Encoding Challenges for Hard Problems

Marijn J.H. Heule

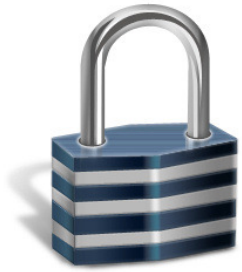
THE UNIVERSITY OF TEXAS AT AUSTIN

Starting at Carnegie Mellon University in August

Matryoshka workshop June 12, 2019

Automated Reasoning Has Many Applications

formal verification

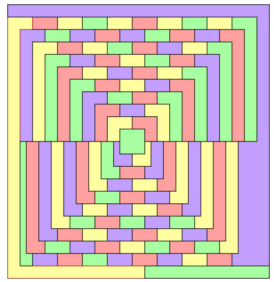


security

bioinformatics

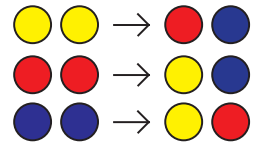
planning and scheduling

train safety



automated theorem proving

exploit generation



term rewriting termination

encode

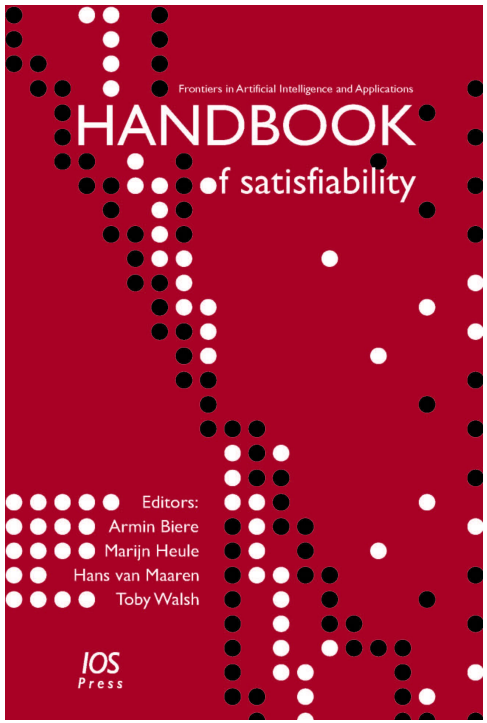
automated reasoner

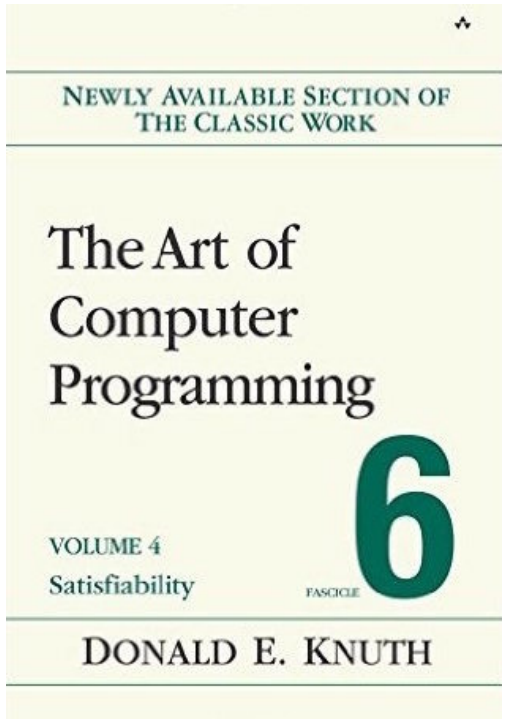
decode

Breakthrough in SAT Solving in the Last 20 Years

Satisfiability (SAT) problem: Can a Boolean formula be satisfied?

mid '90s: formulas solvable with thousands of variables and clauses
now: formulas solvable with millions of variables and clauses





Edmund Clarke: *"a key technology of the 21st century"* [Biere, Heule, vanMaaren, and Walsh '09]

Donald Knuth: *"evidently a killer app, because it is key to the solution of so many other problems"* [Knuth '15]

Representations

Matrix Multiplication

The Collatz Conjecture

Representations

Matrix Multiplication

The Collatz Conjecture

The Right Representation is Crucial

What makes a problem hard?

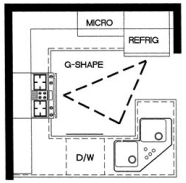
New angle: does the representation enable efficient reasoning?

The famous pigeonhole principle: Can $n+1$ pigeons be placed in n holes such that no hole contains multiple pigeons?

- Hard for many automated reasoning approaches
- Easy for a little kid given the right representation

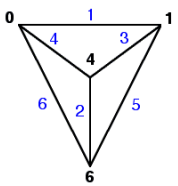
source: pecanpartnership.co.uk/2016/01/05/beware-pigeon-hole-overcoming-stereotypes-build-collaborative-culture

Artisan Representations (joint work)

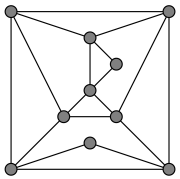


Architectural 3D Layout
[VSMM '07]
Henriette Bier

Edge-matching Puzzles
[LaSh '08]

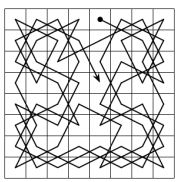


Graceful Graphs
[AAAI '10]
Toby Walsh

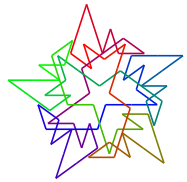


Clique-Width
[SAT '13, TOCL '15]
Stefan Szeider

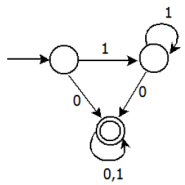
Firewall Verification
[SSS '16]
Mohamed Gouda



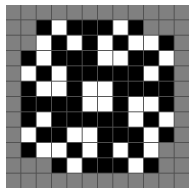
Open Knight Tours
Moshe Vardi



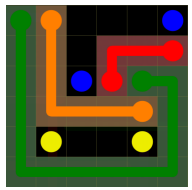
Van der Waerden numbers
[EJoC '07]



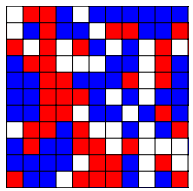
Software Model Synthesis
[ICGI '10, ESE '13]
Sicco Verwer



