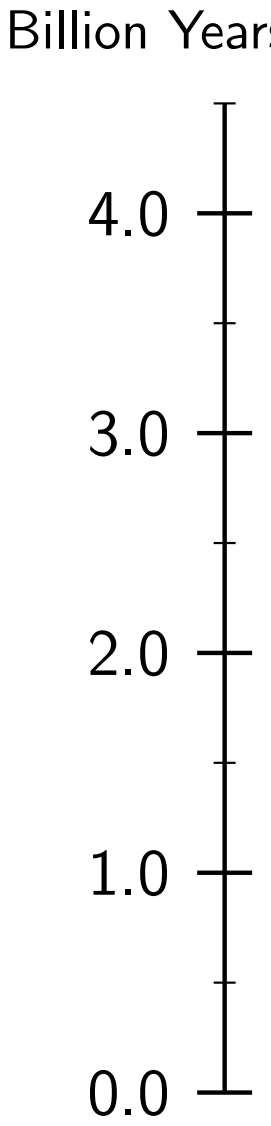
SWITCHING CIRCUIT DESIGNS from BIOLOGICAL EVOLUTION

David M. Clark
Mathematics Department
SUNY, New Paltz

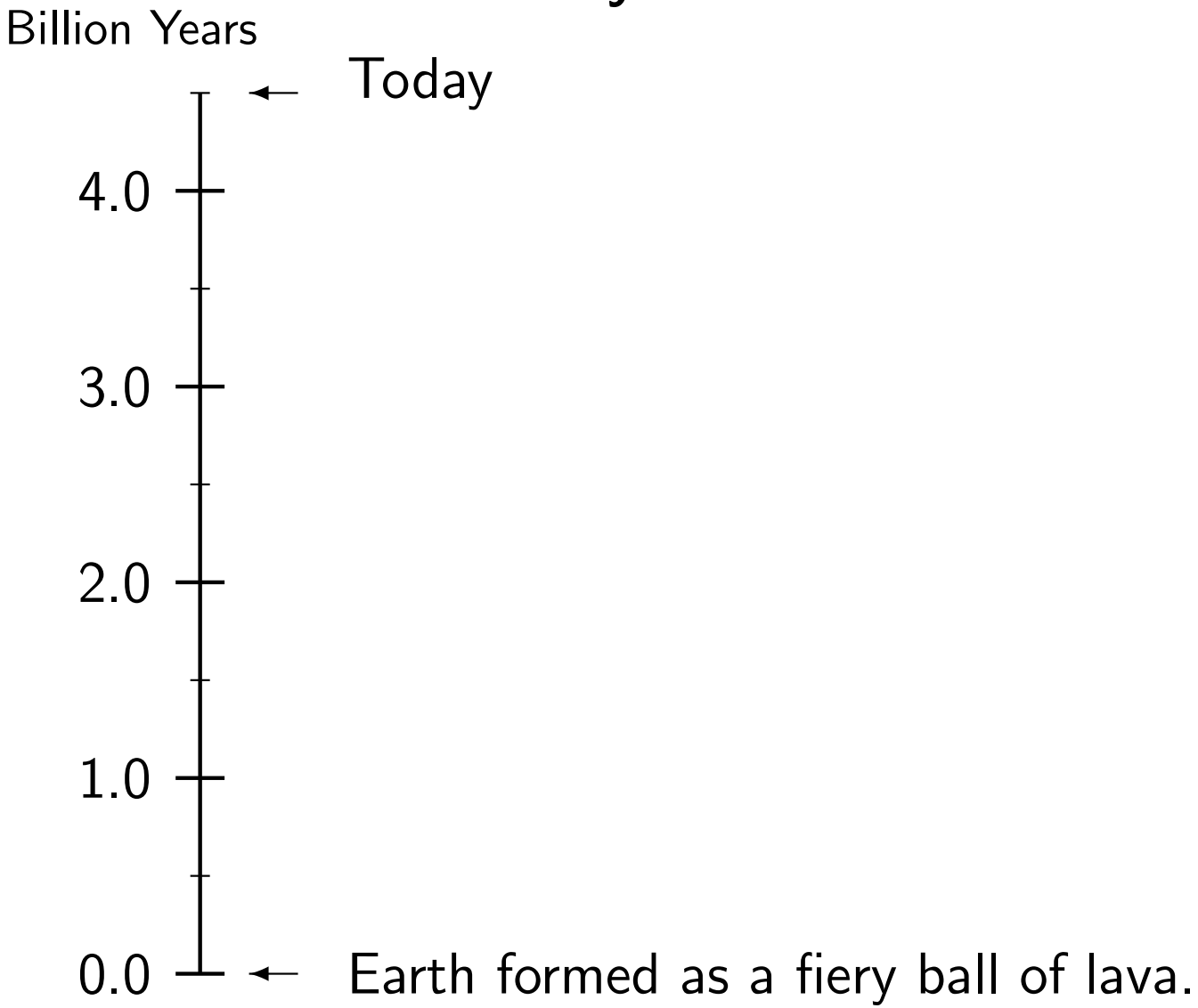
September 19, 2022

1. Biological Evolution

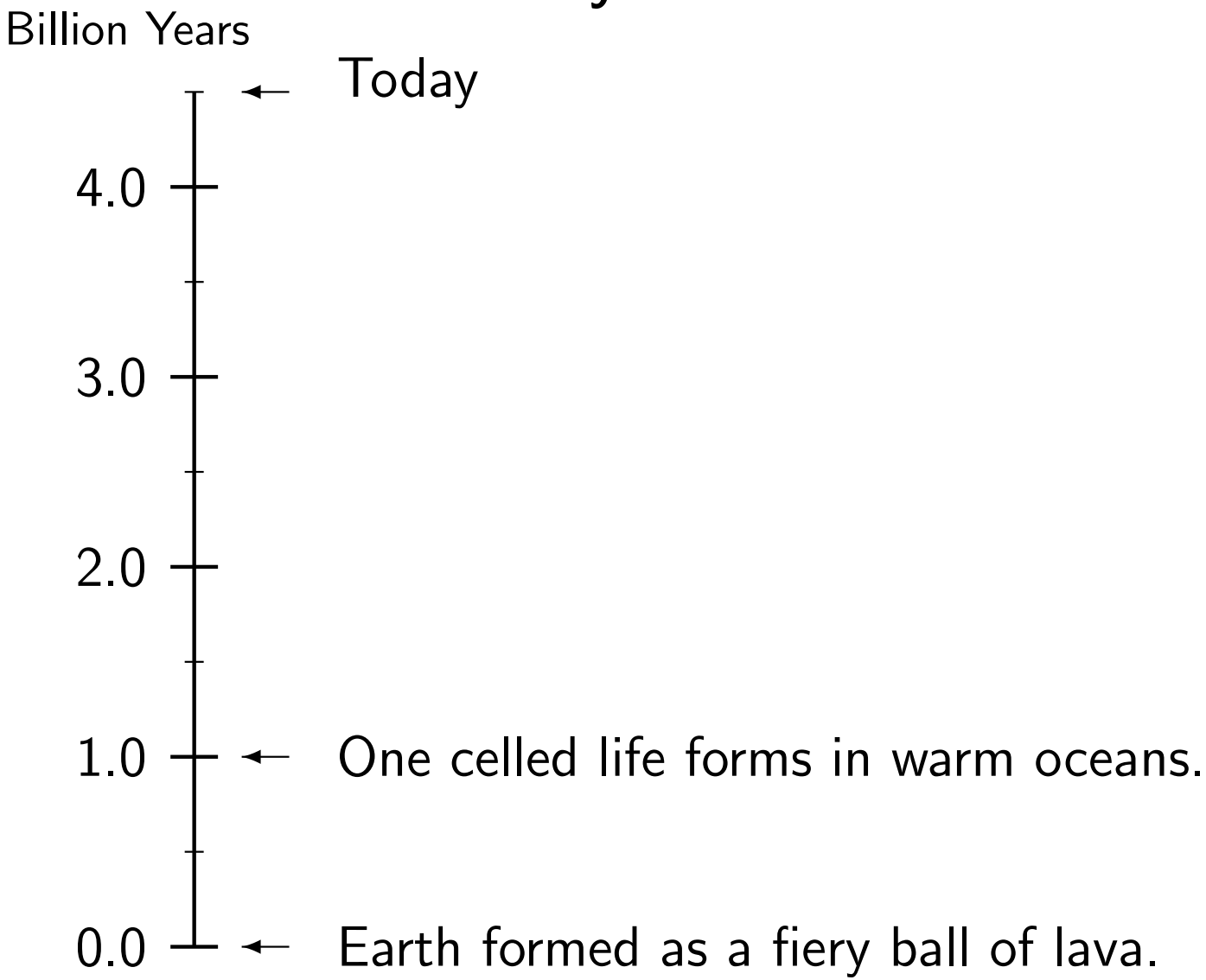
History of Life on Earth



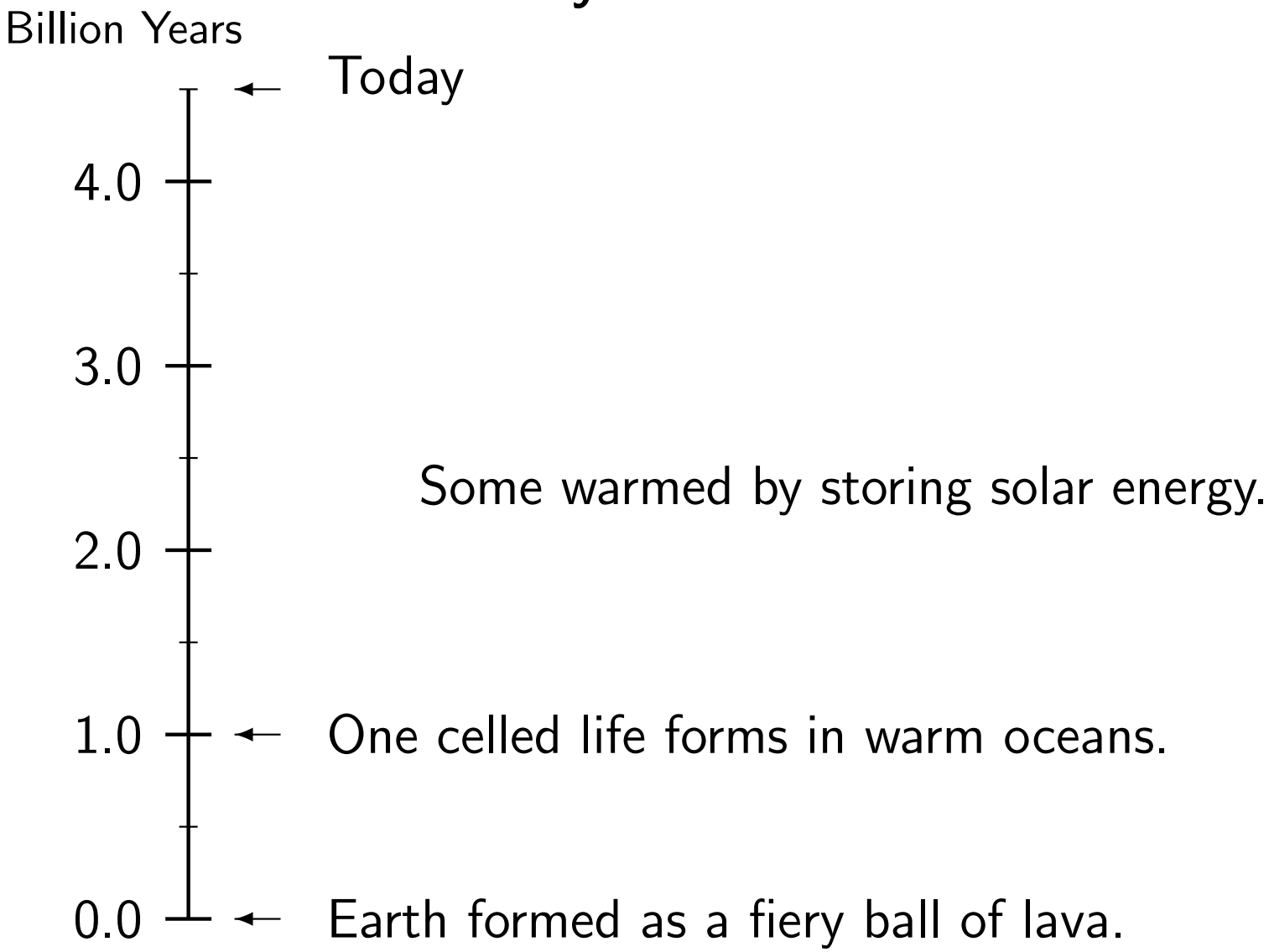
History of Life on Earth



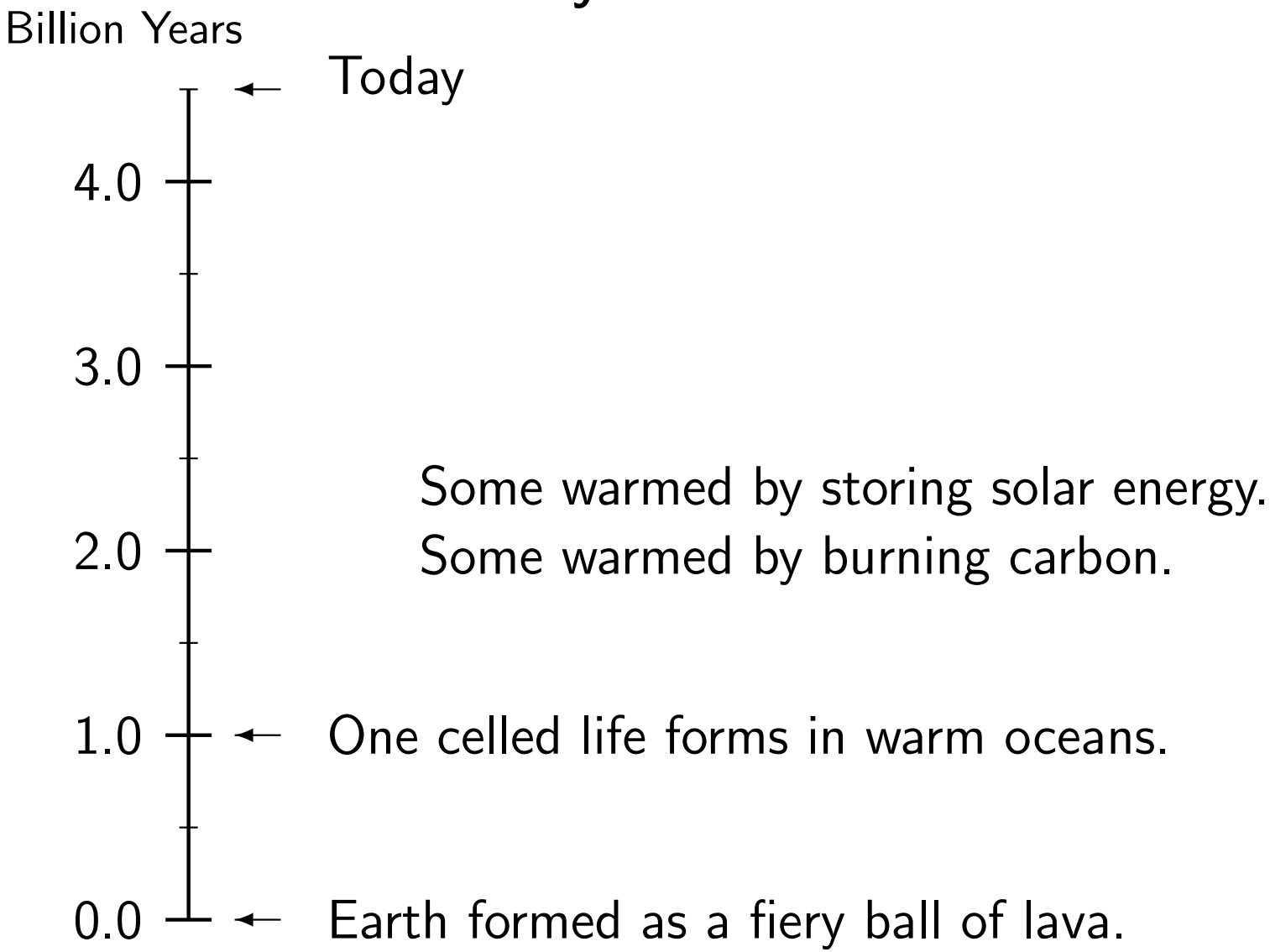
History of Life on Earth



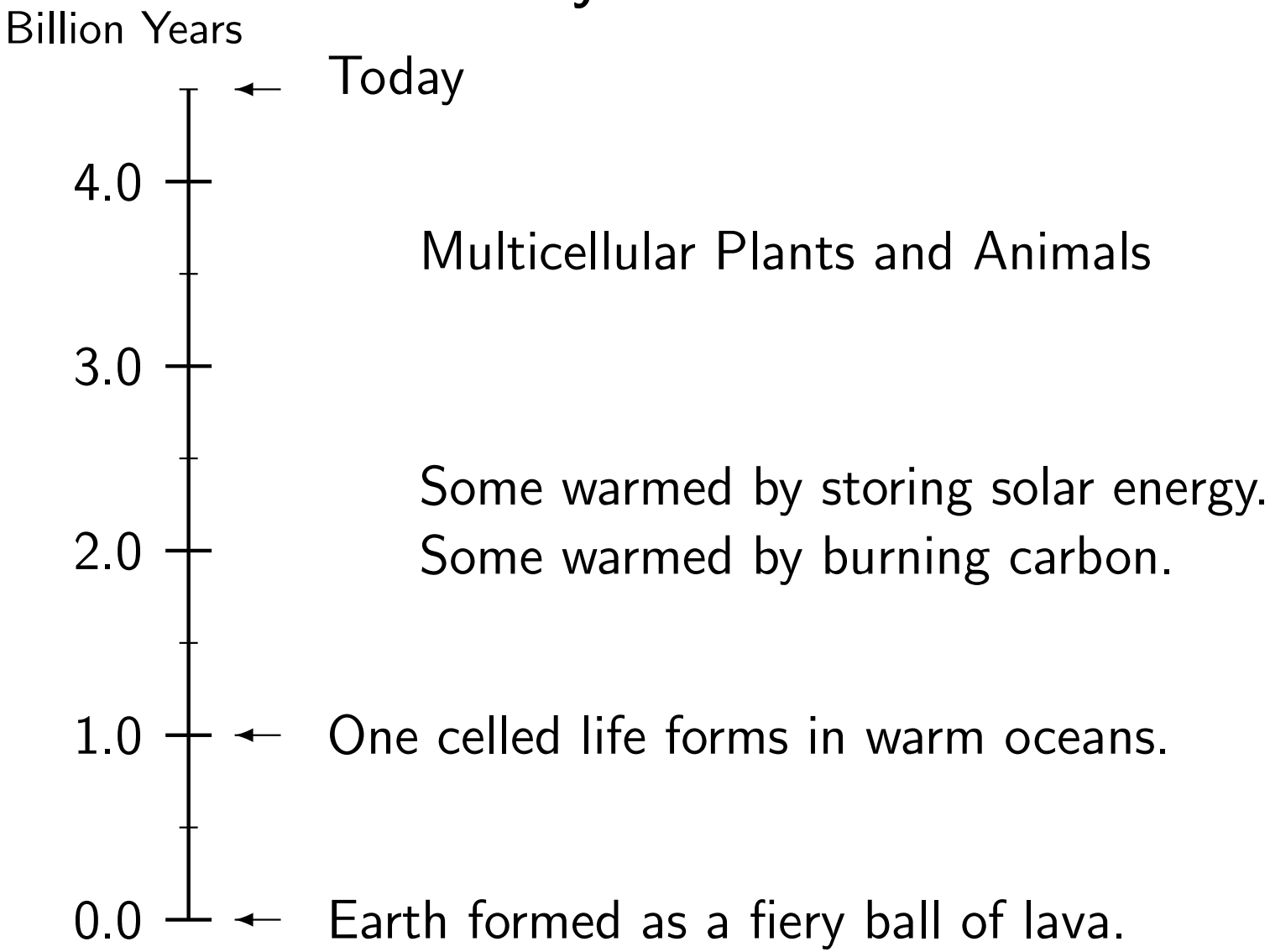
History of Life on Earth



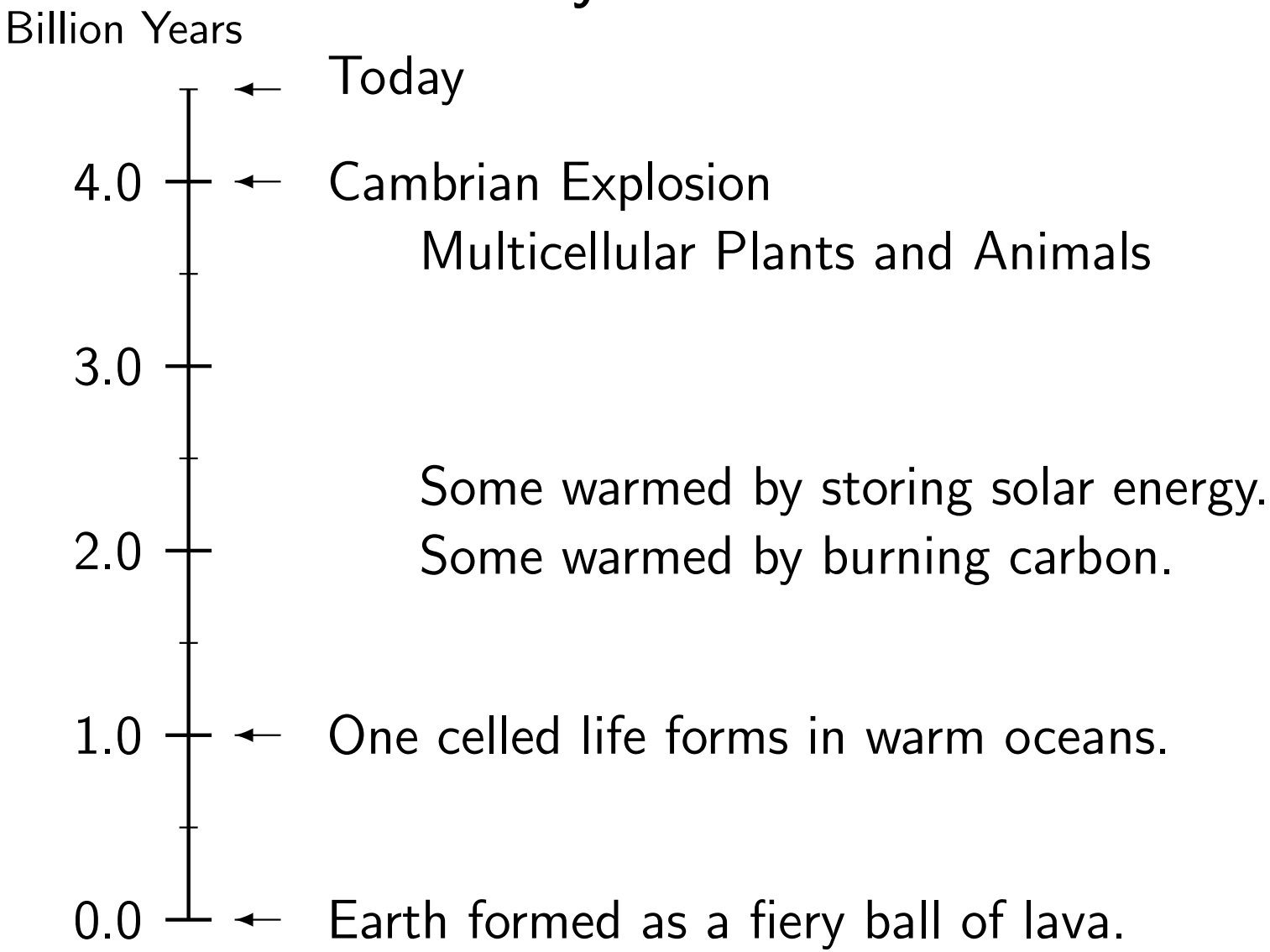
History of Life on Earth



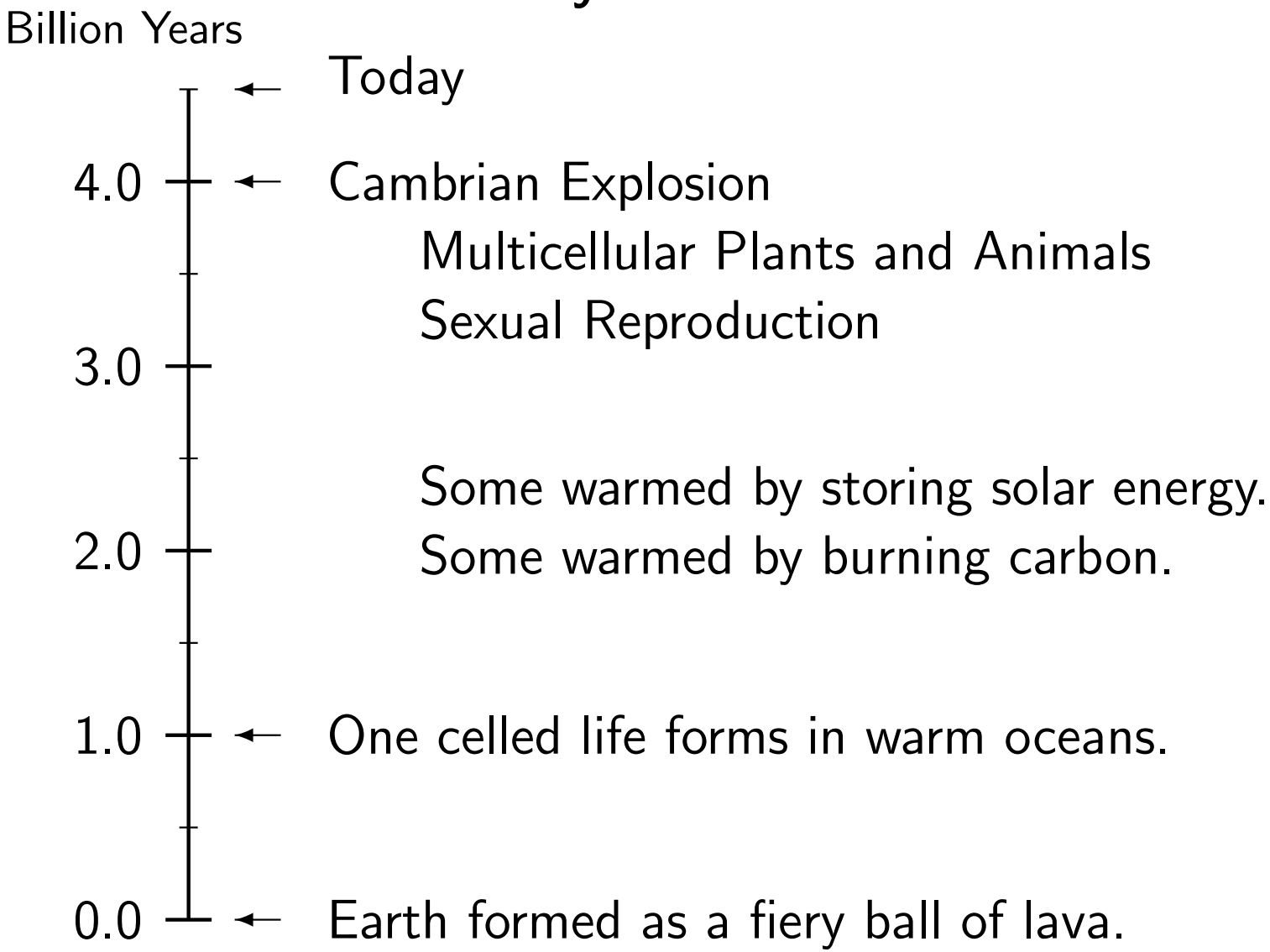
History of Life on Earth



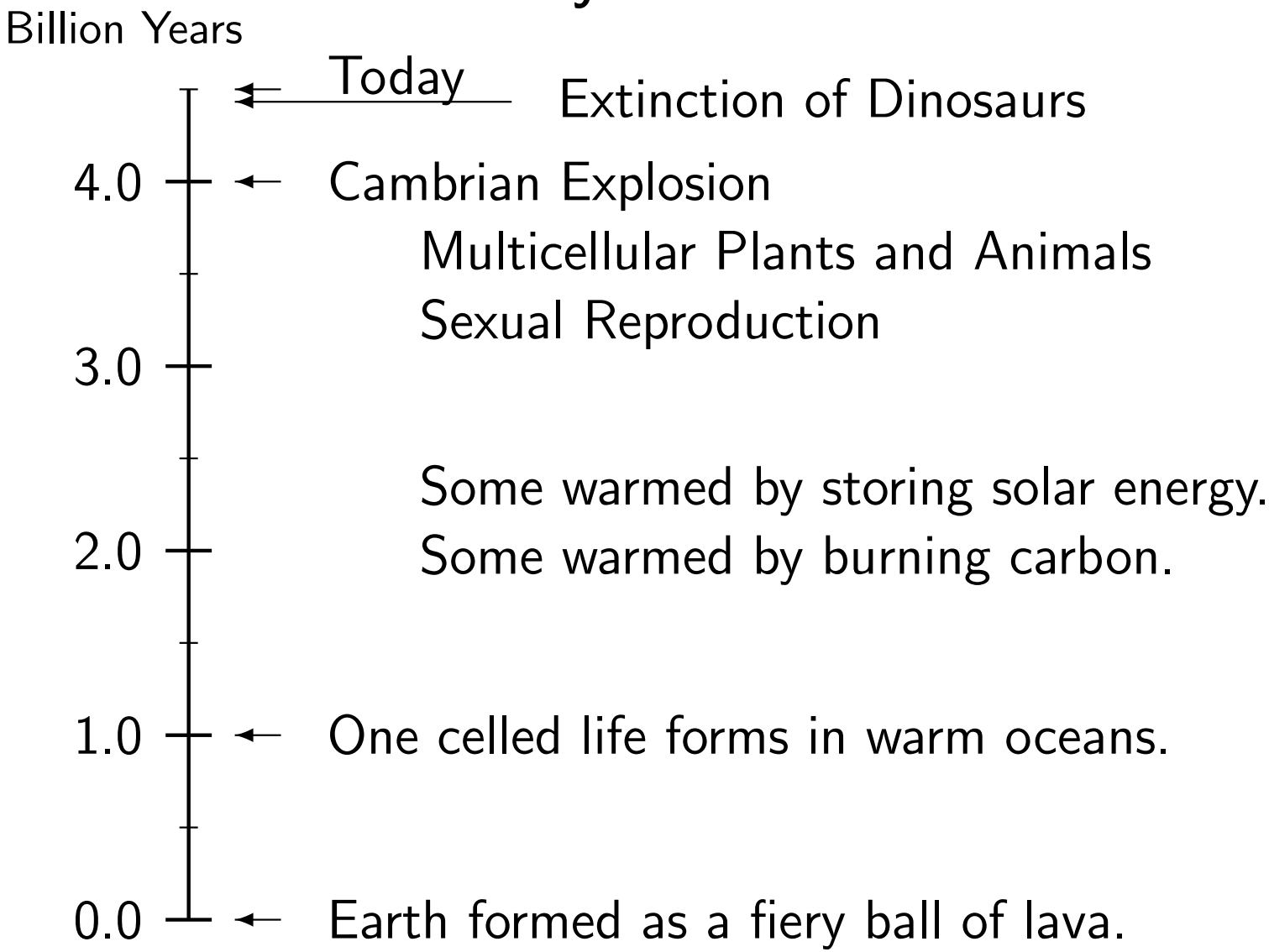
History of Life on Earth



History of Life on Earth



History of Life on Earth



Why is biological evolution so slow?

