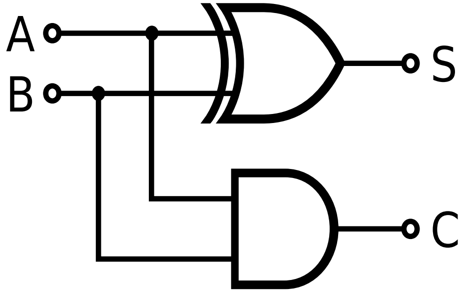
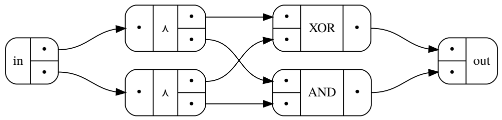
Diagrammatic semantics for digital circuits



George Kaye
University of Birmingham
Conference of Research Skills 2020
January 27



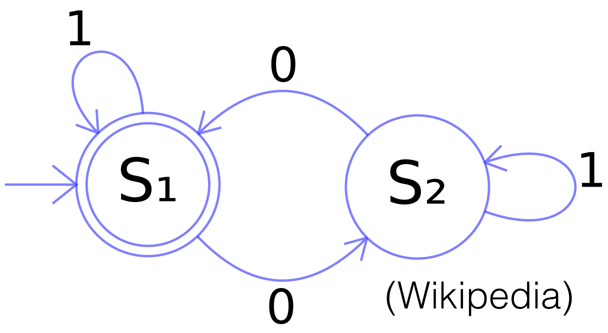
Introduction

There are two main modelling methodologies in computer science:

Software: Operational semantics

```
triple(2 * 3)
 = triple(6)
 = 6 + 6 + 6
 = 12 + 6
 = 18
```

Hardware: Simulation



(Wikipedia)

Simulation obfuscates the design of hardware…
could we use operational semantics instead?

Outline

- Operational semantics
- Graphs and hypergraphs
- Circuits as hypergraphs
  - How to construct them
  - How to reduce them

Operational semantics

Use syntactic reductions with the aim of reaching a value.

We can also use **partial evaluation** to only perform some of the steps.

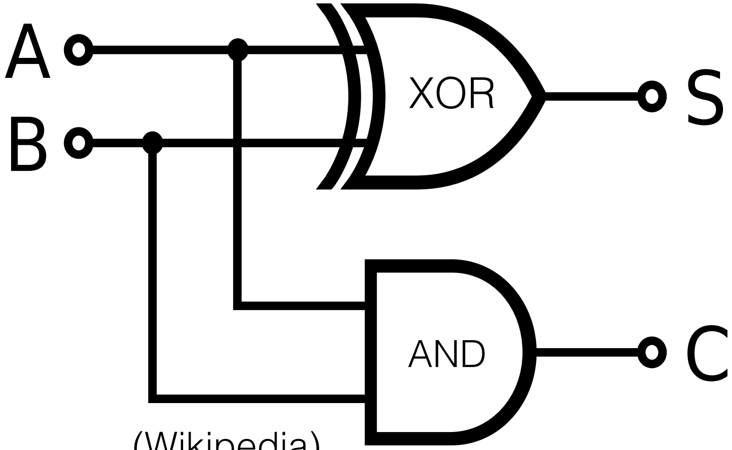
So how could we bring this to hardware?

```
int triple (int x) {
   return x + x + x;
}

triple(2 * 3)
 = triple(6)
 = 6 + 6 + 6
 = 12 + 6
 = 18
```

Circuits as equations?

We could represent circuits as equations...



(Wikipedia)

$$1 \otimes \curlywedge \cdot \curlywedge \otimes 2 \cdot 1 \otimes \times_{1,1} \otimes 1 \cdot \oplus \otimes \wedge$$

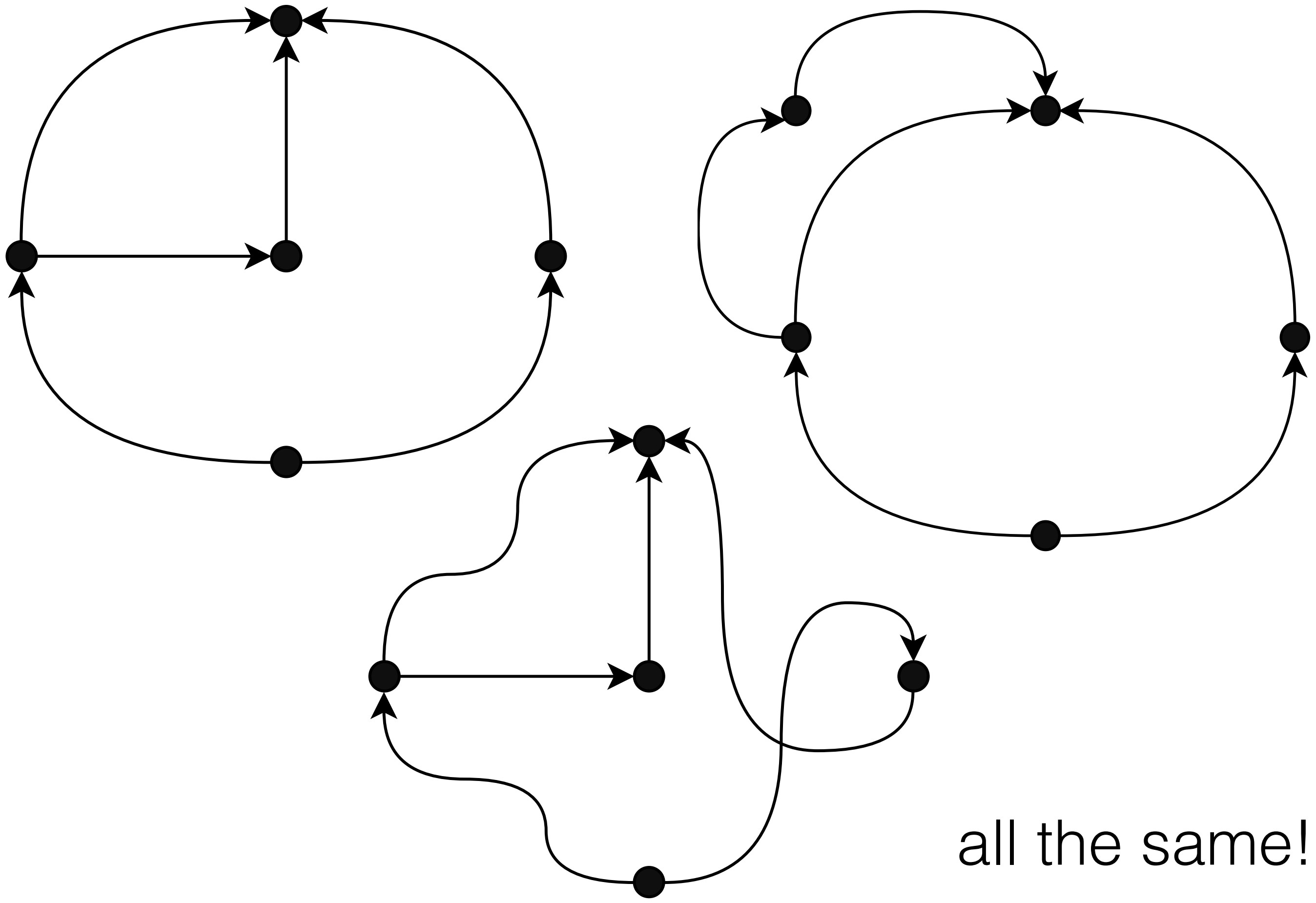
… but this is very confusing!

It's also computationally difficult to identify where we can perform reductions.

Graphs

A collection of nodes and edges that can connect at most two nodes.

We are only concerned with the connections between nodes.

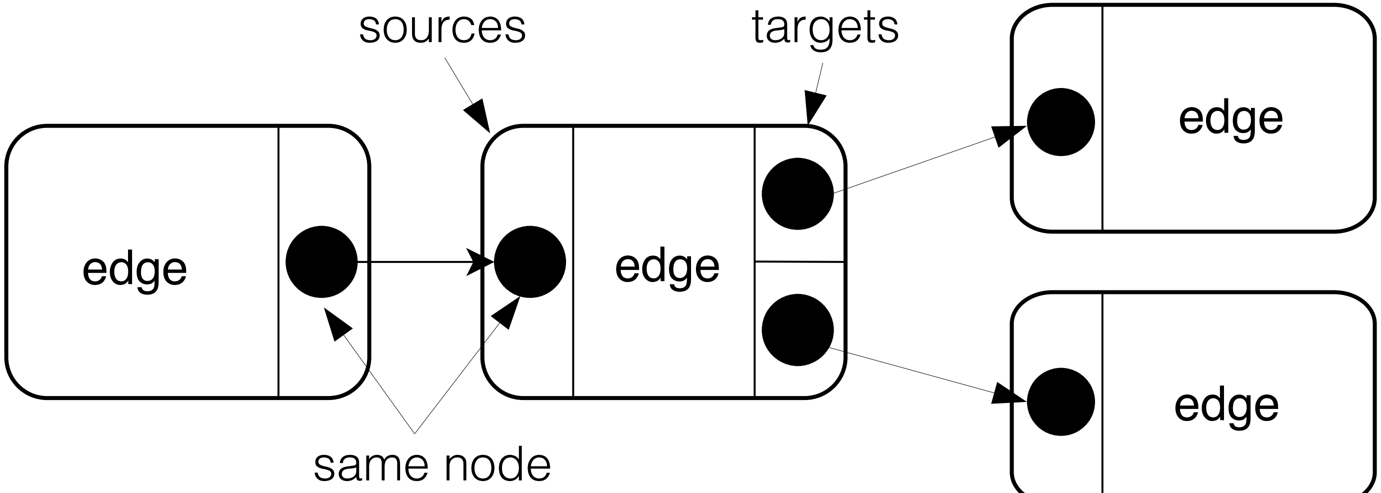


all the same!

