Announcements

- Exam 3 solutions online
- No HW this week
- Final exam next week (comprehensive)

Last Time

- NP Problems and NP completeness
  - Zero-one equations
  - Subset sum
  - Knapsack

NP

Such problems are said to be in Nondeterministic Polynomial time (NP).

NP-Decision problems ask if there is some object that satisfies a polynomial time-checkable property.

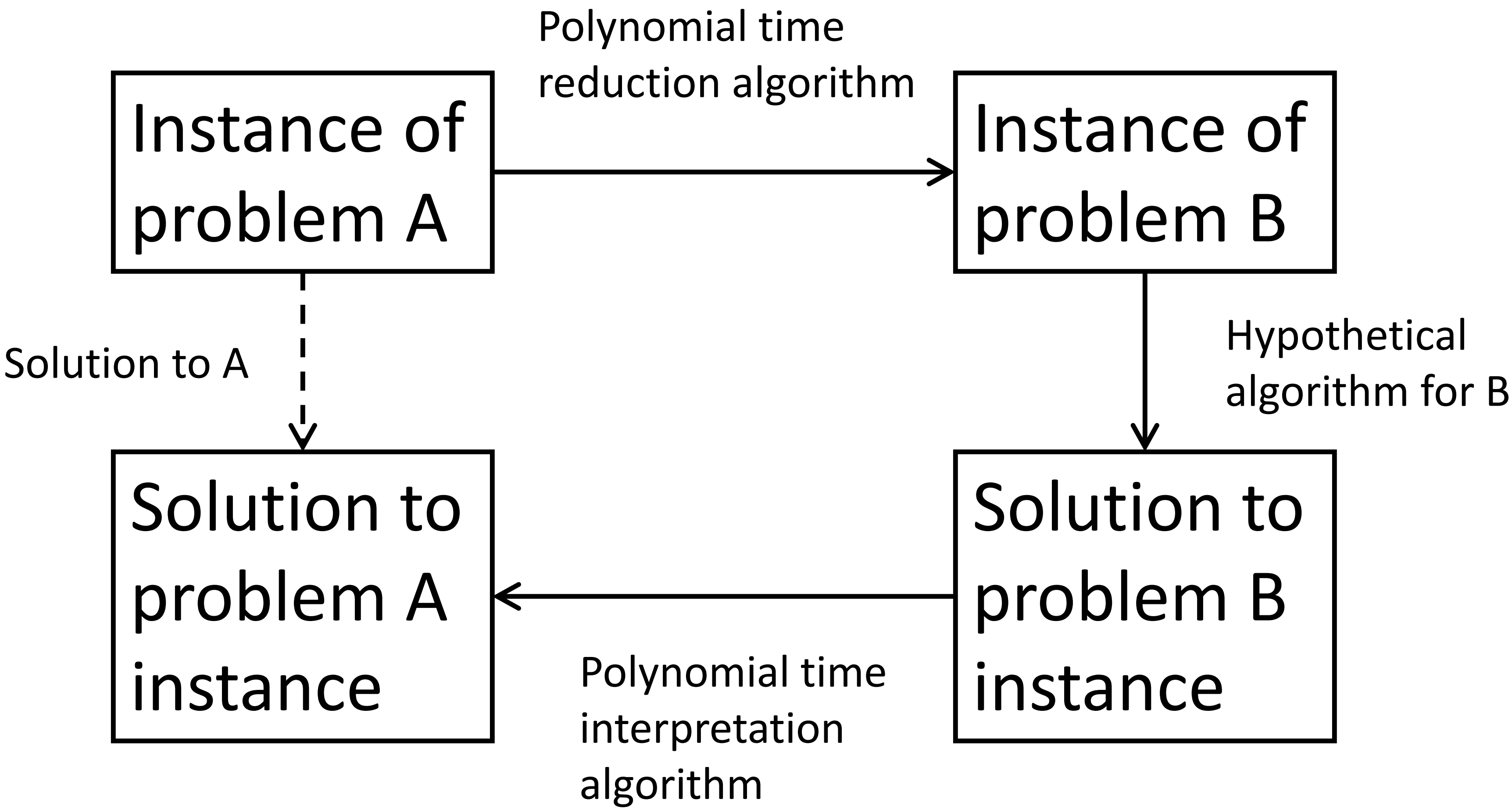
NP-Optimization problems ask for the object that maximizes (or minimizes) some polynomial time-computable objective.

Reductions

Reductions are a method for proving that one problem is at least as hard as another.

We show that if there is an algorithm for solving B, then we can use this algorithm to solve A. Therefore, A is no harder than B.

Reduction A → B



NP-Complete

Circuit-SAT is our first example of an <u>NP-Complete</u> problem. That is a problem in NP that is at least as hard as any other problem in NP.

<u>Examples:</u>

Circuit SAT

3SAT

Maximum Independent Set

Zero-One Equations

Subset Sum

Knapsack

Zero-One Equations

Problem: Given a matrix A with only 0 and 1 as entries and b a vector of 1s, determine whether or not there is an x with 0 and 1 entries so that

$$Ax = b.$$

Example

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Equivalently, do there exist x_1, x_2, $x_3 \in \{0,1\}$ so that

$x_1+x_3 = 1$

$x_1+x_2 = 1$

Generally, this is what a ZoE looks like. A bunch of sets of x_is that need to add to 1.

Today

- NP Completeness of Hamiltonian Cycle
- Dealing with NP completeness

One Final Reduction

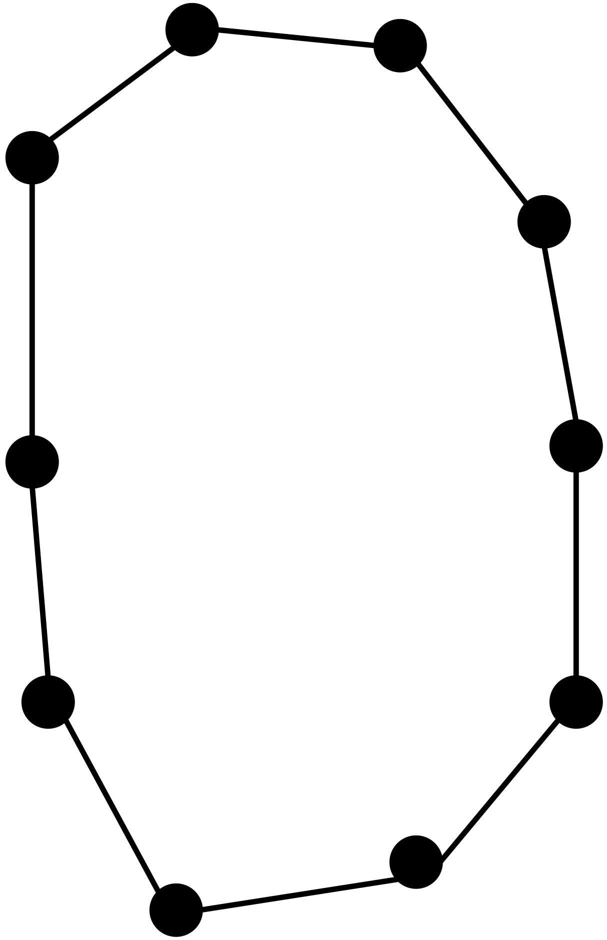
The last reduction we are going to show is ZOE → Hamiltonian Cycle. This will show that both Hamiltonian Cycle and TSP are NP-Complete/Hard.

Strategy

Often in order to show that a problem is NP-Complete, you want to be able to show that it can simulate logic somehow.

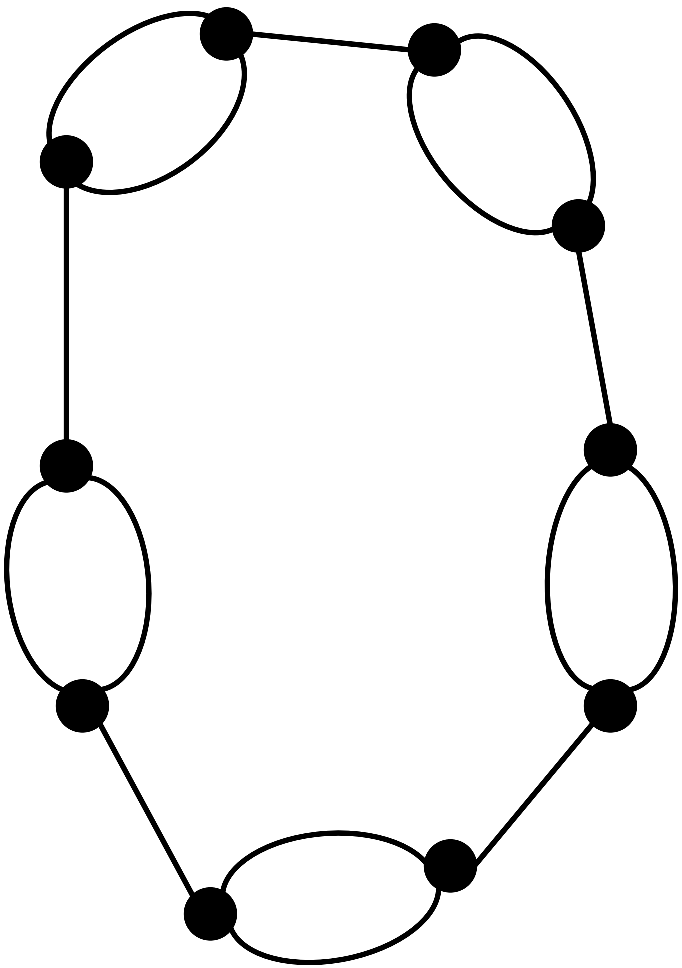
This is a bit difficult for Hamiltonian Cycle as most graphs have too many options, so we will want to find specific graphs with clear, binary choices.

Strategy



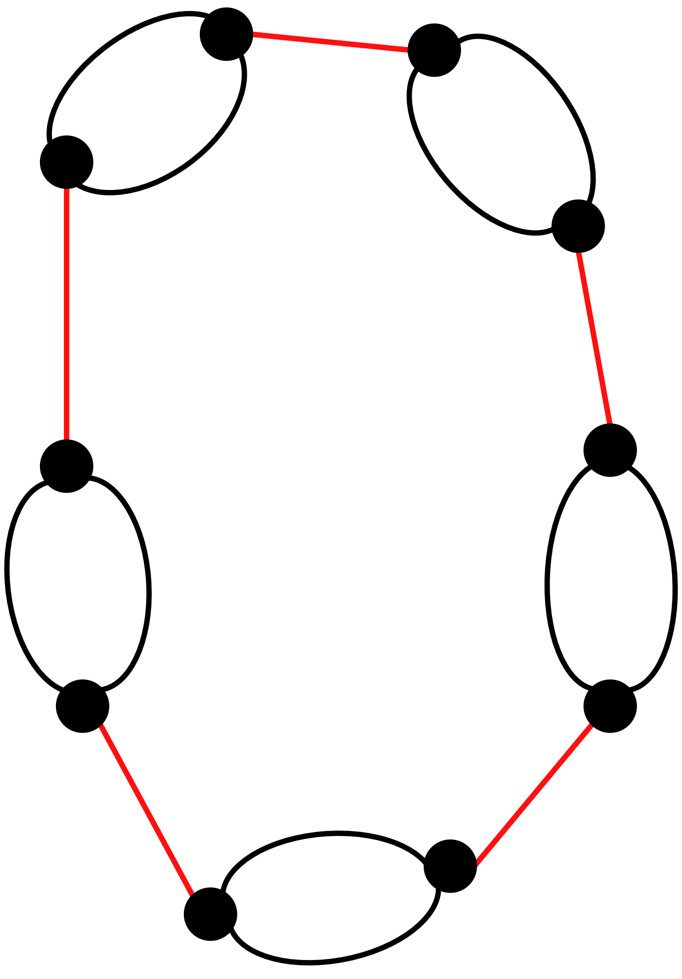
- Start with a cycle

Strategy



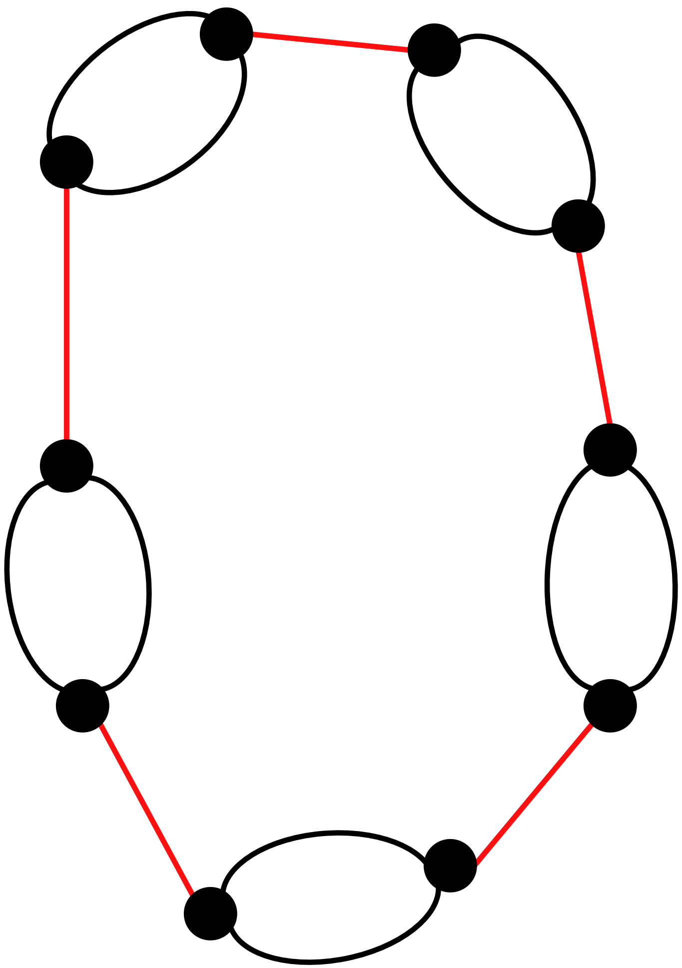
- Start with a cycle
- Double up some edges