Conway's Game of Life
[EJoC '13]
Willem van der Poel



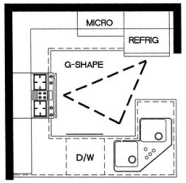
Connect the Pairs
Donald Knuth



Pythagorean Triples
[SAT '16, CACM '17]
Victor Marek

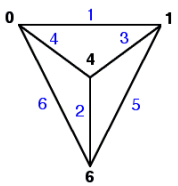
Collatz conjecture [Open]
Scott Aaronson

Artisan Representations (joint work)

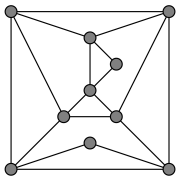


Architectural 3D Layout
[VSMM '07]
Henriette Bier

Edge-matching Puzzles
[LaSh '08]

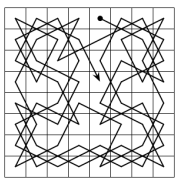


Graceful Graphs
[AAAI '10]
Toby Walsh

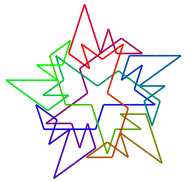


Clique-Width
[SAT '13, TOCL '15]
Stefan Szeider

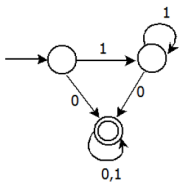
Firewall Verification
[SSS '16]
Mohamed Gouda



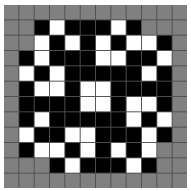
Open Knight Tours
Moshe Vardi



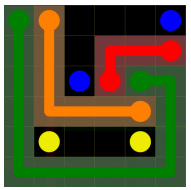
Van der Waerden numbers
[EJoC '07]



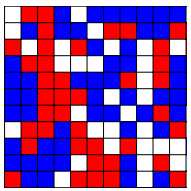
Software Model Synthesis
[ICGI '10, ESE '13]
Sicco Verwer



Conway's Game of Life
[EJoC '13]
Willem van der Poel



Connect the Pairs
Donald Knuth

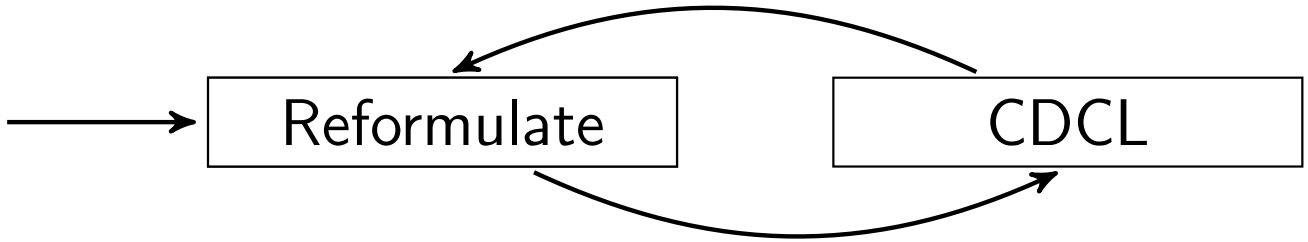


Pythagorean Triples
[SAT '16, CACM '17]
Victor Marek

Collatz conjecture [Open]
Scott Aaronson

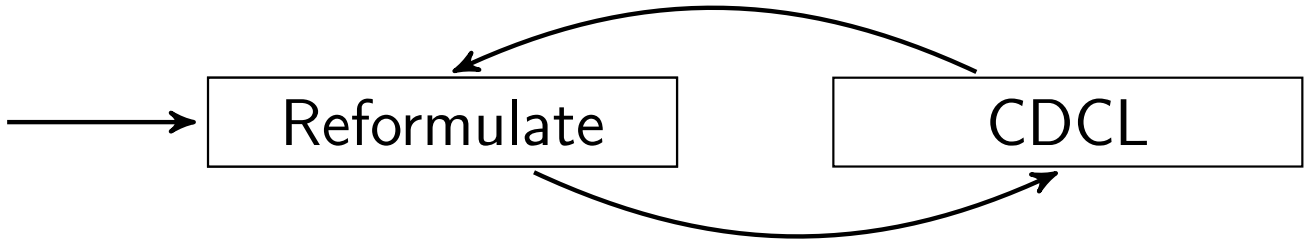
Inprocessing [Järvisalo, Heule, and Biere '12]

How to fix a poor representation fully automatically?



Inprocessing [Järvisalo, Heule, and Biere '12]

How to fix a poor representation fully automatically?



Example: Bounded Variable Addition [Manthey, Heule, and Biere '12]

Replace $(a \vee d)$ $(a \vee e)$
$(b \vee d)$ $(b \vee e)$
$(c \vee d)$ $(c \vee e)$

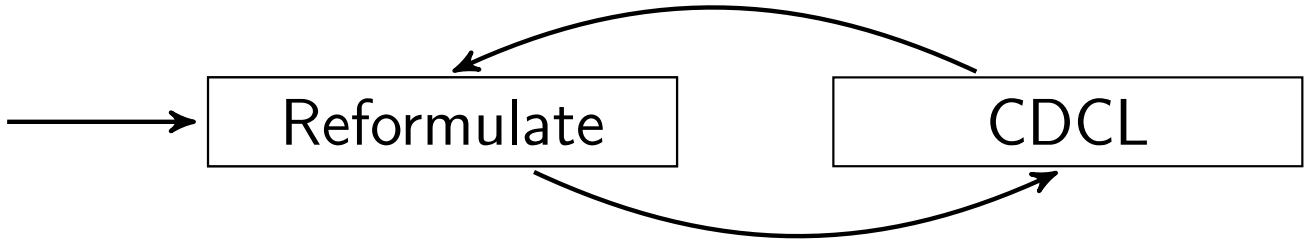
by $(\overline{x} \vee a)$ $(\overline{x} \vee b)$ $(\overline{x} \vee c)$
$(x \vee d)$ $(x \vee e)$

adds 1 variable *removes 1 clause*

This technique is crucial for hard bioinformatics problems and turns the naive encoding of AtMostOne into the optimal one.

