Fine-grained Language Composition: A Case Study

Edd Barrett

Carl Friedrich Bolz

Lukas Diekmann

Laurence Tratt

KING'S College LONDON

Software Development Team

2016-07-22

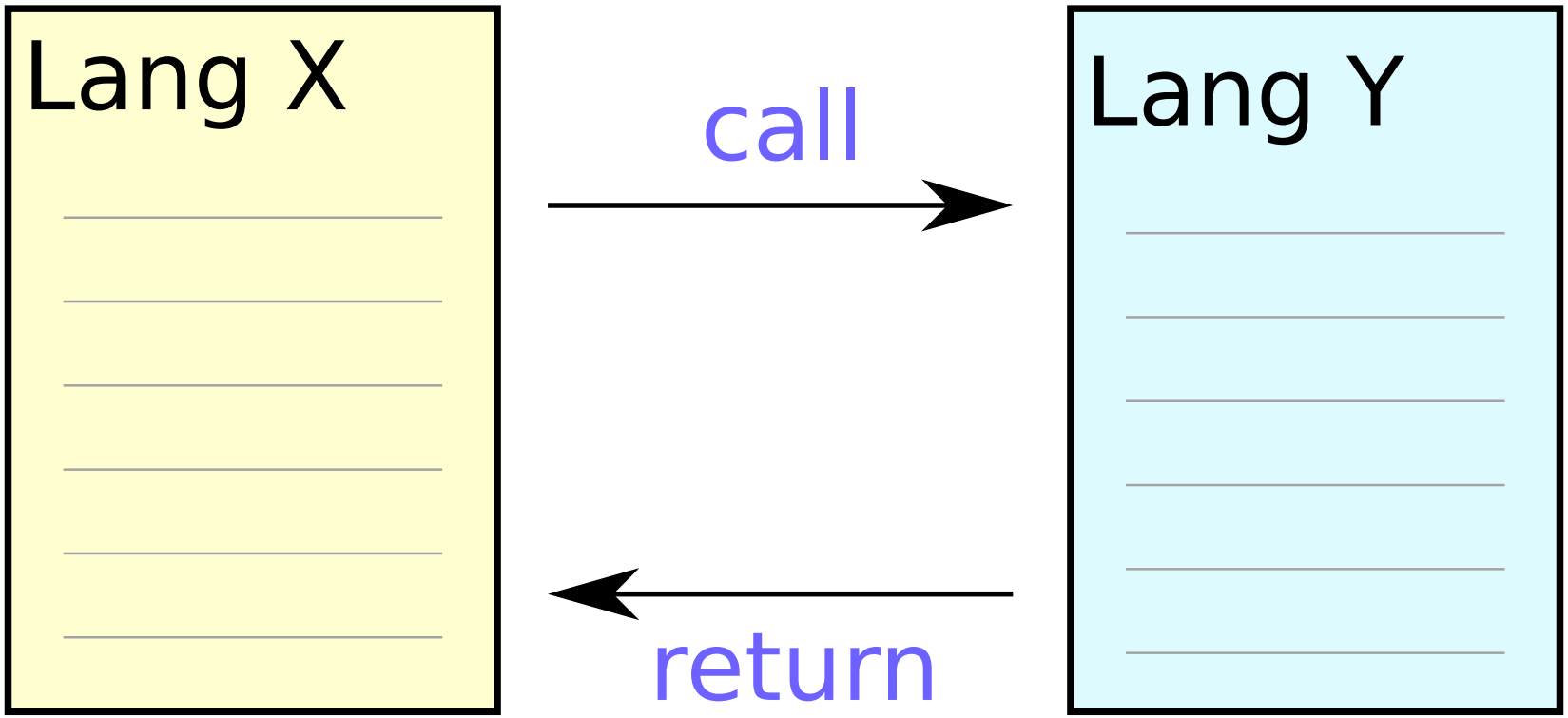
Language Composition

“The ability to write a computer program in a mix of programming languages.”

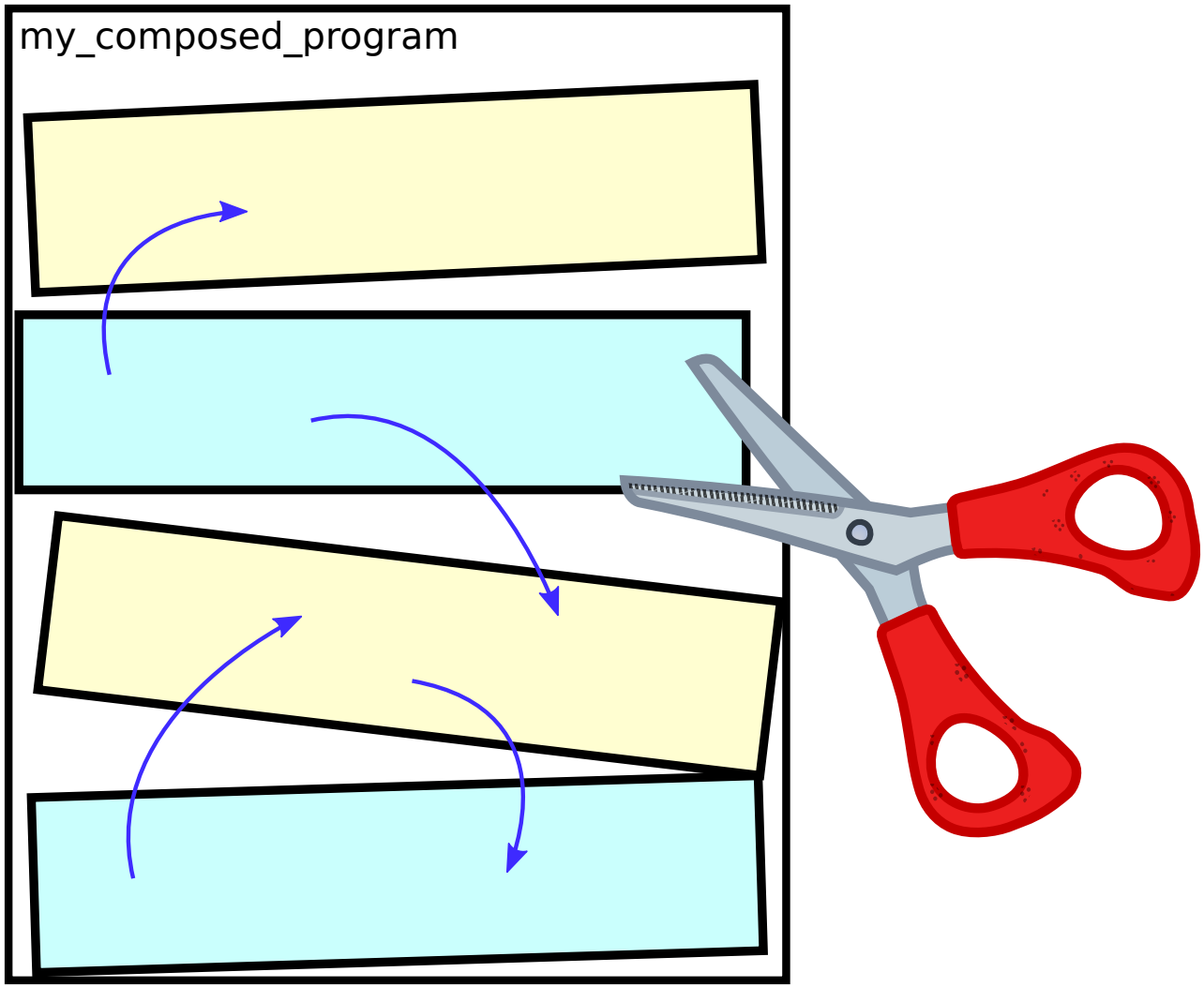
Why Compose Languages?