Why is biological evolution so slow?

Genotype: a design for an individual.

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Evolutionary process:

.. genotypes $\overset{build}{\Rightarrow}$ phenotypes $\overset{select}{\Rightarrow}$ genotypes $\overset{build}{\Rightarrow}$ phenotypes ..

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Evolutionary process:

.. genotypes $\overset{build}{\Rightarrow}$ phenotypes $\overset{select}{\Rightarrow}$ genotypes $\overset{build}{\Rightarrow}$ phenotypes ..

Humans:

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Evolutionary process:

.. genotypes $\overset{build}{\Rightarrow}$ phenotypes $\overset{select}{\Rightarrow}$ genotypes $\overset{build}{\Rightarrow}$ phenotypes ..

Humans: 30 years

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Evolutionary process:

.. genotypes $\overset{build}{\Rightarrow}$ phenotypes $\overset{select}{\Rightarrow}$ genotypes $\overset{build}{\Rightarrow}$ phenotypes ..

Humans:	30 years	3 days

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Evolutionary process:

.. genotypes $\overset{build}{\Rightarrow}$ phenotypes $\overset{select}{\Rightarrow}$ genotypes $\overset{build}{\Rightarrow}$ phenotypes ..

Humans:	30 years	3 days	30 years

Why is biological evolution so slow?

Genotype: a design for an individual.

Phenotype: individual built from that design.

Evolutionary process:

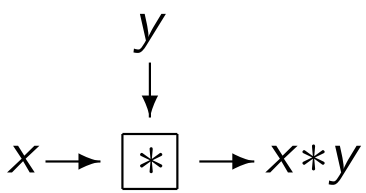
.. genotypes $\overset{build}{\Rightarrow}$ phenotypes $\overset{select}{\Rightarrow}$ genotypes $\overset{build}{\Rightarrow}$ phenotypes ..

Humans:	30 years	3 days	30 years

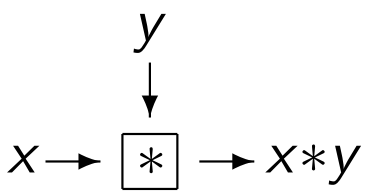
Conclusion: Biological evolution is slow because it uses phenotypes to test the fitness of genotypes.

2. Switching Circuits

A **binary gate** is a small electronic chip with a set, say $\{0, 1, 2\}$, of input/output values. Given inputs values x and y it produces an output denoted by $x*y$



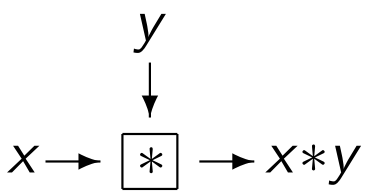
A **binary gate** is a small electronic chip with a set, say $\{0, 1, 2\}$, of input/output values. Given inputs values x and y it produces an output denoted by $x*y$ whose value is given by the $*$-**table**.



	$*$	0	1	2
	0	2	1	2
$x \rightarrow$	1	1	0	0
	2	0	0	1

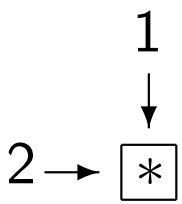
$y \downarrow$

A **binary gate** is a small electronic chip with a set, say $\{0, 1, 2\}$, of input/output values. Given inputs values x and y it produces an output denoted by $x*y$ whose value is given by the $*$-**table**.

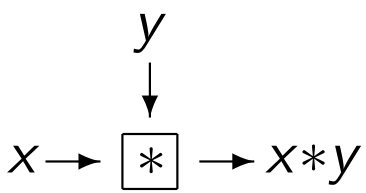


	$*$	0	1	2
	0	2	1	2
$x \rightarrow$	1	1	0	0
	2	0	0	1

For example,

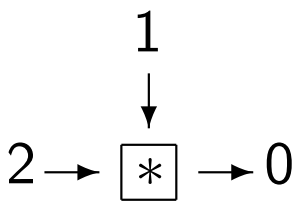


A **binary gate** is a small electronic chip with a set, say $\{0, 1, 2\}$, of input/output values. Given inputs values x and y it produces an output denoted by $x*y$ whose value is given by the $*$-**table**.



	$*$	0	1	2
	0	2	1	2
$x \rightarrow$	1	1	0	0
	2	0	0	1

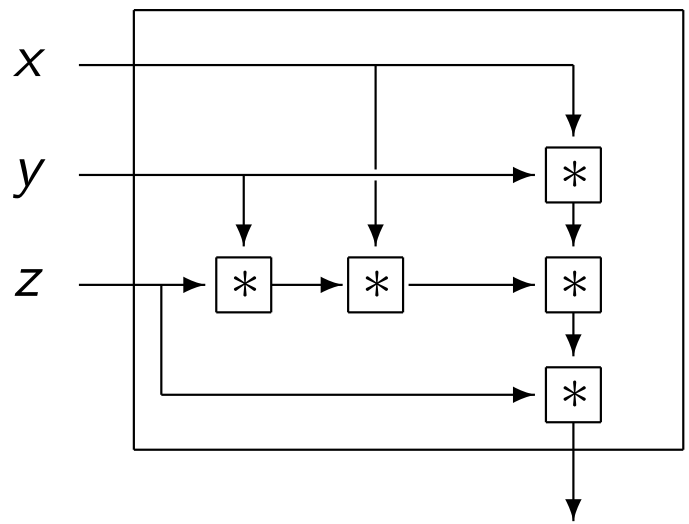
For example,



$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

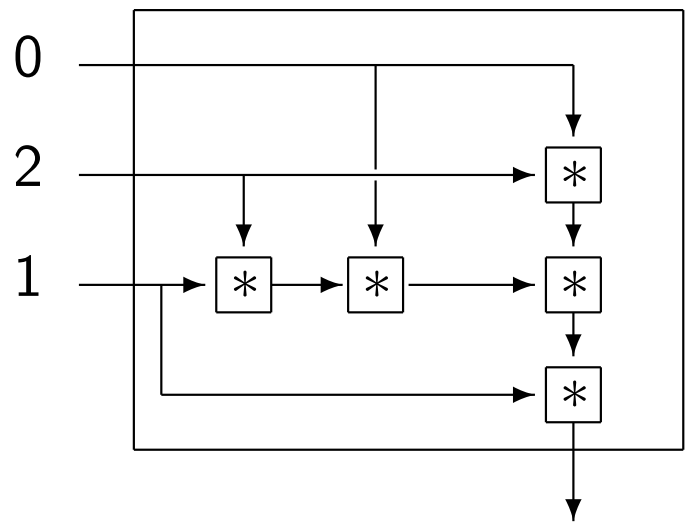
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



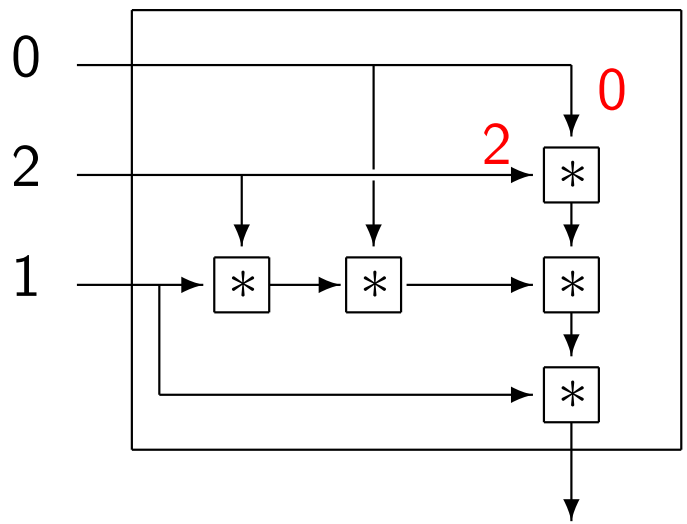
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



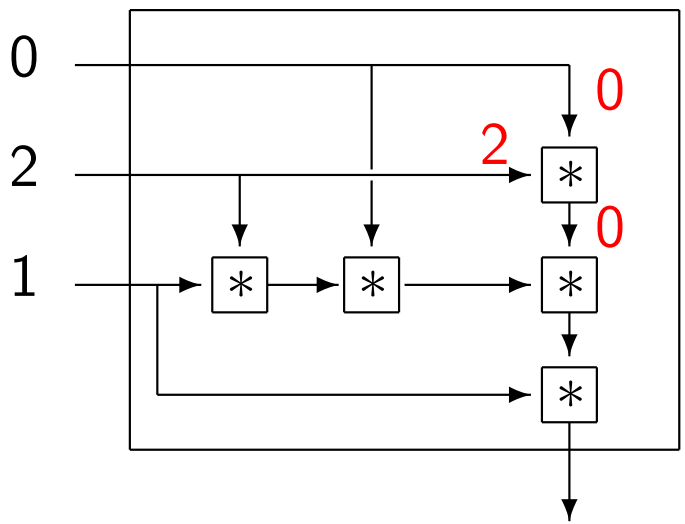
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



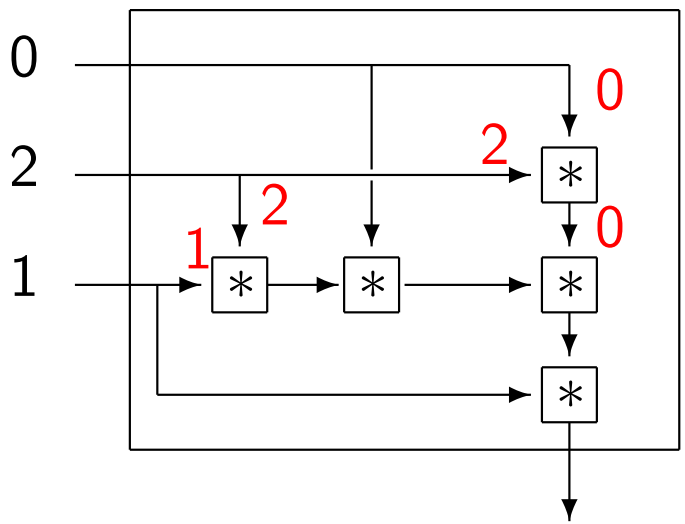
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



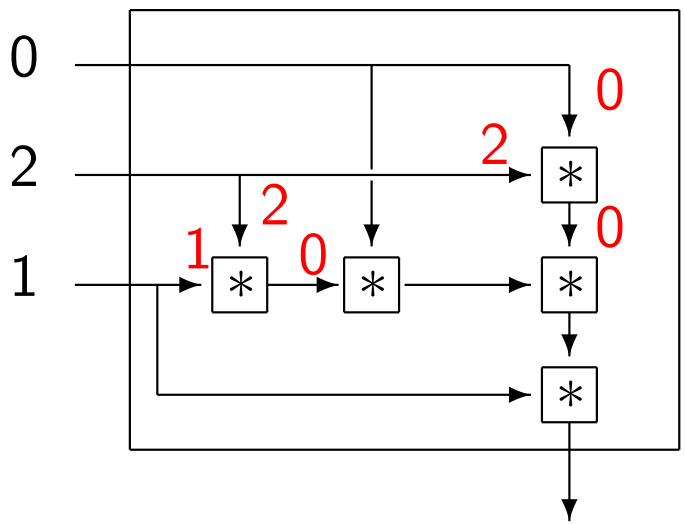
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



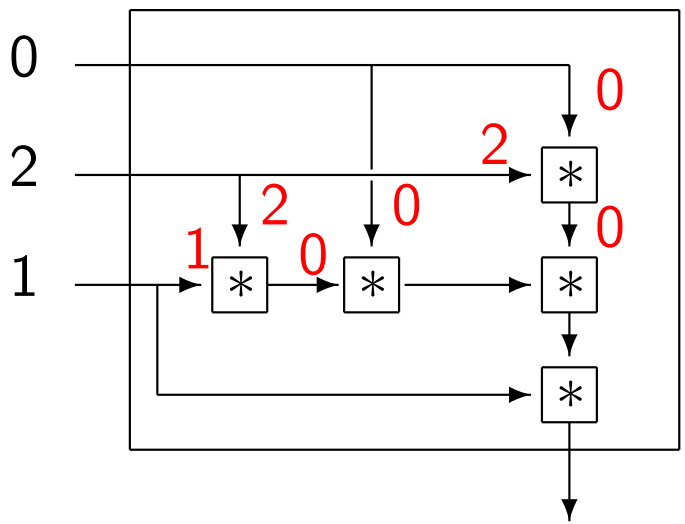
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



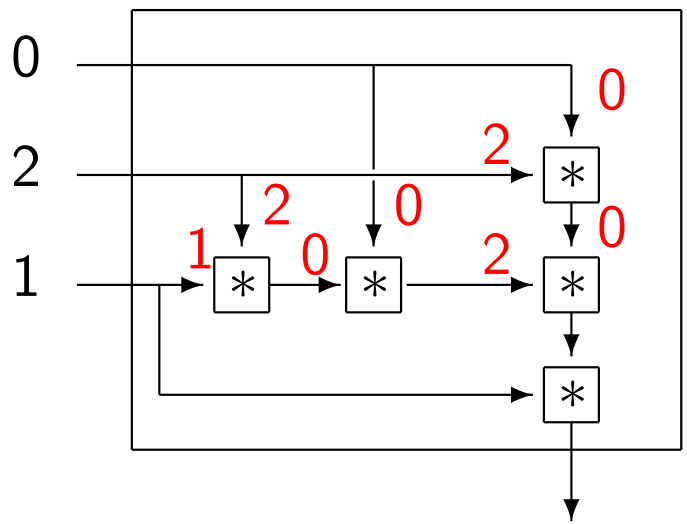
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



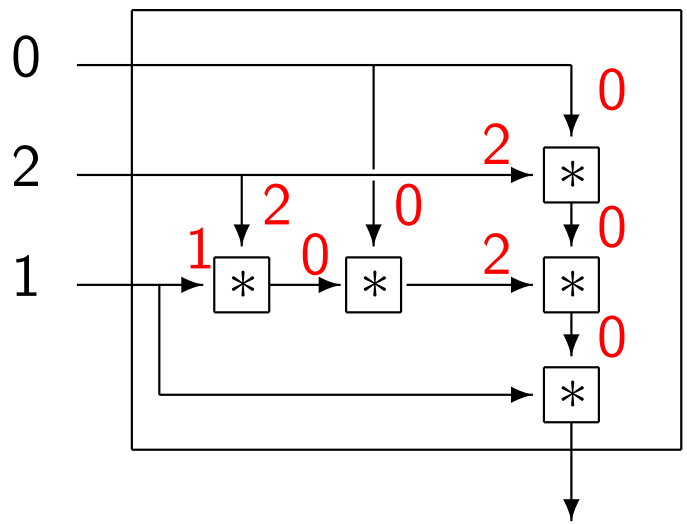
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



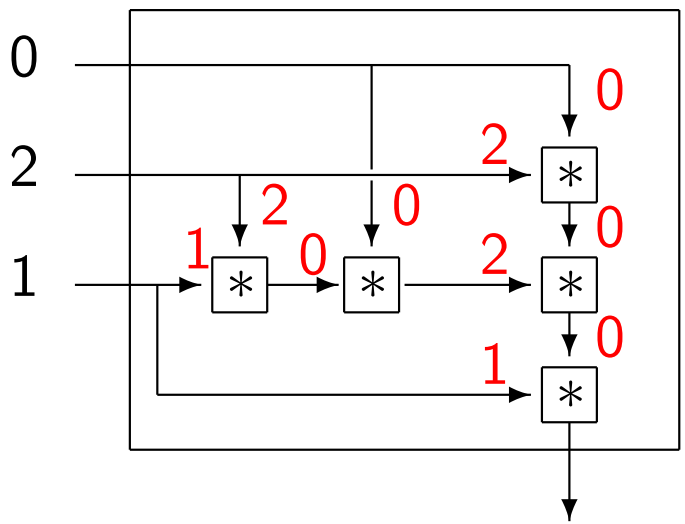
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



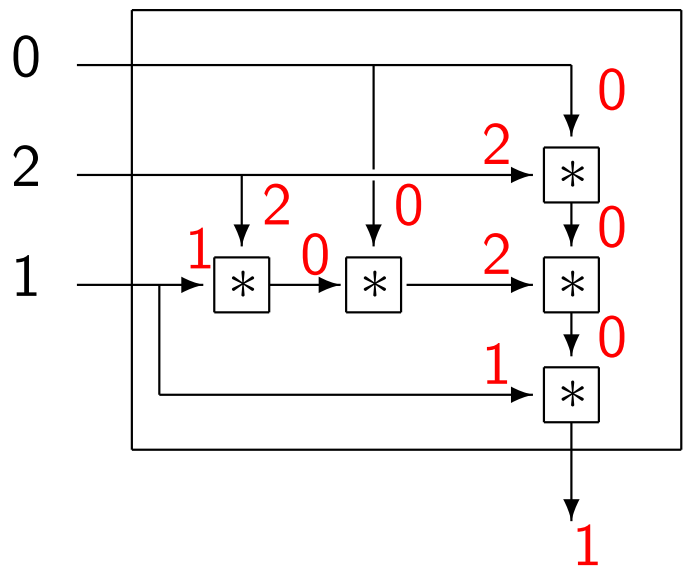
$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Switching Circuit



$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

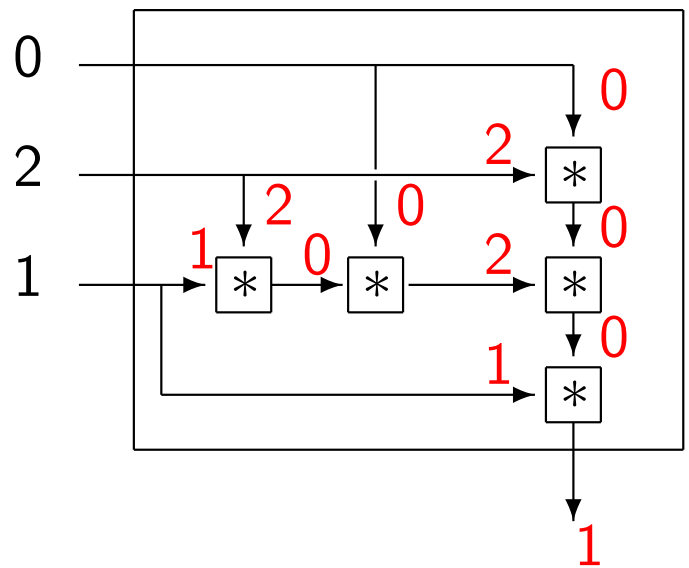
Switching Circuit



x	y	z	Output
0	0	0	
0	0	1	
0	0	2	
0	1	0	
0	1	1	
0	1	2	
0	2	0	
0	2	1	1
0	2	2	
1	0	0	
1	0	1	
:	:	:	
2	1	2	
2	2	0	
2	2	1	
2	2	2	

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

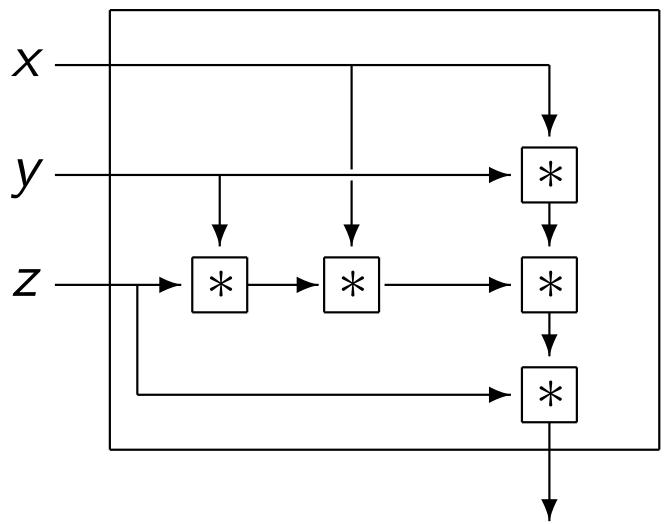
Switching Circuit



x	y	z	Output
0	0	0	2
0	0	1	1
0	0	2	0
0	1	0	2
0	1	1	1
0	1	2	0
0	2	0	2
0	2	1	1
0	2	2	0
1	0	0	1
.	.	.	.
.	.	.	.
2	1	2	0
2	2	0	2
2	2	1	1
2	2	2	0

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

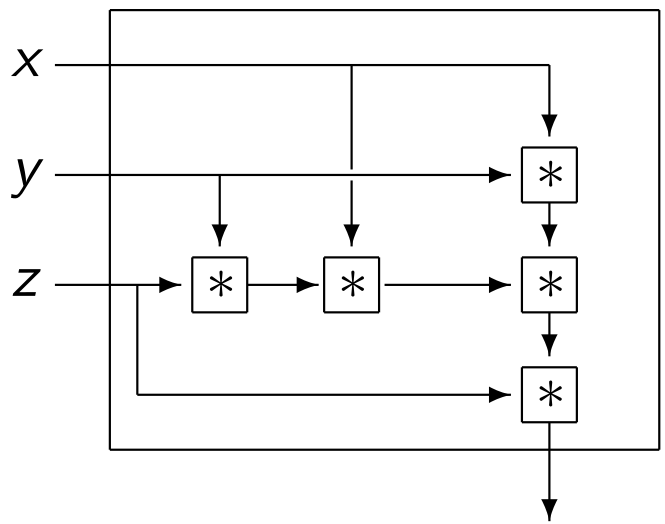
Switching Circuit

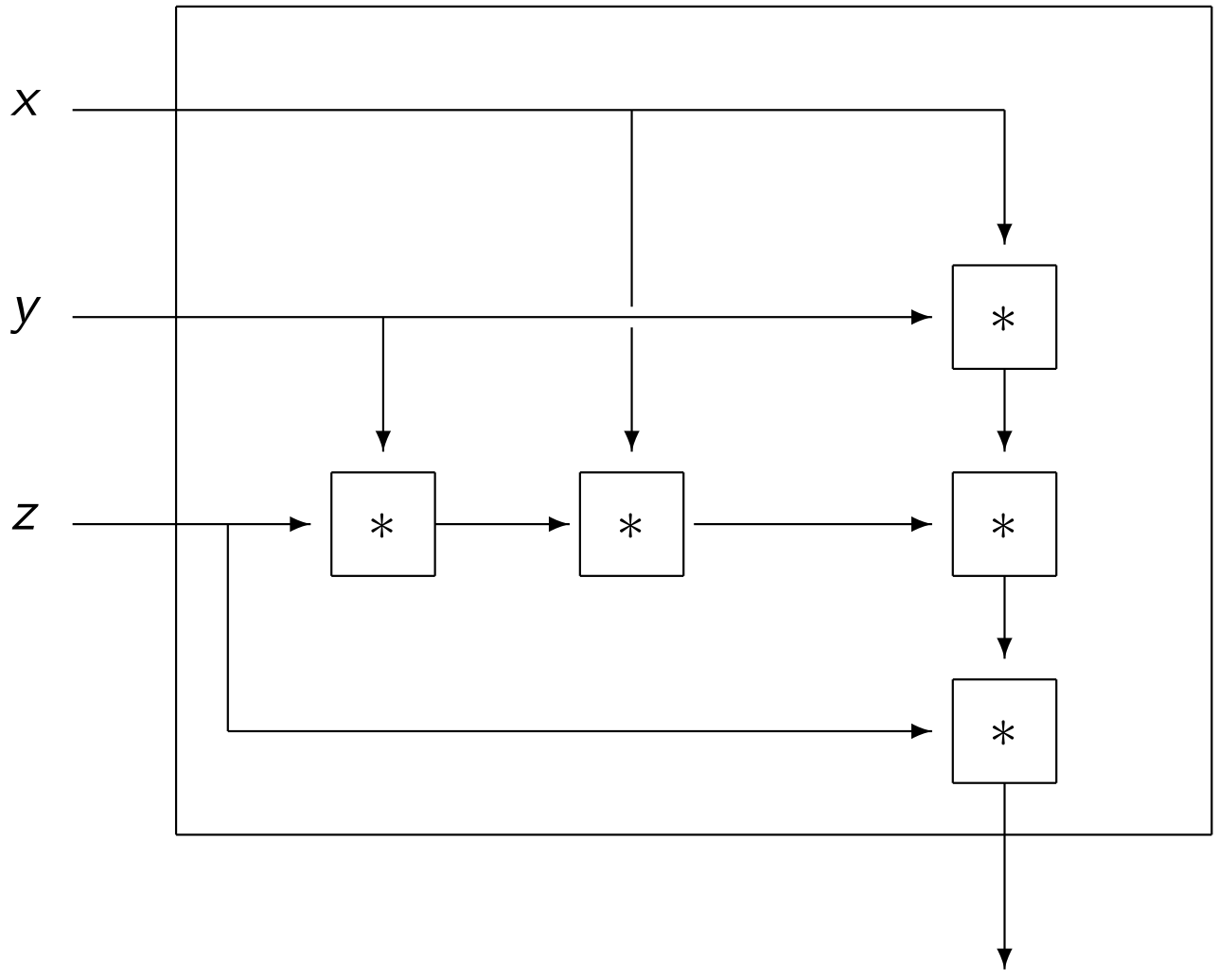


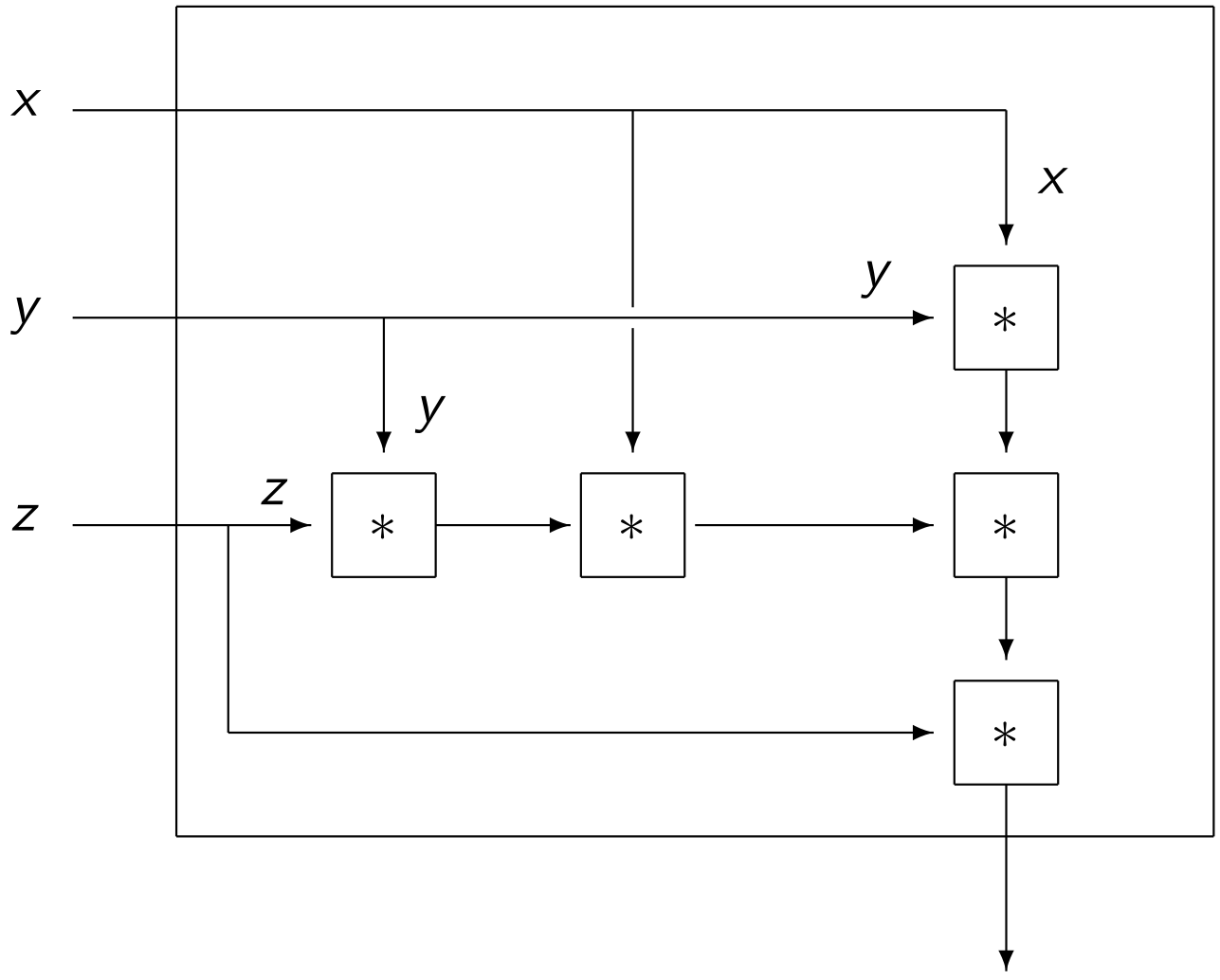
x	y	z	Output
0	0	0	3
0	0	1	0
0	0	2	2
0	0	3	1
0	1	0	2
0	1	1	0
0	1	2	3
0	1	3	3
0	2	0	1
0	2	1	2
0	2	2	0
:	:	:	:
3	3	0	0
3	3	1	3
3	3	2	1
3	3	3	2

