Parsing, Representing and Transforming Units of Measure

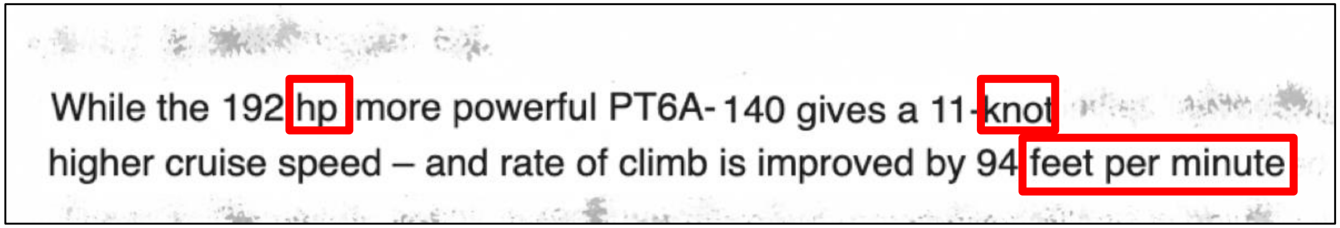
May 14th, 2019

Basel Shbita, Arunkumar Rajendran, Jay Pujara, and Craig A. Knoblock

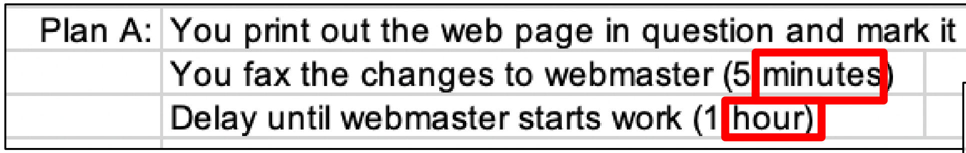
Research funded under the DARPA World Modelers program

Units of Measure

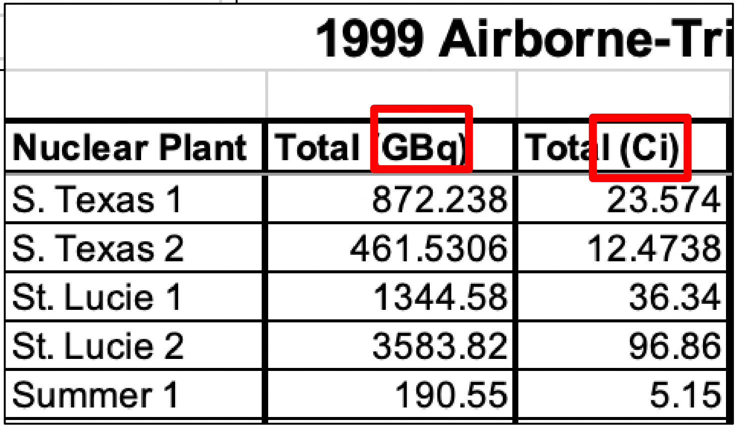
Appear in files within datasets in a textual form



While the 192 hp more powerful PT6A-140 gives a 11-knot higher cruise speed – and rate of climb is improved by 94 feet per minute



Plan A:	You print out the web page in question and mark it up (5 min.)
	You fax the changes to webmaster (5 minutes)
	Delay until webmaster starts work (1 hour)



1999 Airborne-Tri

Nuclear Plant	Total (GBq)	Total (Ci)
S. Texas 1	872.238	23.574
S. Texas 2	461.5306	12.4738
St. Lucie 1	1344.58	36.34
St. Lucie 2	3583.82	96.86
Summer 1	190.55	5.15

Does not carry any semantic or dimensional meaning

Not easily recognized

SI prefixes make it even harder

Ambiguous

Data Normalization

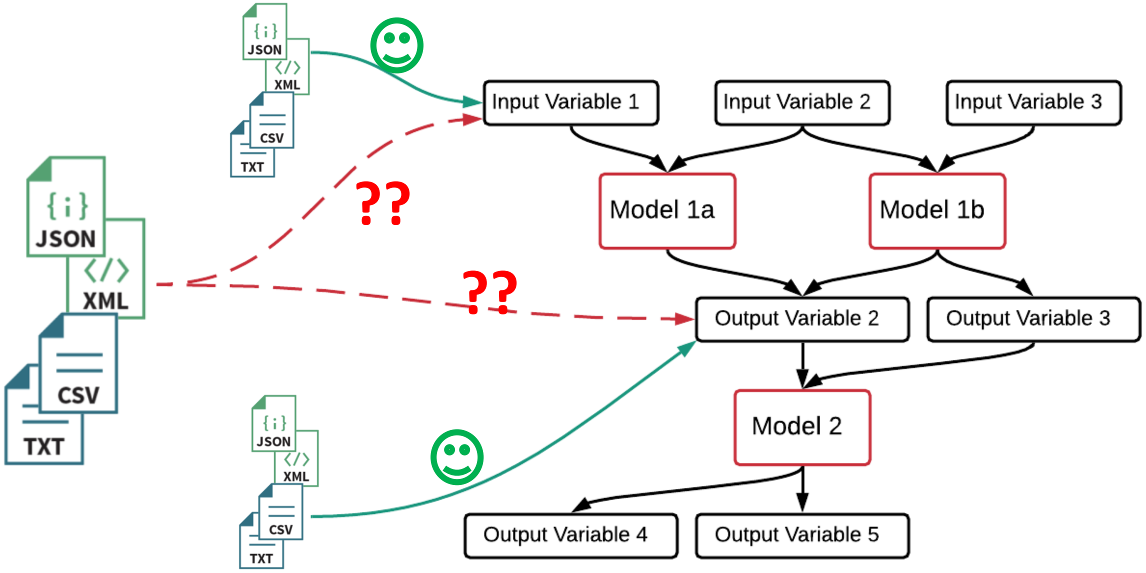
Data normalization is a **difficult** task!

