Analysing Structural Dependencies in Software Product Lines

Using Distribution and Network Statistics

by Sven Apel, Sergiy Kolnesikov, Stefan Sobernig;

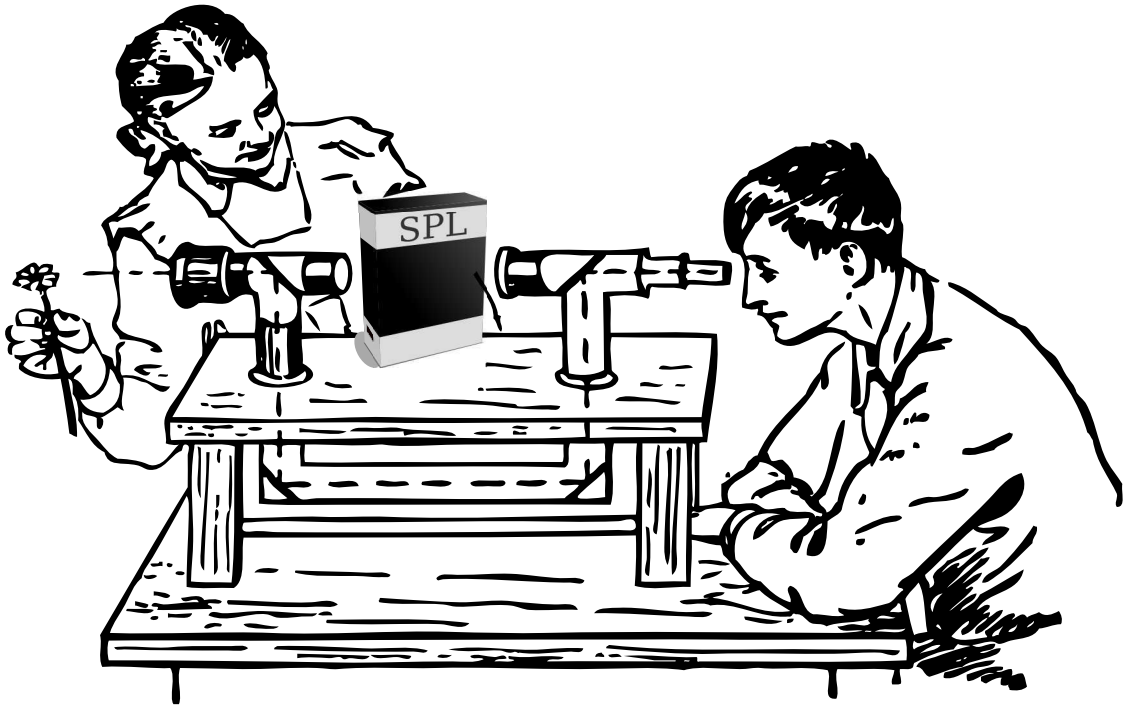
in cooperation with Jörg Liebig and Norbert Siegmund

March 21, 2012

Structure of the talk

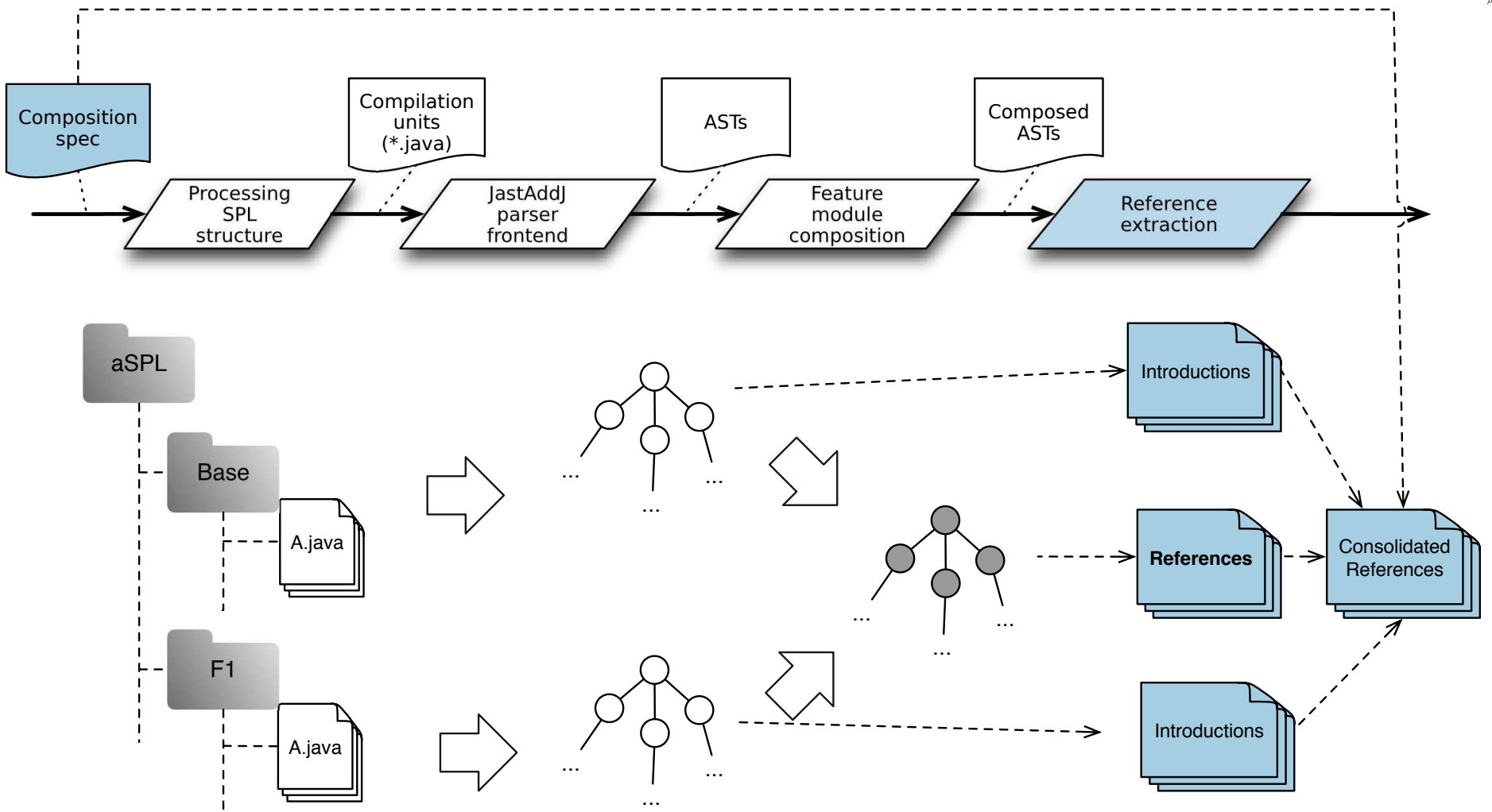
WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

1. Motivation
2. Gathering syntactic reference data from Java-based SPLs (Fuji)
3. Notions of reference
4. Degree statistics over reference data



“Ufff! A false x-ray apparatus for SPLs?”
(Illustration adapted from Perelman (1913/2008): Physics for Entertainment, Fig. 95, p. 132)