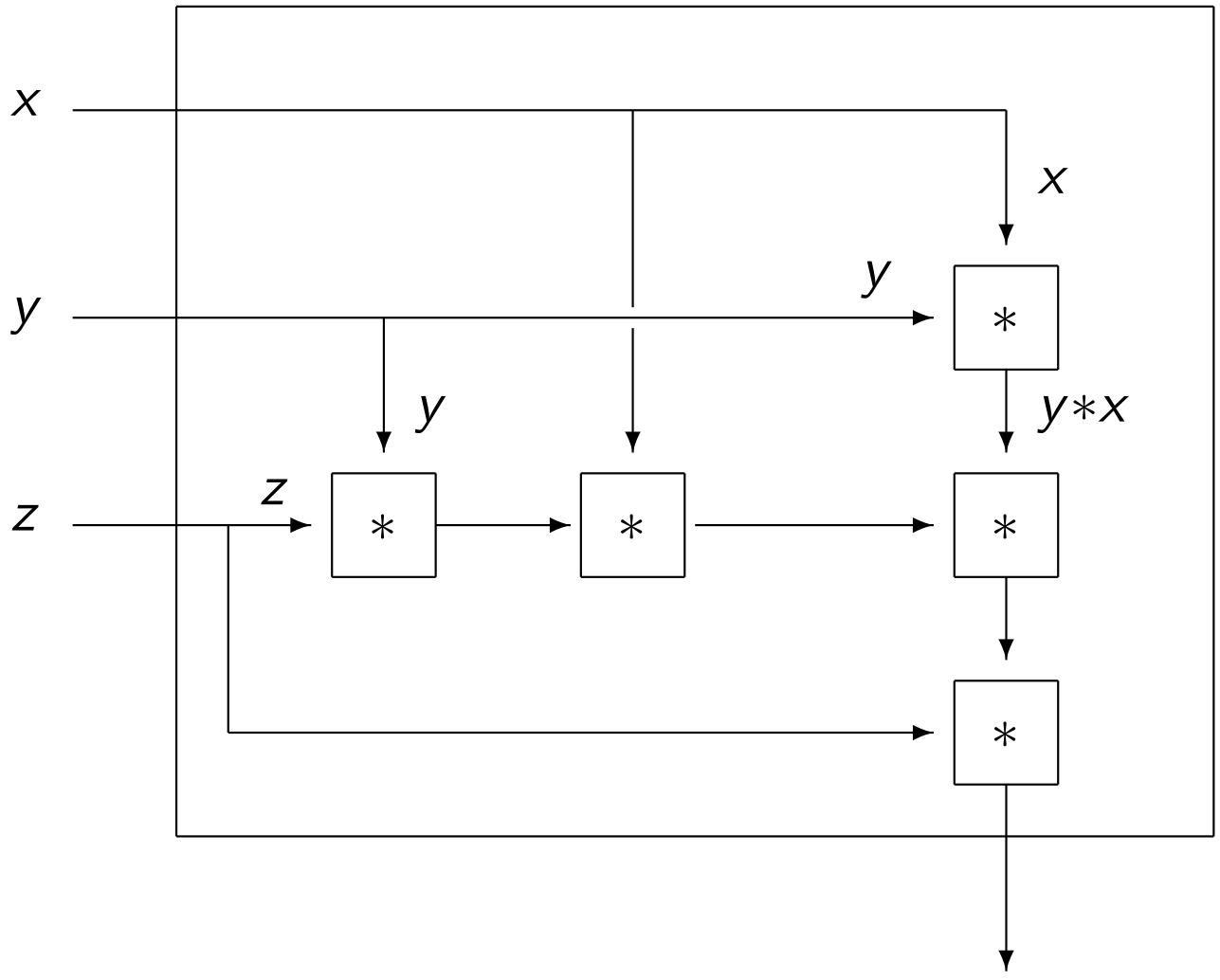
$*$	0	1	2	3
0	3	0	0	3
1	2	2	1	0
2	3	0	2	0
3	2	0	1	3

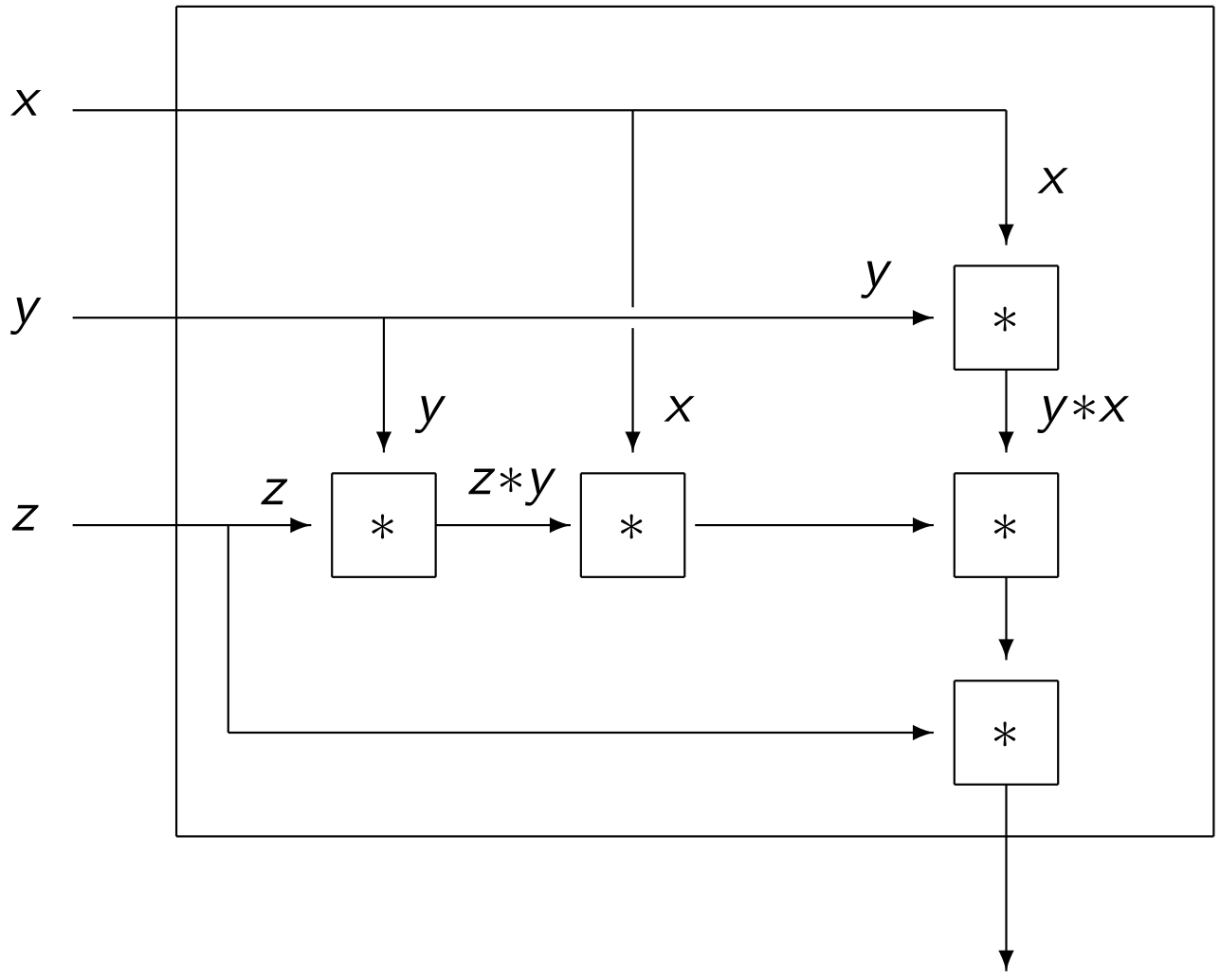
Switching Circuit

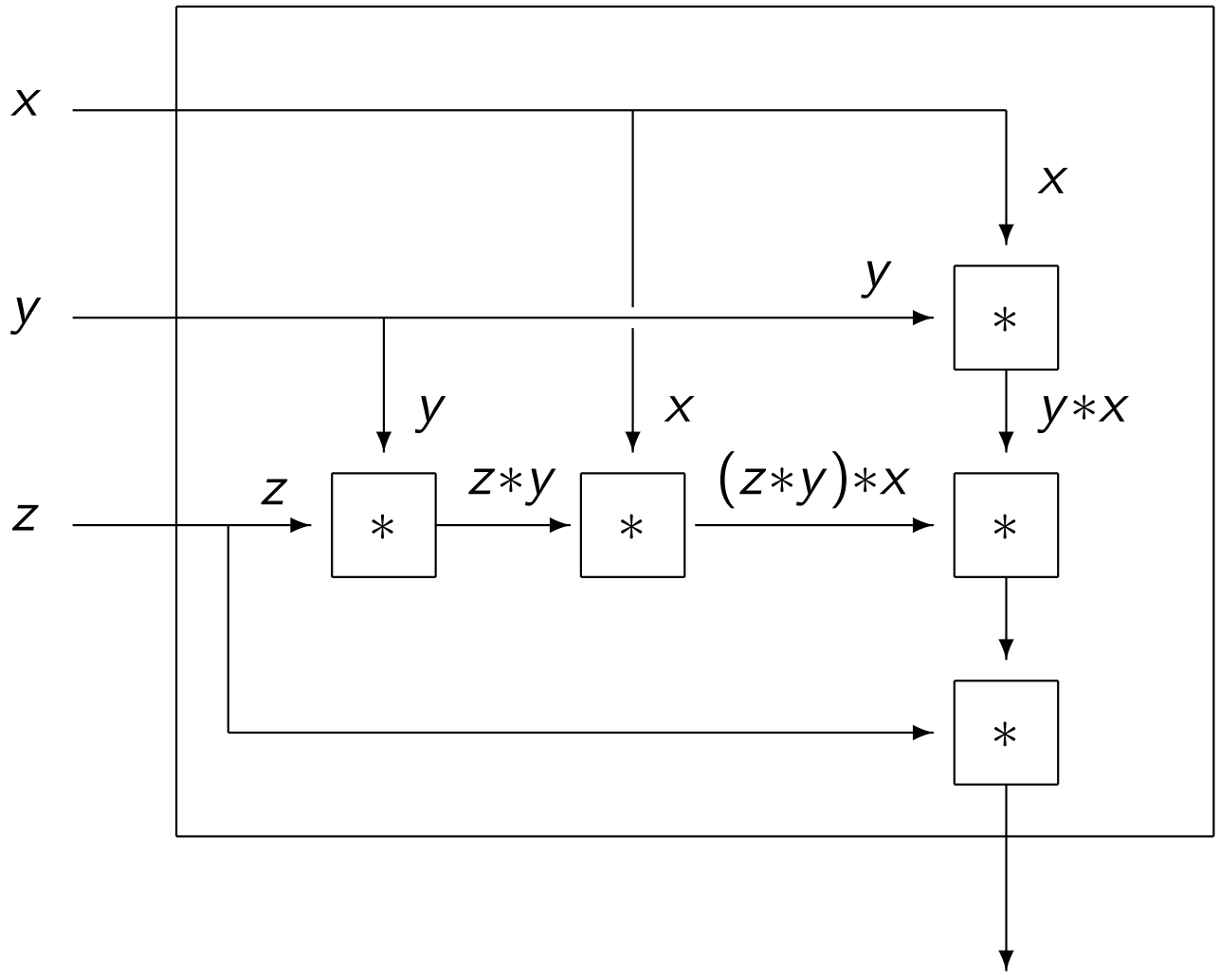


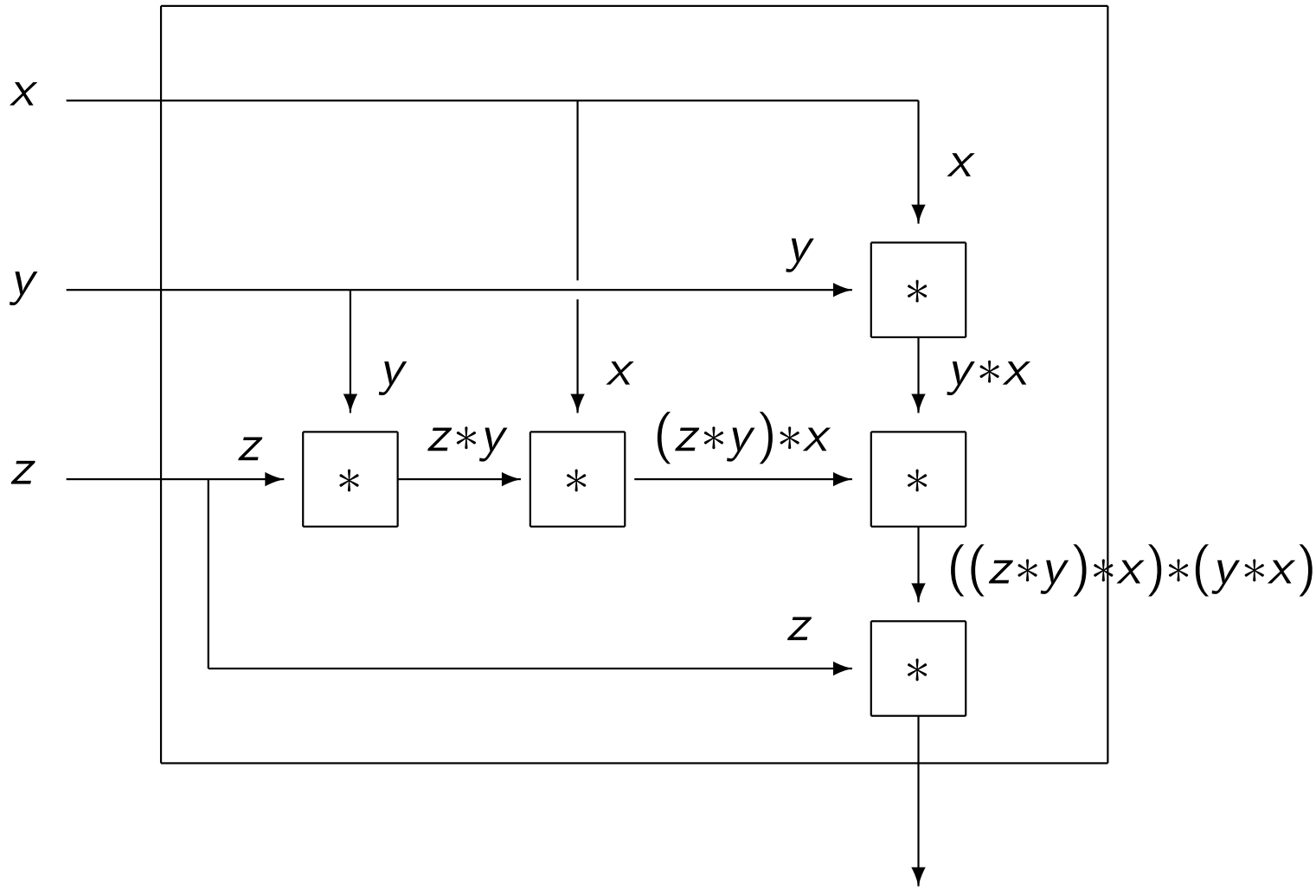

x
y
z
*
*
*
*
*

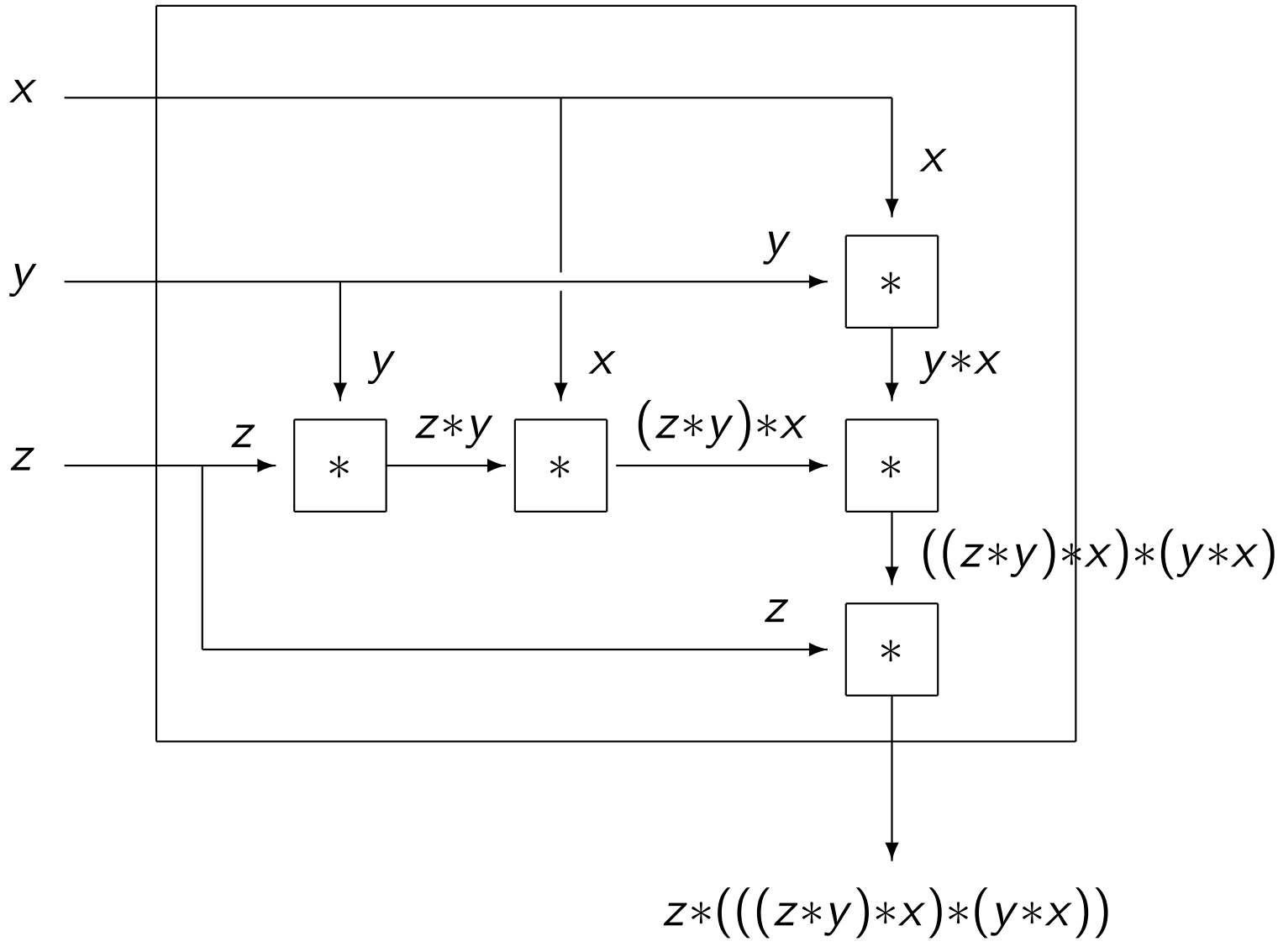

x
y
z
x
y
y
z
*
*
*
*
*


x
y
z
x
y
y
z
*
*
*
*
*
y∗x


x
y
z
x
y
*
y
x
y∗x
z
*
z∗y
*
*
*

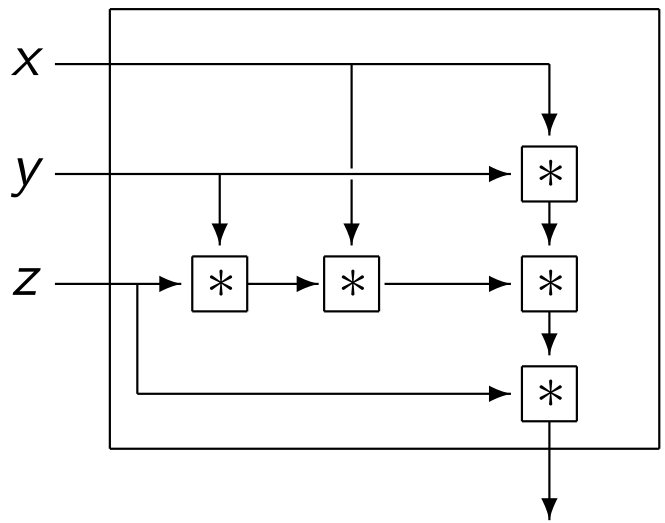

x
y
z
x
y
y
x
y∗x
z
z∗y
(z∗y)∗x
∗
∗
∗
∗
∗


x
y
z
x
y
y
x
z
z∗y
(z∗y)∗x
y∗x
((z∗y)∗x)∗(y∗x)
z
∗
∗
∗
∗
∗


x
y
z
x
y
y
x
z
z∗y
(z∗y)∗x
y∗x
((z∗y)∗x)∗(y∗x)
z
z∗(((z∗y)∗x)∗(y∗x))
∗
∗
∗
∗

The expression $z*(((z*y)*x)*(y*x))$ is both

- the circuit output if its inputs are x, y and z and the gate operation is $*$ and
- a complete design or blueprint for the circuit itself.



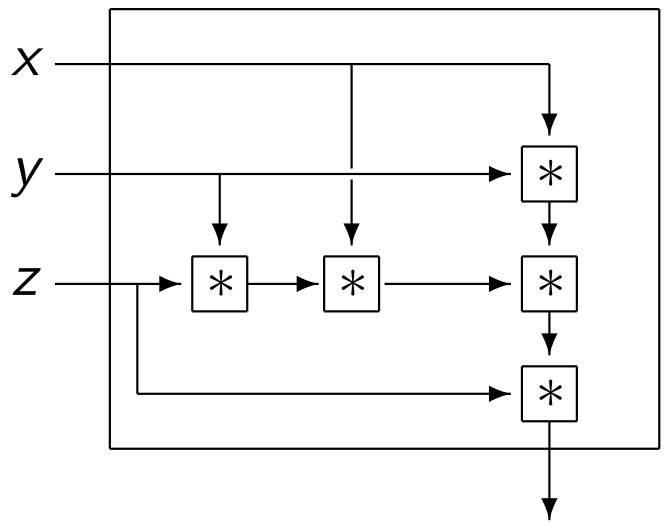
The expression $z*(((z*y)*x)*(y*x))$ is both

- the circuit output if its inputs are x, y and z and the gate operation is $*$ and
- a complete design or blueprint for the circuit itself.

We will call it the

DNA

for the circuit.

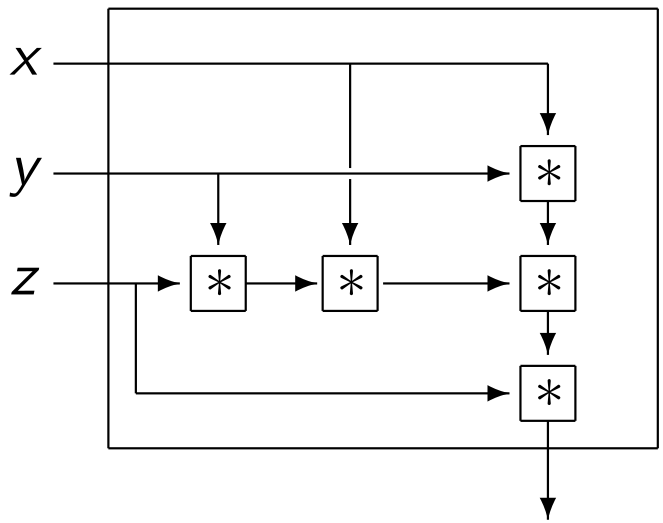


3. Design Problem

x	y	z	Output
0	0	0	
0	0	1	
0	0	2	
0	1	0	
0	1	1	
0	1	2	
0	2	0	
0	2	1	???
0	2	2	
1	0	0	
1	0	1	
:	:	:	:
2	1	2	
2	2	0	
2	2	1	
2	2	2	

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

$\Longleftarrow$

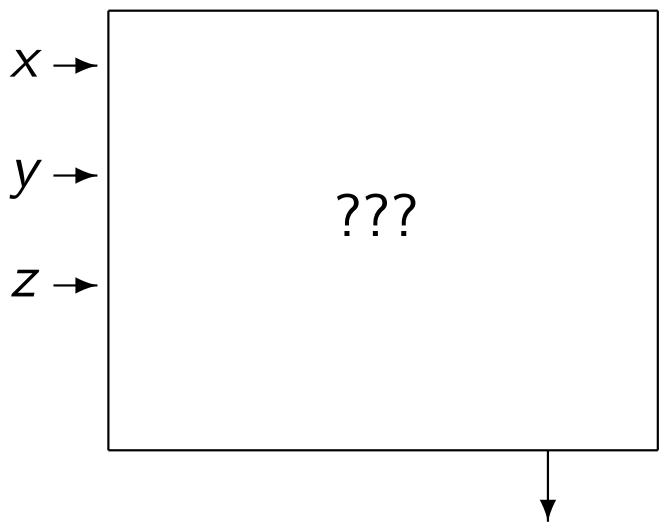


x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	0
0	1	0	2
0	1	1	1
0	1	2	0
0	2	0	2
0	2	1	1
0	2	2	0
1	0	0	1
1	0	1	2
:	:	:	:
2	1	2	0
2	2	0	2
2	2	1	1
2	2	2	0

$\Longrightarrow$

Design Problem

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

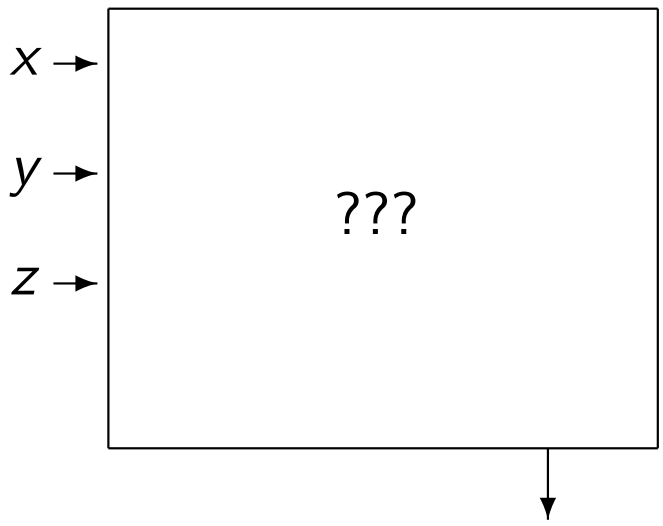


x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	0
0	1	0	2
0	1	1	1
0	1	2	0
0	2	0	2
0	2	1	1
0	2	2	0
1	0	0	1
1	0	1	2
:	:	:	:
2	1	2	0
2	2	0	2
2	2	1	1
2	2	2	0

Design Problem

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

$\Longrightarrow$



- The number of different possible target sequences is $3^{27} \approx 10^{13}$.

x	y	z	Target
0	0	0	0
0	0	1	0
0	0	2	2
0	1	0	0
0	1	1	1
0	1	2	2
0	2	0	1
0	2	1	0
0	2	2	1
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	0
2	2	0	1
2	2	1	1
2	2	2	0

Design Problem

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

- A fast computer can solve the design problem by a **random search**, generating and testing about a million circuits per second until a solution is found.

x	y	z	Target
0	0	0	0
0	0	1	0
0	0	2	2
0	1	0	0
0	1	1	1
0	1	2	2
0	2	0	1
0	2	1	0
0	2	2	1
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	0
2	2	0	1
2	2	1	1
2	2	2	0

Design Problem

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

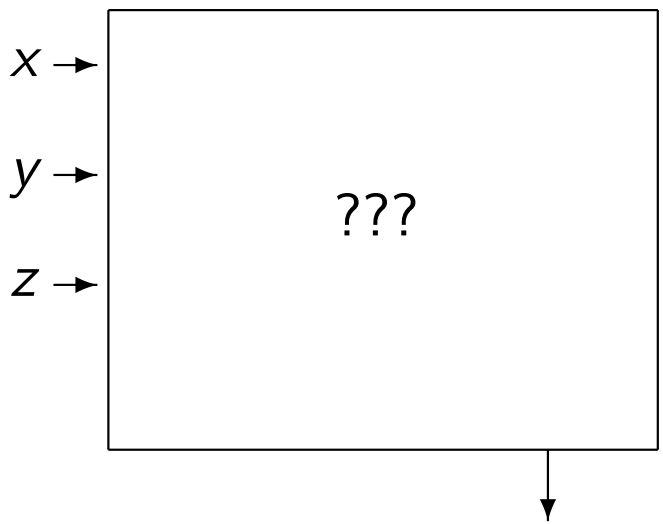
- A fast computer can solve the design problem by a **random search**, generating and testing about a million circuits per second until a solution is found.
- The expected time to find a solution would be $10^{13}/10^{6} \approx 10^{7}$ seconds, or about **4 months**.

