

Semantic Search for Quantity Expressions

Tom Wiesing

Supervisor: Michael Kohlhase

Co-supervisor: Tobias Preusser

May 20, 2015

110392 Guided Research Applied and Computational
Mathematics & Thesis

Overview

Overview

- ▶ Motivation: Problem and State Of The Art

Overview

- ▶ Motivation: Problem and State Of The Art
- ▶ Our Approach: Structure Of The Search Engine

Overview

- ▶ Motivation: Problem and State Of The Art
- ▶ Our Approach: Structure Of The Search Engine
 - ▶ The Unit System

Overview

- ▶ Motivation: Problem and State Of The Art
- ▶ Our Approach: Structure Of The Search Engine
 - ▶ The Unit System
 - ▶ The Search Algorithm

Overview

- ▶ Motivation: Problem and State Of The Art
- ▶ Our Approach: Structure Of The Search Engine
 - ▶ The Unit System
 - ▶ The Search Algorithm
- ▶ Conclusion: The Implementation

Overview

- ▶ Motivation: Problem and State Of The Art
- ▶ Our Approach: Structure Of The Search Engine
 - ▶ The Unit System
 - ▶ The Search Algorithm
- ▶ Conclusion: The Implementation
- ▶ Time for Questions

Motivation (1)

- ▶ We use units every day

Motivation (1)


- ▶ We use units every day
- ▶ We encounter them everywhere:

Motivation (1)


- ▶ We use units every day
- ▶ We encounter them everywhere:
 - ▶ When driving, there are speed limits, for example:




Motivation (1)

- ▶ We use units every day
- ▶ We encounter them everywhere:
 - ▶ When driving, there are speed limits, for example:  $\frac{\text{km}}{\text{h}}$
 - ▶ When baking, it often says in recipes something like: “add 3 tea spoons of sugar”


Motivation (1)

- ▶ We use units every day
- ▶ We encounter them everywhere:
 - ▶ When driving, there are speed limits, for example:  $\frac{\text{km}}{\text{h}}$
 - ▶ When baking, it often says in recipes something like: “add 3 tea spoons of sugar”
 - ▶ When shopping for shoes there are different sizes


Motivation (1)

- ▶ We use units every day
- ▶ We encounter them everywhere:
 - ▶ When driving, there are speed limits, for example:  $\frac{\text{km}}{\text{h}}$
 - ▶ When baking, it often says in recipes something like: “add 3 tea spoons of sugar”
 - ▶ When shopping for shoes there are different sizes
- ▶ In scientific papers they occur a lot

Motivation (1)

- ▶ We use units every day
- ▶ We encounter them everywhere:
 - ▶ When driving, there are speed limits, for example:  $\frac{\text{km}}{\text{h}}$
 - ▶ When baking, it often says in recipes something like: “add 3 tea spoons of sugar”
 - ▶ When shopping for shoes there are different sizes
- ▶ In scientific papers they occur a lot
- ▶ everything which somehow models a real system has at least one quantity expression

Motivation (1)

- ▶ We use units every day
- ▶ We encounter them everywhere:
 - ▶ When driving, there are speed limits, for example:  $\frac{\text{km}}{\text{h}}$
 - ▶ When baking, it often says in recipes something like: “add 3 tea spoons of sugar”
 - ▶ When shopping for shoes there are different sizes
- ▶ In scientific papers they occur a lot
- ▶ everything which somehow models a real system has at least one quantity expression
- ▶ everything is quantified

Motivation (2)

- ▶ But where is the problem?

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths:

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths:

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter*,

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch,*

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch, Foot,*

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch, Foot, Mile,*

Motivation (2)

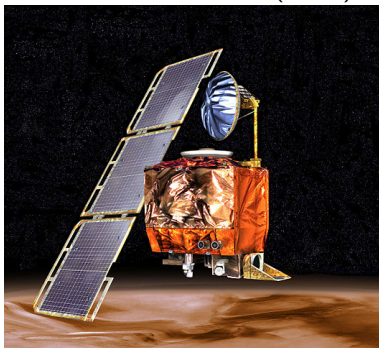
- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch, Foot, Mile, Nautical Mile,*

Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch, Foot, Mile, Nautical Mile, ...*
- ▶ This can cause problems when not converting properly