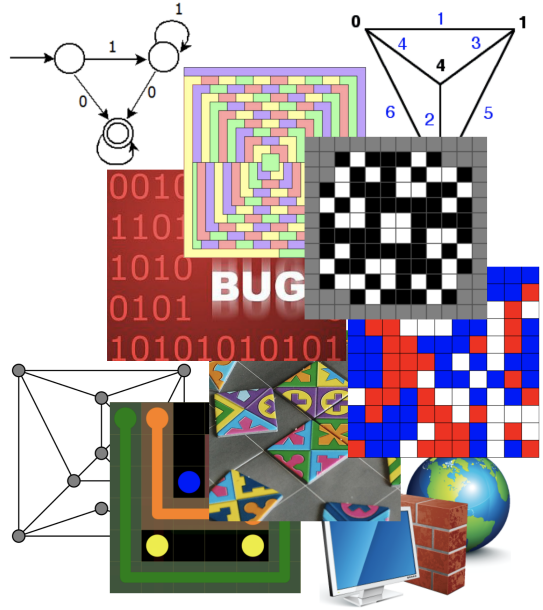
Inprocessing [Järvisalo, Heule, and Biere '12]

How to fix a poor representation fully automatically?



Example: Bounded Variable Addition [Manthey, Heule, and Biere '12]

Replace $(a \vee d)$ $(a \vee e)$
$(b \vee d)$ $(b \vee e)$
$(c \vee d)$ $(c \vee e)$



by $(\overline{x} \vee a)$ $(\overline{x} \vee b)$ $(\overline{x} \vee c)$
$(x \vee d)$ $(x \vee e)$

adds 1 variable *removes 1 clause*

This technique is crucial for hard bioinformatics problems and turns the naive encoding of AtMostOne into the optimal one.

Matrix Multiplication

joint work with Manuel Kauers and Martina Seidl

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

$$\begin{aligned} c_{1,1} &= a_{1,1} \cdot b_{1,1} + a_{1,2} \cdot b_{2,1} \\ c_{1,2} &= a_{1,1} \cdot b_{1,2} + a_{1,2} \cdot b_{2,2} \\ c_{2,1} &= a_{2,1} \cdot b_{1,1} + a_{2,2} \cdot b_{2,1} \\ c_{2,2} &= a_{2,1} \cdot b_{1,2} + a_{2,2} \cdot b_{2,2} \end{aligned}$$

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

$$\begin{aligned} c_{1,1} &= M_1 + M_4 - M_5 + M_7 \\ c_{1,2} &= M_3 + M_5 \\ c_{2,1} &= M_2 + M_4 \\ c_{2,2} &= M_1 - M_2 + M_3 + M_6 \end{aligned}$$

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

. . . where

$$\begin{aligned} M_1 &= (a_{1,1} + a_{2,2}) \cdot (b_{1,1} + b_{2,2}) \\ M_2 &= (a_{2,1} + a_{2,2}) \cdot b_{1,1} \\ M_3 &= a_{1,1} \cdot (b_{1,2} - b_{2,2}) \\ M_4 &= a_{2,2} \cdot (b_{2,1} - b_{1,1}) \\ M_5 &= (a_{1,1} + a_{1,2}) \cdot b_{2,2} \\ M_6 &= (a_{2,1} - a_{1,1}) \cdot (b_{1,1} + b_{1,2}) \\ M_7 &= (a_{1,2} - a_{2,2}) \cdot (b_{2,1} + b_{2,2}) \end{aligned}$$

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

- This scheme needs 7 multiplications instead of 8.

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

- This scheme needs 7 multiplications instead of 8.
- Recursive application allows to multiply $n \times n$ matrices with $\mathcal{O}(n^{\log_2 7})$ operations in the ground ring.

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

- This scheme needs 7 multiplications instead of 8.
- Recursive application allows to multiply $n \times n$ matrices with $\mathcal{O}(n^{\log_2 7})$ operations in the ground ring.
- Let ω be the smallest number so that $n \times n$ matrices can be multiplied using $\mathcal{O}(n^{\omega})$ operations in the ground domain.

Matrix Multiplication: Introduction

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{pmatrix}$$

- This scheme needs 7 multiplications instead of 8.
- Recursive application allows to multiply $n \times n$ matrices with $\mathcal{O}(n^{\log_2 7})$ operations in the ground ring.
- Let ω be the smallest number so that $n \times n$ matrices can be multiplied using $\mathcal{O}(n^{\omega})$ operations in the ground domain.
- Then $2 \leq \omega < 3$. What is the exact value?

Efficient Matrix Multiplication: Theory

- Strassen 1969: $\omega \leq \log_2 7 \leq 2.807$

Efficient Matrix Multiplication: Theory

- Strassen 1969: $\omega \leq \log_2 7 \leq 2.807$
- Pan 1978: $\omega \leq 2.796$
- Bini et al. 1979: $\omega \leq 2.7799$
- Schönhage 1981: $\omega \leq 2.522$
- Romani 1982: $\omega \leq 2.517$
- Coppersmith/Winograd 1981: $\omega \leq 2.496$
- Strassen 1986: $\omega \leq 2.479$
- Coppersmith/Winograd 1990: $\omega \leq 2.376$

Efficient Matrix Multiplication: Theory

- Strassen 1969: $\omega \leq \log_2 7 \leq 2.807$
- Pan 1978: $\omega \leq 2.796$
- Bini et al. 1979: $\omega \leq 2.7799$
- Schönhage 1981: $\omega \leq 2.522$
- Romani 1982: $\omega \leq 2.517$
- Coppersmith/Winograd 1981: $\omega \leq 2.496$
- Strassen 1986: $\omega \leq 2.479$
- Coppersmith/Winograd 1990: $\omega \leq 2.376$
- Stothers 2010: $\omega \leq 2.374$
- Williams 2011: $\omega \leq 2.3728642$
- Le Gall 2014: $\omega \leq 2.3728639$

Efficient Matrix Multiplication: Practice

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.

Efficient Matrix Multiplication: Practice

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.

Efficient Matrix Multiplication: Practice

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.
- Idea: instead of dividing the matrices into 2×2-block matrices, divide them into 3×3-block matrices.