- Choose the best language for the job.
- Access to a broader set of libraries.
- Language migration.

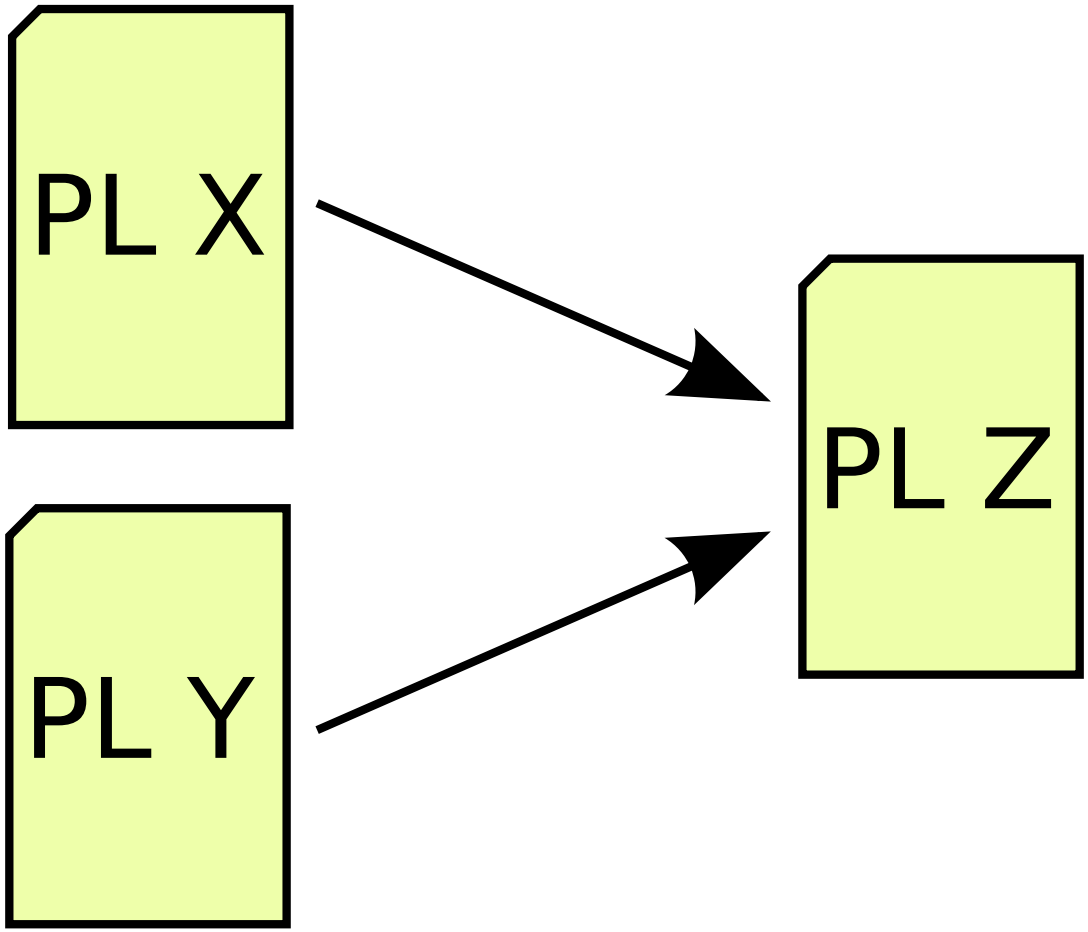
Beyond the FFI



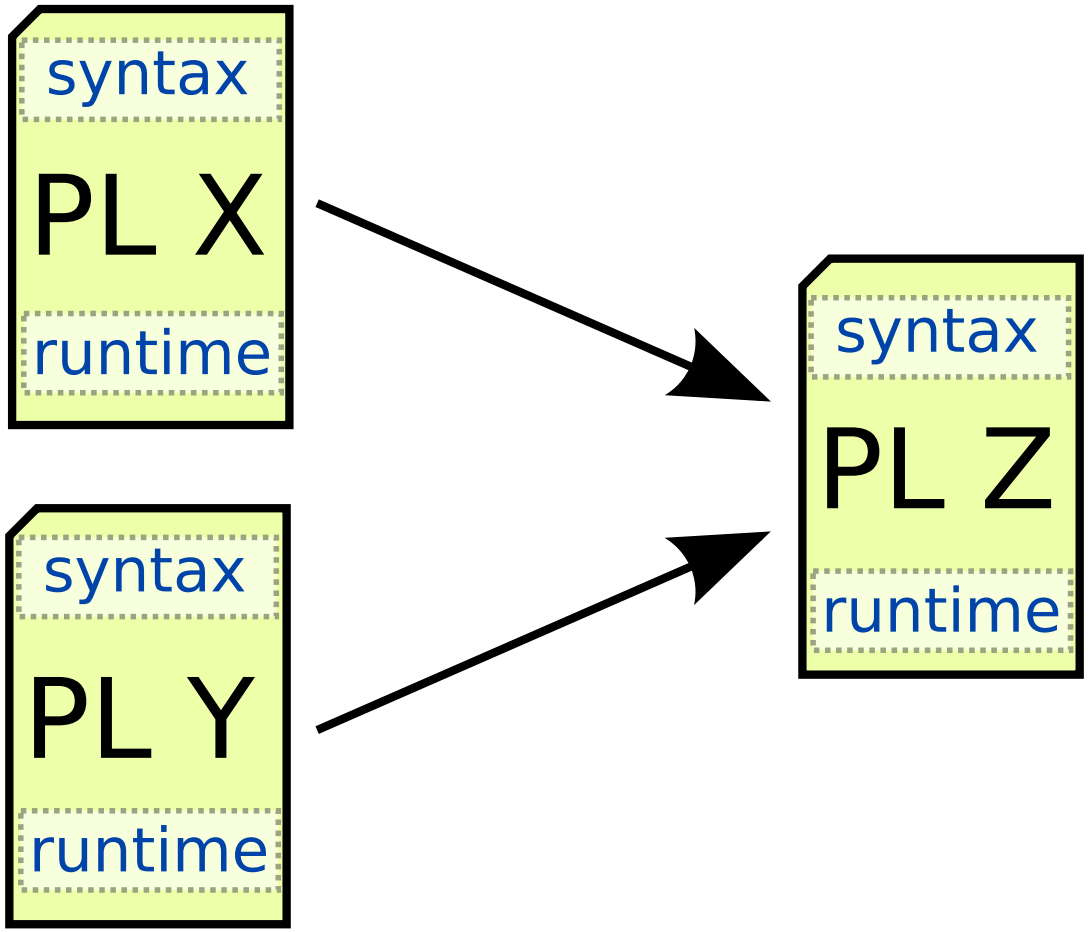
Beyond the FFI



Breaking it Down

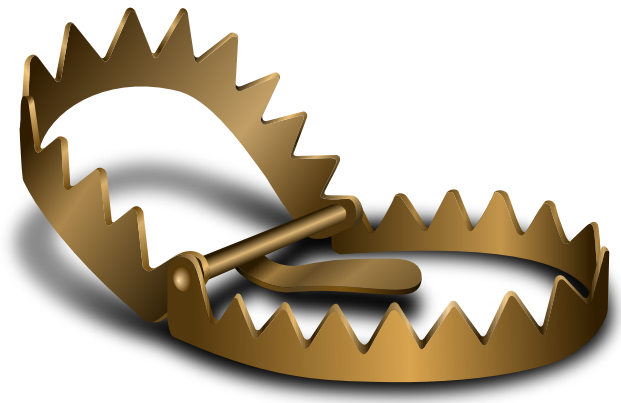


Breaking it Down



Composing Grammars

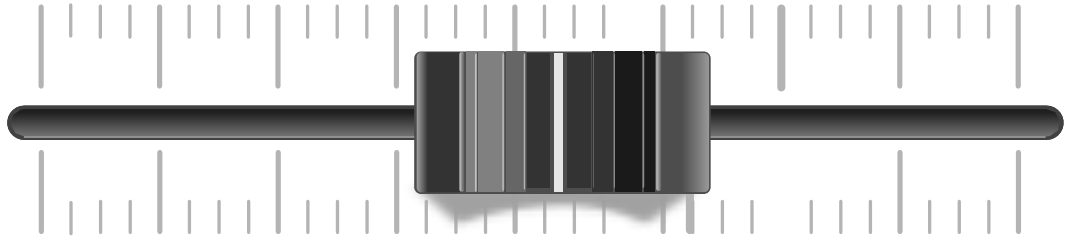
Shadowing Ambiguity Undefined



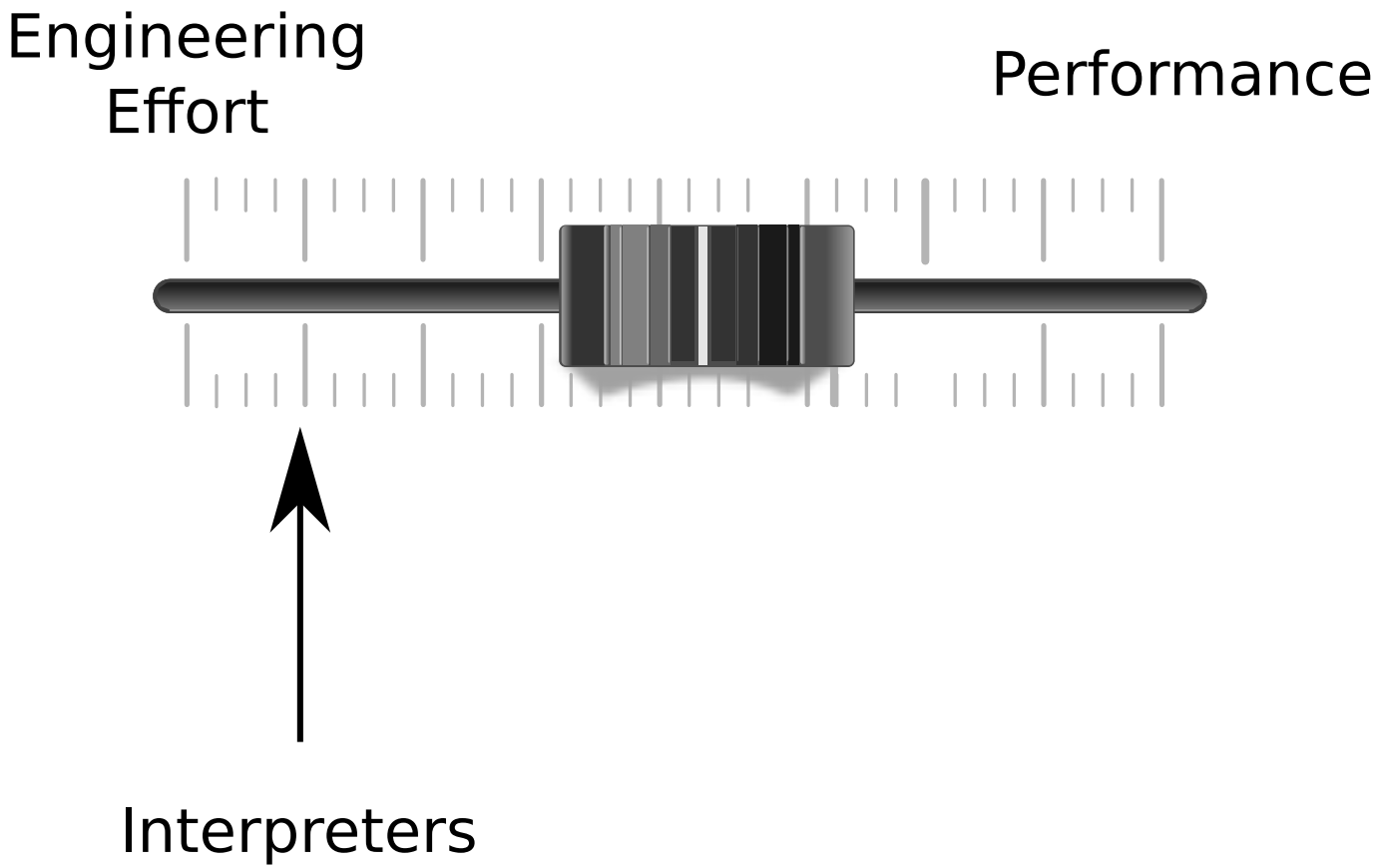
Runtime composition

Engineering Effort

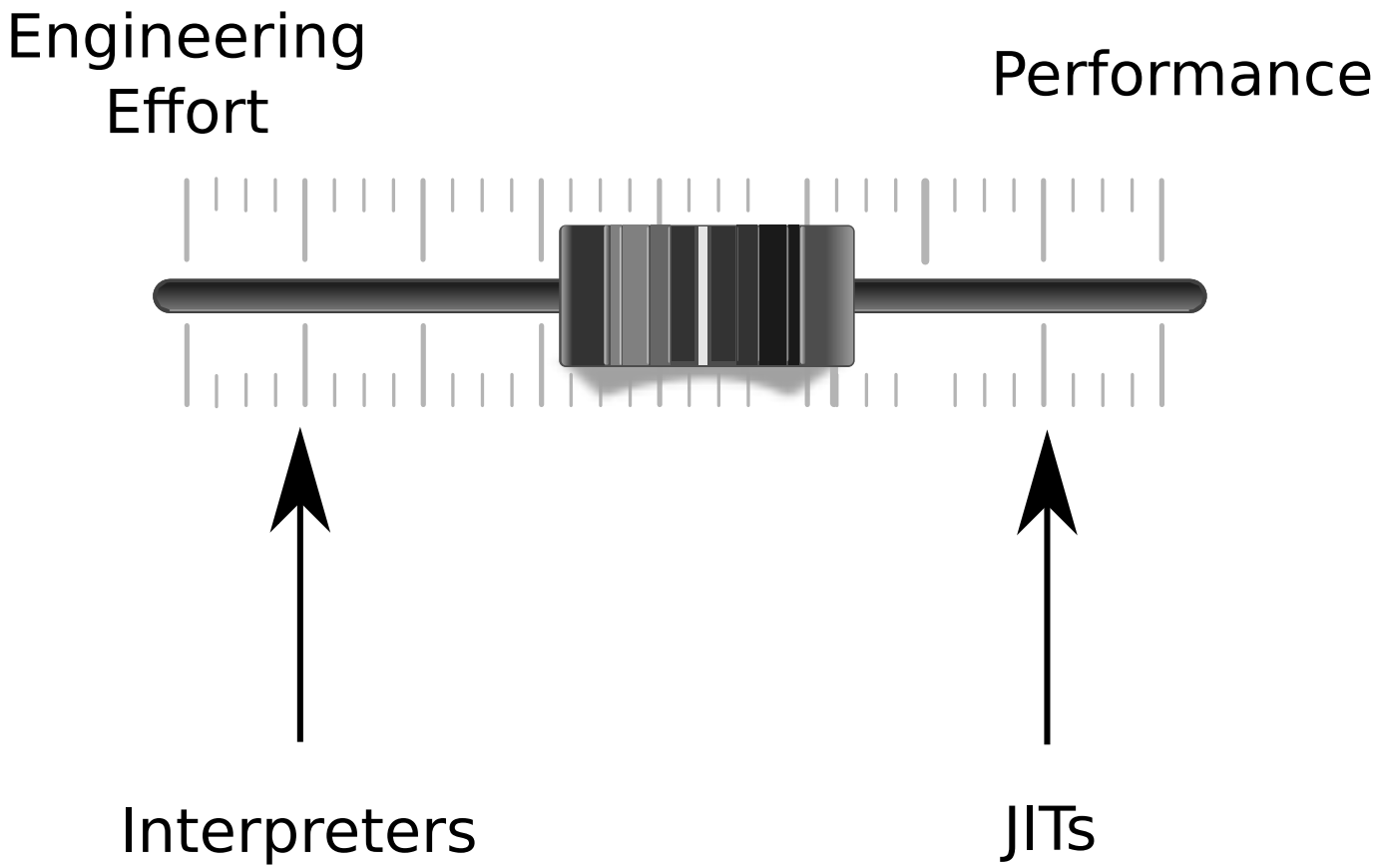