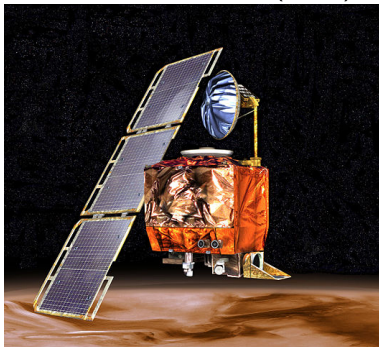
Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch, Foot, Mile, Nautical Mile, ...*
- ▶ This can cause problems when not converting properly
- ▶ Mars Climate Orbiter (1999)



Motivation (2)

- ▶ But where is the problem?
- ▶ Within one paper only a handful of units is used
- ▶ In general there are **a lot** of **different** units to describe **the same** quantity
- ▶ Just for lengths: *Meter, Inch, Foot, Mile, Nautical Mile, ...*
- ▶ This can cause problems when not converting properly
- ▶ Mars Climate Orbiter (1999)



- ▶ Different Units are a big problem

Motivation (3)

Motivation (3)

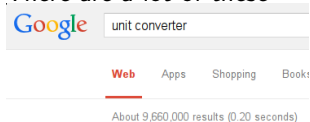
- ▶ What is the most common solution?

Motivation (3)

- ▶ What is the most common solution?
- ▶ Unit Converters

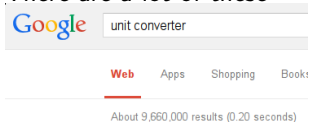
Motivation (3)

- ▶ What is the most common solution?
- ▶ Unit Converters
 - ▶ There are a lot of these

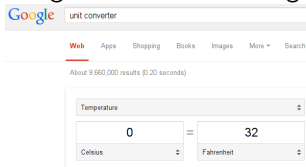


Motivation (3)

- ▶ What is the most common solution?
- ▶ Unit Converters
 - ▶ There are a lot of these



- ▶ Google itself has one integrated



Motivation (4)

- ▶ user needs to find out that conversion is required

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**
- ▶ this is not integrated into the search process itself

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**
- ▶ this is not integrated into the search process itself
- ▶ Wouldn't it be nice:

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**
- ▶ this is not integrated into the search process itself
- ▶ Wouldn't it be nice:
 - ▶ when searching for $25 \frac{\text{m}}{\text{s}}$

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**
- ▶ this is not integrated into the search process itself
- ▶ Wouldn't it be nice:
 - ▶ when searching for $25 \frac{\text{m}}{\text{s}}$
 - ▶ we also find $90 \frac{\text{km}}{\text{h}}$

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**
- ▶ this is not integrated into the search process itself
- ▶ Wouldn't it be nice:
 - ▶ when searching for $25 \frac{\text{m}}{\text{s}}$
 - ▶ we also find $90 \frac{\text{km}}{\text{h}}$
 - ▶ we did not have to search for all the representations of the same quantity expression

Motivation (4)

- ▶ user needs to find out that conversion is required
- ▶ both input **and** output units need to be **given**
- ▶ this is not integrated into the search process itself
- ▶ Wouldn't it be nice:
 - ▶ when searching for $25 \frac{\text{m}}{\text{s}}$
 - ▶ we also find $90 \frac{\text{km}}{\text{h}}$
 - ▶ we did not have to search for all the representations of the same quantity expression
- ▶ This is the kind of search engine we have built

Our Approach (1)

- ▶ What components do we need for a semantic search engine?

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE
 2. A *Spotter* that finds representations of QEs inside documents

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE
 2. A *Spotter* that finds representations of QEs inside documents
 3. A *Search Algorithm* that given a QE finds all its representations in the system

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE
 2. A *Spotter* that finds representations of QEs inside documents
 3. A *Search Algorithm* that given a QE finds all its representations in the system
 4. A *Frontend* that allows queries to be made

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE
 2. A *Spotter* that finds representations of QEs inside documents
 3. A *Search Algorithm* that given a QE finds all its representations in the system
 4. A *Frontend* that allows queries to be made
- ▶ Spotter is done by *Stiv Sherko*

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE
 2. A *Spotter* that finds representations of QEs inside documents
 3. A *Search Algorithm* that given a QE finds all its representations in the system
 4. A *Frontend* that allows queries to be made
- ▶ Spotter is done by *Stiv Sherko*
- ▶ We only need to worry about the *Unit System*, the *Search Algorithm* and the *Frontend*

