP -> S

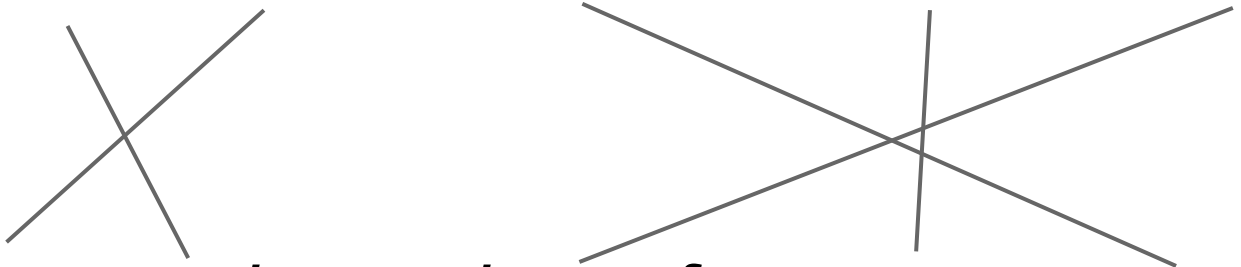
Mod : AP -> CN -> CN

Love : V2

Concrete syntax:	**English**	**Latin**
Pred s v o	*s v o*	*s o v*
Mod a n	*a n*	*n a*
Love	*“love”*	*“amare”*

Word alignment

the clever woman loves the handsome man



femina sapiens virum formosum amat

Pred (Def (Mod Clever Woman)) Love
(Def (Mod Handsome Man))

Linearization types (PMCFG)

	English	Latin
CN	*{s : Number => Str}*	*{s : Number => Case => Str ; g : Gender}*
AP	*{s : Str}*	*{s : Gender => Number => Case => Str}*

Mod ap cn

{s = \\n => ap.s ++ cn.s ! n}

{s = \\n,c => cn.s ! n ! c ++ ap.s ! cn.g ! n ! c ;
g = cn.g
}

Abstract syntax trees

my son is hungry

Hungry (Poss I Son)

Abstract syntax trees

my son is hungry

Hungry (Poss I Son)

Pred (Det (Poss i_NP) son_N)) (CompAP hungry_A)

Abstract syntax trees

my son is hungry

Hungry (Poss I Son)

Pred (Det (Poss i_NP) son_N)) (CompAP hungry_A)

[DetChunk (Poss i_NP), NChunk son_N, copulaChunk, AChunk hungry_A]

Compositional translation

my son is hungry

Hungry (Poss I Son)

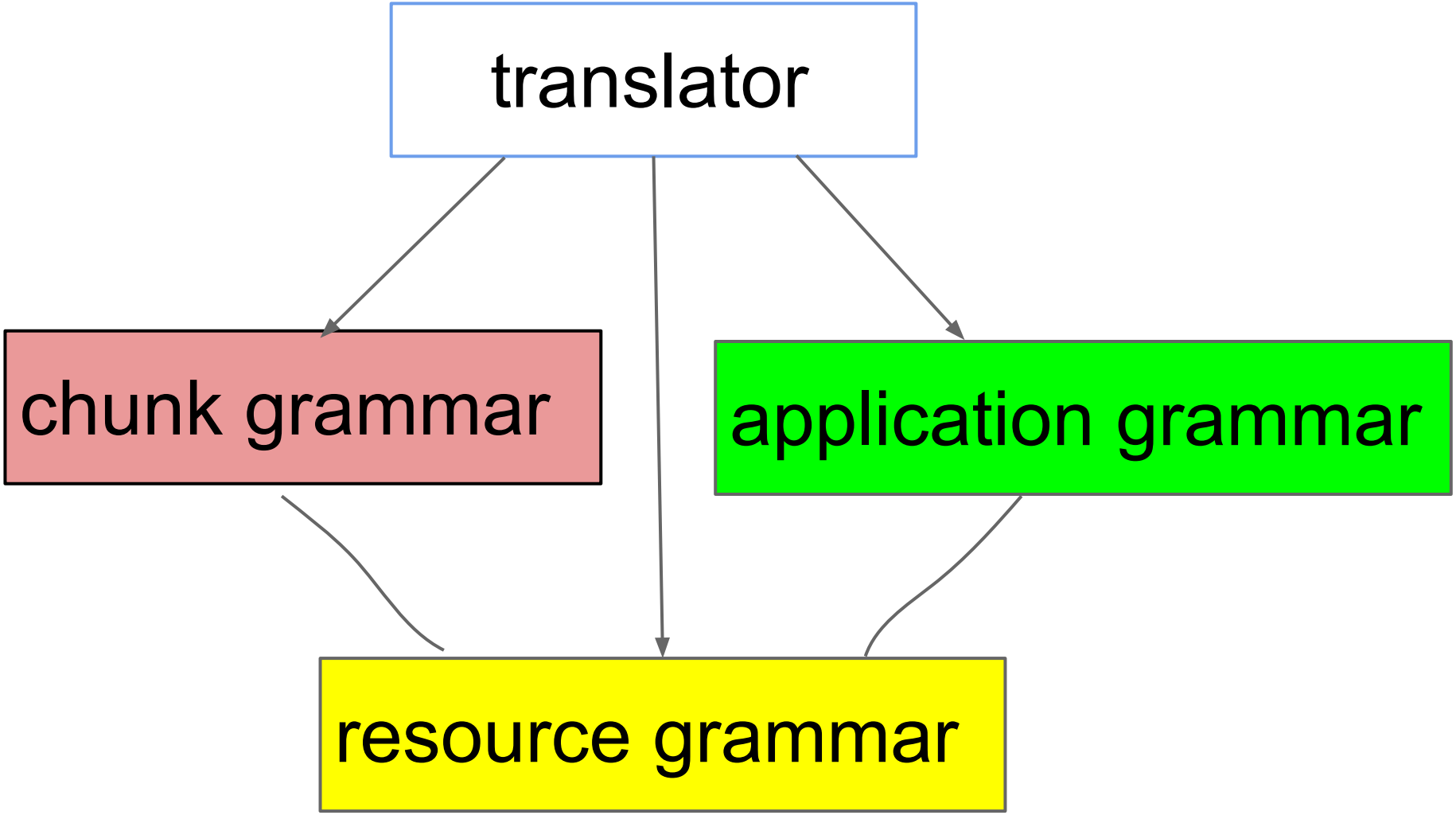
pojallani on nälkä

Pred (Det (Poss i_NP) son_N)) (CompAP hungry_A)

