Parsing, Representing and Transforming Units of Measure

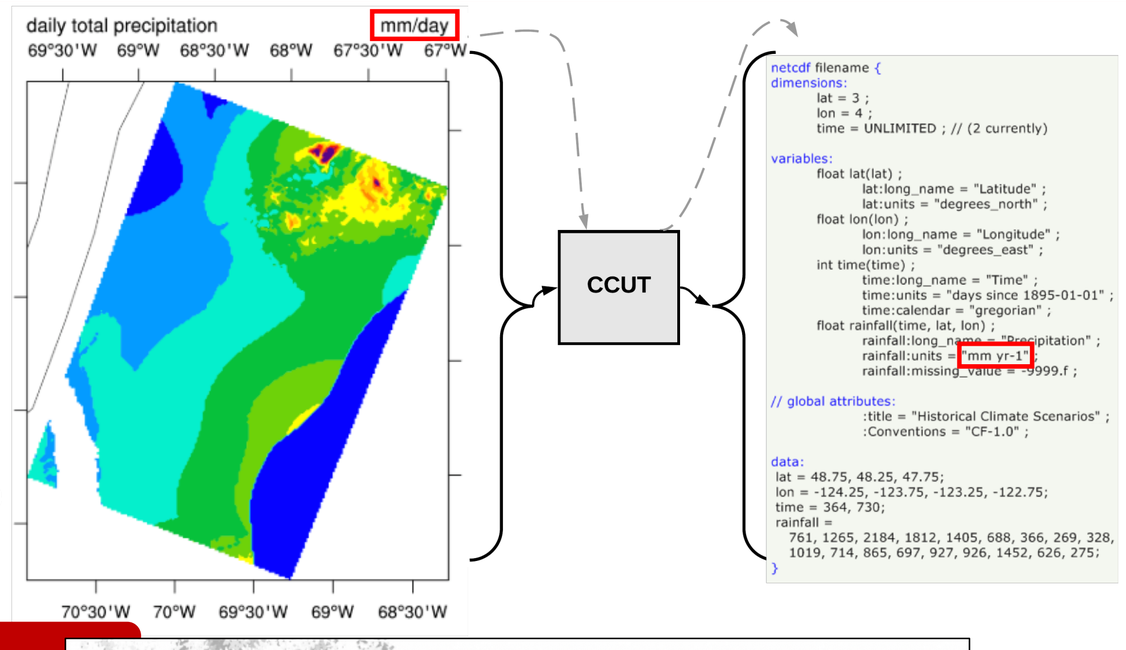
Basel Shbita, Arunkumar Rajendran, Jay Pujara, and Craig A. Knoblock
✉: {shbita, arunkumr, jpujara, knoblock}@usc.edu
Center on Knowledge Graphs, USC Information Sciences Institute

Problem

- **Identification** and **reusability** of **measurement units** in datasets across **domains** is a difficult task
- Units are in textual form with no **semantic** or dimensional meaning
- May require additional inquiry if one needs to perform **transformations** and data alignment

Our Task

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



Challenges

- Units appear in files in a textual representations that is not easily recognized
- Supporting SI prefixes adds an additional layer of combinatorial complexity
- Frequently, an additional investigation is required if one needs to perform transformations such as unit conversions
- We need a structured standard ontologized output that can be easily interpreted and used by humans and machines