Our Approach (1)

- ▶ What components do we need for a semantic search engine?
 1. A *Unit System* that is aware of the different representations of a QE
 2. A *Spotter* that finds representations of QEs inside documents
 3. A *Search Algorithm* that given a QE finds all its representations in the system
 4. A *Frontend* that allows queries to be made
- ▶ Spotter is done by *Stiv Sherko*
- ▶ We only need to worry about the *Unit System*, the *Search Algorithm* and the *Frontend*
- ▶ For the first two points, we use a meta-mathematical approach

Our Approach (2)

- ▶ We use a meta-mathematical model

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*
- ▶ An Import makes Constants from one theory available in another

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*
- ▶ An Import makes Constants from one theory available in another
- ▶ A View is a truth-preserving mapping from Constants of one theory to another

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*
- ▶ An Import makes Constants from one theory available in another
- ▶ A View is a truth-preserving mapping from Constants of one theory to another
- ▶ With the help of these relation we can make a *Theory Graph*

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*
- ▶ An Import makes Constants from one theory available in another
- ▶ A View is a truth-preserving mapping from Constants of one theory to another
- ▶ With the help of these relation we can make a *Theory Graph*
- ▶ You will see examples with our unit system

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*
- ▶ An Import makes Constants from one theory available in another
- ▶ A View is a truth-preserving mapping from Constants of one theory to another
- ▶ With the help of these relation we can make a *Theory Graph*
- ▶ You will see examples with our unit system
- ▶ *MMT* is a software that allows us to make use of these concepts

Our Approach (2)

- ▶ We use a meta-mathematical model
- ▶ This is a model that can be used to describe mathematics
- ▶ We go over this fairly quickly due to time constraints
- ▶ Definitions (= Constants) are contained inside *theories*
- ▶ A *Term* is an expression written using definitions from a Theory
- ▶ Theories can be related using *Imports* and *Views*
- ▶ An Import makes Constants from one theory available in another
- ▶ A View is a truth-preserving mapping from Constants of one theory to another
- ▶ With the help of these relation we can make a *Theory Graph*
- ▶ You will see examples with our unit system
- ▶ *MMT* is a software that allows us to make use of these concepts
- ▶ It is easy to write down your own theories and related them with views

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current
 - ▶ temperature

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current
 - ▶ temperature
 - ▶ luminous intensity

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current
 - ▶ temperature
 - ▶ luminous intensity
 - ▶ amount of substance

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current
 - ▶ temperature
 - ▶ luminous intensity
 - ▶ amount of substance
- ▶ but there are also quantities where we just *count*

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current
 - ▶ temperature
 - ▶ luminous intensity
 - ▶ amount of substance
- ▶ but there are also quantities where we just *count*
- ▶ and *dimensionless quantities* (such as Information)

Our Approach: The Unit System (1)

- ▶ Let us start with building a theory of Quantity Expressions
- ▶ We orient ourselves on the SI specification
- ▶ Each quantity has a dimension
- ▶ According to SI there are 7 basic ones:
 - ▶ length
 - ▶ mass
 - ▶ time
 - ▶ electric current
 - ▶ temperature
 - ▶ luminous intensity
 - ▶ amount of substance
- ▶ but there are also quantities where we just *count*
- ▶ and *dimensionless quantities* (such as Information)
- ▶ so we have 9 basic dimensions

Our Approach: The Unit System (2)

- ▶ we can also multiply these to get new dimensions

Our Approach: The Unit System (2)

- ▶ we can also multiply these to get new dimensions
- ▶ such as $\text{area} = \text{length} \cdot \text{length}$

Our Approach: The Unit System (2)

- ▶ we can also multiply these to get new dimensions
- ▶ such as $\text{area} = \text{length} \cdot \text{length}$
- ▶ similarly we can divide dimensions

Our Approach: The Unit System (2)

- ▶ we can also multiply these to get new dimensions
- ▶ such as $\text{area} = \text{length} \cdot \text{length}$
- ▶ similarly we can divide dimensions
- ▶ such as $\text{velocity} = \frac{\text{length}}{\text{time}}$

Our Approach: The Unit System (2)

