# Understanding Expression Simplification

Jacques Carette
McMaster University

July 5, 2004

# Motivation

What is simpler?

| | |
|---|---|
| 0 | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |

# Motivation

What is simpler?

| | |
|---|---|
| $0$ | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |
| $1+\sqrt{2}$ | $\sqrt[3]{7+5\sqrt{2}}$ |

# Motivation

What is simpler?

| | |
|---|---|
| $0$ | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |
| $1+\sqrt{2}$ | $\sqrt[3]{7+5\sqrt{2}}$ |
| $9$ | $4+2+1+1+1$ |

# Motivation

What is simpler?

| | |
|---|---|
| $0$ | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |
| $1+\sqrt{2}$ | $\sqrt[3]{7+5\sqrt{2}}$ |
| $9$ | $4+2+1+1+1$ |
| $2^{2^{20}} - 1$ | $\ldots$ |

# Motivation

What is simpler?

| | |
|---|---|
| $0$ | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |
| $1+\sqrt{2}$ | $\sqrt[3]{7+5\sqrt{2}}$ |
| $9$ | $4+2+1+1+1$ |
| $2^{2^{20}} - 1$ | $\ldots$ |
| $\texttt{ChebyshevT}(10000, x)$ | $\ldots$ |

# Motivation

What is simpler?

| | |
|---|---|
| $0$ | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |
| $1+\sqrt{2}$ | $\sqrt[3]{7+5\sqrt{2}}$ |
| $9$ | $4+2+1+1+1$ |
| $2^{2^{20}}-1$ | $\ldots$ |
| $\texttt{ChebyshevT}(10000, x)$ | $\ldots$ |
| $\texttt{ChebyshevT}(5, x)$ | $16x^5 - 20x^3 + 5x$ |

# Motivation

What is simpler?

| | |
|---|---|
| $0$ | $(x+3)^3 - x^3 - 9x^2 - 27x - 27$ |
| $1+\sqrt{2}$ | $\sqrt[3]{7+5\sqrt{2}}$ |
| $9$ | $4+2+1+1+1$ |
| $2^{2^{20}}-1$ | $\ldots$ |
| $\texttt{ChebyshevT}(10000, x)$ | $\ldots$ |
| $\texttt{ChebyshevT}(5, x)$ | $16x^5 - 20x^3 + 5x$ |
| $1$ | $\frac{x-1}{x-1}$ |

## Informal Definition

**Definition 1** *An expression* $A$ *is* simpler *than an expression* $B$ *if*

- *in all contexts where* $A$ *and* $B$ *can be used, they mean the same thing, and*

- *the length of the description of* $A$ *is shorter than the length of the description of* $B$*.*

## Informal Definition

**Definition 1** *An expression $A$ is* simpler *than an expression $B$ if*

- *in all contexts where $A$ and $B$ can be used, they mean the same thing, and*
- *the length of the description of $A$ is shorter than the length of the description of $B$.*

## Informal Definition

**Definition 1** *An expression $A$ is* simpler *than an expression $B$ if*

- *in all contexts where $A$ and $B$ can be used, they mean the same thing, and*

- *the length of the description of $A$ is shorter than the length of the description of $B$.*

## Informal Definition

**Definition 1** *An expression* $A$ *is* simpler *than an expression* $B$ *if*

- *in all contexts where* $A$ *and* $B$ *can be used, they mean the same thing, and*

- *the length of the description of* $A$ *is shorter than the length of the description of* $B$*.*

# Kolmogorov Complexity

Basic idea: the complexity of a string is the length of the shortest Turing machine which writes exactly that string on output and halts.

# Kolmogorov Complexity

Basic idea: the complexity of a string is the length of the shortest Turing machine which writes exactly that string on output and halts.

**Definition 2** *For a given universal Turing machine* $\phi$*, the* complexity $C_\phi$ *of* $x$ conditional *to* $y$ *is defined by*

$$C_\phi(x|y) = \min\{\text{length}(p) : \phi(\langle y, p\rangle) = x\},$$

# Kolmogorov Complexity

Basic idea: the complexity of a string is the length of the shortest Turing machine which writes exactly that string on output and halts.

