The algebra of programs

Tobias Kappé

Eerstejaarsreis DLF — January 19th, 2019

Introduction

Alexandra Silva

Dexter Kozen

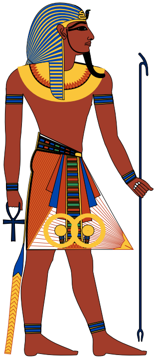
Fredrik Dahlqvist

Justin Hsu

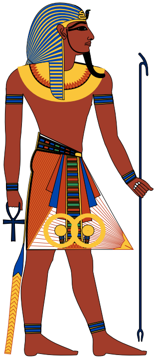
Nate Foster

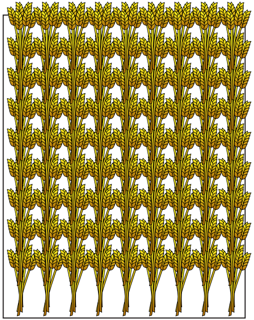
Steffen Smolka

Primitive geometry

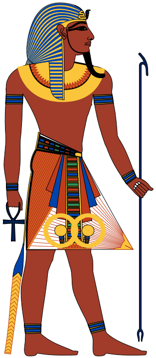


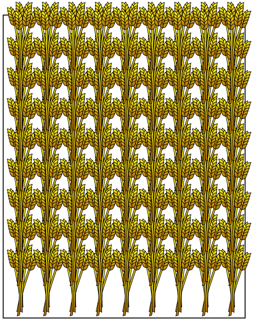
Primitive geometry

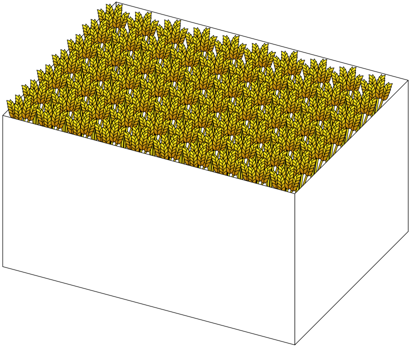




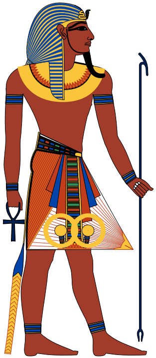
Primitive geometry

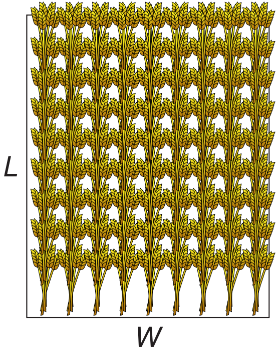


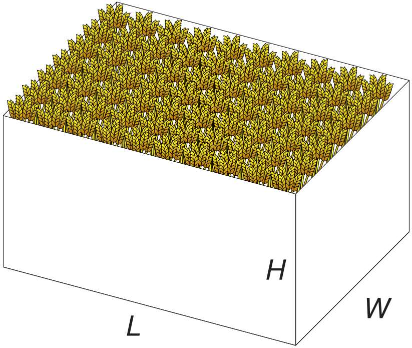




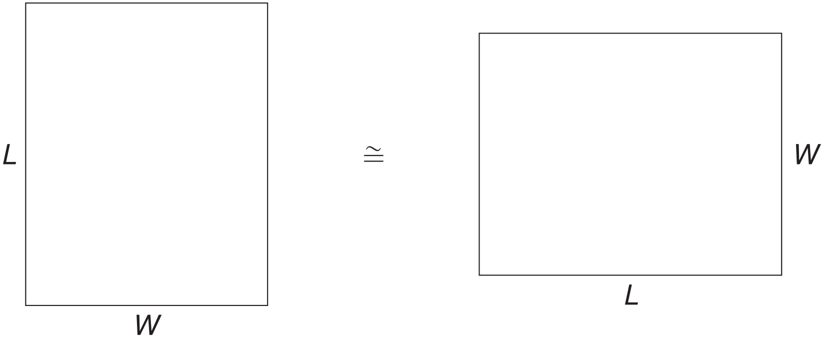
Primitive geometry

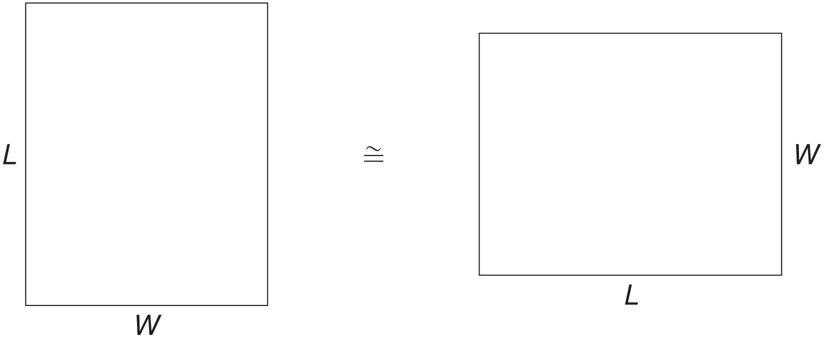






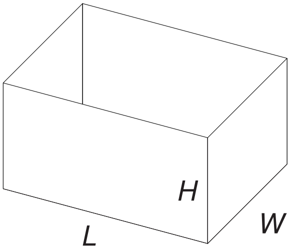
Primitive geometry