Performance



Runtime composition



Runtime composition



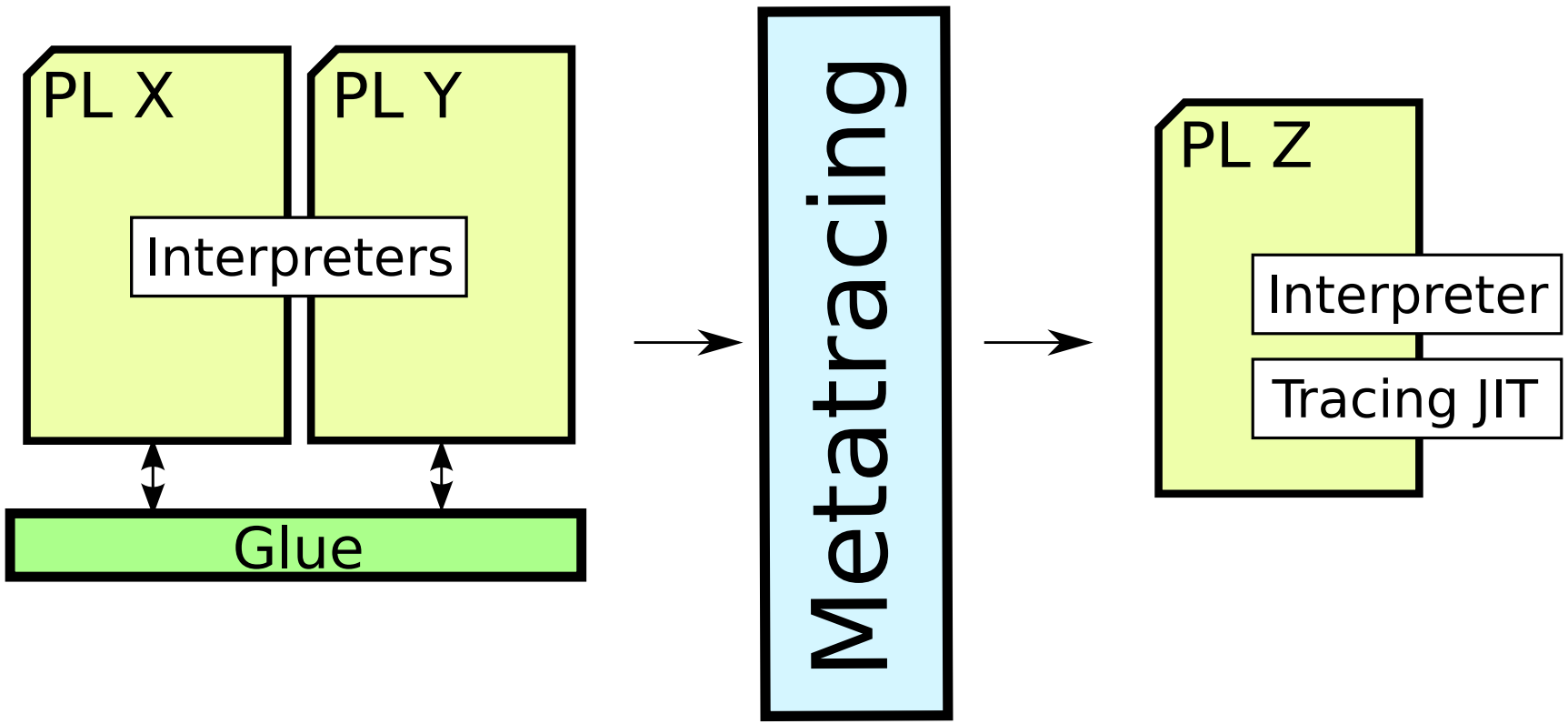
Our Approach

Language Boxes + Meta-tracing

Language Boxes to Compose Syntax

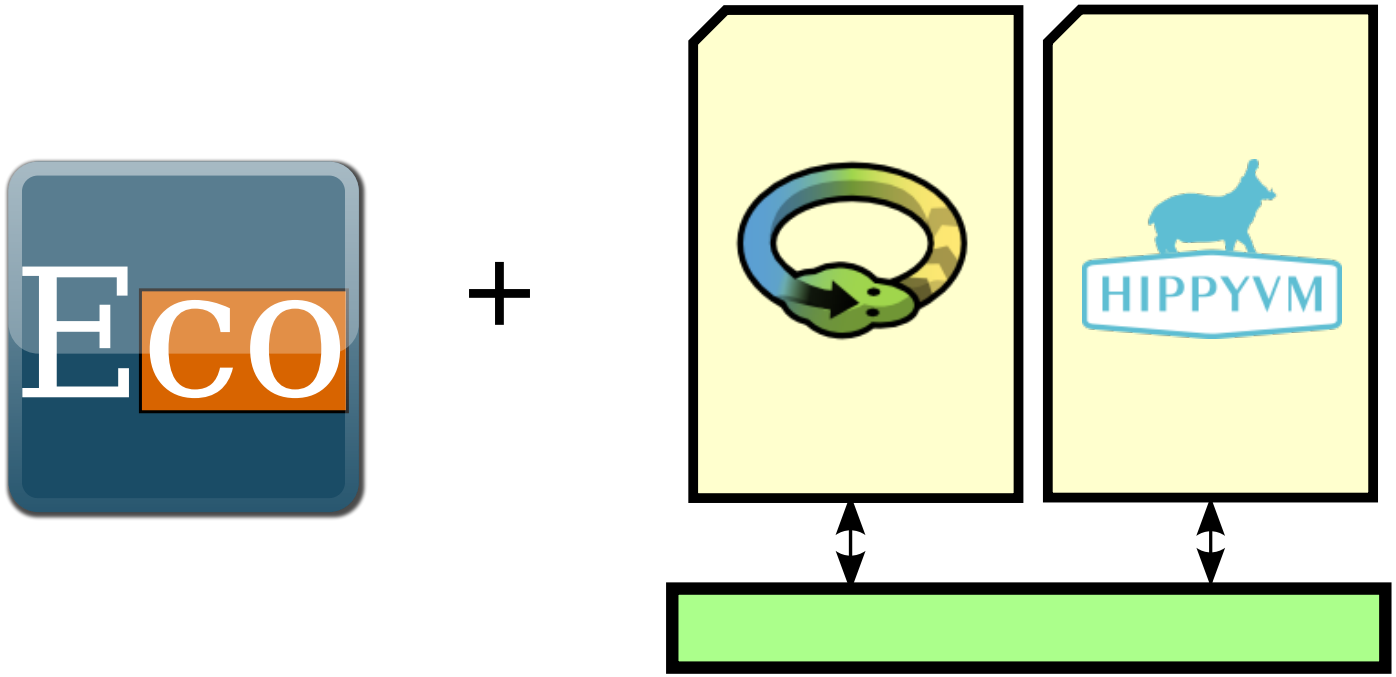
- The best bits from Syntax Directed Editing (SDE)
- Palatable editing experience

Meta-tracing to Compose Runtimes



PyHyp

PHP + Python



Features of PyHyp

- Calling Python functions and methods from PHP
- Calling PHP functions and methods from Python
