Getting the units right

Moritz Schubotz[1], David Veenhuis[1], and Howard S. Cohl[2]

[1] Database Systems and Information Management Group,
Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, Germany
schubotz@tu-berlin.de, david.veenhuis@campus.tu-berlin.de
[2] Applied and Computational Mathematics Division,
National Institute of Standards and Technology, Gaithersburg, Maryland, U.S.A.
howard.cohl@nist.gov
http://units.formulasearchengine.com

Abstract. To understand applied physics, and physical formulae in particular, the investigation of identifier units is beneficial. However, normally the units are not given explicitly in formulae and have to be inferred. In this paper, we investigate how this process can be automated. As an example application, we use physical formulae from Wikipedia together with information from the related knowledge-base Wikidata. We envision that this method can be generalized and describe how, in the future, hard logical constraints may be used as a feedback mechanism for statistical methods in the context of natural language processing.

Keywords: Wikidata, Wikipedia, Units, Natural Language Processing, Inference, Mathematical Language Processing, Constraint propagation

1 Introduction

Units play an essential role in the physical sciences. Especially, *dimensional analysis* is one of the most significant tools for comprehension and understanding of physical formulae. We claim that this technique is not only beneficial for humans on their way to physical understanding, but also to machines that are programmed to semantically enrich mathematical and especially, physical content.

In this paper, we consider physical units in Wikipedia, as a first step towards a general solution of the underlying problem. We aim to: (1) identify formulae that deal with physical relationships; (2) automatically derive the units of the identifiers used in those formulae; and (3) integrate and store the learned data in the central Wikimedia triple store Wikidata.

Our paper is structured as follows. First, we analyze related works that can be used to complete the task at hand. In that context, we recap how possible definitions can automatically be extracted from the text surrounding formulae. Thereafter, we describe our method to relate the identifier to dimensions using the Wikidata knowledge base and our approach to unit constraint propagation. This will be followed by a refinement of the definition candidates based on the

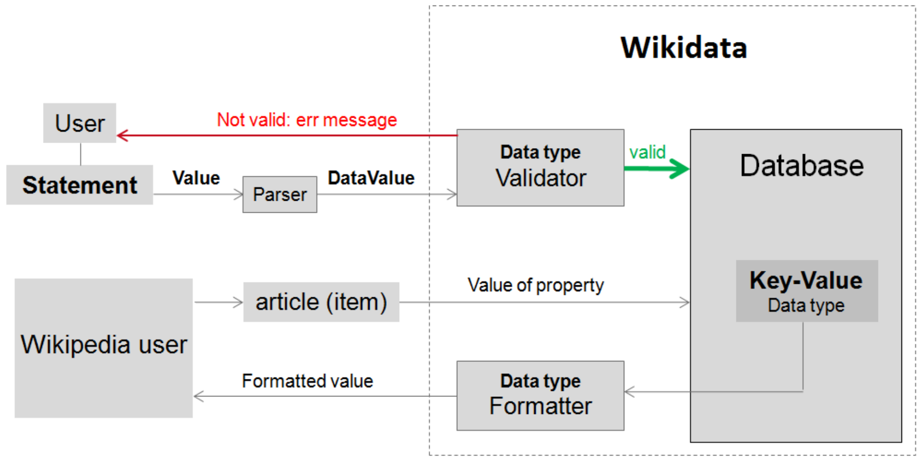


Fig. 1: As of January 2016, Wikidata users (visualized by the grey box 'User', top left in the figure) can store mathematical expressions in Wikidata. Thereafter, these expressions can be displayed in all language versions of Wikipedia (visualized by "Wikipedia user", bottom left). Moreover, other use cases for this data are possible. For example, there is the article place-holder service which displays information regarding a topic in languages, for which human generated articles in that language do not exist yet. For more details see [6] (picture by Julian Hilbig and Duc Linh Tran [6]).

constraints. Lastly, we describe the reinsertion of learned units into the Wikidata knowledge base, which will be used for the formulae we processed. Finally, we provide an outlook on how this method can be used for feedback driven self-tuning of Mathematical Language Processing [11].