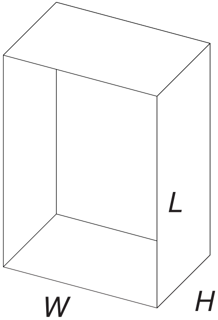


$$W \times L = L \times W$$

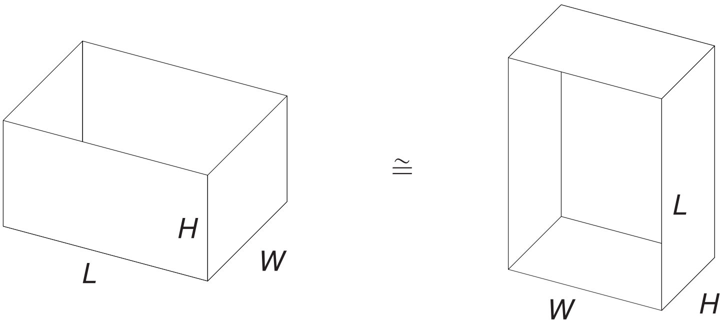
Primitive geometry



$\cong$

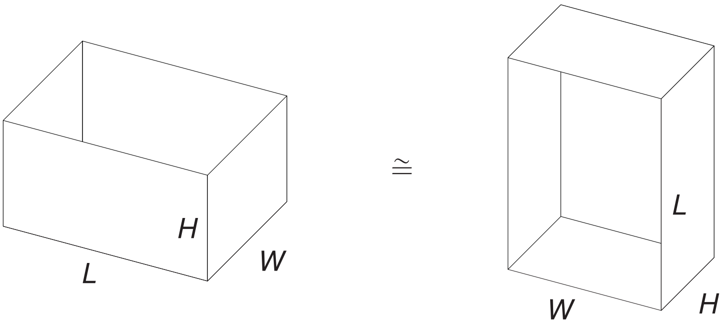


Primitive geometry



$$(L \times W) \times H = (W \times H) \times L$$

Primitive geometry



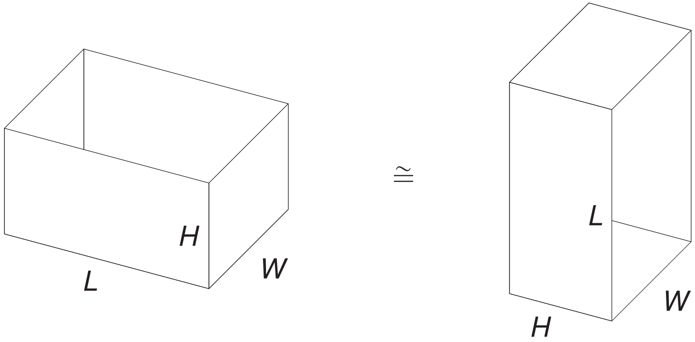
$$(L \times W) \times H = L \times (W \times H)$$

Primitive geometry

$$(L \times W) \times H = L \times (W \times H) = L \times (H \times W) = (H \times W) \times L$$

Primitive geometry

$$(L \times W) \times H = L \times (W \times H) = L \times (H \times W) = (H \times W) \times L$$



Primitive geometry

- You can imagine “laws” of multiplication, even if you know only what it *represents*.
- These laws then allow you to *reason* about what else should be true.