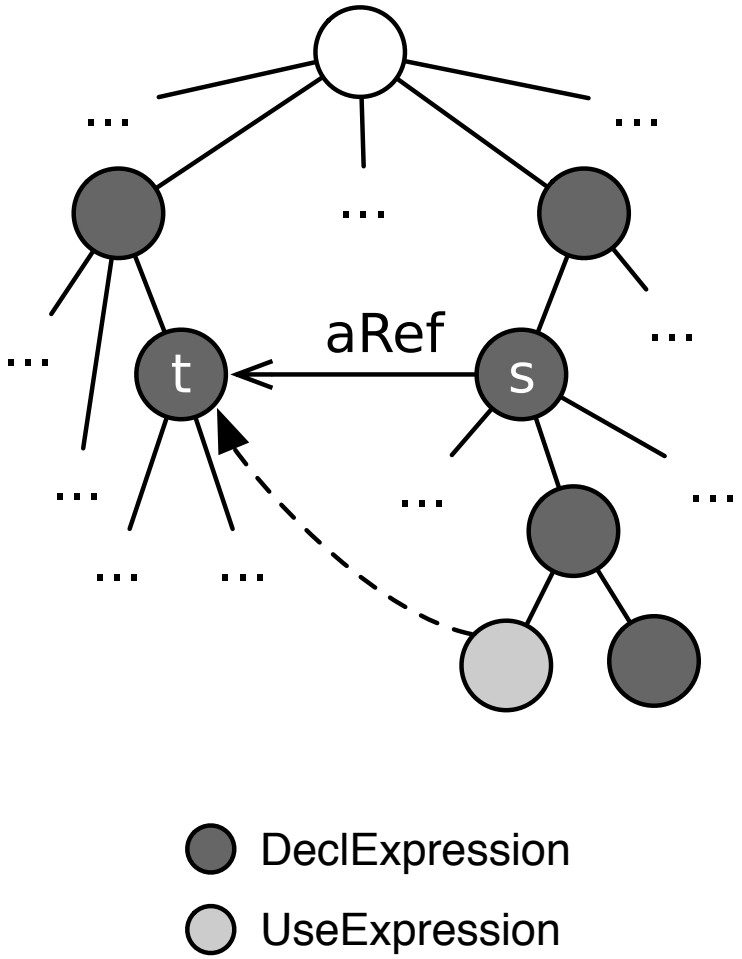
Data collection for Fuji-driven SPLs



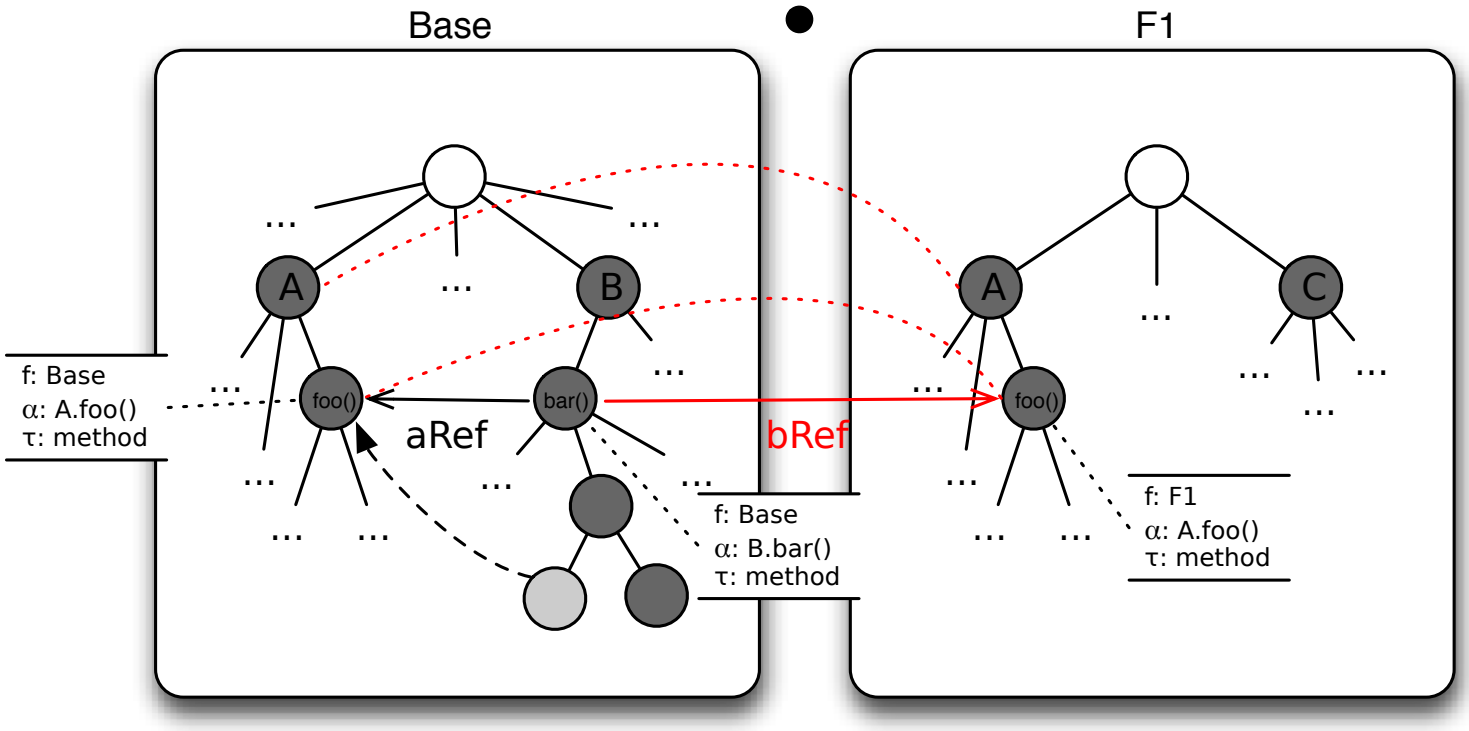
See [Kol11] for details on Fuji

A generic notion of reference

- In the Fuji/Java context, relevant `DeclExpressions` are ...
  - Class declarations
  - Constructor and initializer declarations
  - Method declarations
  - Field declarations
- Relevant `UseExpressions` are ...
  - Class-instance creation expressions
  - Field access expressions
  - Method invocation expressions



An extended notion of “Fuji reference”



$$R : F \times A \times \Gamma \mapsto F \times A \times \Gamma$$
$$= ((\{f_i, \alpha_m, \tau\}, \{f_j, \alpha_n, \tau\}), \dots)$$
$$1 \leq i, j \leq |F|,\ i \neq j$$
$$1 \leq m, n \leq |A|,\ m \neq n$$
$$\tau \in \Gamma : \{\texttt{type}, \texttt{field}, \texttt{method}, \texttt{constructor}\}$$

- Intra-module references, e.g.: `aRef`

  $(\{\texttt{Base}, \texttt{B.bar()}, \texttt{method}\}, \{\texttt{Base}, \texttt{A.foo()}, \texttt{method}\})$

- Inter-module references, e.g.: `bRef`

  $(\{\texttt{Base}, \texttt{B.bar()}, \texttt{method}\}, \{\texttt{F1}, \texttt{A.foo()}, \texttt{method}\})$

Available reference data sets (SPLs)

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- References and introductions data of **28** SPLs
- **Feature modules**: $\Delta = 99$ (BerkeleyDB) vs. $\delta = 5$ (Raroscope)
- **Introductions**: $\Delta = 9379$ (BerkeleyDB) vs. $\delta = 43$ (EPL)
- **(Unique) references**: $\Delta = 52890$ (BerkeleyDB) vs. $\delta = 80$ (EPL)
- **(Unique) inter-module references**: $\Delta = 30061$ (BerkeleyDB) vs. $\delta = 51$ (Raroscope)

Fuji reference space

The reference data can be sliced by . . .