2 Related Work

A method to find *identifier definiens tuples* in natural language text is proposed in [11]. There, the authors use Natural Language Processing (part of speech tagging combined with word distance based scoring) to get the tuples. This approach shows advantages over the more static pattern-matching approaches, because it is able to also retrieve results that do not follow the pattern $\langle identifier \rangle$ *is* $\langle description \rangle$(see also [8]).

This method has been applied to Wikipedia articles to enrich formulae with definitions for their included identifiers. It works for Wikipedia sites with different languages. The result may have more than a single possible definition for an identifier, each with a probability that expresses the likeliness for the definition to be the relevant. The selection of the correct definition is a problem addressed in our paper.

A method to map the meaning to identifiers is used in [12]. In that paper the namespace concept known from programming languages is used to assign documents to namespaces and thus the meaning of the identifiers are mapped to the meaning belonging to the chosen namespace.

In [9], an algorithm is proposed to use the need for dimensional homogeneity in physical formulae compared with constraint propagation to prove formulae that students gave as answers to physics problems. It aims on validating the formulae for known units.

Dimensional analysis is also a widely investigated field in the area of programming languages. In [4], the authors use the need for dimensional homogeneity in physical equations in conjunction with constraint solving to automatically infer unit types for programs handling scientific problems. Their approach infers a general set of unit types using constraints created over the variables and constants occurring in the program. The user can then annotate the inferred unit types with real units thereby cannot violate the dimensional correctness as it is proved for the inferred unit type system. In [1], constraint solving is used to prove the unit correctness of calculations in spreadsheets. More examples for validating dimensional correctness in programs can be found in [3, 7].

As preparation of this work, a new feature in Wikidata, the data-type mathematical expression, has been developed [6]. Properties with data-type mathematical expression with for instance, *defining formulae*, represent mathematical expressions. As of January 2016, these expressions are rendered by the software which runs Wikidata.

3 Our Method

3.1 Identify Physical Formulae

In this paper, we limit ourselves to the following definition.

Definition 1 (physical formula). *A physical formula is a binary mathematical relation of type equation or inequality containing one or more physical quantities.*

Consider the following examples:

$$E = mc^2, \tag{1}$$

$$\sin^2\theta + \cos^2\theta = 1, \tag{2}$$

$$m_{\text{moon}} < m_{\text{earth}} < m_{\text{sun}}, \tag{3}$$

$$\hbar = \frac{\lambda p}{2\pi}. \tag{4}$$

Expressions (1), (4) are physical formulae according to Definition 1. They contain the **physical quantities** $E, m, c, \lambda, p, \hbar$ of which $c, \hbar$ are **physical constants**. Expressions (2), (3) are not physical formulae since (2) does not contain physical quantities, and (3) is not binary. Note that the above definition can be extended to non-binary relation chains without loss of generality.

This definition implies the following algorithm to identify physical formulae:

1. identify mathematical expressions;
2. check if they are binary relations of type equation or inequality;
3. extract identifiers; and
4. decide for each identifier, if it is a physical quantity or expression.

While we will apply the heuristics from [12] for steps 1 and 3, we need to develop new approaches for steps 2 and 4. A simple approach to 2, is to convert the mathematical expression to content MathML using LaTeXML [10] and afterwards analyze the content MathML tree using fixed rules. We thereby rely on LaTeXML. Possibly occurring problems and limitations of LaTeXMLfor our application will be listed in the final report. The main focus of our work will be on step 4. While we can find Wikidata items from the algorithm presented in [12], we might need to improve these algorithms.

Our main focus is on the development of an algorithm which decides if an item is a physical quantity or entity. Our approach to address this problem is to analyze the semantic properties of the relevant Wikidata item using the `SPARQL` [5] [3] endpoint. This means all information from Wikidata that expose information on the units or dimensions respectively. More technically, we will develop a method to check the relatedness to `Q107715` (physical quantity)[4]. This will be one of the key contributions of our research project.