Strategy



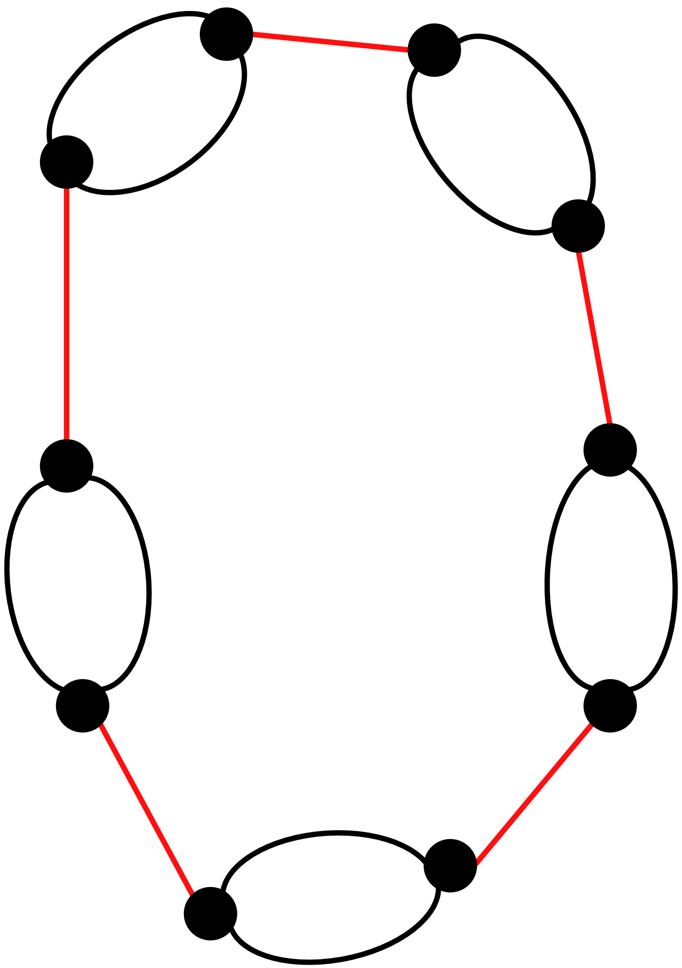
- Start with a cycle
- Double up some edges
- Cycle must pick one edge from each pair.

Strategy



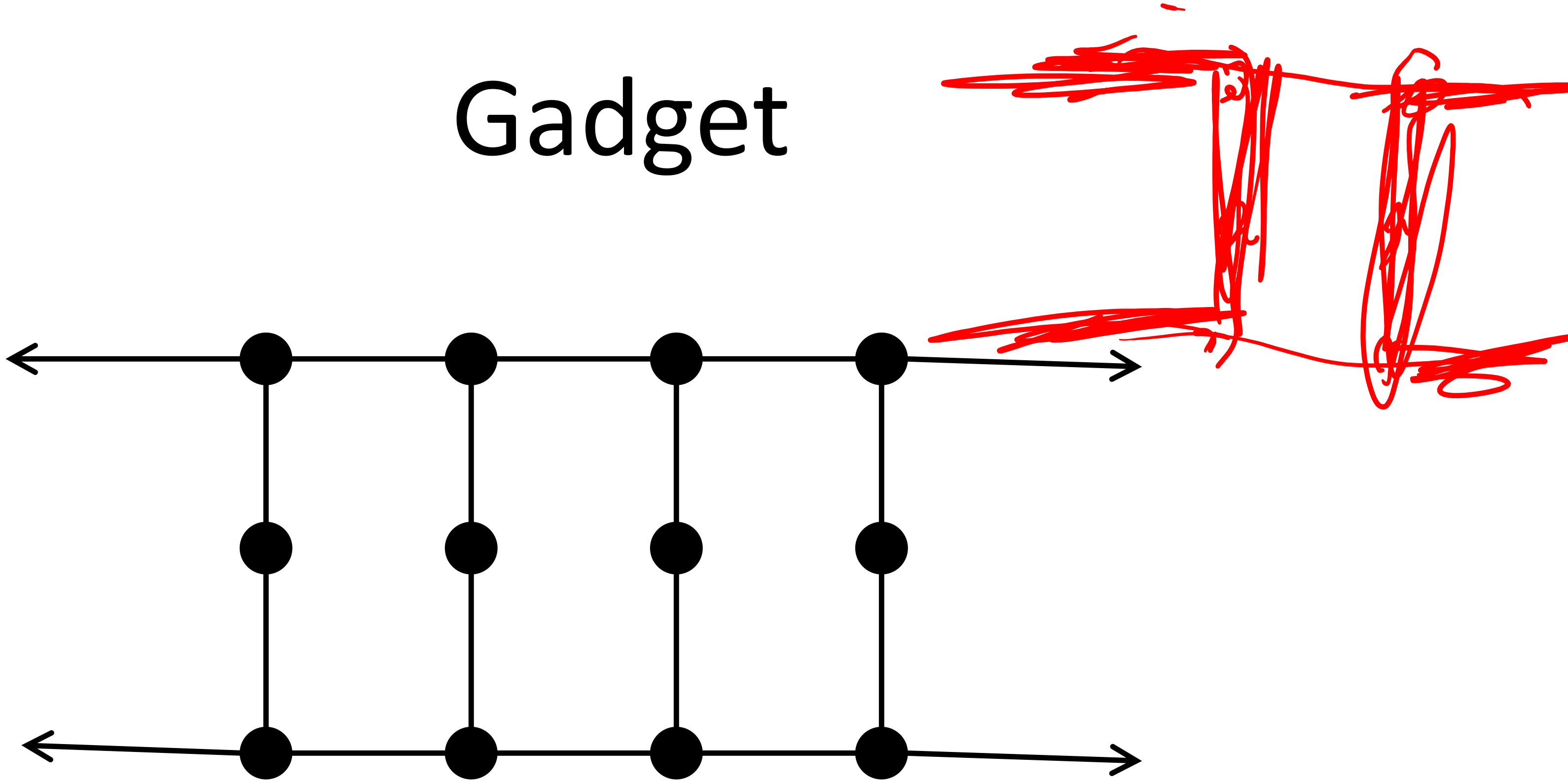
- Start with a cycle
- Double up some edges
- Cycle must pick one edge from each pair.
  - This provides a nice set of binary variables

Strategy

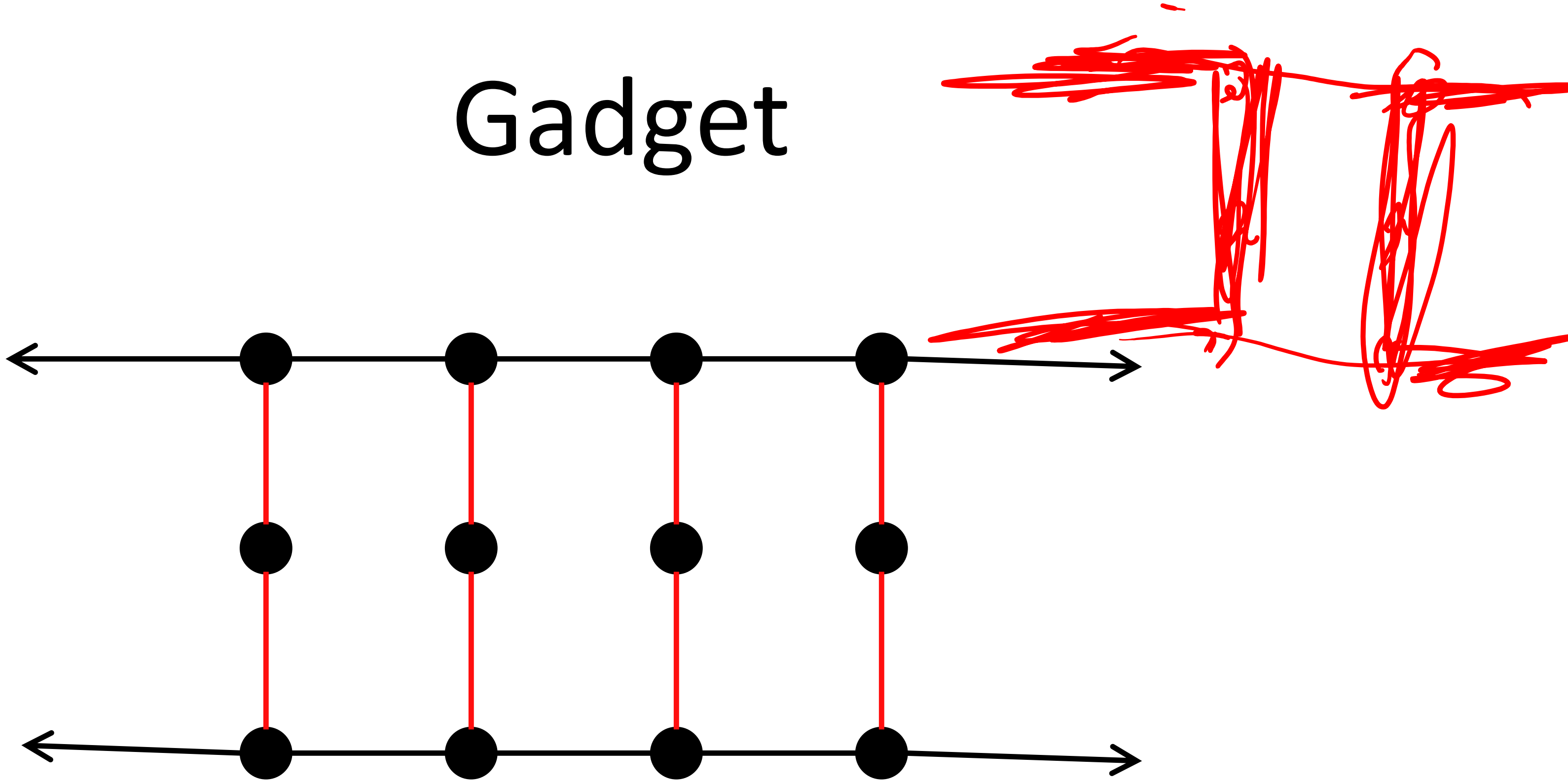


- Start with a cycle
- Double up some edges
- Cycle must pick one edge from each pair.
  - This provides a nice set of binary variables
- Need a way to add restrictions so that we can’t just use <u>any</u> choices.