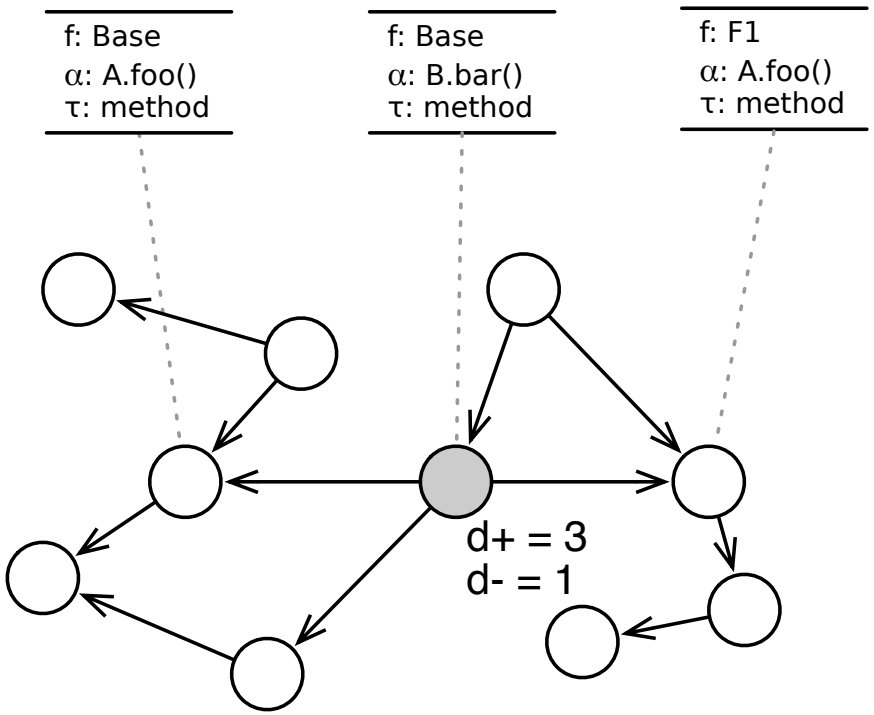
- . . . pairs of syntactic categories:

source → target	field	method	constructor	type
field				
method				
constructor				
type				

  - $4 \times 4 = 16$ subsets of reference data per SPL
  - Column-wise aggregation: field accesses, method accesses

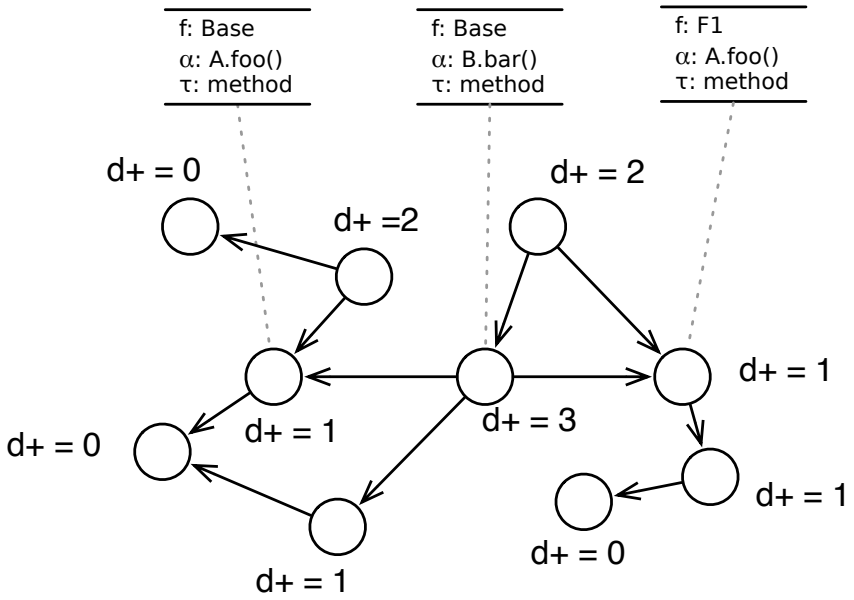
- . . . the level of abstraction: per-element vs. per-feature-module
- . . . the kind of reference: intra-module vs. inter-module
- . . . the reference direction: in- vs. outgoing
- . . . or any combination of the above!

Degrees in reference networks



- For each possible reference-data slice, e.g.:
  - method→method
  - per-element,
  - intra- and inter-module references ...
- ...we can form a graph G_R
- We can identify *local* properties of each node n (e.g., element, feature module), such as:
  - out-degree $d^+(n)$
  - in-degree $d^-(n)$
  - ...as well as their relative forms
- and *global*, degree-based characteristics
  - Average, minimum, and maximum degrees: $\phi d^+(G_R)$, $\phi d^-(G_R)$, $\Delta d^+(G_R)$, $\Delta d^-(G_R)$, $\delta d^+(G_R)$, $\delta d^-(G_R)$
  - **(In-/Out-) Degree distribution**

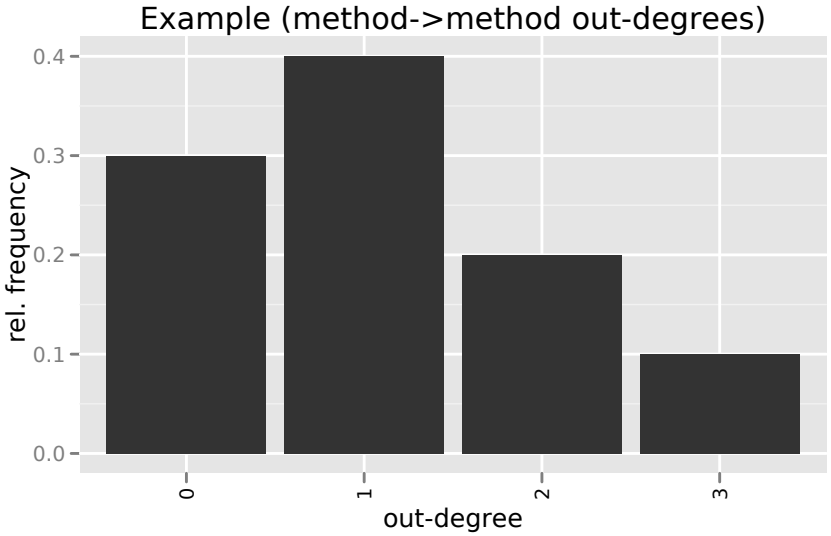
An exemplary out-degree distribution



For the given example, the *global*, out-degree characteristics are:

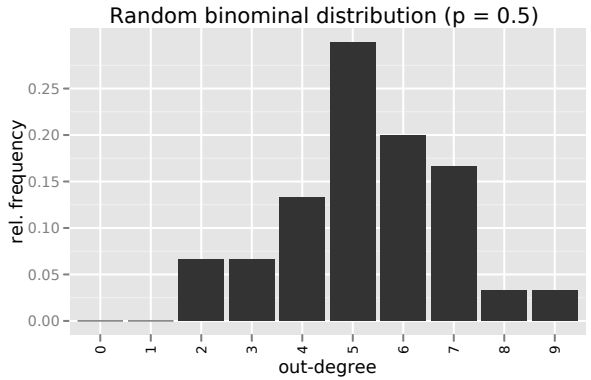
- $\Delta d^+(G_R) = 3$
- $\delta d^+(G_R) = 0$
- $\phi d^+(G_R) = ?$
- The distribution of out-degree frequencies in G_R:

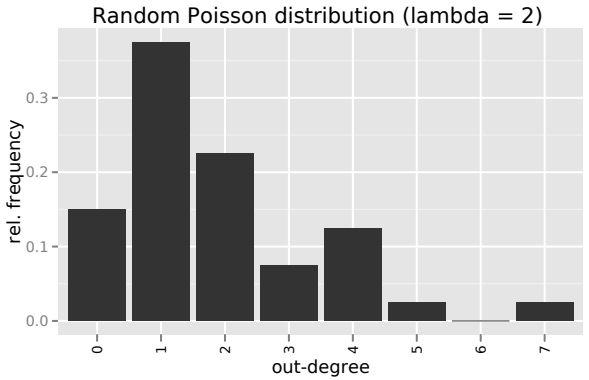
out-degree	0	1	2	3
frequency	3	4	2	1
rel. frequency	0.3	0.4	0.2	0.1

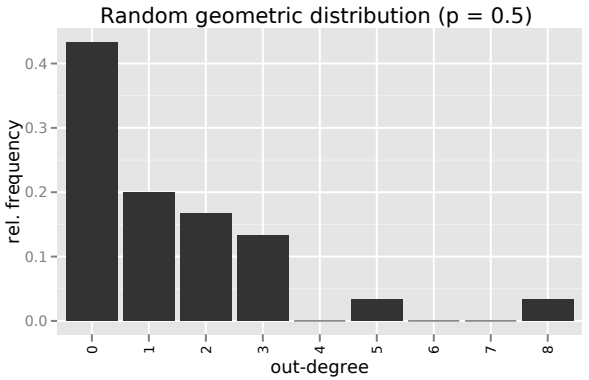


Do degrees follow a particular distribution?

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS







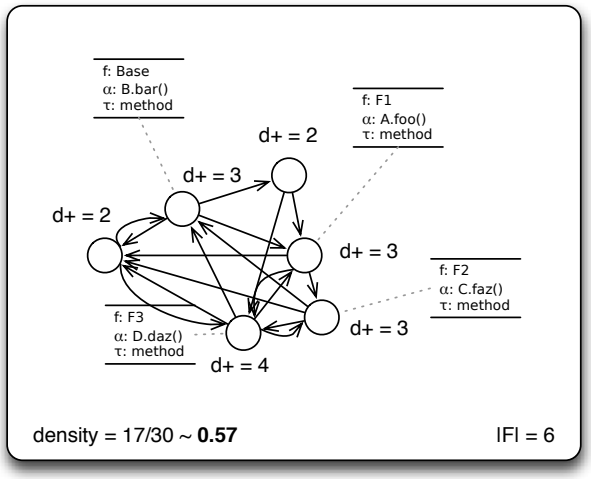
Towards Smell Detection, Ex. 1 (1)

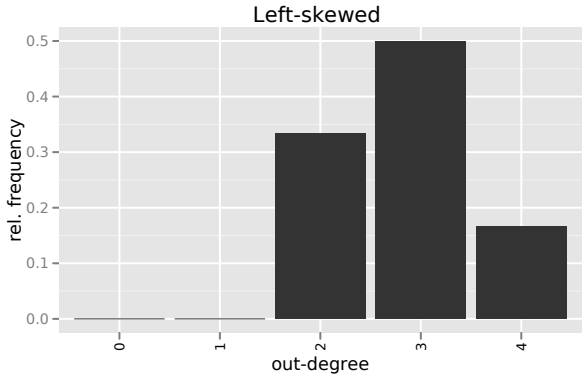
WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

A variant of a Feature Envy [Fow03]: Excessive Inter-Module Method Invocations

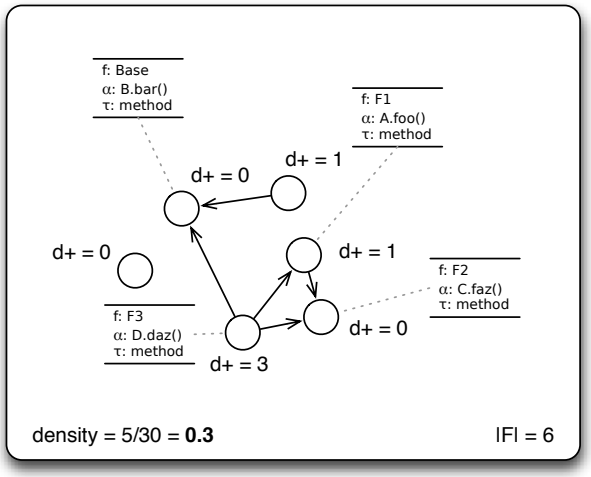
1. For a given SPL, choose the appropriate data slice: per-element, *→method references, out-degrees, inter-module
2. Indicator measures: network density, **strength** distribution (skewness)

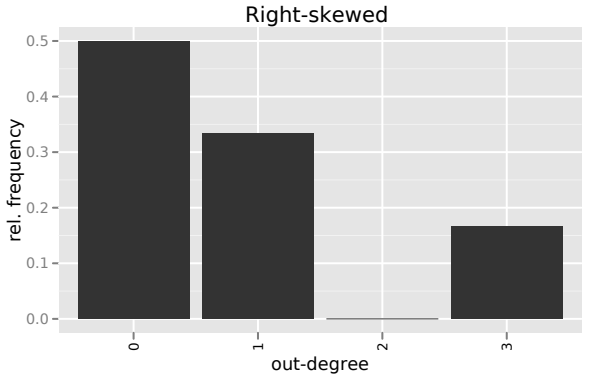
Possible envies?