Efficient Matrix Multiplication: Practice

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.
- Idea: instead of dividing the matrices into 2×2-block matrices, divide them into 3×3-block matrices.
- Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

Efficient Matrix Multiplication: Practice

- Only Strassen's algorithm beats the classical algorithm for reasonable problem sizes.
- Want: a matrix multiplication algorithm that beats Strassen's algorithm for matrices of moderate size.
- Idea: instead of dividing the matrices into 2×2-block matrices, divide them into 3×3-block matrices.
- Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?
- Answer: Nobody knows.

The 3x3 Case is Still Open

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

The 3x3 Case is Still Open

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- naive algorithm: 27

The 3x3 Case is Still Open

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- naive algorithm: 27
- padd with zeros, use Strassen twice, cleanup: 25

The 3x3 Case is Still Open

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- naive algorithm: 27
- padd with zeros, use Strassen twice, cleanup: 25
- best known upper bound: 23 (Laderman 1976)

The 3x3 Case is Still Open

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- naive algorithm: 27
- padd with zeros, use Strassen twice, cleanup: 25
- best known upper bound: 23 (Laderman 1976)
- best known lower bound: 19 (Bläser 2003)

The 3x3 Case is Still Open

Question: What's the minimal number of multiplications needed to multiply two 3×3 matrices?

- naive algorithm: 27
- padd with zeros, use Strassen twice, cleanup: 25
- best known upper bound: 23 (Laderman 1976)
- best known lower bound: 19 (Bläser 2003)
- maximal number of multiplications allowed if we want to beat Strassen: 21 (because $\log_3 21 < \log_2 7 < \log_3 22$).

Laderman's scheme from 1976

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,2} & c_{3,3} \end{pmatrix}$$

$$\begin{aligned}
c_{1,1} &= -M_6 + M_{14} + M_{19} \\
c_{2,1} &= M_2 + M_3 + M_4 + M_6 + M_{14} + M_{16} + M_{17} \\
c_{3,1} &= M_6 + M_7 - M_8 + M_{11} + M_{12} + M_{13} - M_{14} \\
c_{1,2} &= M_1 - M_4 + M_5 - M_6 - M_{12} + M_{14} + M_{15} \\
c_{2,2} &= M_2 + M_4 - M_5 + M_6 + M_{20} \\
c_{3,2} &= M_{12} + M_{13} - M_{14} - M_{15} + M_{22} \\
c_{1,3} &= -M_6 - M_7 + M_9 + M_{10} + M_{14} + M_{16} + M_{18} \\
c_{2,3} &= M_{14} + M_{16} + M_{17} + M_{18} + M_{21} \\
c_{3,3} &= M_6 + M_7 - M_8 - M_9 + M_{23}
\end{aligned}$$

Laderman's scheme from 1976

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,2} & c_{3,3} \end{pmatrix}$$

where . . .

$$\begin{aligned}
M_1 &= (-a_{1,1} + a_{1,2} + a_{1,3} - a_{2,1} + a_{2,2} + a_{3,2} + a_{3,3}) \cdot b_{2,2} \\
M_2 &= (a_{1,1} + a_{2,1}) \cdot (b_{1,2} + b_{2,2}) \\
M_3 &= a_{2,2} \cdot (b_{1,1} - b_{1,2} + b_{2,1} - b_{2,2} - b_{2,3} + b_{3,1} - b_{3,3}) \\
M_4 &= (-a_{1,1} - a_{2,1} + a_{2,2}) \cdot (-b_{1,1} + b_{1,2} + b_{2,2}) \\
M_5 &= (-a_{2,1} + a_{2,2}) \cdot (-b_{1,1} + b_{1,2}) \\
M_6 &= -a_{1,1} \cdot b_{1,1} \\
M_7 &= (a_{1,1} + a_{3,1} + a_{3,2}) \cdot (b_{1,1} - b_{1,3} + b_{2,3}) \\
M_8 &= (a_{1,1} + a_{3,1}) \cdot (-b_{1,3} + b_{2,3}) \\
M_9 &= (a_{3,1} + a_{3,2}) \cdot (b_{1,1} - b_{1,3})
\end{aligned}$$

Laderman's scheme from 1976

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,2} & c_{3,3} \end{pmatrix}$$

where . . .

$$\begin{aligned}
M_{10} &= (a_{1,1} + a_{1,2} - a_{1,3} - a_{2,2} + a_{2,3} + a_{3,1} + a_{3,2}) \cdot b_{2,3} \\
M_{11} &= (a_{3,2}) \cdot (-b_{1,1} + b_{1,3} + b_{2,1} - b_{2,2} - b_{2,3} - b_{3,1} + b_{3,2}) \\
M_{12} &= (a_{1,3} + a_{3,2} + a_{3,3}) \cdot (b_{2,2} + b_{3,1} - b_{3,2}) \\
M_{13} &= (a_{1,3} + a_{3,3}) \cdot (-b_{2,2} + b_{3,2}) \\
M_{14} &= a_{1,3} \cdot b_{3,1} \\
M_{15} &= (-a_{3,2} - a_{3,3}) \cdot (-b_{3,1} + b_{3,2}) \\
M_{16} &= (a_{1,3} + a_{2,2} - a_{2,3}) \cdot (b_{2,3} - b_{3,1} + b_{3,3}) \\
M_{17} &= (-a_{1,3} + a_{2,3}) \cdot (b_{2,3} + b_{3,3}) \\
M_{18} &= (a_{2,2} - a_{2,3}) \cdot (b_{3,1} - b_{3,3})
\end{aligned}$$

Laderman's scheme from 1976

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,2} & c_{3,3} \end{pmatrix}$$

where . . .

$$M_{19} = a_{1,2} \cdot b_{2,1}$$
$$M_{20} = a_{2,3} \cdot b_{3,2}$$
$$M_{21} = a_{2,1} \cdot b_{1,3}$$
$$M_{22} = a_{3,1} \cdot b_{1,2}$$
$$M_{23} = a_{3,3} \cdot b_{3,3}$$

Other schemes with 23 multiplications

- While Strassen's scheme is essentially the only way to do the 2×2 case with 7 multiplications, there are several distinct schemes for 3×3 matrices using 23 multiplications.

Other schemes with 23 multiplications

- While Strassen's scheme is essentially the only way to do the 2×2 case with 7 multiplications, there are several distinct schemes for 3×3 matrices using 23 multiplications.
- If we insist in integer coefficients, there have so far (and to our knowledge) been only three other schemes for 3×3 matrices and 23 multiplications.

