Constraint Satisfaction Problems (CSPs)

CS 221 Section – 11/02/18

Chinmayee Shah and Vivian Hsu

Agenda

- CSP Problem Modeling
- N-ary Constraints
- Elimination Example

- CSP Problem Modeling
- N-ary Constraints
- Elimination Example

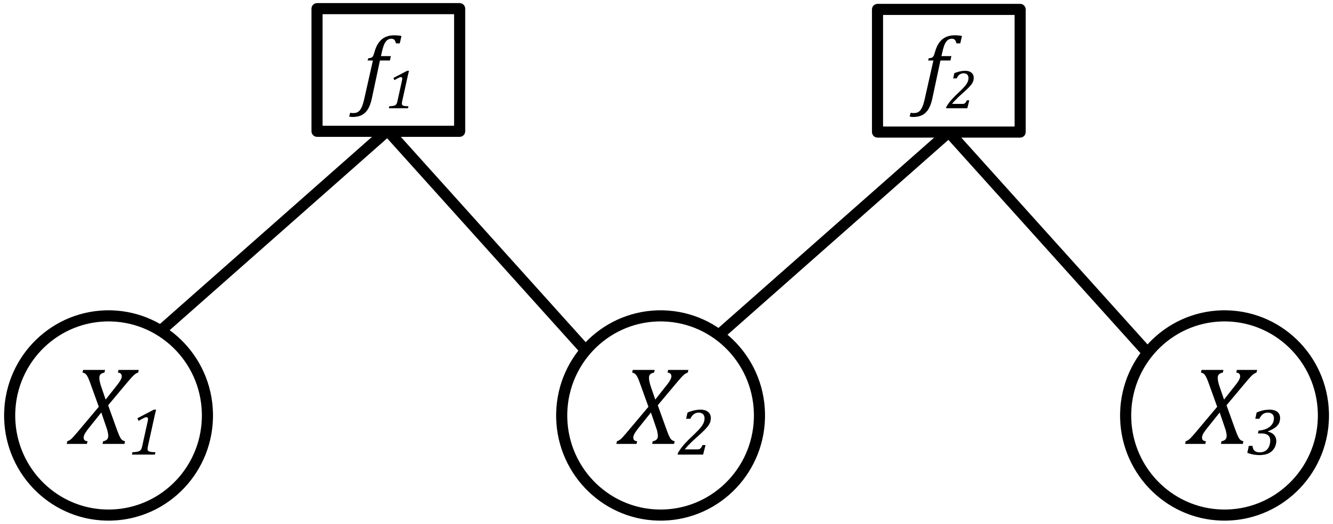
Definition: Factor Graph

Variables:

$X = (X_1, ..., X_n), \quad X_i \in Domain_i$ where

Factors:

$f_1, \ldots, f_m,$ with each $f_j(X) \geq 0$



Definition: Constraint Satisfaction Problem (CSP)

A CSP is a factor graph where all factors are **constraints**:

for all $j = 1, \ldots, m$.

The constraint is satisfied iff $f_j(x) = 1$.

Definition: Consistent Assignments

An assignment x if $Weight(x) = 1$ (i.e., all constraints are satisfied.)

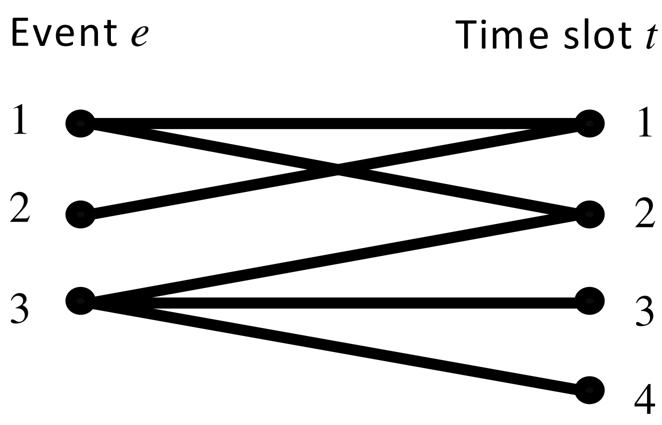
Factor Graph and CSP Applications

- Inferring relations from data
- Scheduling problems: event scheduling, resource and assembly scheduling
- Puzzles: sudoku, crosswards
- Satisfiability problems
- Map and graph coloring
- Object tracking
- Decoding noisy signals (images, messages etc.)

Event Scheduling

Setup:

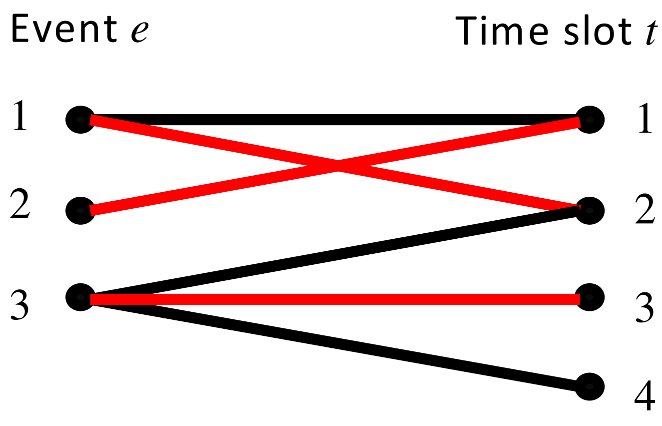
- Have E events and T time slots
- Each event e must be put in **exactly one** time slot
- Each time slot t can have **at most one** event
- Event e only allowed at time slot t if (e, t) in A



Event Scheduling

Setup:

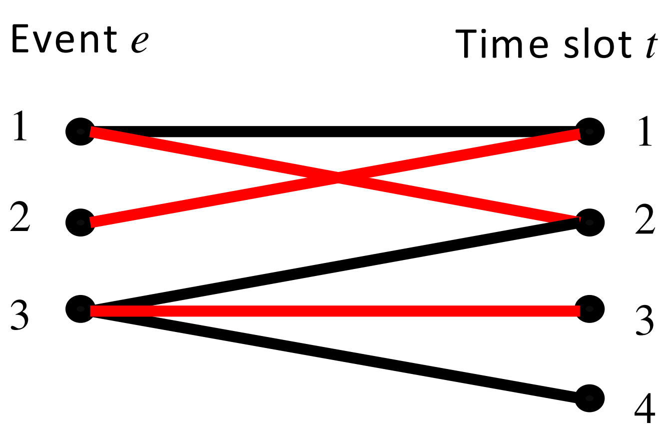
- Have E events and T time slots
- Each event e must be put in **exactly one** time slot
- Each time slot t can have **at most one** event
- Event e only allowed at time slot t if (e, t) in A



Event Scheduling

Formulation 1a:

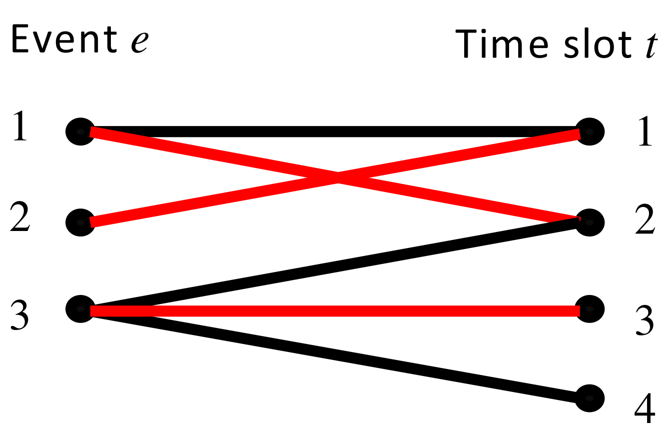
- Variables for each event e, $X_e \in \{1,\ldots,T\}$



Event Scheduling

Formulation 1a:

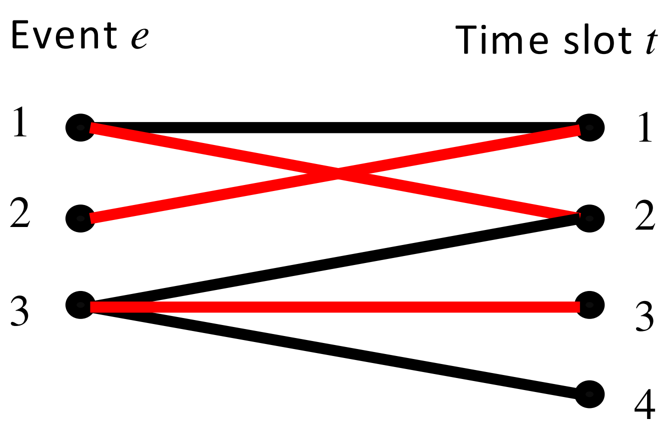
- Variables for each event e, $X_e \in \{1,\ldots,T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$



Event Scheduling

Formulation 1a:

- Variables for each event e, $X_e \in \{1,\ldots,T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$

