CSC 384
Introduction to Artificial Intelligence

CSP 2

Alice Gao and Randy Hickey

Winter 2023

Learning Goals

By the end of this lecture, you should be able to

Arc Consistency

- Determine whether a binary constraint is arc-consistent.
- Explain why the arc consistency of a binary constraint is not symmetric.

Forward Checking

- Explain the idea of Forward Checking and
- Explain how to combine Backtracking Search and Forward Checking.
- Explain how Forward Checking can reduce the size of the search tree.
- Trace the execution of Backtracking Search and Forward Checking.

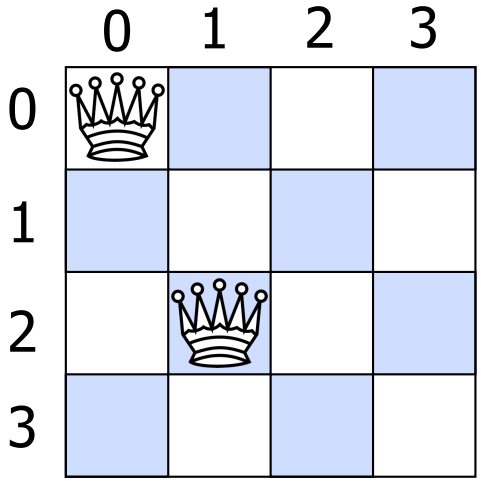
Outline

1. Arc Consistency
2. Backtracking Search with Forward Checking

ARC CONSISTENCY

Motivating Arc Consistency

- $x_0 = 0$ and $x_1 = 2$ do not lead to a solution. Why?



Notation for an Arc

- X and Y are two random variables.
- D_X and D_Y are their respective domains.
- $c(X, Y)$ is a binary constraint.

Arc Consistency Definition

$< X, c(X, Y) >$ is arc-consistent if and only if

for every value v in D_X,

there exists a value w in D_Y,

such that (v, w) satisfies the constraint $c(X, Y)$.

Question 1: Checking Arc Consistency

Assume that $D_X = \{1\}$ and $D_Y = \{1,2\}$.

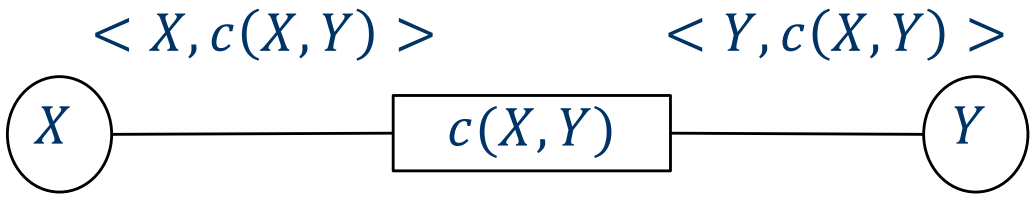
Consider the constraint $c(X, Y)$: $X < Y$.

Is the arc $< X, c(X, Y) >$ consistent?

A. Yes.

B. No.

C. I don't know.



Question 2: Checking Arc Consistency

Assume that $D_X = \{1\}$ and $D_Y = \{1,2\}$.

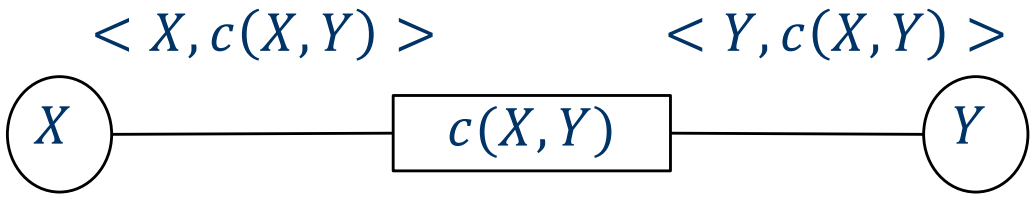
Consider the constraint $c(X, Y)$: $X < Y$.

Is the arc $< Y, c(X, Y) >$ consistent?

A. Yes.

B. No.

C. I don't know.



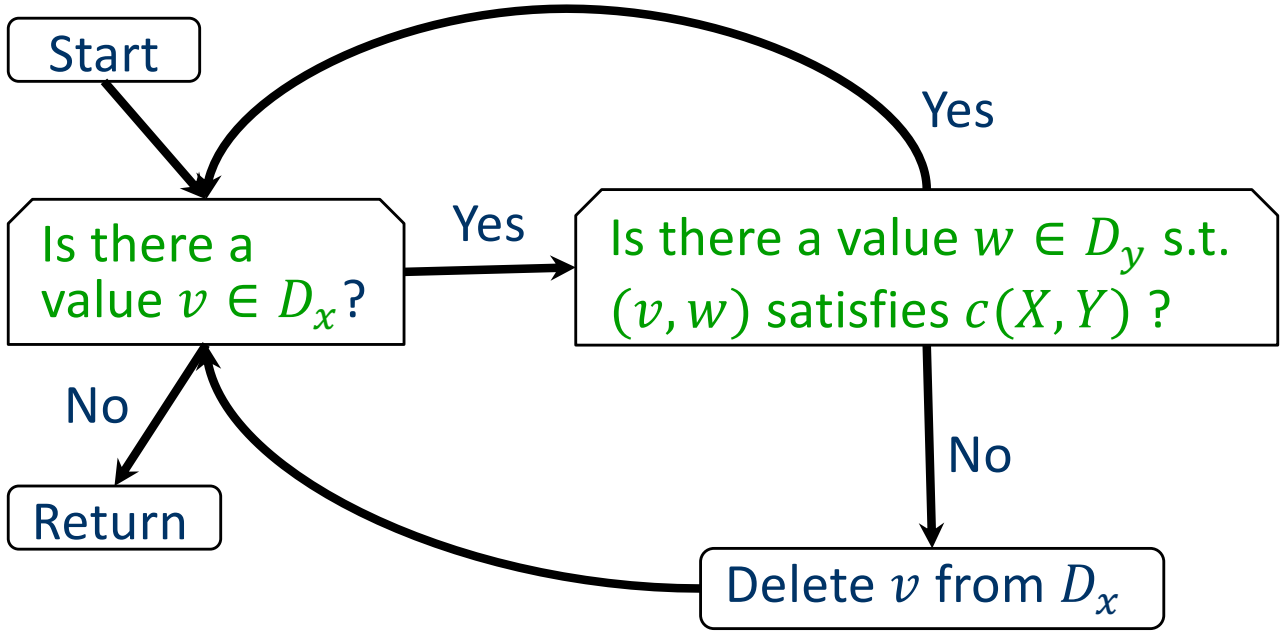
Arc Consistency is not Symmetric

$$< X, c(X,Y) > \; is \; consistent$$
$$\nLeftrightarrow$$
$$< Y, c(X,Y) > \; is \; consistent$$

Restore Arc Consistency

- Suppose that $< X, c(X, Y) >$ is not consistent.
- At least one value $v \in D_x$ is causing the inconsistency.
  - For every $w \in D_y$, (v, w) violates the constraint $c(X, Y)$.
- Restore arc consistency by removing every value in D_x causing the inconsistency.

Revise Domain to Restore Arc-Consistency




```
1. function REVISE(csp, X, Y)
2.     for each v in D_x do
3.         if no value w in D_y allows (v,w) to satisfy the
   constraint between X and Y then
4.             delete v from D_x
```

BACKTRACKING SEARCH WITH FORWARD CHECKING

Forward Checking (for binary constraints)

After assigning variable X to a value

For each unassigned variable Y connected to X
by a constraint $c(X, Y)$

make <$Y, c(X, Y)$> arc-consistent.
(remove any value w in D_y violating the constraint c.)