And now for something completely different

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \fatsemi Q \qquad P \oplus_{\phi} Q \qquad P^{\phi}$$

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \fatsemi Q \qquad P \oplus_\phi Q \qquad P^\phi$$

abort if ϕ is false

Reasoning about programs

- Consider this “programming language”:

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_{\phi} Q \qquad P^{\phi}$$

first execute P, then execute Q

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_\phi Q \qquad P^\phi$$

if ϕ holds, run P, otherwise run Q.

Reasoning about programs

- Consider this “programming language”:

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_{\phi} Q \qquad P^{\phi}$$

run P for as long as ϕ holds.

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_{\phi} Q \qquad P^{\phi}$$

- Write $P \leqq Q$ if P and Q agree on the inputs where P succeeds.

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_\phi Q \qquad P^\phi$$

- Write $P \leqq Q$ if P and Q agree on the inputs where P succeeds.

Q "simulates" P

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_\phi Q \qquad P^\phi$$

- Write $P \leqq Q$ if P and Q agree on the inputs where P succeeds.
- If $P \leqq Q$ and $Q \leqq P$, we write $P \equiv Q$.

Reasoning about programs

- Consider this “programming language”:

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_{\phi} Q \qquad P^{\phi}$$

- Write $P \leqq Q$ if P and Q agree on the inputs where P succeeds.
- If $P \leqq Q$ and $Q \leqq P$, we write $P \equiv Q$.
- For example, we have:

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \mathbin{;} Q \qquad P \oplus_\phi Q \qquad P^\phi$$

- Write $P \leqq Q$ if P and Q agree on the inputs where P succeeds.
- If $P \leqq Q$ and $Q \leqq P$, we write $P \equiv Q$.
- For example, we have:

$$[\texttt{false}] \leqq P$$

Reasoning about programs

- Consider this "programming language":

$$[\phi] \qquad P \fatsemi Q \qquad P \oplus_{\phi} Q \qquad P^{\phi}$$

- Write $P \leqq Q$ if P and Q agree on the inputs where P succeeds.
- If $P \leqq Q$ and $Q \leqq P$, we write $P \equiv Q$.
- For example, we have:

$$[\texttt{false}] \leqq P \qquad Q \oplus_{\neg\phi} P \equiv P \oplus_{\phi} Q$$

Reasoning about programs

Other “laws of programming”?

Reasoning about programs

Other “laws of programming”?

$$[\phi] \mathbin{;} [\psi] \equiv [\phi \wedge \psi]$$

Reasoning about programs

Other “laws of programming”?

$$[\phi] \mathbin{;} [\psi] \equiv [\phi \wedge \psi] \qquad\qquad P \oplus_\phi Q \equiv ([\phi] \mathbin{;} P) \oplus_\phi Q$$

Other “laws of programming”?

$$[\phi] \fatsemi [\psi] \equiv [\phi \wedge \psi]$$

$$P \oplus_\phi Q \equiv ([\phi] \fatsemi P) \oplus_\phi Q$$

$$P^\phi \equiv (P \fatsemi P^\phi) \oplus_\phi [\mathtt{true}]$$

Other "laws of programming"?

$$[\phi] \mathbin{;} [\psi] \equiv [\phi \wedge \psi]$$

$$P \oplus_\phi Q \equiv ([\phi] \mathbin{;} P) \oplus_\phi Q$$

$$P^\phi \equiv (P \mathbin{;} P^\phi) \oplus_\phi [\mathtt{true}]$$

$$(P \mathbin{;} R) \oplus_\phi (Q \mathbin{;} R) \equiv (P \oplus_\phi Q) \mathbin{;} R$$

We also have the *fixpoint rule*:

$$\frac{P \equiv (Q \fatsemi P) \oplus_{\phi} R}{Q^{\phi} \fatsemi R \leqq P}$$

Reasoning about programs

We also have the *fixpoint rule*:

$$\frac{P \equiv (Q \fatsemi P) \oplus_\phi R}{Q^\phi \fatsemi R \leqq P}$$

If P is a program which does the following:

- If ϕ holds, execute Q and start again with P.
- Otherwise, execute the program R.

then P can simulate $Q^\phi \fatsemi R$.