4. One Parent Evolution

— Lee Spector

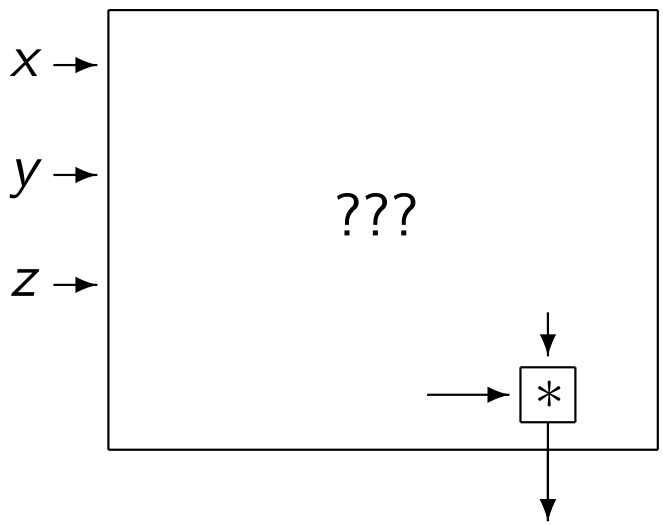
x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	2
0	1	0	1
0	1	1	0
0	1	2	2
0	2	0	0
0	2	1	2
0	2	2	0
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	2
2	2	1	0
2	2	2	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1



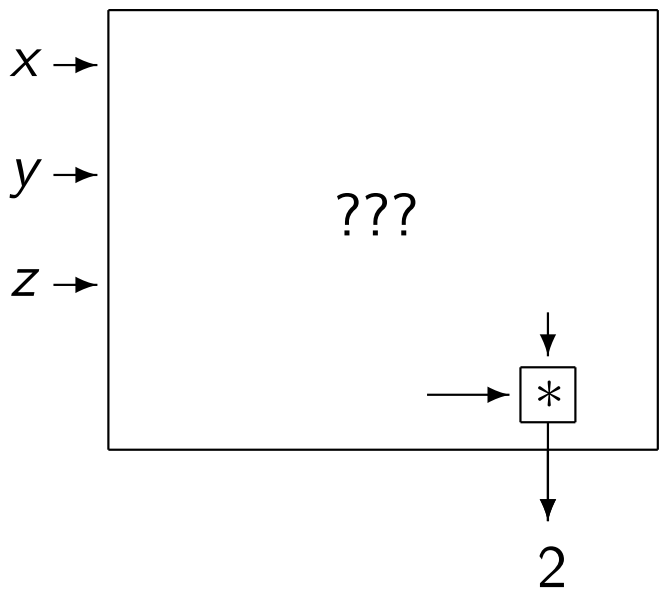
x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	2
0	1	0	1
0	1	1	0
0	1	2	2
0	2	0	0
0	2	1	2
0	2	2	0
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	2
2	2	1	0
2	2	2	1

$*$	0	1	2
0	2	2	2
1	2	2	2
2	2	2	2



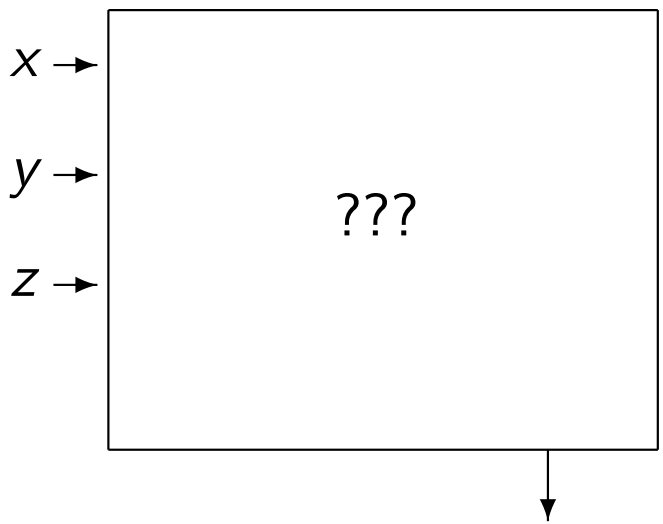
x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	2
0	1	0	1
0	1	1	0
0	1	2	2
0	2	0	0
0	2	1	2
0	2	2	0
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	2
2	2	1	0
2	2	2	1

$*$	0	1	2
0	2	2	2
1	2	2	2
2	2	2	2



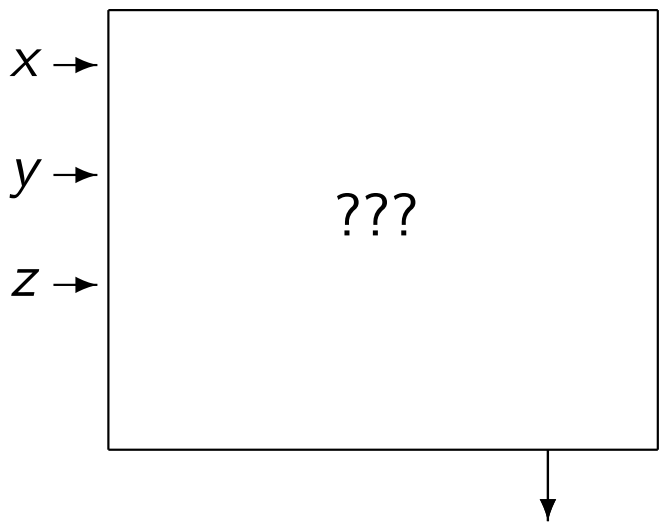
x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	2
0	1	0	1
0	1	1	0
0	1	2	2
0	2	0	0
0	2	1	2
0	2	2	0
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	2
2	2	1	0
2	2	2	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1



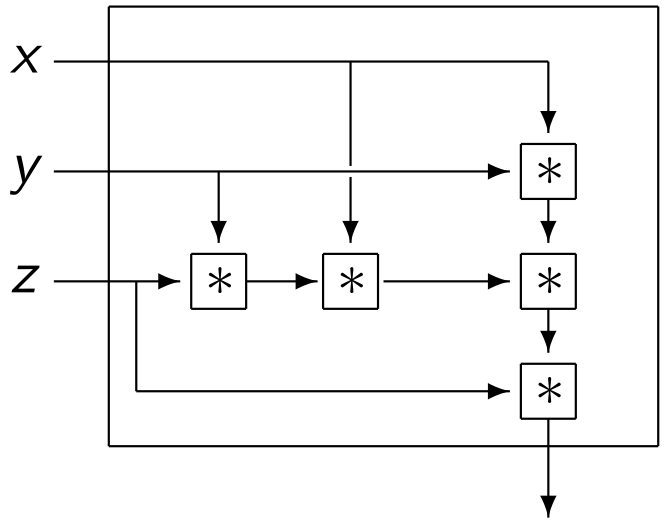
x	y	z	Target
0	0	0	2
0	0	1	1
0	0	2	2
0	1	0	1
0	1	1	0
0	1	2	2
0	2	0	0
0	2	1	2
0	2	2	0
1	0	0	2
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	2
2	2	1	0
2	2	2	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

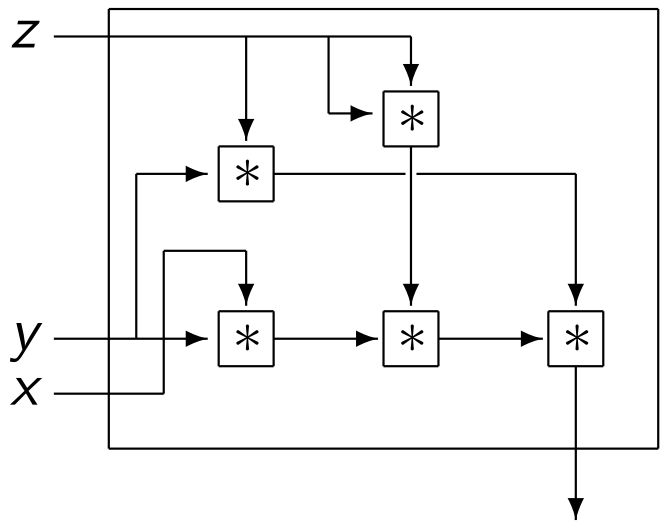


x	y	z	Target	SC1	SC2
0	0	0	2	2	0
0	0	1	1	1	1
0	0	2	2	0	2
0	1	0	1	2	1
0	1	1	0	1	1
0	1	2	2	0	2
0	2	0	0	2	0
0	2	1	2	1	0
0	2	2	0	0	0
1	0	0	2	1	2
1	0	1	0	0	1
:	:	:	:	:	:
2	1	2	2	0	2
2	2	0	2	2	2
2	2	1	0	1	1
2	2	2	1	0	1

Switching Circuit 1



Switching Circuit 2



x	y	z	Target	SC1	SC2
0	0	0	2	2	0
0	0	1	1	1	1
0	0	2	2	0	2
0	1	0	1	2	1
0	1	1	0	1	1
0	1	2	2	0	2
0	2	0	0	2	0
0	2	1	2	1	0
0	2	2	0	0	0
1	0	0	2	1	2
1	0	1	0	0	1
:	:	:	:	:	:
2	1	2	2	0	2
2	2	0	2	2	2
2	2	1	0	1	1
2	2	2	1	0	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

x	y	z	Target	SC1	SC2
0	0	0	2	2	0
0	0	1	1	1	1
0	0	2	2	0	2
0	1	0	1	2	1
0	1	1	0	1	1
0	1	2	2	0	2
0	2	0	0	2	0
0	2	1	2	1	0
0	2	2	0	0	0
1	0	0	2	1	2
1	0	1	0	0	1
:	:	:	:	:	:
2	1	2	2	0	2
2	2	0	2	2	2
2	2	1	0	1	1
2	2	2	1	0	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Given any target and gate, we define the **fitness** of a circuit to be the number of its outputs that match the target.

x	y	z	Target	SC1	SC2
0	0	0	2	2	0
0	0	1	1	1	1
0	0	2	2	0	2
0	1	0	1	2	1
0	1	1	0	1	1
0	1	2	2	0	2
0	2	0	0	2	0
0	2	1	2	1	0
0	2	2	0	0	0
1	0	0	2	1	2
1	0	1	0	0	1
:	:	:	:	:	:
2	1	2	2	0	2
2	2	0	2	2	2
2	2	1	0	1	1
2	2	2	1	0	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Given any target and gate, we define the **fitness** of a circuit to be the number of its outputs that match the target.

fitness of SC1 ≈ 9
fitness of SC2 ≈ 18

x	y	z	Target	SC1	SC2
0	0	0	2	2	0
0	0	1	1	1	1
0	0	2	2	0	2
0	1	0	1	2	1
0	1	1	0	1	1
0	1	2	2	0	2
0	2	0	0	2	0
0	2	1	2	1	0
0	2	2	0	0	0
1	0	0	2	1	2
1	0	1	0	0	1
:	:	:	:	:	:
2	1	2	2	0	2
2	2	0	2	2	2
2	2	1	0	1	1
2	2	2	1	0	1

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Given any target and gate, we define the **fitness** of a circuit to be the number of its outputs that match the target.

fitness of SC1 ≈ 9
fitness of SC2 ≈ 18

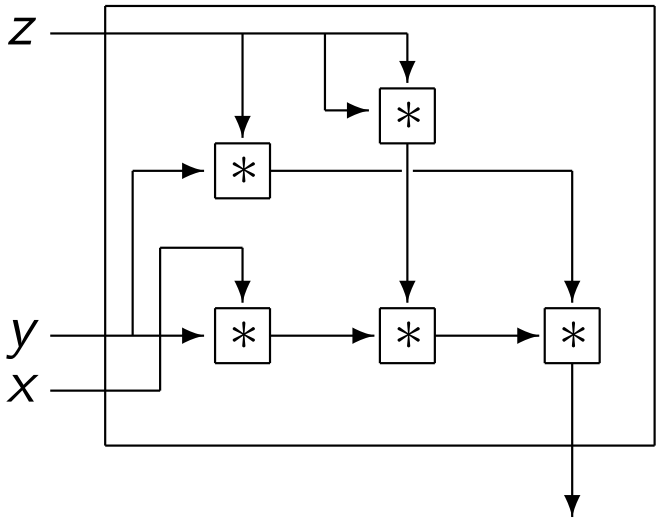
A solution to the Design Problem will be any switching circuit with a fitness of 27.

$$((y * x) * (z * z)) * (y * z)$$

DNA Sequences

$$((y * x) * (z * z)) * (y * z)$$

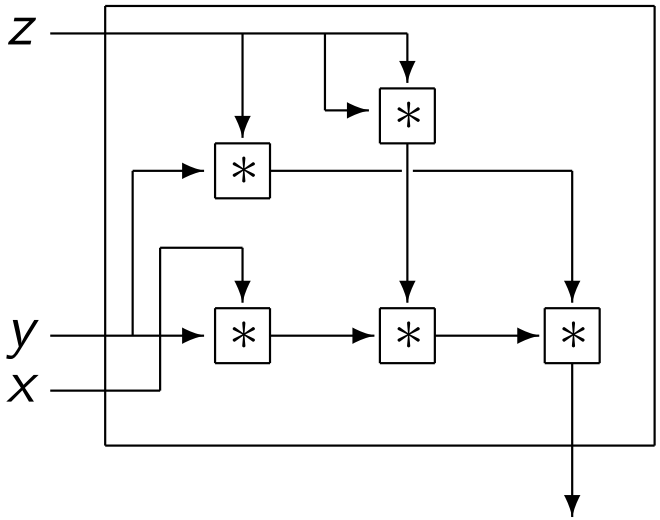
DNA Sequences
(Genotypes)



Physical Switching Circuits
(Phenotypes)

$\longleftarrow ((y * x) * (z * z)) * (y * z)$

DNA Sequences
(Genotypes)



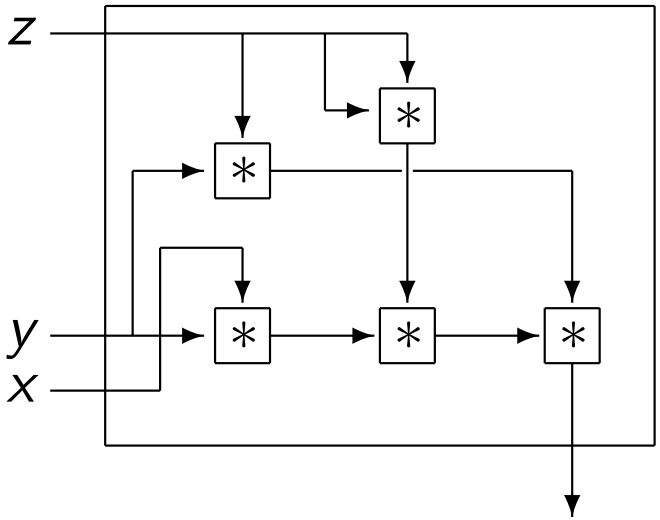
Physical Switching Circuits
(Phenotypes)

$\leftarrow ((y * x) * (z * z)) * (y * z)$

DNA Sequences
(Genotypes)

$\downarrow$

Calculate Fitnesses



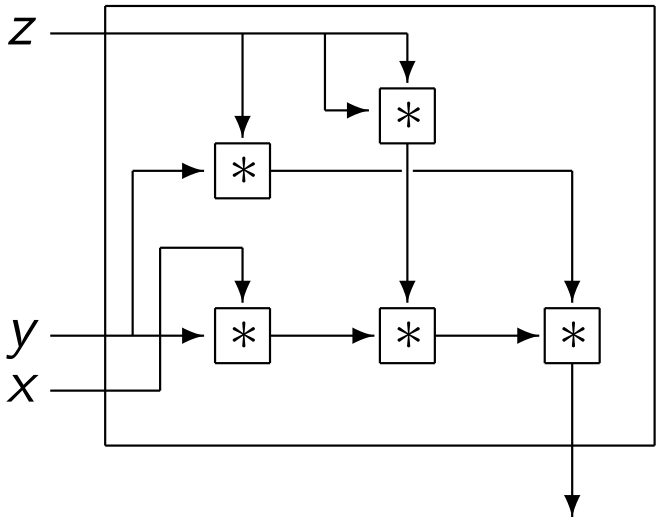
Physical Switching Circuits
(Phenotypes)

$\longleftarrow$ $((y * x) * (z * z)) * (y * z)$

DNA Sequences
(Genotypes)

$\downarrow$

Calculate Fitnesses $\longrightarrow$ Generate DNA for the Fittest Circuits



Physical Switching Circuits
(Phenotypes)

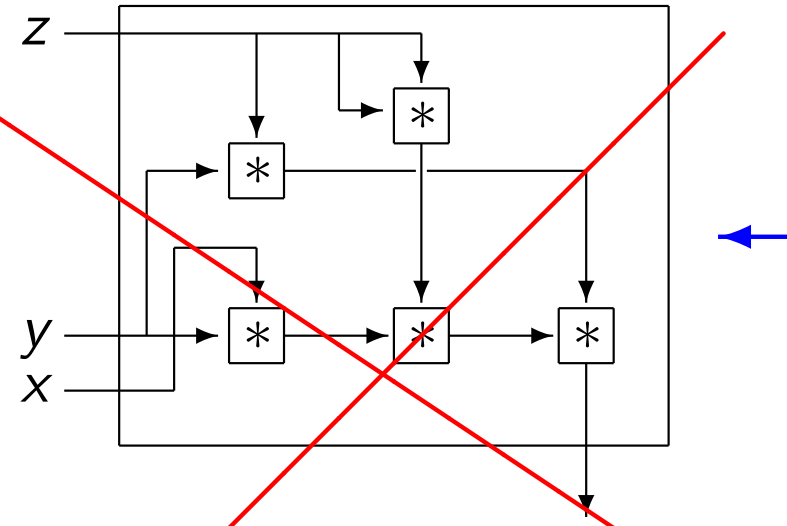
$((y * x) * (z * z)) * (y * z)$

DNA Sequences
(Genotypes)

Add
Variations

Calculate Fitnesses

Generate DNA for the
Fittest Circuits



Physical Switching Circuits
(Phenotypes)

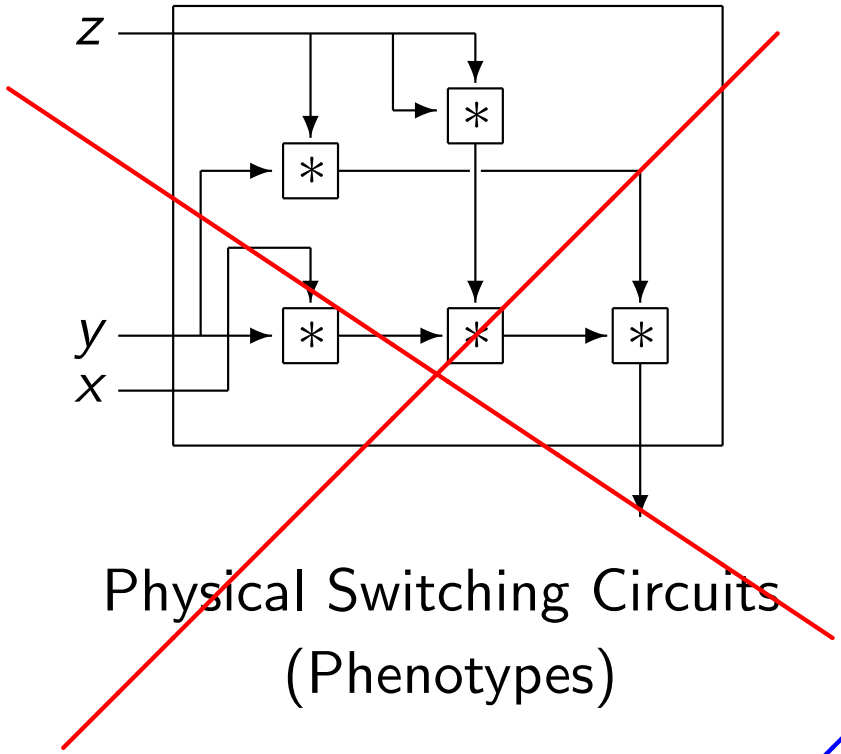
$((y * x) * (z * z)) * (y * z)$

DNA Sequences
(Genotypes)

Add
Variations

Calculate Fitnesses

Generate DNA for the
Fittest Circuits



Physical Switching Circuits
(Phenotypes)

$((y * x) * (z * z)) * (y * z)$

DNA Sequences
(Genotypes)

Add
Variations

Calculate Fitnesses → Generate DNA for the Fittest Circuits

x	y	z	Target
0	0	0	0
0	0	1	1
0	0	2	2
0	1	0	0
0	1	1	0
0	1	2	0
0	2	0	0
0	2	1	0
0	2	2	0
1	0	0	1
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	0
2	2	1	1
2	2	2	2

Possible target sequences:
$3^{27} \approx 10^{13}$

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

x	y	z	Target
0	0	0	0
0	0	1	1
0	0	2	2
0	1	0	0
0	1	1	0
0	1	2	0
0	2	0	0
0	2	1	0
0	2	2	0
1	0	0	1
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	0
2	2	1	1
2	2	2	2

Possible target sequences:
$3^{27} \approx 10^{13}$

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Typical solution to design problem, generated by **random search:**

x	y	z	Target
0	0	0	0
0	0	1	1
0	0	2	2
0	1	0	0
0	1	1	0
0	1	2	0
0	2	0	0
0	2	1	0
0	2	2	0
1	0	0	1
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	0
2	2	1	1
2	2	2	2

Possible target sequences:
$3^{27} \approx 10^{13}$

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Typical solution to design problem, generated by **random search: 4 months**

x	y	z	Target
0	0	0	0
0	0	1	1
0	0	2	2
0	1	0	0
0	1	1	0
0	1	2	0
0	2	0	0
0	2	1	0
0	2	2	0
1	0	0	1
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	0
2	2	1	1
2	2	2	2

Possible target sequences:
$3^{27} \approx 10^{13}$

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Typical solution to design problem, generated

by **random search: 4 months**

by **evolutionary computation:**

x	y	z	Target
0	0	0	0
0	0	1	1
0	0	2	2
0	1	0	0
0	1	1	0
0	1	2	0
0	2	0	0
0	2	1	0
0	2	2	0
1	0	0	1
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	0
2	2	1	1
2	2	2	2

Possible target sequences:
$3^{27} \approx 10^{13}$

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Typical solution to design problem, generated

by **random search: 4 months**

by **evolutionary computation: 5 minutes**

$$((((((((x*(y*x))*x)*z)*(z*x))*((x*(z*(x*(z*y))))*z))*z)*z)*(z*((((x*(((z*z)*x)*(z*x)))*x)*y)*(((y*(z*(z*y)))*(((y*y)*x)*z))*(x*(((z*z)*x)*(z*(x*(z*y))))))))$$

x	y	z	Target
0	0	0	0
0	0	1	1
0	0	2	2
0	1	0	0
0	1	1	0
0	1	2	0
0	2	0	0
0	2	1	0
0	2	2	0
1	0	0	1
1	0	1	0
:	:	:	:
2	1	2	2
2	2	0	0
2	2	1	1
2	2	2	2

Possible target sequences:
$3^{27} \approx 10^{13}$

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Typical solution to design problem, generated

by **random search: 4 months**

by **evolutionary computation: 5 minutes**

$$((((((((x*(y*x))*x)*z)*(z*x))*((x*(z*(x*(z*y))))*z))*z)*z)*(z*((((x*(((z*z)*x)*(z*x)))*x)*y)*(((y*(z*(z*y)))*(((y*y)*x)*z))*(x*(((z*z)*x)*(z*(x*(z*y))))))))$$

(Won the $5000 gold metal award in the 2008 International HUMIE Competition sponsored by the Association for Computing Machinery.)

5. Two Parent Evolution

— Lee Spector

— Nick Falco

A gate is **term continuous** if small changes in a circuit built with that gate result in small changes it its output sequence.

A gate is **term continuous** if small changes in a circuit built with that gate result in small changes it its output sequence.

1. D. Clark, Evolution of algebraic terms 1: term to term operation continuity, *International Journal of Algebra and Computation*, Vol. 23, No. 5 (2013) 1175-1205.

A gate is **term continuous** if small changes in a circuit built with that gate result in small changes it its output sequence.

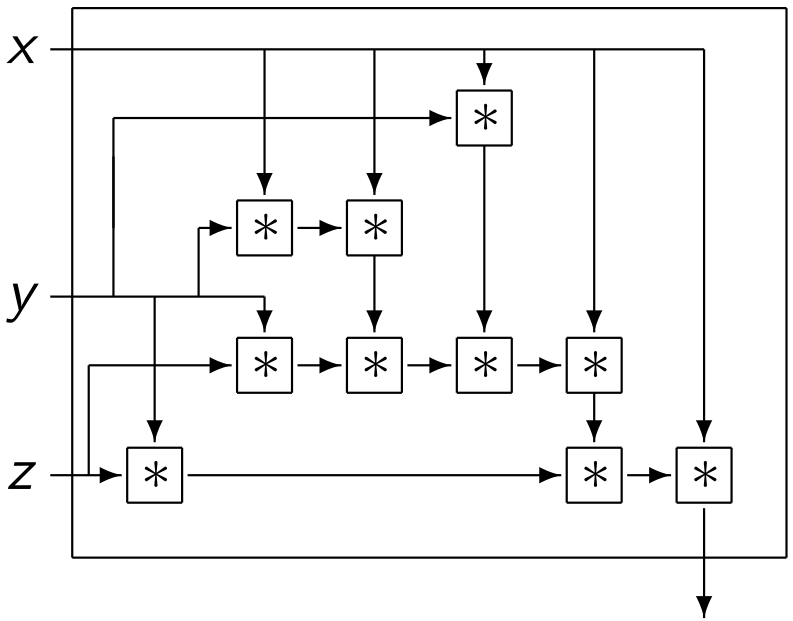
1. D. Clark, Evolution of algebraic terms 1: term to term operation continuity, *International Journal of Algebra and Computation*, Vol. 23, No. 5 (2013) 1175-1205.

Theorem. *A gate is term continuous if and only if it is AC and has NSR.*

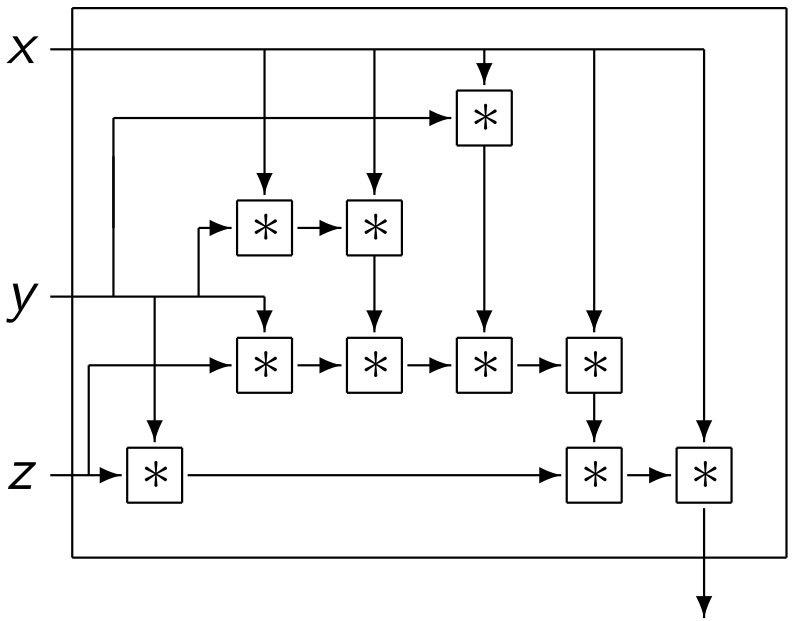
A gate is **term continuous** if small changes in a circuit built with that gate result in small changes it its output sequence.