Q1: Forward Checking for Binary Constraints

After assigning $X = 0$,

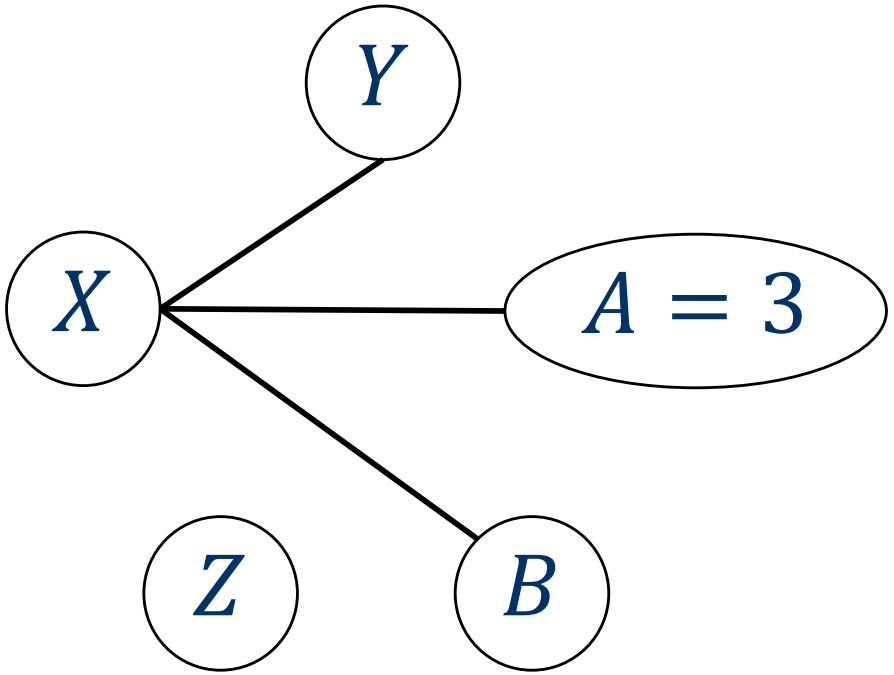
which arcs do we need to check for consistency?

(A) $< \mathrm{Y}, \mathrm{c}(\mathrm{X}, \mathrm{Y}) >$

(B) $< \mathrm{Z}, \mathrm{c}(\mathrm{X}, \mathrm{Z}) >$

(C) $< \mathrm{A}, \mathrm{c}(\mathrm{X}, \mathrm{A}) >$

(D) $< \mathrm{B}, \mathrm{c}(\mathrm{X}, \mathrm{B}) >$



Forward Checking (for all constraints)

After assigning variable X to a value

For every constraint c involving X

If c has exactly one unassigned variable Y in its scope

make <Y, c> arc-consistent.

(remove any value w in D_y violating the constraint c.)

Q2: Forward Checking for All Constraints

After assigning $X = 0$,

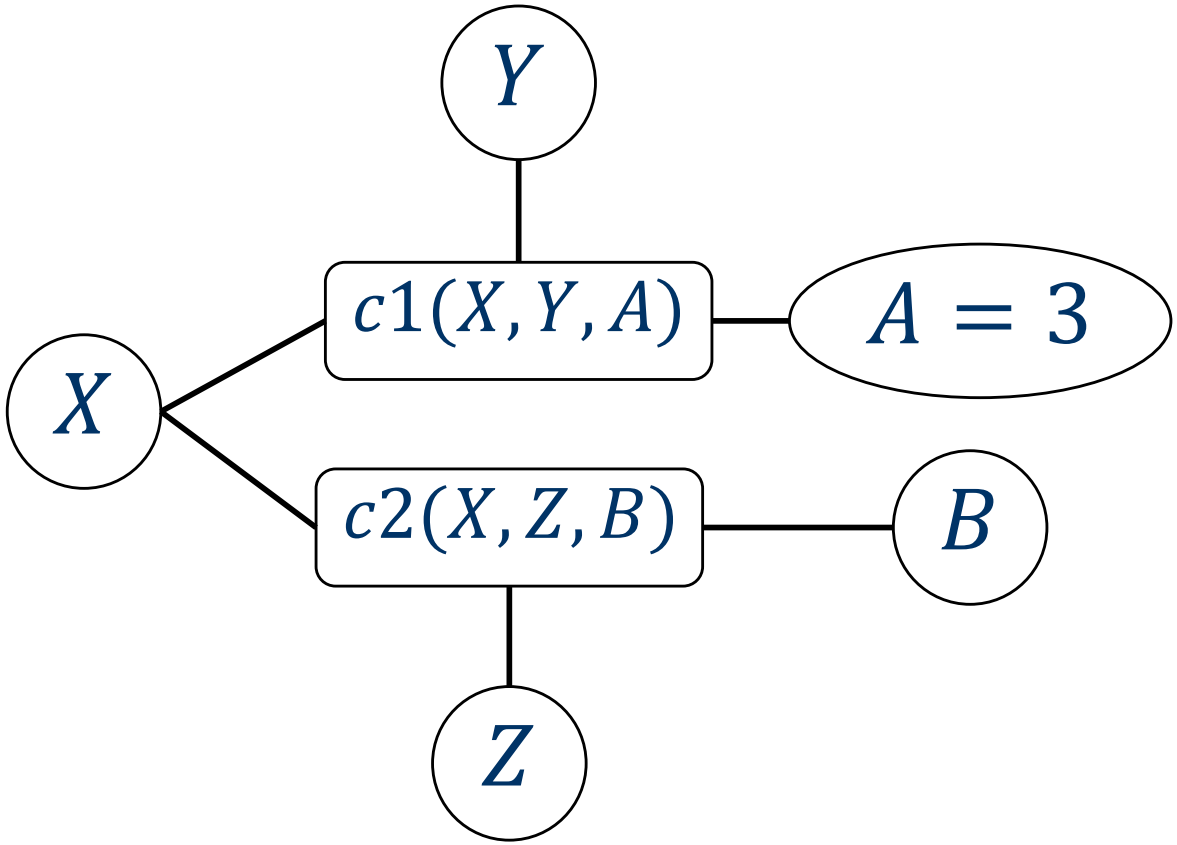
which arcs do we need to check for consistency?