- Transparent type conversions
- Arbitrary nesting of foreign functions
- Python expressions in PHP
- “Embedding” Python methods inside PHP classes
- Adds support for references to Python
- Cross-language scoping
- Cross-language exceptions

PyHyp Demo

PyHyp Performance

Benchmark	HippyVM	$\text{PyHyp}_{\text{PHP}}$	PyHyp_{Py}	PyPy
instchain	0.912 ±0.0011	1.000		0.675 ±0.0007
l1a0r	1.368 ±0.0004	1.000	1.360 ±0.0003	1.340 ±0.0106
l1a1r	1.306 ±0.0017	1.000	1.303 ±0.0016	1.140 ±0.0022
...				
total_list	0.864 ±0.0002	1.000	1.508 ±0.0004	0.587 ±0.0003
walk_list	0.779 ±0.0011	1.000	1.601 ±0.0026	1.080 ±0.0015
deltablue	4.325 ±0.0212	1.000		0.457 ±0.0026
fannkuch	1.848 ±0.0007	1.000	1.891 ±0.0005	1.005 ±0.0004
mandel	0.921 ±0.0005	1.000	0.999 ±0.0003	
richards	0.853 ±0.0010	1.000		0.488 ±0.0005
Geometric Mean	1.222 ±0.0006	1.000	0.963 ±0.0003	0.813 ±0.0007

Worst case: 2.6x overhead

Qualitative Results

Implementing desired behaviour: relatively easy

Deciding the correct behaviours: hard

“Semantic friction”