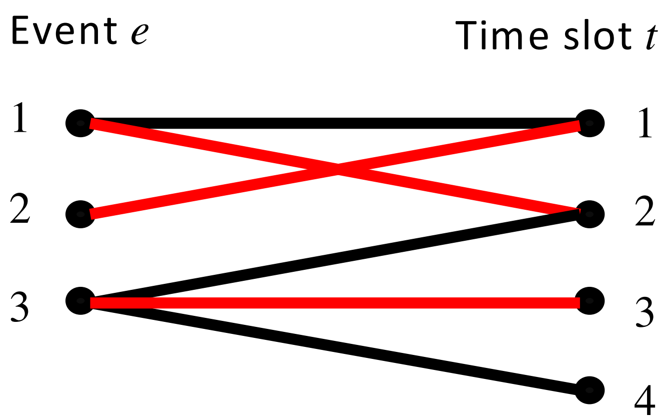


$\{X_1 : 1, X_2 : 1, X_3 : 3\}$

Event Scheduling

Formulation 1a:

- Variables for each event e, $X_e \in \{1, \ldots, T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$



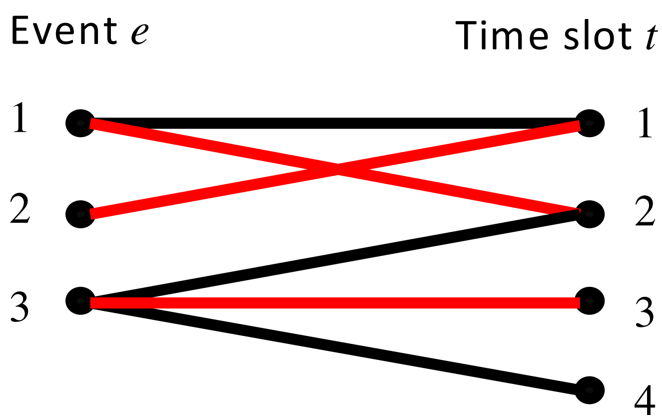
$\{X_1 : 1, X_2 : 1, X_3 : 3\}$

Bad! ($X_1 = X_2$)

Event Scheduling

Formulation 1a:

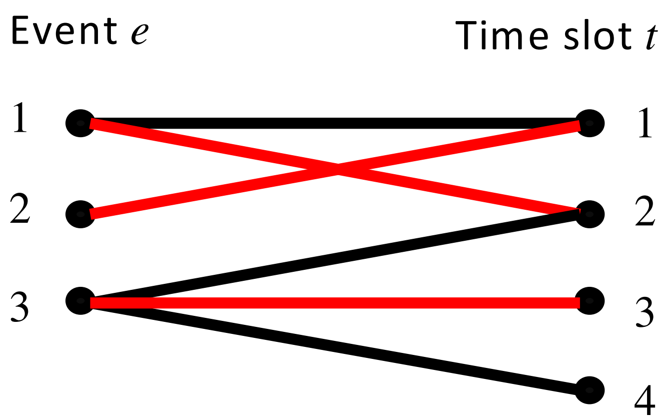
- Variables for each event e, $X_e \in \{1,\ldots,T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$
- Constraints (only schedule allowed times): for each event e, enforce $[(e, X_e) \in A]$



Event Scheduling

Formulation 1a:

- Variables for each event e, $X_e \in \{1, \ldots, T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$
- Constraints (only schedule allowed times): for each event e, enforce $[(e, X_e) \in A]$

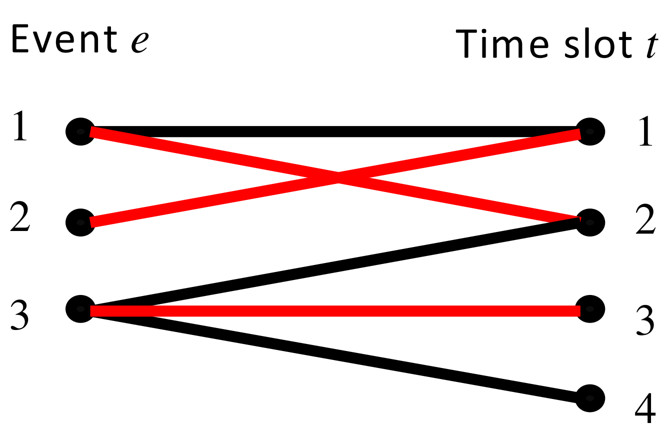


$\{X_1 : 1, X_2 : 4, X_3 : 3\}$

Event Scheduling

Formulation 1a:

- Variables for each event e, $X_e \in \{1, \ldots, T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$
- Constraints (only schedule allowed times): for each event e, enforce $[(e, X_e) \in A]$



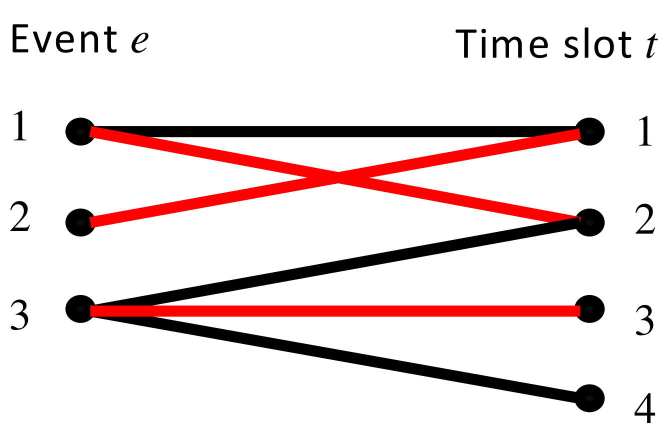
$\{X_1 : 1, X_2 : 4, X_3 : 3\}$

Bad! ($(2, 4)$ not in A)

Event Scheduling

Formulation 1a:

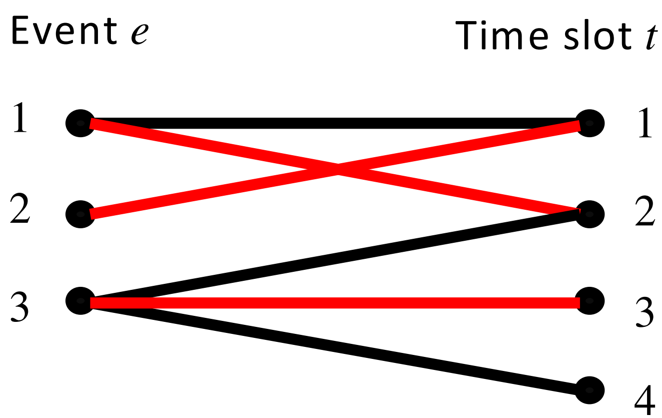
- Variables for each event e, $X_e \in \{1,\ldots,T\}$
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$
- Constraints (only schedule allowed times): for each event e, enforce $[(e, X_e) \in A]$



Event Scheduling

Formulation 1b:

- Variables for each event e, $X_1, \ldots, X_E$

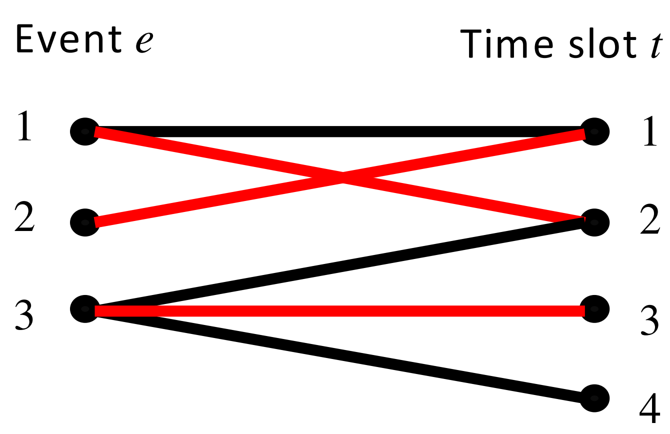


Event Scheduling

Formulation 1b:

- Variables for each event e, $X_1, \ldots, X_E$

$$Domain_i = \{t : (i,t) \in A\}$$



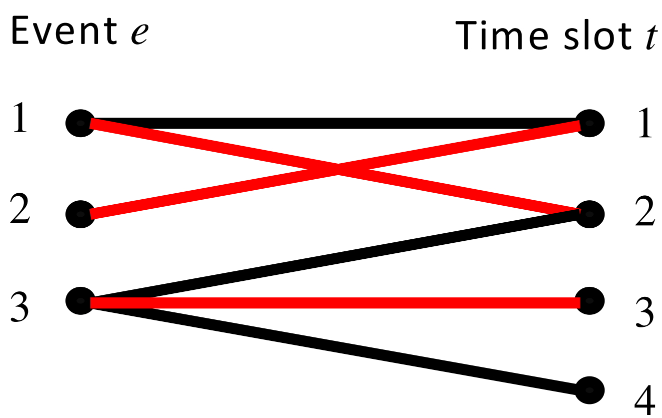
Event Scheduling

Formulation 1b:

- Variables for each event e, $X_1, \ldots, X_E$

$$Domain_i = \{t : (i,t) \in A\}$$

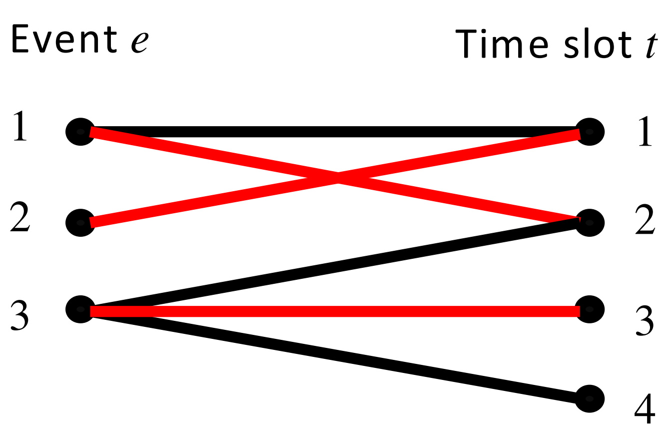
- Constraints (only one event per time slot): for each pair of events $e \neq e'$, enforce $[X_e \neq X_{e'}]$



Event Scheduling

Formulation 2a:

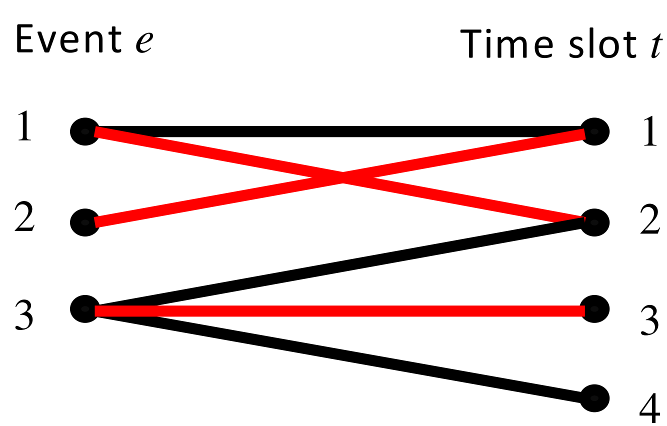
- Variables for each time slot t: $Y_t \in \{1,\ldots,E\} \cup \{\varnothing\}$



Event Scheduling

Formulation 2a:

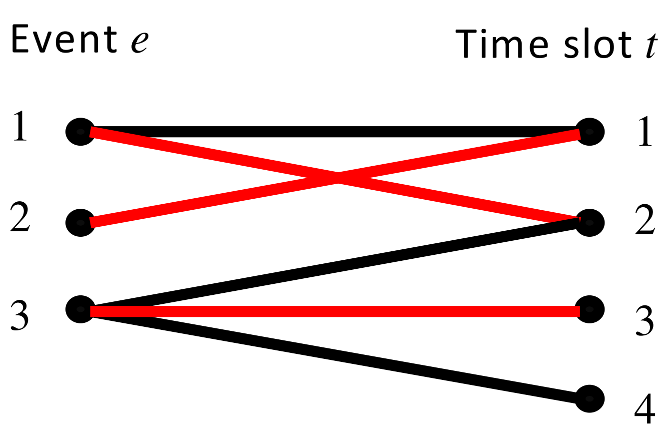
- Variables for each time slot t: $Y_t \in \{1, \ldots, E\} \cup \{\varnothing\}$
- Constraints (each event is scheduled exactly once): for each event e, enforce $[Y_t = e \text{ for exactly one } t]$



Event Scheduling

Formulation 2a:

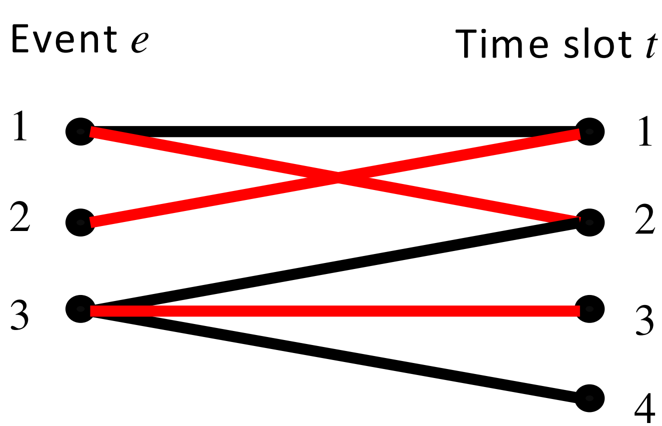
- Variables for each time slot t: $Y_t \in \{1, \ldots, E\} \cup \{\varnothing\}$
- Constraints (each event is scheduled exactly once): for each event e, enforce $[Y_t = e \text{ for exactly one } t]$
- Constraints (only schedule allowed times): for each time slot t, enforce $[Y_t = \varnothing \text{ or } (Y_t, t) \in A]$



Event Scheduling

Formulation 2a:

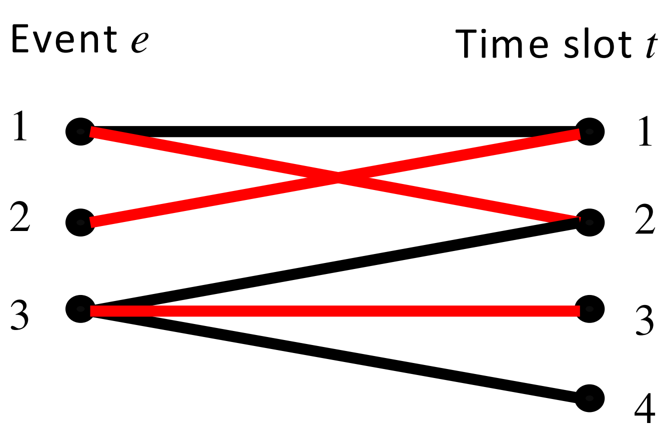
- Variables for each time slot t: $Y_t \in \{1,\ldots,E\} \cup \{\varnothing\}$
- Constraints (each event is scheduled exactly once): for each event e, enforce $[Y_t = e$ for exactly one $t]$
- Constraints (only schedule allowed times): for each time slot t, enforce $[Y_t = \varnothing$ or $(Y_t, t) \in A]$



Event Scheduling

Formulation 2a:

- Variables for each time slot t: $Y_1, \ldots, Y_T$

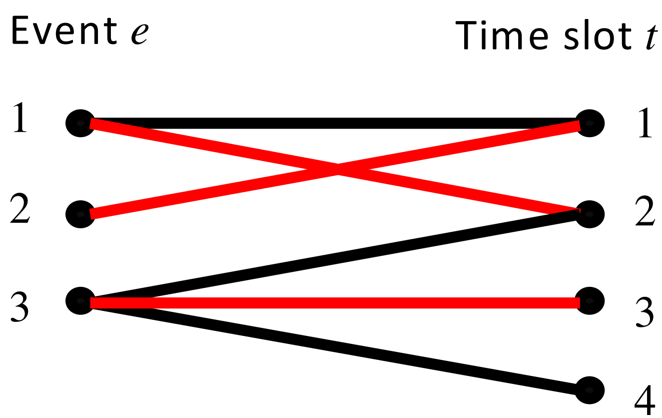


Event Scheduling

Formulation 2a:

- Variables for each time slot t: $Y_1, \ldots, Y_T$

$$Domain_i = \{e : (e, i) \in A\} \cup \{\varnothing\}$$



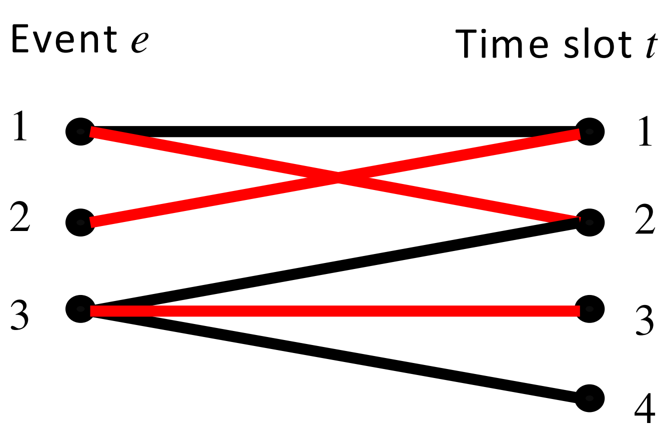
Event Scheduling

Formulation 2a:

- Variables for each time slot t: $Y_1, \ldots, Y_T$

$$Domain_i = \{e : (e,i) \in A\} \cup \{\emptyset\}$$

- Constraints (each event is scheduled exactly once): for each event e, enforce $[Y_t = e \text{ for exactly one } t]$