(A) $< \mathrm{Y}, \mathrm{c1} >$

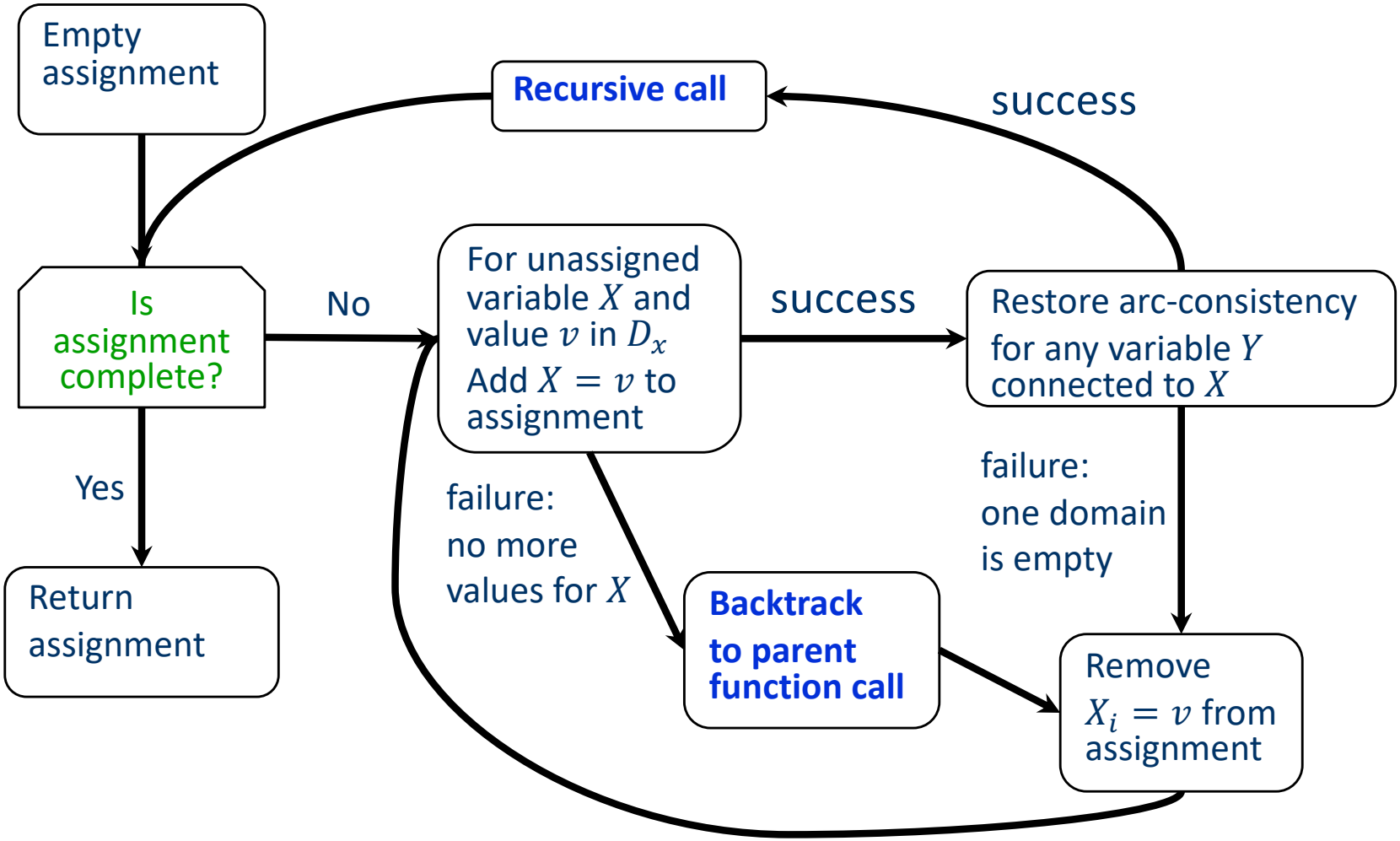
(B) $< \mathrm{A}, \mathrm{c1} >$

(C) $< \mathrm{Z}, \mathrm{c2} >$

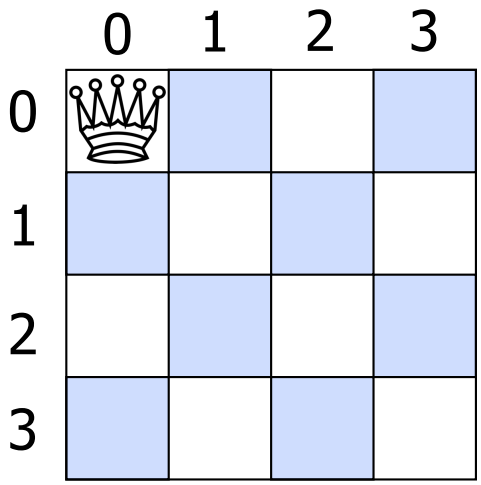
(D) $< \mathrm{B}, \mathrm{c2} >$

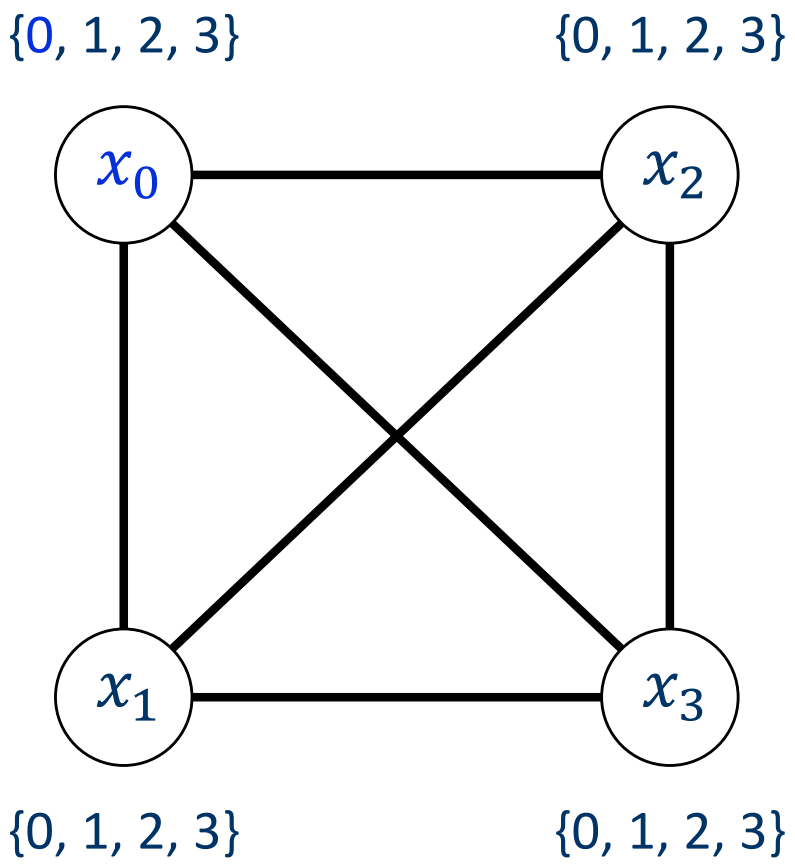


Backtracking Search w/ Forward Checking

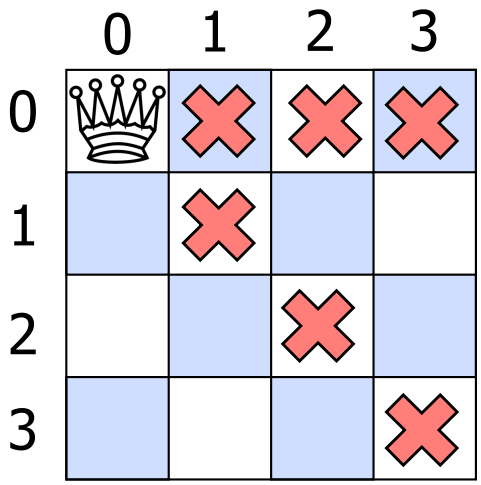


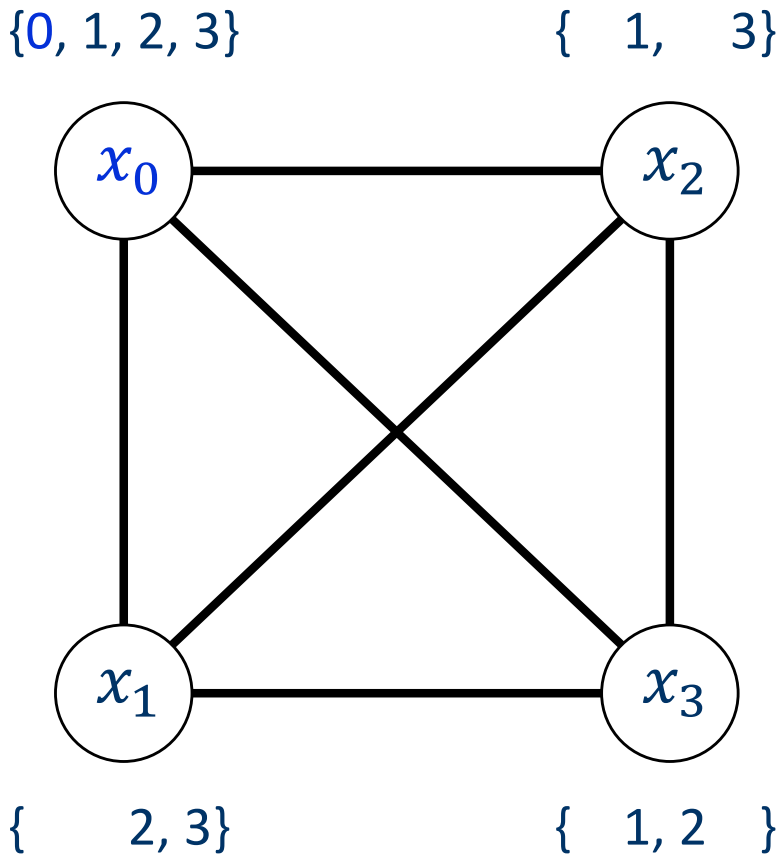
Solving 4-Queens using Forward Checking