Other schemes with 23 multiplications

- While Strassen's scheme is essentially the only way to do the 2×2 case with 7 multiplications, there are several distinct schemes for 3×3 matrices using 23 multiplications.
- If we insist in integer coefficients, there have so far (and to our knowledge) been only three other schemes for 3×3 matrices and 23 multiplications.
- Using altogether about 35 years of computation time, we found more than 13000 new schemes for 3×3 and 23, and we expect that there are many others.

Other schemes with 23 multiplications

- While Strassen's scheme is essentially the only way to do the 2×2 case with 7 multiplications, there are several distinct schemes for 3×3 matrices using 23 multiplications.
- If we insist in integer coefficients, there have so far (and to our knowledge) been only three other schemes for 3×3 matrices and 23 multiplications.
- Using altogether about 35 years of computation time, we found more than 13000 new schemes for 3×3 and 23, and we expect that there are many others.
- Unfortunately we found no scheme with only 22 multiplications

How to Search for a Matrix Multiplication Scheme? (1)

$$M_1 = (\alpha^{(1)}_{1,1} a_{1,1} + \alpha^{(1)}_{1,2} a_{1,2} + \cdots)(\beta^{(1)}_{1,1} b_{1,1} + \cdots)$$
$$M_2 = (\alpha^{(2)}_{1,1} a_{1,1} + \alpha^{(2)}_{1,2} a_{1,2} + \cdots)(\beta^{(2)}_{1,1} b_{1,1} + \cdots)$$
$$\vdots$$
$$c_{1,1} = \gamma^{(1)}_{1,1} M_1 + \gamma^{(2)}_{1,1} M_2 + \cdots$$
$$\vdots$$

Set $c_{i,j} = \sum_k a_{i,k} b_{k,j}$ for all i, j and compare coefficients.

How to Search for a Matrix Multiplication Scheme? (2)

This gives the Brent equations (for 3×3 with 23 multiplications)

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

The $\delta_{u,v}$ on the right refer to the Kronecker-delta, i.e., $\delta_{u,v} = 1$ if $u = v$ and $\delta_{u,v} = 0$ otherwise.

- $3^6 = 729$ cubic equations
- $23 \cdot 9 \cdot 3 = 621$ variables

How to Search for a Matrix Multiplication Scheme? (2)

This gives the Brent equations (for 3×3 with 23 multiplications)

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

The $\delta_{u,v}$ on the right refer to the Kronecker-delta, i.e., $\delta_{u,v} = 1$ if $u = v$ and $\delta_{u,v} = 0$ otherwise.

$3^6 = 729$ cubic equations

$23 \cdot 9 \cdot 3 = 621$ variables

Laderman claims that he solved this system by hand, but he doesn't say exactly how.

How to Search for a Matrix Multiplication Scheme? (3)

This gives the Brent equations (for 3×3 with 23 multiplications)

$$\forall\, i, j, k, l, m, n \in \{1, 2, 3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k} \delta_{i,m} \delta_{l,n}$$

The search space of the 3×3 case is enormous, even if $\alpha_{i,j}^{(q)}$, $\beta_{k,l}^{(q)}$, $\gamma_{m,n}^{(q)}$ are restricted to the values in $\{-1, 0, 1\}$

How to Search for a Matrix Multiplication Scheme? (3)

This gives the Brent equations (for 3×3 with 23 multiplications)

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

The search space of the 3×3 case is enormous, even if $\alpha_{i,j}^{(q)}$, $\beta_{k,l}^{(q)}$, $\gamma_{m,n}^{(q)}$ are restricted to the values in $\{-1,0,1\}$

Solution: Solve this system in $\mathbb{Z}_2$.

Reading $\alpha_{i,j}^{(q)}$, $\beta_{k,l}^{(q)}$, $\gamma_{m,n}^{(q)}$ as boolean variables and $+$ as XOR, the problem becomes a SAT problem.

Notice that solutions in $\mathbb{Z}_2$ may not be solutions in $\mathbb{Z}$

Lifting

Remember the Brent equations:

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

- Suppose we know a solution in $\mathbb{Z}_2$.
- Assume it came from a solution in $\mathbb{Z}$ with coefficients in $\{-1, 0, +1\}$.
- Then each $0 \in \mathbb{Z}_2$ was $0 \in \mathbb{Z}$ and each $1 \in \mathbb{Z}_2$ was $-1 \in \mathbb{Z}$ or $+1 \in \mathbb{Z}$.
- Plug the 0s of the $\mathbb{Z}_2$-solution into the Brent equations.
- Solve the resulting equations.

Lifting

Remember the Brent equations:

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

- Suppose we know a solution in $\mathbb{Z}_2$.
- Assume it came from a solution in $\mathbb{Z}$ with coefficients in $\{-1, 0, +1\}$.
- Then each $0 \in \mathbb{Z}_2$ was $0 \in \mathbb{Z}$ and each $1 \in \mathbb{Z}_2$ was $-1 \in \mathbb{Z}$ or $+1 \in \mathbb{Z}$.
- Plug the 0s of the $\mathbb{Z}_2$-solution into the Brent equations.
- Solve the resulting equations.

Can every $\mathbb{Z}_2$-solution be lifted to a $\mathbb{Z}$-solution in this way?

- No, and we found some which don't admit a lifting.
- But they are very rare. In almost all cases, the lifting succeeds.

How to Search for a Matrix Multiplication Scheme? (4)

This gives the Brent equations (for 3×3 with 23 multiplications)

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

Another solution: Solve this system by restricting equations with a zero righthand side to zero or two.

Still treat $\alpha_{i,j}^{(q)}$, $\beta_{k,l}^{(q)}$, $\gamma_{m,n}^{(q)}$ as boolean variables.

Notice that this restriction removes solutions, but it even works for Laderman.

How to Search for a Matrix Multiplication Scheme? (4)

This gives the Brent equations (for 3×3 with 23 multiplications)

$$\forall\, i,j,k,l,m,n \in \{1,2,3\} : \sum_{q=1}^{23} \alpha_{i,j}^{(q)} \beta_{k,l}^{(q)} \gamma_{m,n}^{(q)} = \delta_{j,k}\delta_{i,m}\delta_{l,n}$$

Another solution: Solve this system by restricting equations with a zero righthand side to zero or two.