**Definition 2** *For a given universal Turing machine $\phi$, the* complexity $C_\phi$ *of $x$* conditional *to $y$ is defined by*

$$C_\phi(x|y) = \min\{\text{length}(p) : \phi(\langle y, p\rangle) = x\},$$

This is universal in the following exact sense:

**Theorem 1 (Kolmogorov)** *For all universal Turing machines $\psi$ and $\phi$,*

$$|C_\psi(y|x) - C_\phi(y|x)| \leq c_{\psi,\phi}$$

# Kolmogorov Complexity

We fix a universal Turing machine $\phi$, and then

**Definition 3** *The (unconditional) Kolmogorov Complexity* $C(x)$ *of a string* $x$ *is* $\phi(x|\epsilon)$.

# Kolmogorov Complexity

We fix a universal Turing machine $\phi$, and then

**Definition 3** *The (unconditional) Kolmogorov Complexity* $C(x)$ *of a string* $x$ *is* $\phi(x|\epsilon)$*.*

Problems:

- $C(x)$ is not computable!

# Kolmogorov Complexity

We fix a universal Turing machine $\phi$, and then

**Definition 3** *The (unconditional) Kolmogorov Complexity* $C(x)$ *of a string* $x$ *is* $\phi(x|\epsilon)$.

Problems:

- $C(x)$ is not computable! But it is approximable

# Kolmogorov Complexity

We fix a universal Turing machine $\phi$, and then

**Definition 3** *The (unconditional) Kolmogorov Complexity* $C(x)$ *of a string* $x$ *is* $\phi(x|\epsilon)$*.*

Problems:

- $C(x)$ is not computable! But it is approximable
- KC is an asymptotic theory

# Minimum Description Length (MDL)

KC can be rewritten as

$$C(x) = \min_{p,y \,\mid\, \phi(p)(y)=x} \text{length}(p) + \text{length}(y)$$

where $\phi$ is a universal interpreter, $p$ is a program and $y$ is a binary string. $p$ is a model for the regularities in $x$.

## Minimum Description Length (MDL)

KC can be rewritten as

$$C(x) = \min_{p,y \,\mid\, \phi(p)(y)=x} \text{length}(p) + \text{length}(y)$$

where $\phi$ is a universal interpreter, $p$ is a program and $y$ is a binary string. $p$ is a model for the regularities in $x$.

Instead of minimizing over all programs (models), MDL fixes an effectively enumerable class of models $\mathcal{M}$ over which to minimize.

## Biform Theories

- A **biform theory** is a triple $T = (\mathbf{K}, L, \Gamma)$ where:
  - $\mathbf{K}$ is an admissible background logic
  - $L$ is a language of $\mathbf{K}$
  - $\Gamma$ is a set of formuloids of $L$ called the **axiomoids** of $T$

- The axiomoids are used to specify:
  - The basic objects and concepts of $T$
  - The basic deduction and computation rules of $T$

- $T$ can be viewed as being simultaneously an **algorithmic theory** and an **axiomatic theory**. Many more details in the paper and references.

# Formuloids

- A **formuloid** is a pair $\theta = (\Pi, M)$ where:
  - $\Pi$ is a transformer from $L$ to $L$
  - $M$ is a function that maps each $E \in \mathrm{dom}(\Pi)$ to a formula of $L$

- $M$ is intended to give the **meaning** of applying $\Pi$ to an expression $E$
  - For many formuloids, $M(E)$ is $E = \Pi(E)$
  - The **span** of $\theta$ is: $\{M(E) \mid E \in \mathrm{dom}(\Pi)\}$

- The **algorithmic meaning** of $\theta$ is its transformer

- The **axiomatic meaning** of $\theta$ is its span

## Biform Theories for simplification

A biform theory $T$ suitable for simplification satisfies:

- the language $L$ contains the syntactic representation of a Turing complete programming language,

## Biform Theories for simplification

A biform theory $T$ suitable for simplification satisfies:

- the language $L$ contains the syntactic representation of a Turing complete programming language,
- there exists a total length function length : $L \rightarrow \mathbb{N}$ compatible with the subexpression relation,

## Biform Theories for simplification

A biform theory $T$ suitable for simplification satisfies:

- the language $L$ contains the syntactic representation of a Turing complete programming language,
- there exists a total length function length : $L \longrightarrow \mathbb{N}$ compatible with the subexpression relation,
- all formuloids $\theta = (\Pi, M)$ are such that the function $M$ is expressible in $L$,

## Biform Theories for simplification

A biform theory $T$ suitable for simplification satisfies:

- the language $L$ contains the syntactic representation of a Turing complete programming language,
- there exists a total length function length : $L \longrightarrow \mathbb{N}$ compatible with the subexpression relation,
- all formuloids $\theta = (\Pi, M)$ are such that the function $M$ is expressible in $L$,
- $\Gamma$ is finite, and the domains of the axiomoids of $\Gamma$ are finitely representable,

# Biform Theories for simplification

A biform theory $T$ suitable for simplification satisfies:

- the language $L$ contains the syntactic representation of a Turing complete programming language,
- there exists a total length function length : $L \longrightarrow \mathbb{N}$ compatible with the subexpression relation,
- all formuloids $\theta = (\Pi, M)$ are such that the function $M$ is expressible in $L$,
- $\Gamma$ is finite, and the domains of the axiomoids of $\Gamma$ are finitely representable,
- $\Gamma$ always contains at least the axiomoid corresponding to the identity transformer,

## Biform Theories for simplification

A biform theory $T$ suitable for simplification satisfies:

- the language $L$ contains the syntactic representation of a Turing complete programming language,
- there exists a total length function length : $L \rightarrow \mathbb{N}$ compatible with the subexpression relation,
- all formuloids $\theta = (\Pi, M)$ are such that the function $M$ is expressible in $L$,
- $\Gamma$ is finite, and the domains of the axiomoids of $\Gamma$ are finitely representable,
- $\Gamma$ always contains at least the axiomoid corresponding to the identity transformer,
- we are given an equivalence relation $\sim$ on $L$, which is interpreted as a means the same thing as relation.

Such theories are called reflexive.

# Length in context

Consider a lattice $\mathfrak{T}$ of reflexive theories, where meet and join are given by intersection and union of sets of axiomoids, with the additional restriction that if $\Gamma_i \subseteq \Gamma_j$ then $T_j$ must be a conservative extension of $T_i$.

# Length in context

Consider a lattice $\mathfrak{T}$ of reflexive theories, where meet and join are given by intersection and union of sets of axiomoids, with the additional restriction that if $\Gamma_i \subseteq \Gamma_j$ then $T_j$ must be a conservative extension of $T_i$.

Given an expression $e$ of $L$, let theory($e$) be the smallest theory $T_i$ of $\mathfrak{T}$ such that $e = e$ is a theorem of $T_i$. This is not trivial – $\frac{1}{0} = \frac{1}{0}$ is usually not a theorem!

# Length in context

Consider a lattice $\mathfrak{T}$ of reflexive theories, where meet and join are given by intersection and union of sets of axiomoids, with the additional restriction that if $\Gamma_i \subseteq \Gamma_j$ then $T_j$ must be a conservative extension of $T_i$.

Given an expression $e$ of $L$, let theory($e$) be the smallest theory $T_i$ of $\mathfrak{T}$ such that $e = e$ is a theorem of $T_i$. This is not trivial – $\frac{1}{0} = \frac{1}{0}$ is usually not a theorem!

Given a transformer $\Theta = (\Pi, M)$ such that $e \sim M(e)$, theory($e, \Pi$) is the smallest theory such that $e = \Pi(e)$ is a theorem.

# Length in context

Consider a lattice $\mathfrak{T}$ of reflexive theories, where meet and join are given by intersection and union of sets of axiomoids, with the additional restriction that if $\Gamma_i \subseteq \Gamma_j$ then $T_j$ must be a conservative extension of $T_i$.

Given an expression $e$ of $L$, let theory($e$) be the smallest theory $T_i$ of $\mathfrak{T}$ such that $e = e$ is a theorem of $T_i$. This is not trivial – $\frac{1}{0} = \frac{1}{0}$ is usually not a theorem!

Given a transformer $\Theta = (\Pi, M)$ such that $e \sim M(e)$, theory($e, \Pi$) is the smallest theory such that $e = \Pi(e)$ is a theorem.

$$\text{length}_{\mathfrak{T}}(e) = \text{length}(e) + \text{length}(\text{theory}(e))$$

# Simplifying

Given a reflexive theory lattice $\mathfrak{T}$ and a recursively enumerable sequence of transformers $\langle \Pi_1, \Pi_2, \ldots \rangle$ which are all known to preserve equivalence, then simplify$(e) = e_i$ such that length$_{\mathfrak{T}}(e_i)$ is minimal amongst all $e_j = \Pi_j(e)$.