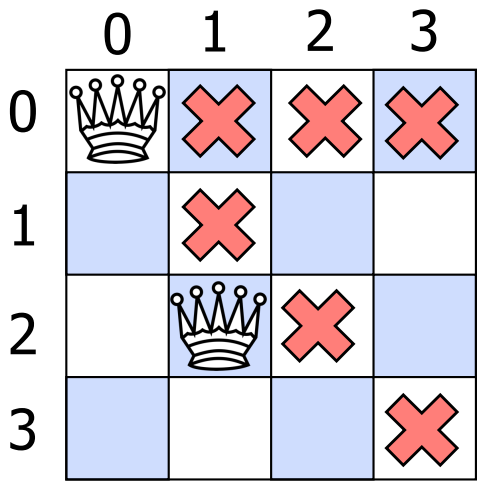


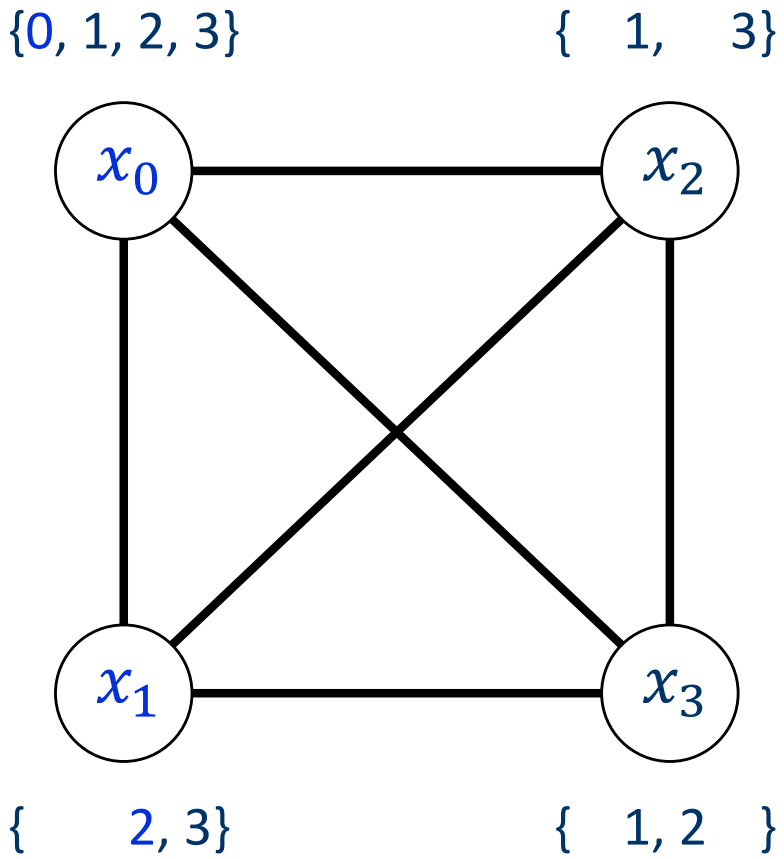
Solving 4-Queens using Forward Checking



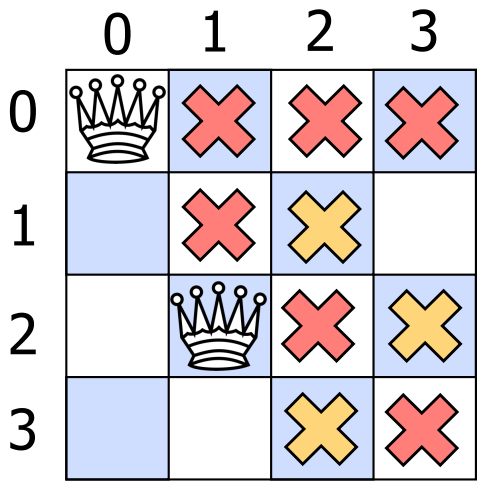


Solving 4-Queens using Forward Checking

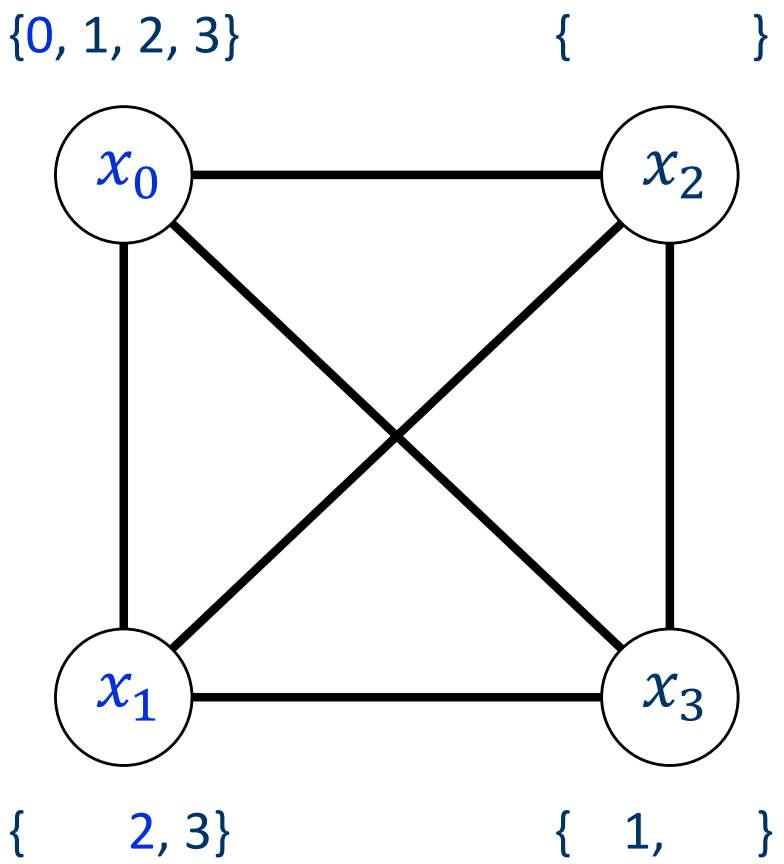




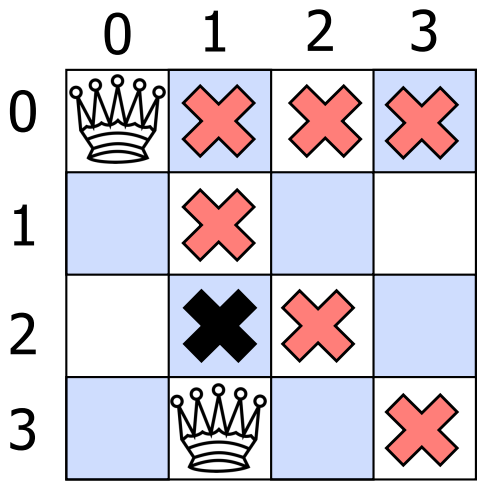
Solving 4-Queens using Forward Checking

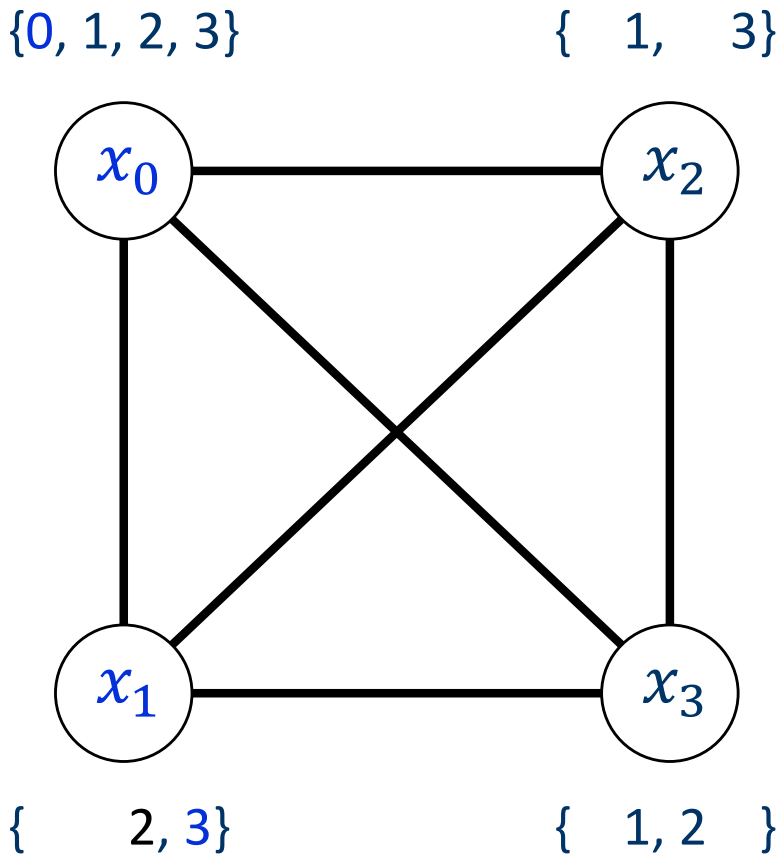


x_2's domain is empty! backtrack!