Occupies as much as **80%** [Dasu and Johnson] of total data analysis time

To combine datasets, scientists must

select, **understand**, and **align** them **manually**

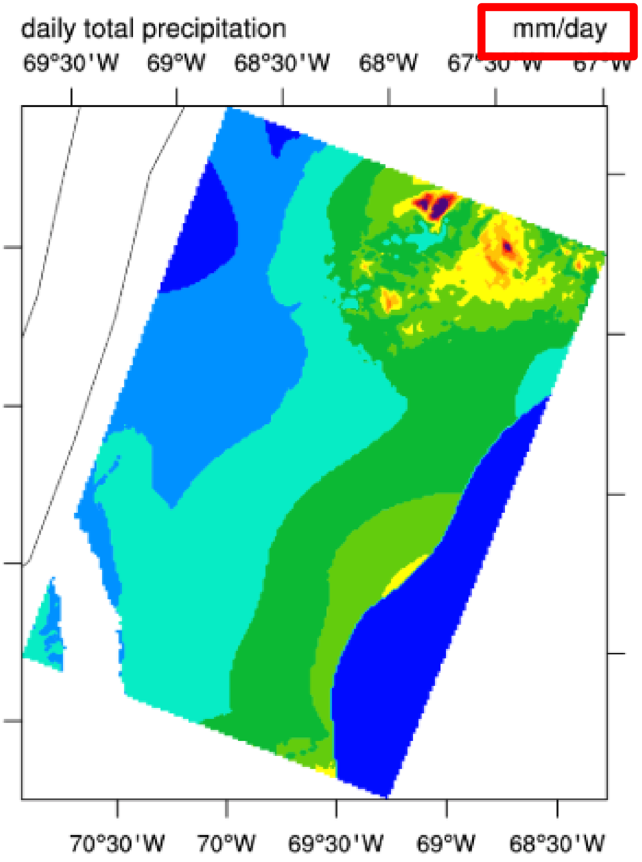
Requires understanding **different domains** and **formats**

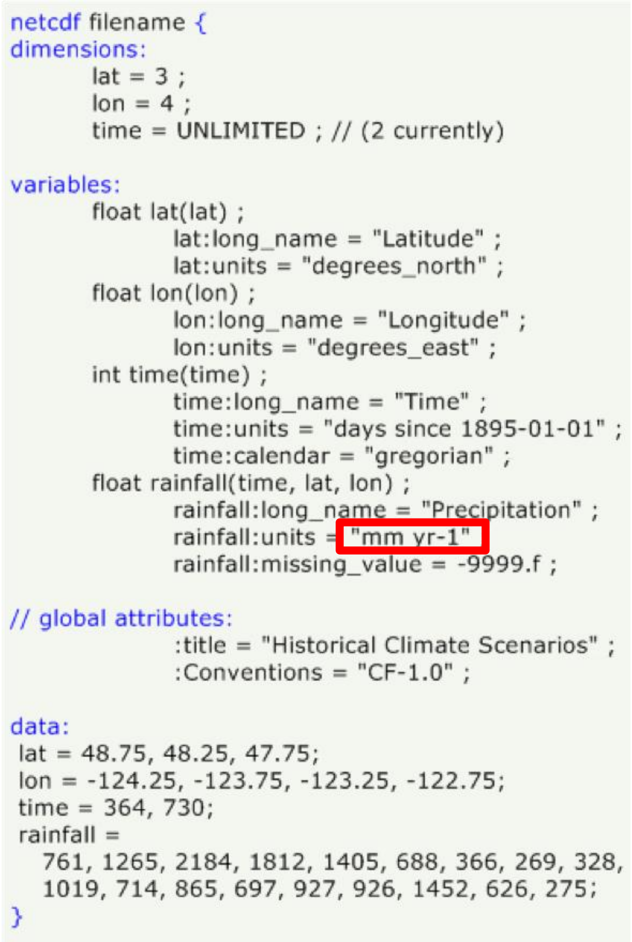


Data Normalization

Data normalization is a **difficult** task!

Occupies as much as **80%** [Dasu and Johnson] of total data analysis time





```
netcdf filename {
dimensions:
        lat = 3 ;
        lon = 4 ;
        time = UNLIMITED ; // (2 currently)

variables:
        float lat(lat) ;
                lat:long_name = "Latitude" ;
                lat:units = "degrees_north" ;
        float lon(lon) ;
                lon:long_name = "Longitude" ;
                lon:units = "degrees_east" ;
        int time(time) ;
                time:long_name = "Time" ;
                time:units = "days since 1895-01-01" ;
                time:calendar = "gregorian" ;
        float rainfall(time, lat, lon) ;
                rainfall:long_name = "Precipitation" ;
                rainfall:units = "mm yr-1" ;
                rainfall:missing_value = -9999.f ;

// global attributes:
                :title = "Historical Climate Scenarios" ;
                :Conventions = "CF-1.0" ;

data:
 lat = 48.75, 48.25, 47.75;
 lon = -124.25, -123.75, -123.25, -122.75;
 time = 364, 730;
 rainfall =
   761, 1265, 2184, 1812, 1405, 688, 366, 269, 328,
   1019, 714, 865, 697, 927, 926, 1452, 626, 275;
}
```

Our Task

Identify and provide a **semantic representation** for **units** of measure associated with **data**

Challenges:

- Textual Form
  - abbreviations, compound units, prefixes
- Reusable semantic format
- Automated process (i.e. transformation)

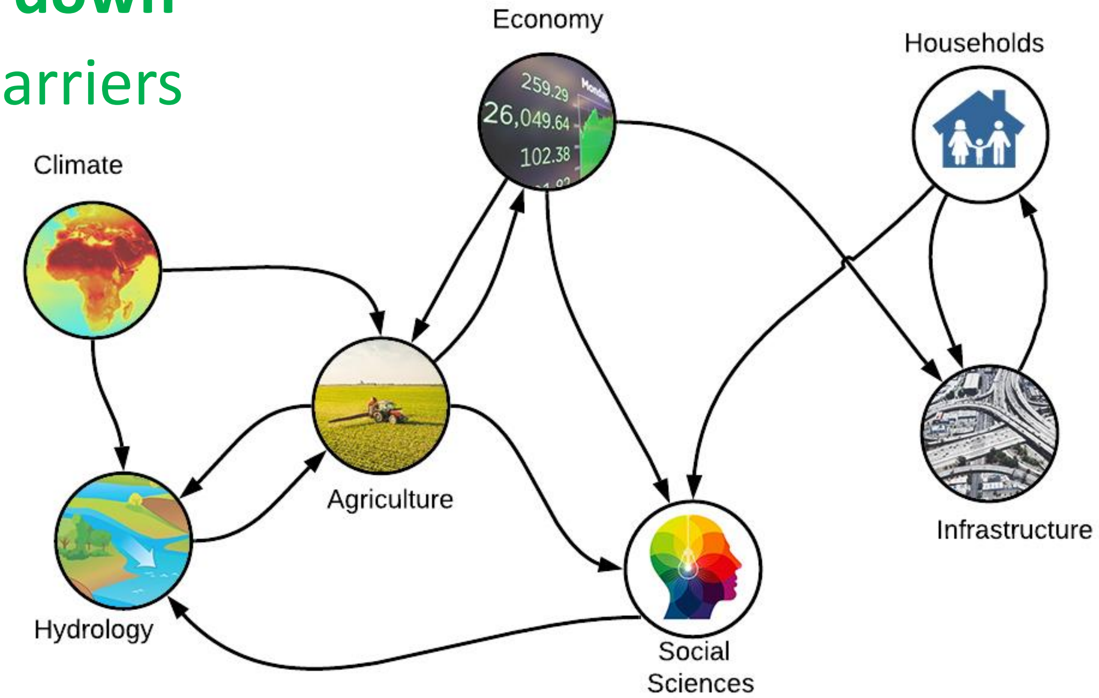
Need an **automated** pipeline from raw data to semantic representation which can be easily **interpreted** by humans & machines

Why?