Still treat $\alpha_{i,j}^{(q)}$, $\beta_{k,l}^{(q)}$, $\gamma_{m,n}^{(q)}$ as boolean variables.

Notice that this restriction removes solutions, but it even works for Laderman.

Important challenge: how to break the symmetries?

Most effective approach so far: sort the $\delta_{j,k}\delta_{i,m}\delta_{l,n} = 1$ terms

So what?

- Okay, so there are many more matrix multiplication methods for 3×3 matrices with 23 coefficient multiplications than previously known.

So what?

- Okay, so there are many more matrix multiplication methods for 3×3 matrices with 23 coefficient multiplications than previously known.
- In fact, we have shown that the dimension of the algebraic set defined by the Brent equation is much larger than was previously known.

So what?

- Okay, so there are many more matrix multiplication methods for 3×3 matrices with 23 coefficient multiplications than previously known.
- In fact, we have shown that the dimension of the algebraic set defined by the Brent equation is much larger than was previously known.
- But none of this has any immediate implications on the complexity of matrix multiplication, neither theoretically nor practically.

So what?

- Okay, so there are many more matrix multiplication methods for 3×3 matrices with 23 coefficient multiplications than previously known.
- In fact, we have shown that the dimension of the algebraic set defined by the Brent equation is much larger than was previously known.
- But none of this has any immediate implications on the complexity of matrix multiplication, neither theoretically nor practically.
- In particular, it remains open whether there is a multiplication method for 3×3 matrices with 22 coefficient multiplications. If you find one, let us know.

Scheme Database

Check out our website for browsing through
the schemes and families we found:

http://www.algebra.uni-linz.ac.at/research/matrix-multiplication/

The Collatz Conjecture

joint work with Scott Aaronson

Beyond NP: The Collatz Conjecture

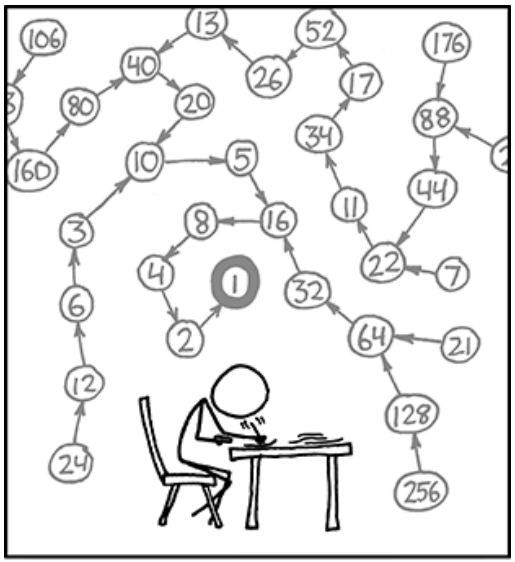
Resolving foundational algorithm questions

$$Col(n) = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (3n+1)/2 & \text{if } n \text{ is odd} \end{cases}$$

Does `while`$(n > 1)\ n = Col(n);$ terminate?

Find a non-negative function $fun(n)$ s.t.

$$\forall n > 1 : fun(n) > fun(Col(n))$$



THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.

source: xkcd.com/710

Beyond NP: The Collatz Conjecture

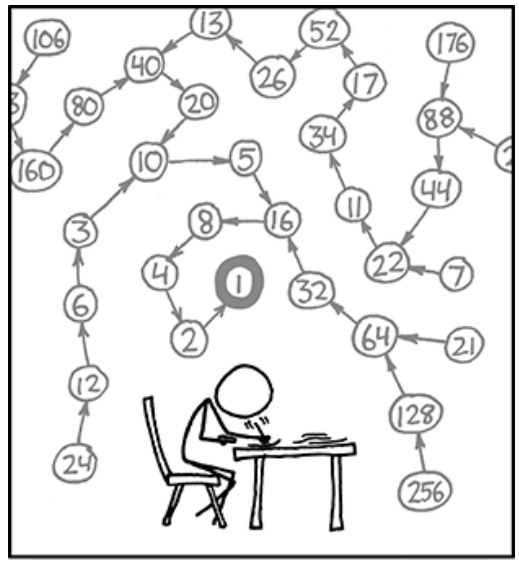
Resolving foundational algorithm questions

$$Col(n) = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (3n+1)/2 & \text{if } n \text{ is odd} \end{cases}$$

Does `while`$(n > 1)\ n = Col(n);$ terminate?

Find a non-negative function $fun(n)$ s.t.

$$\forall n > 1 : fun(n) > fun(Col(n))$$



THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.

source: xkcd.com/710

Can we construct a function for which $fun(n) > fun(Col(n))$ holds?

$fun(3)$	$fun(5)$	$fun(8)$	$fun(4)$	$fun(2)$	$fun(1)$
5	4	3	2	1	0

Term Rewriting Termination Example

Given a set of rewriting rules,
will rewriting always terminate?

Term Rewriting Termination Example

Given a set of rewriting rules,
will rewriting always terminate?

Example set of rules (Zantema's "other" Problem):

- $aa \rightarrow_R bc$
- $bb \rightarrow_R ac$
- $cc \rightarrow_R ab$

Term Rewriting Termination Example

Given a set of rewriting rules,
will rewriting always terminate?

Example set of rules (Zantema's "other" Problem):

- $aa \rightarrow_R bc$
- $bb \rightarrow_R ac$
- $cc \rightarrow_R ab$

$$bb\underline{aa} \rightarrow_R b\underline{bb}c \rightarrow_R ba\underline{cc} \rightarrow_R b\underline{aa}b \rightarrow_R \underline{bb}cb \rightarrow_R$$
$$a\underline{cc}b \rightarrow_R aa\underline{bb} \rightarrow_R a\underline{aa}c \rightarrow_R ab\underline{cc} \rightarrow_R abab$$

Term Rewriting Termination Example

Given a set of rewriting rules,
will rewriting always terminate?