Semantic Friction: Collection Types

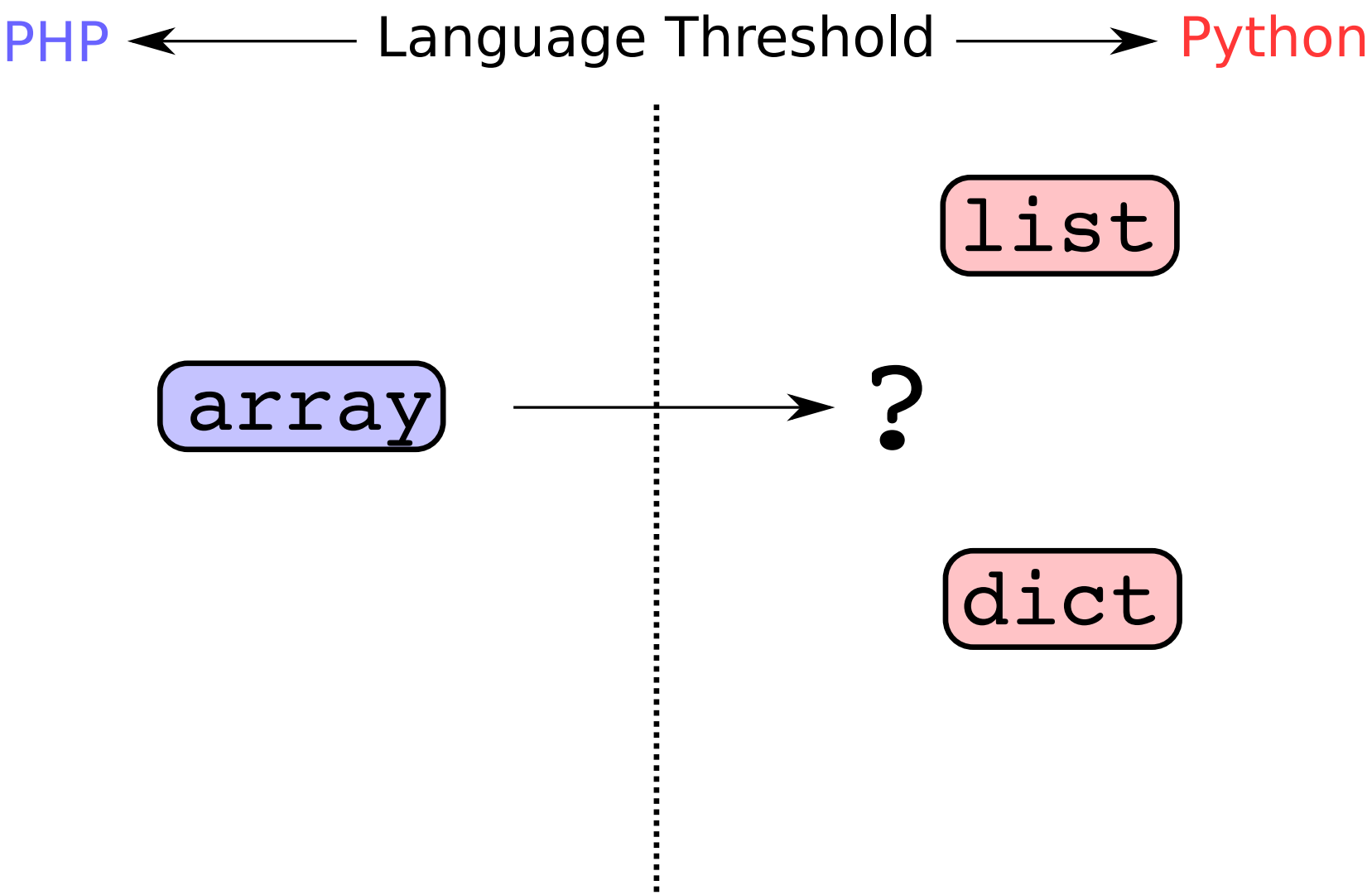
	PHP	Python
Sequence type	`array`	`list`
Mapping type	`array`	`dict`

Semantic Friction: Collection Types

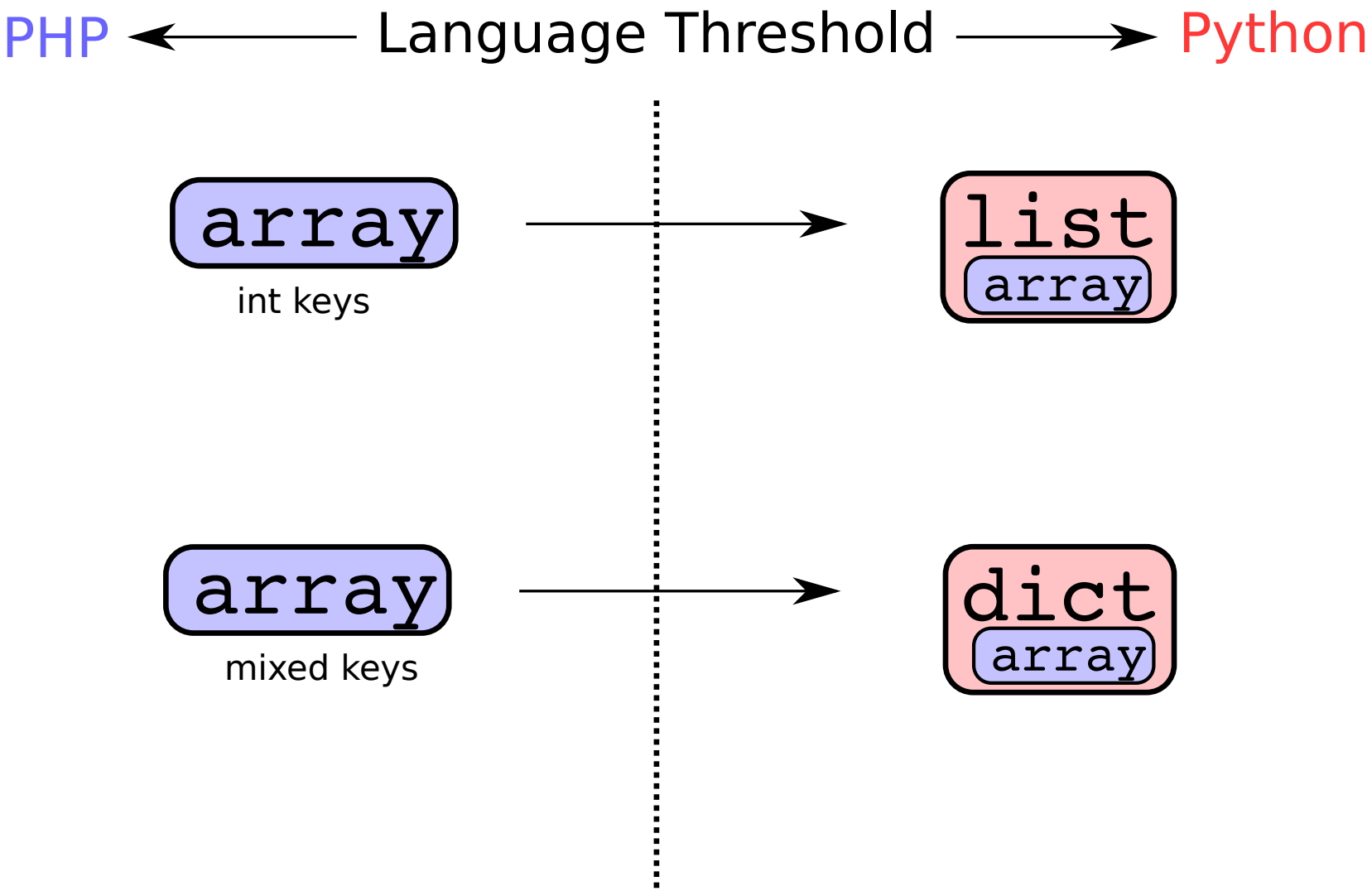
```
["a", "b", "c"]
```

```
Array ([0] => "a", [1] => "b", [2] => "c")
```