Gadget

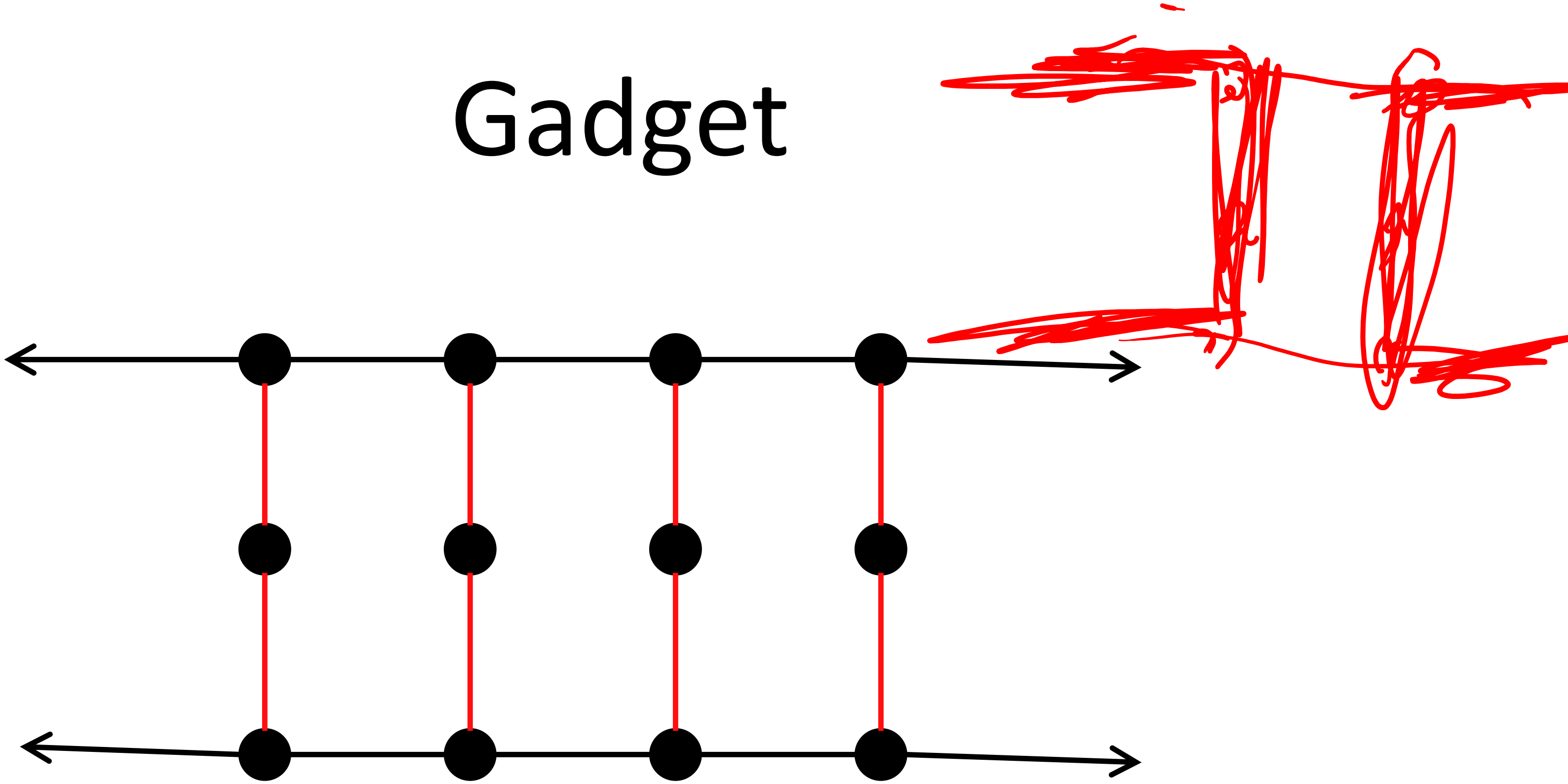


Gadget



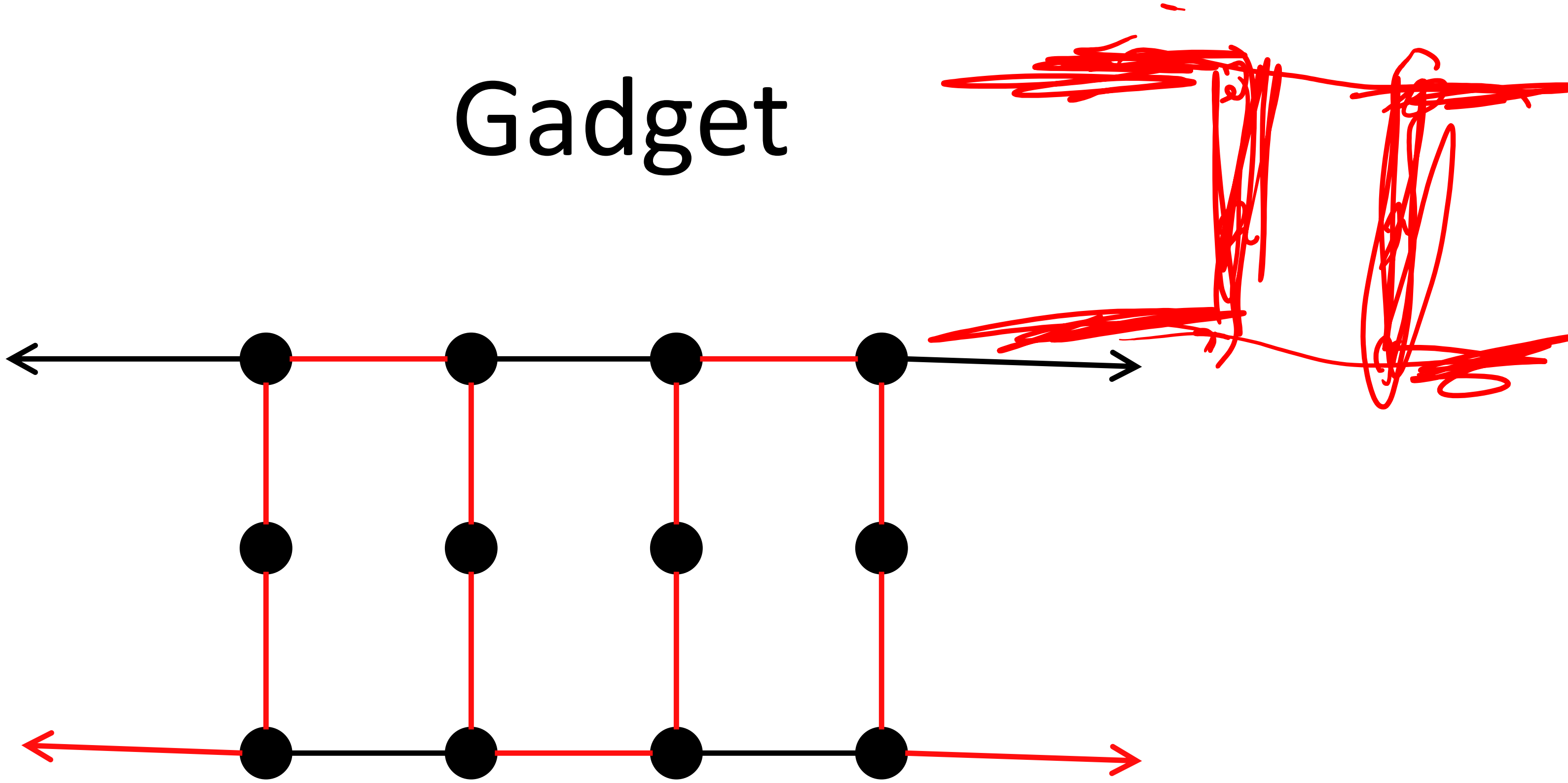
- Must use these edges.

Gadget



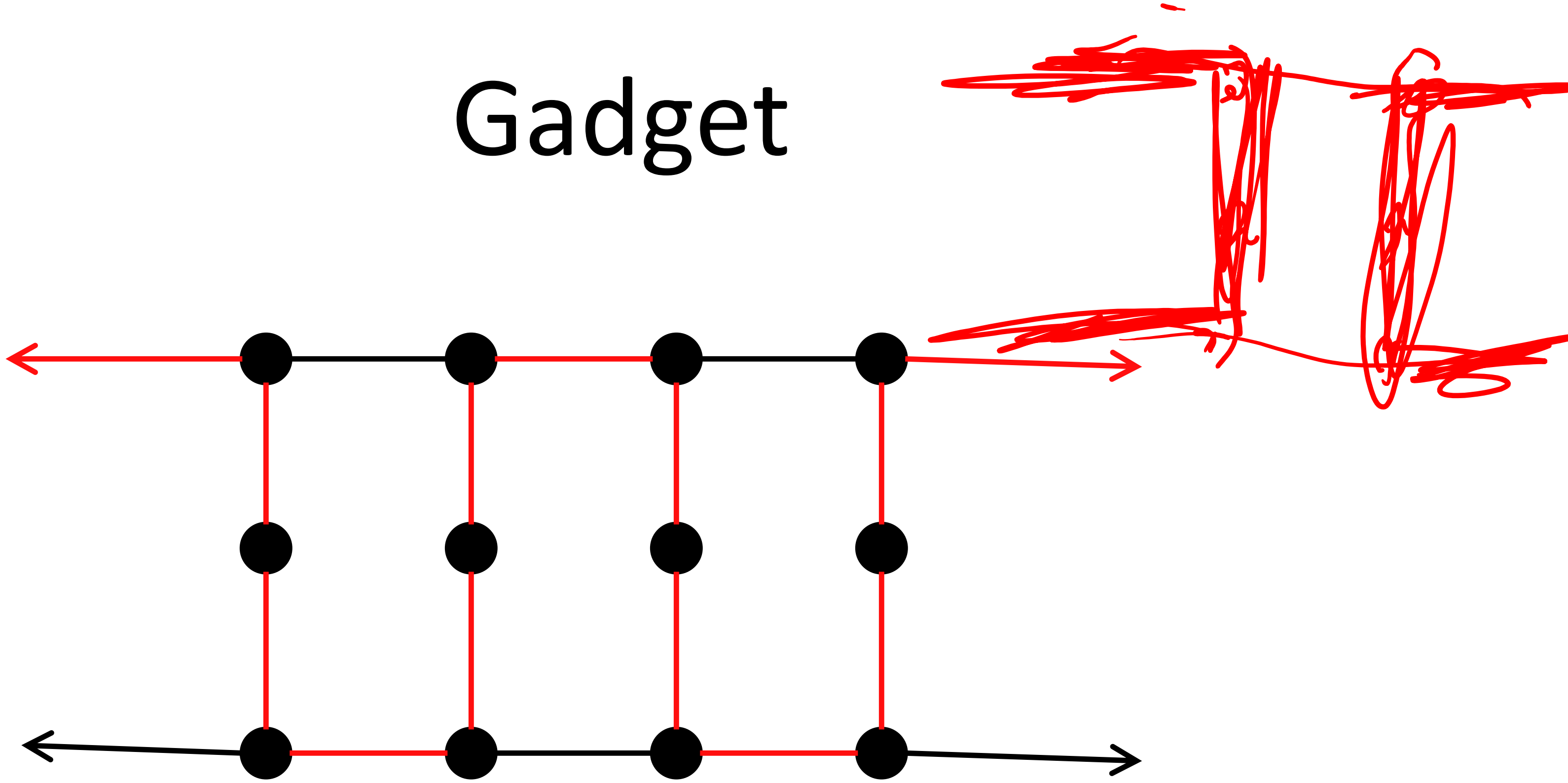
- Must use these edges.
- Two ways to fill out.