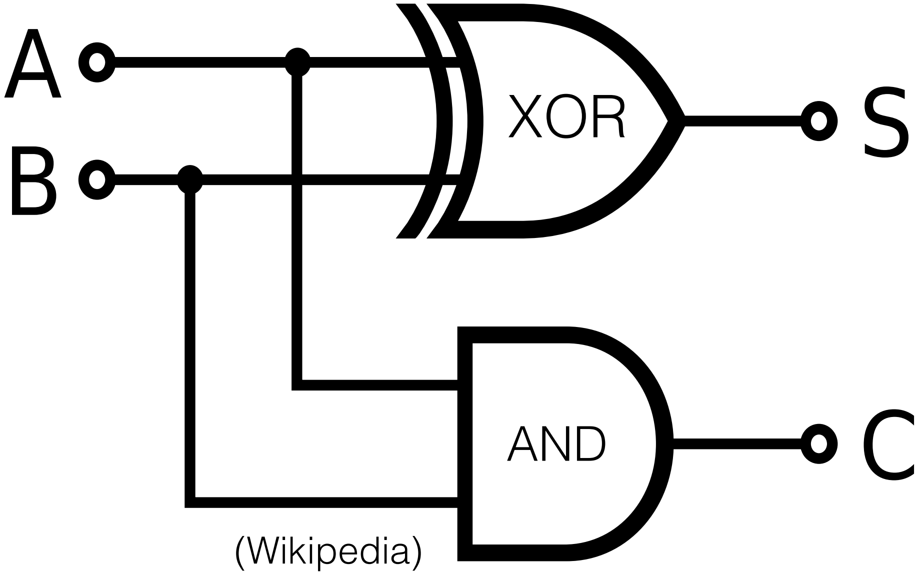
Hypergraphs

A graph with *hyperedges* that can connect to multiple nodes.

Hyperedges can have *source* nodes and *target* nodes.

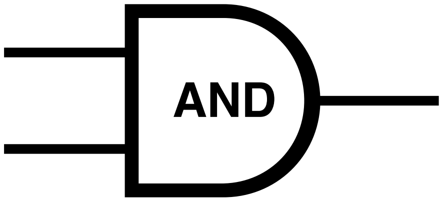


So how do we make a circuit hypergraph?

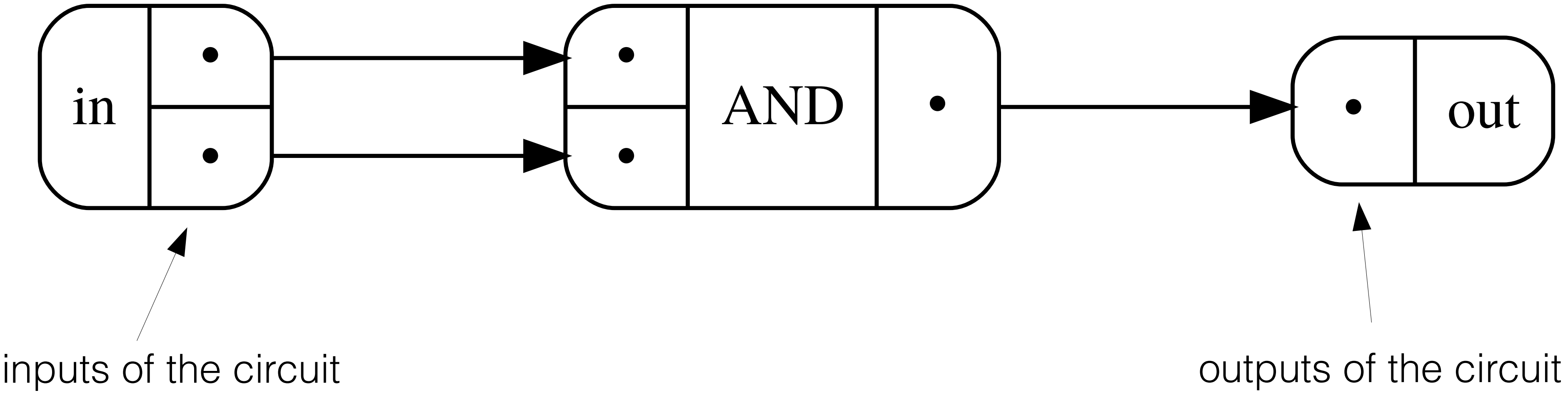


(Wikipedia)

Circuits as hypergraphs



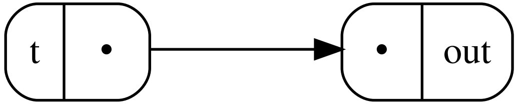
We keep track of the inputs and outputs:



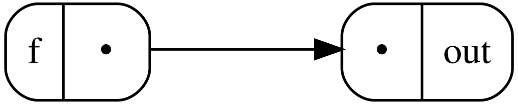
diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Circuits as hypergraphs

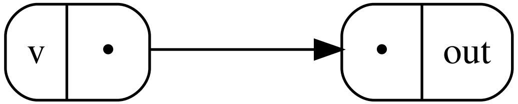
Values are what we can feed to our circuits:



true / 1



false / 0

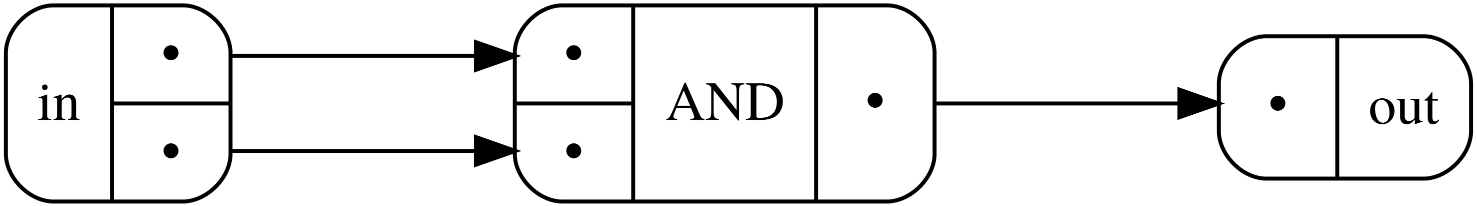


variable

diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Circuits as hypergraphs

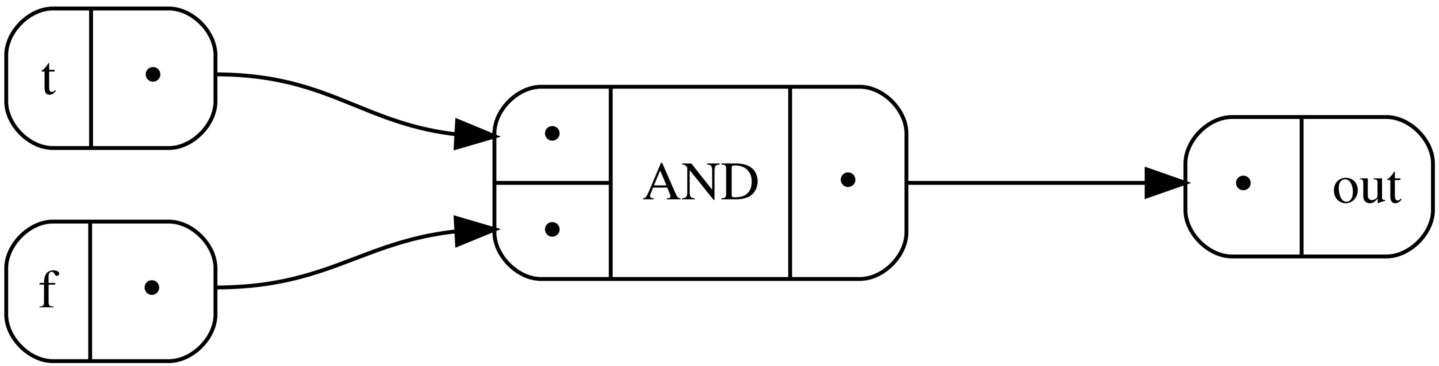
To apply a circuit to values, we replace the input edge:



diagrams generated by *https://georgejkaye.com/circuits/visualiser*

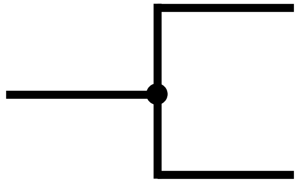
Circuits as hypergraphs

To apply a circuit to values, we replace the input edge:

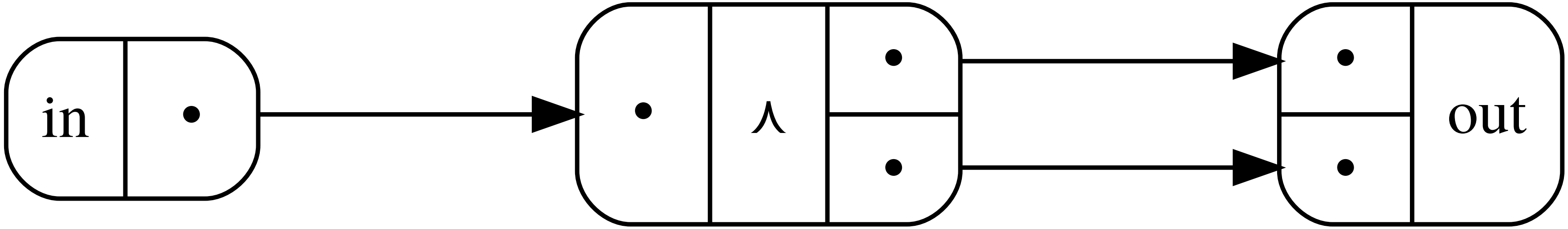


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Circuits as hypergraphs



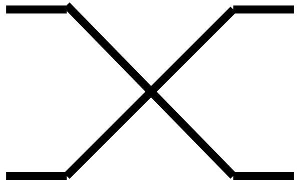
We can also **fork** (⋏) wires:



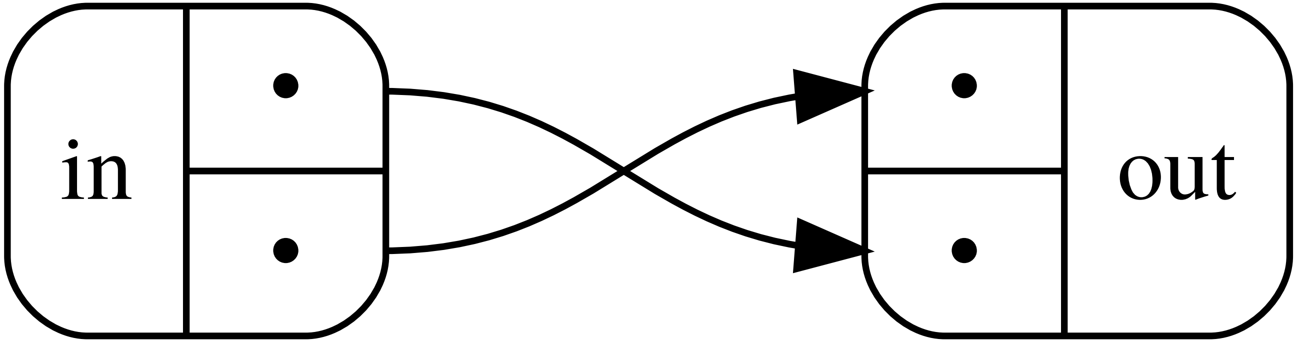
This represents the duplication of a value.

diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Circuits as hypergraphs

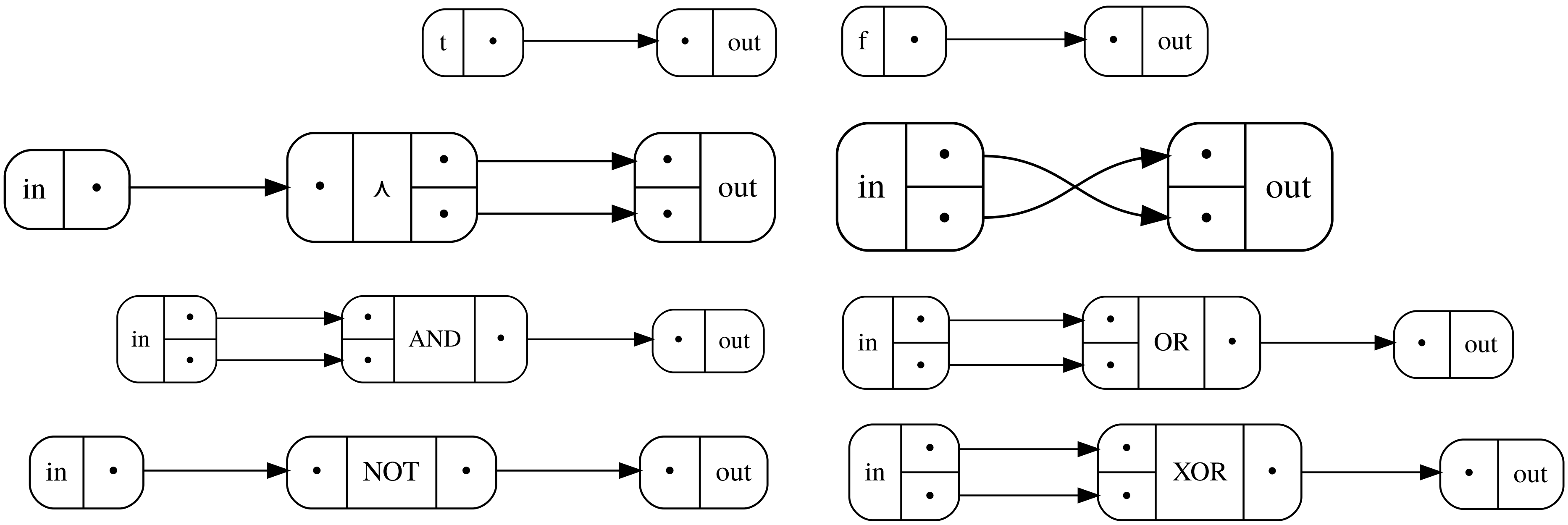


And **swap** wires:



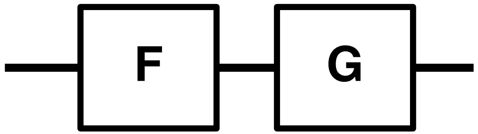
diagrams generated by *https://georgejkaye.com/circuits/visualiser*

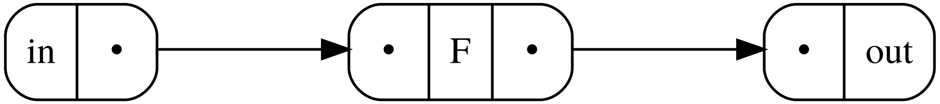
Making bigger circuits

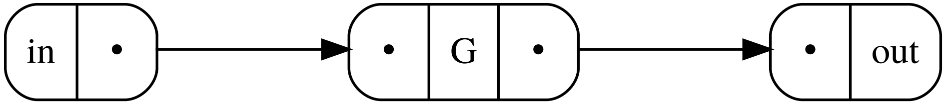


Composing circuit hypergraphs

We can compose hypergraphs **sequentially**...



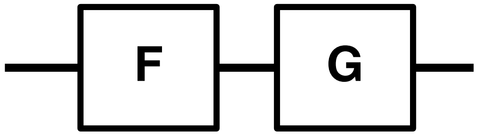




diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Composing circuit hypergraphs

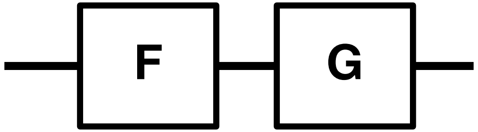
We can compose hypergraphs **sequentially**...

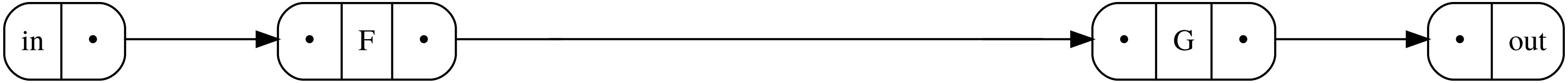


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Composing circuit hypergraphs

We can compose hypergraphs **sequentially**...

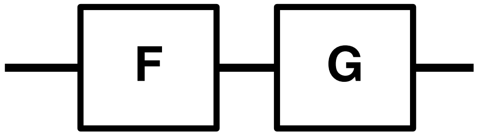


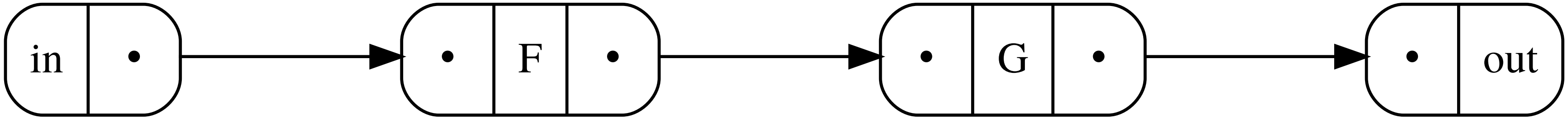


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Composing circuit hypergraphs

We can compose hypergraphs **sequentially**...