Reasoning about programs

The the dual of the fixpoint rule does *not* hold in general:

$$\frac{P \equiv (Q \mathbin{;} P) \oplus_{\phi} R}{P \leqq Q^{\phi} \mathbin{;} R}$$

Reasoning about programs

The the dual of the fixpoint rule does *not* hold in general:

$$\frac{P \equiv (Q \fatsemi P) \oplus_{\phi} R}{P \leqq Q^{\phi} \fatsemi R}$$

Counterexample: consider that

$$[\mathtt{true}] \equiv ([\mathtt{true}] \fatsemi [\mathtt{true}]) \oplus_{\mathtt{true}} [\mathtt{true}]$$

Reasoning about programs

The the dual of the fixpoint rule does *not* hold in general:

$$\frac{P \equiv (Q \fatsemi P) \oplus_\phi R}{P \leqq Q^\phi \fatsemi R}$$

Counterexample: consider that

$$[\texttt{true}] \equiv ([\texttt{true}] \fatsemi [\texttt{true}]) \oplus_{\texttt{true}} [\texttt{true}]$$

while the following is false:

$$[\texttt{true}] \leqq [\texttt{true}]^{\texttt{true}} \fatsemi [\texttt{true}]$$

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv ([\phi] \mathbin{;} P)^{\phi}$

Proof.

First, note that

$$\begin{aligned} P^{\phi} &\equiv (P \mathbin{;} P^{\phi}) \oplus_{\phi} [\mathtt{true}] \\ &\equiv ([\phi] \mathbin{;} P \mathbin{;} P^{\phi}) \oplus_{\phi} [\mathtt{true}] \end{aligned}$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv ([\phi] \fatsemi P)^{\phi}$

Proof.

First, note that

$$\begin{aligned} P^{\phi} &\equiv (P \fatsemi P^{\phi}) \oplus_{\phi} [\texttt{true}] \\ &\equiv ([\phi] \fatsemi P \fatsemi P^{\phi}) \oplus_{\phi} [\texttt{true}] \end{aligned}$$

Thus, by the fixpoint rule

$$([\phi] \fatsemi P)^{\phi} \equiv ([\phi] \fatsemi P)^{\phi} \fatsemi [\texttt{true}] \leq P^{\phi}$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv ([\phi] \fatsemi P)^{\phi}$

Proof.

For the other direction, we note

$$
\begin{aligned}
([\phi] \fatsemi P)^{\phi} &\equiv ([\phi] \fatsemi P \fatsemi ([\phi] \fatsemi P)^{\phi}) \oplus_{\phi} [\mathtt{true}] \\
&\equiv (P \fatsemi ([\phi] \fatsemi P)^{\phi}) \oplus_{\phi} [\mathtt{true}]
\end{aligned}
$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv ([\phi] \fatsemi P)^{\phi}$

Proof.

For the other direction, we note

$$\begin{aligned}([\phi] \fatsemi P)^{\phi} &\equiv ([\phi] \fatsemi P \fatsemi ([\phi] \fatsemi P)^{\phi}) \oplus_{\phi} [\mathtt{true}] \\ &\equiv (P \fatsemi ([\phi] \fatsemi P)^{\phi}) \oplus_{\phi} [\mathtt{true}]\end{aligned}$$

Thus, by the fixpoint rule

$$P^{\phi} \equiv P^{\phi} \fatsemi [\mathtt{true}] \leq ([\phi] \fatsemi P)^{\phi}$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv P^{\phi} \fatsemi [\neg\phi]$.

Proof.

First, derive that

$$\begin{aligned} P^{\phi} &\equiv (P \fatsemi P^{\phi}) \oplus_{\phi} [\mathtt{true}] \\ &\equiv (P \fatsemi P^{\phi}) \oplus_{\phi} [\neg\phi] \end{aligned}$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv P^{\phi} \fatsemi [\neg\phi]$.

Proof.

First, derive that

$$\begin{aligned} P^{\phi} &\equiv (P \fatsemi P^{\phi}) \oplus_{\phi} [\mathtt{true}] \\ &\equiv (P \fatsemi P^{\phi}) \oplus_{\phi} [\neg\phi] \end{aligned}$$

Thus, by the fixpoint rule

$$P^{\phi} \fatsemi [\neg\phi] \leqq P^{\phi}$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^{\phi} \equiv P^{\phi} \fatsemi [\neg\phi]$.

Proof.