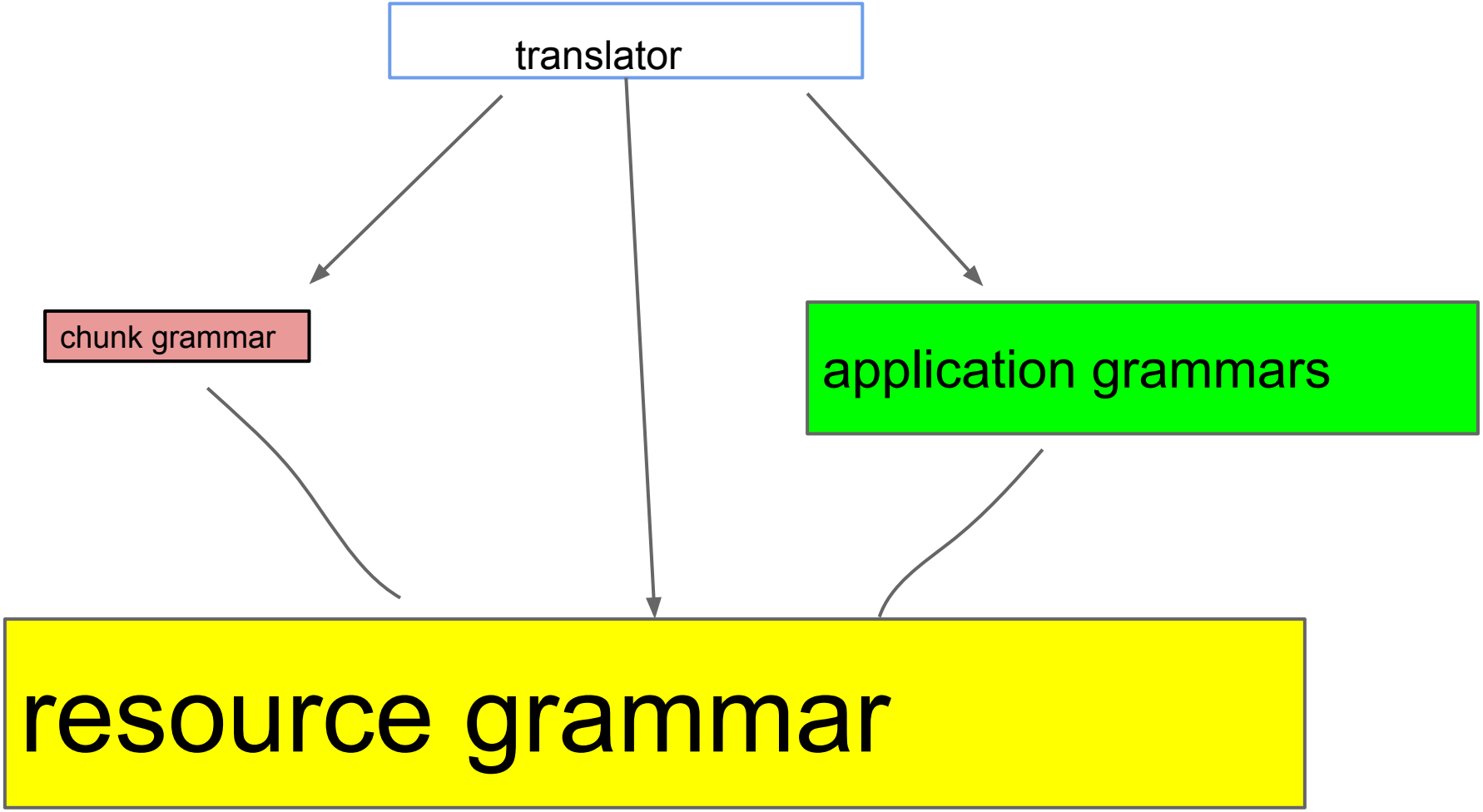
minun poikani on nälkäinen

[DetChunk (Poss i_NP), NChunk son_N, copulaChunk, AChunk hungry_A]

minun poika olla nälkäinen


translator
chunk grammar
application grammar
resource grammar

How much work is needed?


translator
chunk grammar
application grammars
resource grammar

resource grammar

- morphology
- syntax
- generic lexicon

precise linguistic knowledge

- 2-6 months manual work

application grammars

domain semantics, domain idioms

- need domain expertise

use resource grammar as library

- minimize hand-hacking

the work never ends

- we can only cover some domains

chunk grammar

words
suitable word sequences

- local agreement
- local reordering

derived from resource grammar
minimize hand-hacking

translator

PGF run-time system

- parsing
- linearization
- disambiguation

generic for all grammars

portable to different user interfaces

- web
- mobile

Disambiguation?

Grammatical: give priority to green over yellow, yellow over red

Statistical: use a distribution model for grammatical constructs (incl. word senses)

Interactive: for the last mile in the green zone

Demos

Demo 1: MOLTO Phrasebook

Source: **controlled language input**

Always green

Based on **domain semantics**

http://www.grammaticalframework.org/demos/phrasebook/

Demo 2: resource grammar

Source: **predictive input**

Always yellow

Based on **syntactic structure**

http://cloud.grammaticalframework.org/minibar

Demo 3: wide-coverage translation

Source: any text

Can be green, yellow, or red.

Based on **semantics, grammar, or chunks.**

http://cloud.grammaticalframework.org/wc.html

Demo 4: mobile translation app

Source: **text or speech** in any language

Can be green, yellow, or red.