Break down

data barriers

In-domain



Across-domains

Create ideal setting for **multidisciplinary** scientists

Existing Approaches

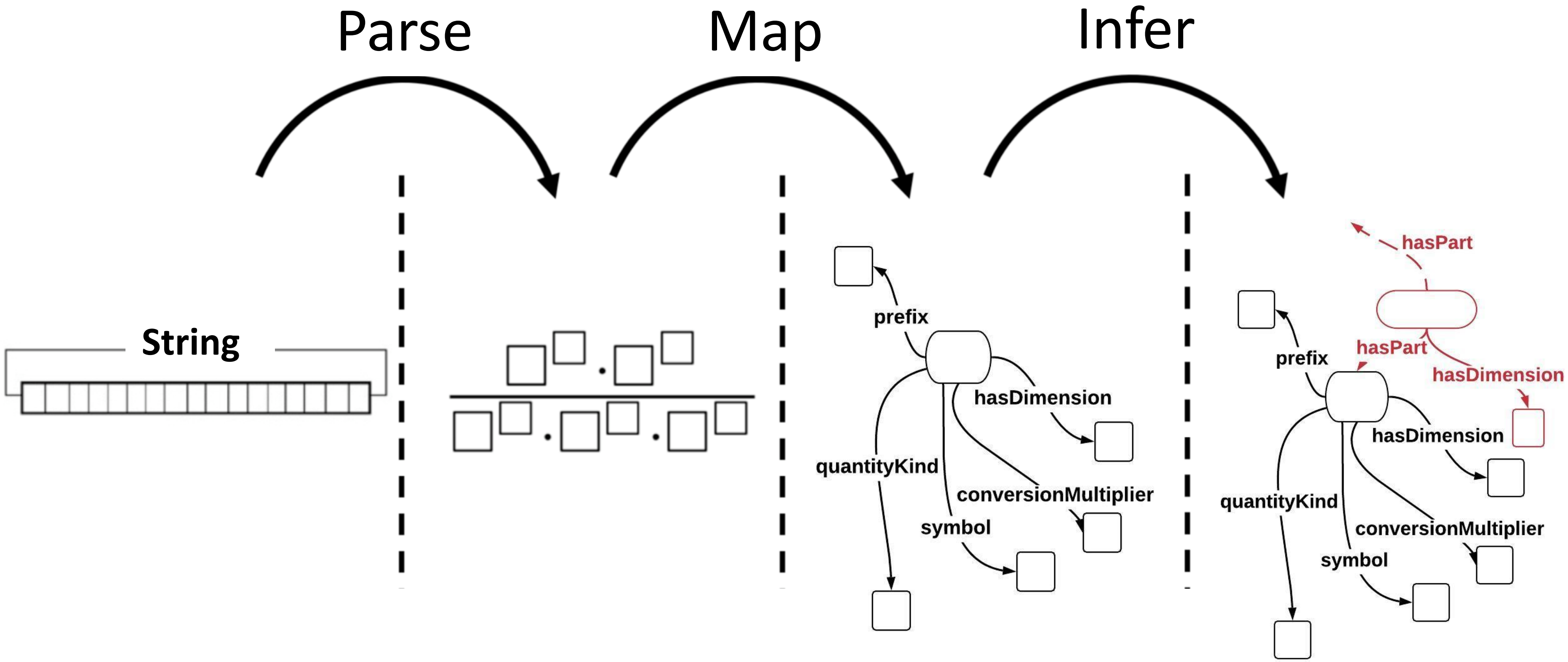
- Semi-automated or ad-hoc strategies
  - Harm the transparency and reproducibility of the results
  - Intractable and tedious
  - Susceptible to human error

- State of the Art:
  - **Measurement units in R** [Pebesma et al.] and **the *yt* project** [Turk et al.] allow automatic unit conversion
    - Requires user interaction
    - No automatic detection or semantics that can be interchanged
  - **quantulum** extracts units from unstructured text and associates it with a corresponding Wikipedia page
    - Requires a numeric value within the context of the textual form of the unit