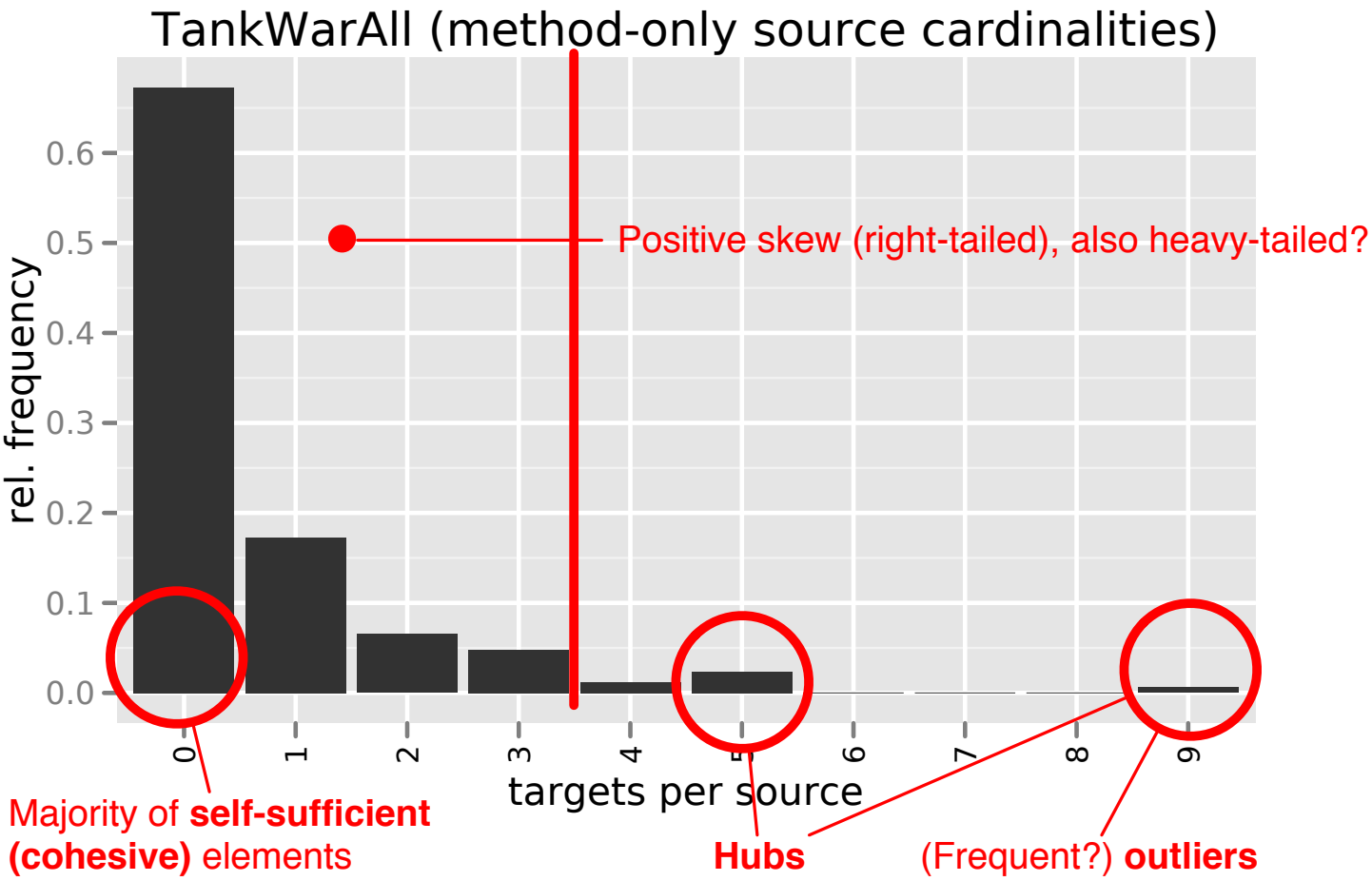
Mostly ego-centric?





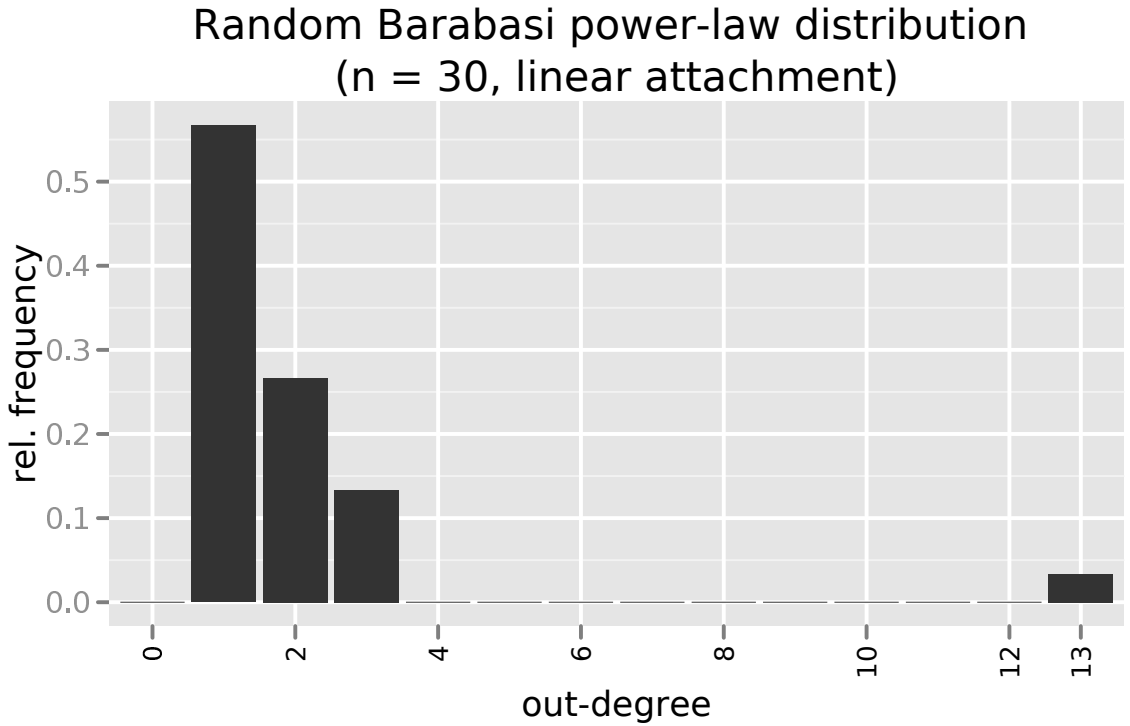
Towards Smell Detection, Ex. 1 (2)

Can one observe method-level feature Envies in the TankWar Fuji-SPL?



Research question(s): Are we looking at . . .

. . . variants of a *power-law* distribution for our SPLs and for all/certain slices?



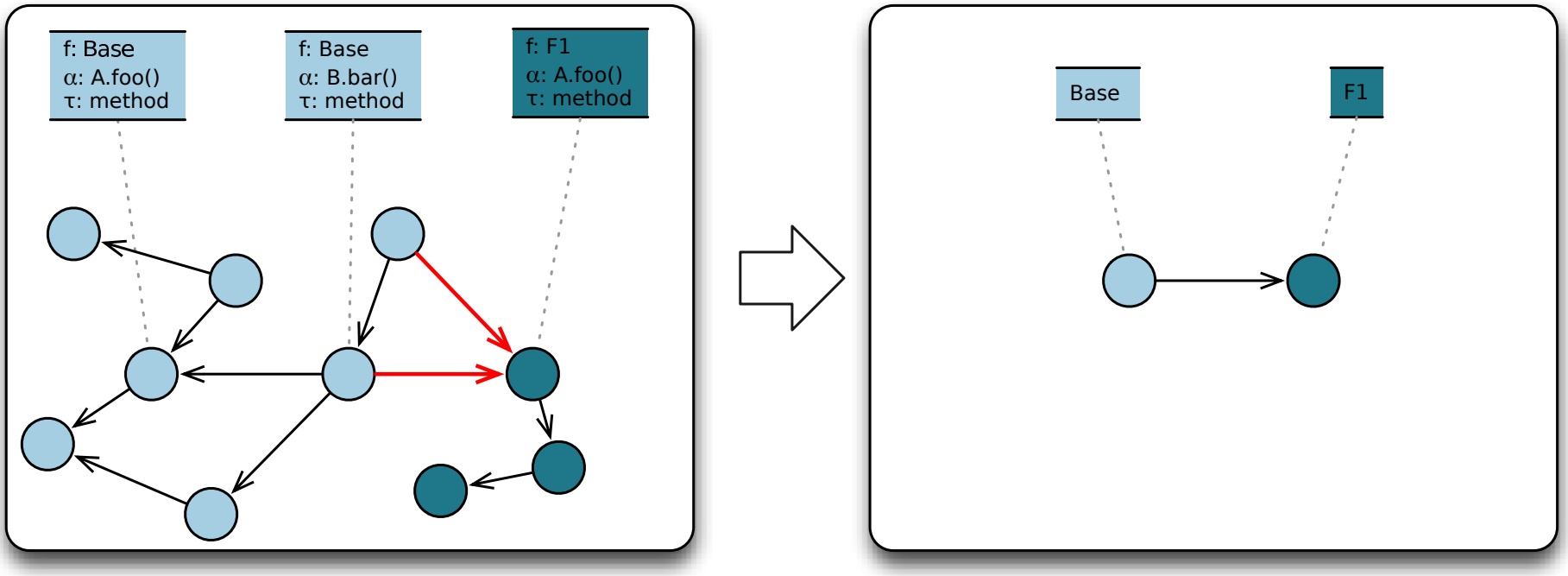
Why would one expect to find such a degree distribution in a Fuji SPL?