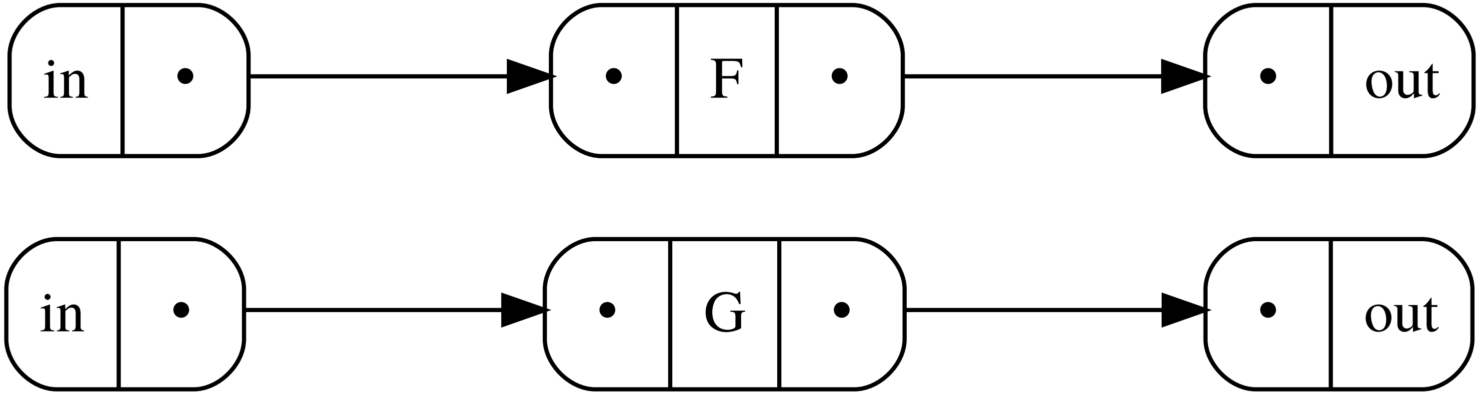


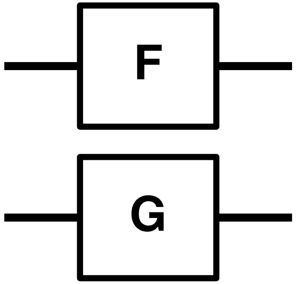


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Composing circuit hypergraphs

...or in **parallel**!

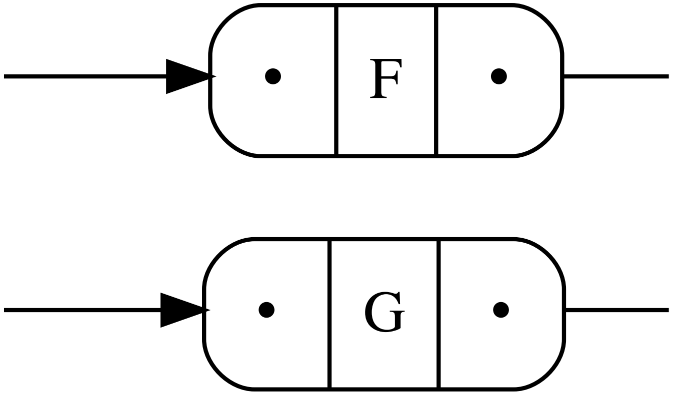


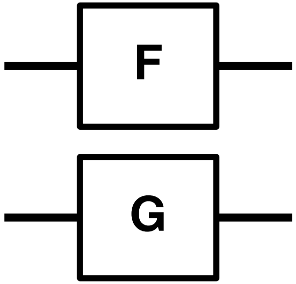


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Composing circuit hypergraphs

...or in **parallel**!

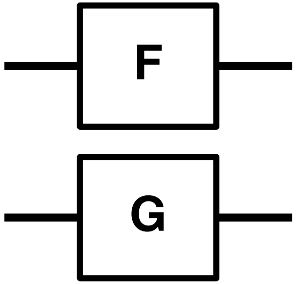


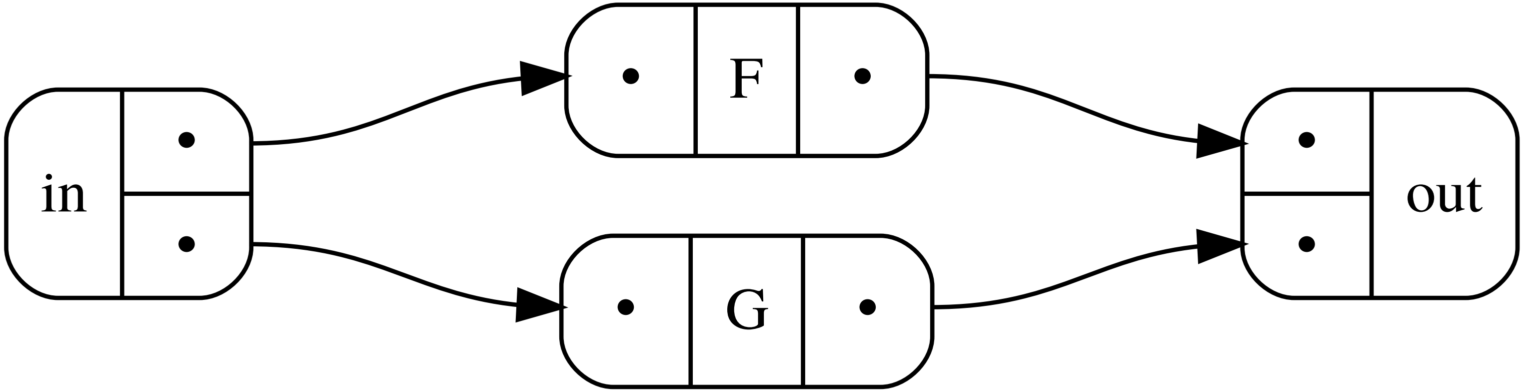


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Composing circuit hypergraphs

...or in **parallel**!

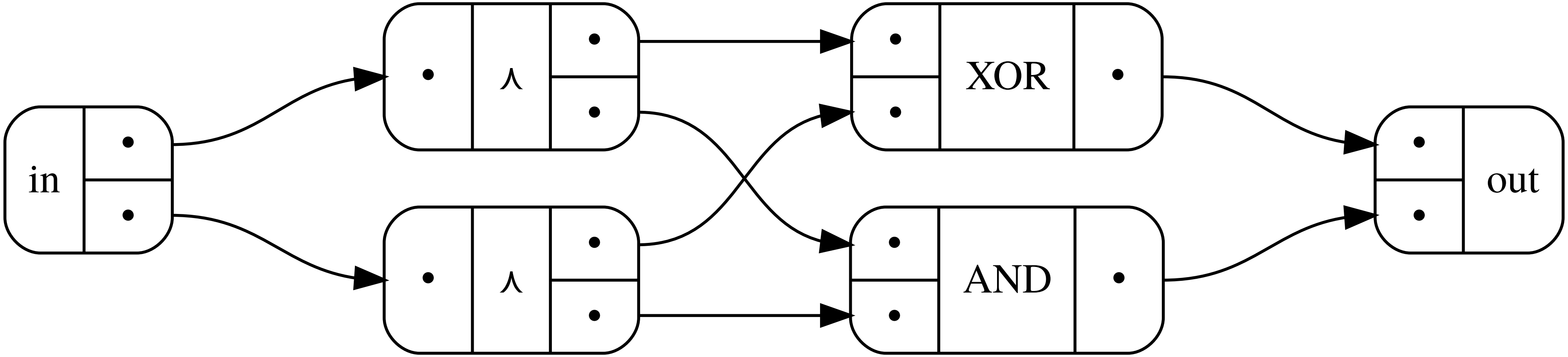




diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Reducing circuit hypergraphs

The purpose of operational semantics was to be able to reduce circuits to some value.

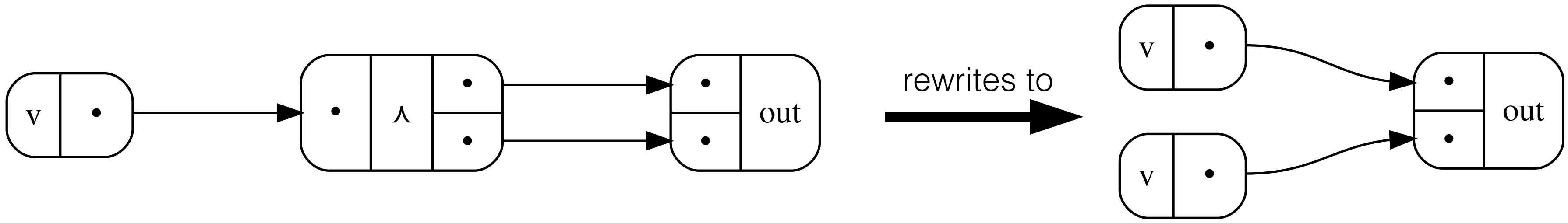


diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Reducing circuit hypergraphs

We can use **graph rewrites** to perform reductions:

e.g. **forking** a wire

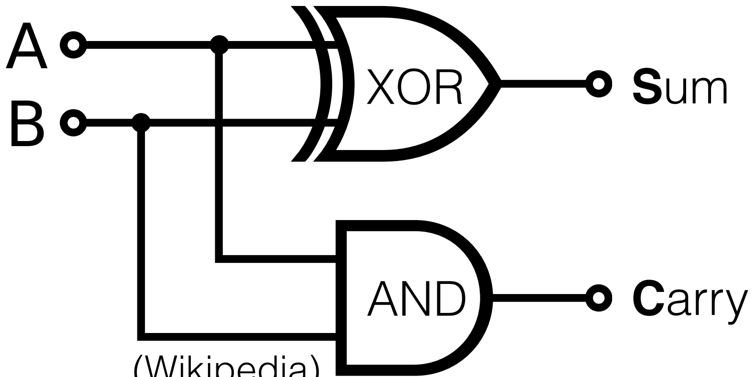


This is **efficient** since we can identify rewrites in linear time.