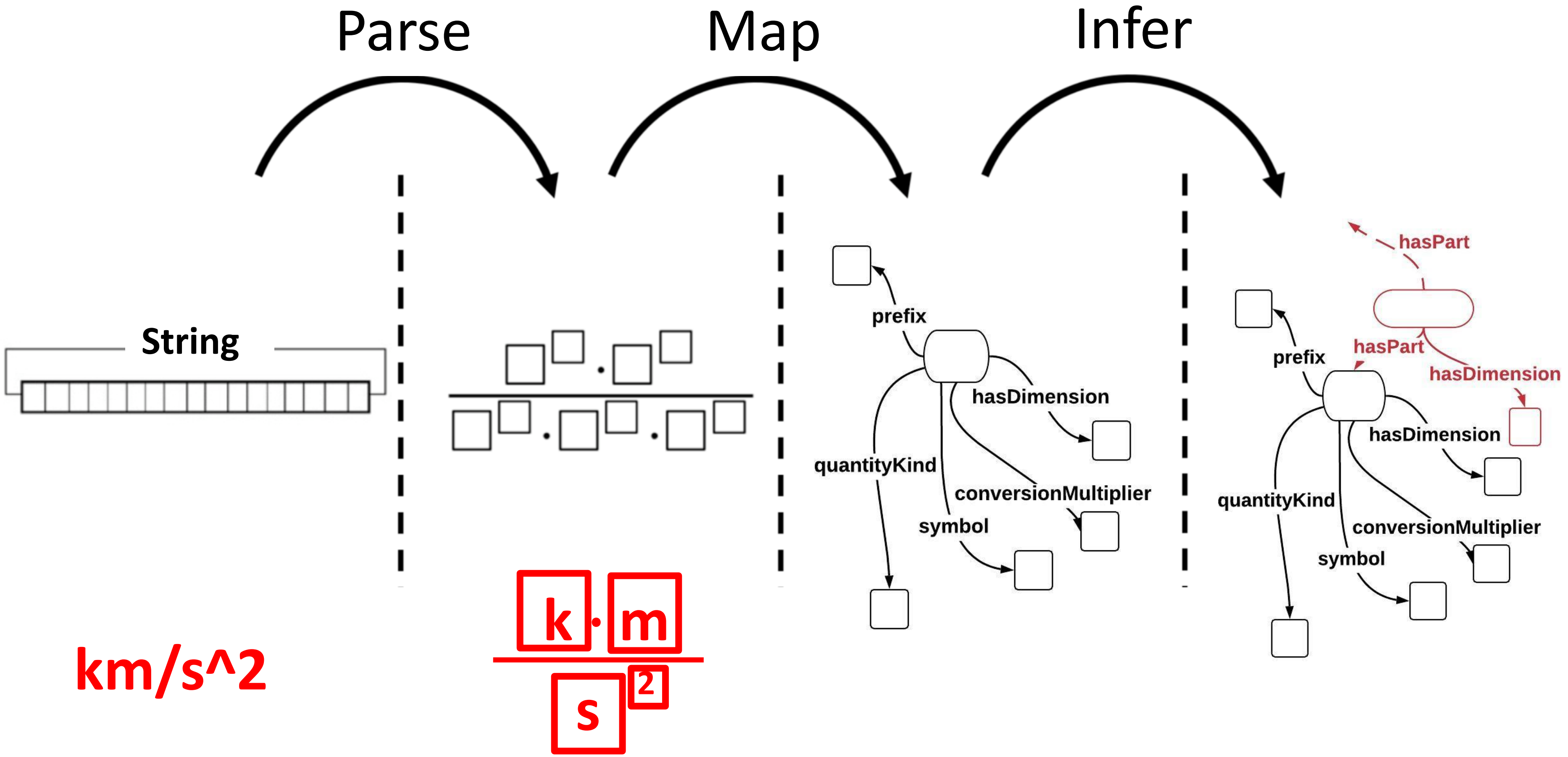
Our Approach: Motivation

- Published ontologies are a beneficial resource:
  - NASA published **QUDT**
    - Defines the base classes and attributes for modeling physical **quantities**, **units** of measure, and their **dimensions**
- Compound units (i.e. ’A/cm^2’) can be **decomposed** to **components** which include
  - Atomic units (i.e. ’cm’)
  - Composing elements (i.e. exponents, prefixes)
  - Relations between them
- Intuitive to integrate QUDT into a framework
  - Enable automatic data understanding, normalization and transformation

