Comparing *Bottom-up* with *Top-down* Parsing Architectures for the **S**yntax **D**efinition **F**ormalism from a Disambiguation Standpoint

Jurgen J. Vinju

Centrum Wiskunde & Informatica
TU Eindhoven
Swat.engineering
VERSEN

The Syntax Definition Formalism

NOT a complete picture!

1966 Eelco Visser 👶

1977 Jurgen Vinju 👶

1982 Summer

Paul Klint

1984 **SDF**

Jan Heering
Paul Klint

Pregmatic

1992 Incremental SDF

Jan Rekers

2001 Stratego

1998 ToolBus

1997 Scannerless

Mark van den Brand

Eelco Visser

1999 ATerms

Mark van den Brand
Hayco de Jong
Paul Klint
Pieter Olivier

2001 ASF+SDF bootstrapped

2002 Scannerless GLR “finished”

Mark van den Brand
Jeroen Scheerder
Eelco Visser
Jurgen Vinju

2005 ambiguity diagnostics

Jurgen Vinju

2010 Data-dependent Earley

2007 SRNGLR

2008 StrategoXT 0.17

Rob Economopoulos

2009 Rascal

2010 Spoofax

2010 GLL parsing

2010 Rascal bootstrapped

Jurgen Vinju
Paul Klint
Tijs van der Storm
Jurgen Vinju

Elizabeth Scott
Adrian Johnstone

Lennart Kats
Karl Trygve Kalleberg
Rob Vermaas
Eelco Visser

Arnold Lankamp
Jurgen Vinju

2015 Data-dependent GLL

Anastastasia Izmaylova
Ali Afroozeh

Trevor Jim
Yitzha k Mandelbaum
David Walker

scope of this paper

definite clause grammars

Earley’s Algorithm

Tomita’s GLR

Rekers & Farshi’s GLR

ASF+SDF

Stratego XT, Spoofax

Rascal

Izmaylova & Afroozeh’s DD-GLL

Scott & Johnstone’s GLL

Economopoulus’ SRNGLR

Vissers’ Scannerless GLR

Ambiguity in context-free grammars

- context-free general parsing allows non-determinism and ambiguity
- this **enables** modular and extensible syntax definition
- including unpredicted compositions of lexical syntax (“scannerless”)
- but.. **ambiguity** seems to be the “communicating vessel” of modularity
- hence: **declarative disambiguations** and their challenging *implementation*

{ A* b; }

IF IF
ELSE

```
if x == a:
␣␣␣␣print (“not offside”)
```

IF IF=FI THEN
THEN=FI FI

leta = b

Duality of Parsing & Disambiguation

- **Parsing** algorithms create parse trees
- **Disambiguation** algorithms remove parse trees
- **Parsing** is defined by context-free grammars
- **Disambiguation** is defined by disambiguation constructs

```
Stat = “if” Exp “then” Stat “else” Stat
     | “while” Exp Stat
```

In theory disambiguation is orthogonal to/ compositional with parsing!

```
left Exp “*” Exp > left Exp “+” Exp
```