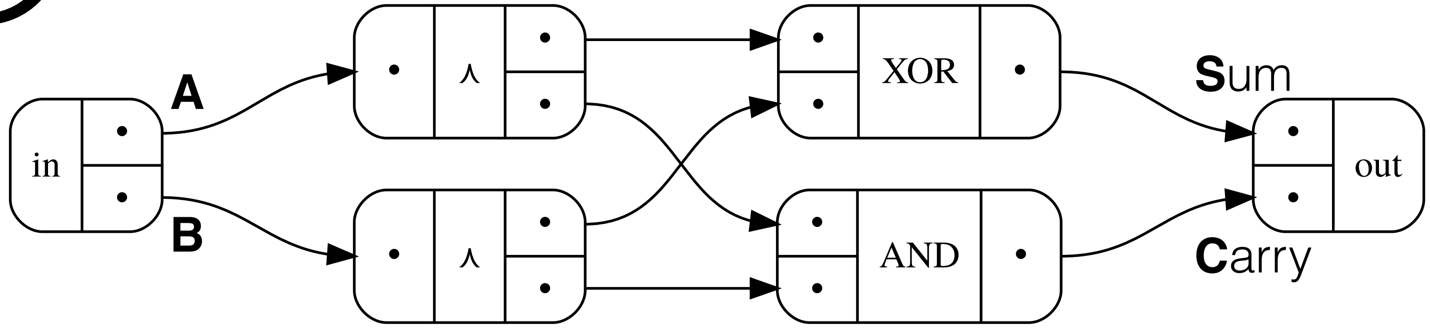
diagrams generated by *https://georgejkaye.com/circuits/visualiser*

Example Half-adder

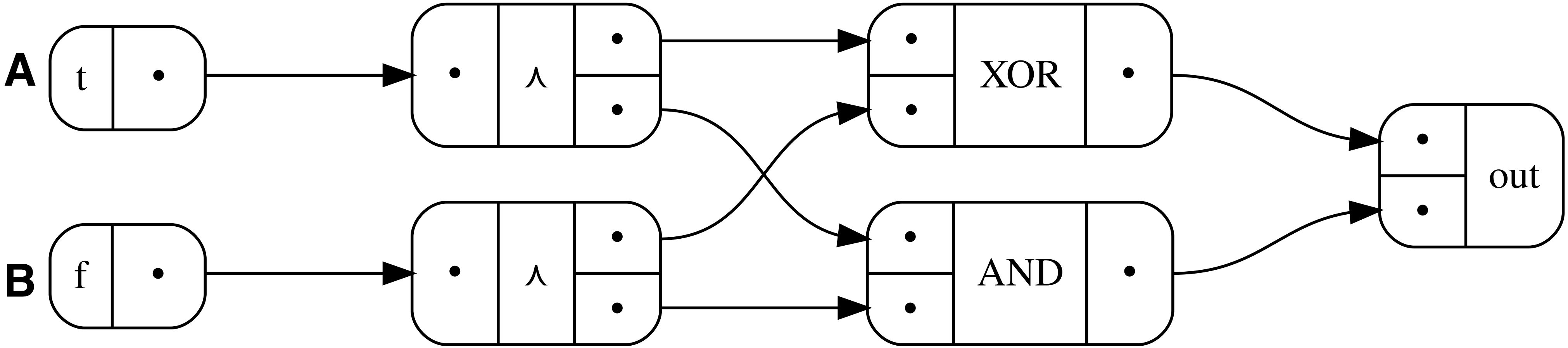
A	B	Sum	Carry
0	0	0	0
1	0	1	0
0	1	1	0
1	1	1	1



(Wikipedia)