Our Approach



Our Approach



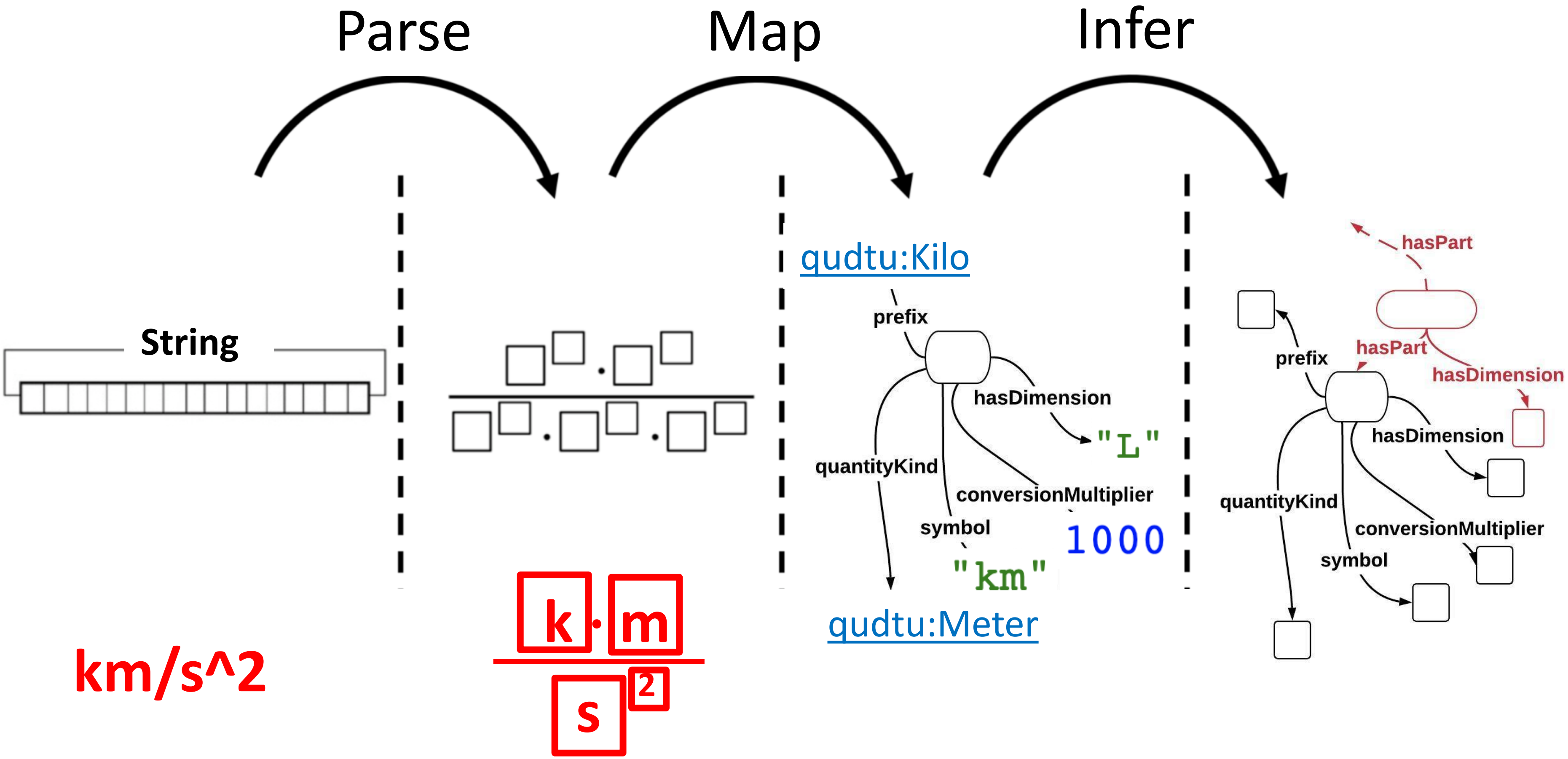
Parse

Map

Infer

km/s^2

Our Approach



Parse

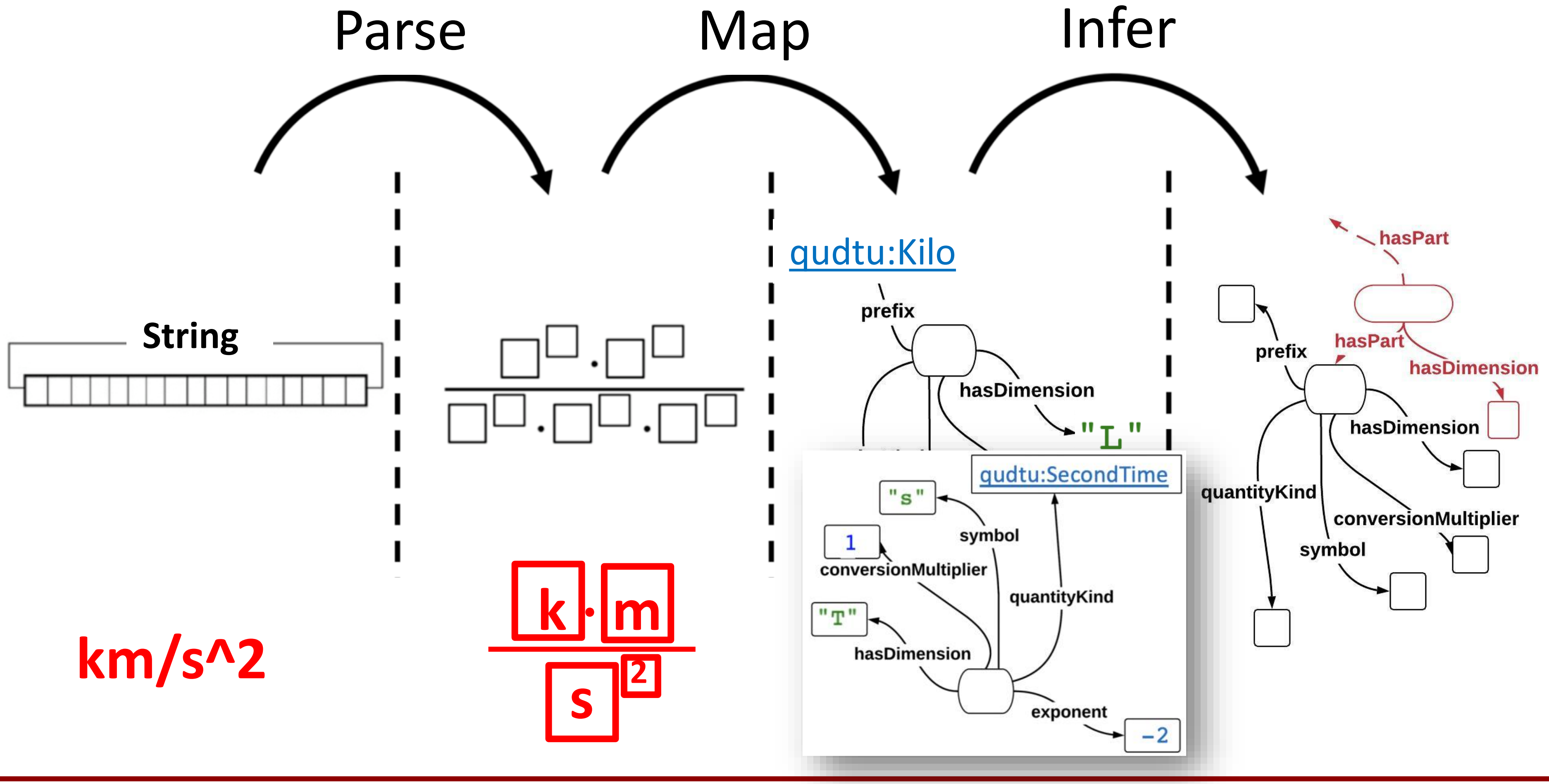
Map

Infer

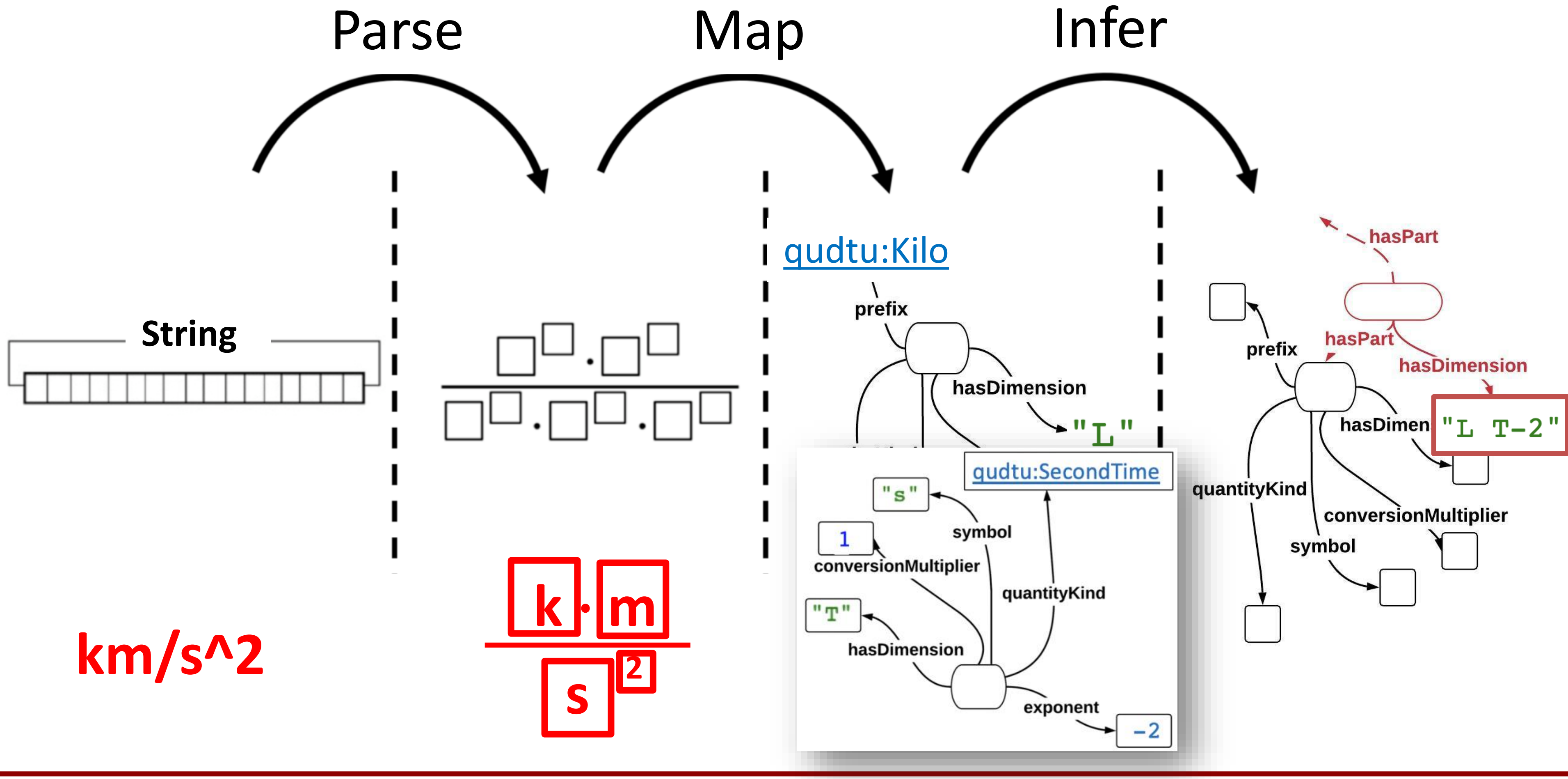
String

km/s^2

Our Approach

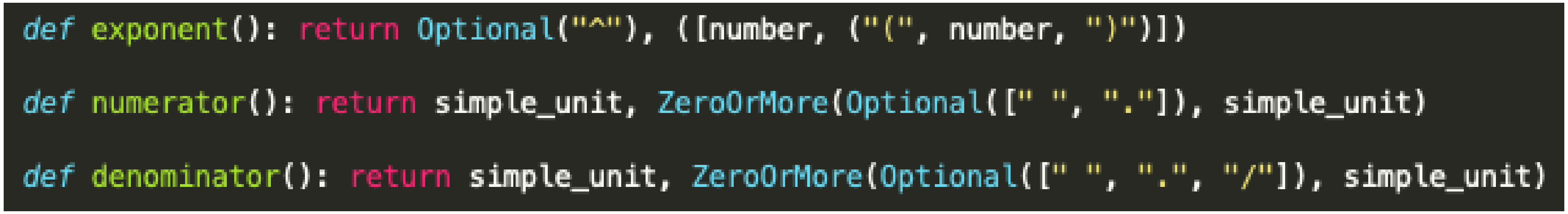


Our Approach



Parsing

- Goal: string → structured form with relations
  - Parse **nominators**, **denominators**, **exponents**, **multipliers** and **prefixes**
  - Re-decompose recursively
- We defined a **grammar** using Arpeggio
  - **Recursive** descent parser
  - Based on a Parsing Expression Grammar formalism



```
def exponent(): return Optional("^"), ([number, ("(", number, ")")])

def numerator(): return simple_unit, ZeroOrMore(Optional([" ", "."]), simple_unit)

def denominator(): return simple_unit, ZeroOrMore(Optional([" ", ".", "/"]), simple_unit)
```

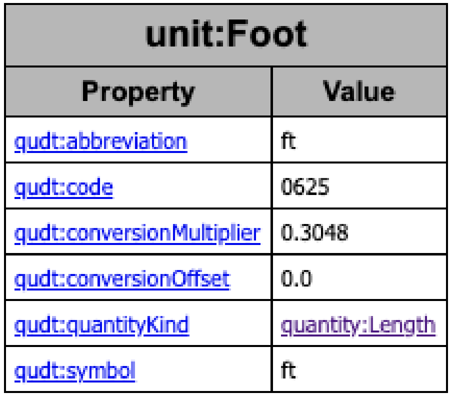
Parsing Challenges