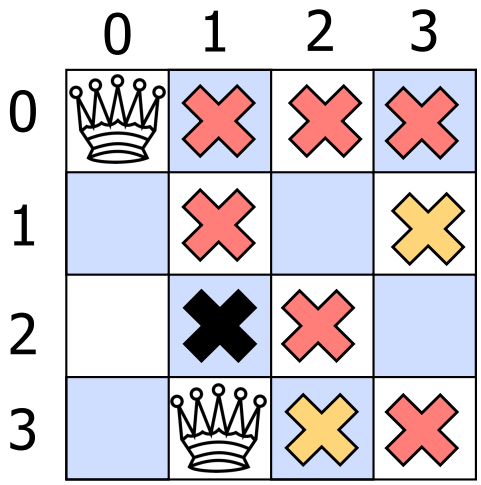


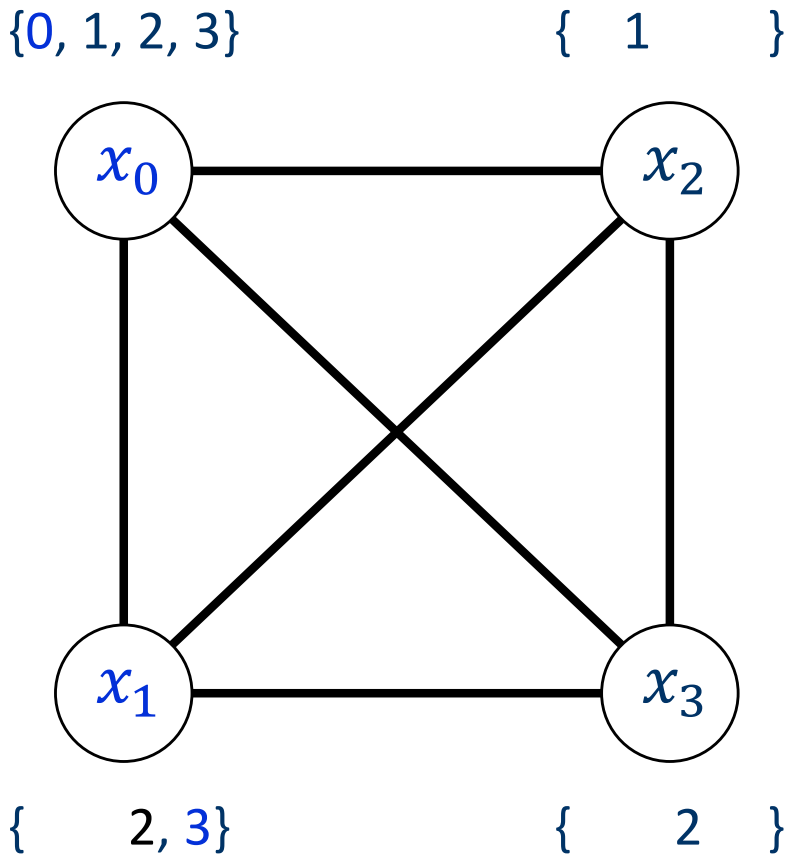
Solving 4-Queens using Forward Checking



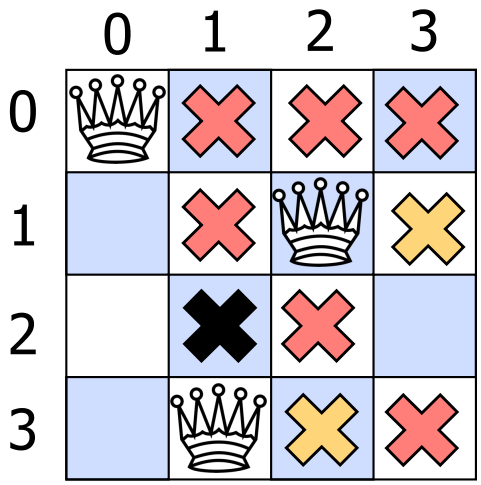


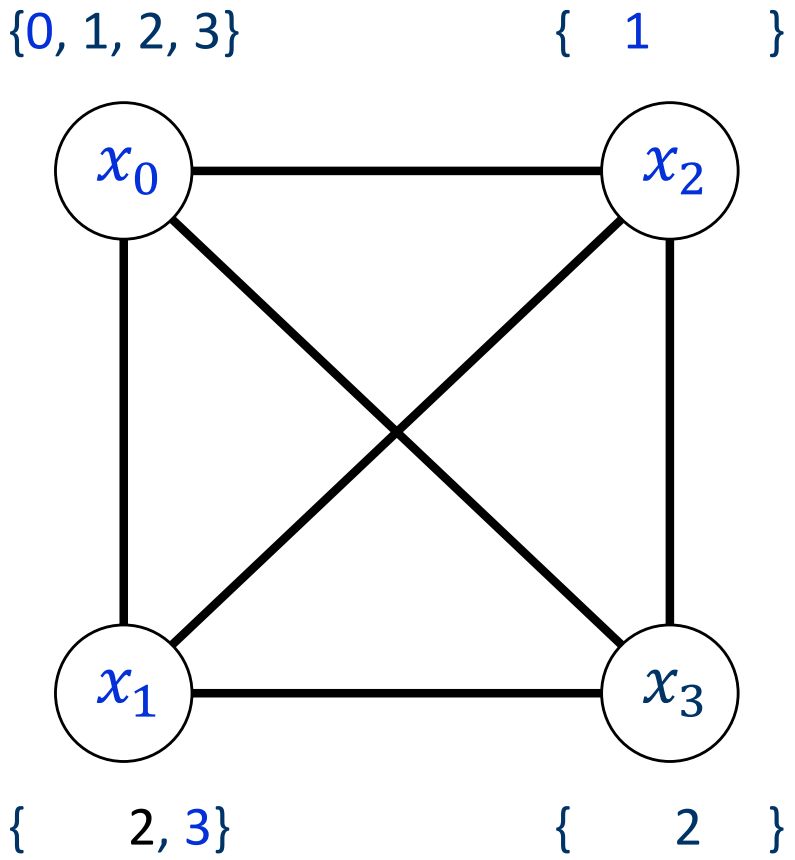
Solving 4-Queens using Forward Checking



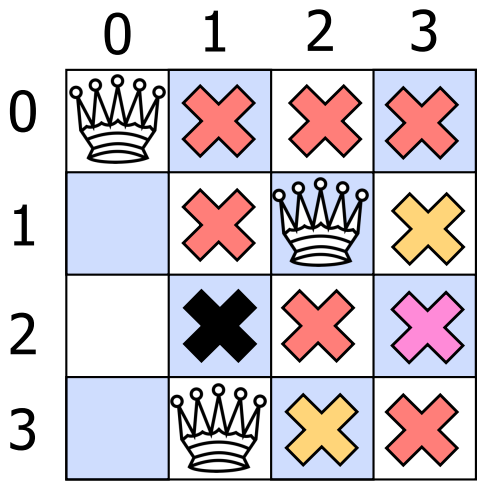