3.2 Identifying units and dimension of physical quantities

Table 1: Dimension, base unit, and symbol according to the international system of units (SI)

Dimension	**Unit**	**Symbol**
Length	meter	L
Mass	kilogram	M
Time	second	T
Electric Current	ampere	I
Luminous Intensity	candela	J
Temperature	kelvin	θ
Amount of Substance	mole	N

The dimension of a physical quantity is an inherent property of each quantity. Dimensions are for example length L, mass M or time T. The derived quantity

[3] SPARQL is used to query RDF (Ressource Description Framework) triples. They consist of subject, predicate, object. Example: Find all subjects (items) in Wikidata that have predicate "subclass of" and object "physical quantity"

[4] Wikidata stores information in form of triples like ("length","subclass of","physical quantity") where the unique id of "physical quantity" in Wikidata is Q107715. If a Item like "length" has a relation like "subclass of" or "instance of" to the Item for "physical quantity" we suppose it to be of kind physical quantity.

speed has the dimension LT^{-1}. For all physical quantities, we try to derive their dimension from Wikidata using SPARQL queries.

To obtain that, we also take unit information into account, since it is more prevalent in the Wikidata dataset compared to dimension information. Because there are multiple unit systems (e.g., imperial units using yard for length versus SI units using meter) more than one unit per dimension exists. However, physical laws are usually valid independent from the unit system that is used. In this context, it has to be noted that some adjustment needs to be done for units that disregard conceptually important physical properties. For instance the Carnot efficiency η_{Carnot} depends on the absolute temperature scale (e.g., kelvin) of the hot T_{H} and the cold T_{C} reservoir via

$$\eta_{\text{Carnot}} = 1 - \frac{T_{\text{C}}}{T_{\text{H}}}. \tag{5}$$

Note that the fact that those temperatures are based on an absolute temperature scale is essential. This requires that non-cardinal temperature units such as Celsius and Fahrenheit to be converted to prior to computation.

Given the extracted dimensions, the mathematical domain of physical quantities can be specified better. Approaches to formally describe this domain are presented in [2]. We introduce the following notation for a physical quantity x

$$x \equiv \begin{bmatrix} \mathsf{x} \\ d \end{bmatrix},$$

where x is the *spatial part* of x as defined in [2] and d is the dimension of the unit of x. To simplify readability, we use the identifier for x and its spatial part. We write $x = \mathsf{x}$ if $d = 1$. Thus, (1) can be written as

$$\begin{bmatrix} \mathsf{E} \\ ML^2T^{-2} \end{bmatrix} = \begin{bmatrix} \mathsf{m} \\ M \end{bmatrix} \begin{bmatrix} \mathsf{c} \\ LT^{-1} \end{bmatrix}^2,$$

and (5) reads

$$\eta_{\text{Carnot}} = 1 - \frac{\begin{bmatrix} \mathsf{T}_{\text{C}} \\ \theta \end{bmatrix}}{\begin{bmatrix} \mathsf{T}_{\text{H}} \\ \theta \end{bmatrix}} = 1 - \begin{bmatrix} \frac{\mathsf{T}_{\text{C}}}{\mathsf{T}_{\text{H}}} \\ \theta\theta^{-1} \end{bmatrix} = 1 - \frac{\mathsf{T}_{\text{C}}}{\mathsf{T}_{\text{H}}}. \tag{6}$$

3.3 Compatible operations

This notation leads to our next definition

Definition 2 (valid physical formula). *We call a physical formula valid, if the dimensions are* compatible *with the mathematical operators used in that formula.*

Table 2: Compatibility of Mathematical Operations with Physical units. Here, $|.|_1$ denotes the 1-norm.

class	rule	constraint	operators
3: map	$o\left(\begin{bmatrix}\mathsf{a}\\x\end{bmatrix},\begin{bmatrix}\mathsf{b}\\y\end{bmatrix}\right)\to\begin{bmatrix}\mathsf{o(a,b)}\\\frac{o(x,y)}{\lvert o(x,y)\rvert_1}\end{bmatrix}$		times, division, integration, differentiation
2: restrict	$o\left(\begin{bmatrix}\mathsf{a}\\x\end{bmatrix},\begin{bmatrix}\mathsf{b}\\y\end{bmatrix}\right)\to\begin{bmatrix}\mathsf{o(a,b)}\\x\end{bmatrix}$	$x=y$	plus, minus, equals
1: apply	$o\left(\begin{bmatrix}\mathsf{a}\\x\end{bmatrix},\begin{bmatrix}\mathsf{b}\\y\end{bmatrix}\right)\to\begin{bmatrix}\mathsf{o(a,b)}\\o(x,b)\end{bmatrix}$	$y=1$	power, roots
0: unitless	$o\left(\begin{bmatrix}\mathsf{a}\\y\end{bmatrix}\right)\to\begin{bmatrix}\mathsf{o(a)}\\y\end{bmatrix}$	$y=1$	function application