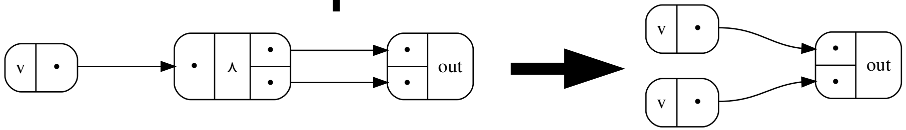


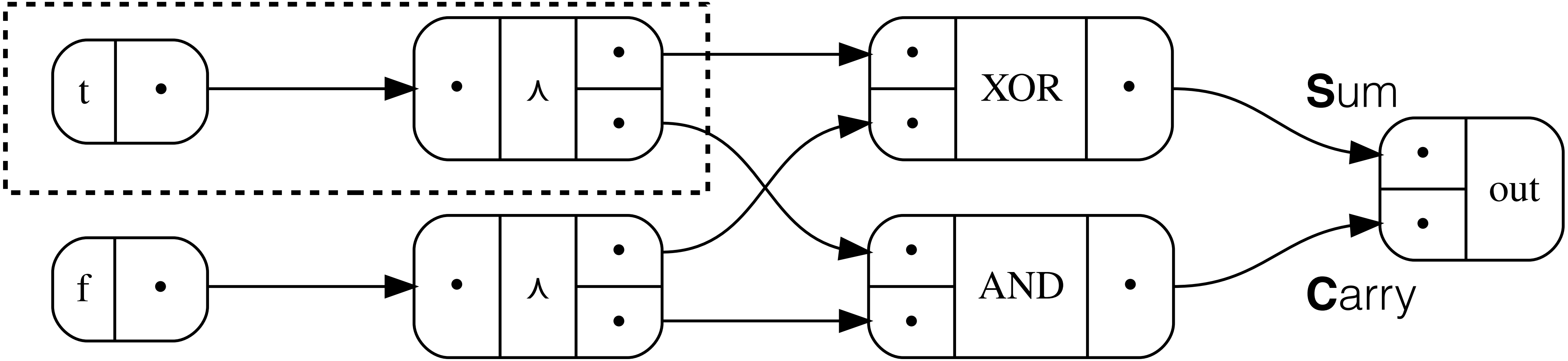
Example Half-adder applied: 1 + 0



Example

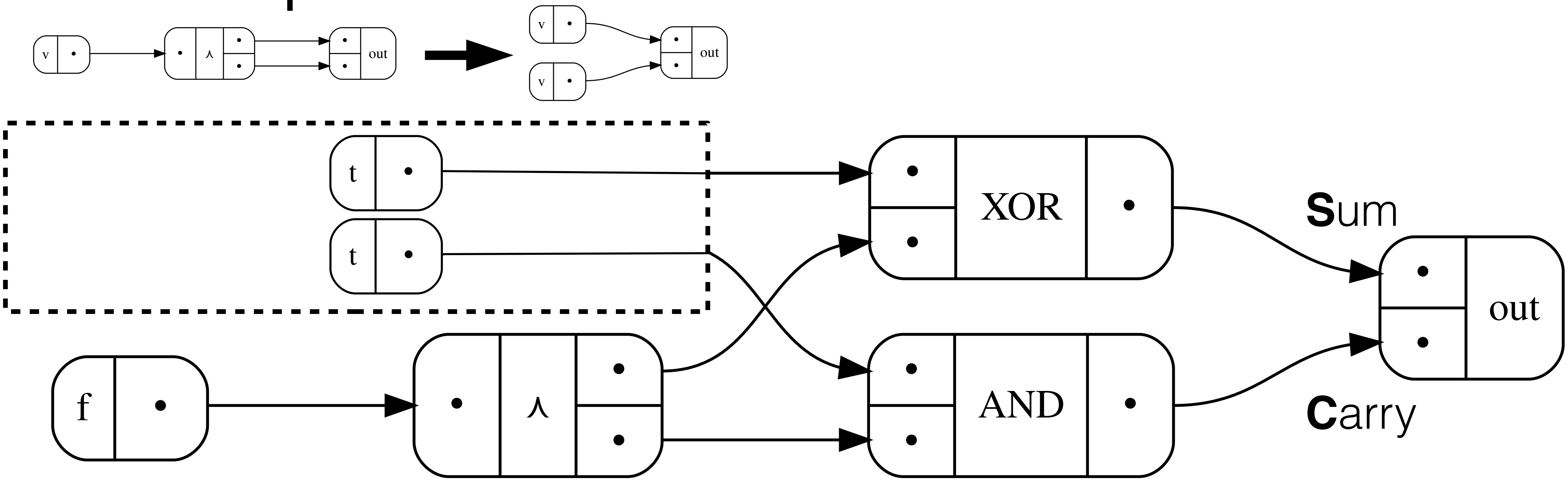
Half-adder applied: 1 + 0



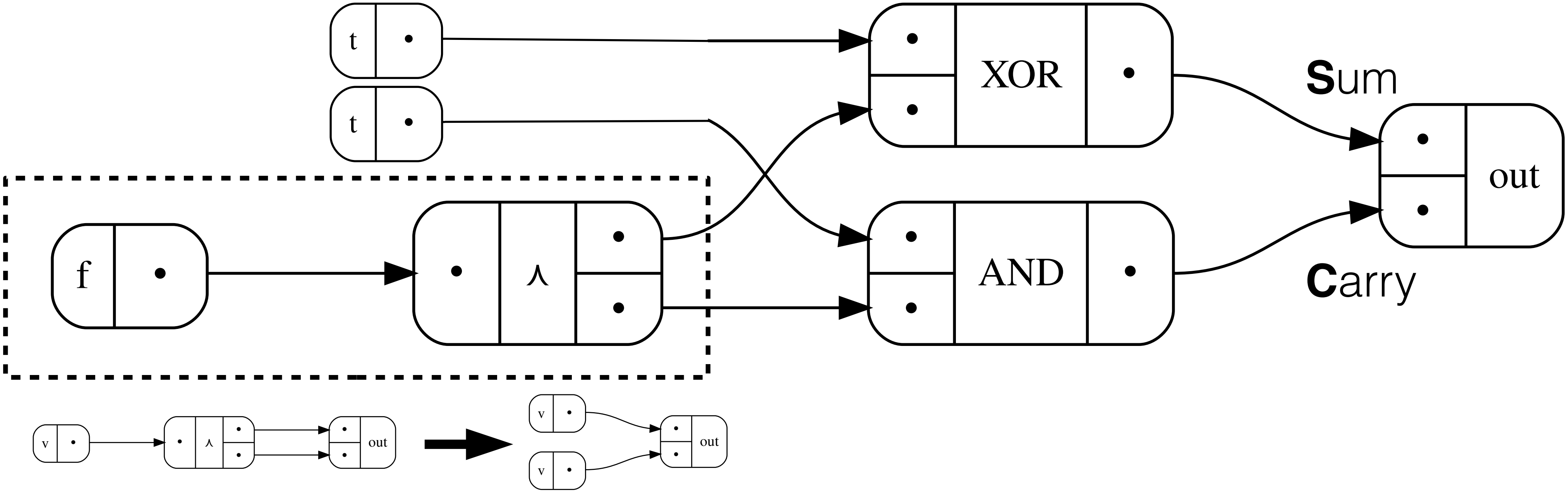


Example

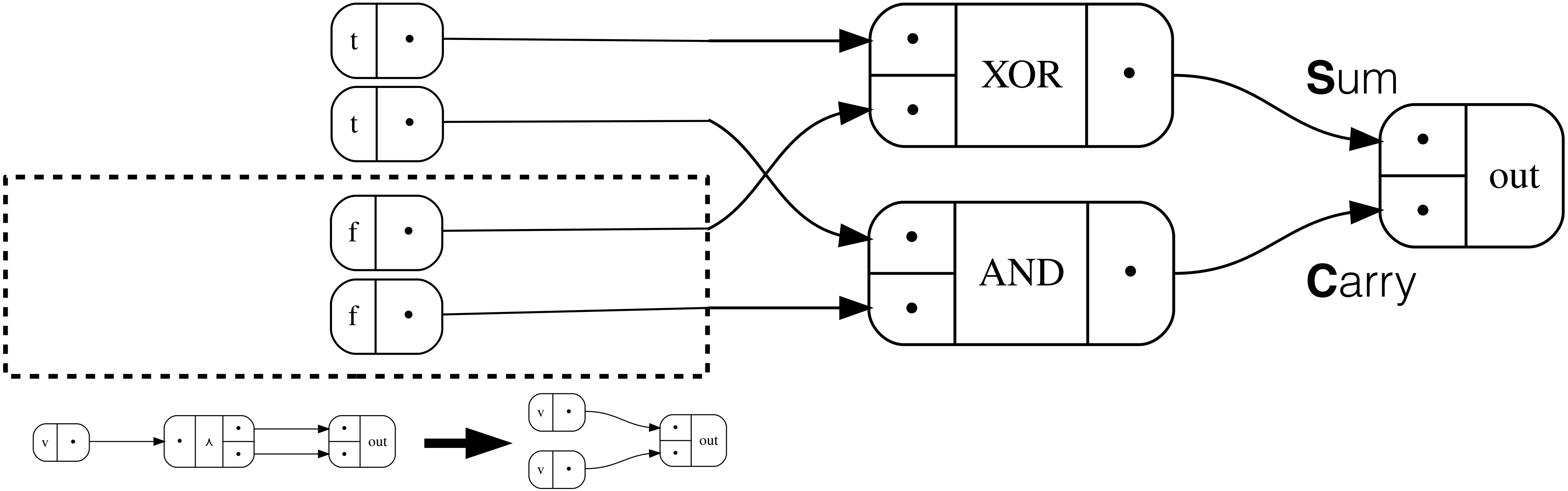
Half-adder applied: 1 + 0



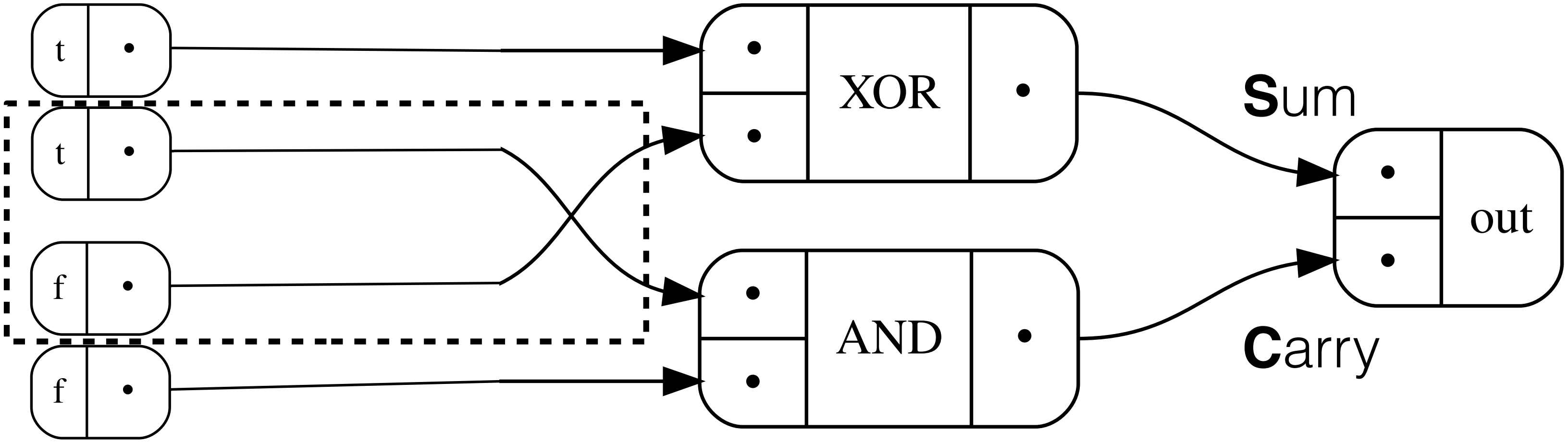
Example Half-adder applied: 1 + 0



Example Half-adder applied: 1 + 0

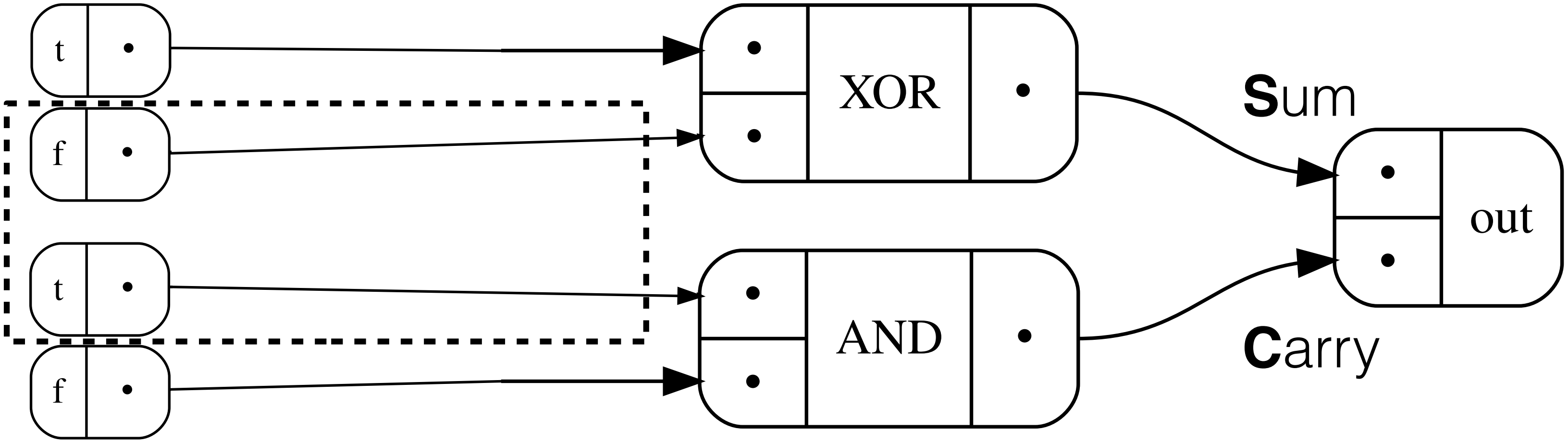


Example Half-adder applied: 1 + 0



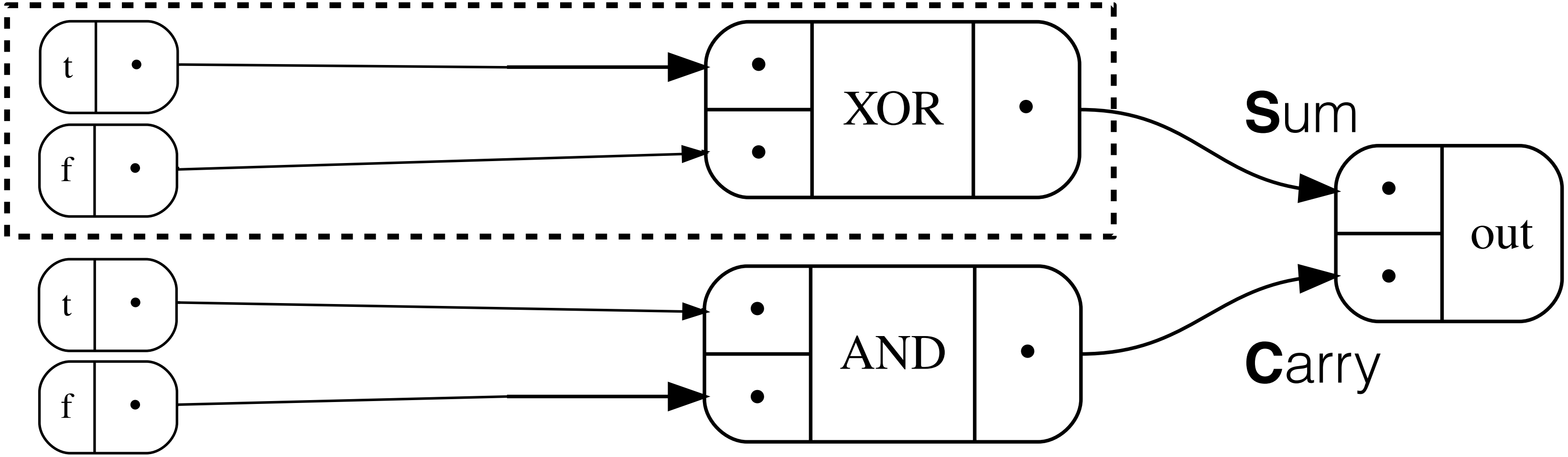
Example

Half-adder applied: 1 + 0



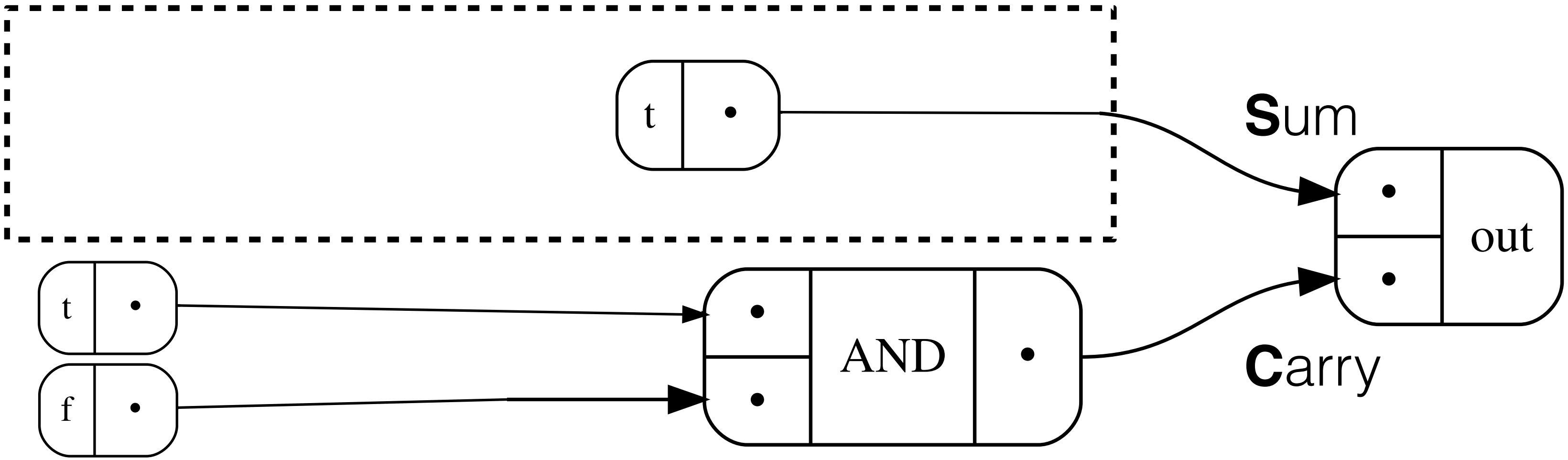