On Power Laws (or the like)

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

- Numerous reports for important OO program structures:
  - Many hierarchical (ownership) and non-hierarchical (statements) relations within and between class boundaries; Qualitas Corpus [TSWW11]; also states the correspondence between overall connectivity and between-class connectivity.
  - Classes and class relations such as inheritance and aggregation; JDK class libraries [WC03]
  - Variable type annotations, object creation, and parameter passing; Squeak and VisualWorks Smalltalk [MPST04].
  - Global layout of 60 object graphs, extracted from object-object references from sources as diverse as C++ frontend to GCC, the Self runtime, and Java ArgoUML [PNFB05]
- Underlying forces (development processes and practises)
  - Preferential attachment in the development process
  - OO design patterns (e.g., template & hook methods, abstract class designs)
  - Key classes

Per-feature-module statistics



- Only inter-module reference are considered
- Per-element references become aggregated/subsumed by per-module references (edge weights)

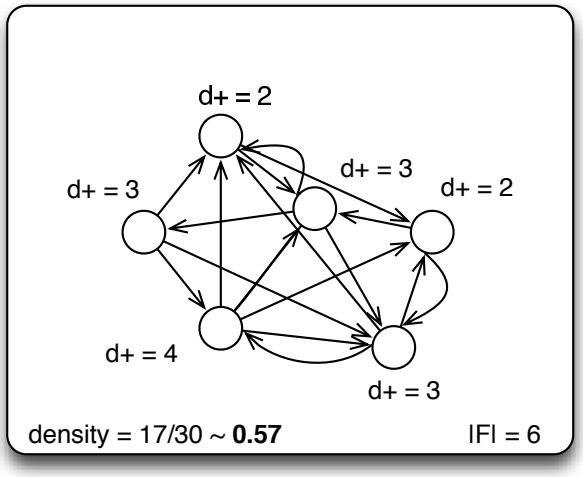
Towards Smell Detection, Ex. 2

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

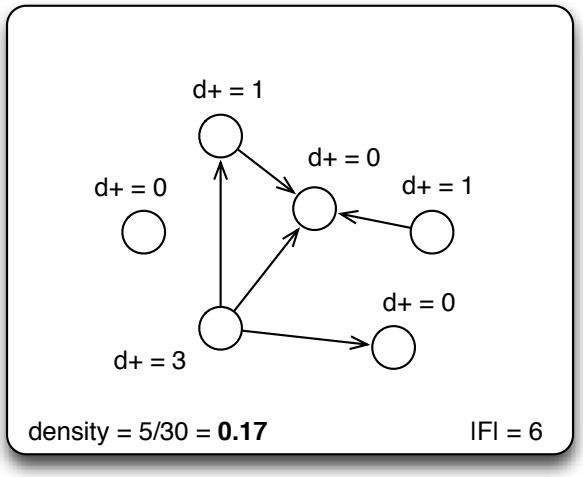
Direct Field Accesses vs. Feature-Module Boundaries [AKL+12]

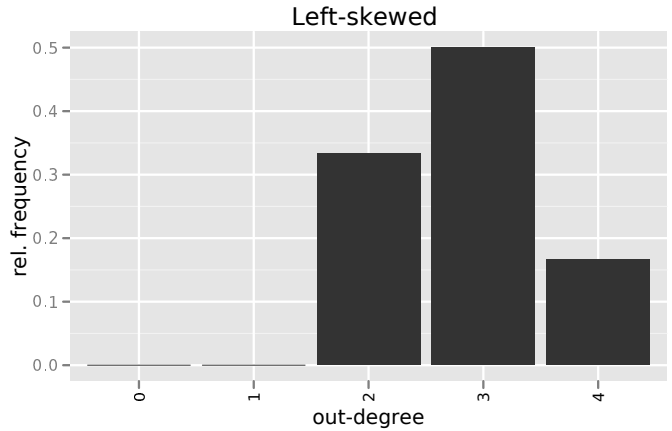
1. For a given SPL, choose the appropriate data slice: per-feature, *→field references, out-degrees, inter-module
2. Indicator measures: network density, degree distribution (skewness)