associativity & priority

```
Exp = Id | “if” {reject}
```

keyword reservation

```
Id -/- [a-z]        Id = [a-z]+ !>> [a-z]
```

longest match / maximal munch / shift over reduce

Parsing⊥Disambiguation Architectures

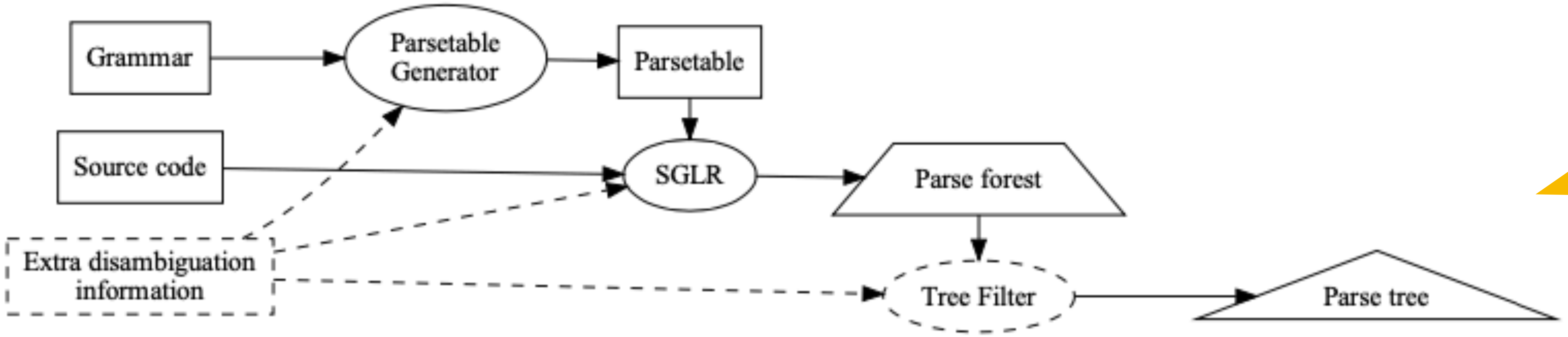


Figure 4 Bottom-up SDF2 architecture based on SGLR

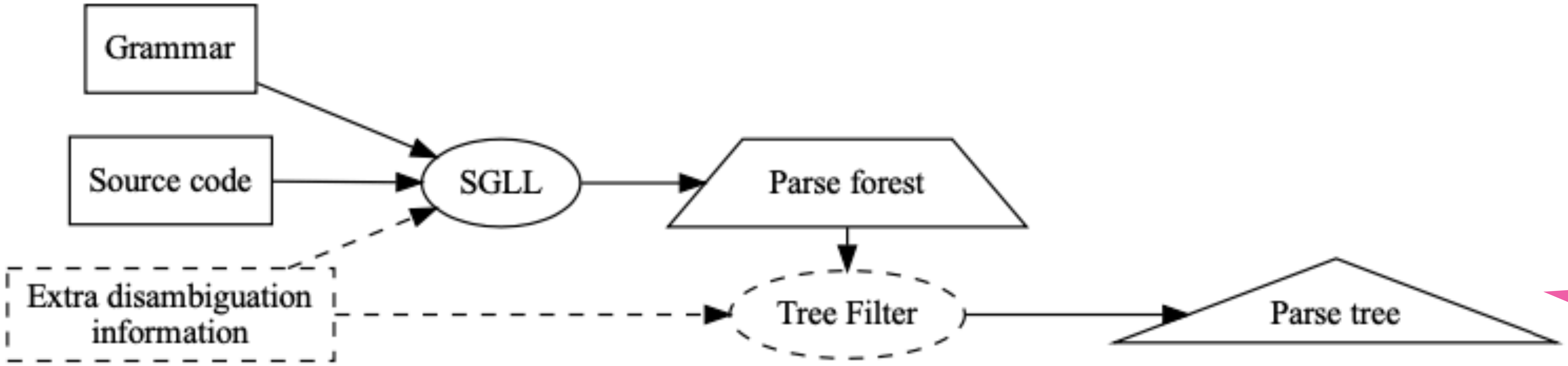


Figure 5 Top-down Rascal architecture based on SGLL

In practice disambiguation is scattered and tangled!

parse tables are partially evaluated grammars

parse functions are “grammars as code”

Bottom-up disambiguation = partially evaluated grammars + ad-hoc disambiguation filters

- priorities/associativity became **reduction filters** in the **modified** SLR table
- reject rules became additional synchronization of **reductions** per “level”
- follow restrictions became both **goto sets filters** plus reduction filters
- **disambiguation code easily breaks parsing algorithm correctness**
- Every disambiguation filter requires **a new theory**: is the new algorithm correct? are the old data structures still sufficient? do generated parsers still compose? etc.

Top-down disambiguation = parsing functions plus prediction & completion predicates

- Every disambiguation construct is a predicate kind over state of the parser
- Parsing functions can **transparently** fail for more reasons than grammar+input
- The concept of disambiguation code can be **generalized and encapsulated**
- New disambiguation constructs do **not require new implementation theory**
- Data structures must be protected for non-context-freeness like GLR.