Example set of rules (Zantema's "other" Problem):

- $aa \rightarrow_R bc$
- $bb \rightarrow_R ac$
- $cc \rightarrow_R ab$

$$bb\underline{aa} \rightarrow_R b\underline{bb}c \rightarrow_R ba\underline{cc} \rightarrow_R b\underline{aa}b \rightarrow_R \underline{bb}cb \rightarrow_R$$
$$a\underline{cc}b \rightarrow_R aa\underline{bb} \rightarrow_R a\underline{aa}c \rightarrow_R ab\underline{cc} \rightarrow_R abab$$

Strongest rewriting solvers use SAT (e.g. AProVE)

Example first solved by Hofbauer and Waldmann (2006)

Term Rewriting Proof Outline

Prove termination of Zantema's "other" Problem:

- $aa \to_R bc$
- $bb \to_R ac$
- $cc \to_R ab$

Proof outline:

- Interpret a,b,c by linear functions $[a], [b], [c]$ from $\mathbf{N}^4$ to $\mathbf{N}^4$
- Interpret string concatenation by function composition
- Show that if $[uaav]\ (0,0,0,0) = (x_1, x_2, x_3, x_4)$ and $[ubcv]\ (0,0,0,0) = (y_1, y_2, y_3, y_4)$ then $x_1 > y_1$
- Similar for $bb \to ac$ and $cc \to ab$
- Hence every rewrite step gives a decrease of $x_1 \in \mathbf{N}$, so rewriting terminates

Term Rewriting Termination Proof

The linear functions:

$$[a](\vec{x}) = \begin{pmatrix} 1 & 0 & 0 & 3 \\ 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \vec{x} + \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$[b](\vec{x}) = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \vec{x} + \begin{pmatrix} 0 \\ 2 \\ 0 \\ 0 \end{pmatrix}$$

$$[c](\vec{x}) = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \end{pmatrix} \vec{x} + \begin{pmatrix} 1 \\ 0 \\ 3 \\ 0 \end{pmatrix}$$

Checking decrease properties using linear algebra

Collatz Conjecture (2)

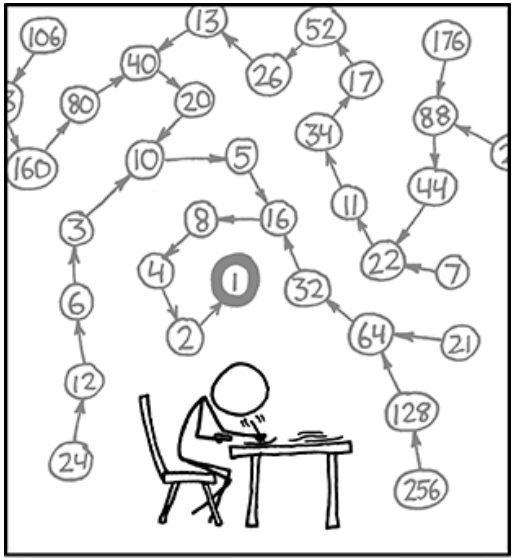
Resolving foundational algorithm questions

$$Col(n) = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (3n+1)/2 & \text{if } n \text{ is odd} \end{cases}$$

Does `while`$(n > 1)\ n = Col(n);$ terminate?

Find a non-negative function $fun(n)$ s.t.

$$\forall n > 1 : fun(n) > fun(Col(n))$$



THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.

source: xkcd.com/710

Collatz Conjecture (2)

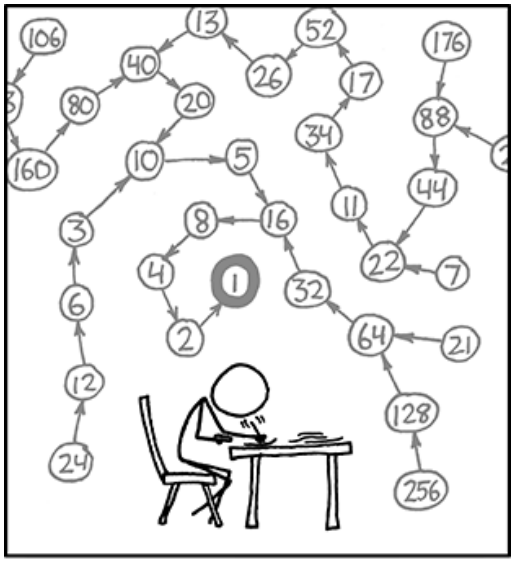
Resolving foundational algorithm questions

$$Col(n) = \begin{cases} n/2 & \text{if } n \text{ is even} \\ (3n+1)/2 & \text{if } n \text{ is odd} \end{cases}$$

Does `while`$(n > 1)\ n = Col(n);$ terminate?

Find a non-negative function $fun(n)$ s.t.

$$\forall n > 1 : fun(n) > fun(Col(n))$$



THE COLLATZ CONJECTURE STATES THAT IF YOU PICK A NUMBER, AND IF IT'S EVEN DIVIDE IT BY TWO AND IF IT'S ODD MULTIPLY IT BY THREE AND ADD ONE, AND YOU REPEAT THIS PROCEDURE LONG ENOUGH, EVENTUALLY YOUR FRIENDS WILL STOP CALLING TO SEE IF YOU WANT TO HANG OUT.

source: xkcd.com/710

$fun(3)$	$fun(5)$	$fun(8)$	$fun(4)$	$fun(2)$	$fun(1)$
$\mathbf{t}(\mathbf{t}(\vec{0}))$	$\mathbf{t}(\mathbf{f}(\mathbf{t}(\vec{0})))$	$\mathbf{t}(\mathbf{f}(\mathbf{f}(\mathbf{f}(\vec{0}))))$	$\mathbf{t}(\mathbf{f}(\mathbf{f}(\vec{0})))$	$\mathbf{t}(\mathbf{f}(\vec{0}))$	$\mathbf{t}(\vec{0})$
$\begin{pmatrix}5\\1\end{pmatrix}$	$\begin{pmatrix}4\\1\end{pmatrix}$	$\begin{pmatrix}3\\1\end{pmatrix}$	$\begin{pmatrix}2\\1\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$

using $\mathbf{t}(\vec{x}) = \begin{pmatrix}1&5\\0&0\end{pmatrix}\vec{x} + \begin{pmatrix}0\\1\end{pmatrix}$ and $\mathbf{f}(\vec{x}) = \begin{pmatrix}1&3\\0&0\end{pmatrix}\vec{x} + \begin{pmatrix}1\\0\end{pmatrix}$