Example
Half-adder applied: 1 + 0

true **XOR** false = true

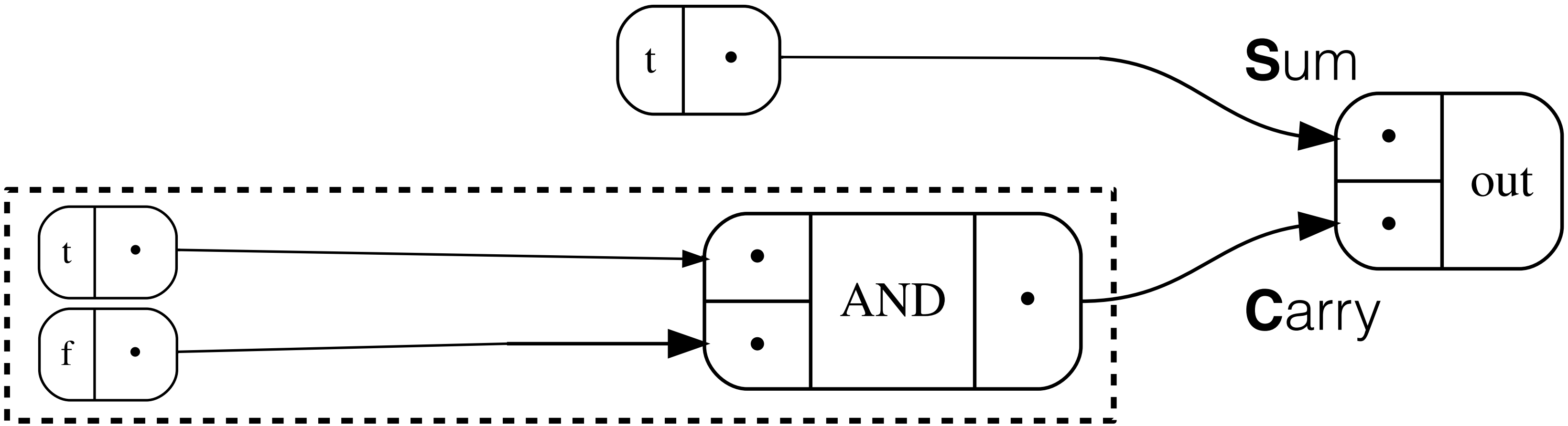


Example Half-adder applied: 1 + 0

true **XOR** false = true

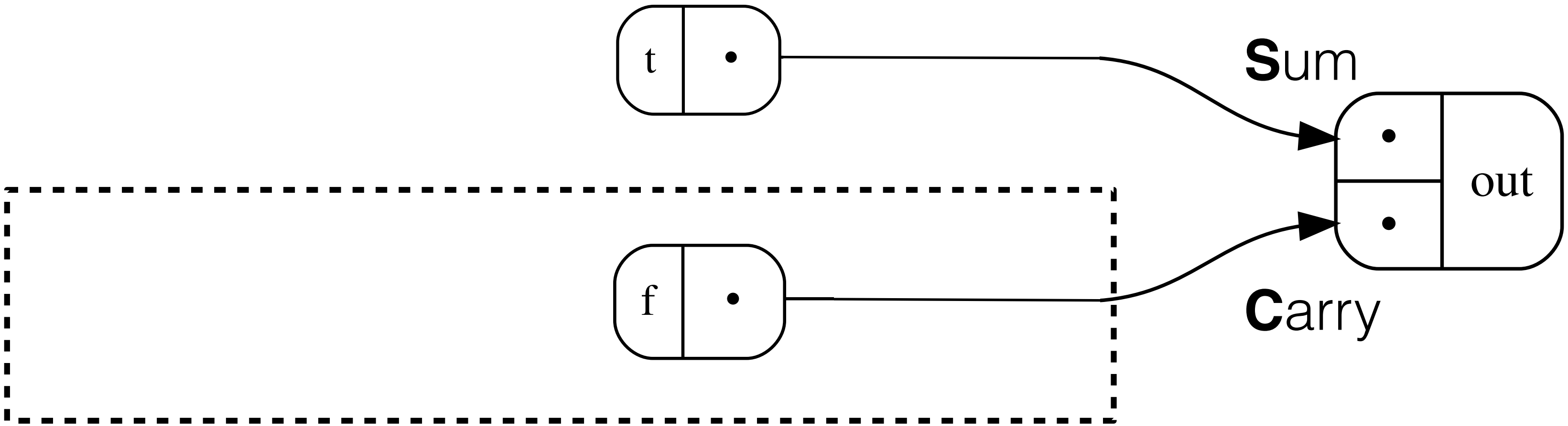


Example Half-adder applied: 1 + 0



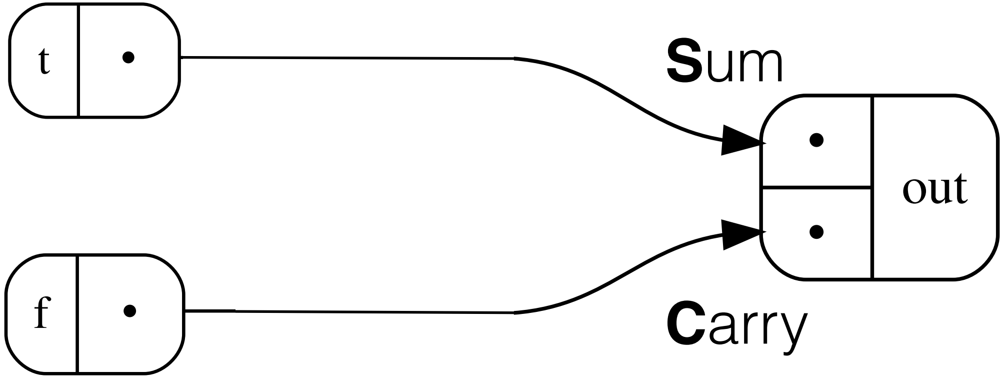
true **AND** false = false

Example Half-adder applied: 1 + 0

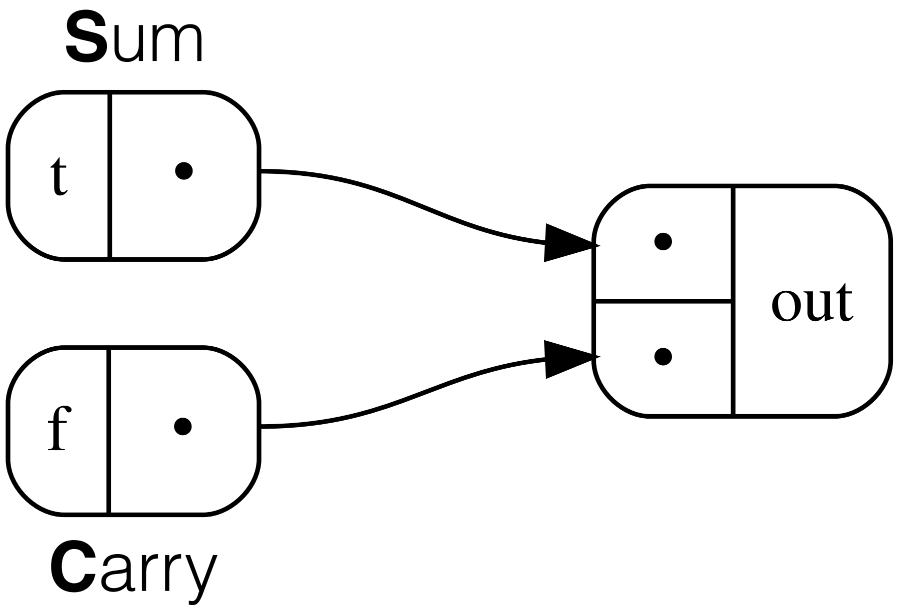


true **AND** false = false

Example Half-adder applied: 1 + 0



Example Half-adder applied: 1 + 0



A	**B**	**S**um	**C**arry
0	0	0	0
1	**0**	**1**	**0**
0	1	1	0
1	1	1	1

it's correct!

Conclusion