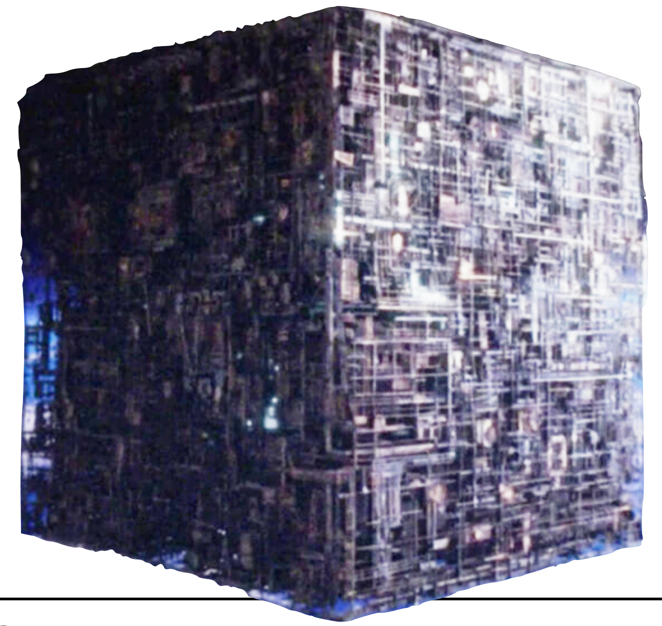
One more step: a general theory of disambiguation

- Generalized prediction/completion filters with GLL:
  - **Implementation** theory for disambiguations.
  - Preferable from an understandability viewpoint
  - just as fast or faster than SGLR
- What is a generalized theory of disambiguation on the grammar level?
  - data-dependent context-free grammars [Jim, Mandelbaum, Walker]
  - grammar rules + data parameters + **predicates**
  - “Iguana” is a top-down data-dependent GLL [Izmaylova, Afroozeh]
  - DD-CFG’s lift filtering predicates from the implementation to the specification level
- From *orthogonal* ad-hoc specification to *integrated* generalized disambiguation

orthogonality brought us **understanding**, now integration is bringing us **generality**.

disambiguation != grammar transformation

context-sensitive grammars != data-dependent grammars



The Syntax Definition Formalism

NOT a complete picture!

1966 Eelco Visser 👶

1977 Jurgen Vinju 👶

1982 Summer

Paul Klint

1984 **SDF** Pregmatic

Jan Heering
Paul Klint

1992 Incremental SDF

Jan Rekers

1997 Scannerless

Mark van den Brand
Eelco Visser

1998 ToolBus

1999 ATerms

2001 Stratego

2001 ASF+SDF bootstrapped

2002 Scannerless GLR “finished”

Mark van den Brand
Hayco de Jong
Paul Klint
Pieter Olivier

Mark van den Brand
Jeroen Scheerder
Eelco Visser
Jurgen Vinju

2005 ambiguity diagnostics

Jurgen Vinju

2007 SRNGLR

Rob Economopoulos

2008 StrategoXT 0.17

2009 Rascal

2010 Spoofax

2010 GLL parsing

2010 Rascal bootstrapped

2010 Data-dependent Earley

Jurgen Vinju
Paul Klint
Tijs van der Storm
Jurgen Vinju

Elizabeth Scott
Adrian Johnstone

Lennart Kats
Karl Trygve Kalleberg
Rob Vermaas
Eelco Visser

Arnold Lankamp
Jurgen Vinju

Trevor Jim
Yitzha k Mandelbaum
David Walker

2015 Data-dependent GLL

Anastastasia Izmaylova
Ali Afroozeh

scope of this paper

definite clause grammars

Earley’s Algorithm

Tomita’s GLR

Rekers & Farshi’s GLR

ASF+SDF

Stratego XT, Spoofax

Rascal

Izmaylova & Afroozeh’s DD-GLL

Scott & Johnstone’s GLL

Economopoulus’ SRNGLR

Vissers’ Scannerless GLR