For the other direction, derive that

$$
\begin{aligned}
P^{\phi} \fatsemi [\neg\phi] &\equiv ((P \fatsemi P^{\phi}) \oplus_{\phi} [\mathtt{true}]) \fatsemi [\neg\phi] \\
&\equiv (P \fatsemi P^{\phi} \fatsemi [\neg\phi]) \oplus_{\phi} [\neg\phi] \\
&\equiv (P \fatsemi P^{\phi} \fatsemi [\neg\phi]) \oplus_{\phi} [\mathtt{true}]
\end{aligned}
$$

□

Reasoning about programs

Lemma

For all P and ϕ, we have $P^\phi \equiv P^\phi \fatsemi [\neg\phi]$.

Proof.

For the other direction, derive that

$$
\begin{aligned}
P^\phi \fatsemi [\neg\phi] &\equiv ((P \fatsemi P^\phi) \oplus_\phi [\mathtt{true}]) \fatsemi [\neg\phi] \\
&\equiv (P \fatsemi P^\phi \fatsemi [\neg\phi]) \oplus_\phi [\neg\phi] \\
&\equiv (P \fatsemi P^\phi \fatsemi [\neg\phi]) \oplus_\phi [\mathtt{true}]
\end{aligned}
$$

Thus, by the fixpoint rule

$$P^\phi \equiv P^\phi \fatsemi [\mathtt{true}] \leq P^\phi \fatsemi [\neg\phi]$$

□

Open questions

A function $[\![-]\!] : \mathrm{Prog} \to S$ is called a *model*

A model is[1]

- *sound* if whenever $P \leqq Q$ we have $[\![P]\!] \subseteq [\![Q]\!]$

[1] Stretching established terms a bit here.

Open questions

A function $[\![-]\!] : \mathrm{Prog} \to S$ is called a *model*

A model is[1]

- *sound* if whenever $P \leqq Q$ we have $[\![P]\!] \subseteq [\![Q]\!]$
- *complete* if whenever $[\![P]\!] \subseteq [\![Q]\!]$ we have $P \leqq Q$

[1] Stretching established terms a bit here.

Open questions

A function $[\![-]\!] : \mathrm{Prog} \to S$ is called a *model*

A model is[1]

- *sound* if whenever $P \leqq Q$ we have $[\![P]\!] \subseteq [\![Q]\!]$
- *complete* if whenever $[\![P]\!] \subseteq [\![Q]\!]$ we have $P \leqq Q$
- *free* if it is both sound and complete.

[1] Stretching established terms a bit here.

Open questions

A function $[\![-]\!] : \text{Prog} \to S$ is called a *model*

A model is[1]

- *sound* if whenever $P \leqq Q$ we have $[\![P]\!] \subseteq [\![Q]\!]$
- *complete* if whenever $[\![P]\!] \subseteq [\![Q]\!]$ we have $P \leqq Q$
- *free* if it is both sound and complete.

Question: what is the free model of these expressions?

[1] Stretching established terms a bit here.

Open questions

Proofs are hard — can we automate them?

Battle plan:

Open questions

Proofs are hard — can we automate them?

Battle plan:

- Suppose $[\![-]\!]$ is free — then $P \leqq Q \iff [\![P]\!] \subseteq [\![Q]\!]$.

Open questions

Proofs are hard — can we automate them?

Battle plan:

- Suppose $[\![-]\!]$ is free — then $P \leqq Q \iff [\![P]\!] \subseteq [\![Q]\!]$.
- But $[\![P]\!]$ and $[\![Q]\!]$ are (in general) infinite!

Open questions

Proofs are hard — can we automate them?

Battle plan:

- Suppose $[\![-]\!]$ is free — then $P \leq Q \iff [\![P]\!] \subseteq [\![Q]\!]$.
- But $[\![P]\!]$ and $[\![Q]\!]$ are (in general) infinite!
- Create *finite* representation (“automaton”) A_P where $L(A_P) = [\![P]\!]$.

Open questions

Proofs are hard — can we automate them?

Battle plan:

- Suppose $[\![-]\!]$ is free — then $P \leqq Q \iff [\![P]\!] \subseteq [\![Q]\!]$.
- But $[\![P]\!]$ and $[\![Q]\!]$ are (in general) infinite!
- Create *finite* representation (“automaton”) A_P where $L(A_P) = [\![P]\!]$.
- Design an algorithm to check whether $L(A_P) \subseteq L(A_Q)$.

Thank you for your attention

CoNeCo

https://coneco-project.org

For slides, see https://tobias.kap.pe