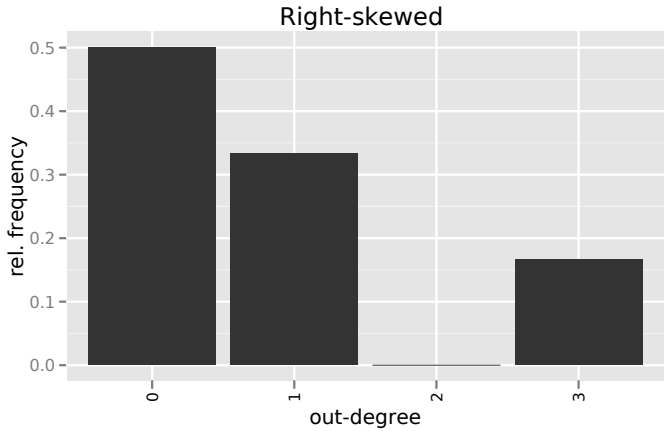
Relatively *high* coupling



Relatively *low* coupling





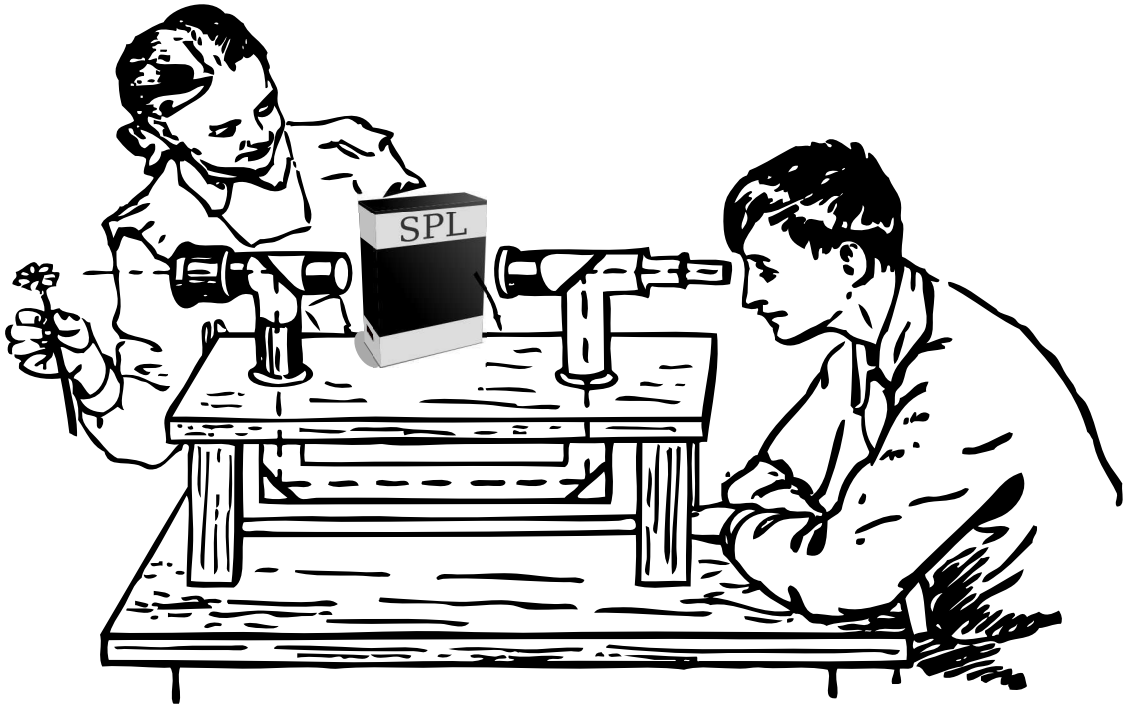


Closing: Further directions of analysis

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

1. Short-term objectives (**Stefan**, **Sven**, **Sergiy**)
   - Additional slicing dimensions: Incorporate co-occurrence dependencies between feature modules (never, always, optional)
   - Usage frequencies of introductions
   - Distribution statistics for field and method accesses per feature module
   - Local/global characteristics of *derivative* features (**Jörg**)
   - Detection/calibration of *intrinsic* (statistical) thresholds for degree and distribution statistics
   - Input for Performance Prediction (see **Norbert**'s and **Sergiy**'s talks): Higher-Order (e.g., triangles) and Hot-Spot Features (hubs, betweeness)
2. Mid-term objectives
   - Detection/calibration of *extrinsic* thresholds, by incorporating the non-functional data profiles of the SPLs (**Sergiy**, **Norbert**, **Jörg**)
   - Weighted network statistics (**Stefan**, **Sven**, **Jörg**)
     - Edge weights based on reference counts (per-element, per-feature)
     - Attribute measures as edge weights (e.g., degrees of coupling and scattering, . . .)
     - . . .

Shooting time!



“What do you see through the walls of SPL code data?”
(Illustration adapted from Perelman (1913/2008): Physics for Entertainment, Fig. 95, p. 132)

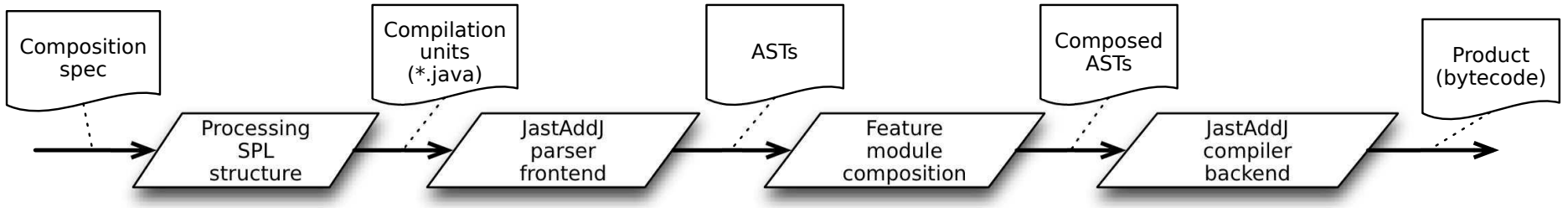
References

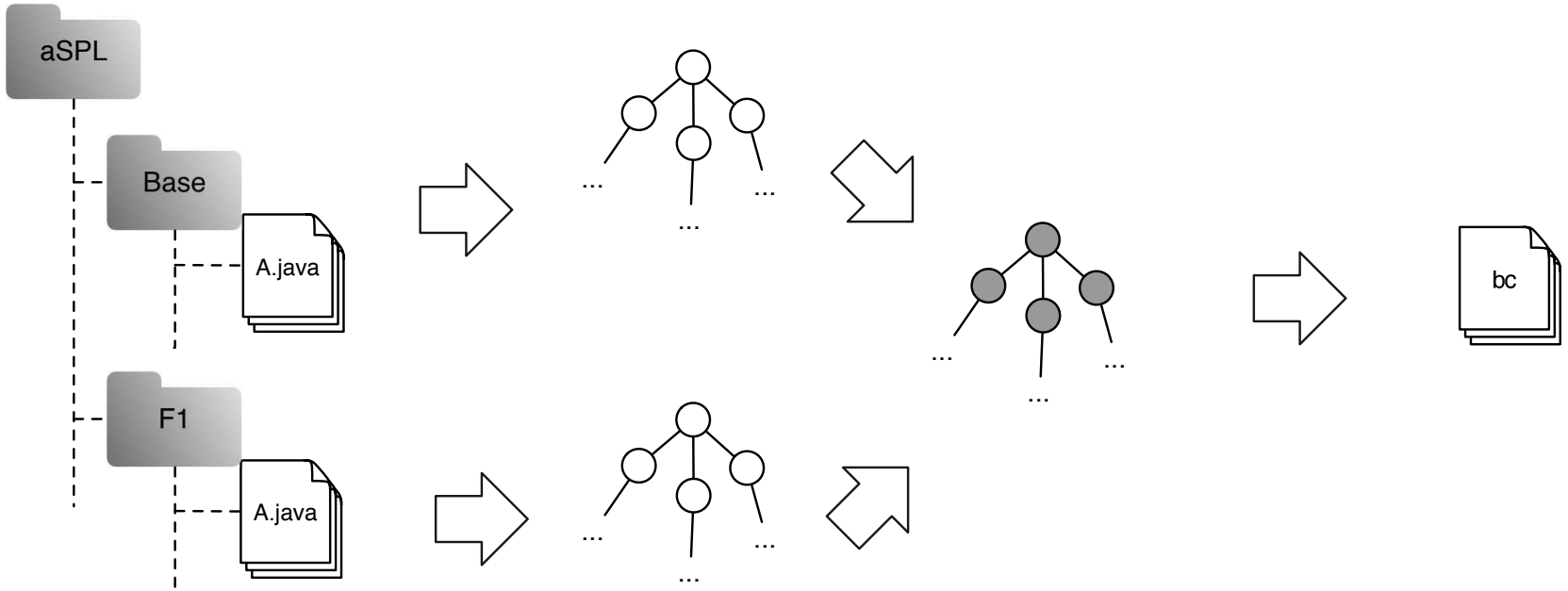
- Sergiy Kolesnikov. Ein erweiterbarer Compiler zum feature-orientierten Programmieren in Java. Talk at the FOSD Annual Meeting 2011, Dresden, März 2011
- Craig Taube-Schock, Robert Walker, and Ian Witten. Can we avoid high coupling? In *ECOOP 2011 – Object-Oriented Programming*, volume 6813 of *Lecture Notes in Computer Science*, pages 204–228. Springer-Verlag, 2011
- M. Marchesi, S. PINNA, N. SERRA, and S. Tuveri. Power laws in smalltalk. In *Proceedings of the 12th Smalltalk Joint Event*, 2004
- Richard Wheeldon and Steve Counsell. Power Law Distributions in Class Relationships. *Proceedings of the IEEE International Workshop on Source Code Analysis and Manipulation (SCAM'03)*, pages 1–10, 2003
- Sven Apel, Sergiy Kolesnikov, Jörg Liebig, Christian Kästner, Martin Kuhlemann, and Thomas Leich. Access control in feature-oriented programming. *Science of Computer Programming*, 77(3):174–187, 2012
- Martin Fowler. *Refactoring – Improving the Design of Existing Code*. The Addison-Wesley object technology series. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003
- Lionel C. Briand, John W. Daly, and Jürgen K. Wüst. A Unified Framework for Coupling Measurement in Object-Oriented Systems. *IEEE Transactions on Software Engineering*, 25:91–121, 1999