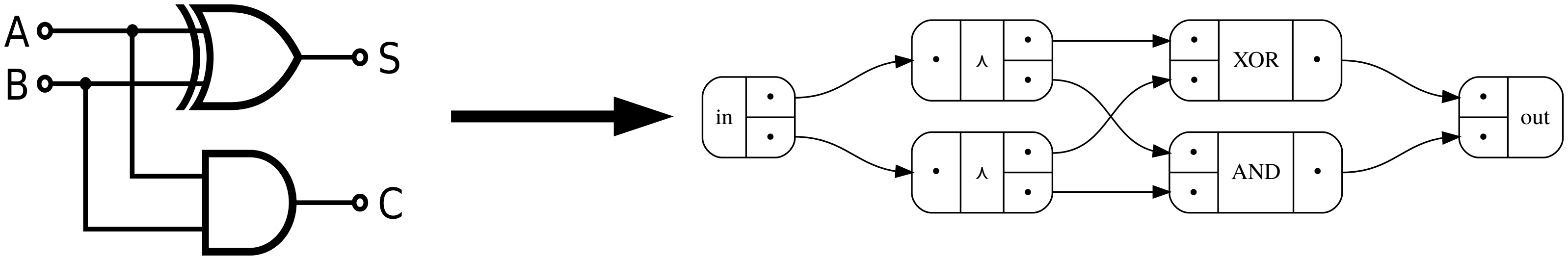
We want to use **operational semantics** for circuits

so we can reduce circuits to a value step by step

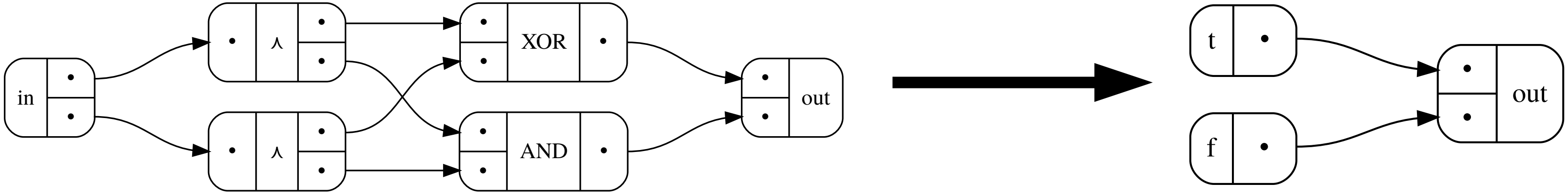
Can we do this? **Yes!**

Conclusion

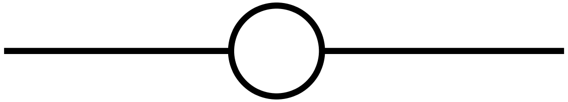
We can represent circuits as hypergraphs



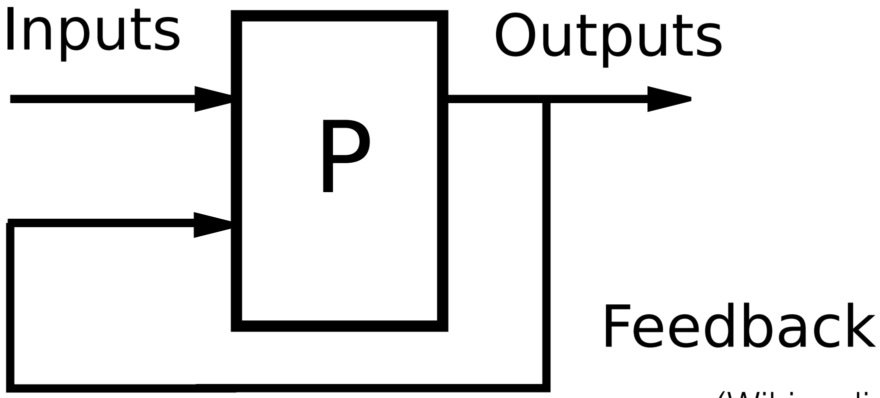
We can use graph rewrites to reduce them



Conclusion - what else?



Delay



(Wikipedia)

Feedback

Conclusion - what next?

$$1 \otimes \curlywedge \cdot \curlywedge \otimes 2 \cdot 1 \otimes \times_{1,1} \otimes 1 \cdot \oplus \otimes \wedge$$

$$\cong \textbf{?}$$