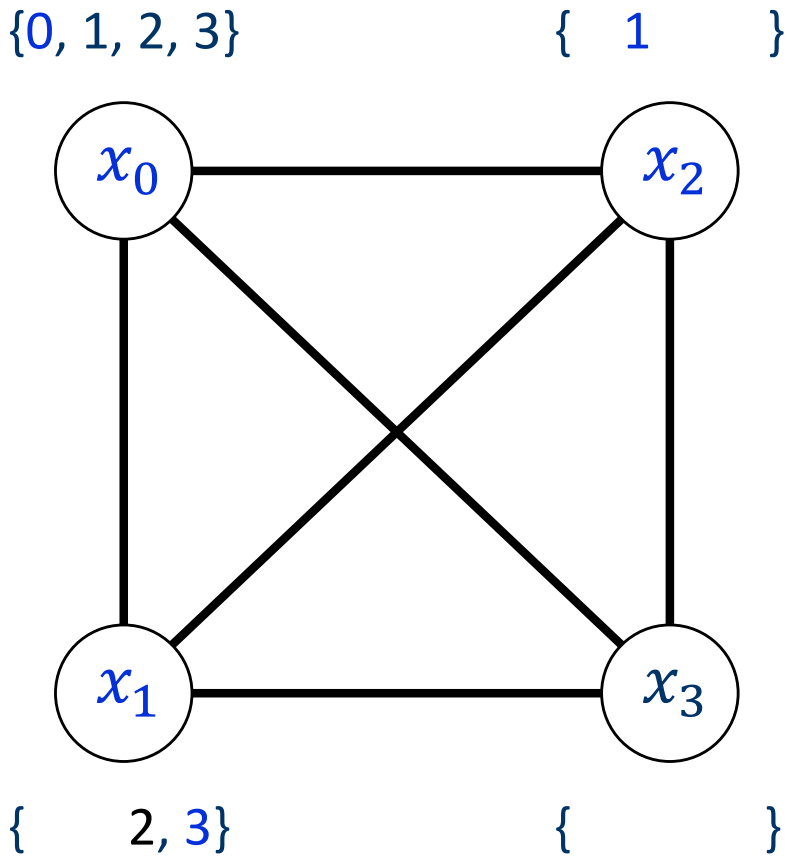
Solving 4-Queens using Forward Checking





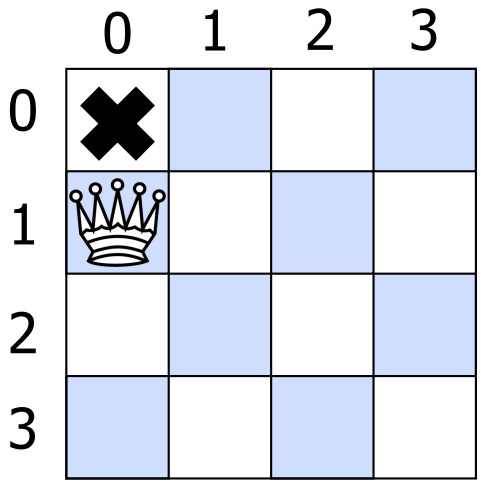
Solving 4-Queens using Forward Checking

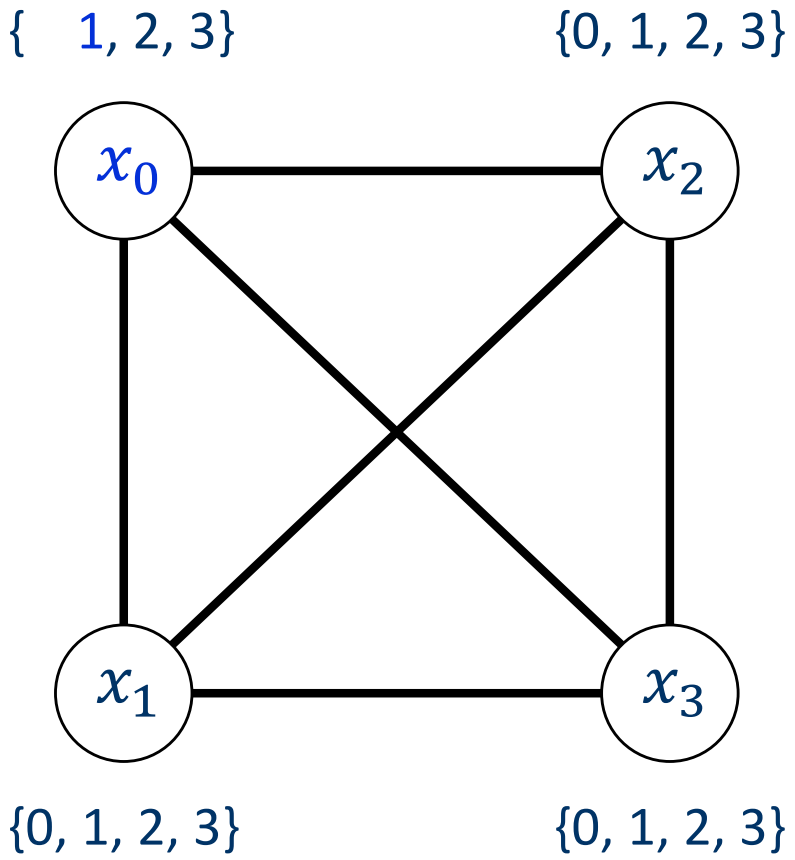




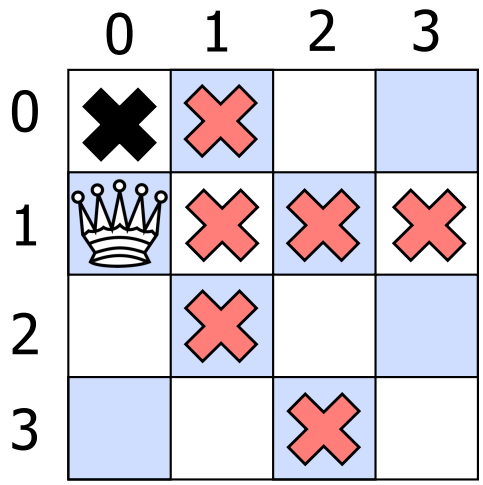
x_3's domain is empty!
backtrack!