Based on **semantics, grammar,** or **chunks.**

https://play.google.com/store/apps/details?id=org.grammaticalframework.ui.android

http://www.grammaticalframework.org/~aarne/App14.apk

How to do it?

some more details

Building the yellow part

Building a basic resource grammar

Programming skills

Theoretical knowledge of language

3-6 months work

3000-5000 lines of GF code

- no full automation

+ only done once per language

Building a large lexicon

Monolingual (morphology + valencies)

- extraction from open sources (SALDO, KOTUS)
- extraction from text (*extract*)
- **smart paradigms**

Multilingual (mapping from abstract syntax)

- extraction from open sources (Wordnet, Wiktionary)
- extraction from parallel corpora (Giza++)

Manual quality control at some point needed

Improving the resources

Multiwords: non-compositional translation

- *secretary of state - ulkoministeri*

Constructions: multiwords with arguments

- *x be(agr(x)) hungry - adessive(x) on nälkä*

Extraction from free resources (Konstruktikon)

Extraction from SMT phrase tables

- **example-based grammar writing**

Building the green part

Define **semantically based abstract syntax**

```
fun Hungry : Person -> Fact
```

Define concrete syntax by mapping to resource grammar structures

```
lin HasName p n = mkCl p hungry_A
    I am hungry

lin HasName p n = mkCl p have_V2 nälkä_N
    minulla on nälkä
```

Resource grammars give crucial help

- application grammarians need not know linguistics
- a substantial grammar can be built in a few days
- adding a new language is a matter of a few hours

MOLTO’s goal was to make this possible.

- EU project 2010-2013: Multilingual Online Translation

These grammars are a source of

- “non-compositional” translations
- idiomatic language
- translating meaning, not syntax

Constructions are the generalized form of this idea, originally domain-specific.

Building the red part

1. Write a grammar that builds sentences from sequences of chunks

```
cat Chunk
fun SChunks : [Chunk] -> S
```

2. Introduce chunks to cover phrases

```
fun NP_nom_Chunk : NP -> Chunk
fun NP_acc_Chunk : NP -> Chunk
fun AP_sg_masc_Chunk : AP -> Chunk
fun AP_pl_fem_Chunk : AP -> Chunk
```

Do this for all categories and feature combinations you want to cover.

Include both long and short phrases

- long phrases have better quality
- short phrases add to robustness

Give long phrases priority by probability settings.

Long chunks are better:

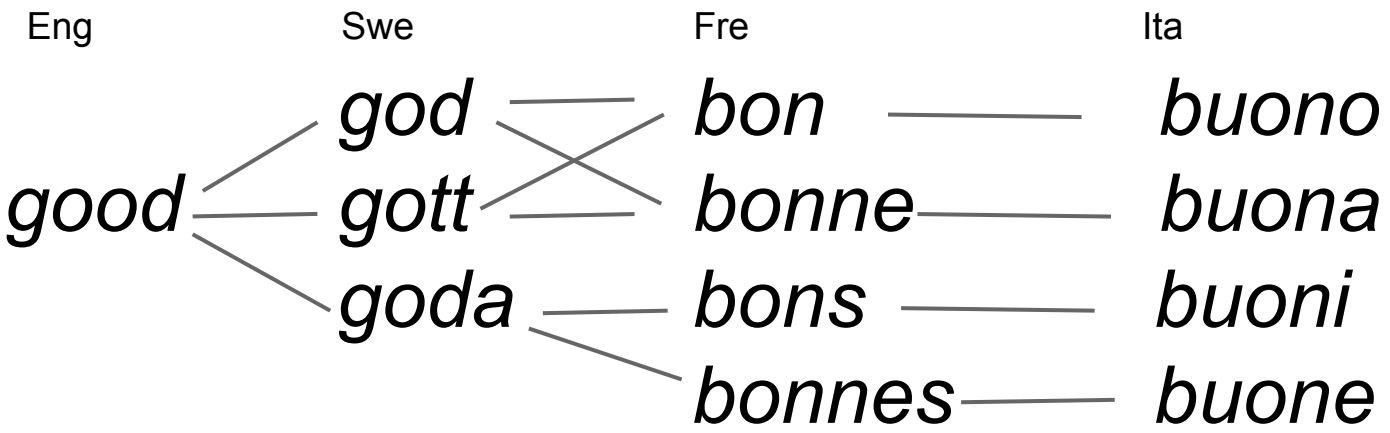
[this yellow house] - *[det här gula huset]*

[this] [yellow house] - *[den här] [gult hus]*

[this] [yellow] [house] - *[den här] [gul] [hus]*

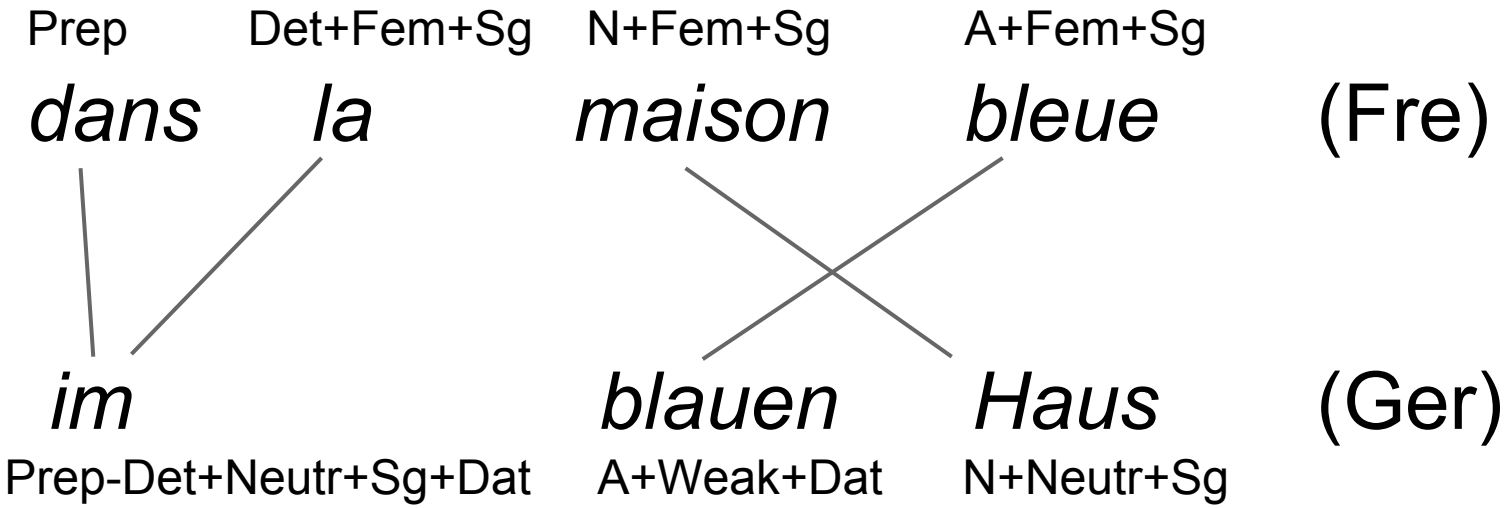
Limiting case: whole sentences as chunks.

