



GIO: Generating Efficient Matrix and Frame Readers for Custom Data Formats by Example



Saeed Fathollahzadeh^{①, ②} Matthias Boehm^③

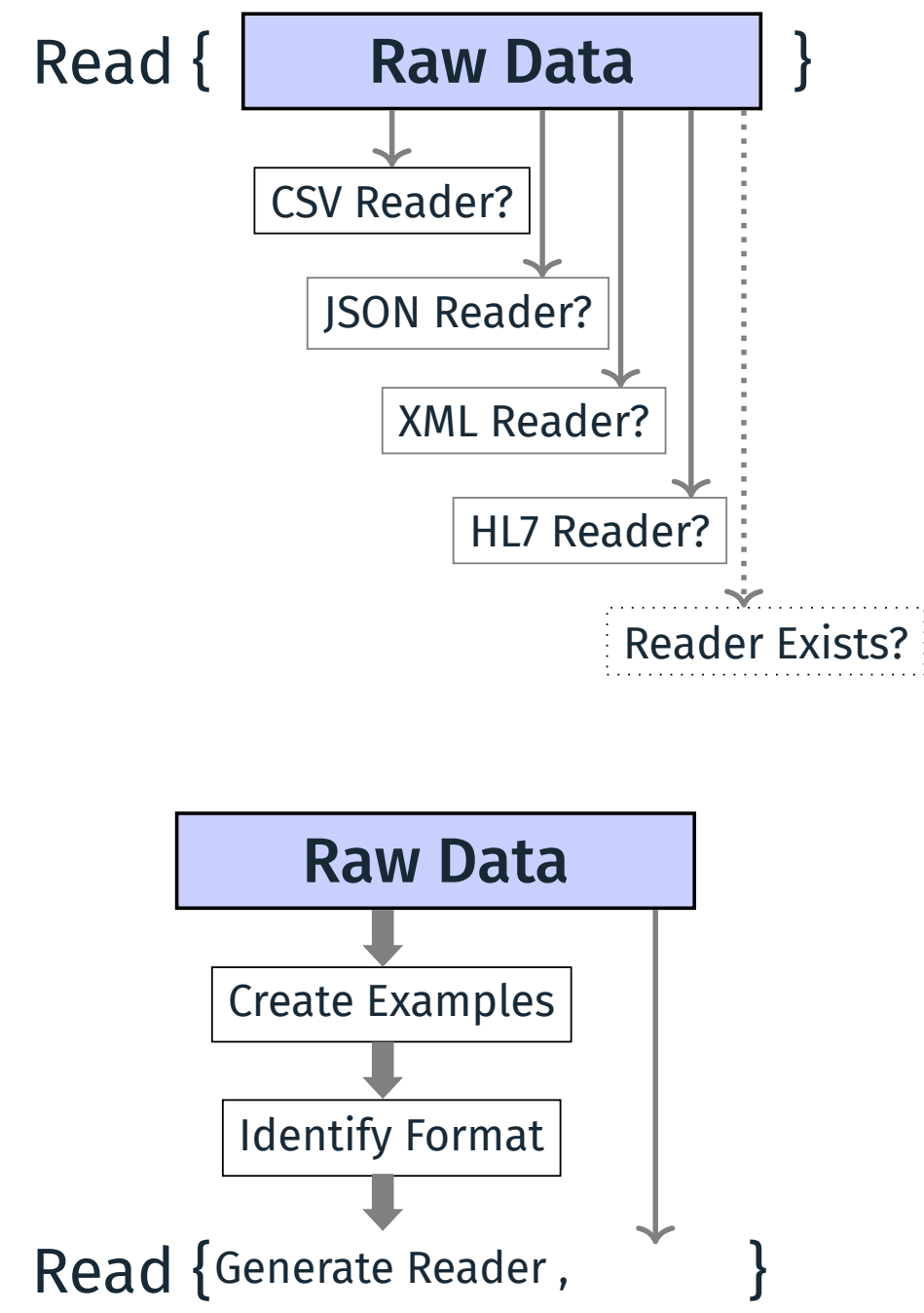
1. Motivation

- Data formats vary → **structure, syntax, semantics**, and **compression**
- Existing systems → **limited support** for custom formats
- Custom readers require → **low-level programming** and **system knowledge**
- Custom readers → **not portable** across systems and languages