# Simplifying

Given a reflexive theory lattice $\mathfrak{T}$ and a recursively enumerable sequence of transformers $\langle \Pi_1, \Pi_2, \ldots \rangle$ which are all known to preserve equivalence, then simplify$(e) = e_i$ such that length$_{\mathfrak{T}}(e_i)$ is minimal amongst all $e_j = \Pi_j(e)$.

Intuitively, this means that an expression is considered short only if it itself is relatively short, but that also the description of the theory behind that expression is also short.

# Simplifying

Given a reflexive theory lattice $\mathfrak{T}$ and a recursively enumerable sequence of transformers $\langle \Pi_1, \Pi_2, \ldots \rangle$ which are all known to preserve equivalence, then simplify$(e) = e_i$ such that length$_{\mathfrak{T}}(e_i)$ is minimal amongst all $e_j = \Pi_j(e)$.

Intuitively, this means that an expression is considered short only if it itself is relatively short, but that also the description of the theory behind that expression is also short.

More concretely, this says that if your base theory is that of expanded polynomials, but you also have a theory of terminating hypergeometrics built on top of that, then you will prefer expanded polynomials for "small" degrees, and eventually will prefer to see a hypergeometric expression instead.

## Examples and magic numbers

**Example 1** *When is the expression* $2^n$ *(for* $n > 0$ *integer) simpler than the integer* $2^n$*?*

## Examples and magic numbers

**Example 1** *When is the expression* $2^n$ *(for* $n > 0$ *integer) simpler than the integer* $2^n$*?*

Using a self-delimiting binary encoding, then in the *same theory*, whenever $n > 7$.

# Examples and magic numbers

**Example 1** *When is the expression* $2^n$ *(for* $n > 0$ *integer) simpler than the integer* $2^n$*?*

Using a self-delimiting binary encoding, then in the *same theory*, whenever $n > 7$.

**Example 2** *When is the expression* ChebyshevT$(n, x)$ *simpler than the corresponding expanded polynomial, where* $\mathfrak{T}$ *consists of the theory* $T_1$ *of expanded polynomials and the conservative extension* $T_2$ *defining Chebyshev polynomials?*

# Examples and magic numbers

**Example 1** *When is the expression* $2^n$ *(for* $n > 0$ *integer) simpler than the integer* $2^n$*?*
Using a self-delimiting binary encoding, then in the *same theory*, whenever $n > 7$.

**Example 2** *When is the expression* ChebyshevT$(n, x)$ *simpler than the corresponding expanded polynomial, where* $\mathfrak{T}$ *consists of the theory* $T_1$ *of expanded polynomials and the conservative extension* $T_2$ *defining Chebyshev polynomials?*
Using a self-delimiting binary encoding, then whenever $n > C$ where

$$C = \frac{1}{2}\sqrt{2b}a\sqrt{W_{-1}(\frac{2a}{b}\exp -2c/b)}, \tag{1}$$

and $a$ depends on the encoding of constants in $T_1$, $b$ on the difference of encoding constants in $T_1$ and $T_2$, and $c$ is essentially $\mathrm{length}_{\mathfrak{T}}(T_2) - \mathrm{length}_{\mathfrak{T}}(T_1)$.

# Implementation

This is not quite how it is done in Maple. . .

## Implementation

This is not quite how it is done in Maple. . . but it is not incompatible!

## Implementation

This is not quite how it is done in Maple. . . but it is not incompatible!

compSeq, constants, infinity, @@, @, limit, Limit, max, min, polar, conjugate, D, diff, Diff, int, Int, sum, Sum, product, Product, RootOf, hypergeom, pochhammer, Si, Ci, LerchPhi, Ei, erf, erfc, LambertW, BesselJ, BesselY, BesselK, BesselI, polylog, dilog, GAMMA, WhittakerM, WhittakerW, LegendreP, LegendreQ, InverseJacobi, Jacobi, JacobiTheta, JacobiZeta, Weierstrass, trig, arctrig, ln, radical, sqrt, power, exp, Dirac, Heaviside, piecewise, abs, csgn, signum, rtable, constant

## Further work

Two conjectures:

- LLL is a global MDL minimizer

## Further work

Two conjectures:

- `LLL` is a global MDL minimizer
- `PSQL` and `Hermite-Pade` are local MDL minimizers.