Accurate feature distinctions can be good for closely related language pairs.



Apertium does this for every language pair.

Resource grammar chunks of course come with reordering and internal agreement



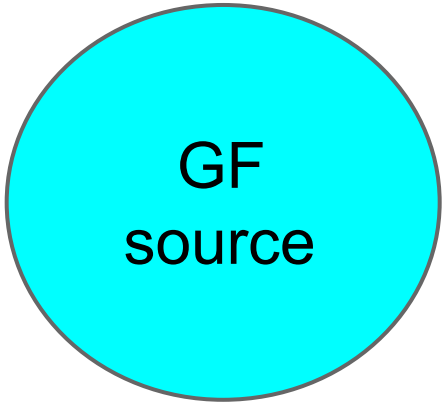
Recall: chunks are just a by-product of the real grammar.

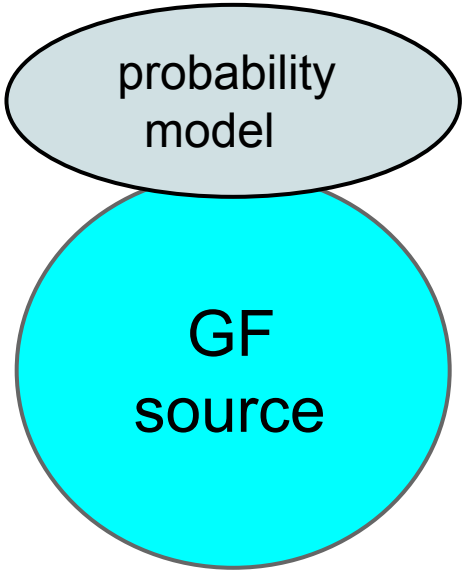
Their size span is

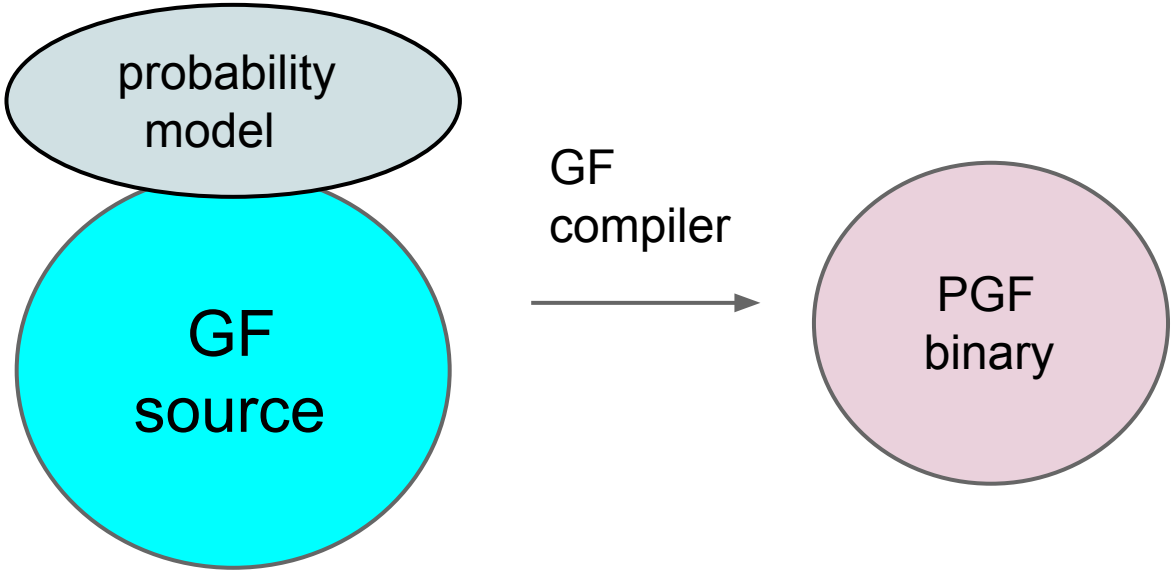
single words <-------> entire sentences

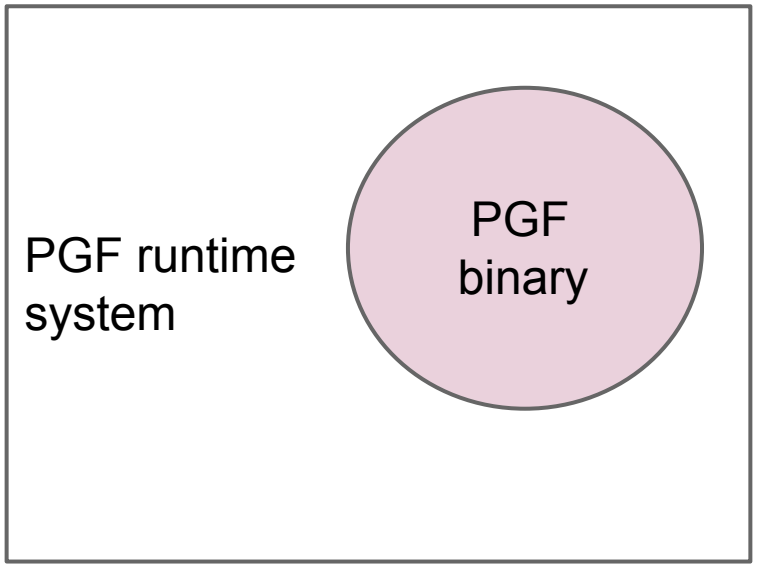
A wide-coverage chunking grammar can be built in a couple of hours **by using the RGL**.