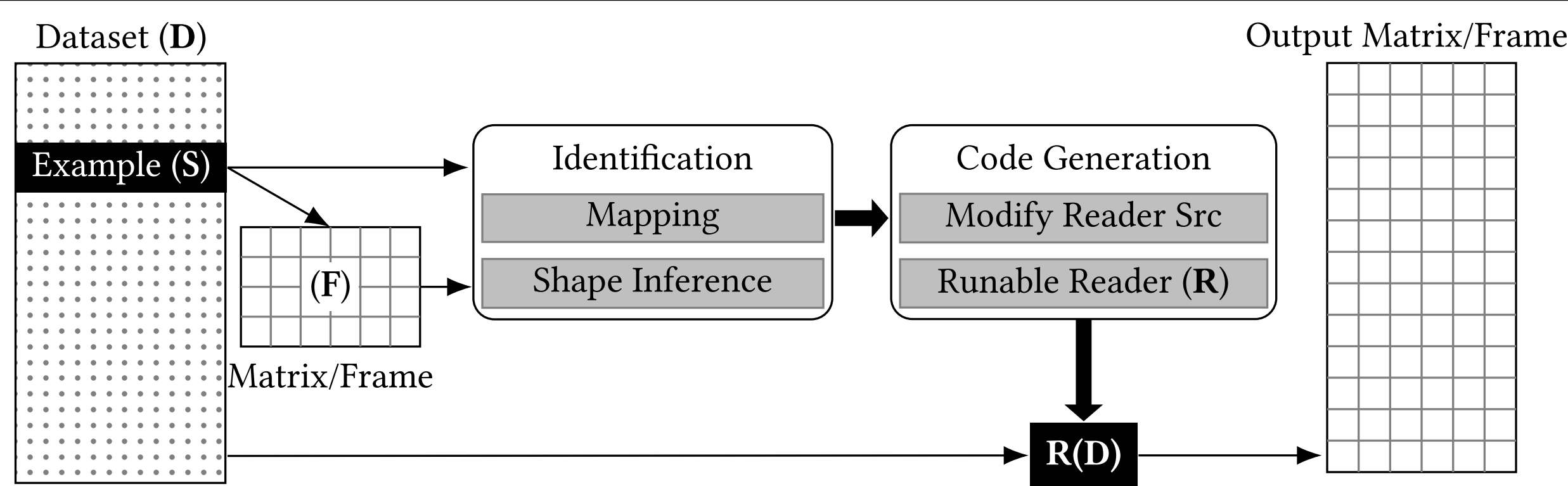
→ Is there an automatic way to get around?

Custom text-based dataset **D**, and user-provided examples:

- Sample Raw (S) Input** → a list of input strings (i.e., selected rows of the input dataset **D**).
- Sample Matrix/Frame (F) Input** → a sample matrix or frame, corresponding to **S** with n records.