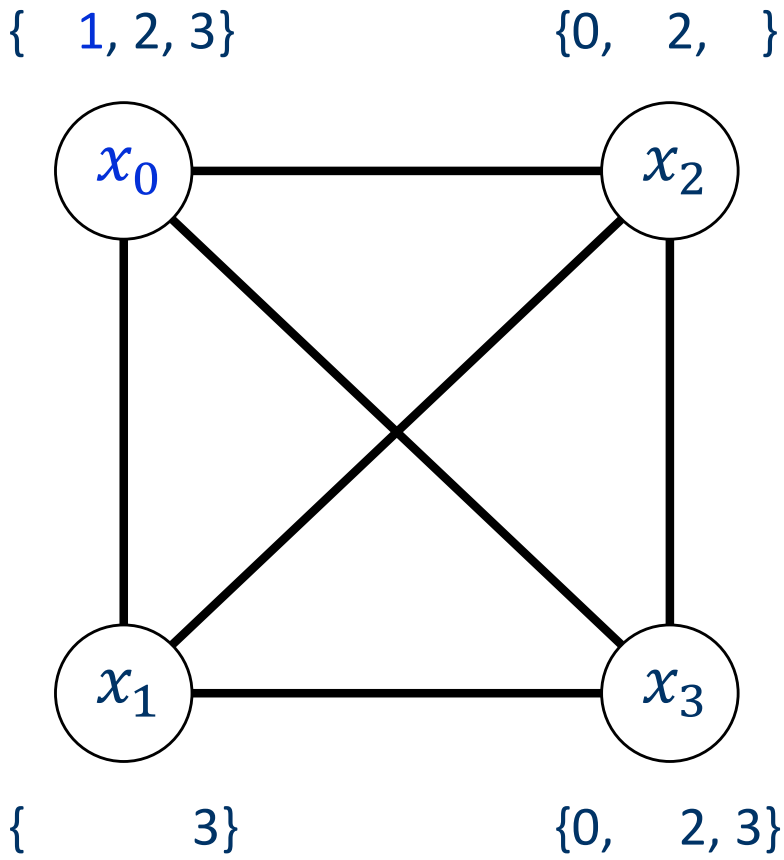
Solving 4-Queens using Forward Checking



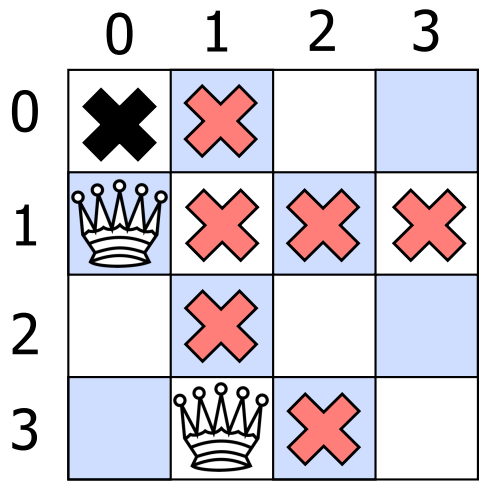


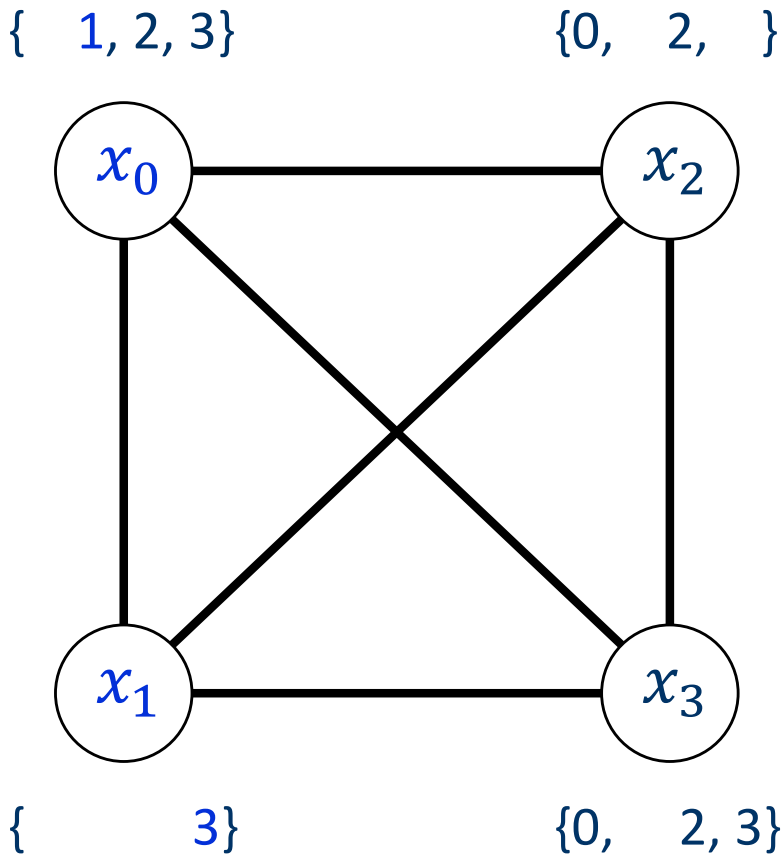
Solving 4-Queens using Forward Checking



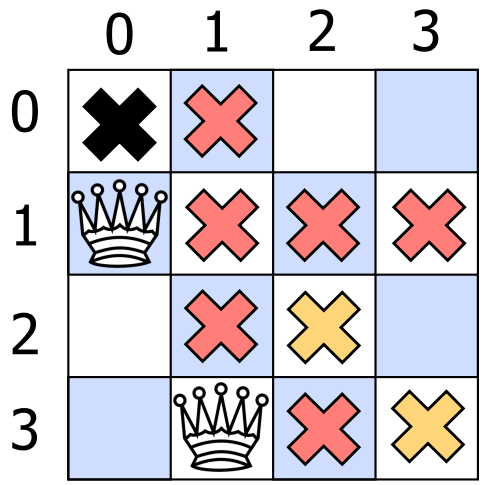


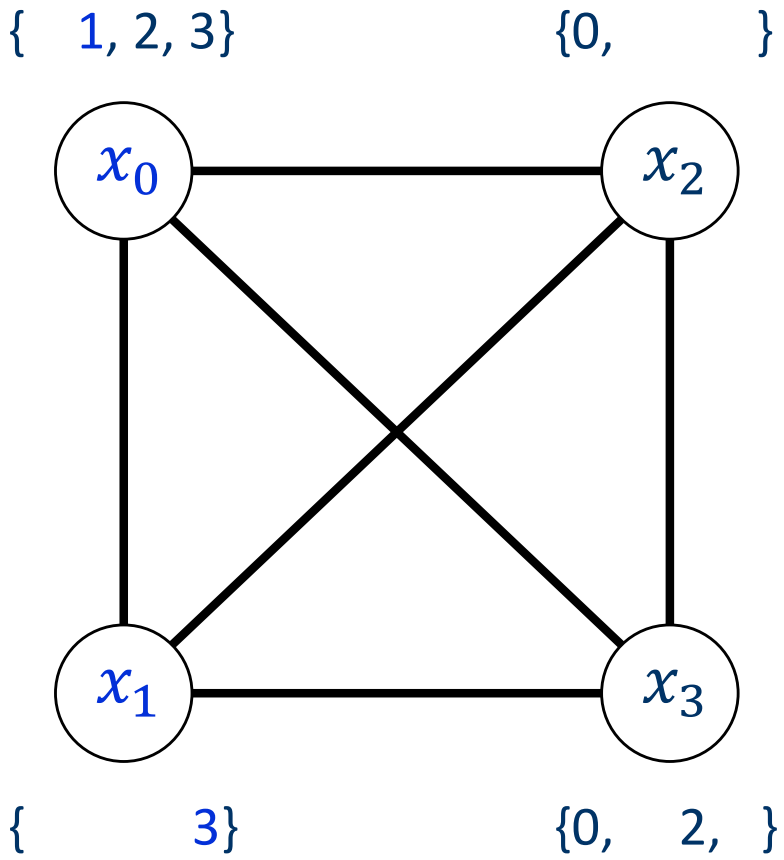
Solving 4-Queens using Forward Checking



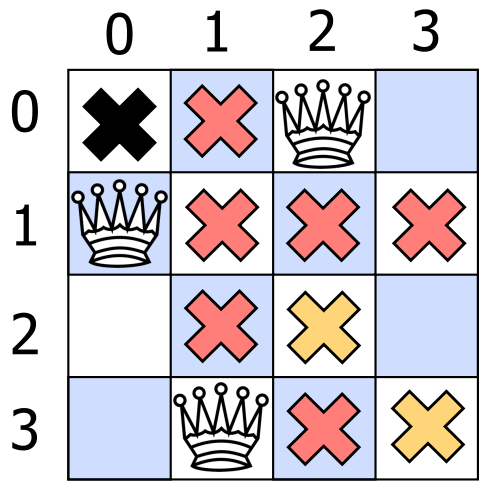


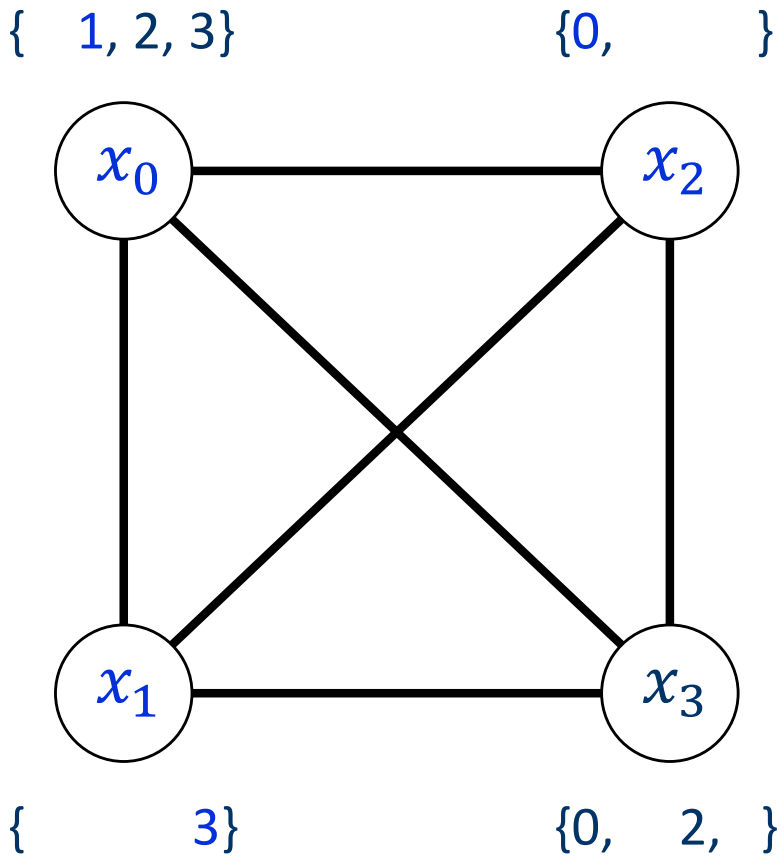
Solving 4-Queens using Forward Checking



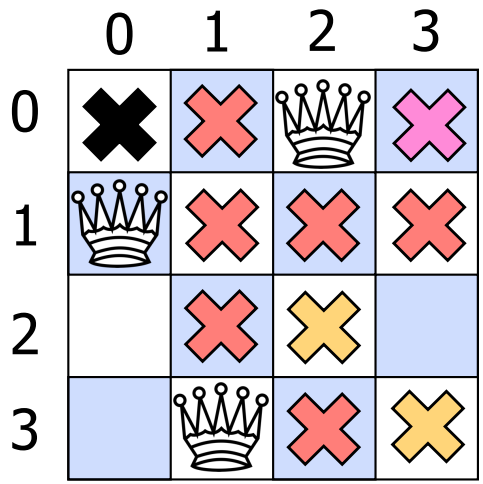


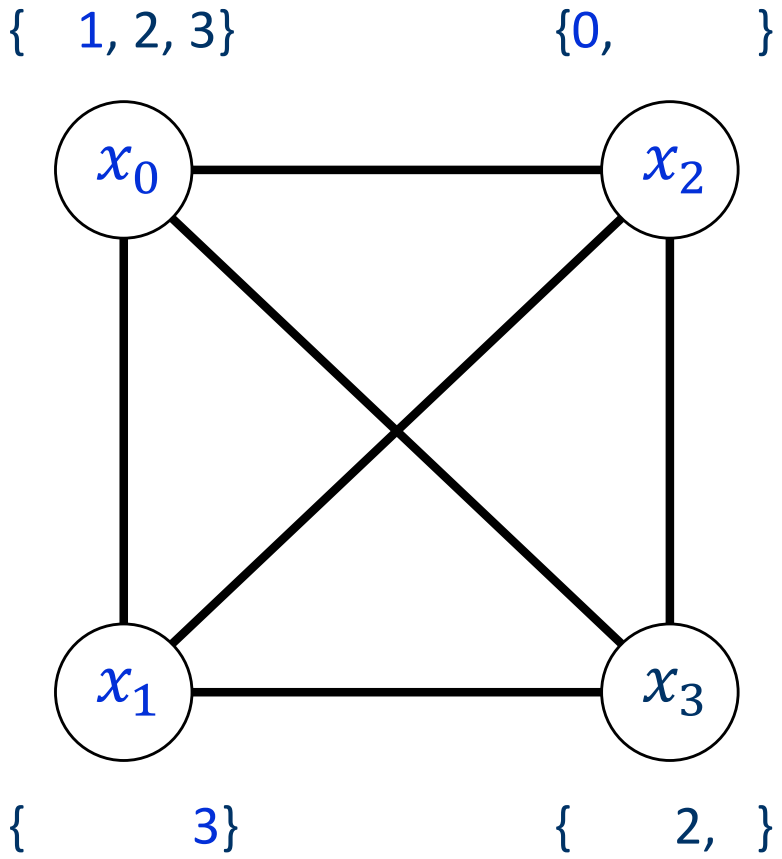
Solving 4-Queens using Forward Checking



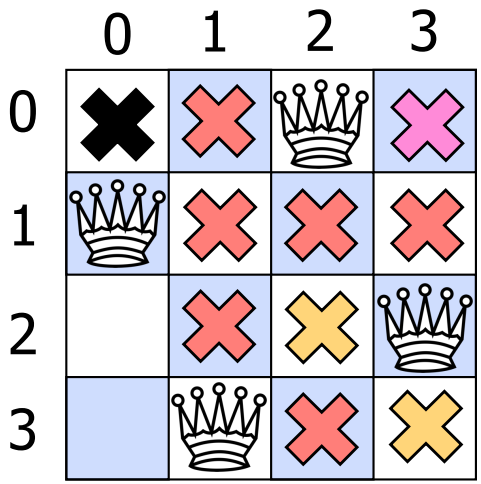


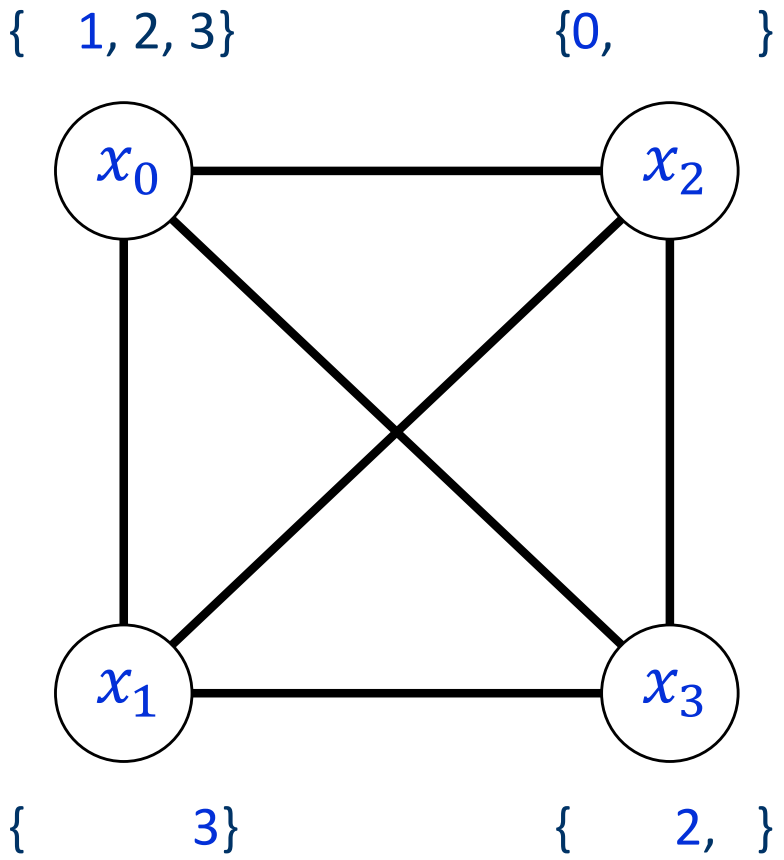
Solving 4-Queens using Forward Checking