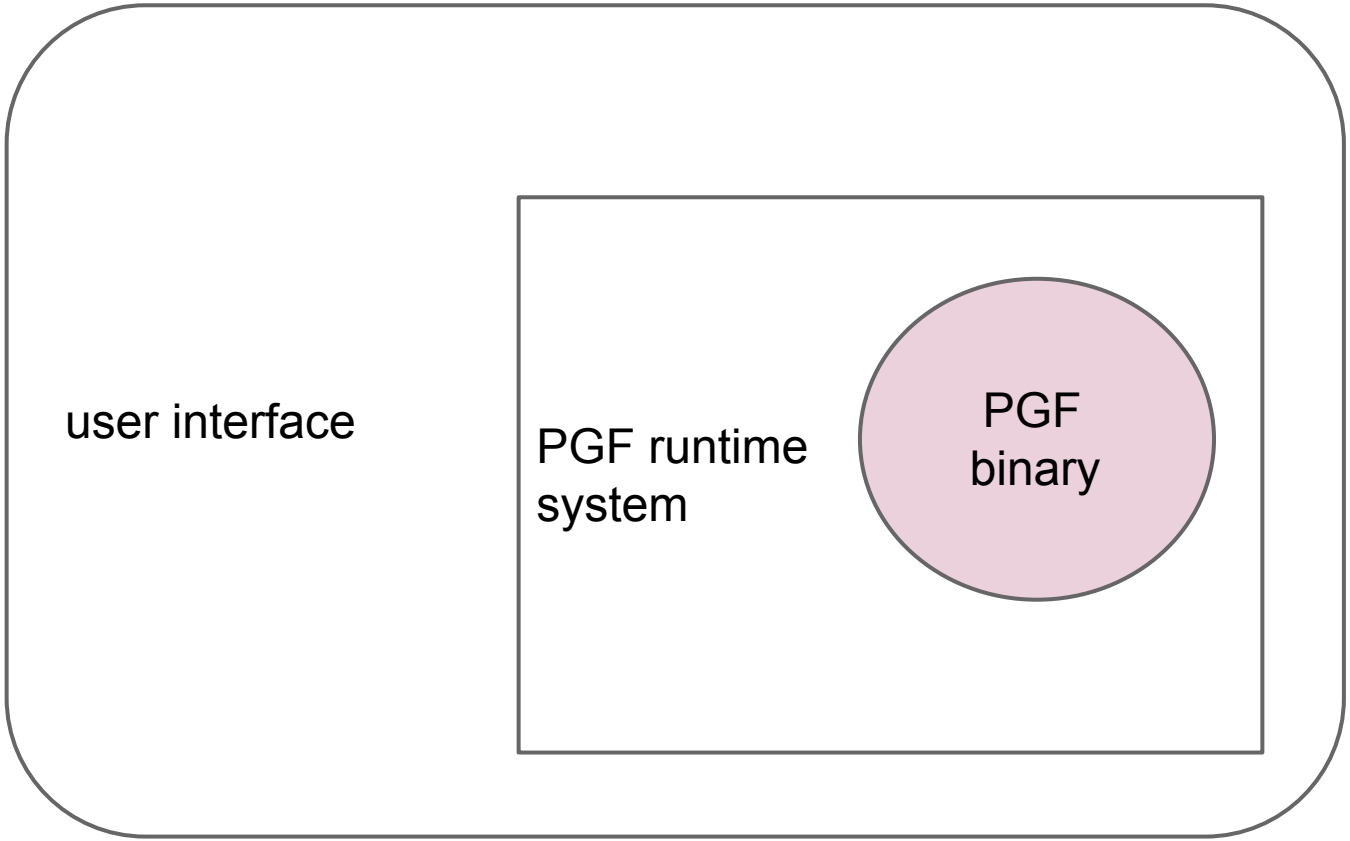
Building the translation system

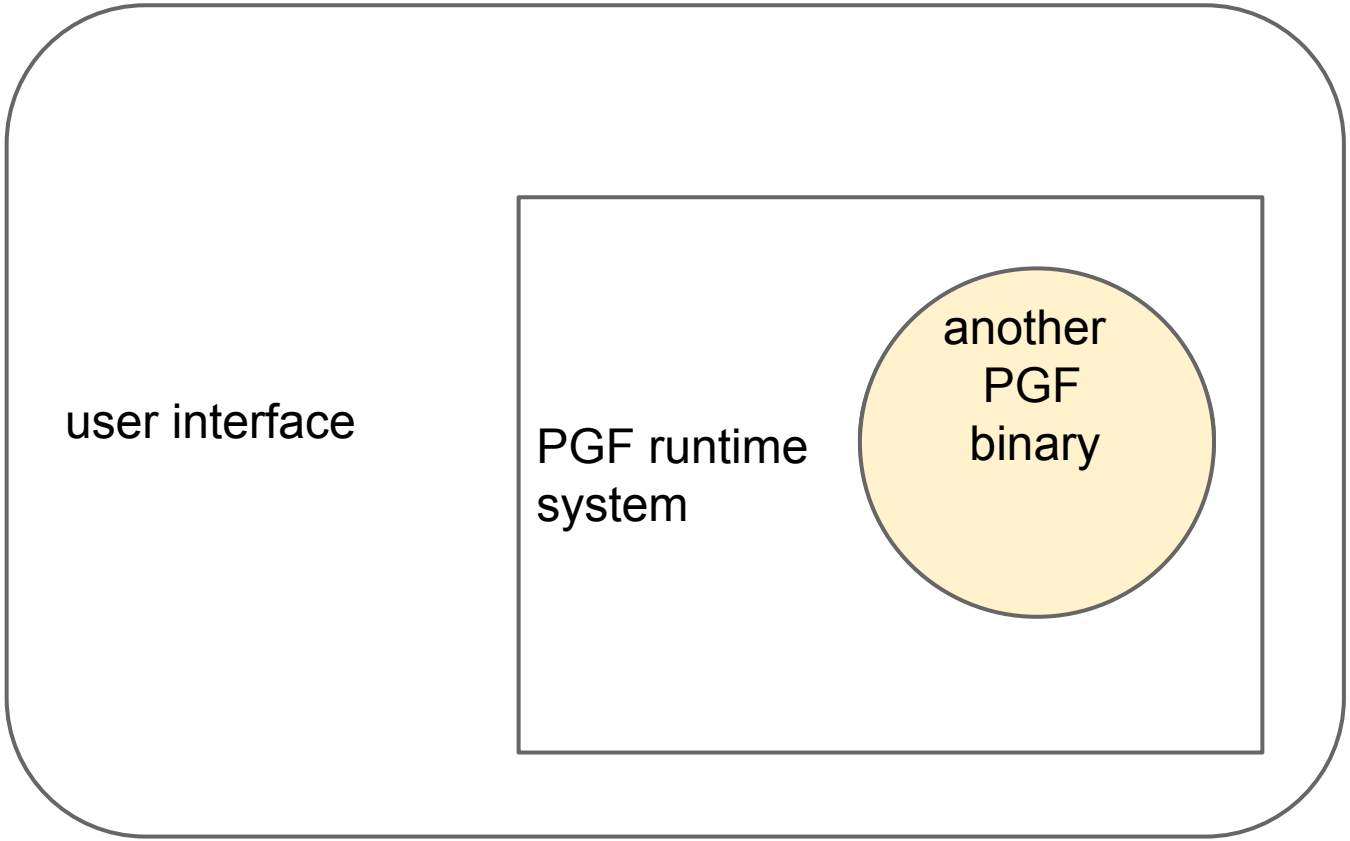

GF
source

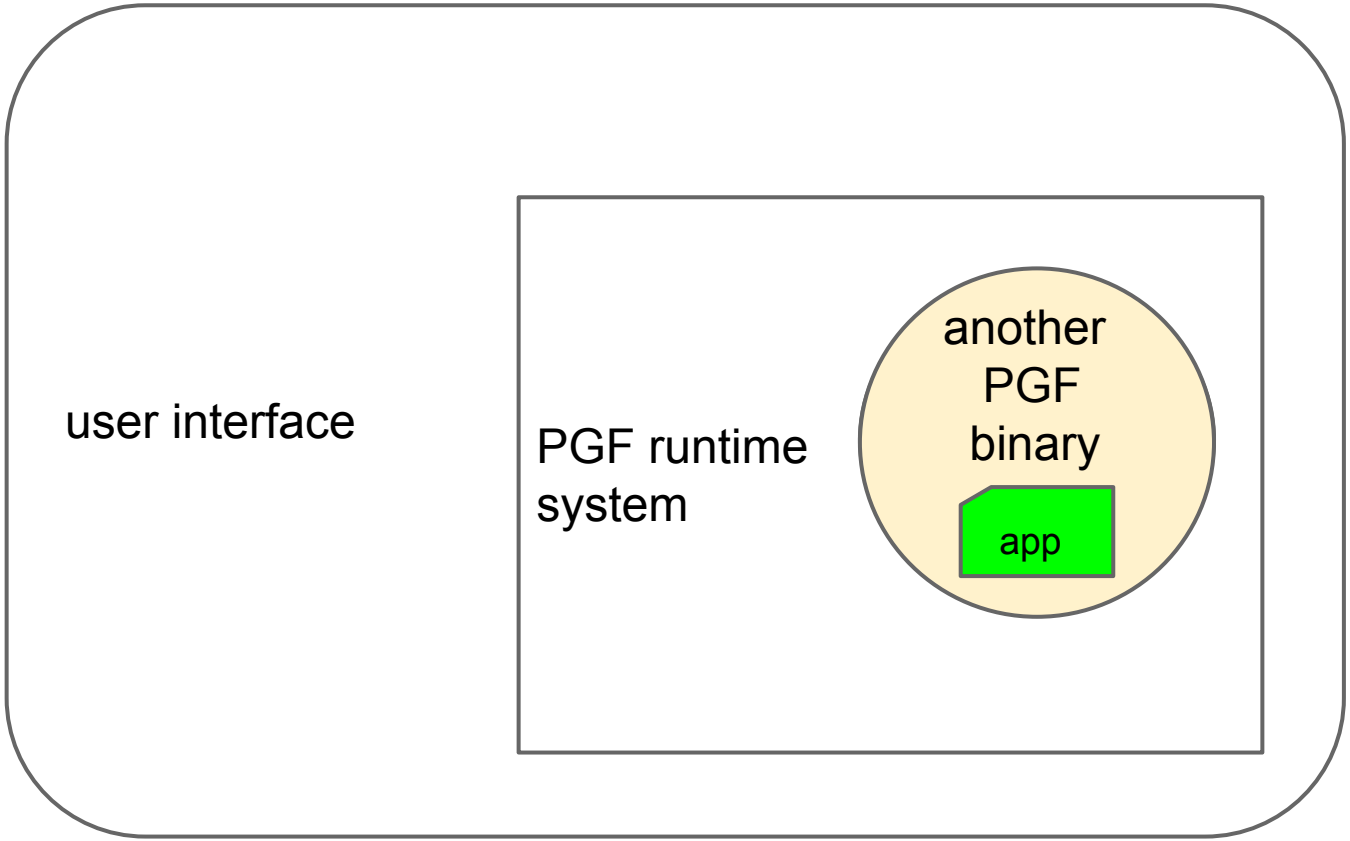

probability
model
GF
source

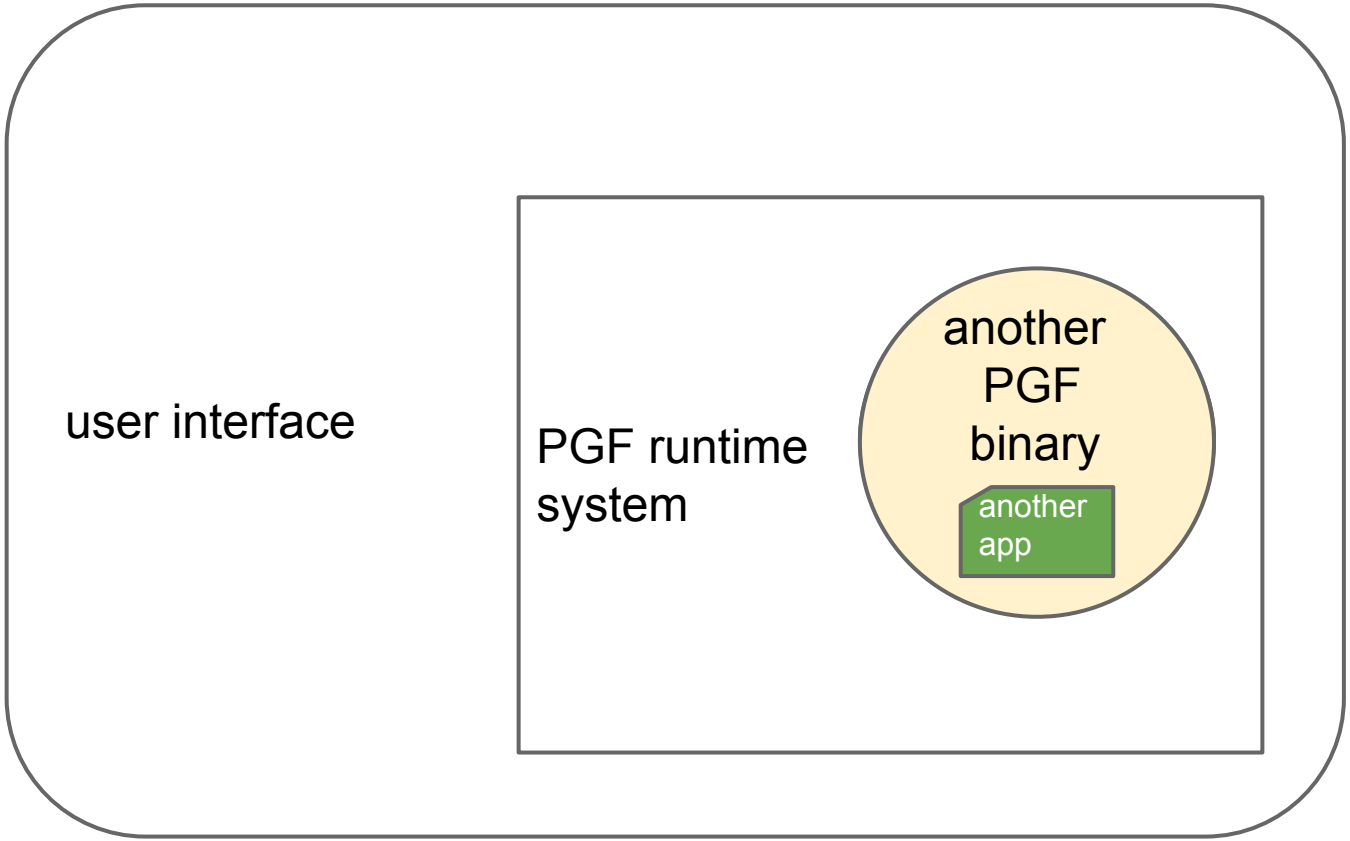

probability
model
GF
source
GF
compiler
PGF
binary


PGF runtime system
PGF binary

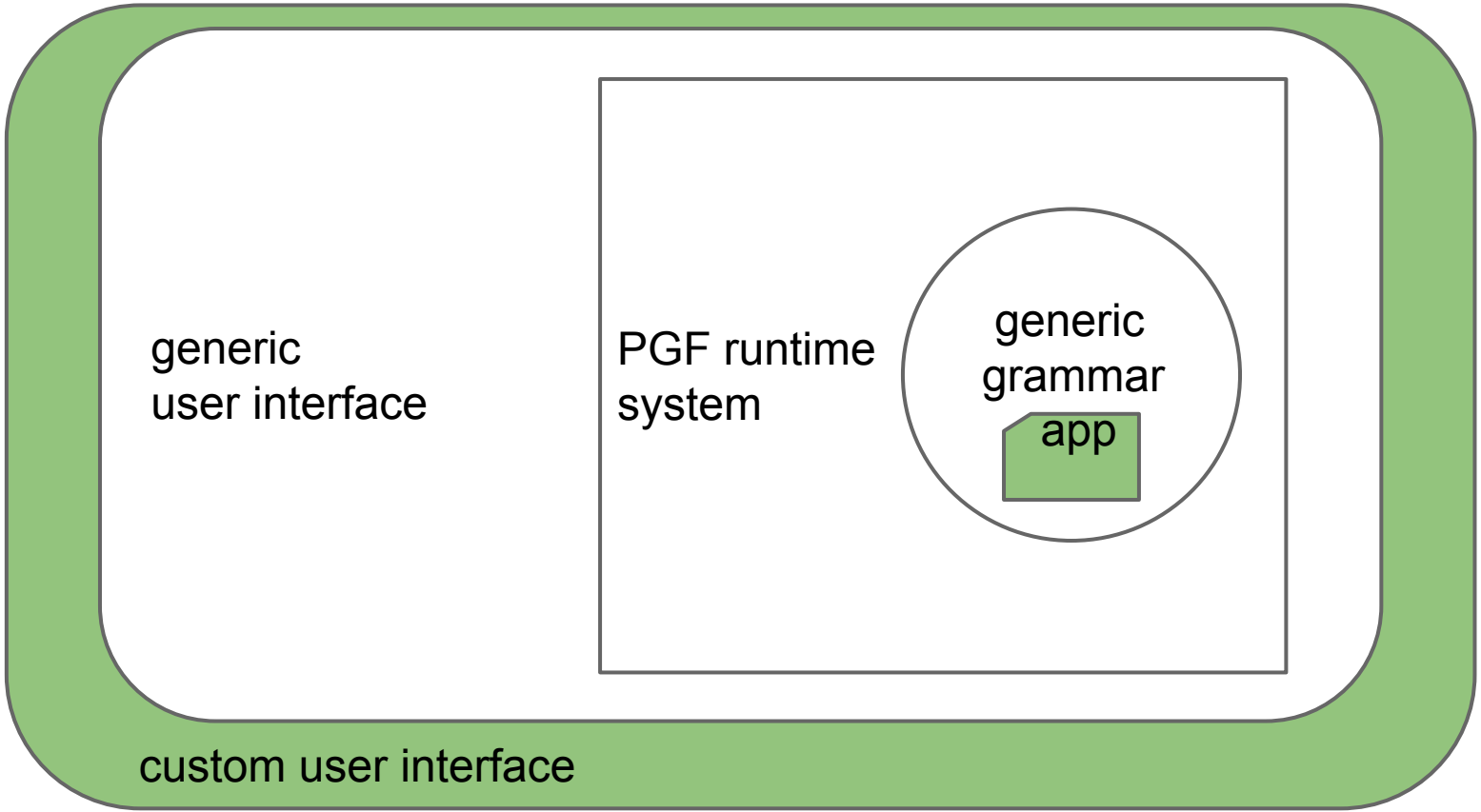

user interface
PGF runtime system
PGF binary


user interface
PGF runtime system
another PGF binary


user interface
PGF runtime system
another PGF binary
app

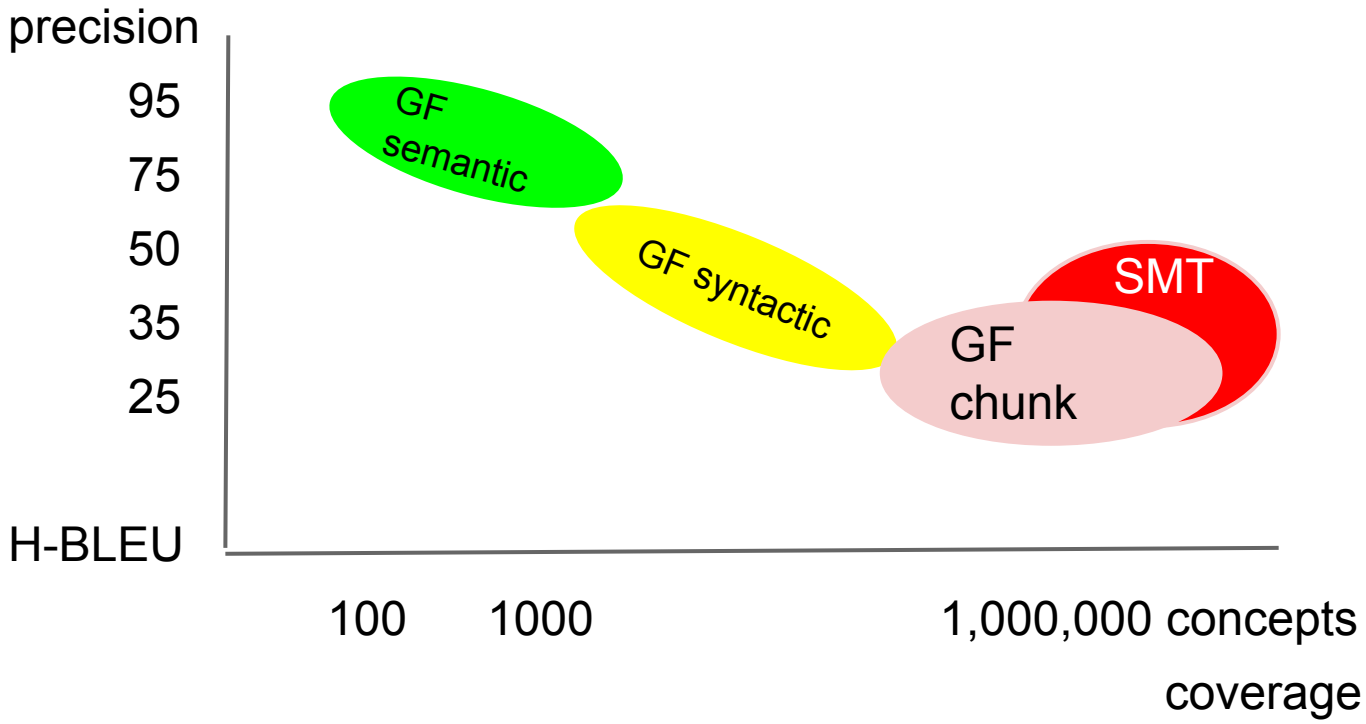

user interface
PGF runtime system
another PGF binary
another app