- Incomplete definition in KB
  - Define closed set of base dimension classes which were derived from the original ontology
- Unit prefixes (micro = mu = µ)
  - Define closed set of SI prefixes with variants
- Compound units ambiguity ('min' = minute or milli-inch?)
  - Iterative joint matching algorithm for {prefix, unit} pairs
  - higher confidence to single atomic unit

Structured Unit Representation

- Goal: capture a semantic meaning
  - Map decomposed elements to QUDT
- Utilize additional grammar elements
- Normalize compound units
- Present cost-free interpretable representation with unique URIs for each individual element

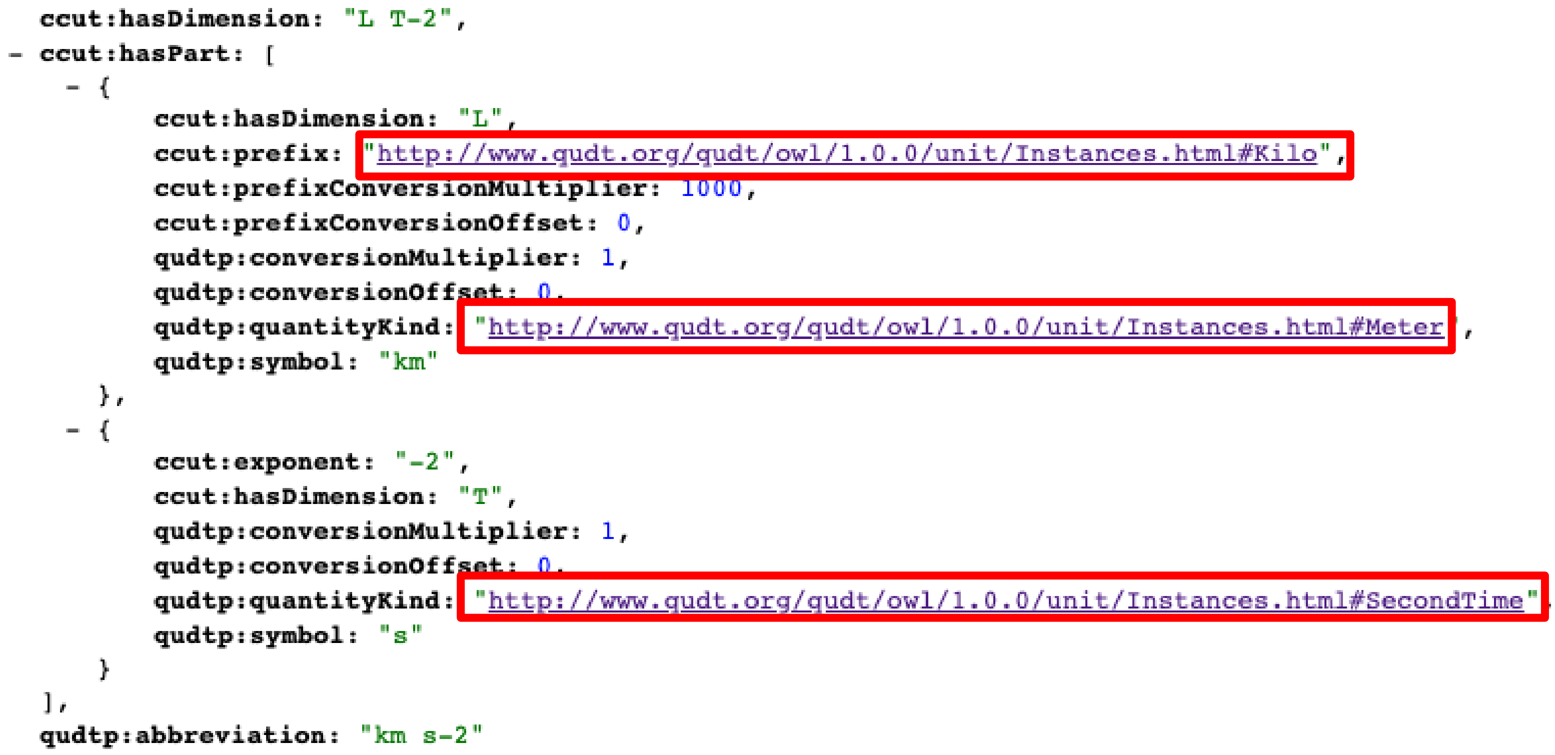
http://data.qudt.org/qudt/owl/1.0.0/unit/Instances.html#Foot



unit:Foot	
Property	**Value**
qudt:abbreviation	ft
qudt:code	0625
qudt:conversionMultiplier	0.3048
qudt:conversionOffset	0.0
qudt:quantityKind	quantity:Length
qudt:symbol	ft