1. D. Clark, Evolution of algebraic terms 1: term to term operation continuity, *International Journal of Algebra and Computation*, Vol. 23, No. 5 (2013) 1175-1205.

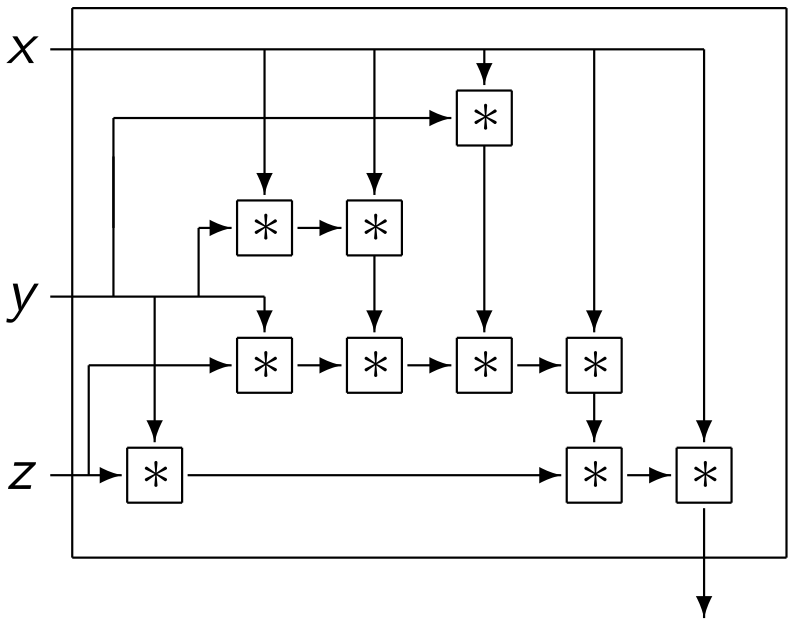
Theorem. *A gate is term continuous if and only if it is AC and has NSR.*



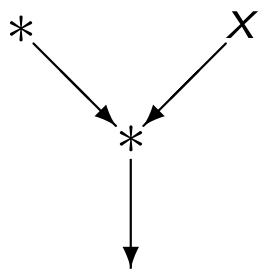
$$\Big[(z*y)*\Big(\Big(\big[(z*y)*((y*x)*x)\big]*(y*x)\Big)*x\Big)\Big]*x$$

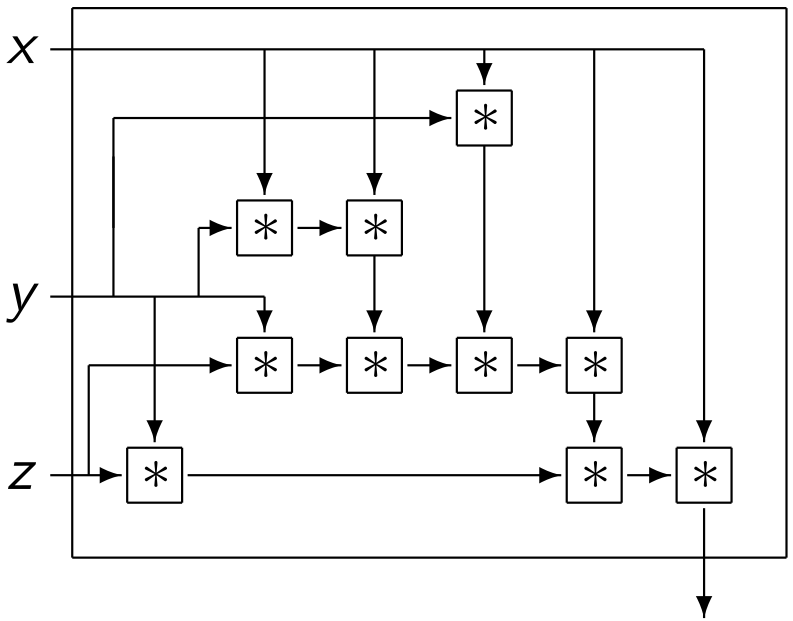


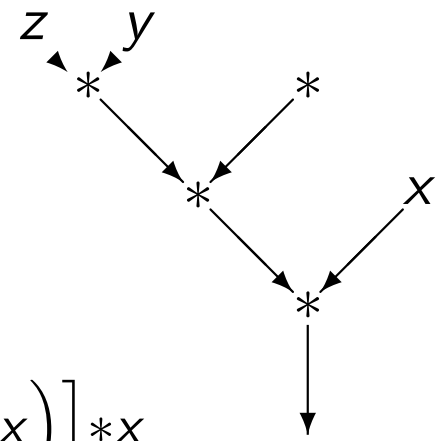
$$\left[(z*y)*\left(\left(\left[(z*y)*((y*x)*x)\right]*(y*x)\right)*x\right)\right]*x$$



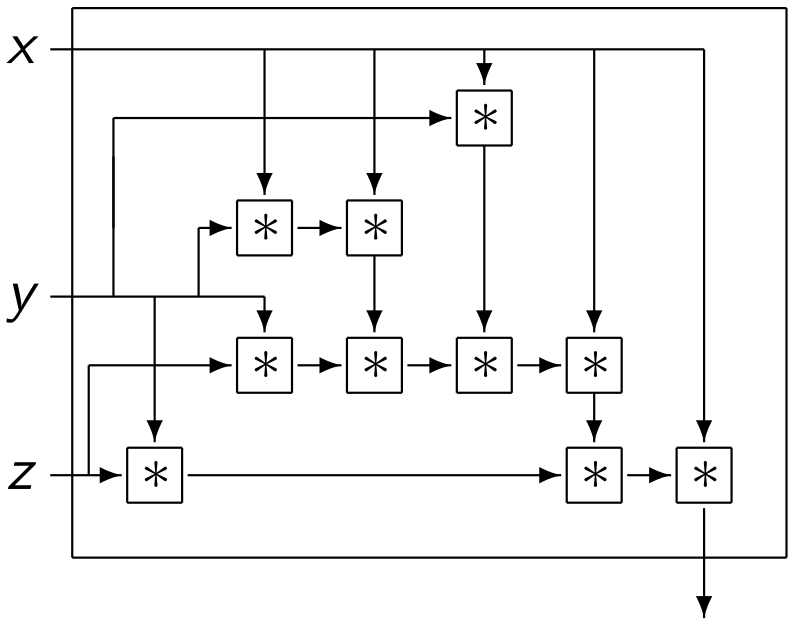
$$\left[(z*y)*\left(\left(\left[(z*y)*((y*x)*x)\right]*(y*x)\right)*x\right)\right]*x$$

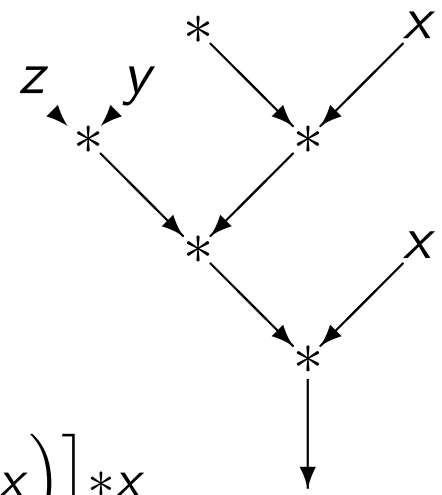




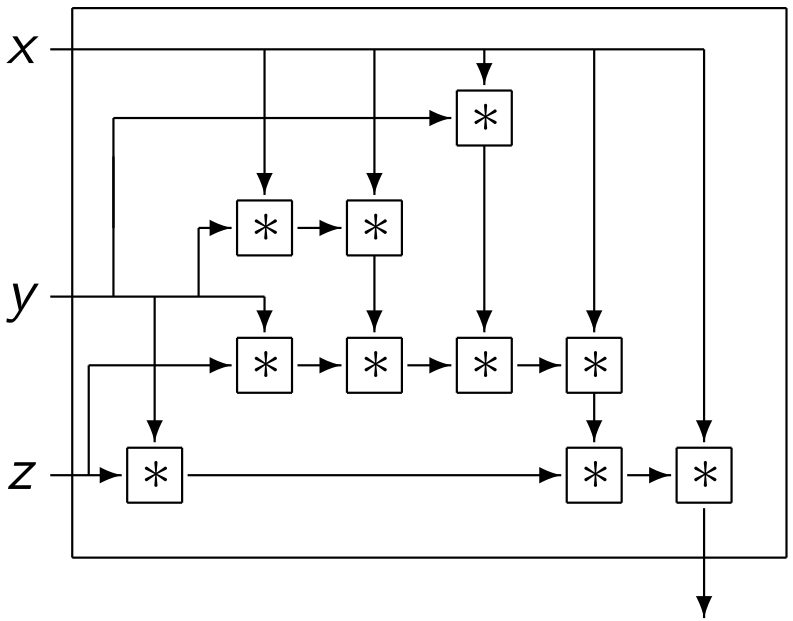


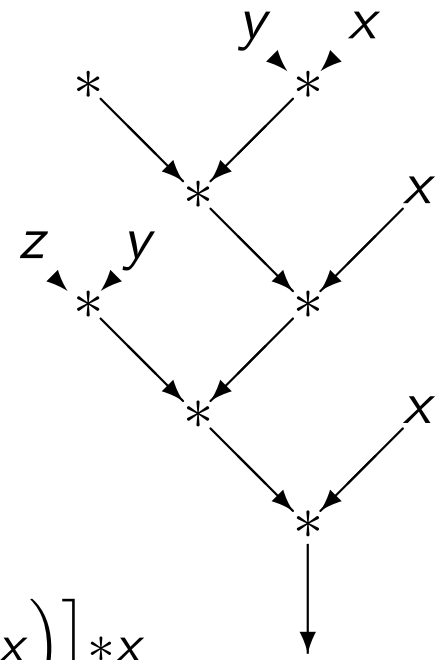
$$\Big[(z*y)*\Big(\Big(\big[(z*y)*((y*x)*x)\big]*(y*x)\Big)*x\Big)\Big]*x$$



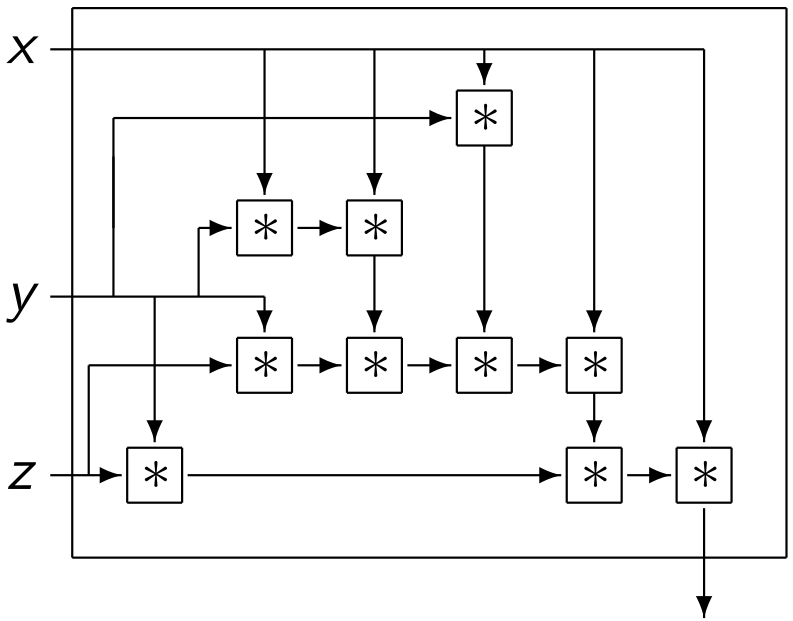


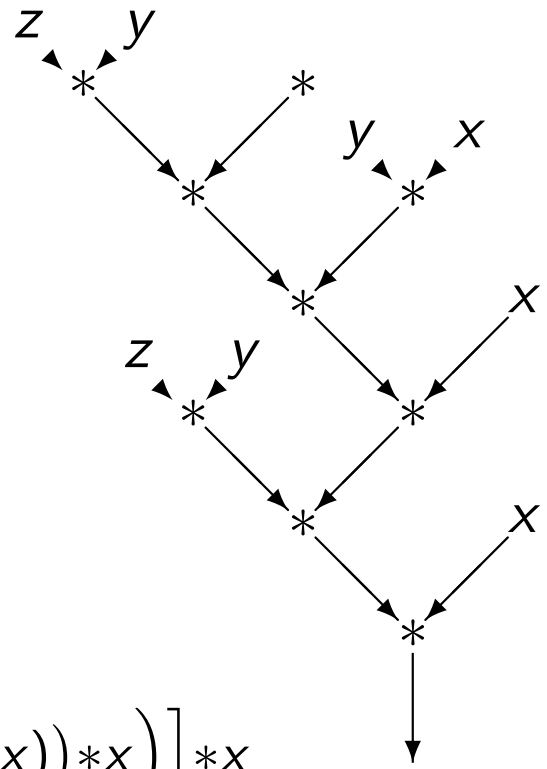
$$\Big[(z*y)*\Big(\Big(\big[(z*y)*((y*x)*x)\big]*(y*x)\Big)*x\Big)\Big]*x$$



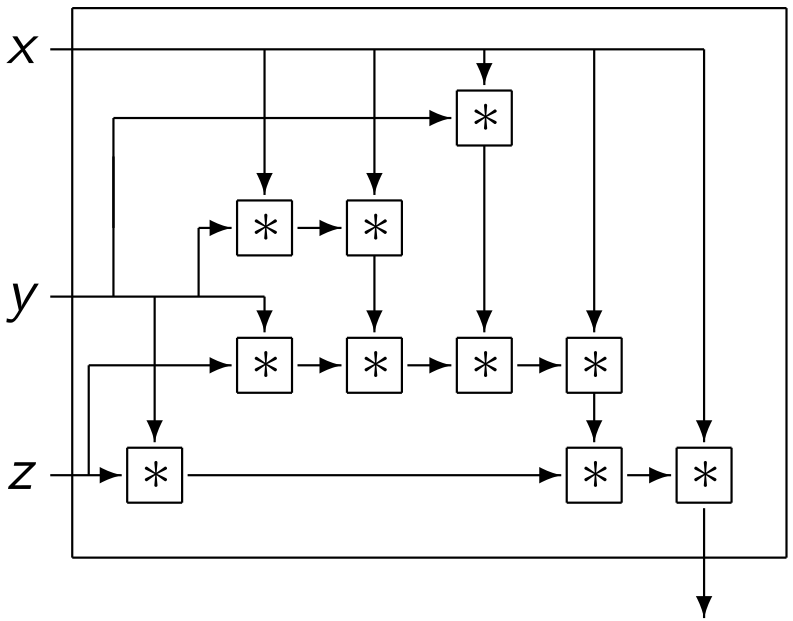


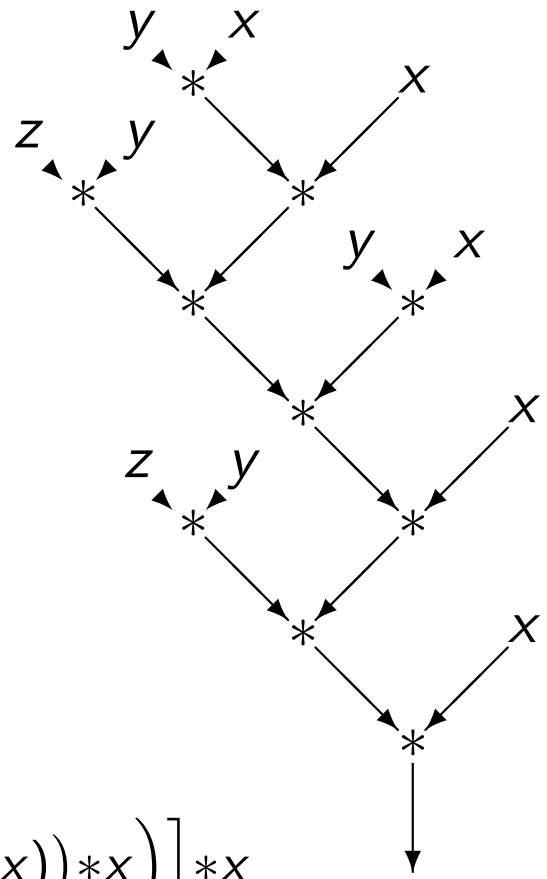
$$\Big[(z*y)*\Big(\Big(\big[(z*y)*((y*x)*x)\big]*(y*x)\Big)*x\Big)\Big]*x$$



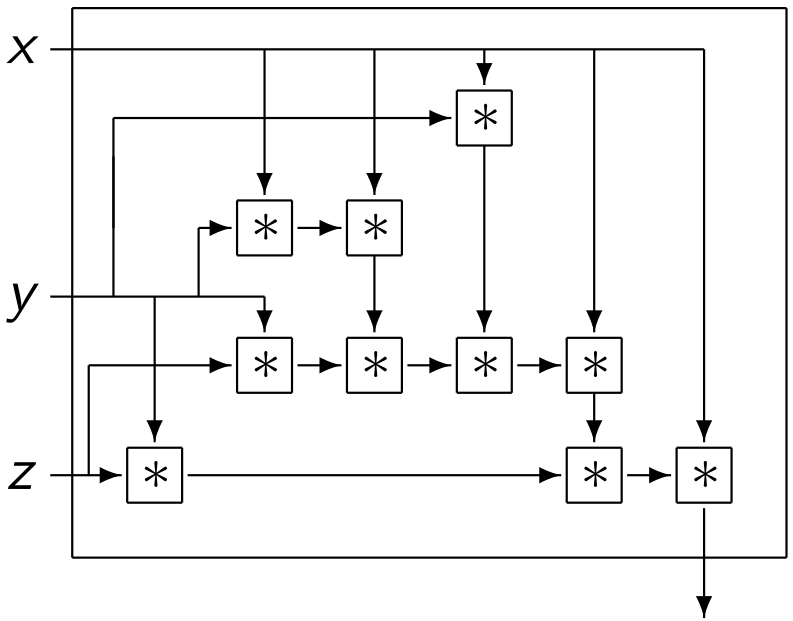


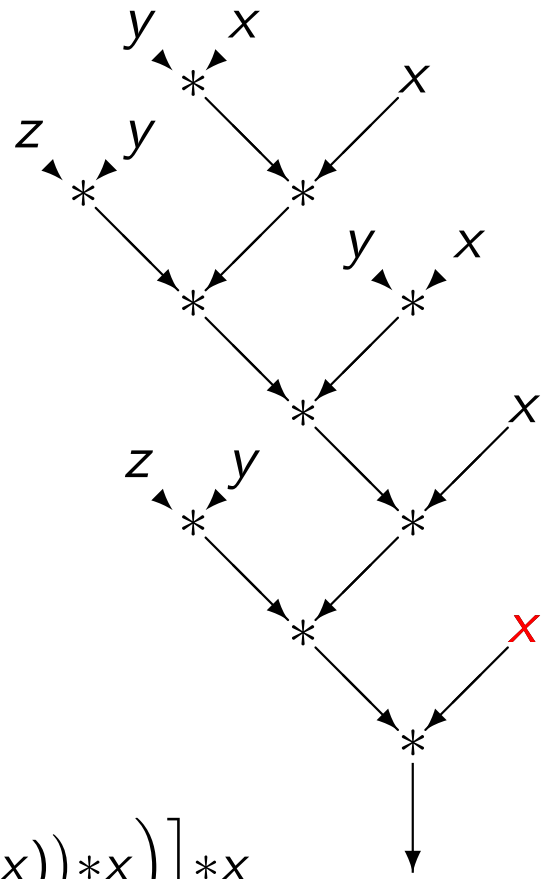
$$\Big[(z*y)*\Big(\Big(\big[(z*y)*((y*x)*x)\big]*(y*x)\Big)*x\Big)\Big]*x$$



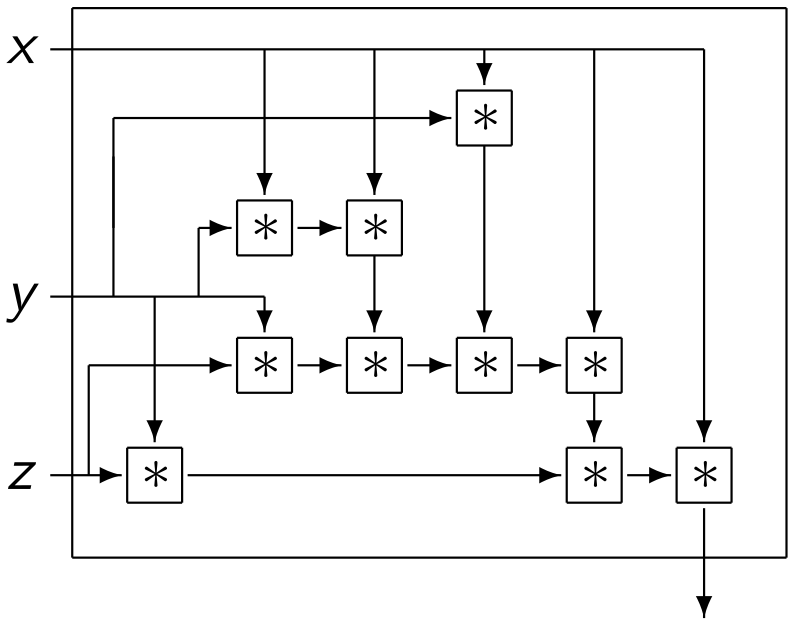


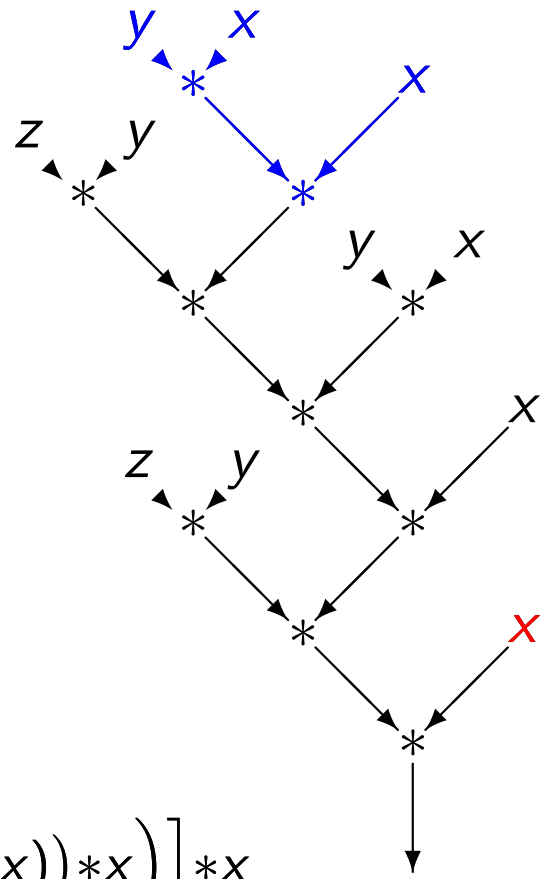
$$\Big[(z{*}y) * \Big(\big(\big[(z{*}y){*}((y{*}x){*}x)\big]{*}(y{*}x)\big){*}x\Big)\Big]{*}x$$



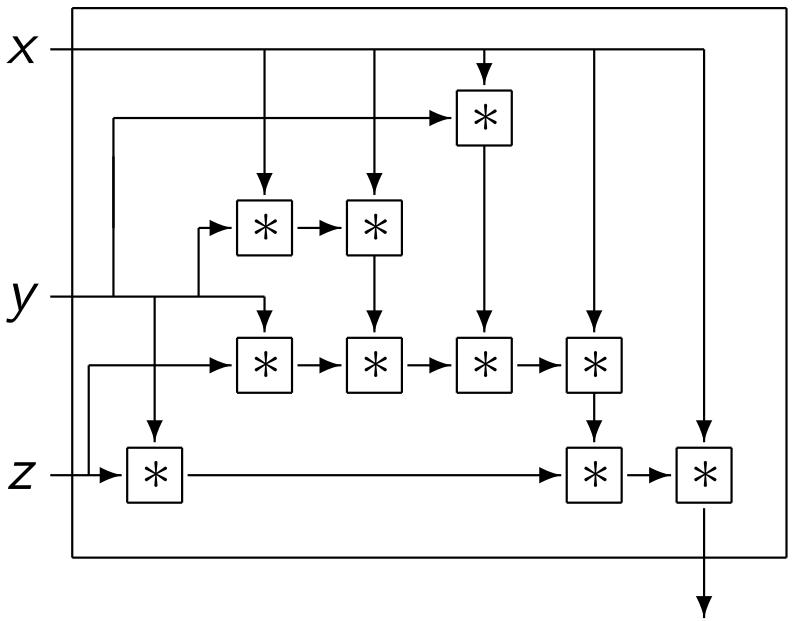


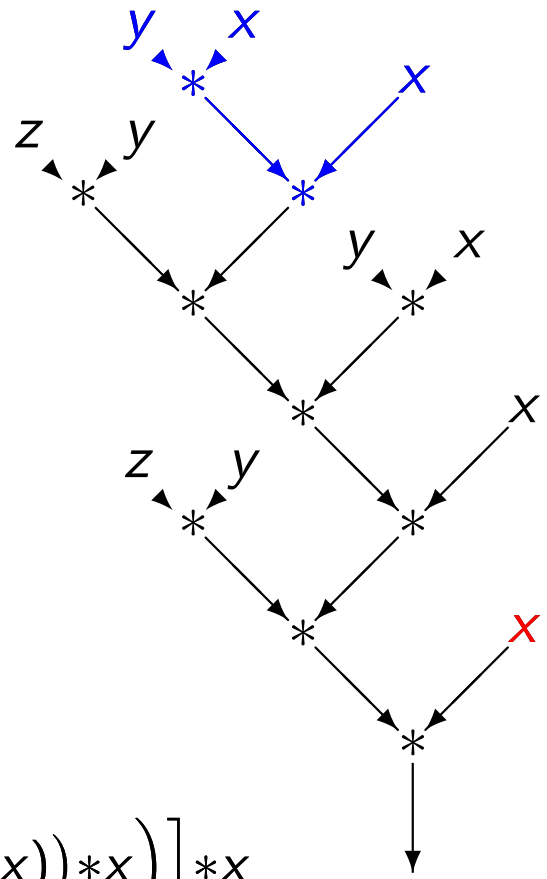
$$\Big[(z{*}y) * \Big(\big(\big[(z{*}y){*}((y{*}x){*}x)\big]{*}(y{*}x)\big){*}x\Big)\Big]{*}x$$





$$\Big[(z{*}y) * \Big(\big(\big[(z{*}y){*}\big((y{*}x){*}x\big)\big]{*}(y{*}x)\big){*}x\Big)\Big]{*}x$$





$$\Big[(z{*}y) * \Big(\big(\big[(z{*}y){*}\big((y{*}x){*}x\big)\big]{*}(y{*}x)\big){*}x\Big)\Big]{*}x$$

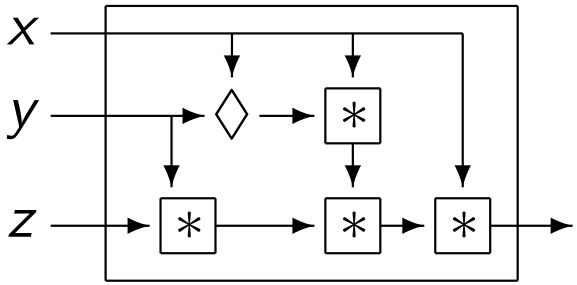
A **small** change in a circuit is a change high in its tree.

1. D. Clark, Evolution of algebraic terms 1: term to term operation continuity, *International Journal of Algebra and Computation*, Vol. 23, No. 5 (2013) 1175-1205.
2. D. Clark, M. Keijzer, L. Spector, Evolution of algebraic terms 2: the deep drilling algorithm, *International Journal of Algebra and Computation*, Vol. 26, No. 6 (2016) 1141- 1176.
3. D. Clark, L. Spector, Evolution of algebraic terms 3: beam algorithms and term continuity, *International Journal of Algebra and Computation*, Vol. 28, No. 05 (2018), 759-790.

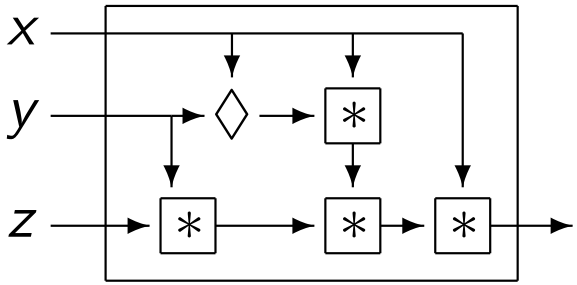
1. D. Clark, Evolution of algebraic terms 1: term to term operation continuity, *International Journal of Algebra and Computation*, Vol. 23, No. 5 (2013) 1175-1205.
2. D. Clark, M. Keijzer, L. Spector, Evolution of algebraic terms 2: the deep drilling algorithm, *International Journal of Algebra and Computation*, Vol. 26, No. 6 (2016) 1141- 1176.
3. D. Clark, L. Spector, Evolution of algebraic terms 3: beam algorithms and term continuity, *International Journal of Algebra and Computation*, Vol. 28, No. 05 (2018), 759-790.