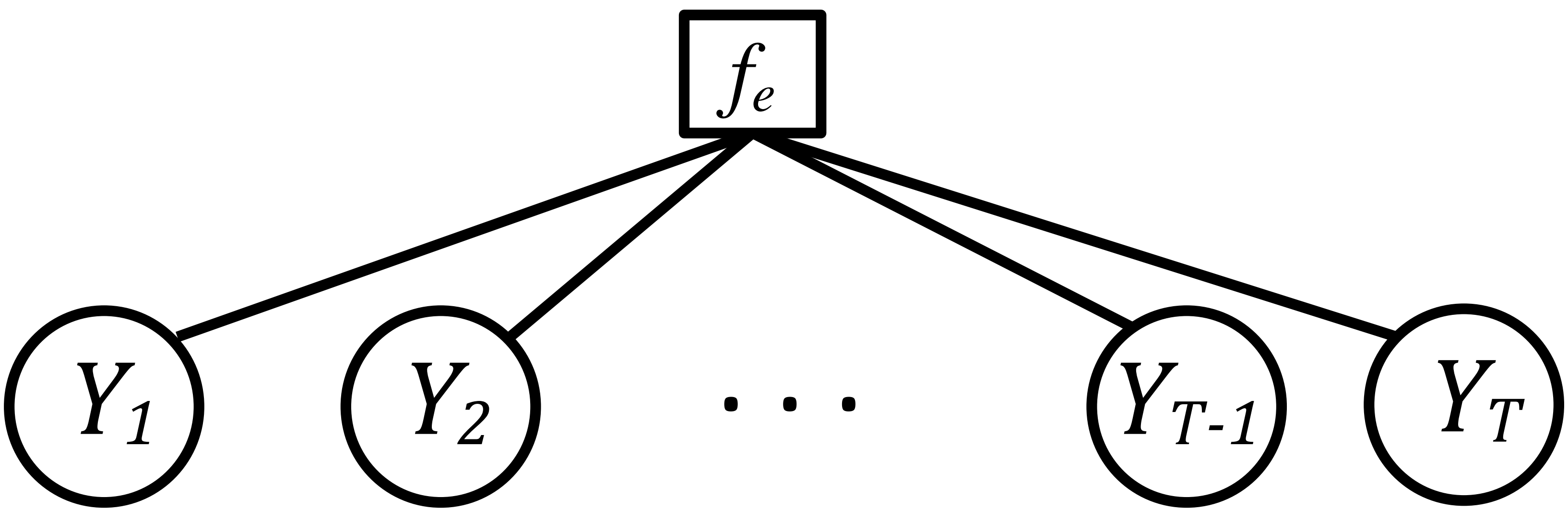


- CSP Problem Modeling
- N-ary Constraints
- Elimination Example

N-ary Constraints

- From event scheduling:
  - Constraints (each event is scheduled exactly once): for each event e, enforce

$$[Y_t = e \text{ for exactly one } t]$$



N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0				

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = A_{i-1} + 1[Y_i = e]]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0				

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0				

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0	0			

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0	0	1		

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0	0	1	1	

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0	0	1	1	2

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

Final Output: $1[A_T = 1]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0	0	1	1	2

N-ary Constraints

Key Idea: Auxiliary Variables

Auxiliary Variables hold intermediate computation.

Represent "for exactly one" as counting the number of values equal to e and constraining that count to be equal to one.

Factors:

Initialization: $[A_0 = 0]$

Processing: $[A_i = \min(A_{i-1} + 1[Y_i = e], 2)]$

Final Output: $1[A_T = 1]$

i	0	1	2	3	4
Y_i		3	1	2	1
A_i	0	0	1	1	2

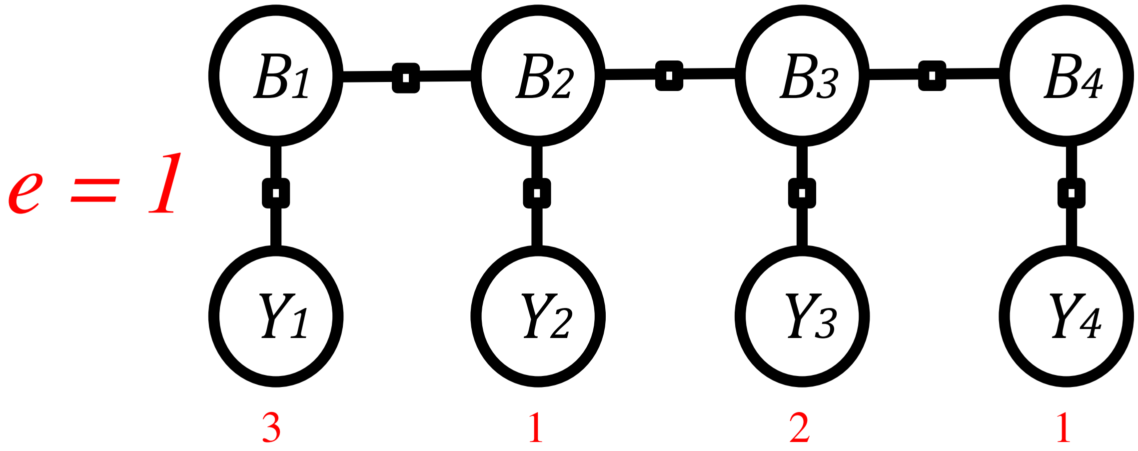
Still have factors with three variables…

N-ary Constraints

Key idea: Combine A_{i-1} and A_i into one variable B_i

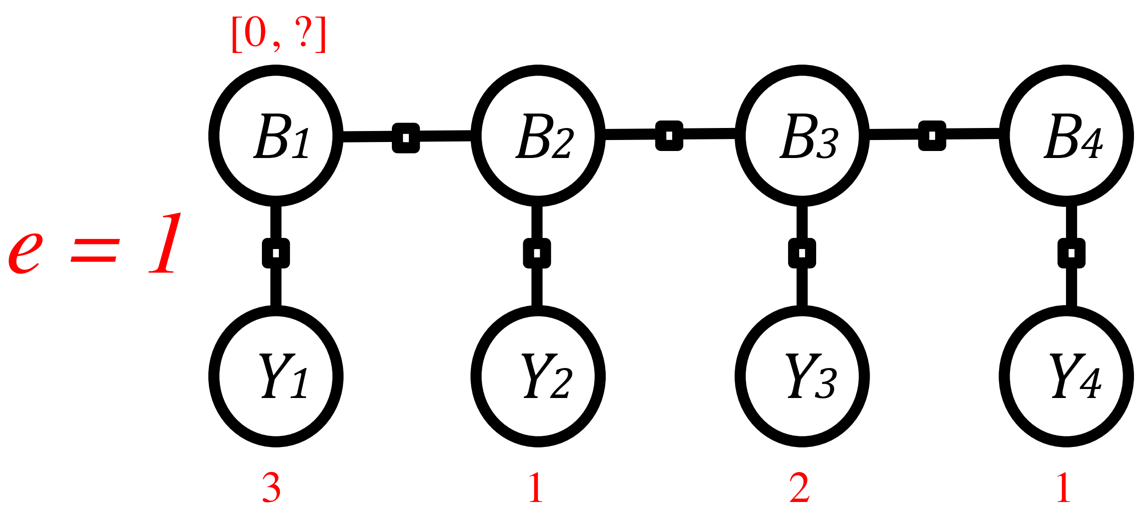
N-ary Constraints

Key idea: Combine A_{i-1} and A_i into one variable B_i



N-ary Constraints

Key idea: Combine A_{i-1} and A_i into one variable B_i

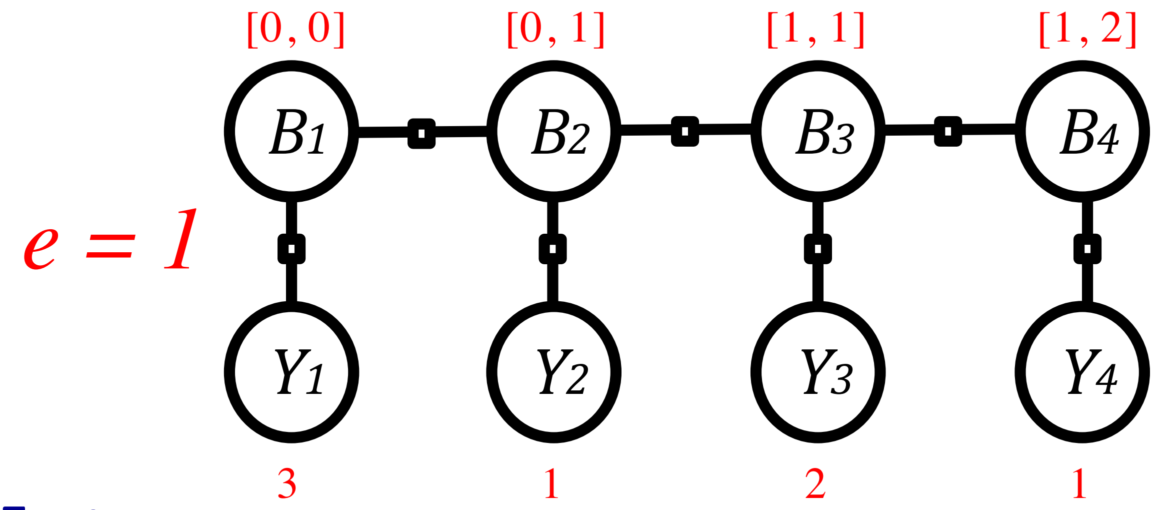


Factors:

Initialization: $[B_1[0] = 0]$

N-ary Constraints

Key idea: Combine A_{i-1} and A_i into one variable B_i



Factors:

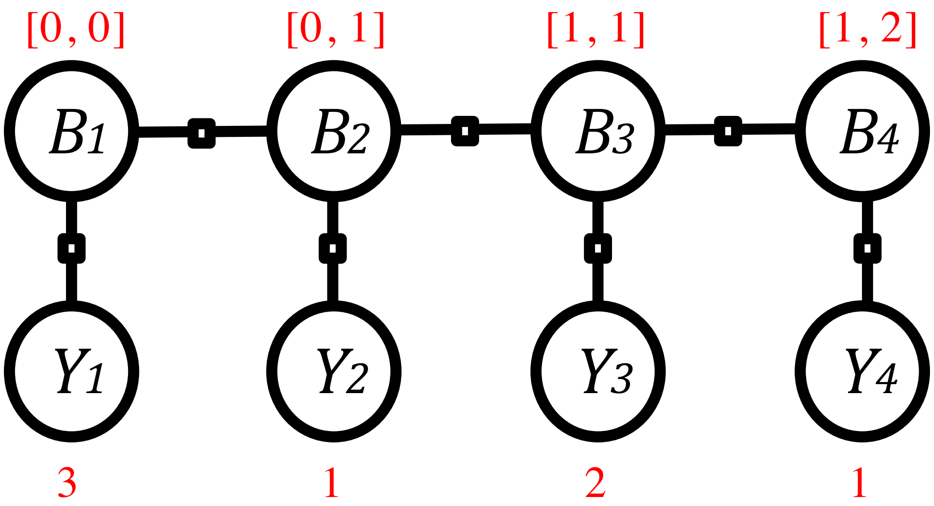
Initialization: $[B_1[0] = 0]$

Processing: $[B_i[1] = \min(B_i[0] + 1[Y_i = e], 2)]$

N-ary Constraints

Key idea: Combine A_{i-1} and A_i into one variable B_i

$e = 1$



Factors:

Initialization: $[B_1[0] = 0]$

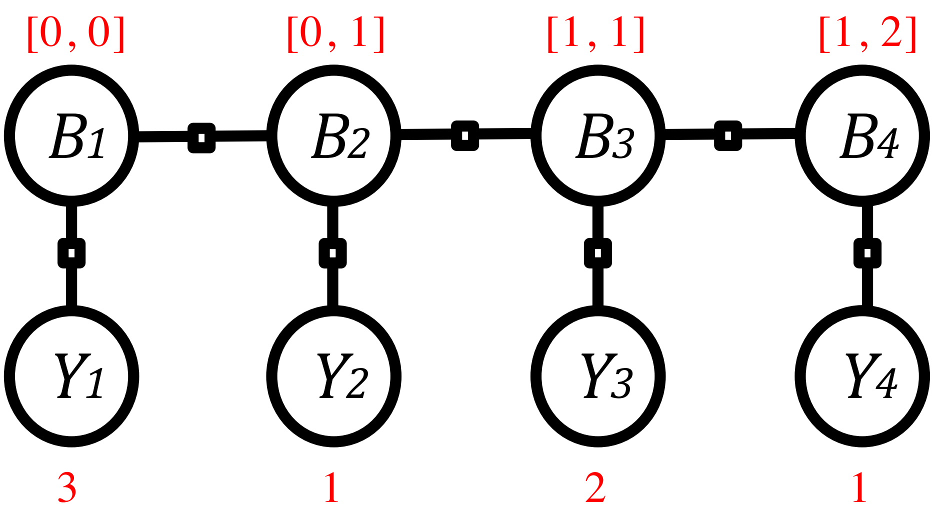
Processing: $[B_i[1] = \min(B_i[0] + 1[Y_i = e], 2)]$

Consistency: $[B_{i-1}[1] = B_i[0]]$

N-ary Constraints

Key idea: Combine A_{i-1} and A_i into one variable B_i

$e = 1$



Factors:

Initialization: $[B_1[0] = 0]$

Processing: $[B_i[1] = \min(B_i[0] + 1[Y_i = e], 2)]$

Consistency: $[B_{i-1}[1] = B_i[0]]$

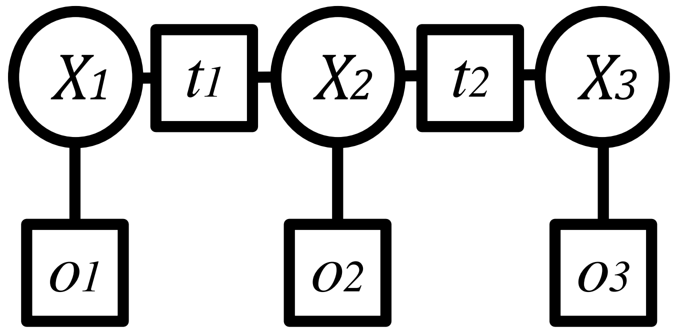
Final Output: $1[B_T[1] = 1]$

- CSP Problem Modeling
- N-ary Constraints
- Elimination Example

Object Tracking

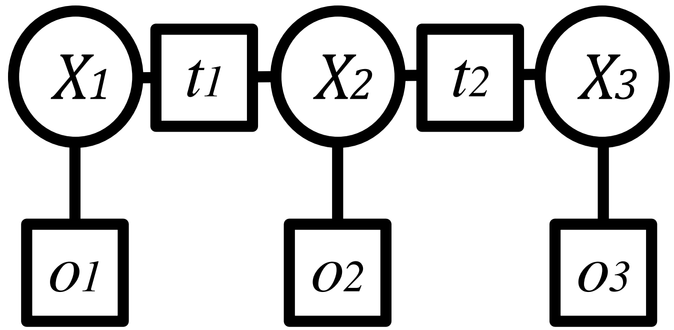
- Sensors provide noisy information about an object's location (e.g., video frames)
- Want to infer object's true location

Person Tracking Example



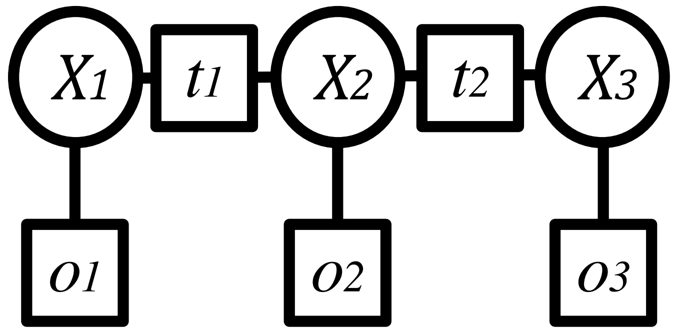
- Variables X_i: Location of object at position i

Person Tracking Example



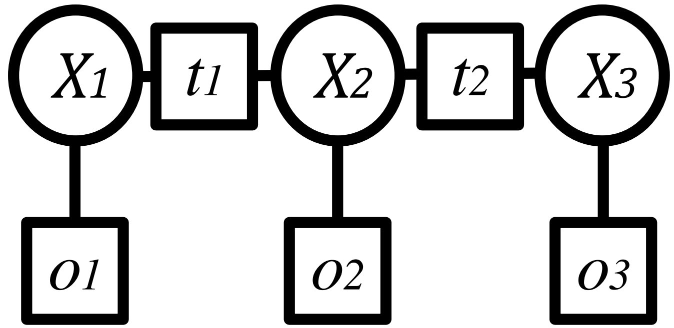
- Variables X_i: Location of object at position i
- Transition Factors $t_i(x_i, x_{i+1})$: object positions can't change too much

Person Tracking Example



- Variables X_i: Location of object at position i
- Transition Factors $t_i(x_i, x_{i+1})$: object positions can't change too much
- Observation Factors $o_i(x_i)$: noisy information compatible with position

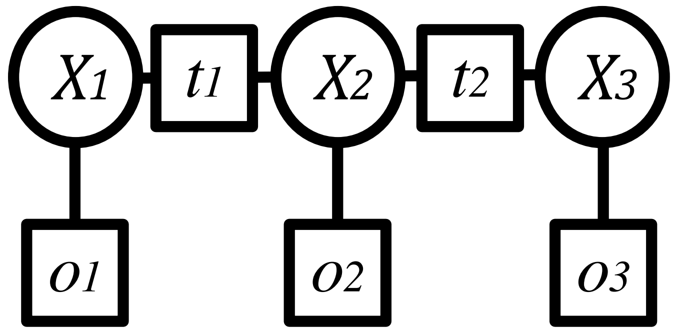
Person Tracking Example



- Variables X_i: Location of object at position i
- Transition Factors $t_i(x_i, x_{i+1})$: object positions can't change too much
- Observation Factors $o_i(x_i)$: noisy information compatible with position

```
def t(x, y):
   if x == y: return 2
   if abs(x – y) == 1: return 1
   return 0
```

Person Tracking Example



- Variables X_i: Location of object at position i
- Transition Factors $t_i(x_i, x_{i+1})$: object positions can't change too much
- Observation Factors $o_i(x_i)$: noisy information compatible with position

```
def t(x, y):
   if x == y: return 2
   if abs(x – y) == 1: return 1
   return 0
```

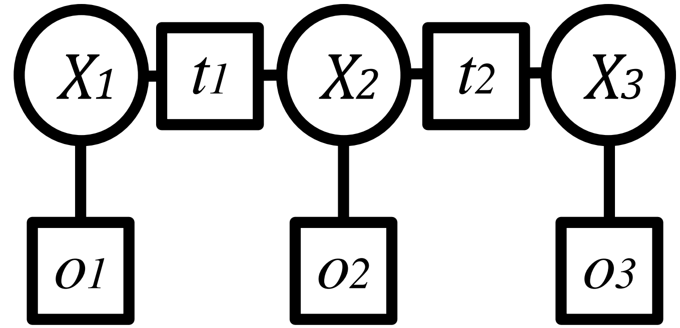
```
def o1(x): return t(x, 0)
def o2(x): return t(x, 2)
def o3(x): return t(x, 2)
```