Structured Unit Representation – Example

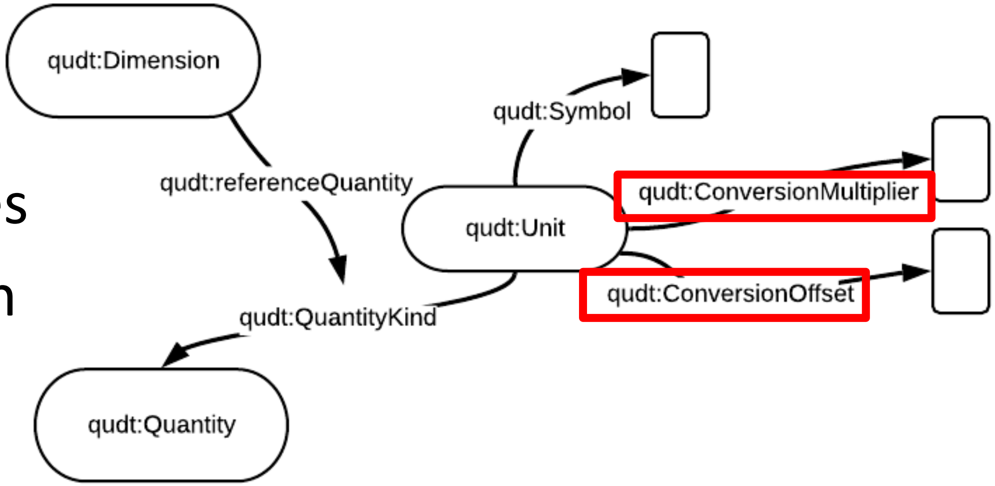
- Example output for ‘km/s^2’:



```
    ccut:hasDimension: "L T-2",
  - ccut:hasPart: [
      - {
            ccut:hasDimension: "L",
            ccut:prefix: "http://www.qudt.org/qudt/owl/1.0.0/unit/Instances.html#Kilo",
            ccut:prefixConversionMultiplier: 1000,
            ccut:prefixConversionOffset: 0,
            qudtp:conversionMultiplier: 1,
            qudtp:conversionOffset: 0,
            qudtp:quantityKind: "http://www.qudt.org/qudt/owl/1.0.0/unit/Instances.html#Meter",
            qudtp:symbol: "km"
        },
      - {
            ccut:exponent: "-2",
            ccut:hasDimension: "T",
            qudtp:conversionMultiplier: 1,
            qudtp:conversionOffset: 0,
            qudtp:quantityKind: "http://www.qudt.org/qudt/owl/1.0.0/unit/Instances.html#SecondTime",
            qudtp:symbol: "s"
        }
    ],
    qudtp:abbreviation: "km s-2"
```

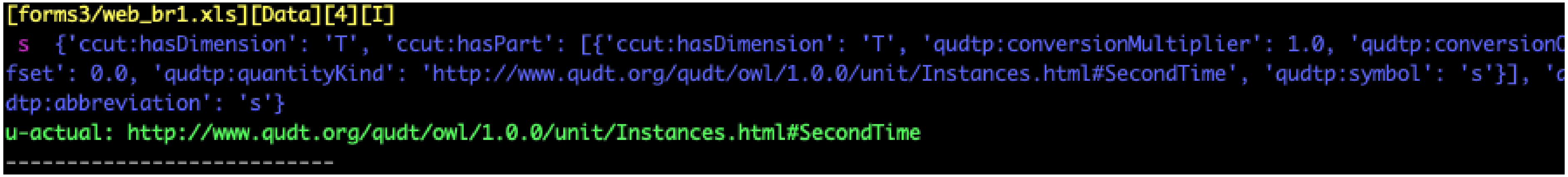
Transforming Units

- Goal: enable arbitrary **transformations** between units
- Given:
  - Structured semantic representation
  - Conversion attributes
  - Grammar elements
- Compute:
  - Transformation Attributes
  - Dimension Normalization