- ▶ we can also multiply these to get new dimensions
- ▶ such as $\text{area} = \text{length} \cdot \text{length}$
- ▶ similarly we can divide dimensions
- ▶ such as $\text{velocity} = \frac{\text{length}}{\text{time}}$
- ▶ We can use this to get a *Theory of Dimensions*:

Our Approach: The Unit System (3)

Dimension		
dim	:	type
none	:	dim
count	:	dim
length	:	dim
mass	:	dim
time	:	dim
current	:	dim
temperature	:	dim
luminous	:	dim
amount	:	dim
.	:	dim \rightarrow dim \rightarrow dim
/	:	dim \rightarrow dim \rightarrow dim

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter
 2. A *multiplication* of a (real) number with an existing QE, such as 5 Meter

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter
 2. A *multiplication* of a (real) number with an existing QE, such as 5 Meter
 3. A *division* of an existing QE by a (non-zero real) number (equivalent to the above)

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter
 2. A *multiplication* of a (real) number with an existing QE, such as 5 Meter
 3. A *division* of an existing QE by a (non-zero real) number (equivalent to the above)
 4. The *product* of two existing QEs such as Newton · Second

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter
 2. A *multiplication* of a (real) number with an existing QE, such as 5 Meter
 3. A *division* of an existing QE by a (non-zero real) number (equivalent to the above)
 4. The *product* of two existing QEs such as Newton · Second
 5. The *sum* of two existing QEs (of the same dimension)

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter
 2. A *multiplication* of a (real) number with an existing QE, such as 5 Meter
 3. A *division* of an existing QE by a (non-zero real) number (equivalent to the above)
 4. The *product* of two existing QEs such as Newton · Second
 5. The *sum* of two existing QEs (of the same dimension)
 6. The *quotient* of two existing QEs such as $1 \frac{\text{Meter}}{\text{Second}}$

Our Approach: The Unit System (4)

- ▶ how can we formalise a quantity expressions?
- ▶ In our model QEs can be one of the following:
 1. (1 times) a *primitive unit*, such as Meter
 2. A *multiplication* of a (real) number with an existing QE, such as 5 Meter
 3. A *division* of an existing QE by a (non-zero real) number (equivalent to the above)
 4. The *product* of two existing QEs such as Newton · Second
 5. The *sum* of two existing QEs (of the same dimension)
 6. The *quotient* of two existing QEs such as $1 \frac{\text{Meter}}{\text{Second}}$
- ▶ this results in the following *Theory of Quantity Expressions*:

Our Approach: The Unit System (5)

Quantity Expression	
import Dimension	
QE	: $\text{dim} \rightarrow \text{type}$
QENMul	: $\forall x : \text{dim}. \mathbb{R} \rightarrow \text{QE}(x) \rightarrow \text{QE}(x)$
QENDiv	: $\forall x : \text{dim}. \text{QE}(x) \rightarrow \mathbb{R} \rightarrow \text{QE}(x)$
QEAdd	: $\forall x : \text{dim}. \text{QE}(x) \rightarrow \text{QE}(x) \rightarrow \text{QE}(x)$
QEMul	: $\forall x, y : \text{dim}. \text{QE}(x) \rightarrow \text{QE}(y) \rightarrow \text{QE}(x \cdot y)$
QEDiv	: $\forall x, y : \text{dim}. \text{QE}(x) \rightarrow \text{QE}(y) \rightarrow \text{QE}\left(\frac{x}{y}\right)$

Our Approach: The Unit System (6)

- ▶ we can now easily create theories that define Units, such as a Meter Theory:

Our Approach: The Unit System (6)

- ▶ we can now easily create theories that define Units, such as a Meter Theory:

Meter
import Quantity Expression
Meter : QE (length)

Our Approach: The Unit System (6)

- ▶ we can now easily create theories that define Units, such as a Meter Theory:

Meter	
import Quantity Expression	
Meter	: QE (length)

- ▶ we can also define some non-metric lengths:

Our Approach: The Unit System (6)

- ▶ we can now easily create theories that define Units, such as a Meter Theory:

Meter	
import Quantity Expression	
Meter	: QE (length)

- ▶ we can also define some non-metric lengths:

Non SI Lengths	
import Quantity Expression	
Thou : QE (length)	
Foot	= QENMul (1000, Thou)
Yard	= QENMul (3, Foot)
Chain	= QENMul (22, Yard)
Furlong	= QENMul (10, Chain)
Mile	= QENMul (8, Furlong)

Our Approach: The Unit System (7)

- ▶ We also need to be able to compare units

Our Approach: The Unit System (7)

- ▶ We also need to be able to compare units
- ▶ We use *Views* (mappings between theories) for this

Our Approach: The Unit System (7)

- ▶ We also need to be able to compare units
- ▶ We use *Views* (mappings between theories) for this
- ▶ In this case we use the view

$$\psi = \{ \text{Thou} \mapsto \text{QENMul}(0.0000254, \text{Meter}) \}$$

Our Approach: The Unit System (7)

- ▶ We also need to be able to compare units
- ▶ We use *Views* (mappings between theories) for this
- ▶ In this case we use the view

$$\psi = \{ \text{Thou} \mapsto \text{QENMul}(0.0000254, \text{Meter}) \}$$

- ▶ This allows us to convert quantity expression representations between different units

Our Approach: The Unit System (7)

- ▶ We also need to be able to compare units
- ▶ We use *Views* (mappings between theories) for this
- ▶ In this case we use the view

$$\psi = \{ \text{Thou} \mapsto \text{QENMul}(0.0000254, \text{Meter}) \}$$

- ▶ This allows us to convert quantity expression representations between different units
- ▶ We have a big (easily extensible) theory graph of unit theories

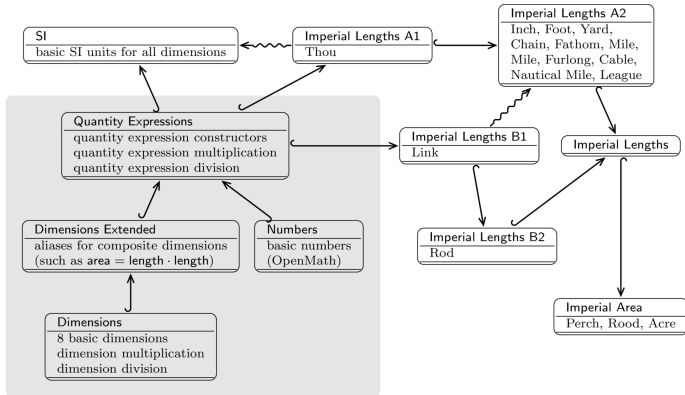
Our Approach: The Unit System (7)

- ▶ We also need to be able to compare units
- ▶ We use *Views* (mappings between theories) for this
- ▶ In this case we use the view

$$\psi = \{ \text{Thou} \mapsto \text{QENMul}(0.0000254, \text{Meter}) \}$$

- ▶ This allows us to convert quantity expression representations between different units
- ▶ We have a big (easily extensible) theory graph of unit theories
- ▶ Here is a small part of it

Our Approach: The Unit System (6)



Our Approach: The Search Algorithm (1)

- ▶ Let us build a search algorithm

Our Approach: The Search Algorithm (1)

- ▶ Let us build a search algorithm
- ▶ We are given a *query* and want to find all equivalent quantity expressions

Our Approach: The Search Algorithm (1)

- ▶ Let us build a search algorithm
- ▶ We are given a *query* and want to find all equivalent quantity expressions
- ▶ We already have a list of quantity expressions from the spotter

Our Approach: The Search Algorithm (1)

- ▶ Let us build a search algorithm
- ▶ We are given a *query* and want to find all equivalent quantity expressions
- ▶ We already have a list of quantity expressions from the spotter
- ▶ We just need an efficient way to check if two QEs are the same

Our Approach: The Search Algorithm (1)

- ▶ Let us build a search algorithm
- ▶ We are given a *query* and want to find all equivalent quantity expressions
- ▶ We already have a list of quantity expressions from the spotter
- ▶ We just need an efficient way to check if two QEs are the same
- ▶ We can just bring QEs to a normal form

Our Approach: The Search Algorithm (1)

- ▶ Let us build a search algorithm
- ▶ We are given a *query* and want to find all equivalent quantity expressions
- ▶ We already have a list of quantity expressions from the spotter
- ▶ We just need an efficient way to check if two QEs are the same
- ▶ We can just bring QEs to a normal form
- ▶ and then have an efficient index

Our Approach: The Search Algorithm (2)