Conjecture. *If a gate is term continuous, then the value computed at many nodes high enough in the circuit tree will have no effect on the output value.*

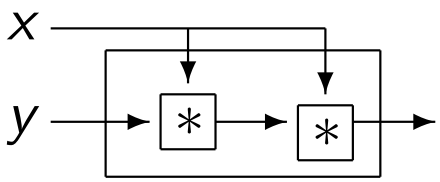
1. D. Clark, Evolution of algebraic terms 1: term to term operation continuity, *International Journal of Algebra and Computation*, Vol. 23, No. 5 (2013) 1175-1205.
2. D. Clark, M. Keijzer, L. Spector, Evolution of algebraic terms 2: the deep drilling algorithm, *International Journal of Algebra and Computation*, Vol. 26, No. 6 (2016) 1141- 1176.
3. D. Clark, L. Spector, Evolution of algebraic terms 3: beam algorithms and term continuity, *International Journal of Algebra and Computation*, Vol. 28, No. 05 (2018), 759-790.
4. D. Clark, N. Falco, Evolution of algebraic terms 4: a two-parent evolutionary algorithm, (under construction).



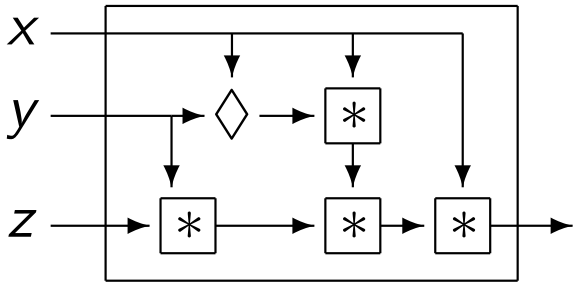
$$\big((z * y) * (\Diamond * x)\big) * x$$



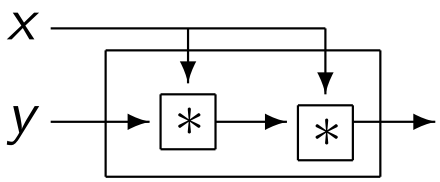
$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$



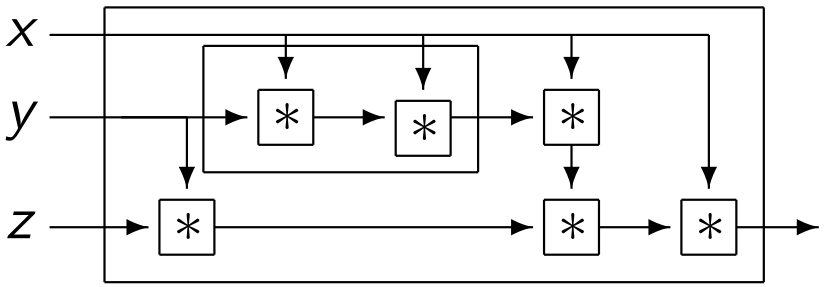
$$(y{*}x){*}x$$



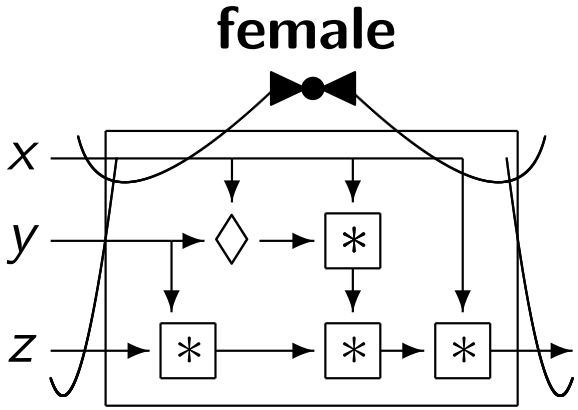
$$\big((z*y)*(\Diamond*x)\big)*x$$



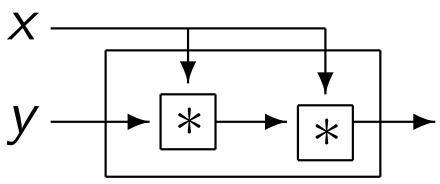
$$(y*x)*x$$



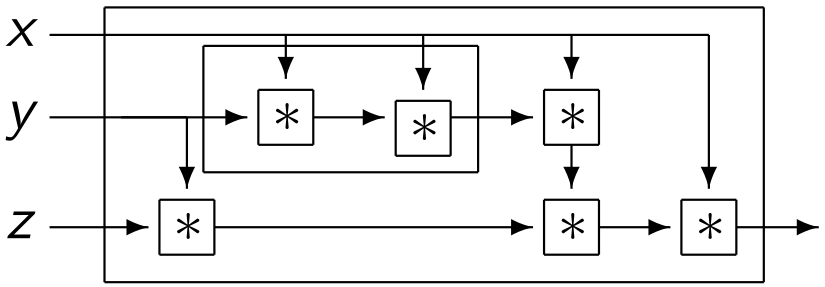
$$\big((z*y)*([(y*x)*x]*x)\big)*x$$



$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

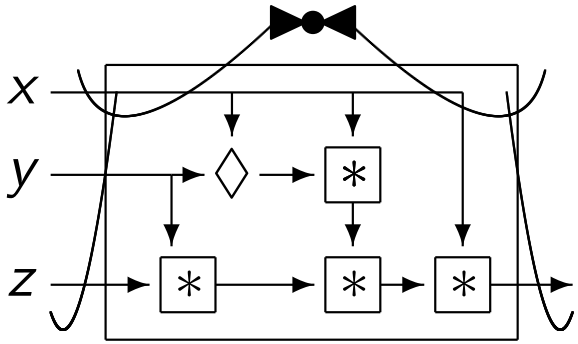


$$(y{*}x){*}x$$



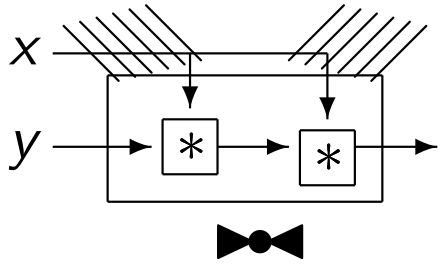
$$\big((z{*}y){*}([(y{*}x){*}x]{*}x)\big){*}x$$

female

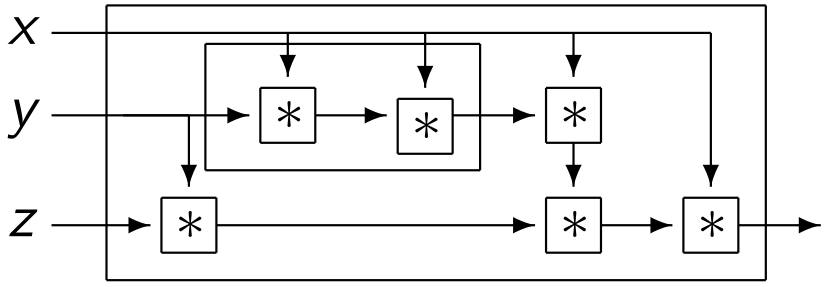


$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

male

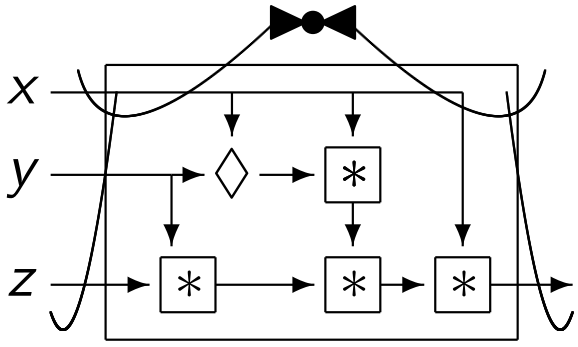


$$(y{*}x){*}x$$



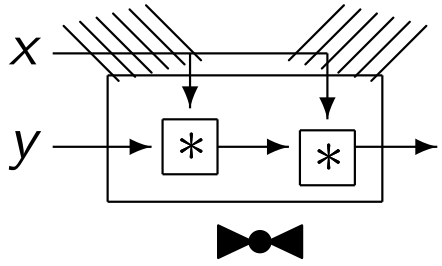
$$\big((z{*}y){*}([(y{*}x){*}x]{*}x)\big){*}x$$

female

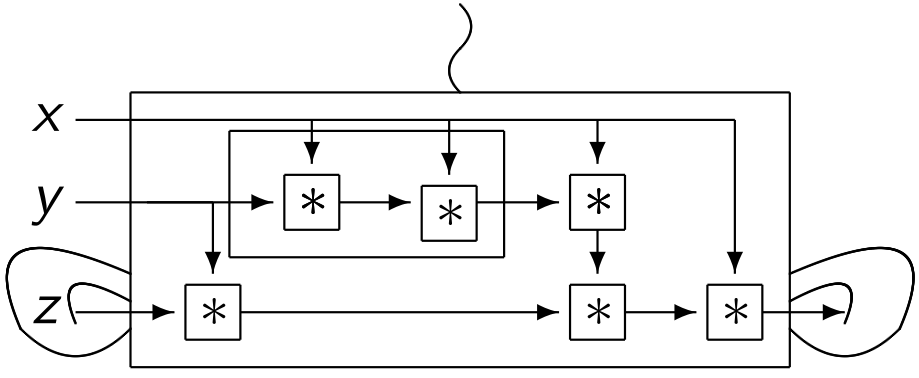


$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

male

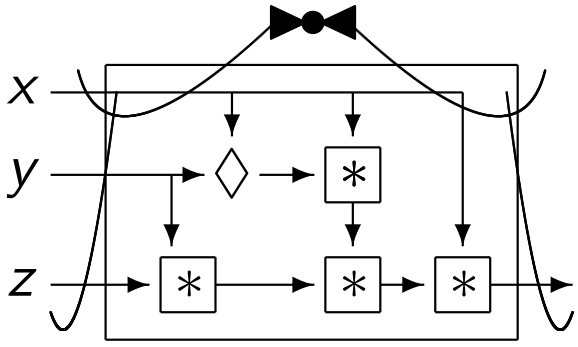


$$(y{*}x){*}x$$



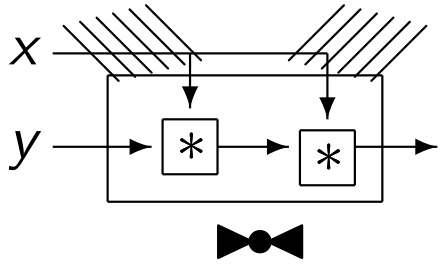
$$\big((z{*}y){*}([(y{*}x){*}x]{*}x)\big){*}x$$

female

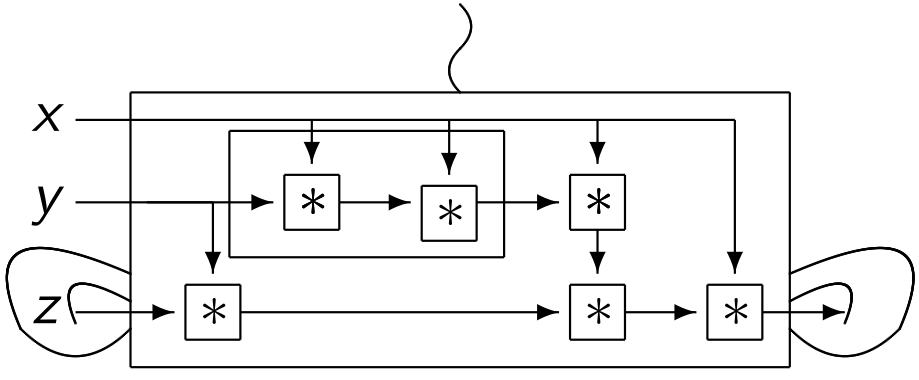


$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

male



$$(y{*}x){*}x$$



$$\big((z{*}y){*}([(y{*}x){*}x]{*}x)\big){*}x$$

son

female

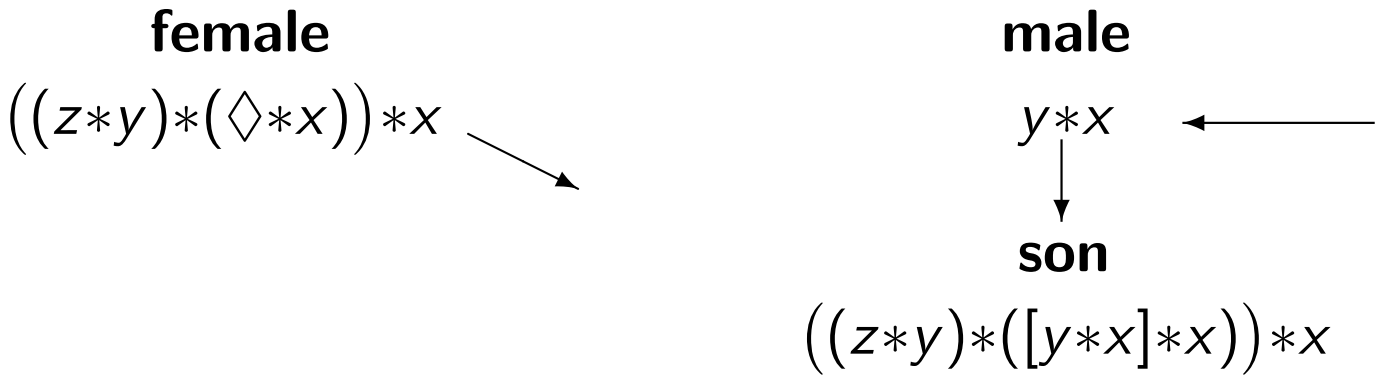
$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

female

$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

male

$$y{*}x \longleftarrow$$


female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$

female

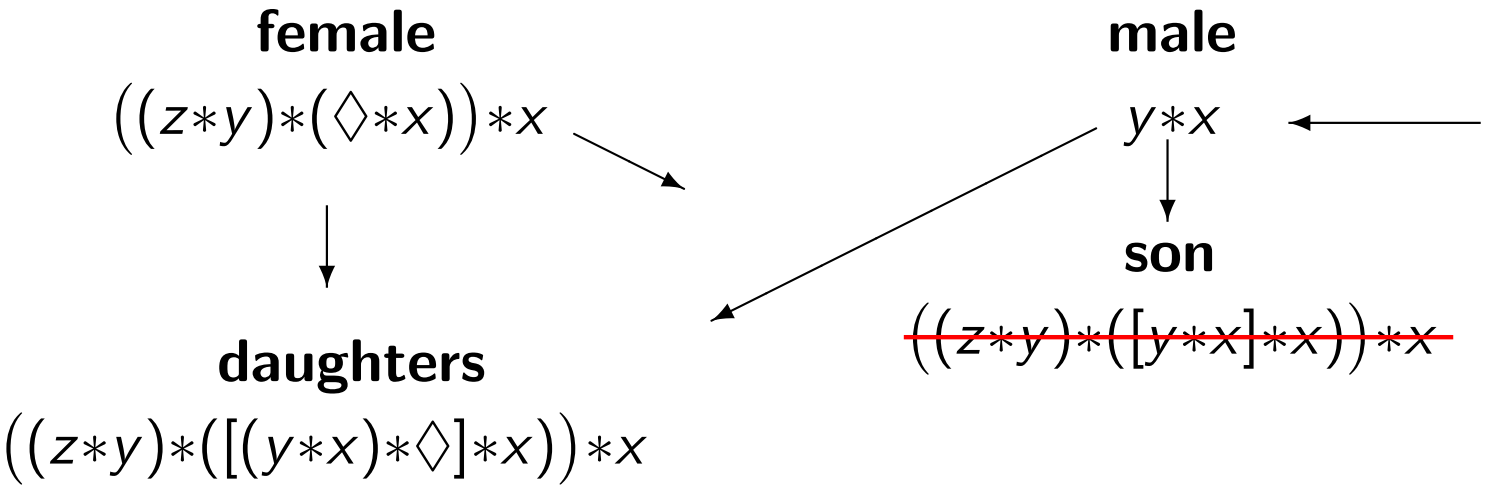
$$\big((z{*}y){*}(\Diamond{*}x)\big){*}x$$

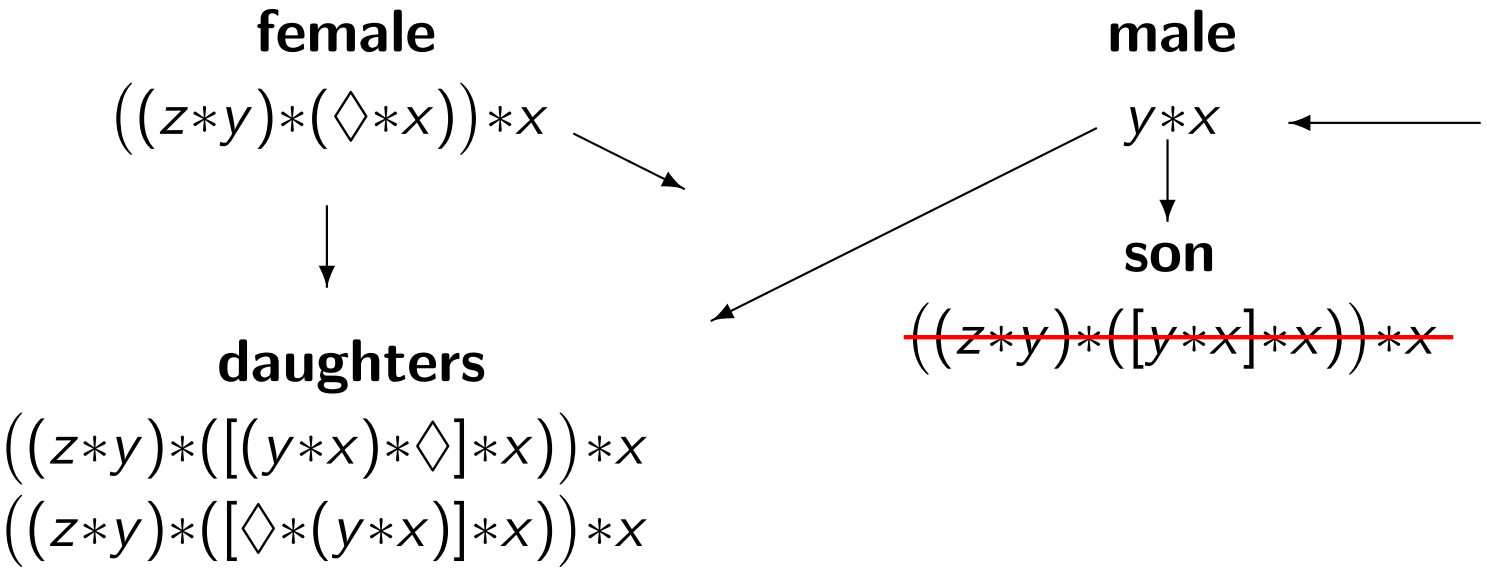
male

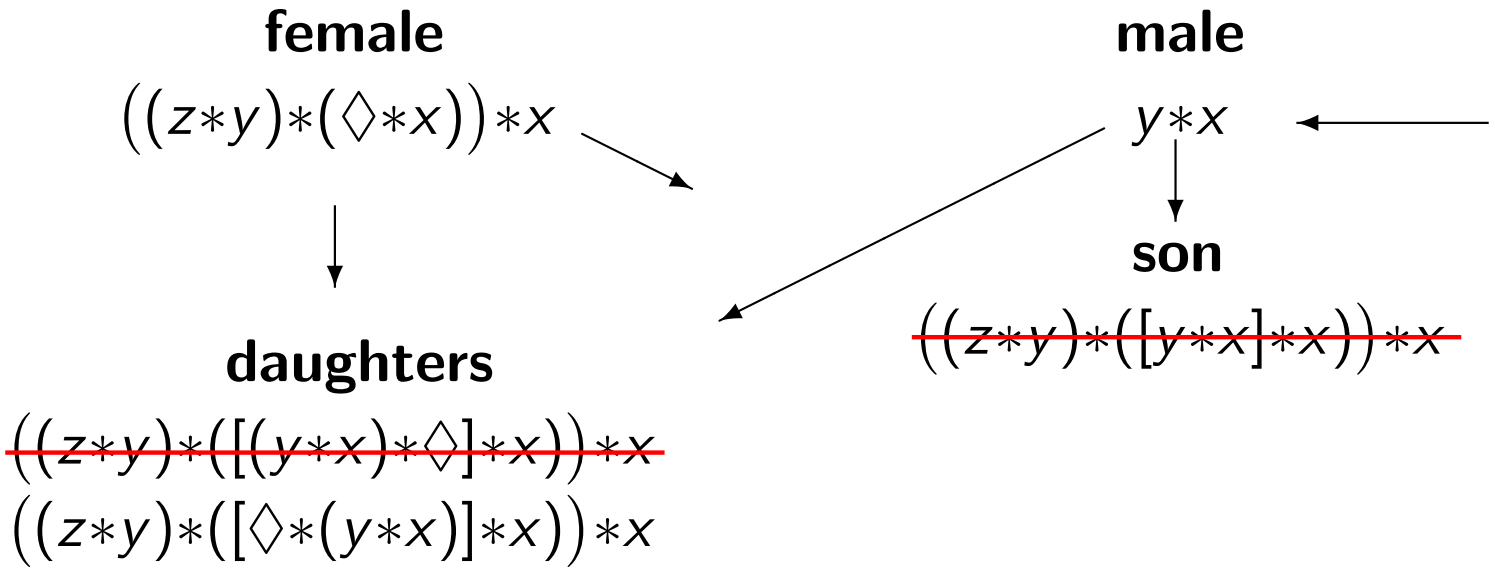
$$y{*}x$$

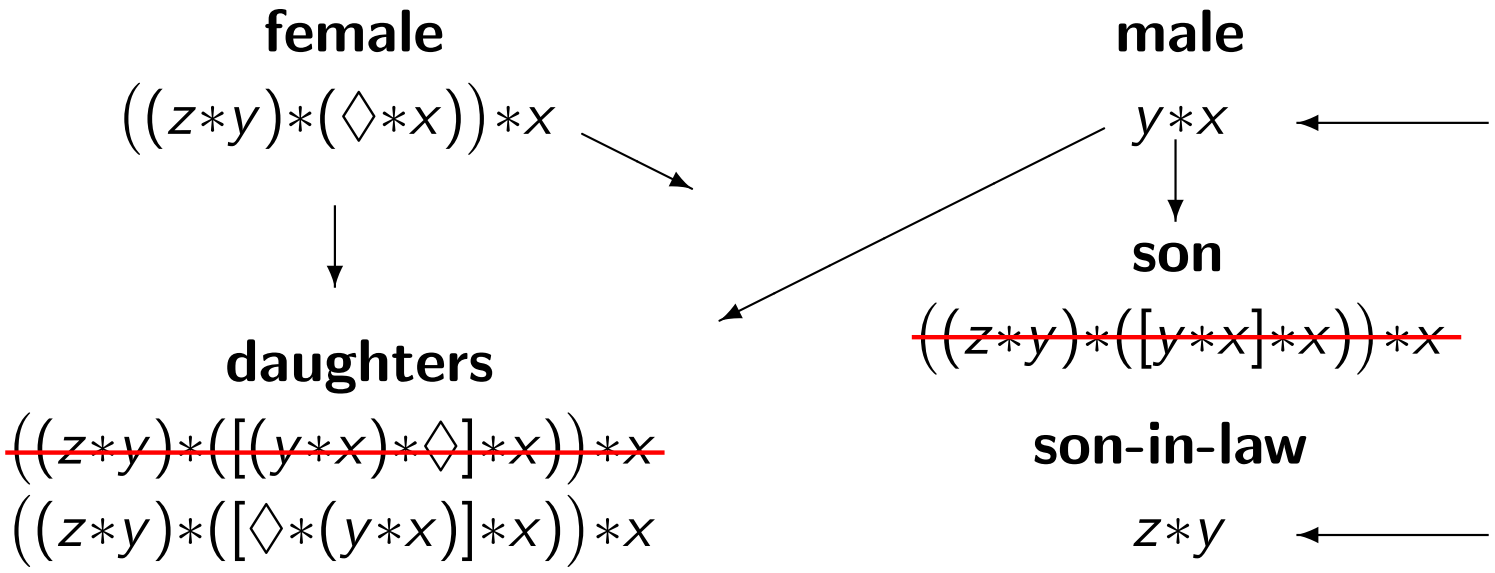
son

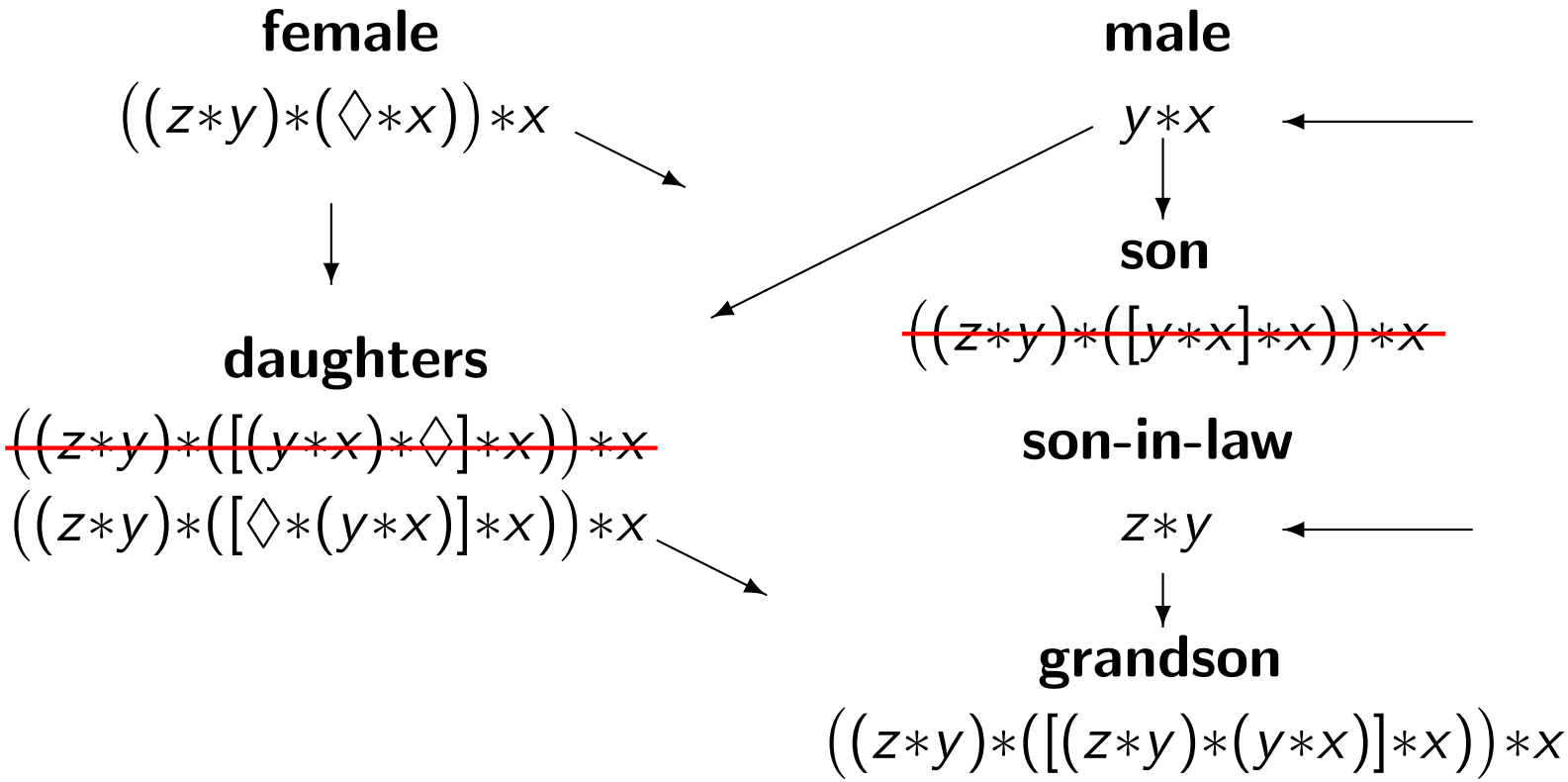
$$\cancel{\big((z{*}y){*}([y{*}x]{*}x)\big){*}x}$$