Gadget



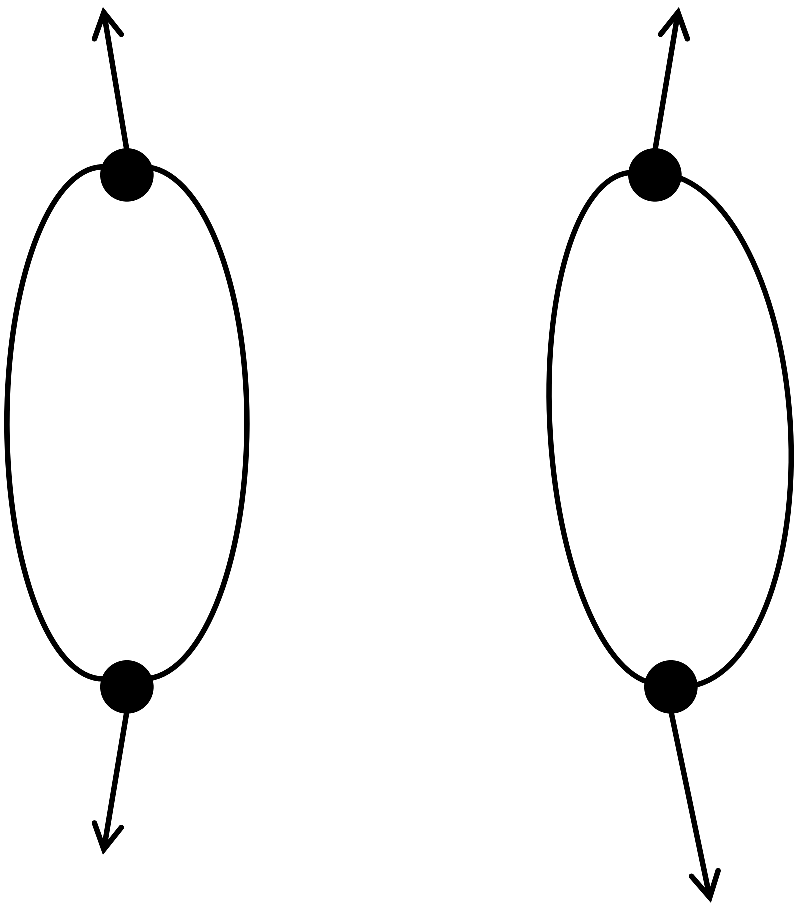
- Must use these edges.
- Two ways to fill out.

Gadget

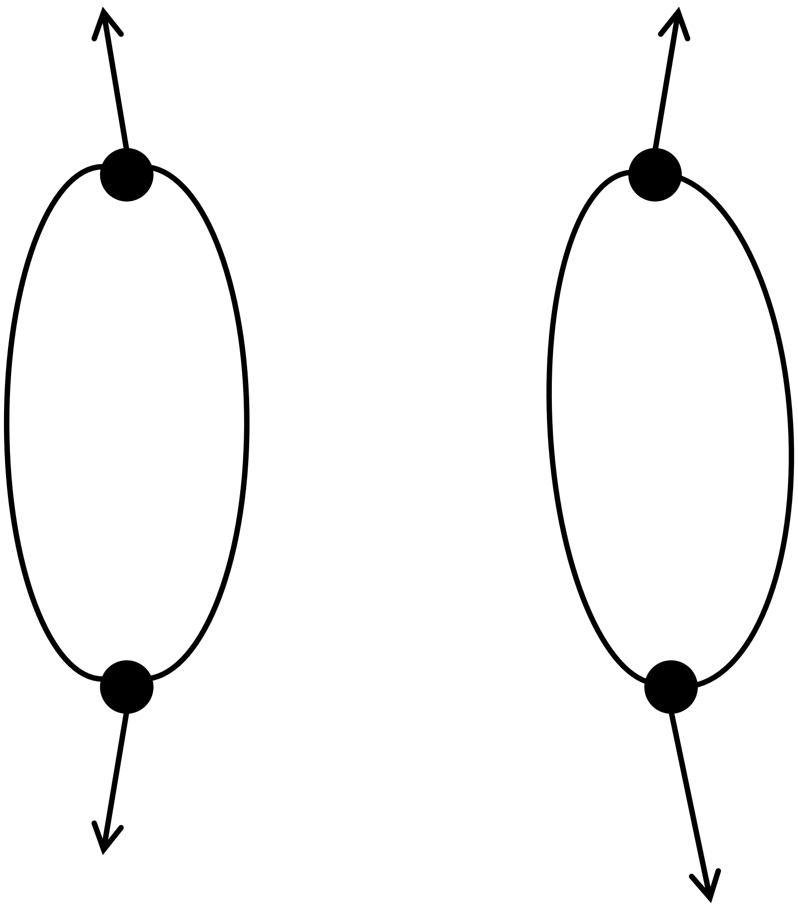


- Must use these edges.
- Two ways to fill out.

Gadget Use

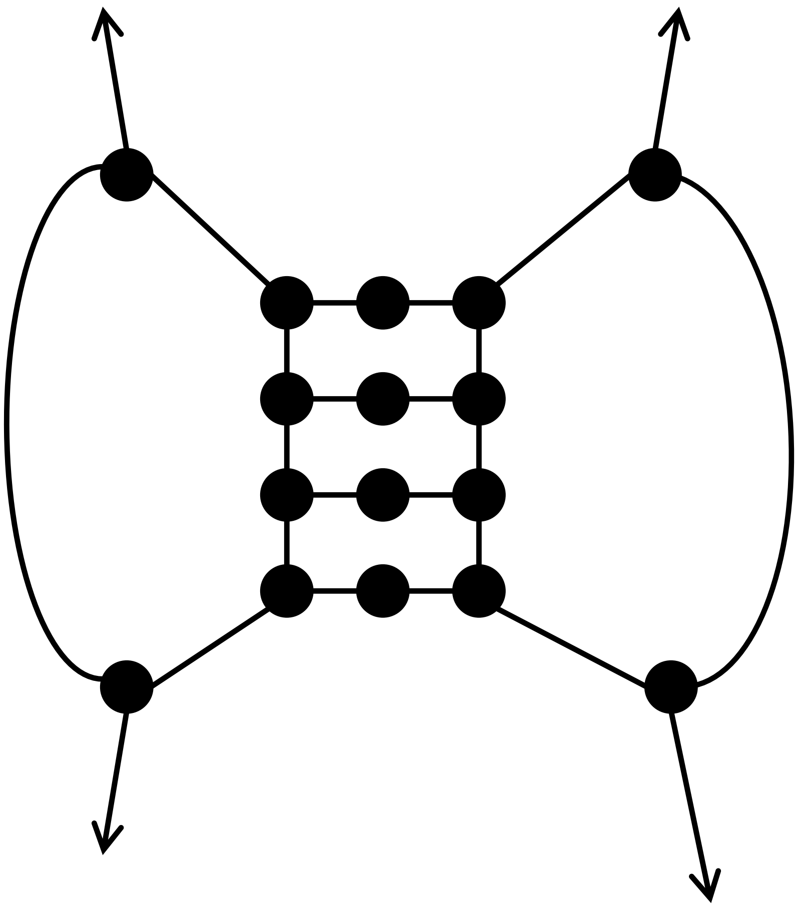


Gadget Use



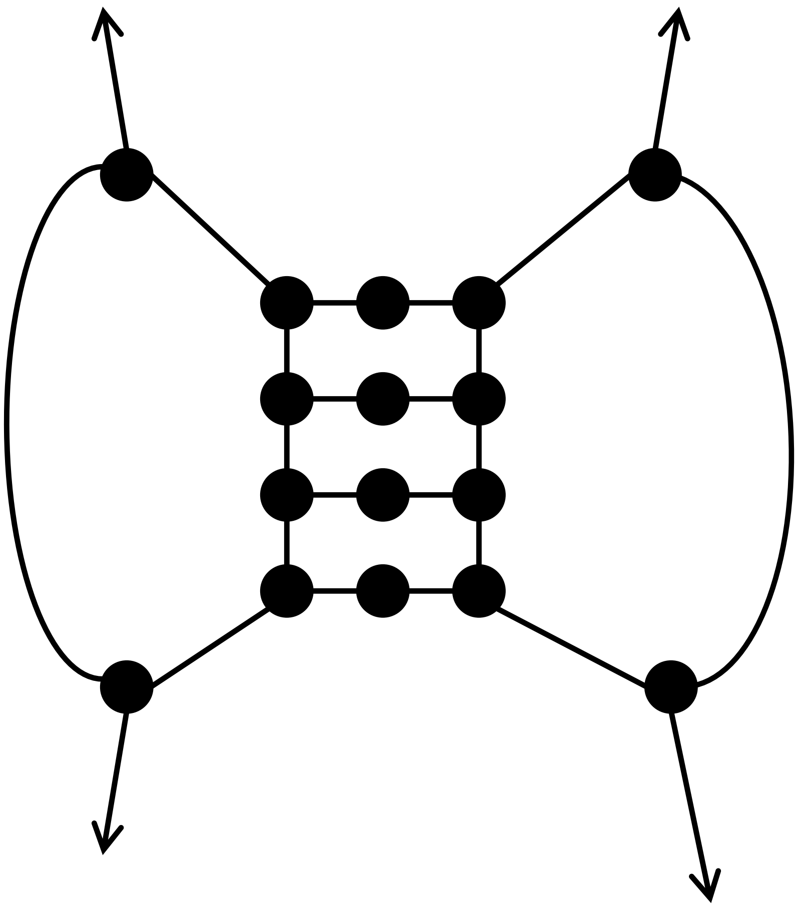
- Hook gadget up between a pair of edges.

Gadget Use



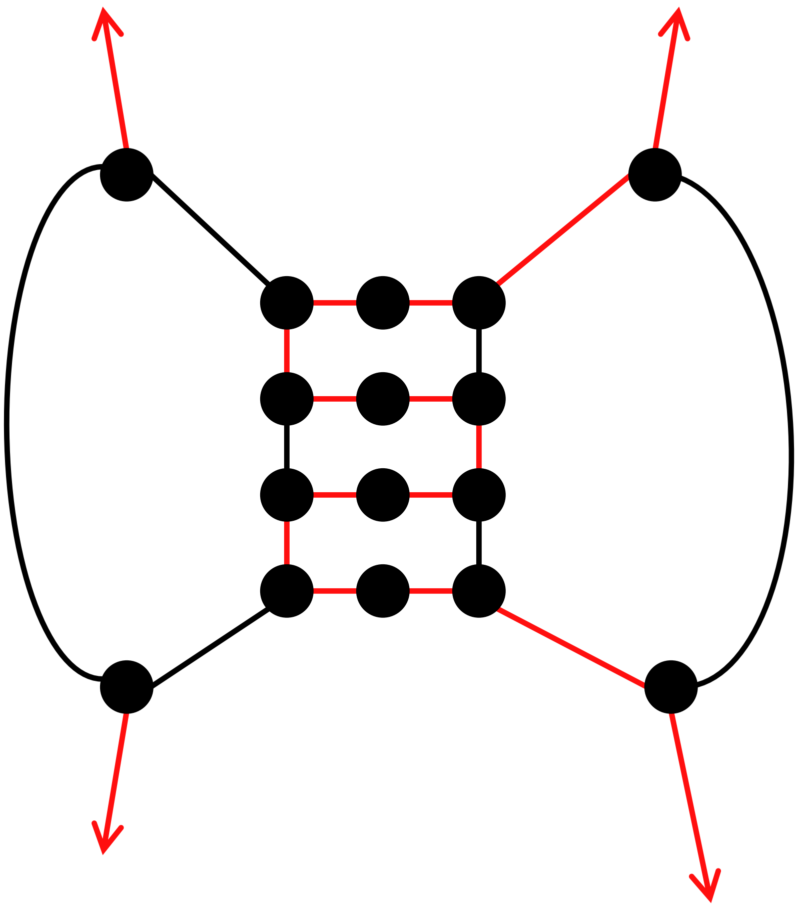
- Hook gadget up between a pair of edges.

Gadget Use



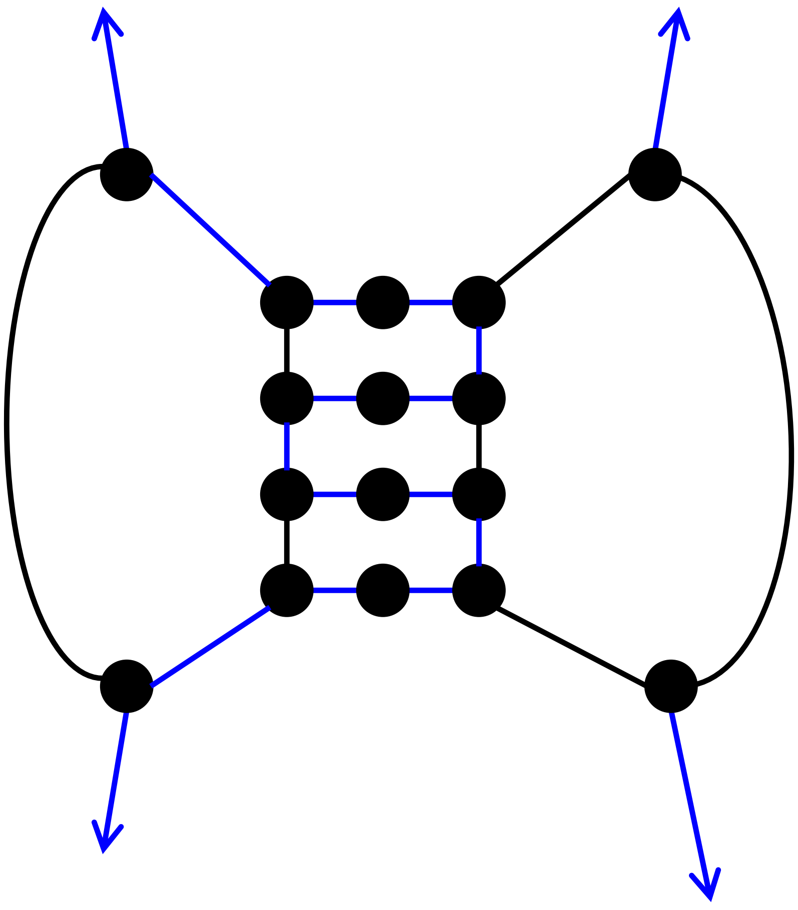
- Hook gadget up between a pair of edges.
- Hamiltonian Cycle must use exactly one of the connected edges.