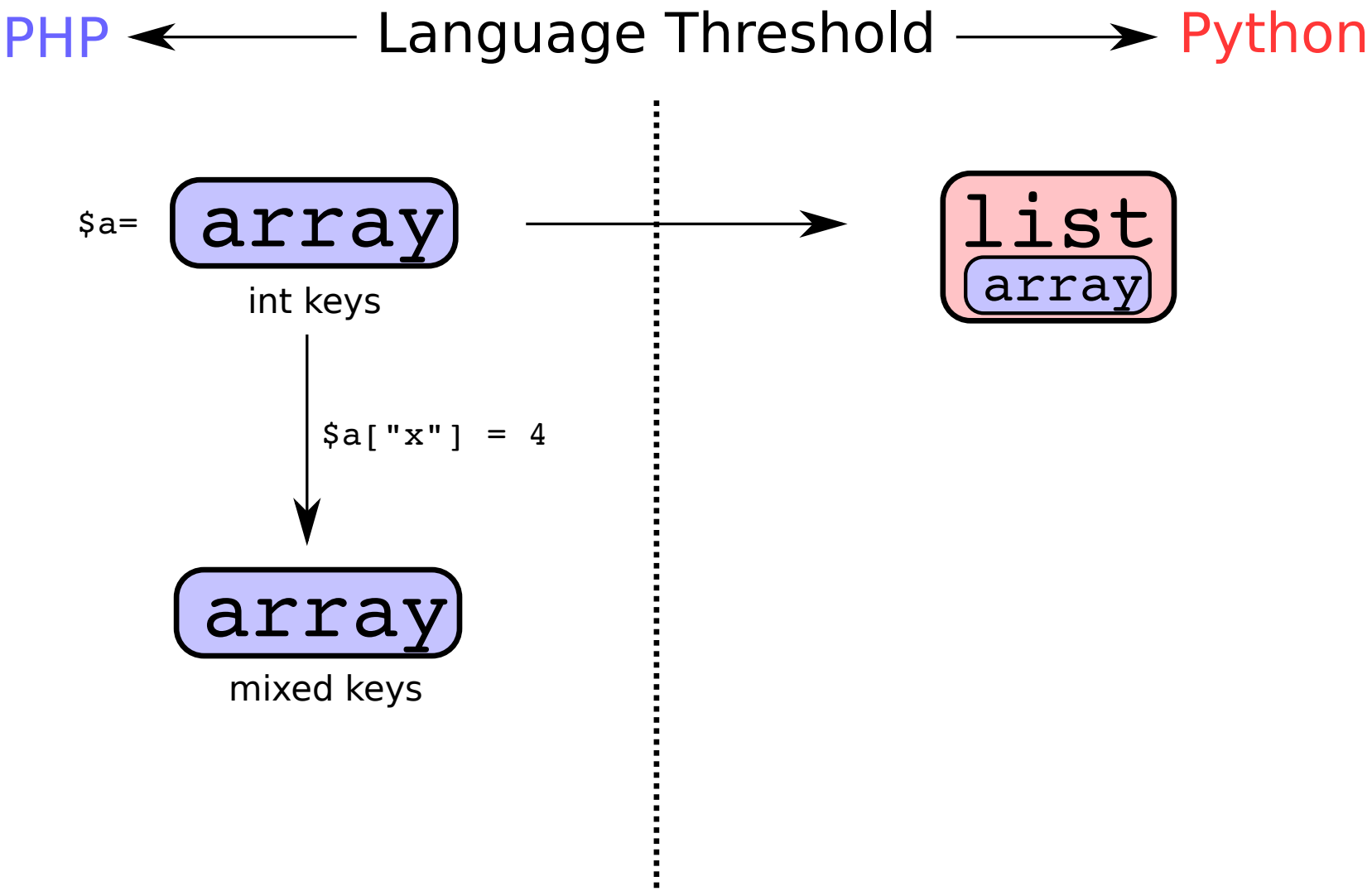
Semantic Friction: Collection Types



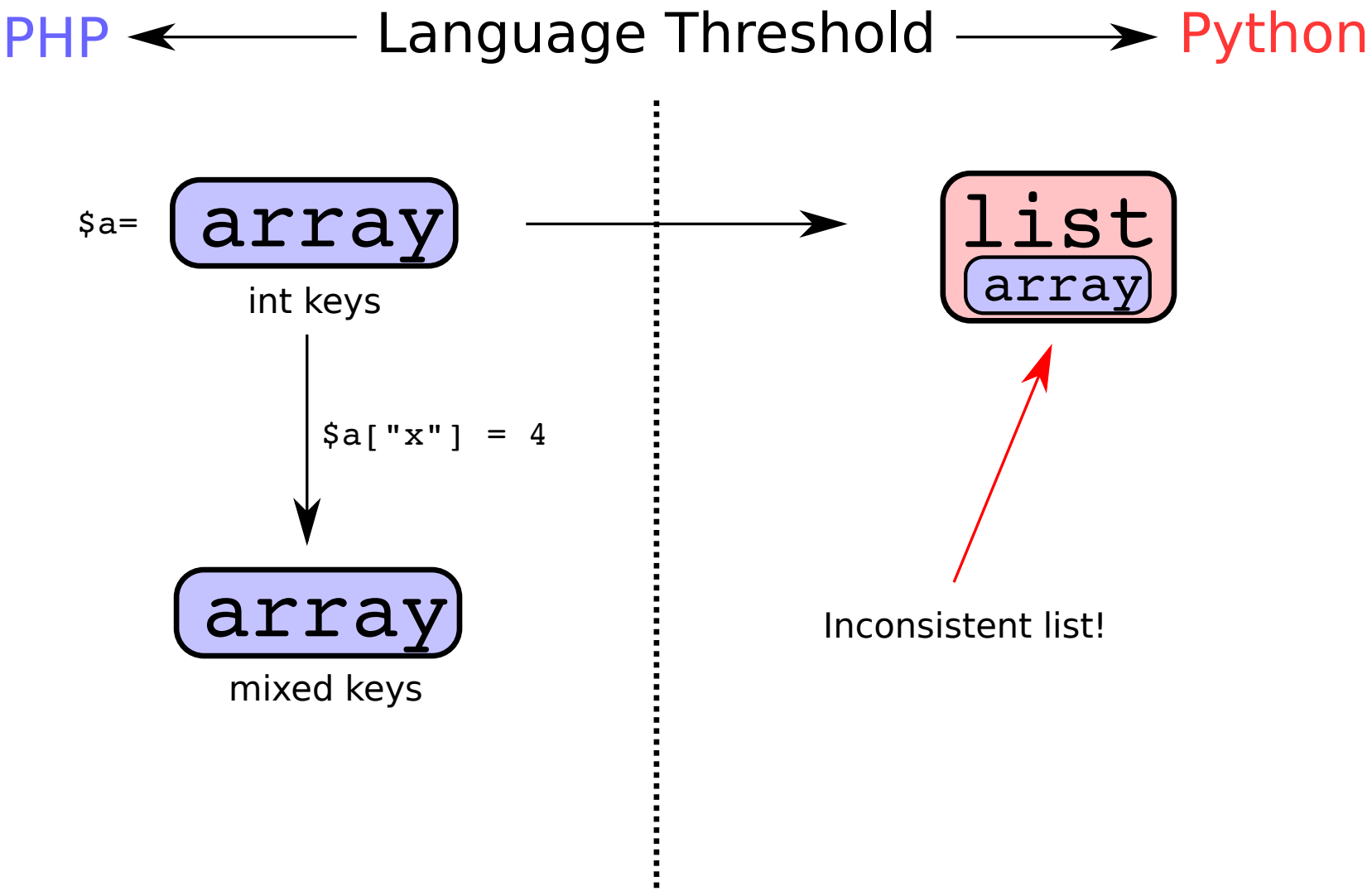
Semantic Friction: Collection Types



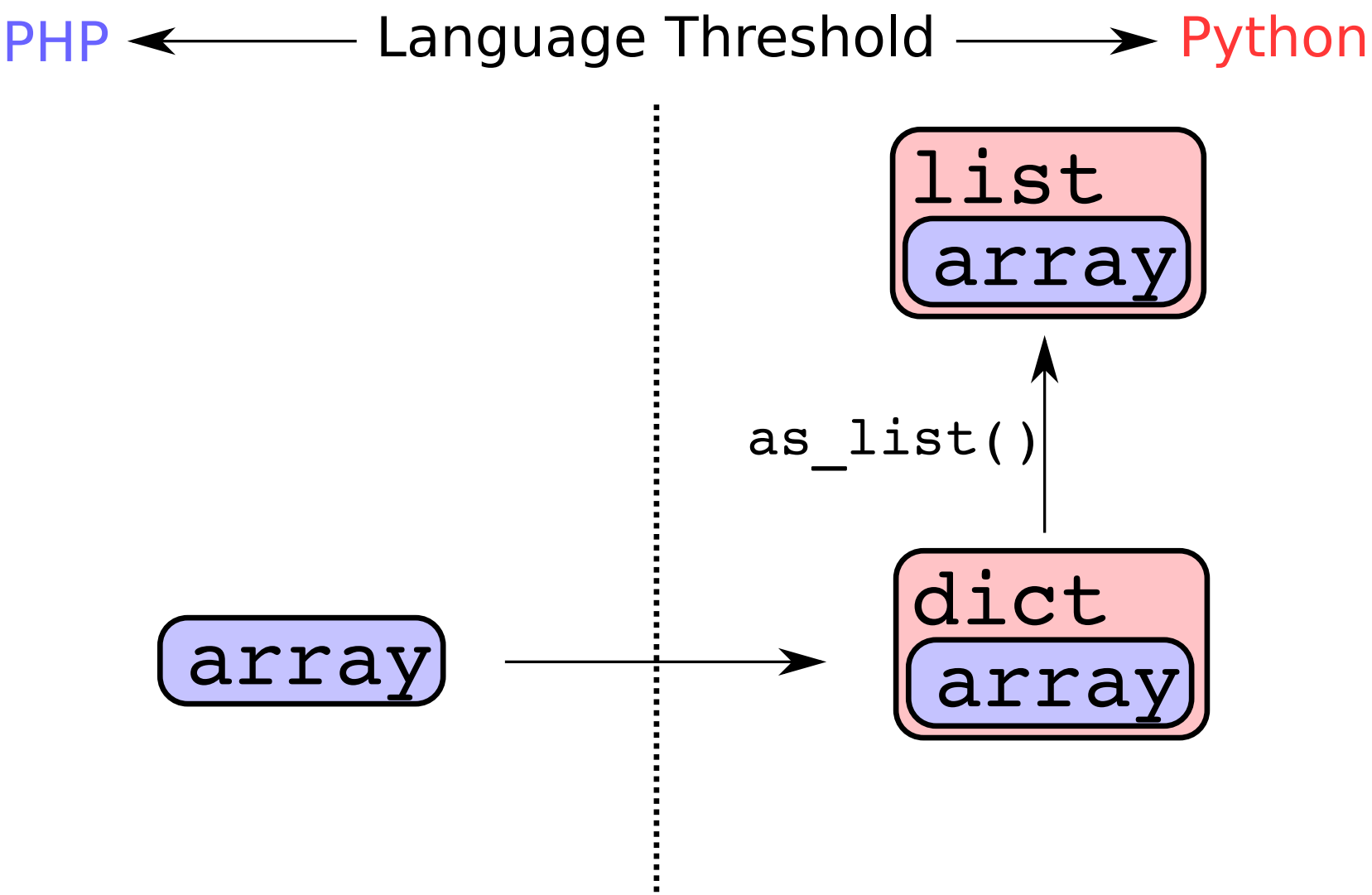
Semantic Friction: Collection Types



Semantic Friction: Collection Types



Semantic Friction: Collection Types



Conclusions

Language boxes and Meta-tracing:

- Fine-grained language composition
- Good editting experience
- Good Performance
- Relatively small engineering effort.

Qualitative outcomes:

- Implementing x-lang behaviours is easy.
- Designing x-lang behaviours is hard.

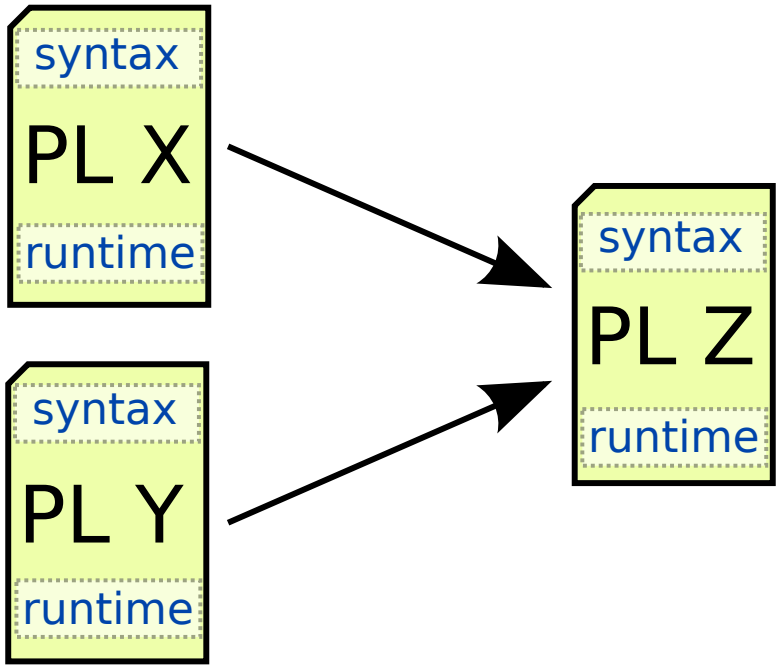
Future Work

- Tools for composed programs
  - Debugging
  - Profiling
  - Version control
  - ...
- Statically typed/functional languages.
- Compositions with >2 languages involved.

References

- *Parsing Composed Grammars with Language Boxes* Lukas Diekmann, Laurence Tratt.
- *Eco: A Language Composition Editor* Lukas Diekmann, Laurence Tratt.
- *Unipycation: A Case Study in Cross-language Tracing*, Edd Barrett, Carl Friedrich Bolz and Laurence Tratt
- *Approaches to Interpreter Composition*, Edd Barrett, Carl Friedrich Bolz and Laurence Tratt
- *Fine-grained Language Composition: A Case Study*, Edd Barrett, Carl Friedrich Bolz, Lukas Diekmann, Laurence Tratt
- *Making an Embedded DBMS JIT-friendly*, Carl Friedrich Bolz, Darya Kurilova, Laurence Tratt

Thanks



Language Boxes + Meta-tracing