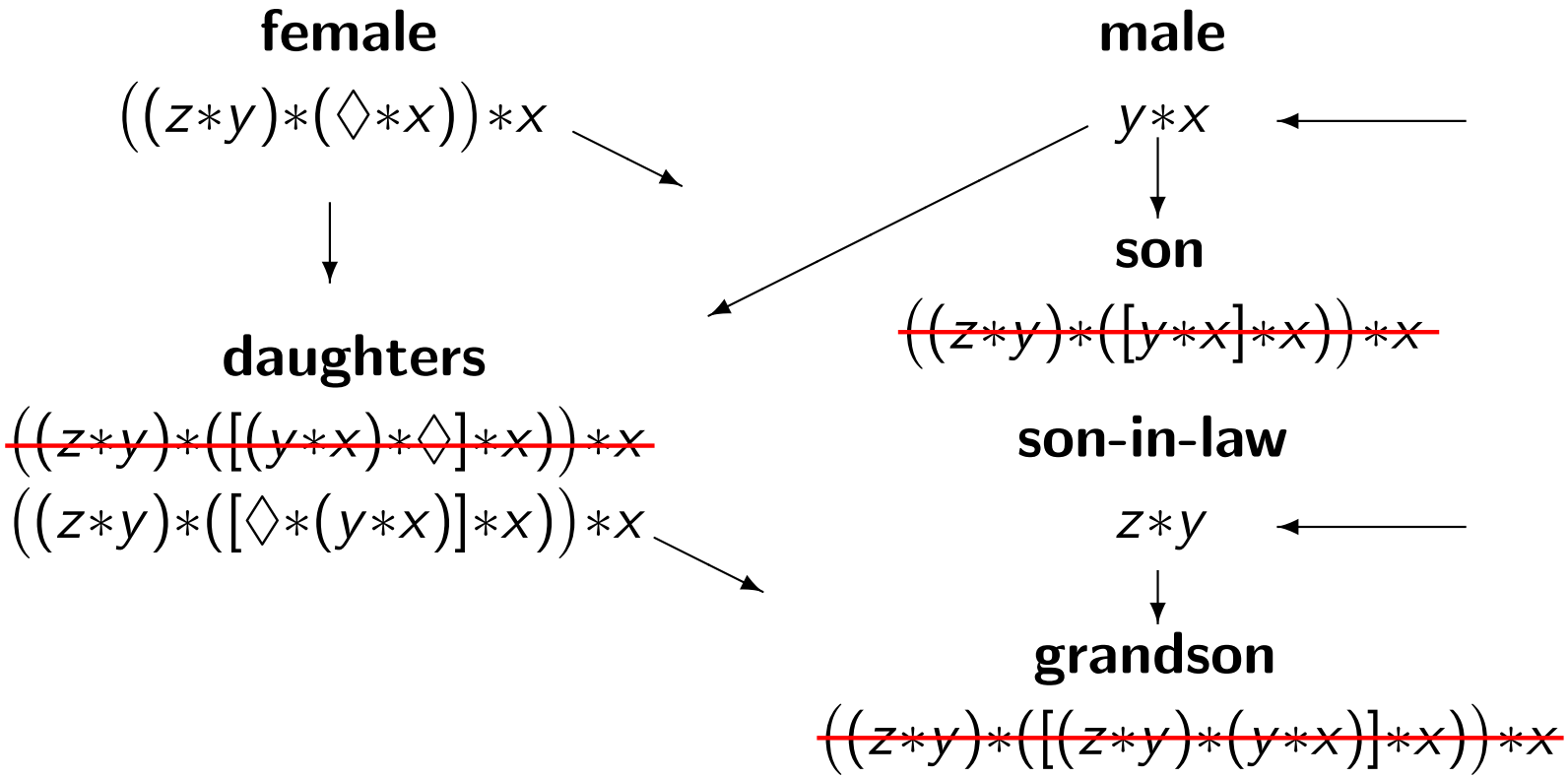

female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$

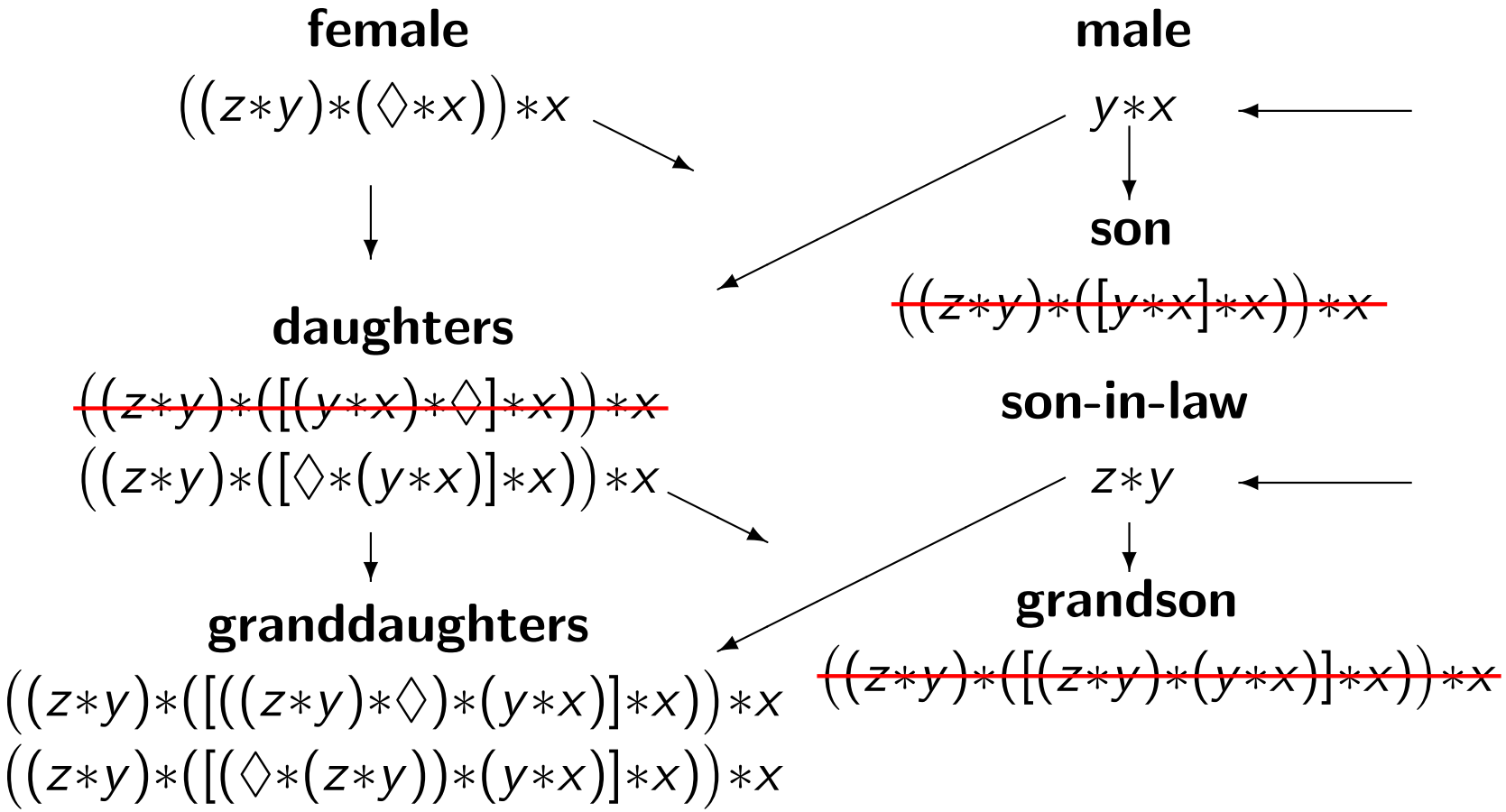

female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$

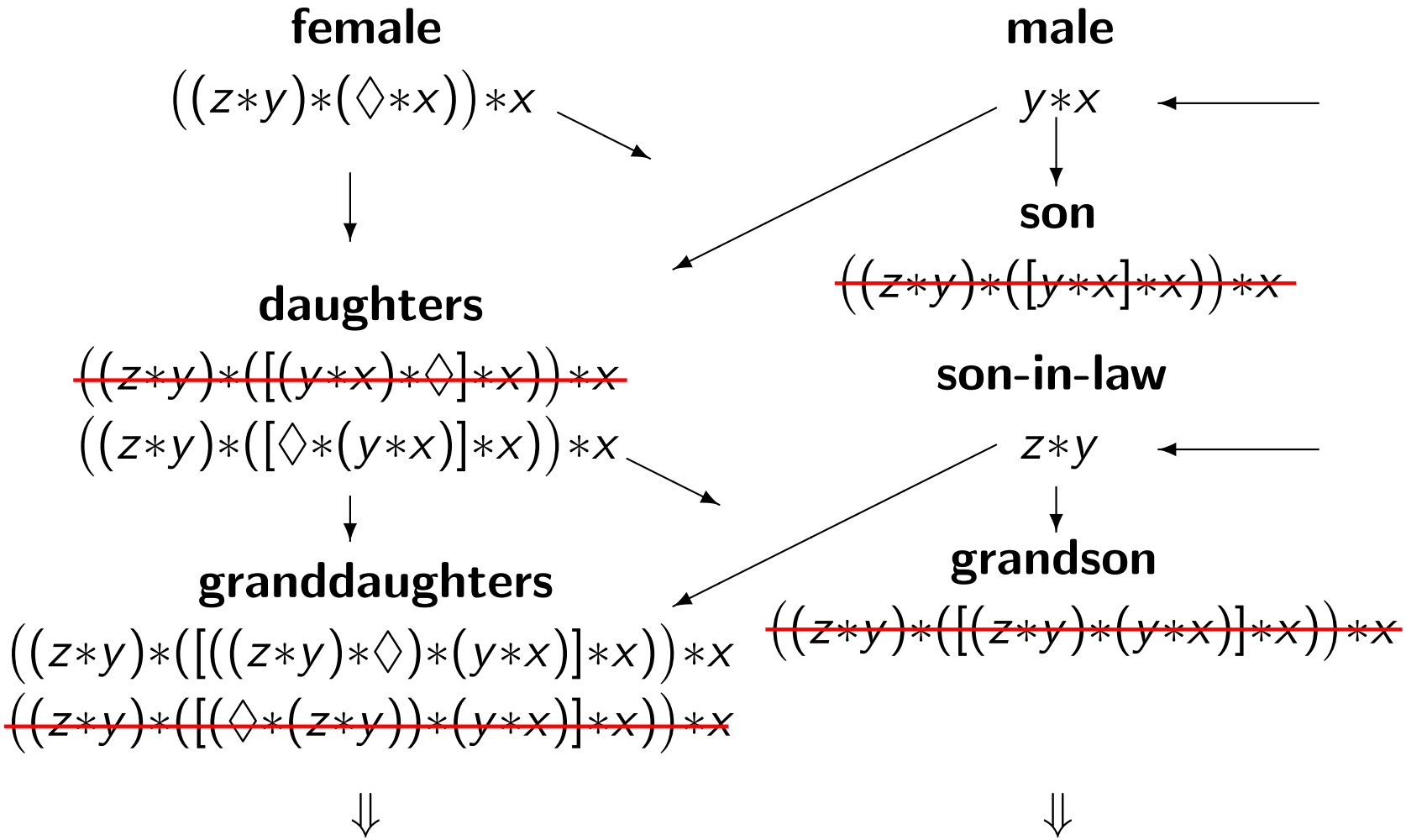

female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$

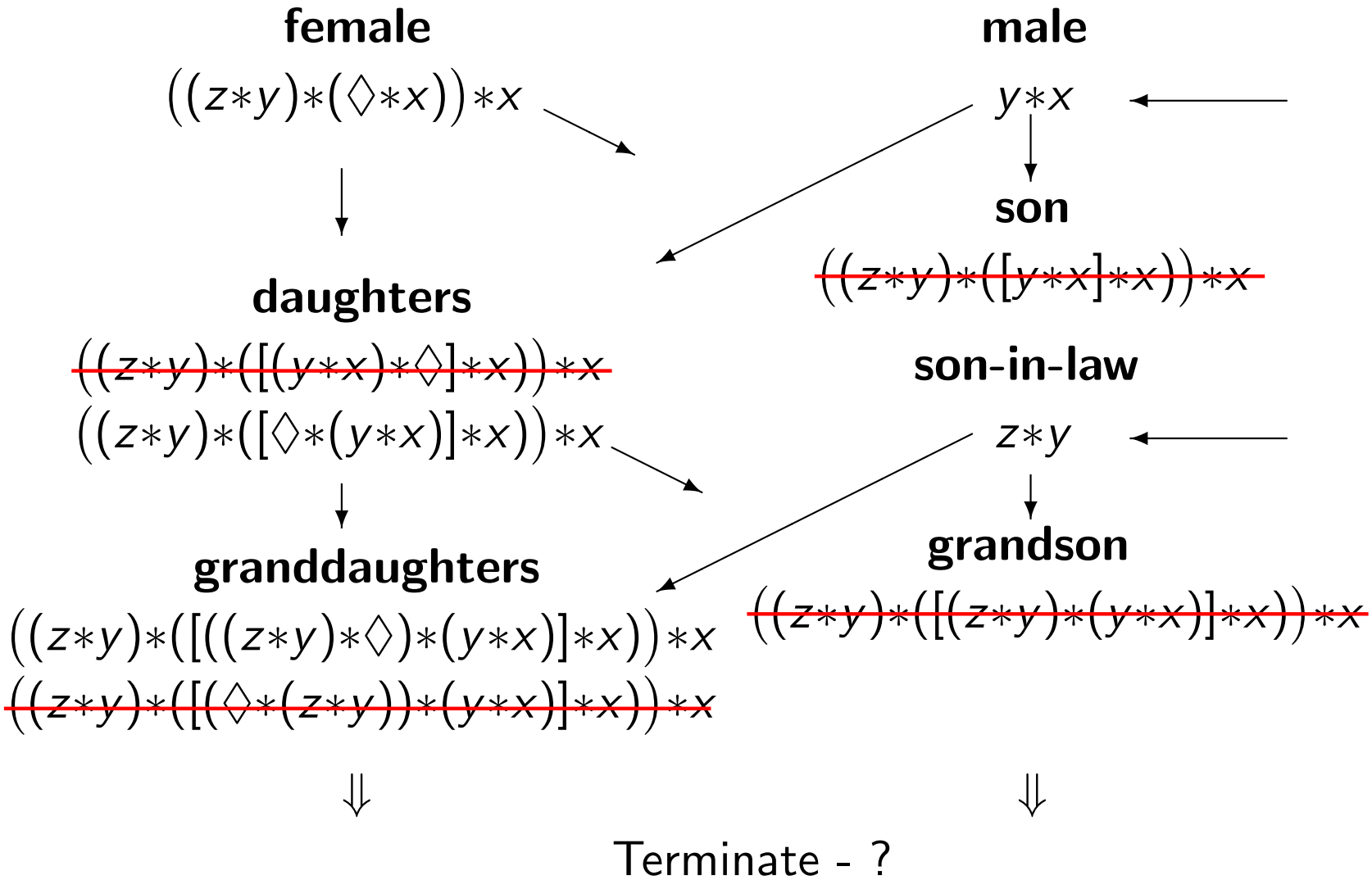

female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$
son-in-law
$z*y$


female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$
son-in-law
$z*y$
grandson
$((z*y)*([(z*y)*(y*x)]*x))*x$

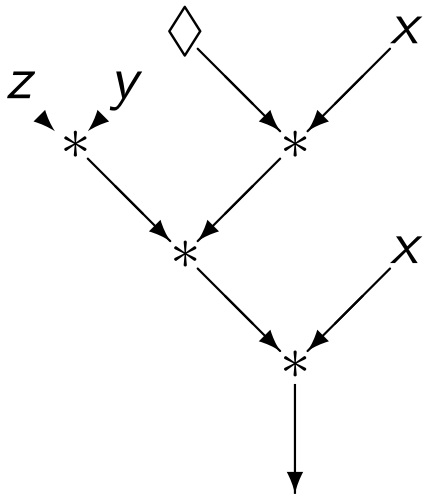

female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$
son-in-law
$z*y$
grandson
$((z*y)*([(z*y)*(y*x)]*x))*x$


female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$
son-in-law
$z*y$
granddaughters
$((z*y)*([((z*y)*\Diamond)*(y*x)]*x))*x$
$((z*y)*([(\Diamond*(z*y))*(y*x)]*x))*x$
grandson
$((z*y)*([(z*y)*(y*x)]*x))*x$

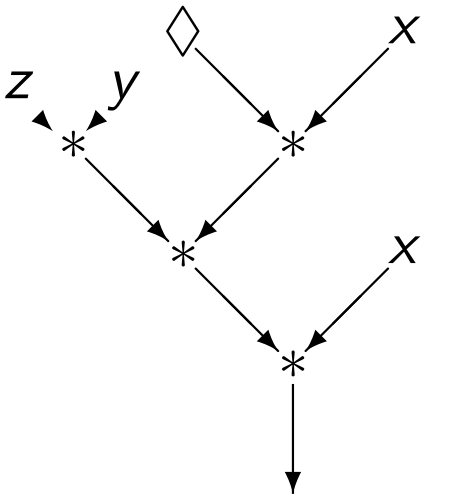

female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$
son-in-law
$z*y$
granddaughters
$((z*y)*([((z*y)*\Diamond)*(y*x)]*x))*x$
$((z*y)*([(\Diamond*(z*y))*(y*x)]*x))*x$
grandson
$((z*y)*([(z*y)*(y*x)]*x))*x$


female
$((z*y)*(\Diamond*x))*x$
male
$y*x$
son
$((z*y)*([y*x]*x))*x$
daughters
$((z*y)*([(y*x)*\Diamond]*x))*x$
$((z*y)*([\Diamond*(y*x)]*x))*x$
son-in-law
$z*y$
granddaughters
$((z*y)*([((z*y)*\Diamond)*(y*x)]*x))*x$
$((z*y)*([(\Diamond*(z*y))*(y*x)]*x))*x$
grandson
$((z*y)*([(z*y)*(y*x)]*x))*x$
Terminate - ?

$$\Big(\big(z{*}y\big){*}\big(\Diamond{*}x\big)\Big){*}x$$

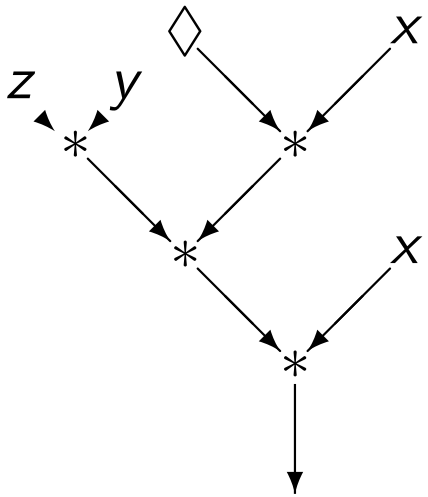


$$\Big(\big(z{*}y\big){*}\big(\Diamond{*}x\big)\Big){*}x$$



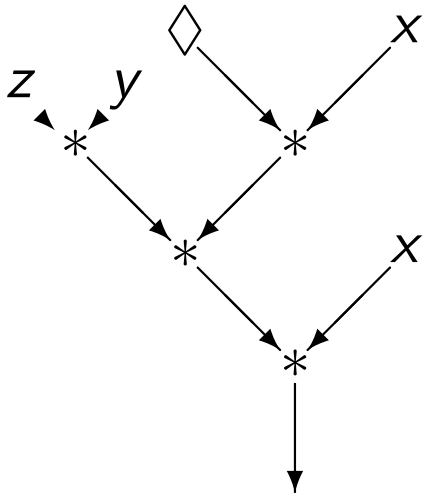
$$\Big(\big(z*y\big)*\big(\Diamond*x\big)\Big)*x$$

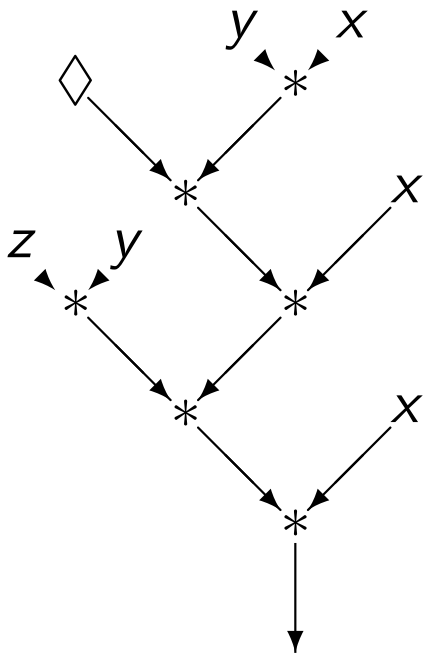
$$y*x$$



$$\Big((z*y)*(\Diamond*x)\Big)*x \qquad \Big((z*y)*\Big((\Diamond*(y*x))*x\Big)\Big)*x$$

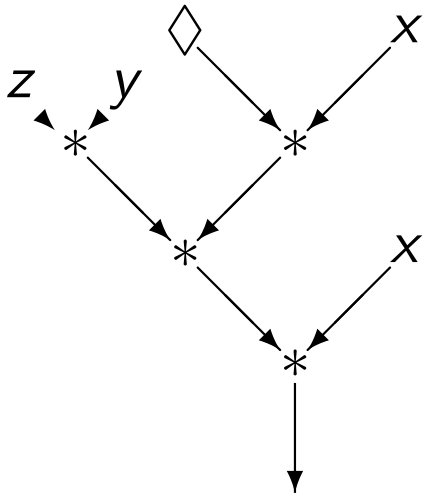
$y * x$





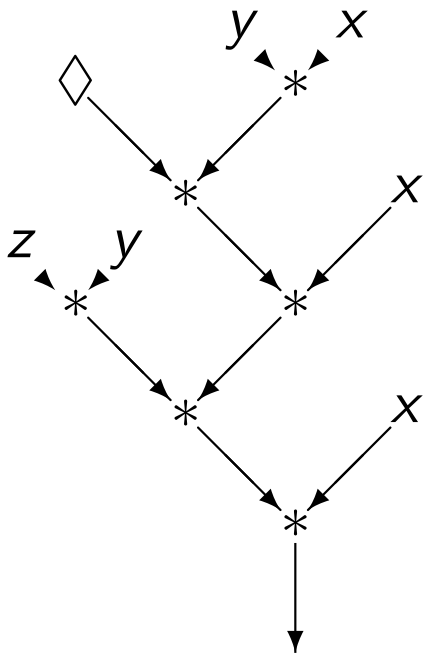
$$\Big((z*y)*(\Diamond*x)\Big)*x \qquad \Big((z*y)*\Big((\Diamond*(y*x))*x\Big)\Big)*x$$

$$y*x$$



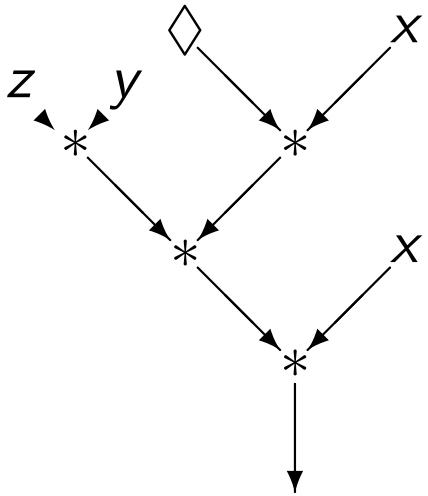
$$\Big((z*y)*(\Diamond*x)\Big)*x$$

$y * x$



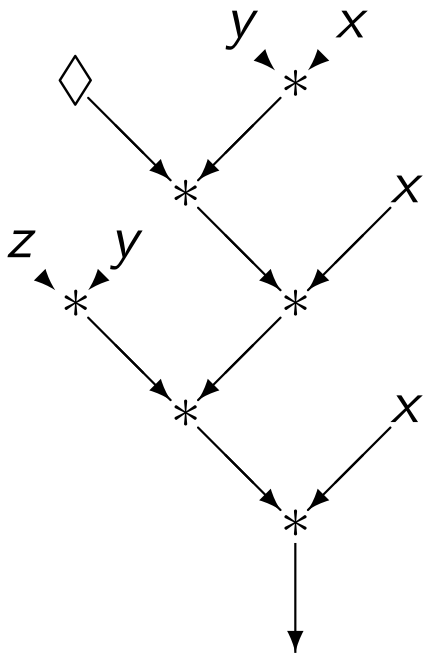
$$\Big((z*y)*\Big((\Diamond*(y*x))*x\Big)\Big)*x$$

$z * y$



$$\Big((z*y)*(\Diamond*x)\Big)*x$$

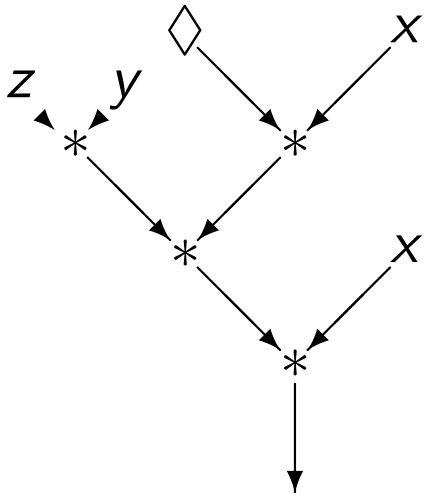
$$y * x$$



$$\Big((z*y)*\Big((\Diamond*(y*x))*x\Big)\Big)*x$$

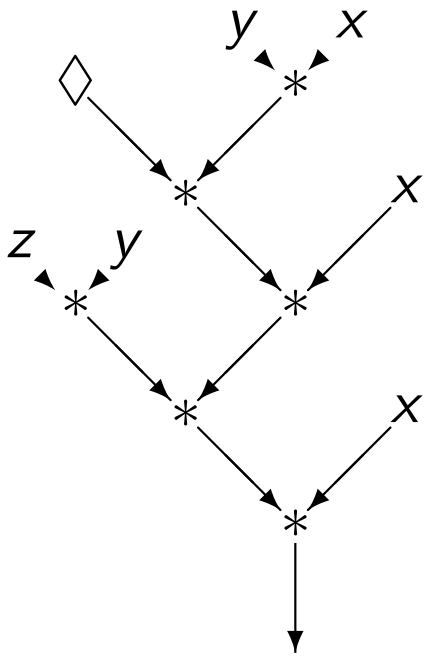
$$z * y$$

$$\Big((z*y)*\Big(\big(((z*y)*\Diamond)*(y*x)\big)*x\Big)\Big)*x$$



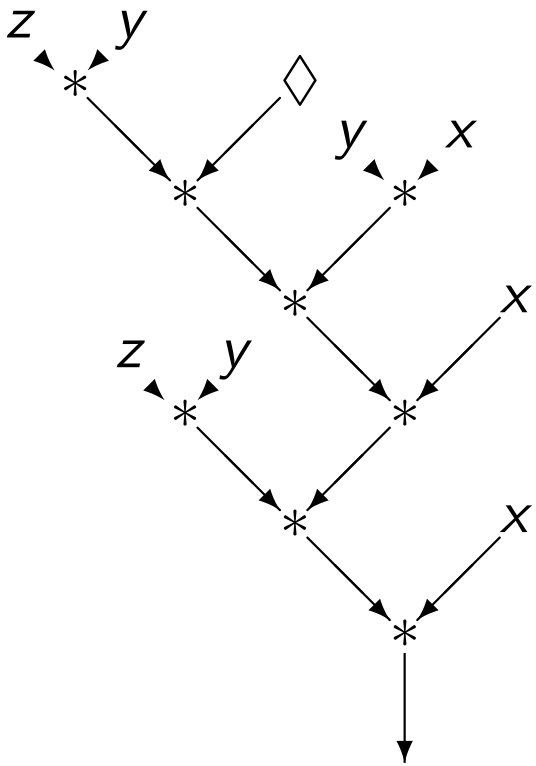
$$\Big((z{*}y){*}(\Diamond{*}x)\Big){*}x$$

$$y * x$$

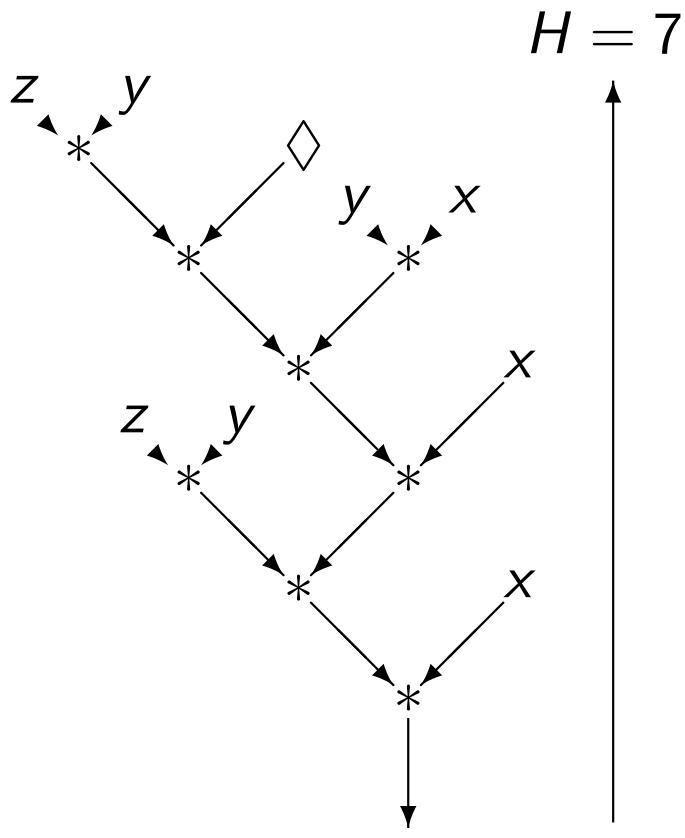


$$\Big((z{*}y){*}\Big((\Diamond{*}(y{*}x)){*}x\Big)\Big){*}x$$

$$z * y$$

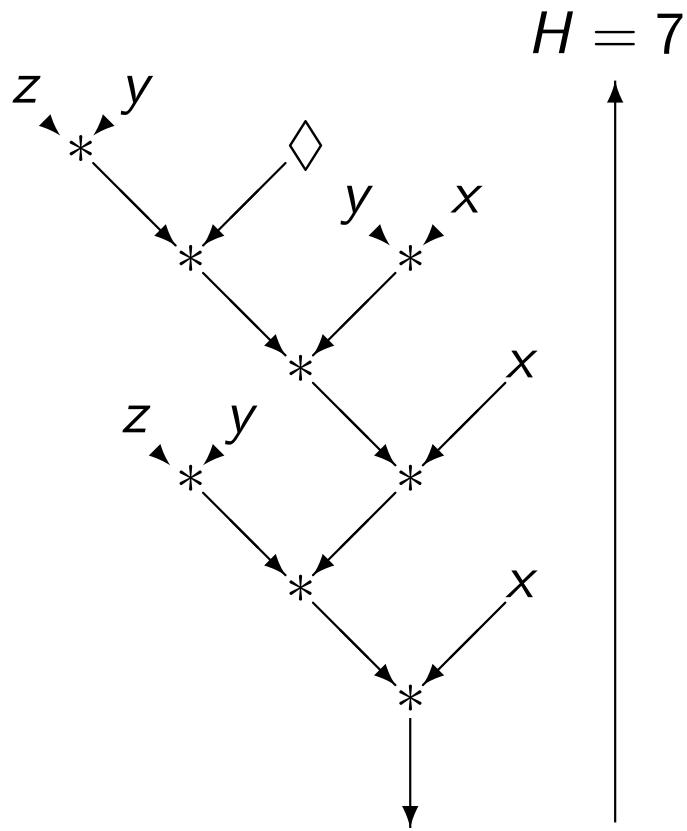


$$\Big((z{*}y){*}\Big(\big(((z{*}y){*}\Diamond){*}(y{*}x)\big){*}x\Big)\Big){*}x$$



$$\Big((z*y)*\Big(\big(\big((z*y)*\Diamond\big)*(y*x)\big)*x\Big)\Big)*x$$

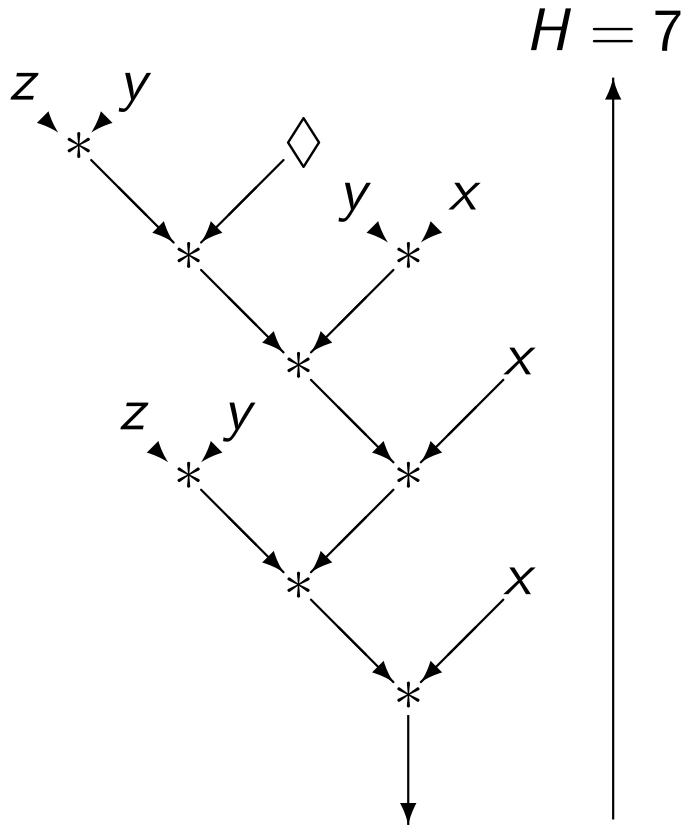
Theorem. *If* **G** *is a term continuous gate, then there is a number* λ *such that* $\frac{2}{3} \leq \lambda \leq 1$ *and*



$$\Big((z{*}y){*}\Big(\big(\big((z{*}y){*}\Diamond\big){*}(y{*}x)\big){*}x\Big)\Big){*}x$$