Gadget Use



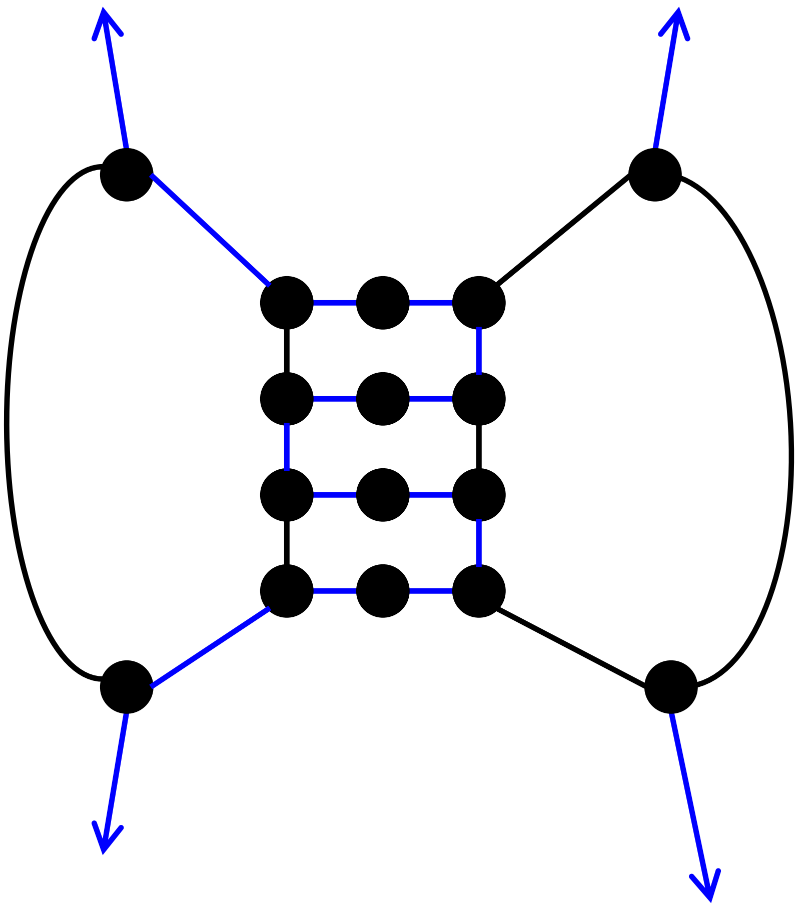
- Hook gadget up between a pair of edges.
- Hamiltonian Cycle must use exactly one of the connected edges.

Gadget Use



- Hook gadget up between a pair of edges.
- Hamiltonian Cycle must use exactly one of the connected edges.

Gadget Use



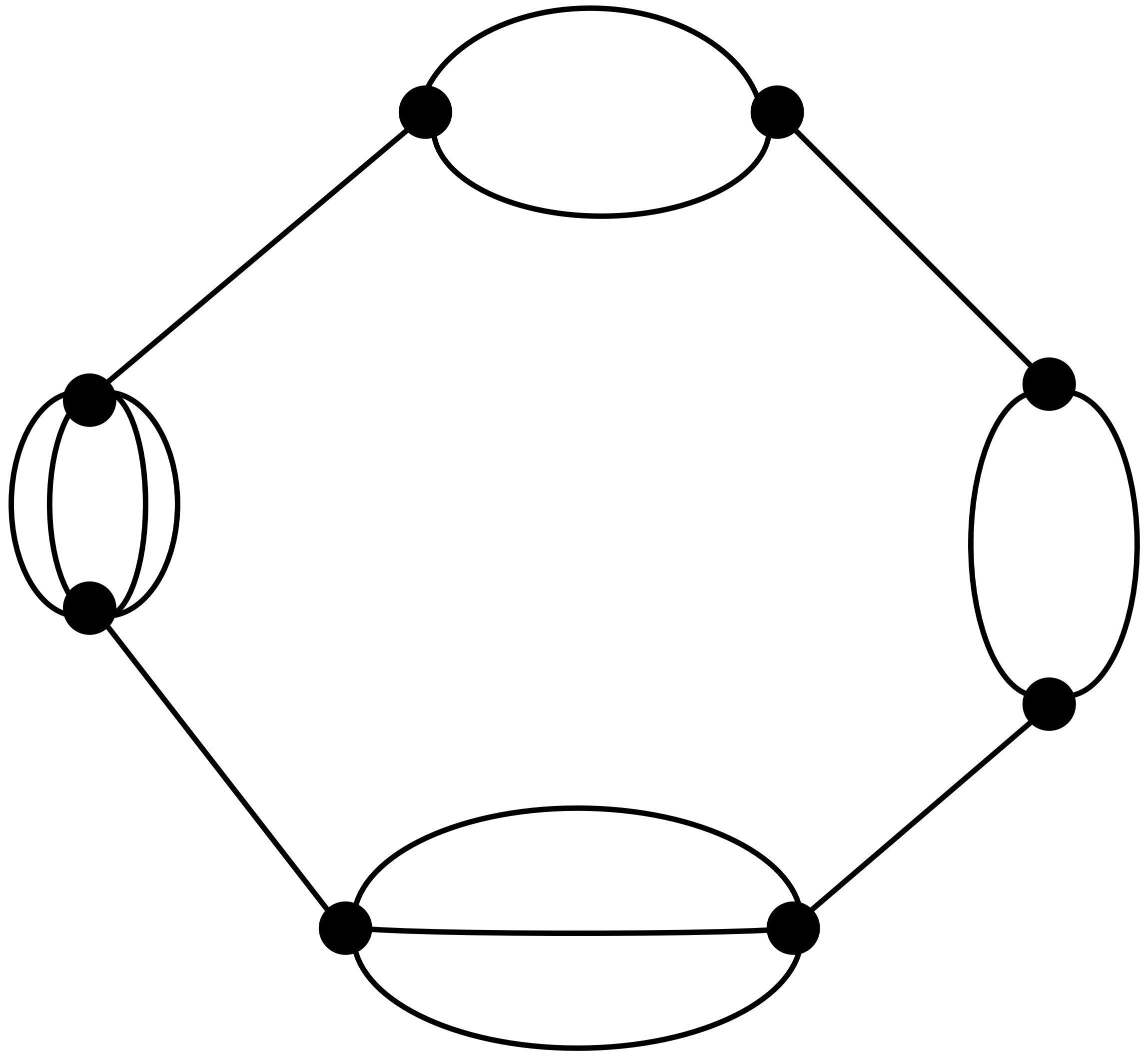
- Hook gadget up between a pair of edges.
- Hamiltonian Cycle must use exactly one of the connected edges.
- This allows us to force logic upon our choices.

Construction

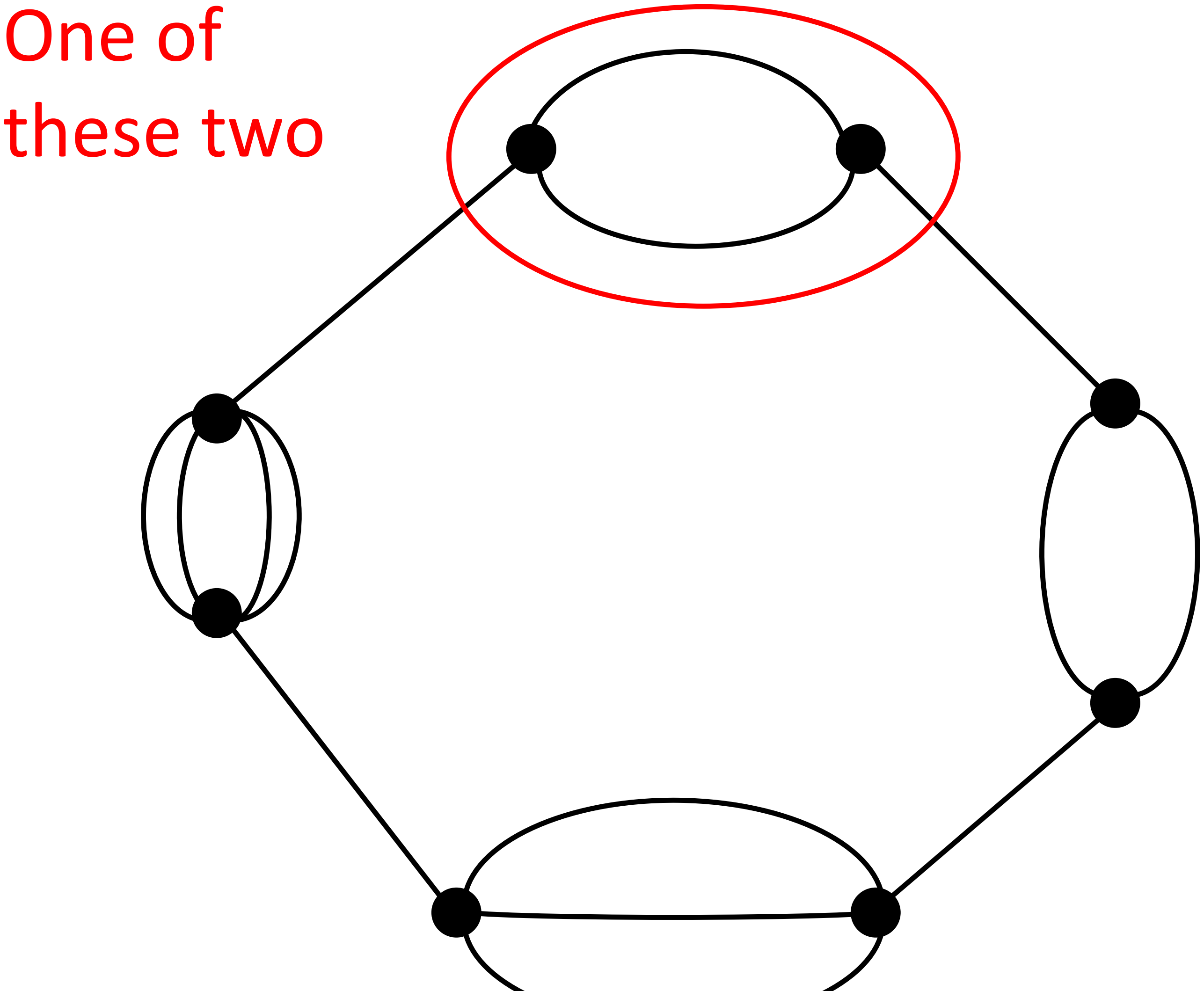
By doing this for several pairs of edges we can construct Hamiltonian Cycle problems equivalent to the following:

- You are given a number of choices where you need to pick one from several options (of multi-edges).
- You have several constraints, that say of two choices you must have picked exactly one of them.