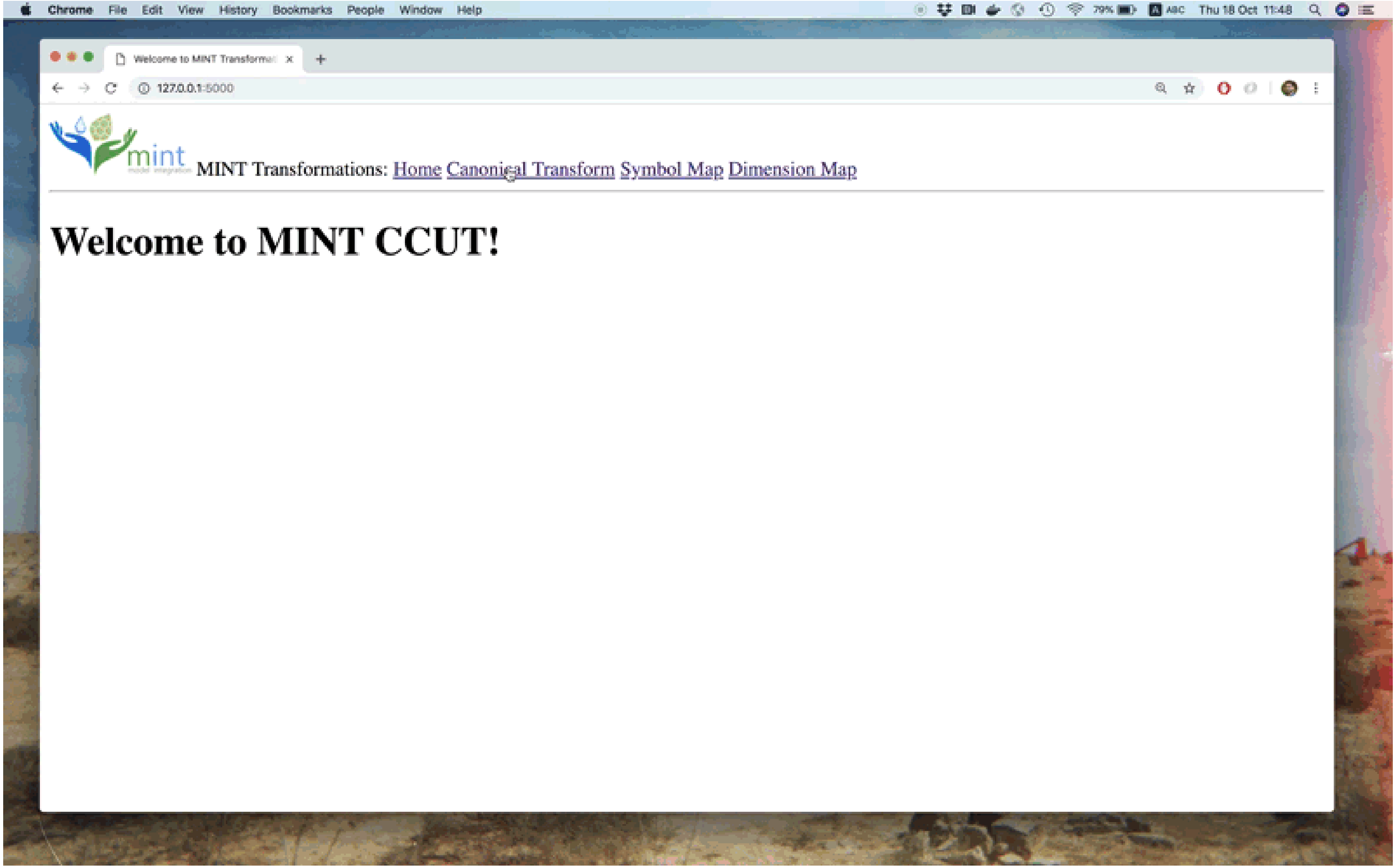
The CCUT Service

- Prototype system: CCUT
  - **C**anonicalization **C**ompound **U**nit Representation and **T**ransformation
- Deployed over a docker image
  - No user additional installations
- Invoked via:
  - Application program interface (API) with an HTTP endpoint
  - User-friendly web service



```
[forms3/web_br1.xls][Data][4][I]
 s  {'ccut:hasDimension': 'T', 'ccut:hasPart': [{'ccut:hasDimension': 'T', 'qudtp:conversionMultiplier': 1.0, 'qudtp:conversionO
fset': 0.0, 'qudtp:quantityKind': 'http://www.qudt.org/qudt/owl/1.0.0/unit/Instances.html#SecondTime', 'qudtp:symbol': 's'}], 'q
dtp:abbreviation': 's'}
u-actual: http://www.qudt.org/qudt/owl/1.0.0/unit/Instances.html#SecondTime
----------------------------
```

CCUT Demonstration



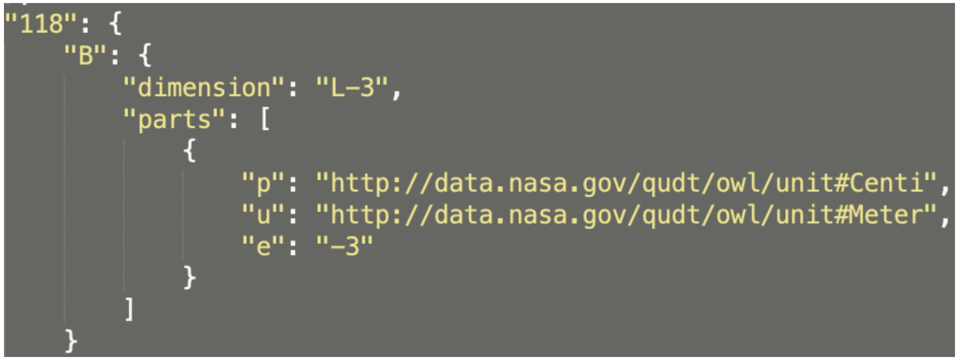
Chrome File Edit View History Bookmarks People Window Help
Thu 18 Oct 11:48
Welcome to MINT Transformat
127.0.0.1:5000
mint

MINT Transformations: Home Canonical Transform Symbol Map Dimension Map

Welcome to MINT CCUT!

Evaluation

- EUSES spreadsheet corpus [Fisher and Rothermel]
  - **1345** files
  - **5891** spreadsheets
  - Different sources (financial, physical, inventories, databases, modeling)
- Random sample:
  - **30** files
  - **112** spreadsheets
  - **267** compound units
    - Total of **530** atomic units
- Spreadsheet file reader as PoC
- Manual annotation to match QUDT URIs



```
"118": {
    "B": {
        "dimension": "L-3",
        "parts": [
            {
                "p": "http://data.nasa.gov/qudt/owl/unit#Centi",
                "u": "http://data.nasa.gov/qudt/owl/unit#Meter",
                "e": "-3"
            }
        ]
    }
```

Results

- Atomic unit **detection**:

Total Detected **(TP + FN)**	**TP** **(True Positives)**	**FP** **(False Positives)**	**Total Misdetected** **(False Negatives)**
882	328	554	150

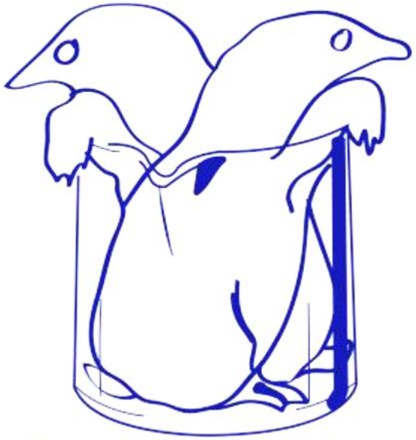
Precision	**Recall**	**F1-score**
37.2%	**68.6%**	**0.48**

- Compound units **representation**: **62.1%**
  - – Normalized correctly (dimension inference was precise)
- Compound units **transformation**: **100%**
  - – Identified **11** distinct dimension groups
    - Total of **42** test cases of pairs
    - Normally what we expected due to correct representation

Results Discussion

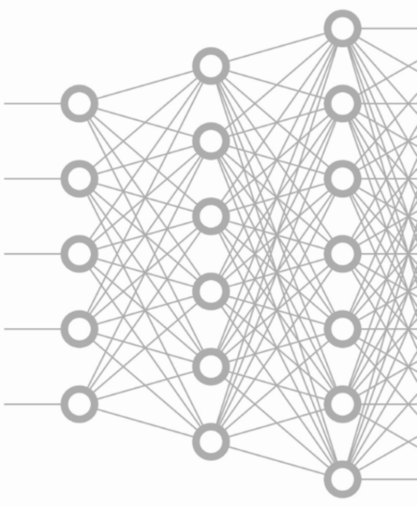
- Why is the performance low?
  - Irrelevant text
  - Abbreviations of entities or organizations
  - Ambiguity
    - ’L’ = liter (volume) vs. lambert (luminance)
  - Incomplete knowledge base

- Several limitations:
  - Naïve text matching
  - Inability to use context for disambiguation



2 moles per liter

Future work



- Use context
  - Co-occurrence of units within a domain
  - Locations in datasets (e.g., column headers)
- Use machine learning techniques
- Expand KB
- UI for self customized units and their attributes
- Detect variables with temporal and geospatial scoping
  - Solve the broader problem of table understanding

Conclusions

- Presented baseline **unsupervised** approach to:
  - **Identify** units of measurement in source data
  - Provide corresponding **semantic representation**
  - Provide a method (API) that enables **unit conversions**
- Our preliminary results demonstrate:
  - **Automatic** capture and transform units over spreadsheets
  - **Easy** deployment over quantitative data resources
  - **Accelerate** modeling process in scientific domains

- Source code available at:

https://github.com/basels/ccut