Variable Elimination

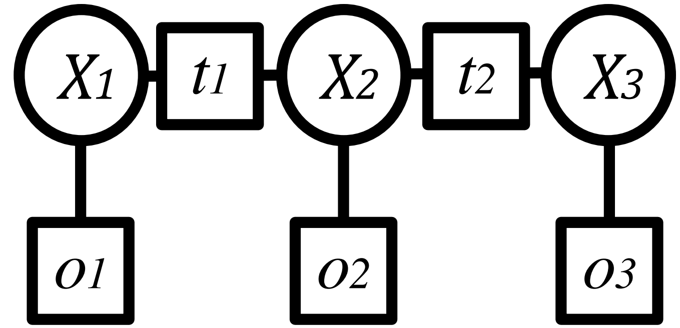
Definition: Elimination

- To **eliminate** a variable X_i, consider all factors f_1, ..., f_k, that depend on X_i
- Remove X_i and f_1, ..., f_k
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

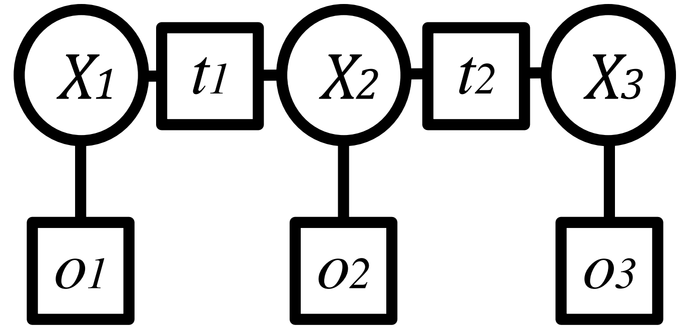
- Eliminate X_1



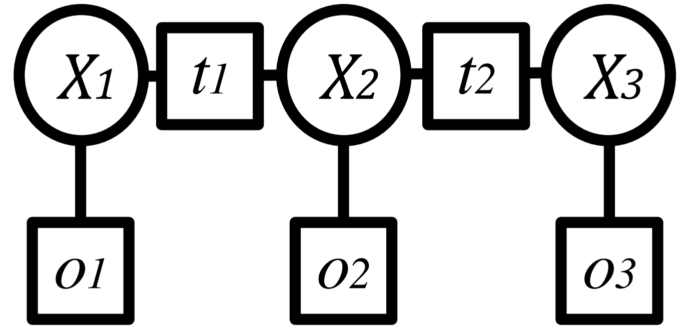
- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1

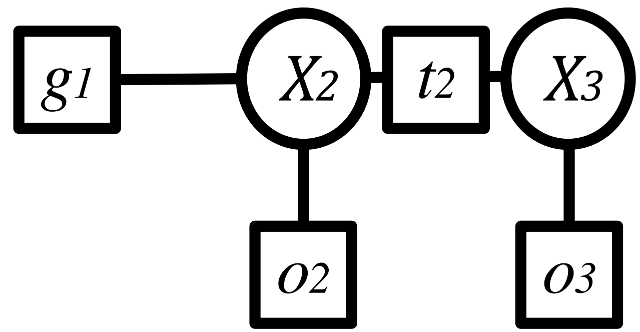


- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

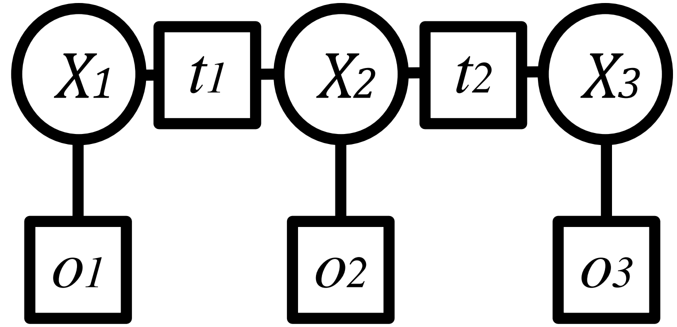


- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$
- $g_1(x_2) = \max_{x_1 \in \{0,1,2\}} o_1(x_1) \cdot t_1(x_1, x_2)$



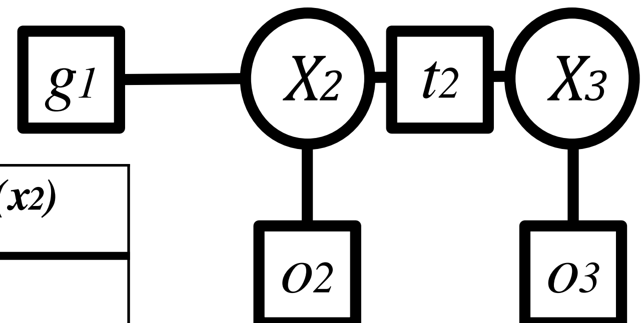


- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1



- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

- $g_1(x_2) = \max_{x_1 \in \{0,1,2\}} o_1(x_1) \cdot t_1(x_1, x_2)$

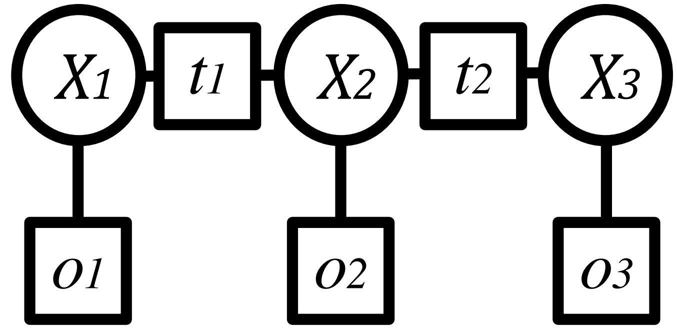


x_2	x_1	$o_1(x_1)$	$t_1(x_1, x_2)$	$o_1(x_1)\, t_1(x_1, x_2)$	$g_1(x_2)$
0	0				
0	1				
0	2				
1	0				
1	1				
1	2				
2	0				
2	1				
2	2				

```
def t(x, y):
   if x == y: return 2
   if abs(x – y) == 1: return 1
   return 0
```

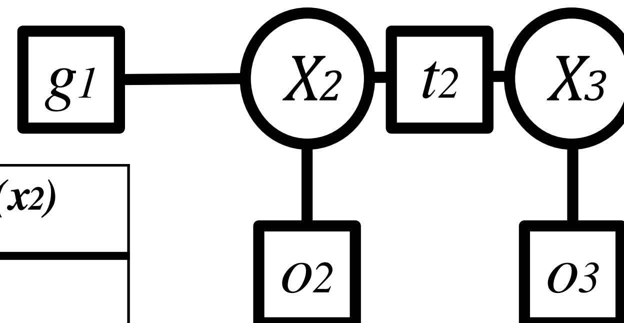
```
def o1(x): return t(x, 0)
def o2(x): return t(x, 2)
def o3(x): return t(x, 2)
```

- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1



- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

- $g_1(x_2) = \max_{x_1 \in \{0,1,2\}} o_1(x_1) \cdot t_1(x_1, x_2)$

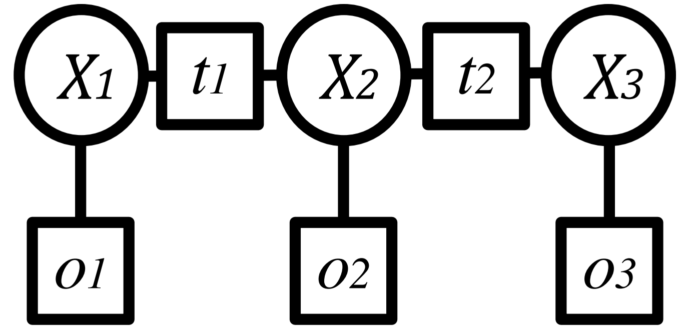


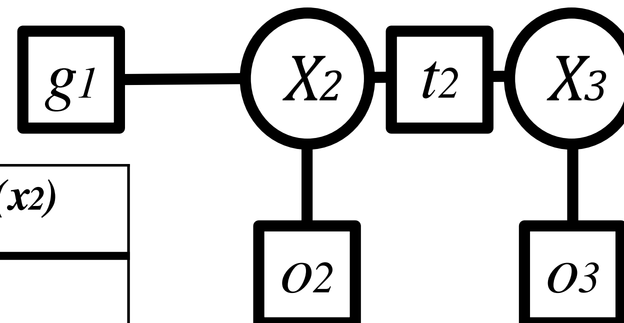
x_2	x_1	$o_1(x_1)$	$t_1(x_1, x_2)$	$o_1(x_1)\ t_1(x_1, x_2)$	$g_1(x_2)$
0	0	2			
0	1	1			
0	2	0			
1	0	2			
1	1	1			
1	2	0			
2	0	2			
2	1	1			
2	2	0			

```
def t(x, y):
  if x == y: return 2
  if abs(x – y) == 1: return 1
  return 0
```

```
def o1(x): return t(x, 0)
def o2(x): return t(x, 2)
def o3(x): return t(x, 2)
```

- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$
- $g_1(x_2) = \max_{x_1 \in \{0,1,2\}} o_1(x_1) \cdot t_1(x_1, x_2)$



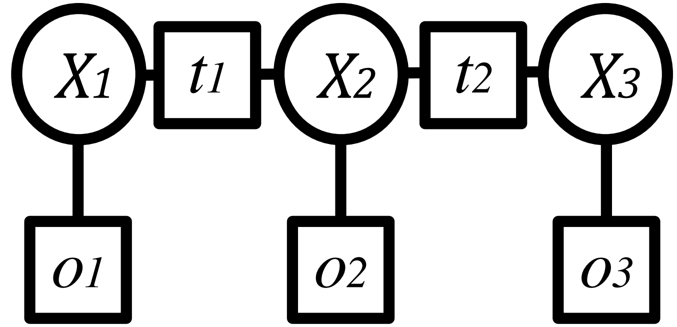


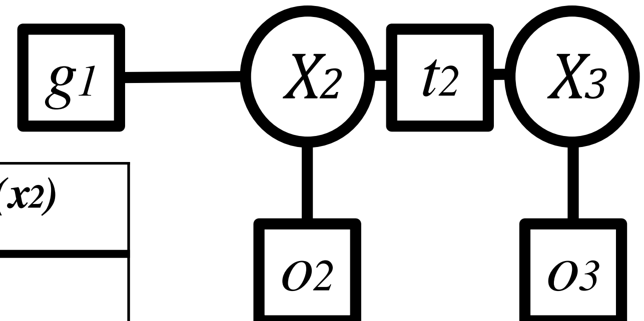
x_2	x_1	$o_1(x_1)$	$t_1(x_1, x_2)$	$o_1(x_1)\ t_1(x_1, x_2)$	$g_1(x_2)$
0	0	2	2		
0	1	1	1		
0	2	0	0		
1	0	2	1		
1	1	1	2		
1	2	0	1		
2	0	2	0		
2	1	1	1		
2	2	0	2		

```
def t(x, y):
  if x == y: return 2
  if abs(x – y) == 1: return 1
  return 0
```

```
def o1(x): return t(x, 0)
def o2(x): return t(x, 2)
def o3(x): return t(x, 2)
```

- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$
- $g_1(x_2) = \max_{x_1 \in \{0,1,2\}} o_1(x_1) \cdot t_1(x_1, x_2)$



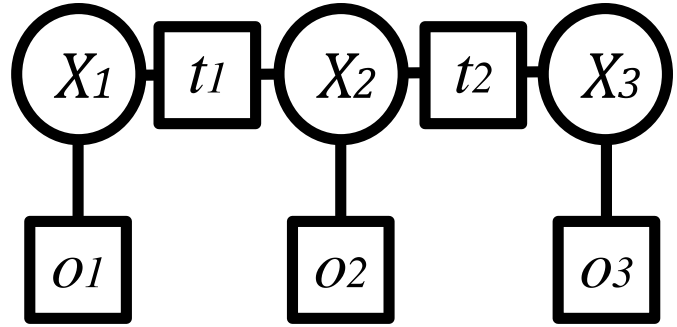


x_2	x_1	$o_1(x_1)$	$t_1(x_1, x_2)$	$o_1(x_1)\ t_1(x_1, x_2)$	$g_1(x_2)$
0	0	2	2	4	
0	1	1	1	1	
0	2	0	0	0	
1	0	2	1	2	
1	1	1	2	2	
1	2	0	1	0	
2	0	2	0	0	
2	1	1	1	1	
2	2	0	2	0	

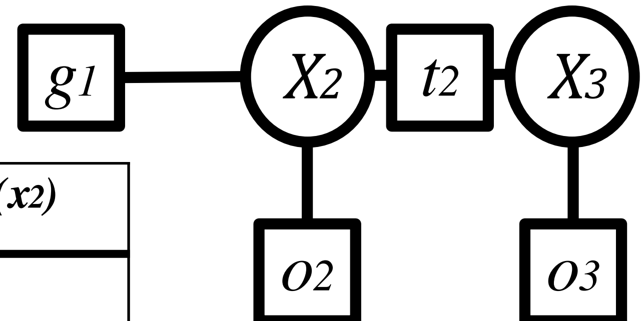
```
def t(x, y):
  if x == y: return 2
  if abs(x – y) == 1: return 1
  return 0
```

```
def o1(x): return t(x, 0)
def o2(x): return t(x, 2)
def o3(x): return t(x, 2)
```

- Eliminate X_1
- Factors that depend on X_1:
  - o_1, t_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$



- $g_1(x_2) = \max_{x_1 \in \{0,1,2\}} o_1(x_1) \cdot t_1(x_1, x_2)$

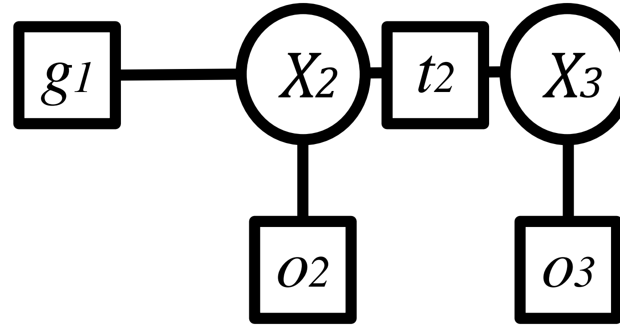


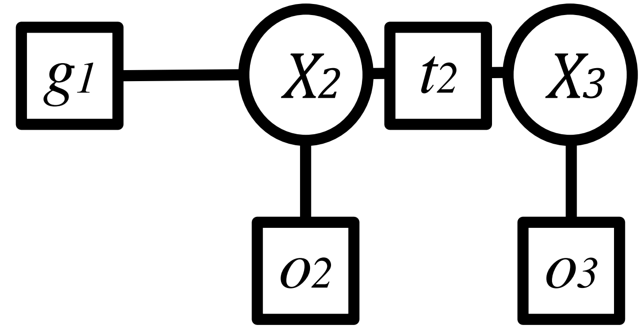
x2	*x1*	*o1(x1)*	*t1(x1, x2)*	*o1(x1) t1(x1, x2)*	*g1(x2)*
0	0	2	2	4	
0	1	1	1	1	4: {*x1:0*}
0	2	0	0	0	
1	0	2	1	2	
1	1	1	2	2	2: {*x1: 1*}
1	2	0	1	0	
2	0	2	0	0	
2	1	1	1	1	1: {*x1: 1*}
2	2	0	2	0	

```
def t(x, y):
  if x == y: return 2
  if abs(x – y) == 1: return 1
  return 0
```

```
def o1(x): return t(x, 0)
def o2(x): return t(x, 2)
def o3(x): return t(x, 2)
```

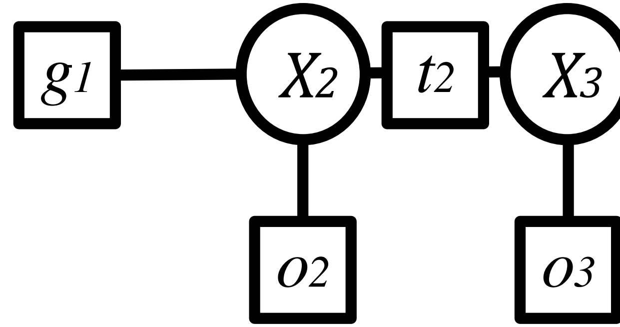
- Eliminate X_2



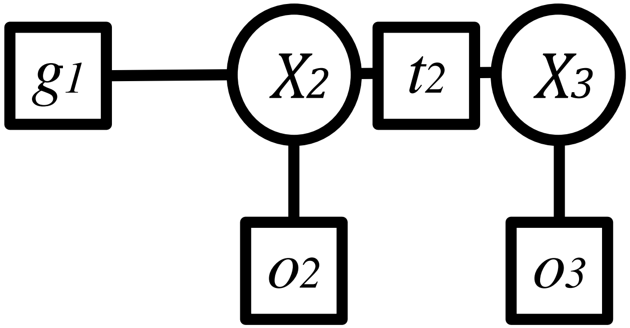


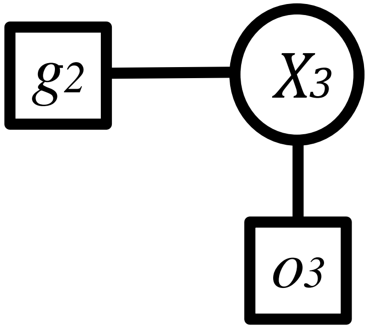
- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1

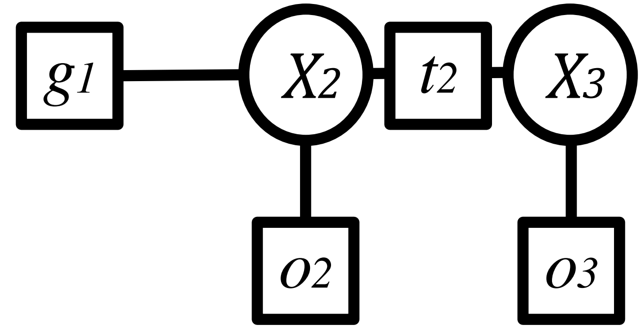
- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $$f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$$