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Appendix

Fuji-driven SPLs

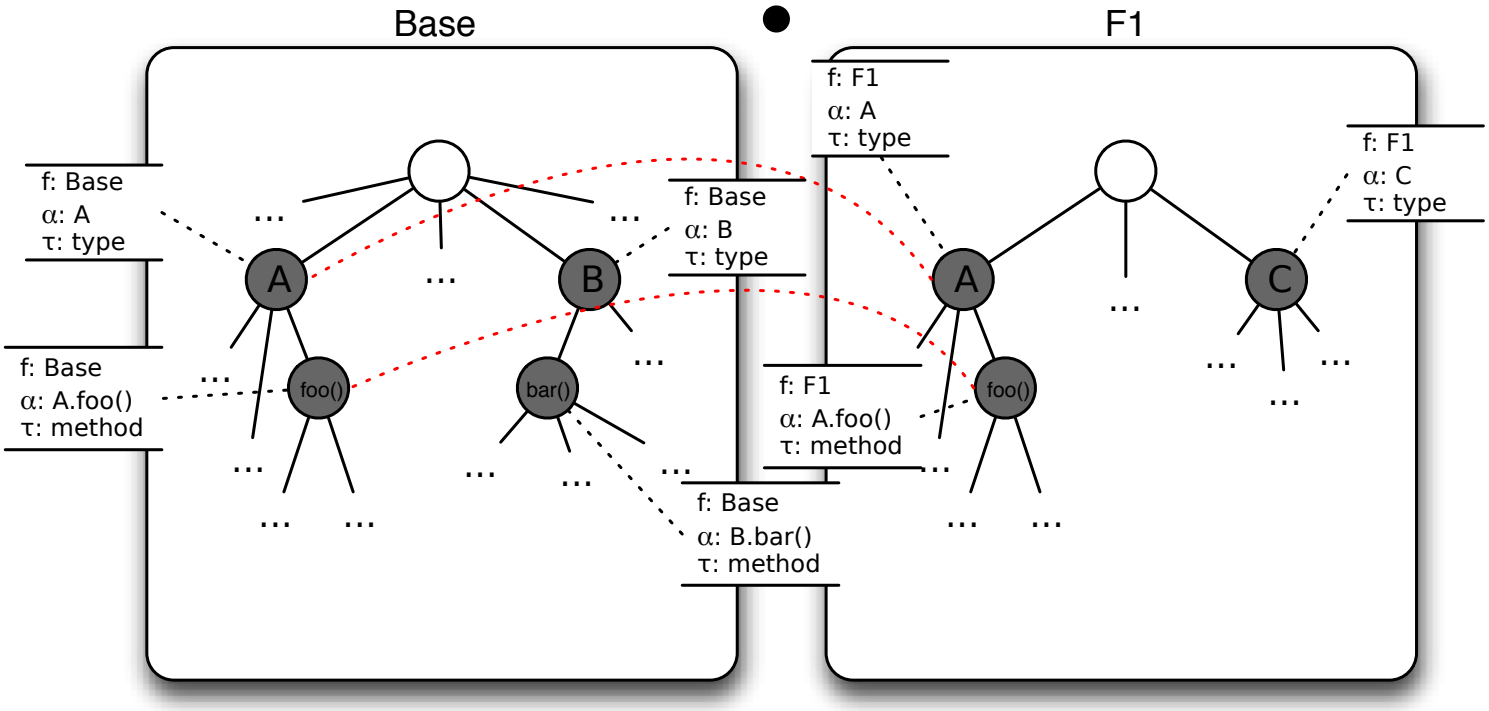




See [Kol11] for details on Fuji

Feature-module composition

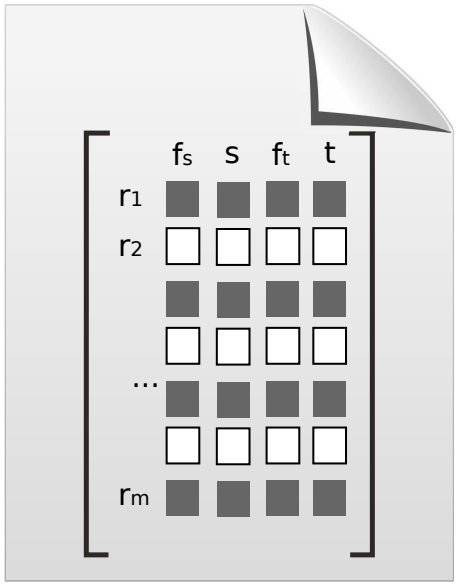
Fuji composes feature modules using AST superimposition: $Base \bullet F1$



Each declaration element are identified by ...

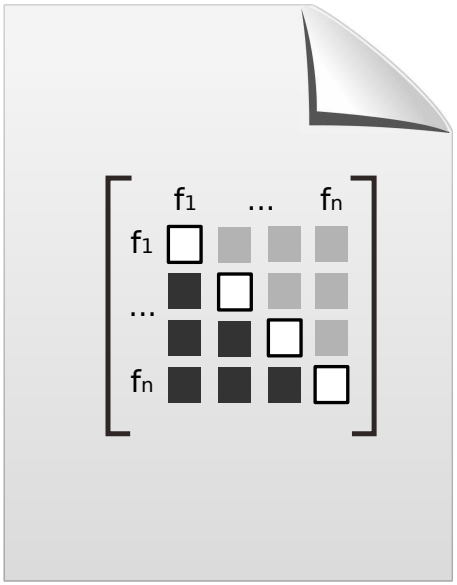
- ... its introducing feature module f: `Base`, `F1`
- ... its label α: `A`, `B`, ..., `foo()`
- ... its syntactic category τ: `type`, `method`

Exploratory issue: Per-feature-module statistics (1)



Reference data
(showing source and target feature per reference)

cross-tabulation



Asymmetric feature-feature matrix