2. GIO Overview



3. Example Parameters

```
#index 2015101 ← beginning of record #1
## NoDB: efficient query execution on raw data files
#@ Ioannis Alagiannis; Renata Borovica; Anastasia Ailamaki
#o EPFL Switzerland; EPFL Switzerland; EPFL Switzerland
#t 2015
#c VLDB Conference
#% ... unlimited set of references
#! As data collections become larger and larger, data loading evolves to a major bottleneck. ...

#index 2019102 ← beginning of record #2
## Pangea: Monolithic Distributed Storage for Data Analytics
#@ Jia Zou; Arun Iyengar; Chris Jermaine
#o Rice University; Rice University; Rice University
#t 2019
#c VLDB Conference
#% ... unlimited set of references
#! Storage and memory systems for modern data analytics are heavily layered, managing shared ...

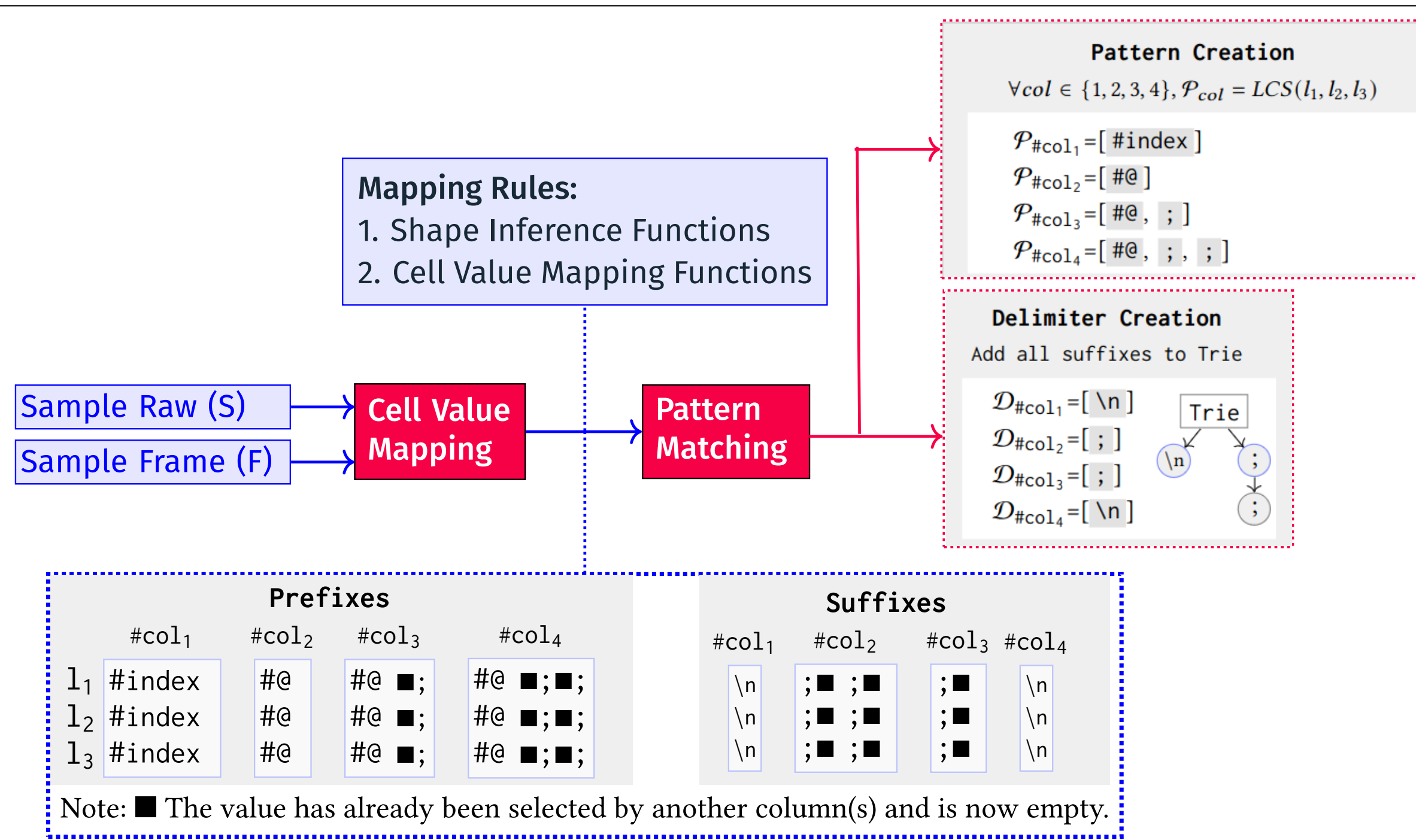
#index 2018103 ← beginning of record #3
## Filter Before You Parse: Faster Analytics on Raw Data
#@ Shoumik Palkar; Firas Abuzaied; Matei Zaharia
#o Stanford InfoLab; Stanford InfoLab; Databricks Inc
#t 2018
#c VLDB Endowment
#% ... unlimited set of references
#! Exploratory big data applications often run on raw unstructured or semi-structured data ...
```

Sample Raw (S)