Theorem. *If* **G** *is a term continuous gate, then there is a number* λ *such that* $\frac{2}{3} \leq \lambda \leq 1$ *and*

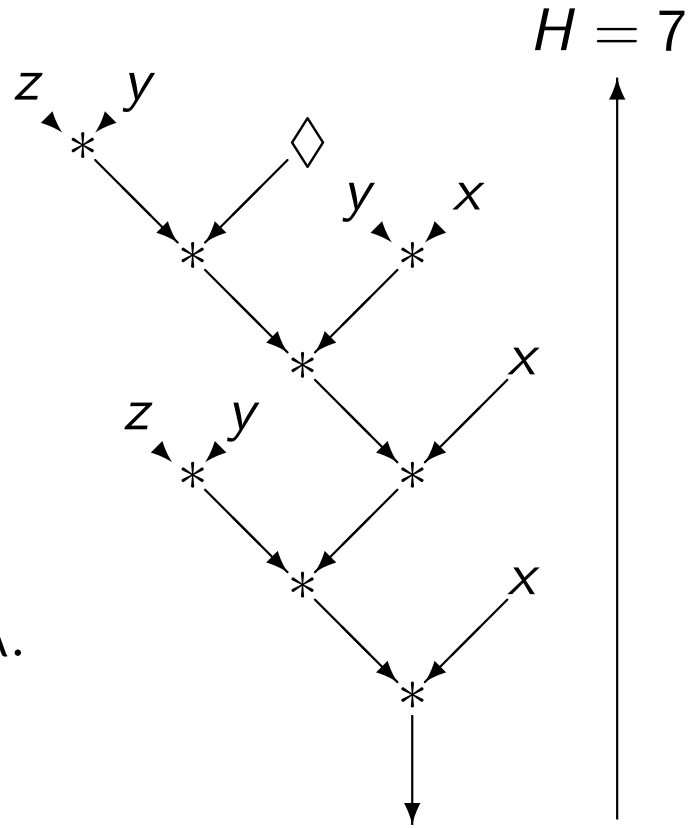
$$\lim_{H \to \infty} \frac{\#\Diamond\text{−dead circuits of height } H}{\#\Diamond\text{−circuits of height } H}$$



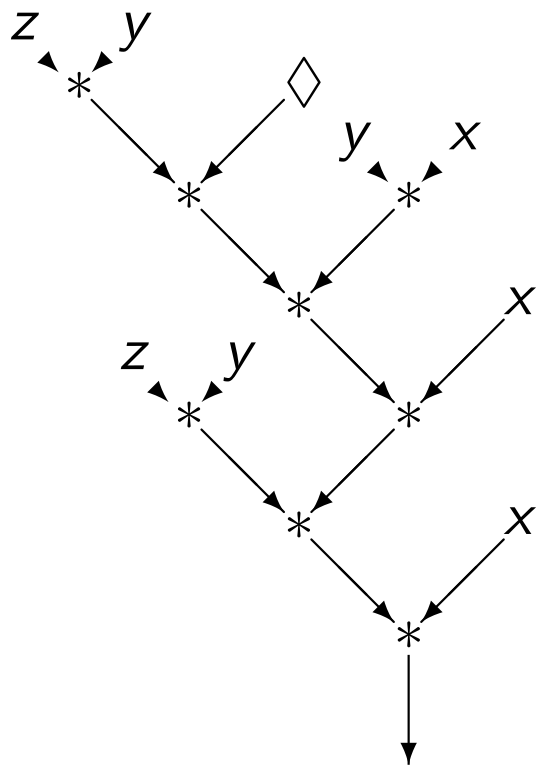
$$\Big((z{*}y){*}\Big(\big(\big((z{*}y){*}\Diamond\big){*}(y{*}x)\big){*}x\Big)\Big){*}x$$

Theorem. *If* **G** *is a term continuous gate, then there is a number* λ *such that* $\frac{2}{3} \leq \lambda \leq 1$ *and*

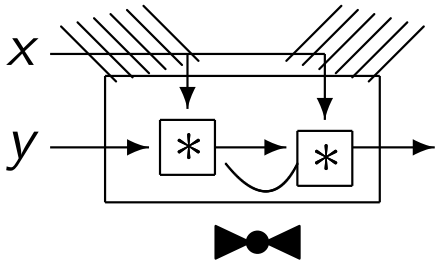
$$\lim_{H \to \infty} \frac{\#\Diamond\text{−dead circuits of height } H}{\#\Diamond\text{−circuits of height } H} = \lambda.$$



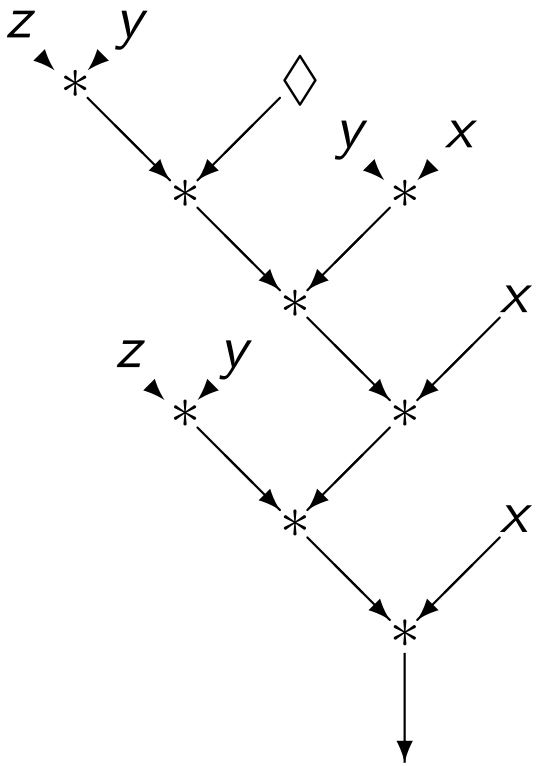
$$\Big((z{*}y){*}\Big(\big(\big((z{*}y){*}\Diamond\big){*}(y{*}x)\big){*}x\Big)\Big){*}x$$



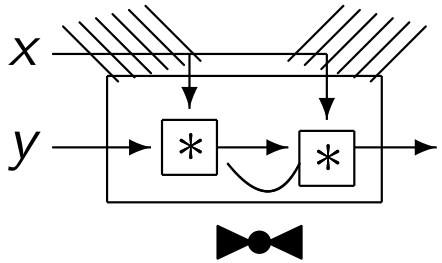
$$\Big((z*y)*\Big(\big(\big((z*y)*\Diamond\big)*(y*x)\big)*x\Big)\Big)*x$$



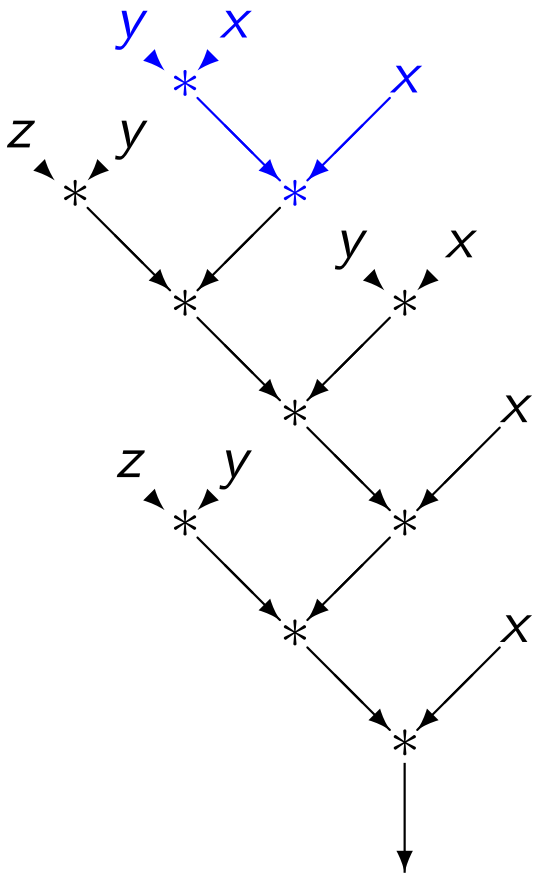
$$(y*x)*x$$



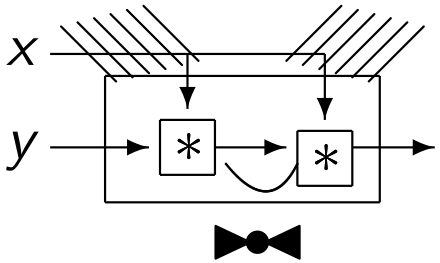
$$\Big((z*y)*\Big(\big(\big((z*y)*\Diamond\big)*(y*x)\big)*x\Big)\Big)*x$$



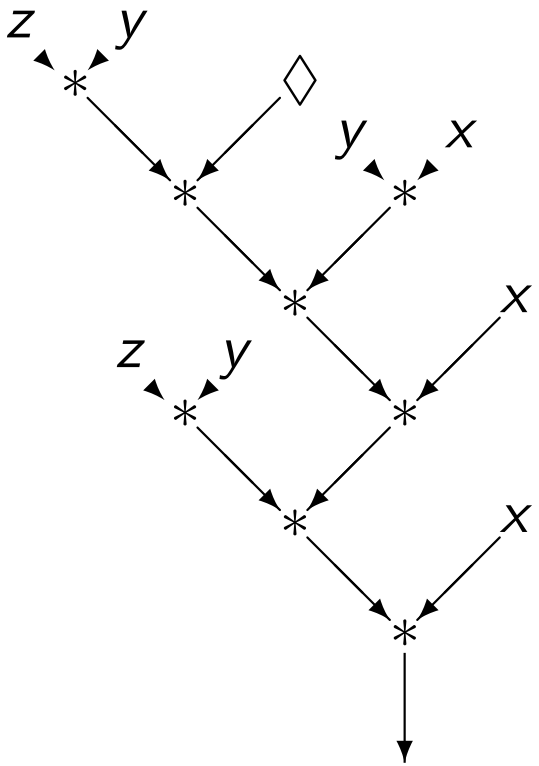
$$(y*x)*x$$



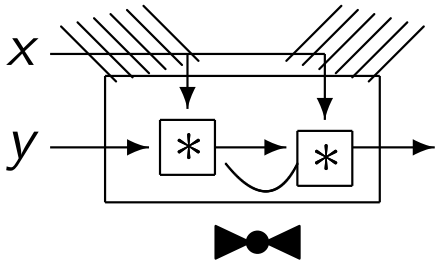
$$\Big((z*y)*\Big(\big(\big(((z*y)*((y*x)*x))*(y*x)\big)*x\big)\Big)*x$$



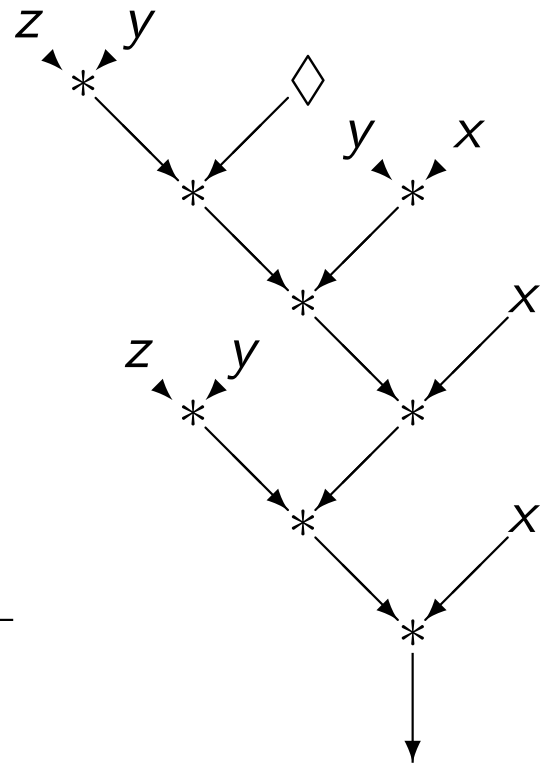
$$(y*x)*x$$

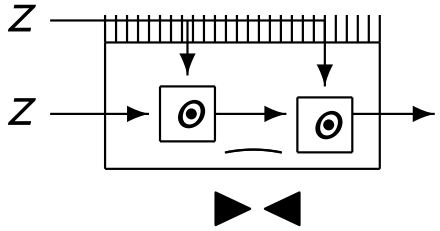


$$\Big((z*y)*\Big(\big(\big((z*y)*\Diamond\big)\big)*(y*x)\big)*x\Big)\Big)*x$$



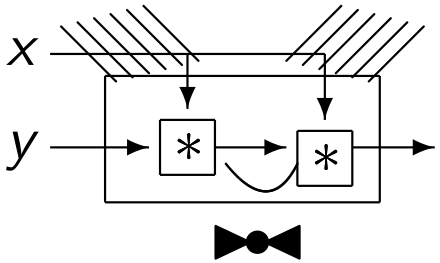
$(y*x)*x$



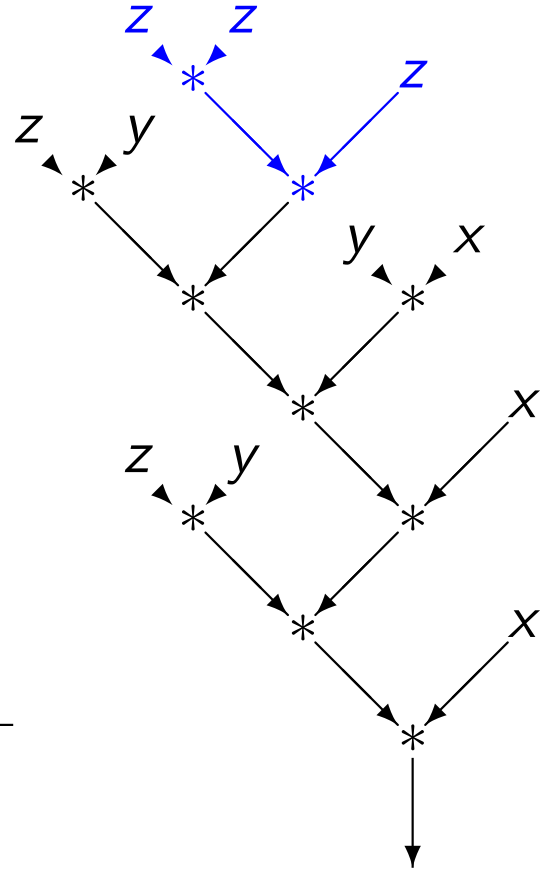


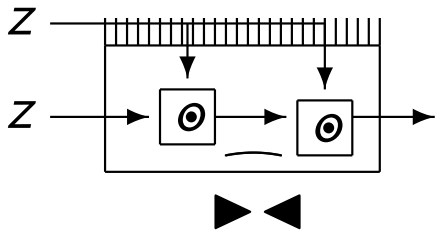
$(z*z)*z$

$$\Big((z*y)*\Big(\big(\big((z*y)*\Diamond\big)\big)*(y*x)\big)*x\Big)\Big)*x$$



$(y*x)*x$





$(z*z)*z$

$$\Big((z*y)*\Big(\big(\big((z*y)*\big((z*z)*z\big)\big)*(y*x)\big)*x\Big)\Big)*x$$

3-Element Gate

Possible target sequences: $3^{27} \approx 10^{13}$.

Typical solution to design problem, generated by **random search:**

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

3-Element Gate

Possible target sequences: $3^{27} \approx 10^{13}$.

Typical solution to design problem, generated by **random search: 4 months**

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

3-Element Gate

Possible target sequences: $3^{27} \approx 10^{13}$.

Typical solution to design problem, generated

- by **random search: 4 months**
- by **1-parent evolution: 5 minutes**

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

3-Element Gate

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Possible target sequences: $3^{27} \approx 10^{13}$.

Typical solution to design problem, generated

- by **random search: 4 months**
- by **1-parent evolution: 5 minutes**
- by **2-parent evolution:**

3-Element Gate

$*$	0	1	2
0	2	1	2
1	1	0	0
2	0	0	1

Possible target sequences: $3^{27} \approx 10^{13}$.

Typical solution to design problem, generated

- by **random search: 4 months**
- by **1-parent evolution: 5 minutes**
- by **2-parent evolution: 0.37 seconds**

$$(((((((((((((((((((((((((z*x)*((((((((z*(x*x))*y)*(x*(z*z)))*(((((y*x)*z)*y)*x))*(z*y))*(((z*y)*z)*y))*((z*x)*y))*(y*z))*(z*x)))*(z*x))*(y*(x*(x*y))))*((y*x)*((z*y)*y)))*(((z*z)*(z*z))*x))*(y*(x*x)))*(y*y))*(z*y))*(x*(((y*z)*y)*(x*z))))*((z*(z*x))*x))*((z*z)*x))*((z*y)*z))*((z*z)*y))*((((y*(z*(((y*((y*(y*z))*z))*x)*z)))*x)*(y*z))*x))*((x*(x*(((y*(z*y))*z)*((z*((y*(x*z))*z))*y))))*x))*((z*((((x*y)*x)*z)*(y*z)))*y))*((z*(y*z))*x))*(((y*z)*x)*(x*x)))*(((y*((x*(z*y))*x))*z)*x))*((x*z)*x))*((x*(z*y))*(x*y)))*((((y*z)*z)*z)*(x*x)))*((x*y)*(y*x)))*((x*z)*(z*x)))$$

5-Element Gate

$*$	0	1	2	3	4
0	3	2	0	4	0
1	2	4	1	1	1
2	1	1	3	0	1
3	1	3	4	1	0
4	0	1	0	1	0

Possible target sequences: $5^{125} \approx 10^{87}$.

5-Element Gate

$*$	0	1	2	3	4
0	3	2	0	4	0
1	2	4	1	1	1
2	1	1	3	0	1
3	1	3	4	1	0
4	0	1	0	1	0

Possible target sequences: $5^{125} \approx 10^{87}$.

Typical solution to design problem, generated by **random search: 10^{74} years**

5-Element Gate

$*$	0	1	2	3	4
0	3	2	0	4	0
1	2	4	1	1	1
2	1	1	3	0	1
3	1	3	4	1	0
4	0	1	0	1	0

Possible target sequences: $5^{125} \approx 10^{87}$.

Typical solution to design problem, generated

- by **random search: 10^{74} years**
- by **2-parent evolution:**

5-Element Gate

$*$	0	1	2	3	4
0	3	2	0	4	0
1	2	4	1	1	1
2	1	1	3	0	1
3	1	3	4	1	0
4	0	1	0	1	0

Possible target sequences: $5^{125} \approx 10^{87}$.

Typical solution to design problem, generated

- by **random search: 10^{74} years**
- by **2-parent evolution: 5.347 sec:**

$$(((((x*y)*(x*z))*((z*(y*(z*y)))*(((z*(y*((z*y)*x)))*((((((((((z*z)*x)*z)*((((((x*x)*y)*z)*(((z*(z*z))*z)*((x*z)*(((x*(z*z))*x)*((y*(z*(y*x)))*((z*(x*x))*((((x*y)*z)*((y*(x*(x*y)))*((x*((z*z)*x))*((y*z)*((((z*z)*z)*z)*(((y*y)*y)*(((x*(x*z))*y)*((((((((y*(z*z))*(z*y))*(y*(x*y)))*(z*(y*(z*x))))*(((x*(y*(y*y)))*z)*z))*(x*((z*y)*x)))*((z*z)*(z*x)))))))))*(z*(y*x)))))))))*(y*z))*(((y*x)*x)*z)))*((x*x)*(x*y)))*((y*(x*z))*z))*(y*(x*y)))*(x*((y*z)*y)))*(x*y))*(z*(x*(z*z))))*(((y*z)*y)*y)))*(y*((z*x)*(x*z))))))*(((x*(x*x))*((((z*(y*y))*((((z*y)*(x*x))*((((((((x*(y*y))*z)*(((x*(x*(y*z)))*((((x*x)*(z*z))*((((z*y)*(x*y))*(((x*((z*y)*y))*(((y*x)*(z*x))*(((y*(z*y))*z)*(((y*((((((y*(z*z))*z)*((y*y)*(((z*x)*(x*(z*(x*y))))))*((x*(x*y))*z))*(z*((z*z)*z))))*((z*((z*y)*x))*z))*(x*((y*z)*x))))))*(y*((z*z)*z)))*((y*(x*x))*x)))*((y*(x*z))*x))))*(((z*z)*z)*y))*(((y*z)*y)*(x*y))))*((x*x)*z))*(((z*x)*y)*z))*(y*(y*(x*z))))*((z*z)*z))*((x*(z*z))*y)))*(z*((z*x)*y)))*((z*x)*(z*z))))*(x*((x*y)*z)))*((z*(x*z))*y)))*(((z*z)*$

$$(((z*(y*x))*((z*y)*(((y*((x*x)*x))*((z*z)*(((((z*z)*(x*z))*((x*(z*z))*((x*(y*(z*y)))*(((y*y)*x)*(((x*x)*(((y*z)*(z*x))*((x*z)*(x*(((x*x)*(z*x))*(x*((x*z)*y)))*(x*(z*y))))))))*(y*((x*z)*y))))*((y*y)*(x*x)))))*((y*y)*x))*((y*x)*(y*z)))))*(x*((x*x)*y)))))*((z*y)*x)))*(((x*((x*x)*x))*(x*(((y*x)*z)*(((z*y)*y)*((((x*(((z*(y*(x*y)))*(((z*(z*z))*y)*(((((z*(y*(z*x)))*(((x*y)*(z*x))*((z*(y*y))*((x*(x*y))*(((x*y)*y)*((z*y)*(((x*z)*x)*(((x*x)*((z*x)*y))*((x*y)*y)))))))))*(((z*y)*z)*z))*(x*(y*(z*z))))*x)*(((y*z)*y)*z))))*(z*(x*x))))*(y*(z*x)))*y)*x)))))*(((x*(x*x))*((x*(y*(y*z)))*(((((z*y)*(x*x))*((x*(y*y))*((x*((y*x)*x))*(((((z*y)*(z*y))*((x*(x*(((y*(y*z))*z)*z))*((z*((z*z)*z))*((((z*(z*y))*z)*z)*(((((x*y)*y)*(((x*z)*(z*z))*x))*(((x*x)*y)*x)))))))*(x*((y*x)*z))))*(z*((x*y)*y)))*(x*(z*(y*y))))))*(((z*x)*y)*z))*((z*(y*z))*y))))*((((z*x)*(((y*x)*y)*x))*((y*x)*(((x*x)*y)*(((((x*y)*z)*z)*((z*((y*y)*z))*((y*(z*x))*(((z*y)*y)*((y*x)*(((x*(z*z))*x)*(((z*y)*y)*((y*(x*y))*((((x*(x*(x*z)))*((z*((z*y)*x))*((y*(y*z))*$$

$$(((z*y)*(y*x))*((y*((y*z)*y))*((((x*z)*z)*x)*(x*y))))))*(z*y))*(x*(y*(x*x)))))))))))))*((((x*(y*(x*z)))*(((y*(y*x))*y)*((y*((y*z)*z))*((y*(x*(z*z)))*((x*z)*(((z*z)*((((y*y)*(x*x))*y)*(((x*(z*z))*z)*(z*(z*(x*z)))*(x*z)))))*(x*(y*y)))))))))*((((((((((x*(z*z))*z)*(((x*(z*y))*z)*((z*(y*((y*y)*y)))*((z*((y*(z*x))*x))*((z*x)*(((y*x)*(z*y))*(((y*y)*(y*z))*((y*(z*y))*(x*y))))))))*(((x*(y*(y*y)))*((x*(y*x))*((y*(y*(x*x)))*((((z*z)*z)*y)*(z*((z*((z*y)*z))*(y*y))))))))*((((((((x*z)*(y*z))*((z*(y*(y*x)))*x))*(((x*(z*(z*x)))*(y*y))*(((((y*y)*z)*z)*(y*z))*((((z*x)*y)*z)*(((y*(y*(y*x)))*z)*((((z*z)*z)*z)*((((y*((y*x)*x))*z)*x)*((x*x)*y))))))*(((((y*y)*z)*z)*x)))*((y*(x*(z*z)))*y))))*((y*(x*(x*z)))*(y*x))))*((y*(z*(y*z)))*x))))*(((((y*y)*y)*y)*y)*y))*(((y*z)*(x*(y*z)))*((y*x)*(((y*(z*x))*x)*((x*(z*z))*(((z*x)*x)*(x*x))))))*((z*(x*(z*y)))*x))*((x*z)*y))))*((z*z)*(x*x)))*((z*((y*z)*x))*((x*(z*(x*(y*z))))*((y*z)*(x*y)))))*(((y*(y*y))*y)*(x*z)))*(((z*y)*y)*((y*(y*x))*$$

$$(((x*z)*y)*(((((x*(z*z))*z)*((x*x)*(((y*y)*(y*z))*(z*y))))*(z*((x*z)*y)))*((x*(y*z))*y)))))*(((x*x)*y)*((x*((x*x)*z))*((z*(y*(z*x)))*z)))))*((((x*z)*z)*x)*(((z*x)*(z*y))*((x*(x*(y*z)))*(((x*(z*x))*(((x*(y*(x*(z*x))))*((y*x)*((((z*x)*z)*z)*(((y*y)*((y*y)*(z*z)))*((x*x)*z)))))*(x*(x*x))))*((z*x)*x)))))))*(((x*z)*(x*x))*((((z*y)*x)*((z*(z*(y*z)))*(((((x*x)*z)*x)*(((((((z*x)*y)*((y*z)*((y*y)*x)))*((x*(x*x))*z)))*(y*(x*y)))*(z*((y*z)*y)))*((y*(y*z))*y))*(x*z))*x))*(((y*(y*x))*z)*z))*(z*(y*(x*y))))))*(x*((y*z)*y))))))))$$

Thanks for listening!

— DC