White: free, open-source. **Green**: a business idea



Results and evaluation

Where we are now



Evaluation, English-Finnish

BLEU scores with human post-edits as reference.
Open-domain results not significant yet.

system	phrasebook (green)	open, aver.	open, yellow	open, red
GF	**89**	26	29	25
Moses		**60**		
Google	44			

Evaluation, English-Chinese (Finnish)

BLEU scores with human post-edits as reference.
Open-domain results not significant yet.

system	phrasebook (green)	open, aver.
GF	**84 (89)**	21 (26)
Moses		**36 (60)**
Google	50 (44)	

Tillgänglighetsdatabasen

Customer of Digital Grammars AB

Texts on accessibility to buildings etc

Corpus 1200 sentences, 8000 words, 1000 unique lemmas.

Evaluation, Swe TD to Fin,Ger,Spa

	GF, correct	GF, BLEU	Google, BLEU
Fin, CNL	48%	**77**	31
Fin, robust	0%	**31**	20
Fin, all	46%	**73**	28
Ger, CNL	44%	**75**	37
Ger, robust	0%	33	**34**
Ger, all	42%	**73**	37
Spa, CNL	34%	**76**	28
Spa, robust	0%	**39**	25
Spa, all	32%	**74**	38

Future work

Improve the lexicon

Split senses

Improve disambiguation

Introduce constructions

Make more comprehensive evaluation

KIITOS

Splitting senses

time

Splitting senses

time_N

time_V

Splitting senses

Zeit

time_N

Mal

Splitting senses

time_1_N Zeit

time_2_N Mal

Splitting senses

time_1_N	Zeit	temps
time_2_N	Mal	fois

Splitting senses

weather_N	Wetter	
time_1_N	Zeit	temps
time_2_N	Mal	fois

Disambiguation

Current model, for abstract trees:

$$P(C\ t_1 \ldots t_n) = P(C) * P(t_1) * \ldots * P(t_n)$$

where *P(C)* for each tree constructor *C* is estimated from its frequency in a corpus.

The context-free tree model

Surprisingly good for syntactic constructors

But almost useless for word senses

*This **time** we will have more **time**.*

Alternative models

Run-time (in “decoding”):
verb + arguments “n-grams” (on tree level)

Compile-time (in grammars):
include constructions and multiwords in lexicon

See also: 4th GF Summer School

July 2015 in Marsalforn, Malta