#col ₁	#col ₂	#col ₃	#col ₄
2015101	Ioannis Alagiannis	Renata Borovica	Anastasia Ailamaki
2019102	Jia Zou	Arun Iyengar	Chris Jermaine
2018103	Shoumik Palkar	Firas Abuzaied	Matei Zaharia

Sample Raw (F)

4. Mapping Identification

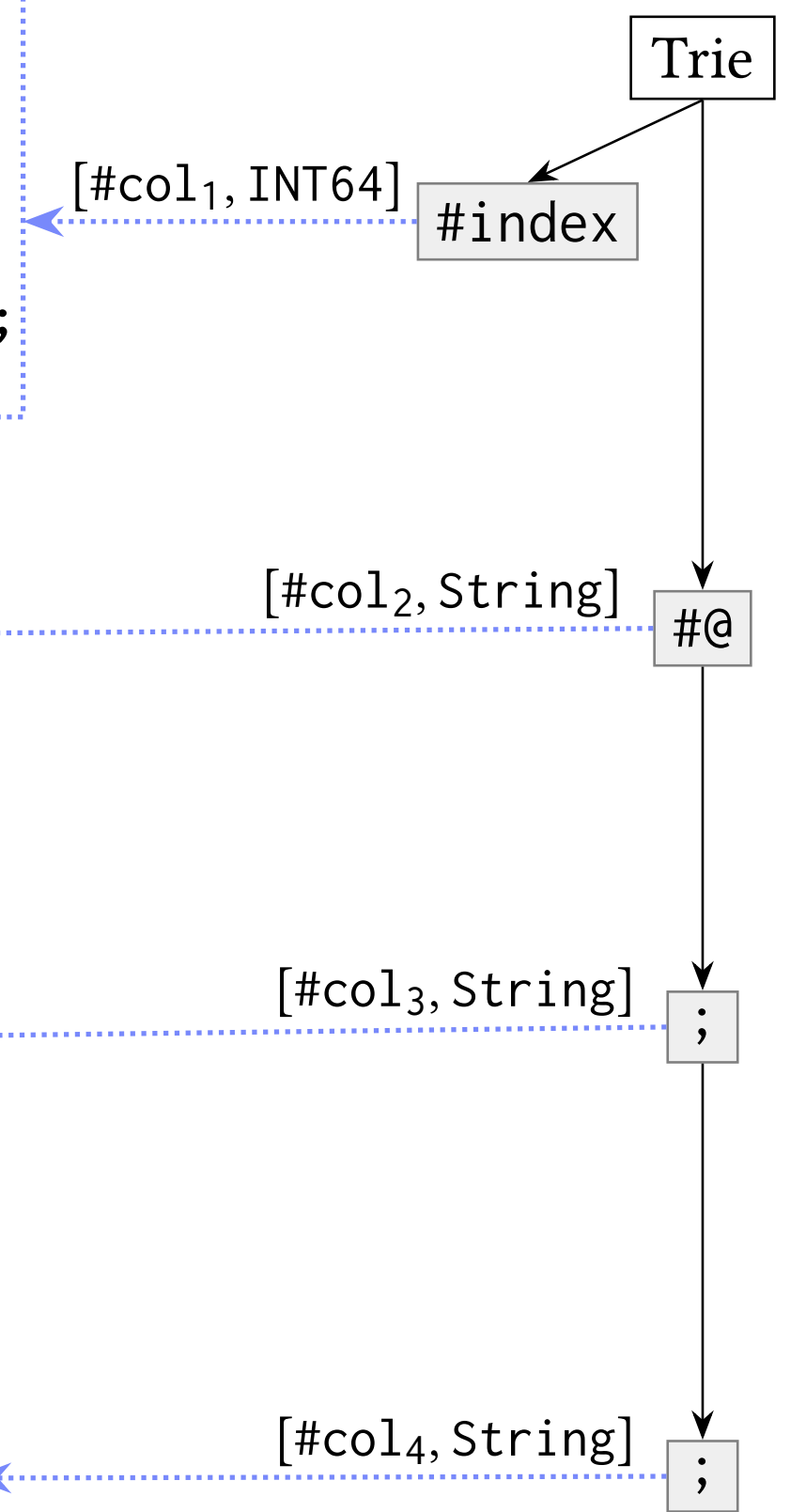


5. Code Generation

- Composed according to the passed mapping rules
- Template-based** Code Generation:
 - pre-pass for obtaining additional **metadata** from data
 - inferring the dimensions
 - iterate over records of the raw dataset
- Indexing Code:**
 - Row Indexing Code → determine the number of rows
 - Column Index and Value Code → code gets column info and values
- Cell Value Code:**
 - Cell Value by Nested Conditions
 - Cell Value by Sequential String Matching
 - Cell Value by Regular Expressions

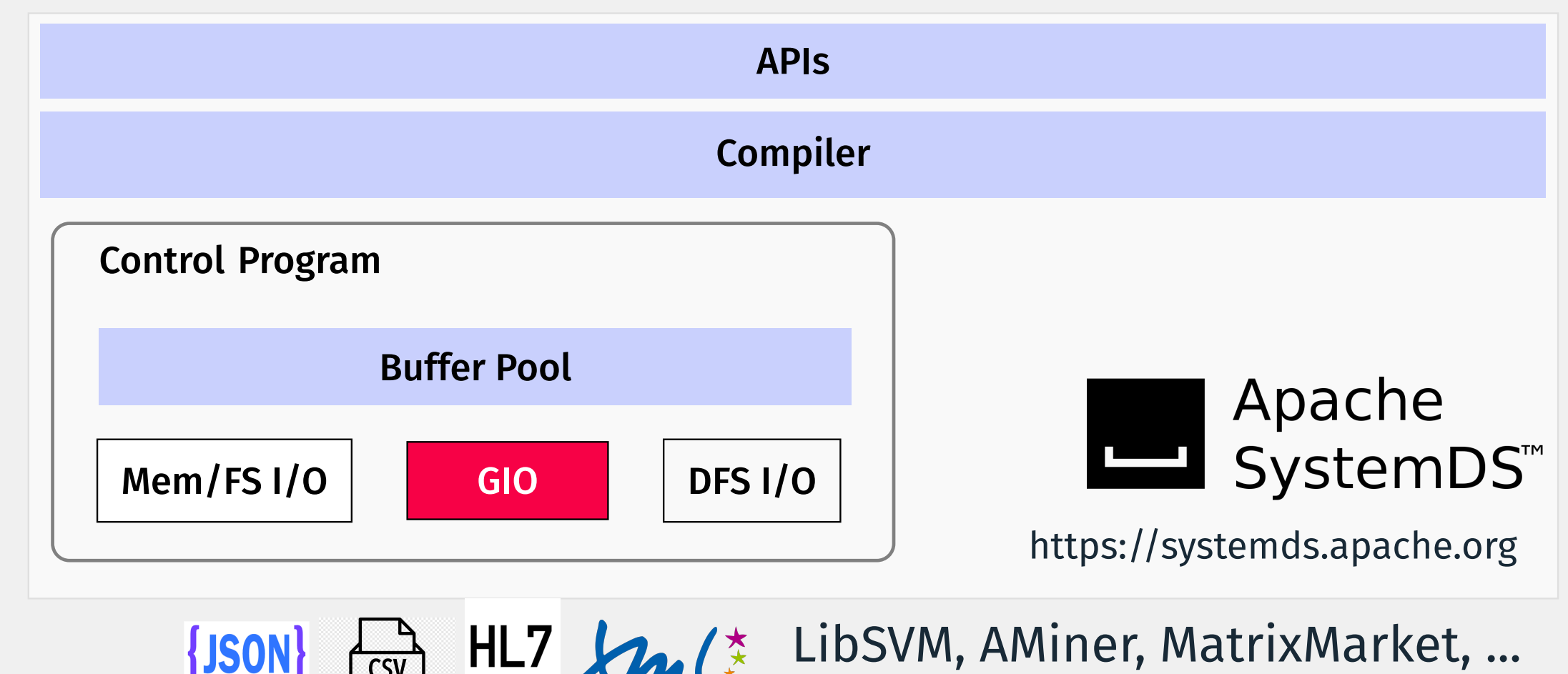
6. Reader Generation Example by Nested Conditions

```
1: _index = r.indexOf("#index", 0);
2: if(_index != -1) {
3:   _index += 6;
4:   _end = r.length();
5:   _text = r.substring(_index, _end);
6:   fb.set(rowIndex, 1, Parse(_text, INT64));
7: }
8: _index = r.indexOf("#@", 0);
9: if(_index != -1) {
10:  _index += 2;
11:  _end = r.indexOf(";", _index);
12:  _text = r.substring(_index, _end);
13:  fb.set(rowIndex, 2, _text);
14:  _index = r.indexOf(";", _end);
15:  if(_index != -1) {
16:    _index += 1;
17:    _end = r.indexOf(";", _index);
18:    _text = r.substring(_index, _end);
19:    fb.set(rowIndex, 3, _text);
20:    _index = r.indexOf(";", _index);
21:    if(_index != -1) {
22:      _index += 1;
23:      _end = r.length();
24:      _text = r.substring(_index, _end);
25:      fb.set(rowIndex, 3, _text);
26:    } } }
```