Solving 4-Queens using Forward Checking



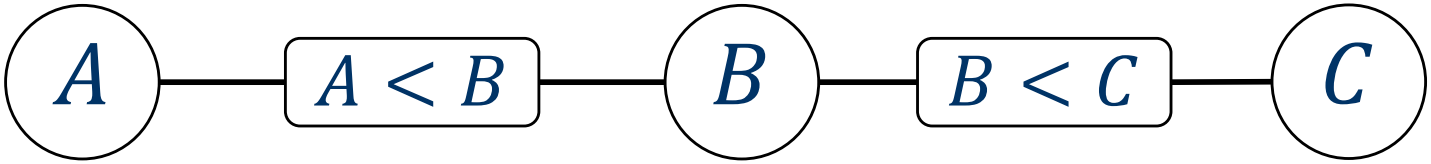


Extra Example 1

Consider the CSP below. Each variable's domain is {1, 2, 3, 4}.

Solve the CSP using Backtracking Search and Forward Checking.

For each variable, consider the values in increasing order.



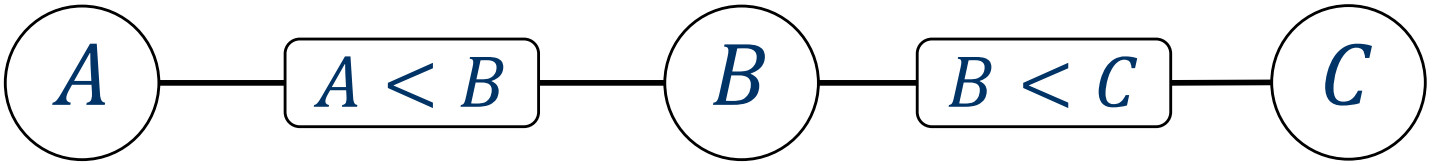
Step	Assigning a Value and Revising Domains			What Next?
	A	B	C	
1				
2				
3				

Extra Example 2

Consider the CSP below. Each variable's domain is {1, 2, 3, 4}.

Solve the CSP using Backtracking Search and Forward Checking.

For each variable, consider the values in decreasing order.



Step	Assigning a Value and Revising Domains			What Next?
	A	B	C	
1				
2				
3				
4				
5				
6				