The Collatz Conjecture as Rewriting System

Consider the following functions:

- Binary system: $f(x) = 2x$, $t(x) = 2x + 1$
- Ternary system: $p(x) = 3x$, $q(x) = 3x + 1$, $r(x) = 3x + 2$
- Start and end symbols: $c(x) = 1$, $d(x) = x$

D_1: $fd \to_R d$
D_2: $td \to_R rd$

F_1: $fp \to_R pf$ $\quad$ T_1: $tp \to_R qt$ $\quad$ C_1: $cp \to_R ct$
F_2: $fq \to_R pt$ $\quad$ T_2: $tq \to_R rf$ $\quad$ C_2: $cq \to_R cff$
F_3: $fr \to_R qf$ $\quad$ T_3: $tr \to_R rt$ $\quad$ C_3: $cr \to_R cft$

Interpretation using the functions above:

$D_1 : 2x \to x$
$D_2 : 2x + 1 \to 3x + 2 \quad (= (3(2x + 1) + 1)/2)$
$F_1 : 6x \to 6x$
$T_3 : 6x + 5 \to 6x + 5$

Collatz Rewriting Example

$D_1\colon\ fd \to_R d$
$D_2\colon\ td \to_R rd$

$F_1\colon\ fp \to_R pf$
$F_2\colon\ fq \to_R pt$
$F_3\colon\ fr \to_R qf$

$T_1\colon\ tp \to_R qt$
$T_2\colon\ tq \to_R rf$
$T_3\colon\ tr \to_R rt$

$C_1\colon\ cp \to_R ct$
$C_2\colon\ cq \to_R cff$
$C_3\colon\ cr \to_R cft$

$$
\begin{array}{ccccccccccccccccc}
c\underline{td} & \to & \underline{cr}d & \to & cf\underline{td} & \to & c\underline{fr}d & \to & \underline{cq}fd & \to & cff\underline{fd} & \to & cf\underline{fd} & \to & c\underline{fd} & \to & cd \\
 & & D_2 & & C_3 & & D_2 & & F_3 & & C_2 & & D_1 & & D_1 & & D_1 \\
3 & \to & 5 & \to & 5 & \to & 8 & \to & 8 & \to & 8 & \to & 4 & \to & 2 & \to & 1
\end{array}
$$

Collatz Rewriting Example

$$D_1\colon fd \to_R d \qquad D_2\colon td \to_R rd$$

$$F_1\colon fp \to_R pf \qquad F_2\colon fq \to_R pt \qquad F_3\colon fr \to_R qf$$

$$T_1\colon tp \to_R qt \qquad T_2\colon tq \to_R rf \qquad T_3\colon tr \to_R rt$$

$$C_1\colon cp \to_R ct \qquad C_2\colon cq \to_R cff \qquad C_3\colon cr \to_R cft$$

$$\begin{array}{ccccccccccccccccc}
c\underline{td} & \to & \underline{cr}d & \to & cf\underline{td} & \to & c\underline{fr}d & \to & \underline{cq}fd & \to & cff\underline{fd} & \to & cf\underline{fd} & \to & c\underline{fd} & \to & cd \\
 & D_2 & & C_3 & & D_2 & & F_3 & & C_2 & & D_1 & & D_1 & & D_1 & \\
3 & \to & 5 & \to & 5 & \to & 8 & \to & 8 & \to & 8 & \to & 4 & \to & 2 & \to & 1
\end{array}$$

Can we prove termination of the Collatz rewriting system?

Collatz Rewriting Example

$D_1\colon\ fd \to_R d$
$D_2\colon\ td \to_R rd$

$F_1\colon\ fp \to_R pf$
$F_2\colon\ fq \to_R pt$
$F_3\colon\ fr \to_R qf$

$T_1\colon\ tp \to_R qt$
$T_2\colon\ tq \to_R rf$
$T_3\colon\ tr \to_R rt$

$C_1\colon\ cp \to_R ct$
$C_2\colon\ cq \to_R cff$
$C_3\colon\ cr \to_R cft$

$$\begin{array}{ccccccccccccccccc}
c\underline{td} & \to & \underline{cr}d & \to & cf\underline{td} & \to & c\underline{fr}d & \to & \underline{cq}fd & \to & cff\underline{fd} & \to & cf\underline{fd} & \to & c\underline{fd} & \to & cd \\
 & D_2 & & C_3 & & D_2 & & F_3 & & C_2 & & D_1 & & D_1 & & D_1 & \\
3 & \to & 5 & \to & 5 & \to & 8 & \to & 8 & \to & 8 & \to & 4 & \to & 2 & \to & 1
\end{array}$$

Can we prove termination of the Collatz rewriting system?

The full system is still too hard, but subsystems (removing one of the rules) are doable (although not with existing tools).

Results on Proving Termination of Subsystems

missing rule	dimension	value limit	runtime
D_1	3	3	1.40
D_2	1	1	0.00
F_1	4	5	5 828.36
F_2	2	3	0.02
F_3	2	2	0.01
T_1	4	7	25 340.99
T_2	5	7	44 056.35
T_3	4	6	37 071.33
C_1	2	2	0.01
C_2	3	4	23.35
C_3	4	5	75.89

Challenges for Collatz Conjecture

The presented system is just one of many possible rewriting systems that captures the Collatz conjecture.

Which system facilitates efficient reasoning?

Challenges for Collatz Conjecture

The presented system is just one of many possible rewriting systems that captures the Collatz conjecture.

Which system facilitates efficient reasoning?

How to encode the SAT formula?

- The order encoding for multiplication is very effective
- Reduce the size of the encoding my reusing calculations

Challenges for Collatz Conjecture

The presented system is just one of many possible rewriting systems that captures the Collatz conjecture.

Which system facilitates efficient reasoning?

How to encode the SAT formula?

- The order encoding for multiplication is very effective
- Reduce the size of the encoding my reusing calculations

Which SAT solving techniques are effective?

- Surprisingly old SAT solvers work better than new ones
- Can local search be effective (we only look for solutions)?

Encoding Challenges for Hard Problems

Marijn J.H. Heule

THE UNIVERSITY OF TEXAS AT AUSTIN

Starting at Carnegie Mellon University in August

Matryoshka workshop June 12, 2019