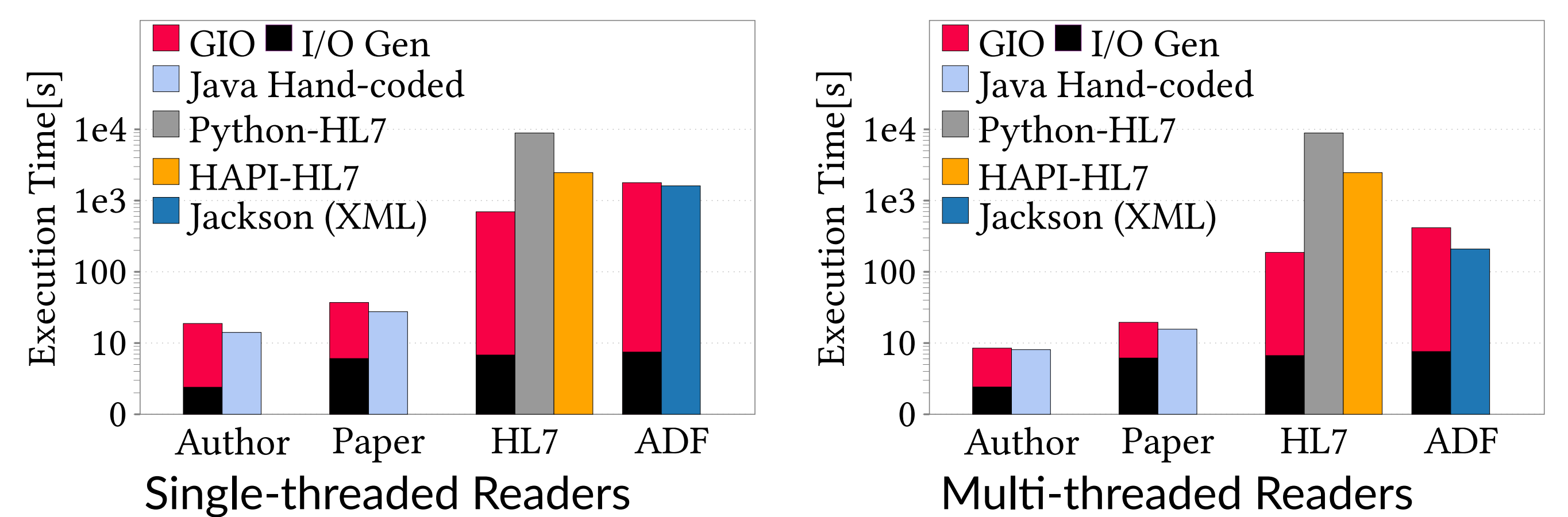


7. The user API of GIO

- M** = **gio_identify(S,F)** ← identification,
- R** = **gio_codegen(M)** ← reader generation
- F2** = **R.read(D)** ← reader usage on full or different datasets



8. Reader Performance on Full Custom Datasets



9. Reader Runtime Comparison with Varying Number of Attributes