- ▶ A normal form should consist of two components:

Our Approach: The Search Algorithm (2)

- ▶ A normal form should consist of two components:
- ▶ A *scalar* component and a (scalar-free) *unit* component

Our Approach: The Search Algorithm (2)

- ▶ A normal form should consist of two components:
- ▶ A *scalar* component and a (scalar-free) *unit* component
- ▶ The *unit* component should be in standard units (in our case SI)

Our Approach: The Search Algorithm (2)

- ▶ A normal form should consist of two components:
- ▶ A *scalar* component and a (scalar-free) *unit* component
- ▶ The *unit* component should be in standard units (in our case SI)
- ▶ For this we use 2-step-normalisation

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard)

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = \dots =
10 (22 (3 (12 (1000 (0.0000254 Meter))))))
- ▶ Fortnight = (2 (7 (24 (60 (60 Second)))))

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = \dots =
10 (22 (3 (12 (1000 (0.0000254 Meter))))))
- ▶ Fortnight = (2 (7 (24 (60 (60 Second)))))
- ▶ Substitute this back into the original expression

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = $\dots = 10 (22 (3 (12 (1000 (0.0000254 \text{ Meter}))))))$
- ▶ Fortnight = $(2 (7 (24 (60 (60 \text{ Second}))))))$
- ▶ Substitute this back into the original expression
- ▶ $42 \frac{10(22(3(1000(0.0000254 \text{ Meter}))))}{2(7(24(60(60 \text{ Second}))))}$

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = ... =
10 (22 (3 (12 (1000 (0.0000254 Meter))))))
- ▶ Fortnight = (2 (7 (24 (60 (60 Second)))))
- ▶ Substitute this back into the original expression
- ▶ $42 \frac{10(22(3(1000(0.0000254 \text{ Meter}))))}{2(7(24(60(60 \text{ Second}))))}$
- ▶ Then extract the *scalar* component and compute it:

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = ... =
10 (22 (3 (12 (1000 (0.0000254 Meter))))))
- ▶ Fortnight = (2 (7 (24 (60 (60 Second)))))
- ▶ Substitute this back into the original expression
- ▶ $42 \frac{10(22(3(1000(0.0000254 \text{ Meter}))))}{2(7(24(60(60 \text{ Second}))))}$
- ▶ Then extract the *scalar* component and compute it:
- ▶ $42 \frac{10(22(3(12(1000(0.0000254))))))}{2(7(24(60(60))))} = 0.006985$

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = ... =
10 (22 (3 (12 (1000 (0.0000254 Meter))))))
- ▶ Fortnight = (2 (7 (24 (60 (60 Second)))))
- ▶ Substitute this back into the original expression
- ▶ $42 \frac{10(22(3(1000(0.0000254 \text{ Meter}))))}{2(7(24(60(60 \text{ Second}))))}$
- ▶ Then extract the *scalar* component and compute it:
- ▶ $42 \frac{10(22(3(12(1000(0.0000254))))))}{2(7(24(60(60))))} = 0.006985$
- ▶ Continue by extracting the *unit* component: $\frac{\text{Meter}}{\text{Second}}$

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = $\dots = 10 (22 (3 (12 (1000 (0.0000254 \text{ Meter}))))))$
- ▶ Fortnight = $(2 (7 (24 (60 (60 \text{ Second}))))))$
- ▶ Substitute this back into the original expression
- ▶ $42 \frac{10(22(3(1000(0.0000254 \text{ Meter}))))}{2(7(24(60(60 \text{ Second}))))}$
- ▶ Then extract the *scalar* component and compute it:
- ▶ $42 \frac{10(22(3(12(1000(0.0000254))))))}{2(7(24(60(60))))} = 0.006985$
- ▶ Continue by extracting the *unit* component: $\frac{\text{Meter}}{\text{Second}}$
- ▶ Finally multiply these components to get the standard form:

Our Approach: The Search Algorithm (3)