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

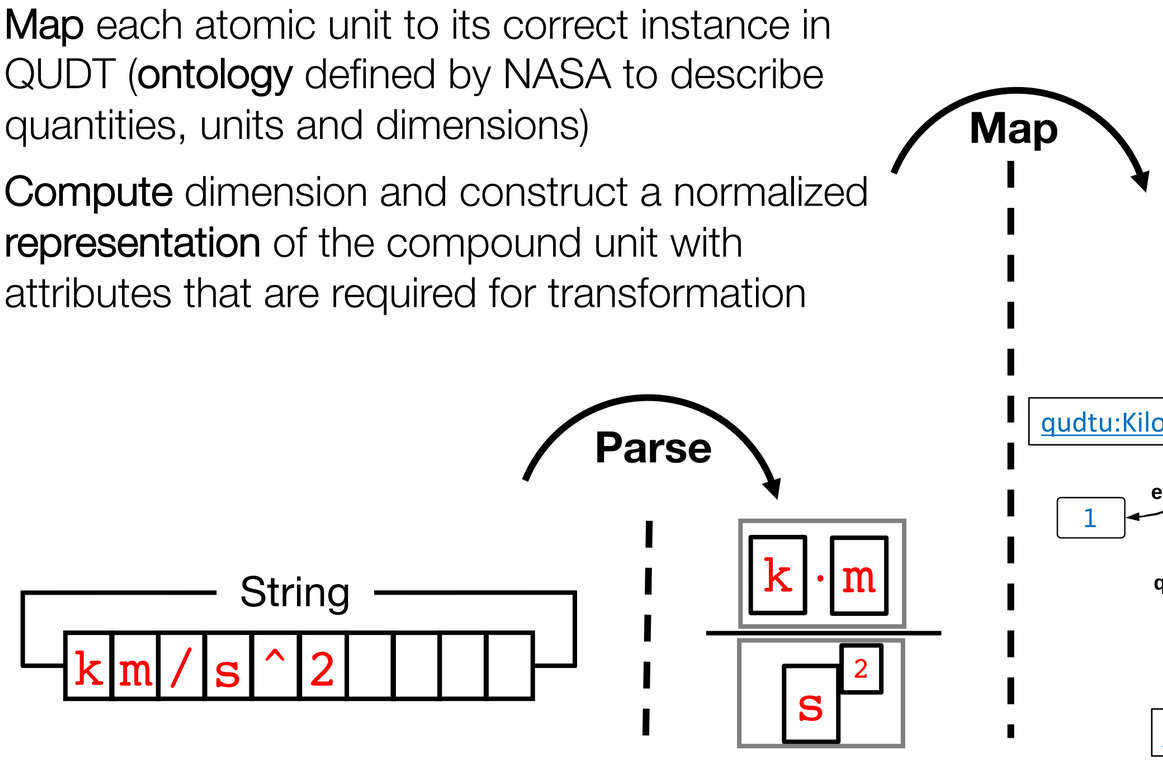
1999 Airborne-Tri

Nuclear Plant	Total GBq)	Tota (Ci)
S. Texas 1	872.238	23.574
S. Texas 2	461.5306	12.4738
	1344.58	36.34
	3583.82	96.86

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)

Our Approach

1. **Identify** and **parse** the composing elements of a compound unit: prefixes, atomic units, exponents and multipliers
2. **Map** each atomic unit to its correct instance in QUDT (**ontology** defined by NASA to describe quantities, units and dimensions)
3. **Compute** dimension and construct a normalized **representation** of the compound unit with attributes that are required for transformation



Parsing

Goal: string → structured form with relations
How? define a grammar using Arpeggio, a 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)
```

Structured Unit Representation

Goal: capture a semantic meaning
How? map decomposed elements to QUDT and utilize additional grammar elements

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

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

Transforming Units

Goal: arbitrary unit transformations
How? leverage QUDT ontology elements by employing cost-free conversion attributes

Input Unit
km^2/s^3
Output Unit
inch^2/hr^3

Evaluation and Results

- Implemented a prototype system, called CCUT
- Randomly sampled 30 files out of 1345 from the EUSES spreadsheet corpus
- Corpus collected from different sources (financial, physical, inventories, modeling)
- Manually annotated the sampled files to match QUDT URIs

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

Detection	Precision	Recall	F1-score
	37.2%	68.6%	0.48

Representation	62.1%

Transformation	100%

Discussion of Results

How can we do better?

- Eliminate cases where units are detected in irrelevant text
- Utilize an NER tagger to avoid annotating entities as units
- Use context for disambiguation:
  - Co-occurrence of units within a domain
  - Locations in datasets
  - ML techniques
- Expand knowledge base due to incompleteness

Contribution

A baseline **unsupervised** approach to:

- **Identify** units of measurement in source data
- Provide a corresponding **semantic representation**
- Provide a method (API) that enables **unit conversions**

Source code available at:
https://github.com/basels/ccut