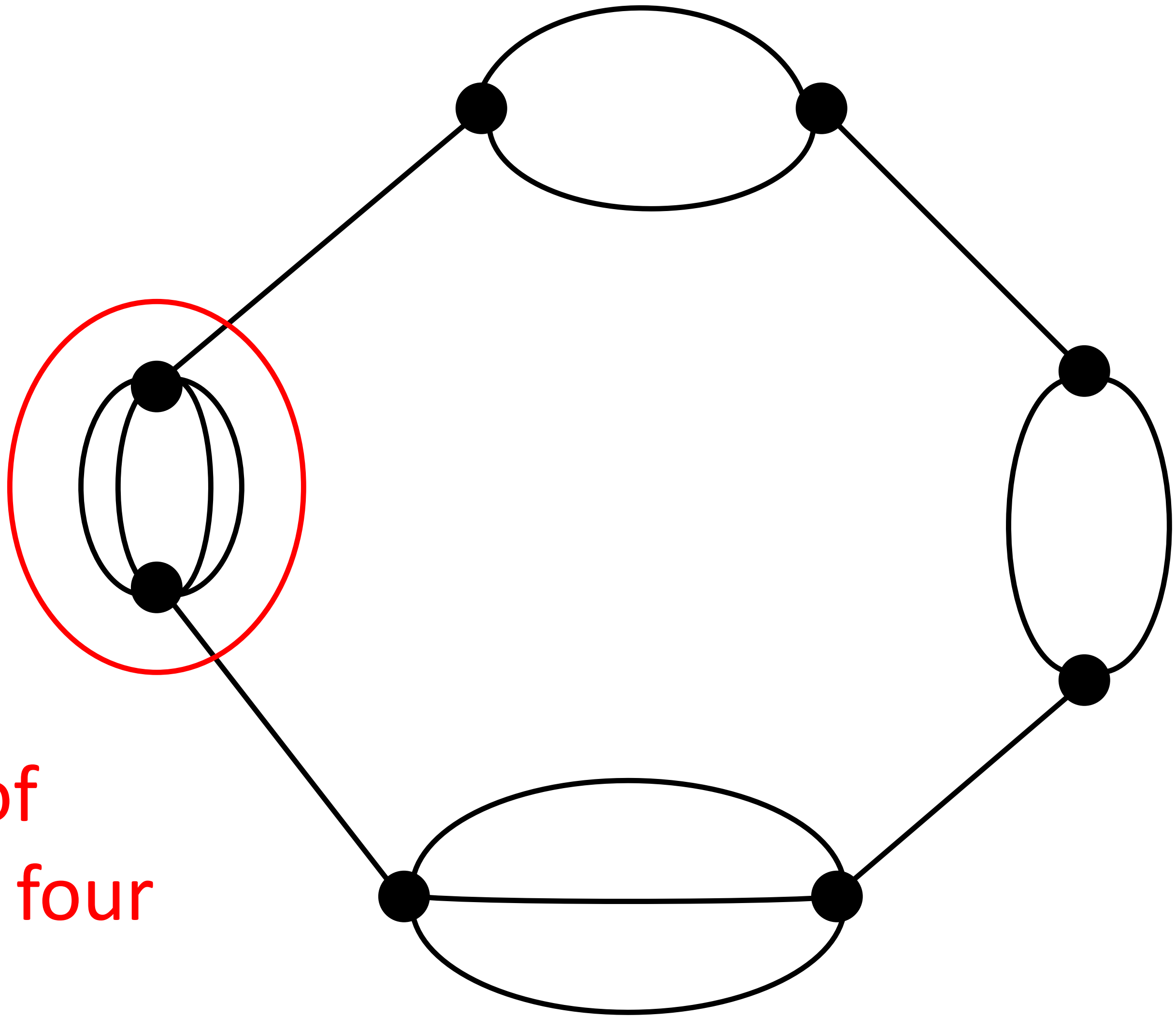
Example



Example

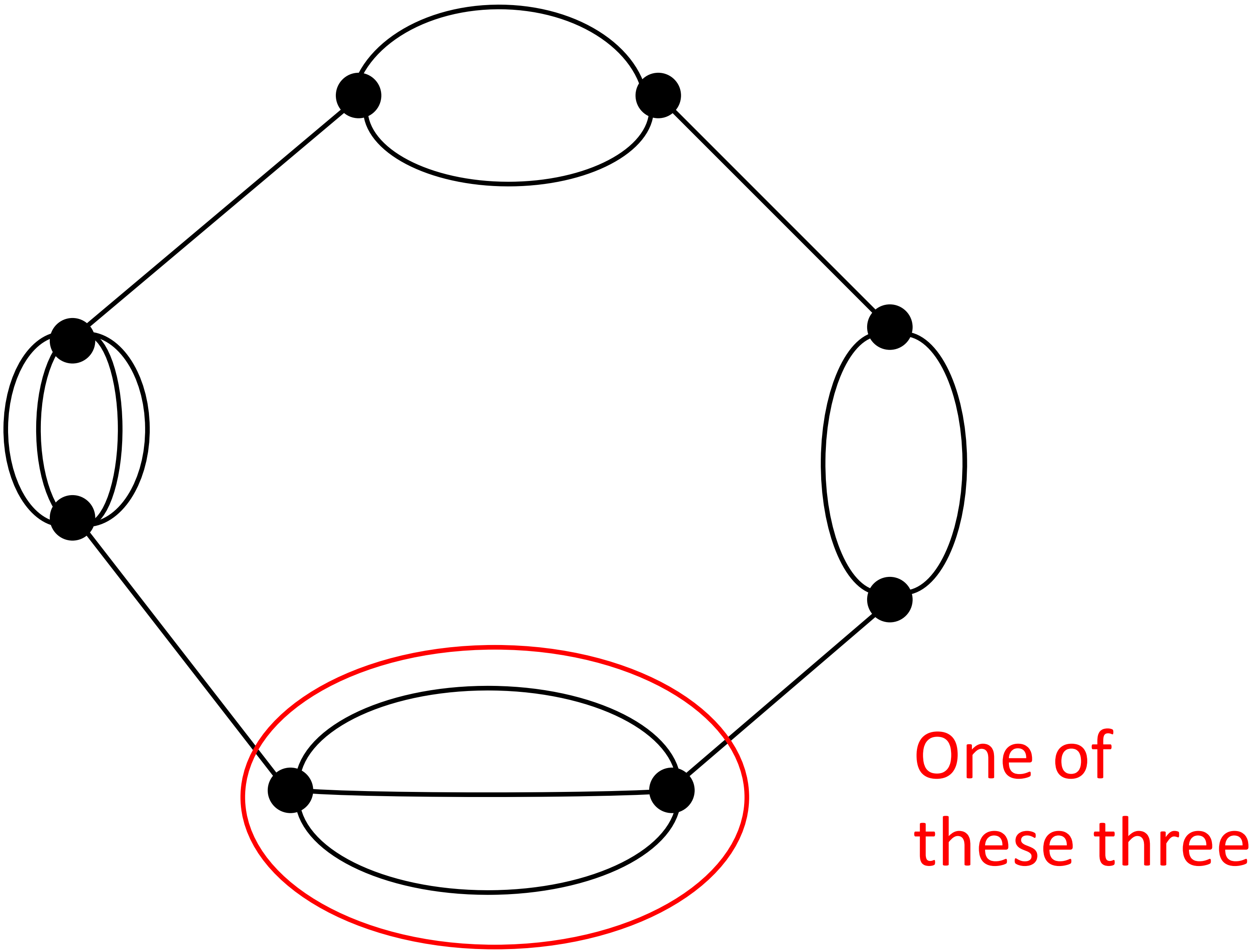


Example

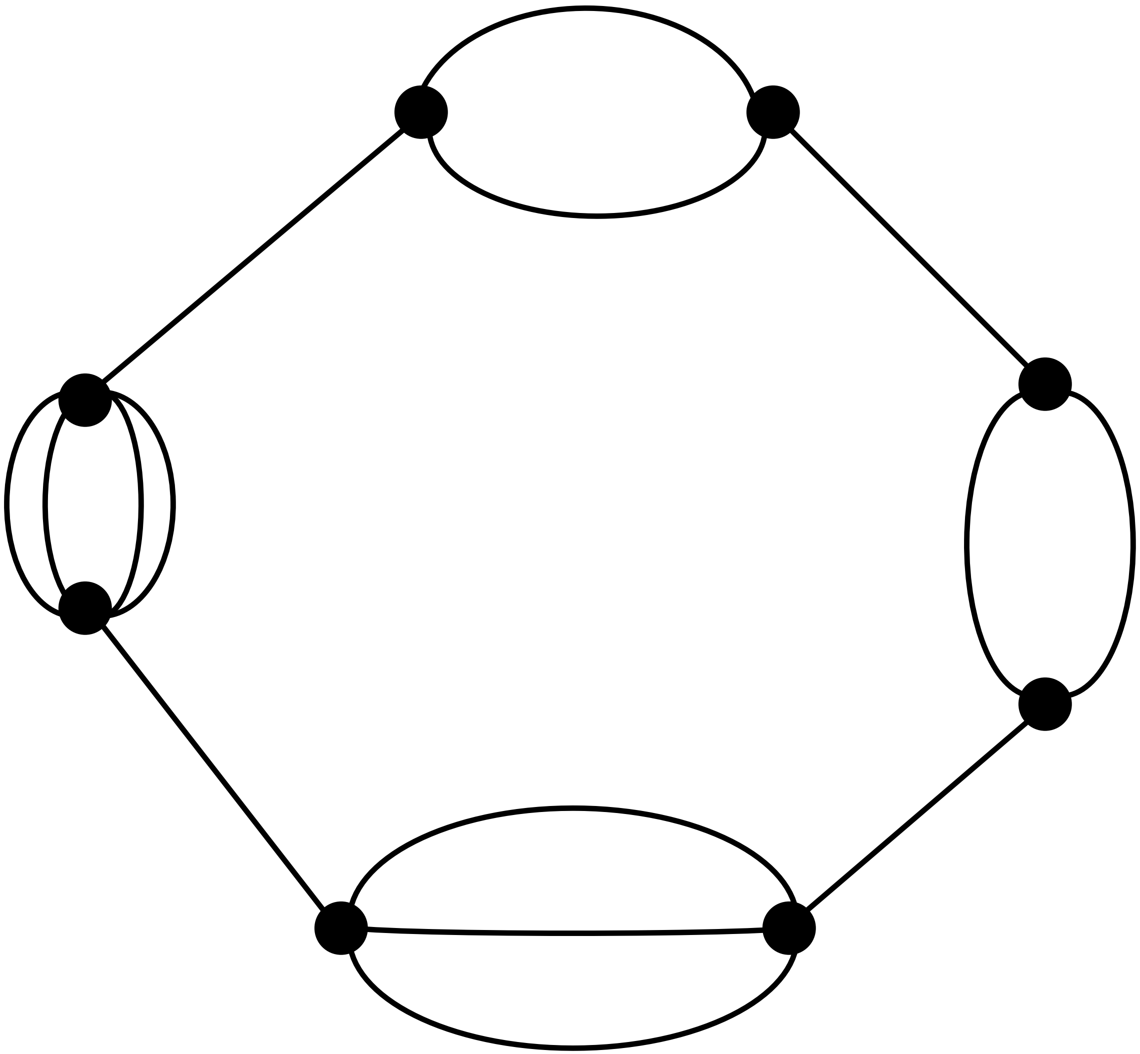


One of
these four

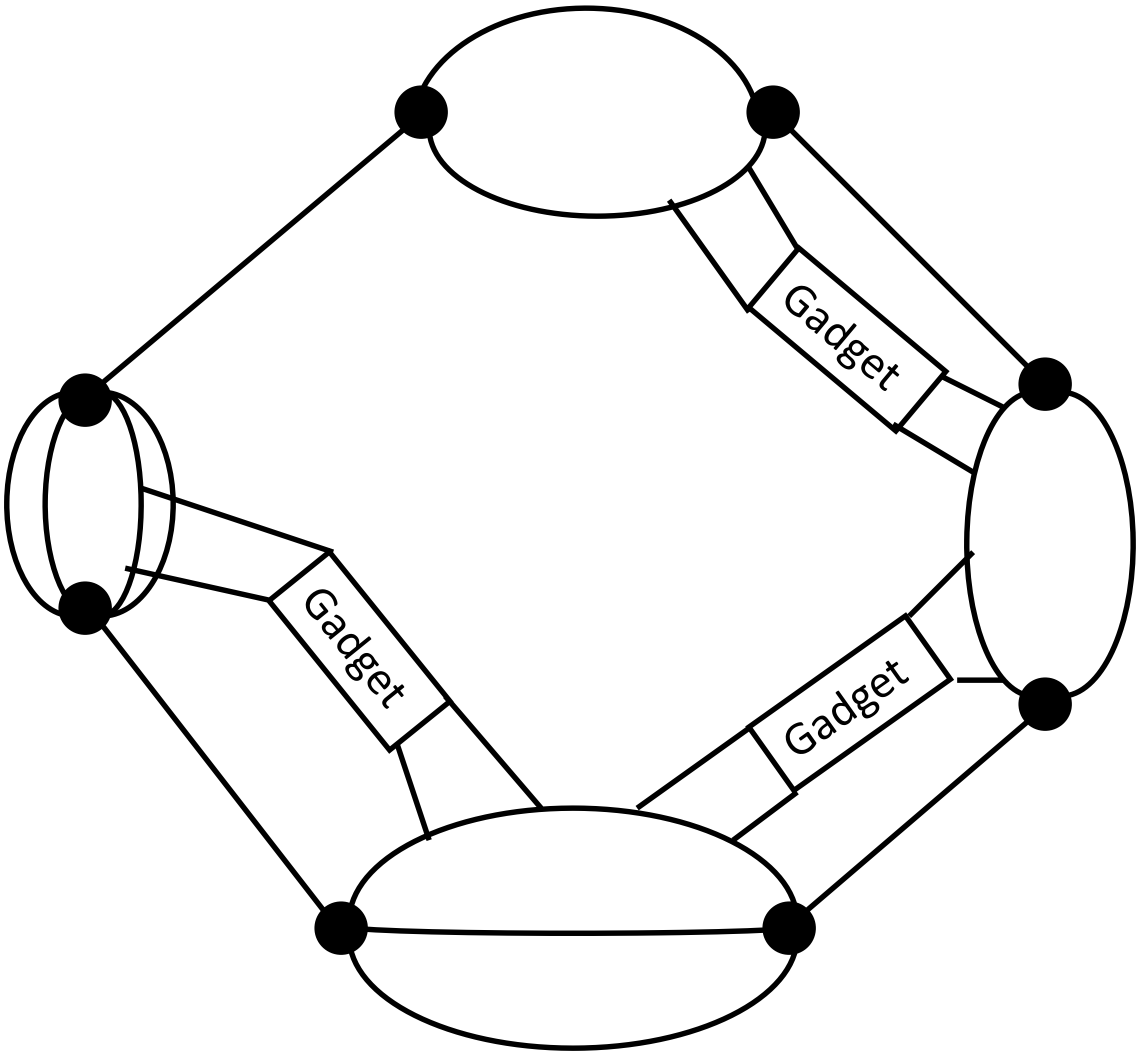
Example



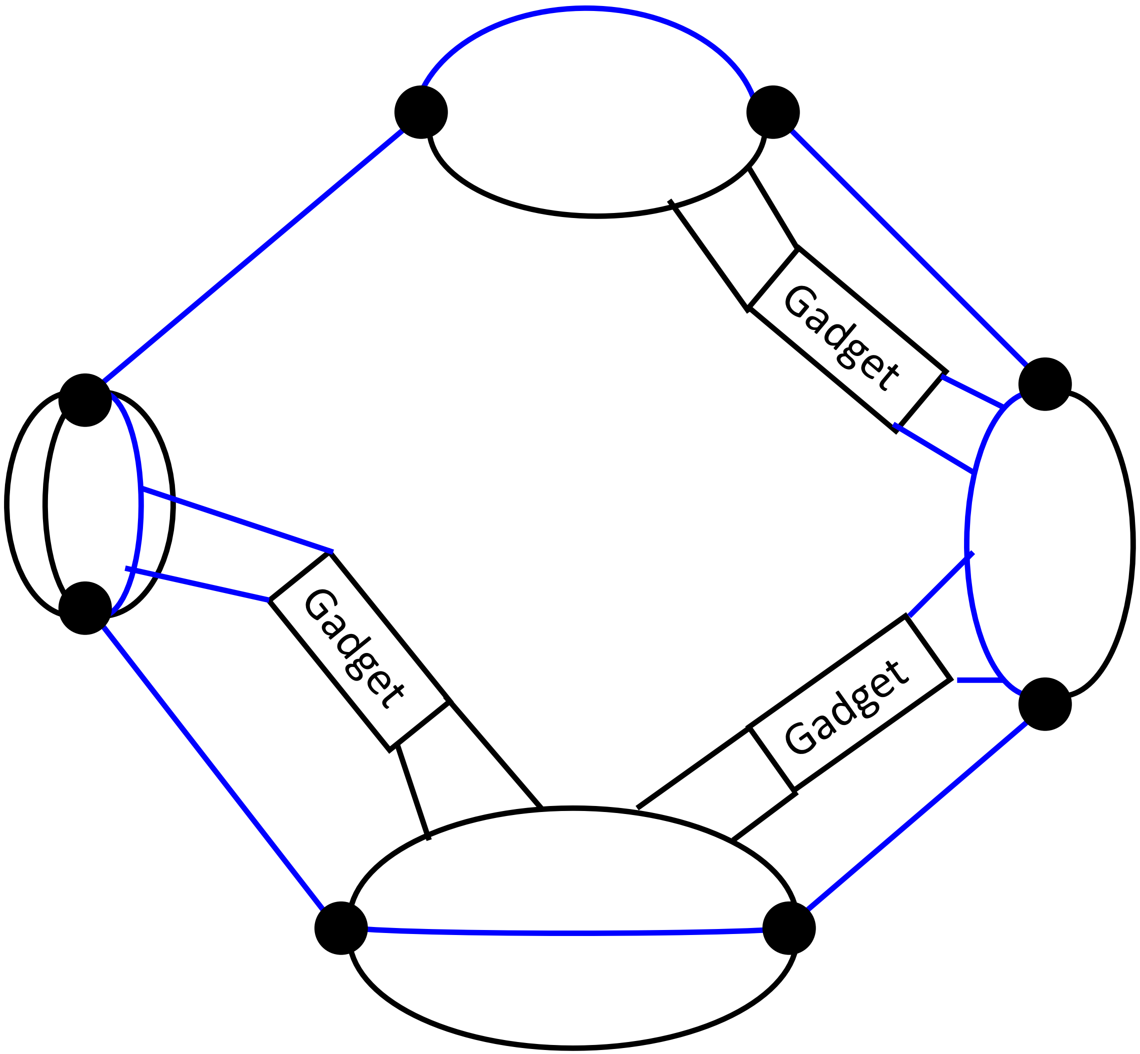
Example



Example



Example



Full Construction

Choices:

- For each variable, choose either 0 or 1.
- For each equation, choose one variable.

Full Construction

Choices:

- For each variable, choose either 0 or 1.
- For each equation, choose one variable.

Constraints:

- For each variable that appears in an equation, exactly one of the following should be selected:
  - That variable in that equation
  - That variable equal to 0