- ▶ Example: Normalise $42 \frac{\text{Furlong}}{\text{Fortnight}}$
- ▶ First, turn all units into standard units by finding an appropriate path in the *theory graph* of units
- ▶ Furlong = 10 Chain = 10 (22 Yard) = $\dots = 10 (22 (3 (12 (1000 (0.0000254 \text{ Meter}))))))$
- ▶ Fortnight = $(2 (7 (24 (60 (60 \text{ Second}))))))$
- ▶ Substitute this back into the original expression
- ▶ $42 \frac{10(22(3(1000(0.0000254 \text{ Meter}))))}{2(7(24(60(60 \text{ Second}))))}$
- ▶ Then extract the *scalar* component and compute it:
- ▶ $42 \frac{10(22(3(12(1000(0.0000254))))))}{2(7(24(60(60))))} = 0.006985$
- ▶ Continue by extracting the *unit* component: $\frac{\text{Meter}}{\text{Second}}$
- ▶ Finally multiply these components to get the standard form:
- ▶ $0.006985 \frac{\text{Meter}}{\text{Second}}$

Our Approach: The Search Algorithm (4)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms

Our Approach: The Search Algorithm (4)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit

Our Approach: The Search Algorithm (4)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter

Our Approach: The Search Algorithm (4)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter
- ▶ Important: We choose SI units for the standard form

Our Approach: The Search Algorithm (4)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter
- ▶ Important: We choose SI units for the standard form
- ▶ This is not required. We could also choose to use imperial units as a normal form

Our Approach: The Search Algorithm (4)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter
- ▶ Important: We choose SI units for the standard form
- ▶ This is not required. We could also choose to use imperial units as a normal form
- ▶ Our unit graph is flexible enough so that we can actually do that

Our Approach: The Search Algorithm (5)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms

Our Approach: The Search Algorithm (5)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit

Our Approach: The Search Algorithm (5)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter

Our Approach: The Search Algorithm (5)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter
- ▶ Important: We choose SI units for the standard form

Our Approach: The Search Algorithm (5)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter
- ▶ Important: We choose SI units for the standard form
- ▶ This is not required. We could also choose to use imperial units as a normal form

Our Approach: The Search Algorithm (5)

- ▶ We can then compare the spotted QEs with the query via directly comparing normal forms
- ▶ To save time, we *cache* the normal form of each unit
- ▶ We can also *cache* the normal forms of the QEs found by the spotter
- ▶ Important: We choose SI units for the standard form
- ▶ This is not required. We could also choose to use imperial units as a normal form
- ▶ Our unit graph is flexible enough so that we can actually do that

Conclusion: The Implementation

- ▶ We have the following implemented components:

Conclusion: The Implementation

- ▶ We have the following implemented components:
- ▶ The frontend, written in HTML, CSS, JavaScript

Conclusion: The Implementation

- ▶ We have the following implemented components:
- ▶ The frontend, written in HTML, CSS, JavaScript
- ▶ when queried, this sends a QE to the backend

Conclusion: The Implementation

- ▶ We have the following implemented components:
- ▶ The frontend, written in HTML, CSS, JavaScript
- ▶ when queried, this sends a QE to the backend
- ▶ The backend, written in scala, uses MMT to normalise this expression

Conclusion: The Implementation

- ▶ We have the following implemented components:
- ▶ The frontend, written in HTML, CSS, JavaScript
- ▶ when queried, this sends a QE to the backend
- ▶ The backend, written in scala, uses MMT to normalise this expression
- ▶ and then searches the harvests (provided by Stiv's Spotter)

Conclusion: The Implementation

- ▶ We have the following implemented components:
- ▶ The frontend, written in HTML, CSS, JavaScript
- ▶ when queried, this sends a QE to the backend
- ▶ The backend, written in scala, uses MMT to normalise this expression
- ▶ and then searches the harvests (provided by Stiv's Spotter)
- ▶ Finally the frontend displays a list of results

Conclusion: The Implementation (2)

- ▶ The unit Graph are extensible

Conclusion: The Implementation (2)

- ▶ The unit Graph are extensible
- ▶ The user does not need to think about what units are inside the documents

Conclusion: The Implementation (2)

- ▶ The unit Graph are extensible
- ▶ The user does not need to think about what units are inside the documents
- ▶ Time for a demo (if there is still time)

Thank You For Listening!

Image sources:

- ▶ http://www.gettingaroundgermany.info/g_imgs/z274.gif
- ▶ http://upload.wikimedia.org/wikipedia/commons/thumb/1/19/Mars_Climate_Orbiter_2.jpg/528px-Mars_Climate_Orbiter_2.jpg