We exemplify the meaning of compatible operations based on (1). This physical formula contains the mathematical operators equals (=), times ($\cdot$), and power ($\wedge$). For 'equals', the dimensions on the right-hand side and left-hand side must be the same, for 'times', no restrictions apply and for 'power' the unit of the exponent must be 1. If any of these constraints are violated (e.g., with the incorrect $E = mc$) then the units or the formula can not be correct. Note also that this method is not limited to scalar physical quantities and can be applied to physical quantities of higher dimensions such as vectors like

$$\begin{bmatrix}\mathsf{W}\\ML^2T^{-2}\end{bmatrix}=\int_C\begin{bmatrix}\underline{\mathsf{F}}\\MLT^{-2}\end{bmatrix}\mathrm{d}\begin{bmatrix}\underline{\mathsf{s}}\\L\end{bmatrix}=\int_{\left[\substack{\mathsf{t}_1\\T}\right]}^{\left[\substack{\mathsf{t}_2\\T}\right]}\begin{bmatrix}\underline{\mathsf{F}}\\MLT^{-2}\end{bmatrix}\begin{bmatrix}\underline{\mathsf{v}}\\LT^{-1}\end{bmatrix}\mathrm{d}\begin{bmatrix}\mathsf{t}\\T\end{bmatrix}.$$

We will implement validation for the mathematical operations enumerated in Table 2, which we group into four classes. For some of those cases [9] defined detailed unit propagation rules.

3.4 Unit Inference

After having defined those fundamental concepts, we apply them to the actual data extracted by the Mathematical Language Processing Project and the Unit information fetched from Wikidata.

As an example, we demonstrate the work-flow for adding dimensional information to formulae extracted from Wikipedia. The result of the process, proposed in [11], is a probability distribution over identifiers and their possible definiens. An identifier can have more than one definition candidate. That may lead to more than one possible dimension for an identifier.

Example 1: Mass-energy equivalence

The relation between energy and mass is described by the mass-energy equivalence formula $E = mc^2$, where E is energy, m is mass, and c is the speed of light.

For Example 1 (from Wikipedia), which was also used in [12], the identifier-definiens pairs for E may have this form:

Id.	Definition	score
E	energy	0.42
E	mass	0.42
E	speed of light	0.16

Id.	Definition	score
m	energy	0.35
m	mass	0.35
m	speed of light	0.30

Id.	Definition	score
c	energy	0.30
c	mass	0.35
c	speed of light	0.35

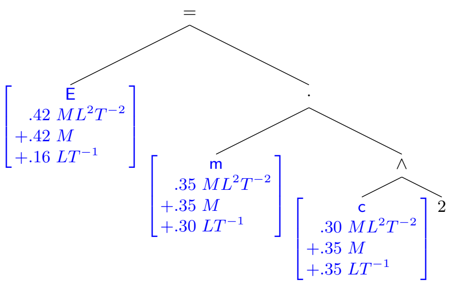
All three definitions are found in the same sentence, but have different distances between identifier and definiens. That results in different scores. Note, that the actual scoring computation is more evolved and the score values have been made up for demonstration purposes. We now describe this more formally.

For a physical formula $f(x_1,\dots,x_n,o_{n+1},\dots,o_m)$, with the identifiers (physical quantities and other identifiers) x_i and mathematical operators o_j, and a set of definiens candidates $\mathcal{N}$, the MLP Project returns a probability distribution for the identifiers $\chi(\underline{x}) = \sum_{N\in\mathcal{N}} \underline{\underline{w}}_{0,n} N$, with $\sum_{N\in\mathcal{N}} w_{0,N} = 1$. From Wikidata we get the unit and implied dimension information denoted as $\rho_1(\underline{x}) = \sum_{N\in\mathcal{N}} \underline{\underline{w}}_{0,n} \dim(N) = \sum_i \underline{\underline{w}}_{1,i} d_i$, where $\underline{\underline{w}}_{1,j}$ is the sum over all $\underline{\underline{w}}_{0,N}$ with $\dim N = d_j$. Next, we apply the operator compatibility rules, which affects the probability distribution according to their constraints. The dimension probability distribution of an operator o, $\dim\left(o(a,b)\right)$ implies operator dependent constraints to $\dim a$ and $\dim b$.