Example

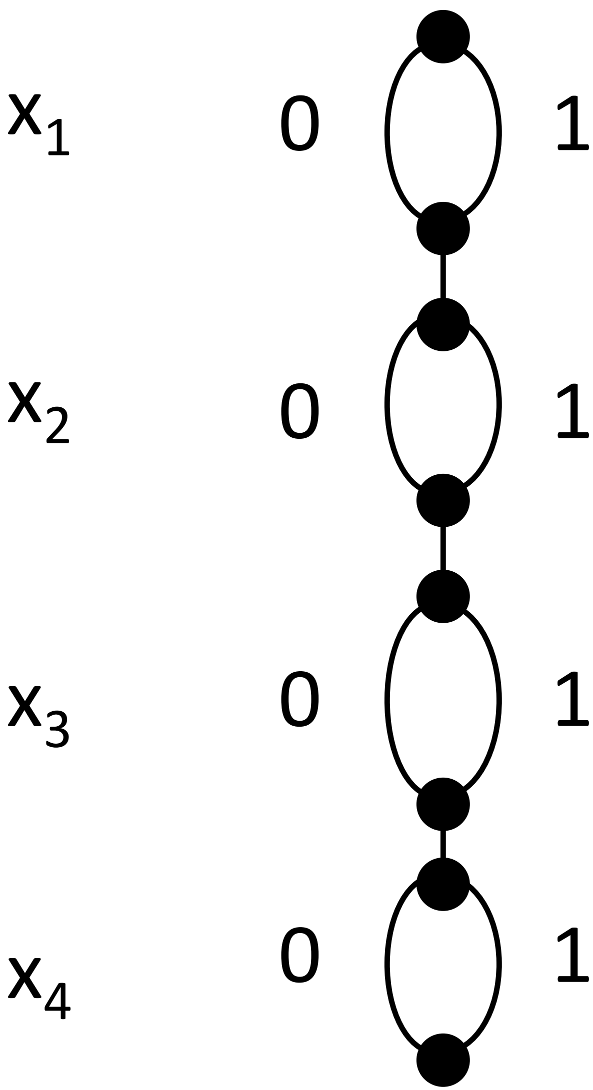
$x1+x2+x3=1$

$x2+x4=1$

Example

x1+x2+x3=1

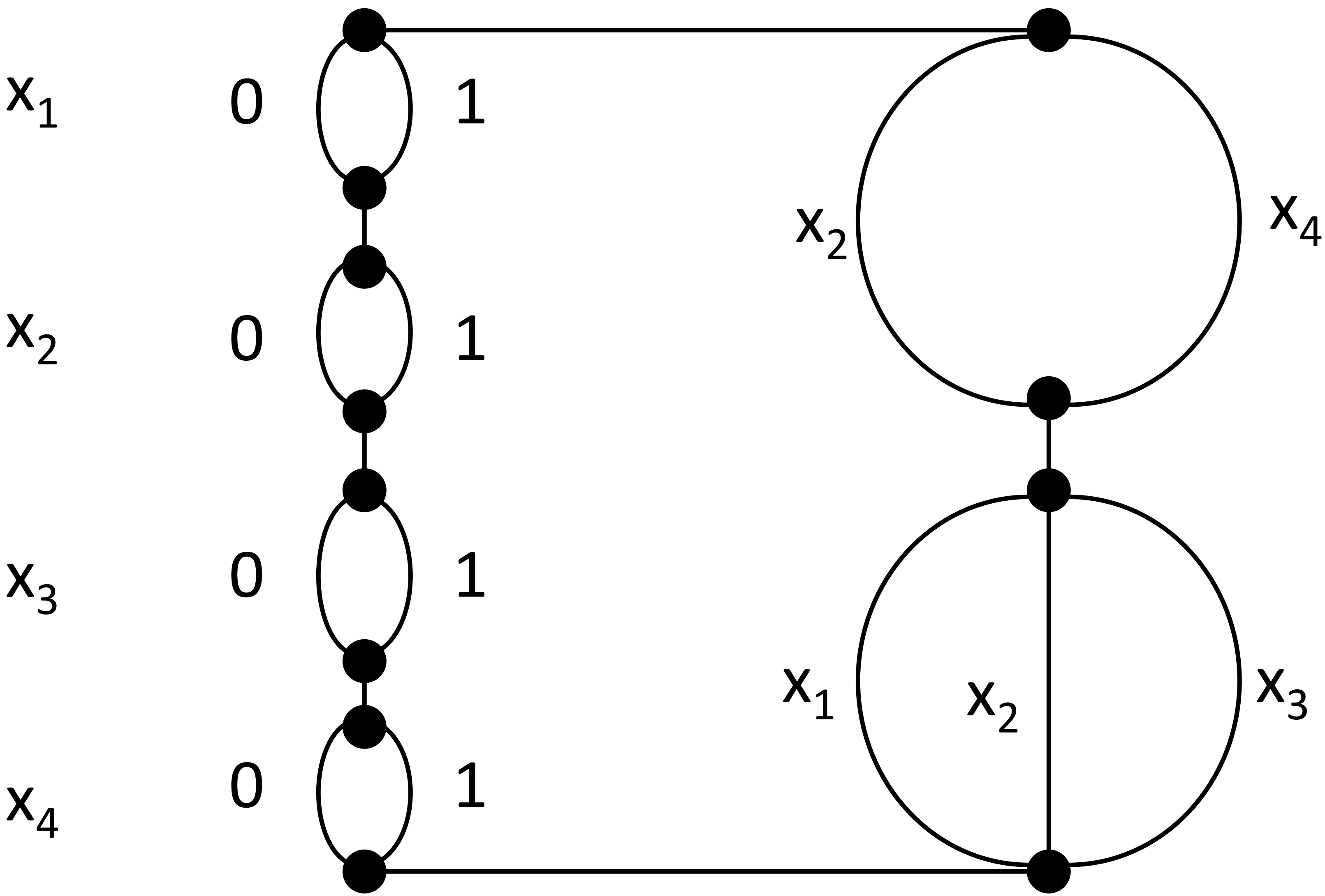
x2+x4=1



Example

$x1+x2+x3=1$

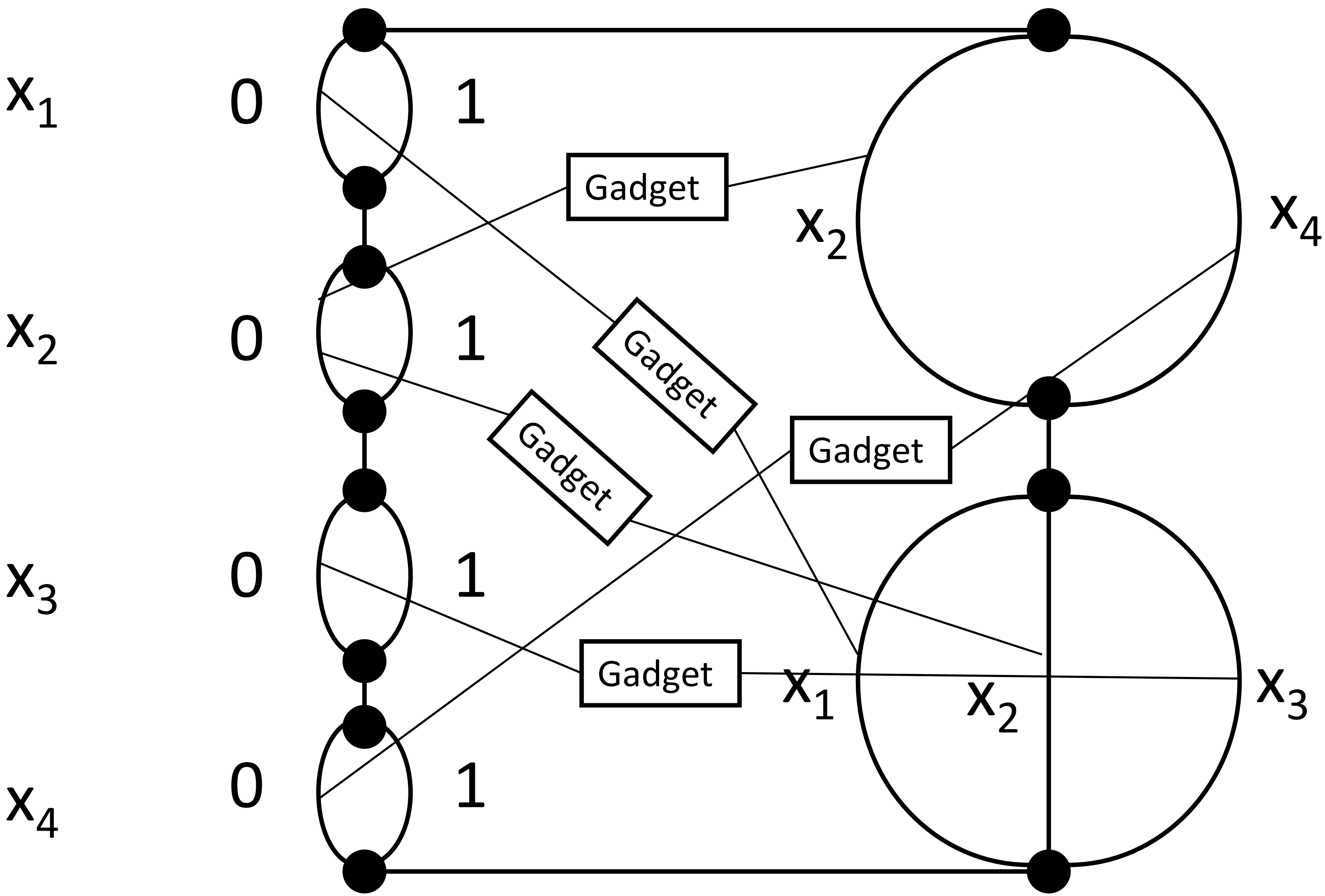
$x2+x4=1$



Example

$x1+x2+x3=1$

$x2+x4=1$

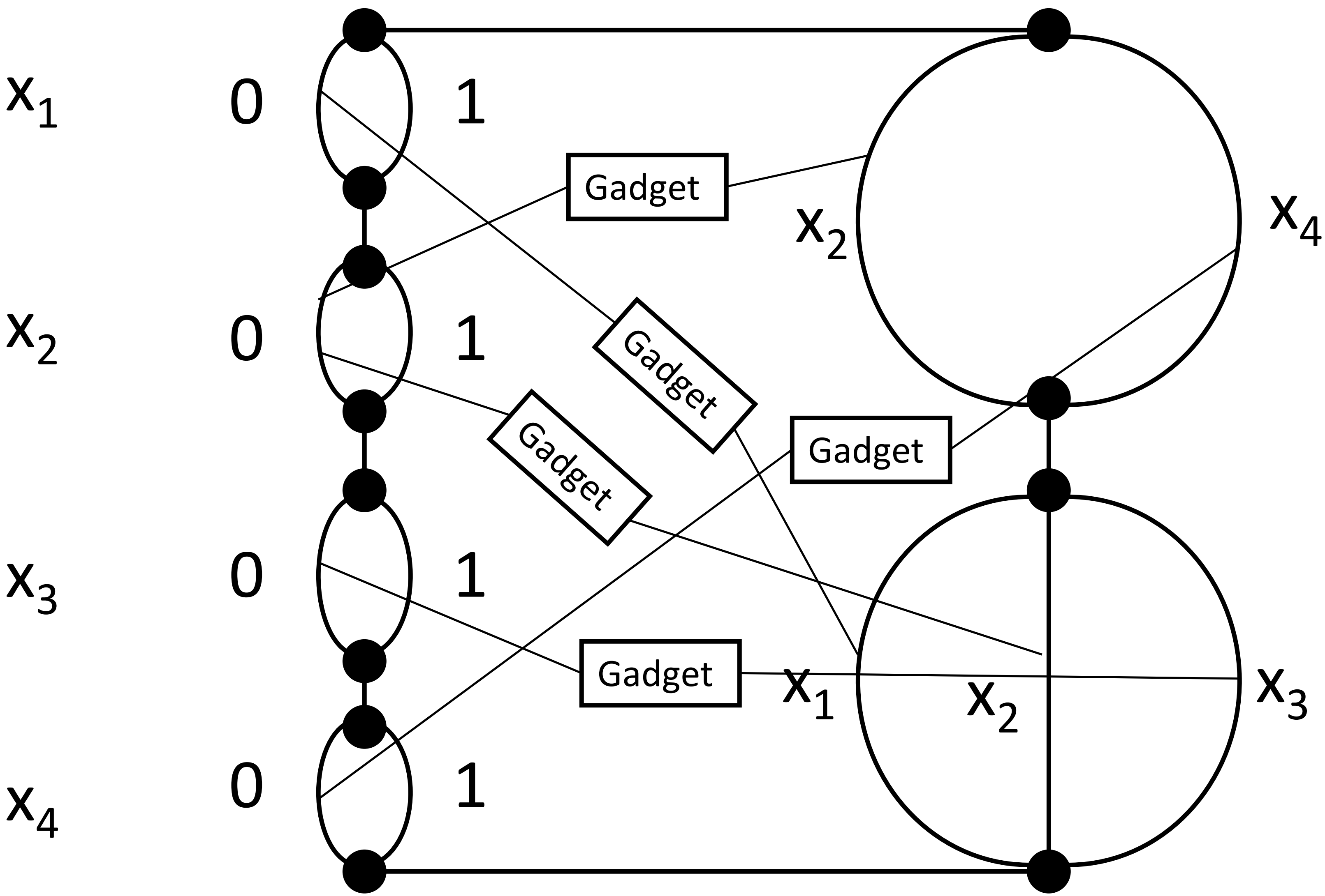


$x_1=1\ x_2=0\ x_3=0\ x_4=1$

Example

x1+x2+x3=1

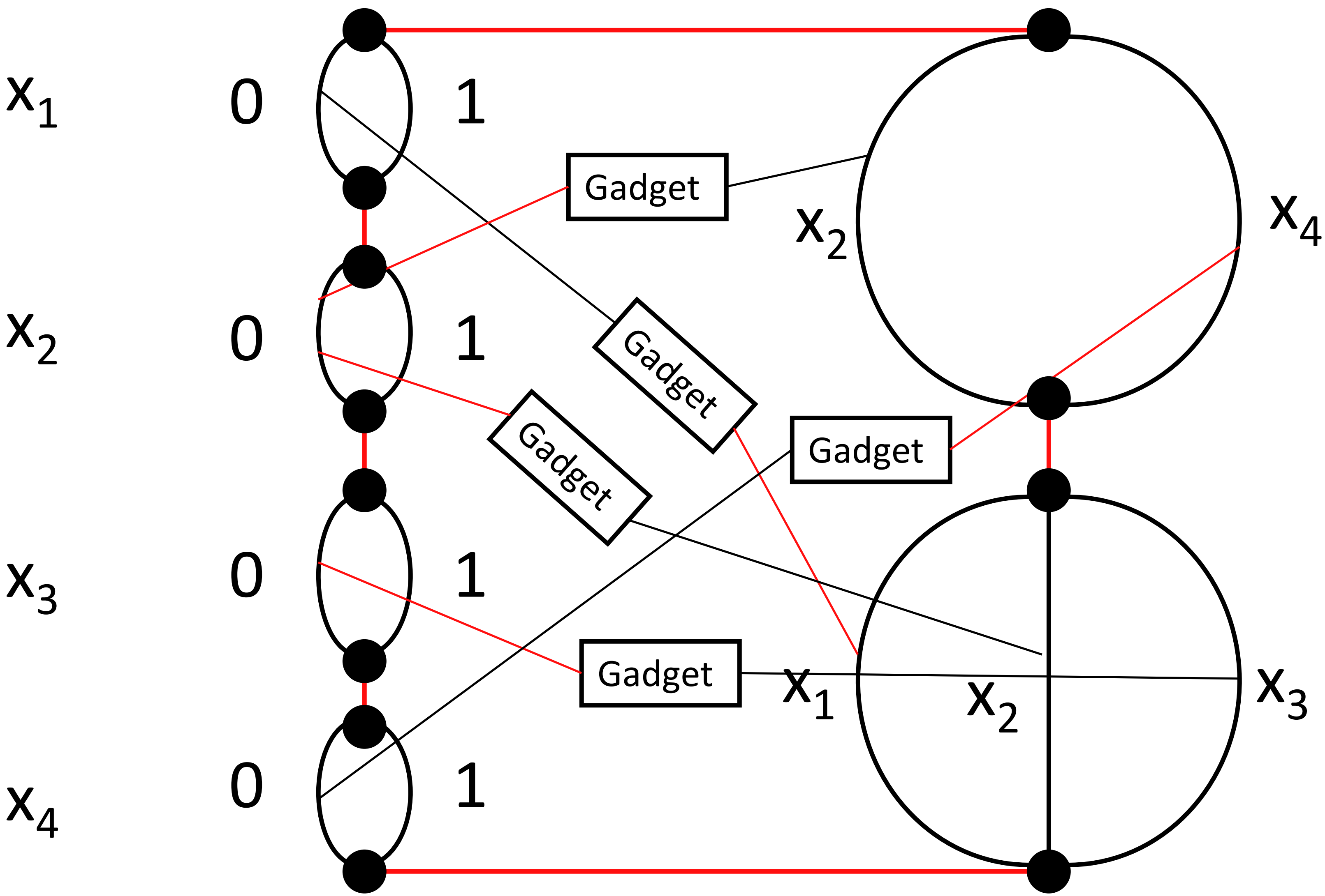
x2+x4=1



Example

x1+x2+x3=1

x2+x4=1



Analysis

If the ZOE has a solution, the Hamiltonian Cycle problem has a solution:

- Select the values of each variable and the variables equal to 1 in each equation.
- One for each and no conflicts.

Analysis

If the ZOE has a solution, the Hamiltonian Cycle problem has a solution:

- Select the values of each variable and the variables equal to 1 in each equation.
- One for each and no conflicts.

If the Hamiltonian Cycle has a solution, so does the ZOE:

- Set variables equal to the values selected in the cycle.
- Gadgets mean that you can select a variable from an equation if and only if that variable is 1.
- Must have exactly one variable from each equation equal 1.
- Solution to ZOE.

Reduction Summary