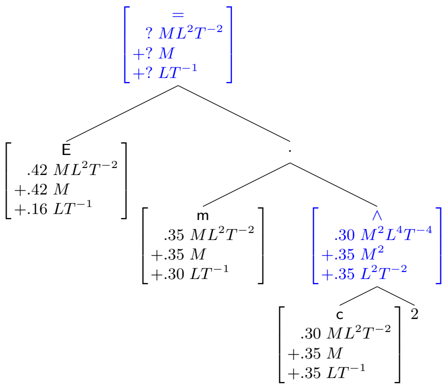
$$\begin{aligned}
\text{class}(o) = 3 &\implies \dim o(a,b) = o(\dim a, \dim b).\\
\text{class}(o) = 2 &\implies \dim a = \dim b.\\
\text{class}(o) = 1 &\implies \dim b = 1.\\
\text{class}(o) = 0 &\implies \dim a = 1,\ o(a,b) = o(a).
\end{aligned}$$

To reflect that, we define a refined probability distribution $\rho(\underline{x}) = \sum_i \underline{\underline{w}}_{2,i} d_i$, with $\sum_i \underline{\underline{w}}_{2,i} d_i = 1$. Finally, we need to solve a system of linear equations describing the $\underline{\underline{w}}_2 = \mathbf{A}\underline{\underline{w}}_1$. We propose the following method to realize that. Assume that the mathematical notation is good enough to extract the operator tree, with tools such as LaTeXML. We convert this operator to the root form and call $\underline{\underline{O}}(f) \in \mathbb{N}^{m\times m}$ the adjacent matrix. Since the tree is not directed $\underline{\underline{O}}(f) = \underline{\underline{O}}(f)^T$, and since identifiers are always connected by an operator, the top left entries of the adjacent matrix are zero, i.e., $\forall i \le n \wedge j \le n \implies \underline{\underline{O}}(f)_{i,j} = 0$. Based on that, we will elaborate different strategies for the most efficient execution of the constraint propagation problem. Our first approach is the following sketch of an algorithm:

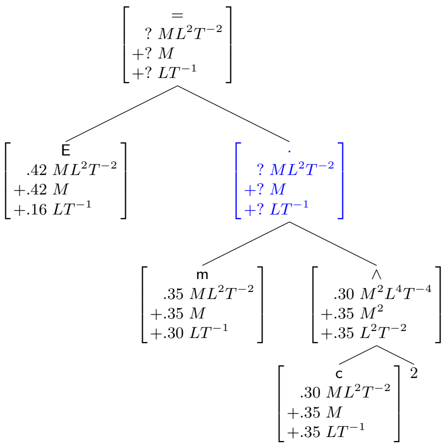
1. start with the identifiers $(\underline{x})$ as working set;



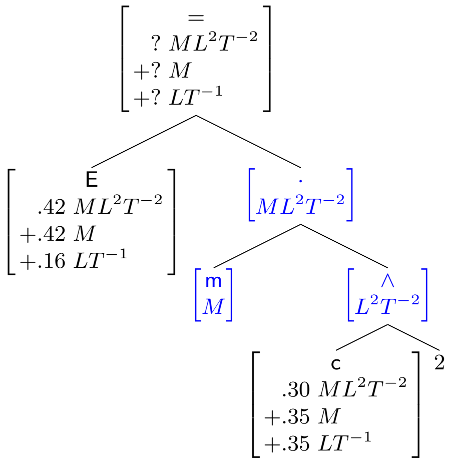
(1) Set identifier dimensions



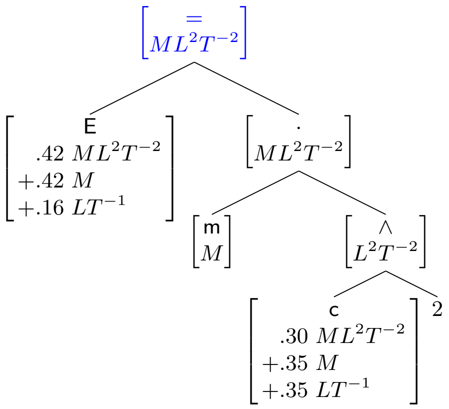
(2) Propagate upwards



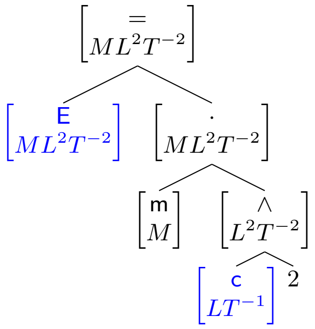
(3) Propagate downwards



(4) Process maps



(5) Propagate upwards



(6) Propagate downwards

Fig. 2: Demonstration of our constraint propagation algorithm.

2. propagate the constraints to their parent operators of classes 0-2 and remember the parent operators of class 3 (maps);
3. update the working set to consist of the updated parents;
4. propagate the constraints downwards;
5. the changed children are now the new working set;
6. continue with upwards and downwards propagation until the working set is empty, and a steady state is reached;
7. update the working set with the class 3 operators;
8. resolve the class 3 operators, in appropriate order (note that this might lead to many possible dimensions for the class 3 operators);
9. propagate the new constraints by going back to step 2;
10. the algorithm terminates if the working set and the set of unprocessed class 3 operators is empty.

We demonstrate this algorithm in figure 2 based on Example 1. Formula (1) depends on $(E, m, c, 2, =, \cdot, \wedge)$ and the adjacent matrix reads

$$O(\text{“}E = mc^2\text{”}) = O\begin{pmatrix} & = & & \\ \overset{\frown}{E \quad \cdot} & & & \\ & \overset{\frown}{m \ \wedge} & & \\ & & \overset{\frown}{c\,2} & \end{pmatrix} = \begin{array}{c|ccccccc} & E & m & c & 2 & = & \cdot & \wedge \\ \hline E & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ m & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ c & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ = & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \cdot & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ \wedge & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{array}.$$

One alternative to this approach is [9]. The evaluation of our approach must show the performance of the probabilistic approach.

3.5 Wikidata insert/update

After having propagated the units, we might have found a unique solution as demonstrated in Example 1. In cases where we found this unique solution, we will write back the dimension information to Wikidata. If the formula already has an item in Wikidata, we check the properties and update/insert the missing information. Since we use the Wikidata database for finding information for all the identifiers in a formula we have automatically a defining item to link to for every identifier.

3.6 Limitations

For level 3 operators, numerical artifacts (such as scalar factors from the integration) and probability density are mixed. For example in $\int s dt$ and a hypothetical $\rho_0(s) = .5M + .5T$ leads to a propagated probability of

$$\rho\left(\int s\mathrm{d}t\right) = \frac{\int \rho_0(s)\mathrm{d}t}{\left|\int \rho_0(s)\mathrm{d}t\right|_1} = \frac{4}{3}\left(\frac{1}{2}MT + \frac{1}{4}T^2\right) = \frac{2}{3}MT + \frac{1}{3}T^2.$$

The likelihood for MT is higher in comparison to T^2, which does not seem plausible at the first place. Consequently, we will search for better suitable ways to re-scale the unit vector to length 1.

4 Future work

The finding of items in Wikidata can be improved by using methods like stemming to match the definition with the item caption. This is due to the fact that the definitions are extracted from free-text and may be conjugated. The identification of the units/dimensions can be done by the querying property, *has quality*. Due to the community-driven inserts of items, it is not guaranteed that every item has all of the possible properties. For example, the authors may omit a property such as, *has quality*. Instead the property, *quantity symbol*, may be queried.

Moreover, our algorithm can be used as a feedback mechanism for the MLP process. With the additional unit and dimension information, the algorithm will learn about mistakes from the unit checking. With this information, the algorithm will be able to tune itself.

5 Conclusion

The knowledge about units in physical formulae is fundamental. However, in most cases they are not marked up explicitly. We investigated how they can be determined automatically. We described a process to automatically infer the unit information for identifiers in physical formulae. This process is based on formulae and identifier-definition pairs that are extracted from text using Natural Language Processing techniques. The process may result in multiple definition candidates for identifiers. To infer a consistent identifier-definition mapping, we use an algorithm based on the principle of dimensional homogeneity in physical formulae. The unit/dimensional information for the definitions is retrieved from the structured data store Wikidata. The results are then used to extend Wikidata with the formulae and links Wikidata entries describing the identifiers.

Acknowledgements. We would like to thank Volker Markl, Michael Kohlhase and Abdou Youssef for their support of this research project.

Bibliography

[1] T. Antoniu, P. A. Steckler, S. Krishnamurthi, E. Neuwirth, and M. Felleisen. Validating the unit correctness of spreadsheet programs. In A. Finkelstein, J. Estublier, and D. S. Rosenblum, editors, *26th International Conference on Software Engineering (ICSE 2004), 23-28 May 2004, Edinburgh, United Kingdom*, pages 439–448. IEEE Computer Society, 2004.

[2] J. B. Collins. A mathematical type for physical variables. In S. Autexier, J. A. Campbell, J. Rubio, V. Sorge, M. Suzuki, and F. Wiedijk, editors, *Intelligent Computer Mathematics, 9th International Conference, AISC 2008, 15th Symposium, Calculemus 2008, 7th International Conference, MKM 2008, Birmingham, UK, July 28 - August 1, 2008. Proceedings*, volume 5144 of *Lecture Notes in Computer Science*, pages 370–381. Springer, 2008.

[3] M. Contrastin, A. C. Rice, M. Danish, and D. A. Orchard. Units-of-measure correctness in fortran programs. *Computing in Science and Engineering*, 18(1):102–107, 2016.

[4] P. Guo and S. McCamant. Annotation-less unit type inference for c. In *Final Project, 6.883: Program Analysis, CSAIL, MIT*, 2005.

[5] S. Harris and A. Seaborne. SPARQL 1.1 Query Language. `https://www.w3.org/TR/sparql11-query`. seen May, 2016.

[6] J. Hilbig and D. L. Tran. Mathematical expression as new data type for WikiData - Database project - supervised by Moritz Schubotz. Technical Report Winter-term 2015/2016, Technische Universität Berlin, feb 2016. `https://github.com/TU-Berlin/WikidataMath/releases/download/v1.0.0/ReportWikiDataDBPRO.pdf`.

[7] L. Jiang and Z. Su. Osprey: A practical type system for validating dimensional unit correctness of C programs. In *Proceedings of the International Conference on Software Engineering*, 2006.

[8] G. Y. Kristianto, G. Topic, and A. Aizawa. Extracting textual descriptions of mathematical expressions in scientific papers. *D-Lib Magazine*, 20(11/12), 2014.

[9] C. W. Liew. Checking for dimensional correctness in physics equations. In *In Proceedings of Fourteenth International Florida AI Research Society Conference*, 2002.

[10] B. R. Miller. `LaTeXML`: A LaTeX to XML converter. `http://dlmf.nist.gov/LaTeXML`. seen May, 2016.

[11] R. Pagel and M. Schubotz. Mathematical language processing project. In M. England, J. H. Davenport, A. Kohlhase, M. Kohlhase, P. Libbrecht, W. Neuper, P. Quaresma, A. P. Sexton, P. Sojka, J. Urban, and S. M. Watt, editors, *Joint Proceedings of the MathUI, OpenMath and ThEdu Workshops and Work in Progress track at CICM*, number 1186 in CEUR Workshop Proceedings, Aachen, 2014.

[12] M. Schubotz, A. Grigorev, M. Leich, H. S. Cohl, N. Meuschke, B. Gipp, A. S. Youssef, and V. Markl. Semantification of identifiers in mathematics for better math information retrieval. In *Proceedings of the 39th Int. ACM SIGIR Conference on Research and Development in Information Retrieval*, 2016.