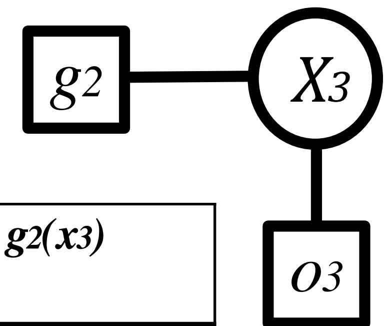
- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$
- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$





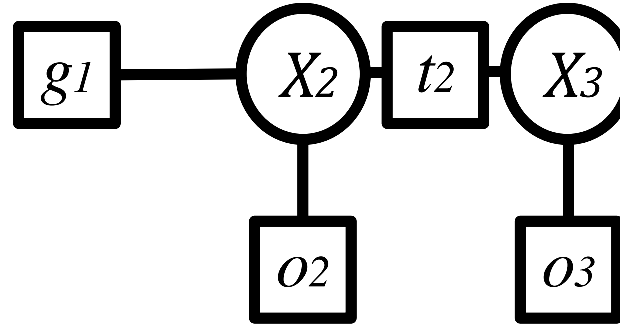


- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

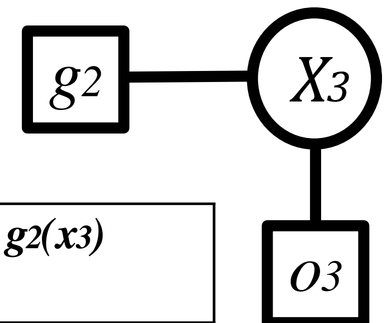


- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$

x3	*x2*	*g1(x2)*	*o2(x2)*	*t2(x2, x3)*	*g1(x2) o2(x2) t2(x2, x3)*	*g2(x3)*
0	0					
0	1					
0	2					
1	0					
1	1					
1	2					
2	0					
2	1					
2	2					

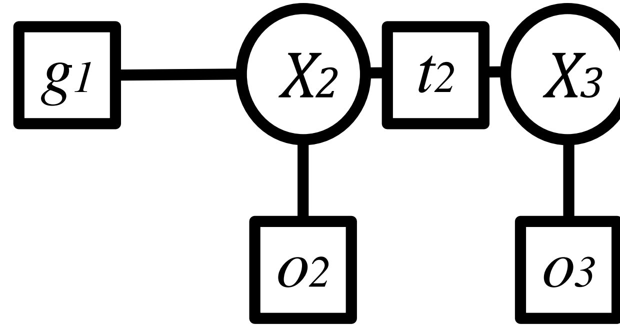


- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

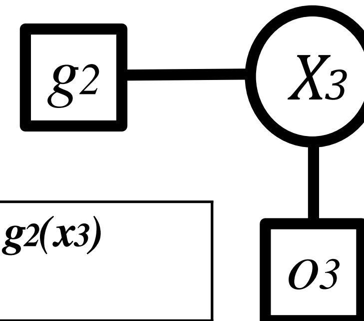


- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$

x3	*x2*	*g1(x2)*	*o2(x2)*	*t2(x2, x3)*	*g1(x2) o2(x2) t2(x2, x3)*	*g2(x3)*
0	0	4: {*x1:0*}				
0	1	2: {*x1: 1*}				
0	2	1: {*x1: 1*}				
1	0	4: {*x1:0*}				
1	1	2: {*x1: 1*}				
1	2	1: {*x1: 1*}				
2	0	4: {*x1:0*}				
2	1	2: {*x1: 1*}				
2	2	1: {*x1: 1*}				

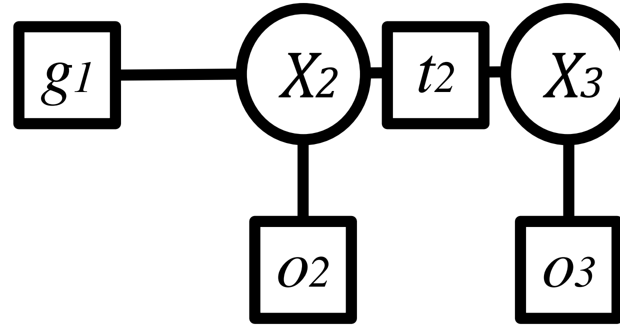


- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

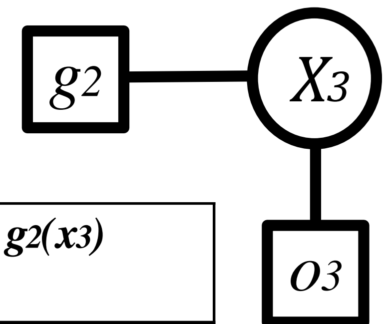


- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$

x3	*x2*	*g1(x2)*	*o2(x2)*	*t2(x2, x3)*	*g1(x2) o2(x2) t2(x2, x3)*	*g2(x3)*
0	0	4: {*x1:0*}	0			
0	1	2: {*x1: 1*}	1			
0	2	1: {*x1: 1*}	2			
1	0	4: {*x1:0*}	0			
1	1	2: {*x1: 1*}	1			
1	2	1: {*x1: 1*}	2			
2	0	4: {*x1:0*}	0			
2	1	2: {*x1: 1*}	1			
2	2	1: {*x1: 1*}	2			



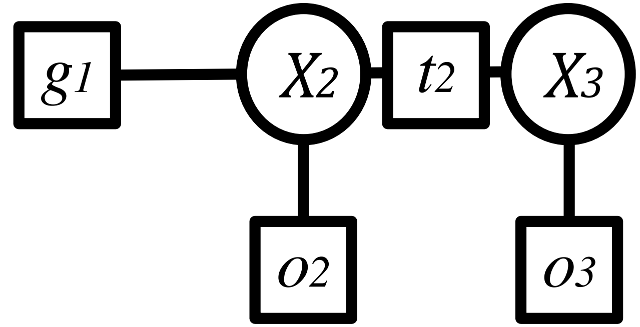
- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$



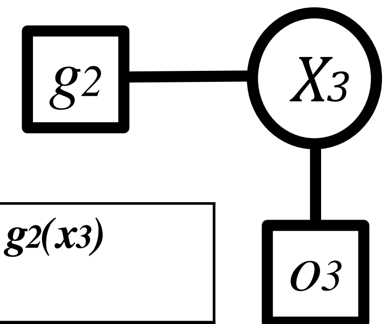
- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$

x3	***x2***	***g1(x2)***	***o2(x2)***	***t2(x2, x3)***	***g1(x2) o2(x2) t2(x2, x3)***	***g2(x3)***
0	0	4: {*x1:0*}	0	2		
0	1	2: {*x1: 1*}	1	1		
0	2	1: {*x1: 1*}	2	0		
1	0	4: {*x1:0*}	0	1		
1	1	2: {*x1: 1*}	1	2		
1	2	1: {*x1: 1*}	2	1		
2	0	4: {*x1:0*}	0	0		
2	1	2: {*x1: 1*}	1	1		
2	2	1: {*x1: 1*}	2	2		

- Eliminate X_2
- Factors that depend on X_2:
    - o_2, t_2, g_1



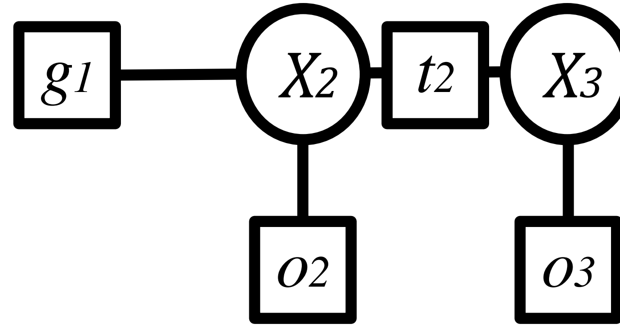
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$



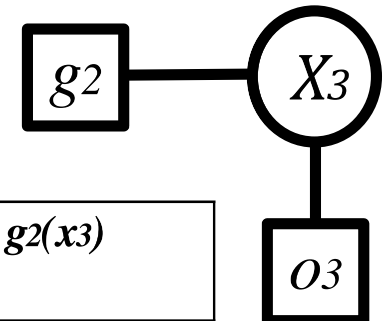
- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$

x3	***x2***	***g1(x2)***	***o2(x2)***	***t2(x2, x3)***	***g1(x2) o2(x2) t2(x2, x3)***	***g2(x3)***
0	0	4: {*x1:0*}	0	2	0	
0	1	2: {*x1: 1*}	1	1	2	
0	2	1: {*x1: 1*}	2	0	0	
1	0	4: {*x1:0*}	0	1	0	
1	1	2: {*x1: 1*}	1	2	4	
1	2	1: {*x1: 1*}	2	1	2	
2	0	4: {*x1:0*}	0	0	0	
2	1	2: {*x1: 1*}	1	1	2	
2	2	1: {*x1: 1*}	2	2	4	

- Eliminate X_2
- Factors that depend on X_2:
  - o_2, t_2, g_1
- Add $f_{new}(x) = \max_{x_i} \prod_{j=1}^{k} f_j(x)$

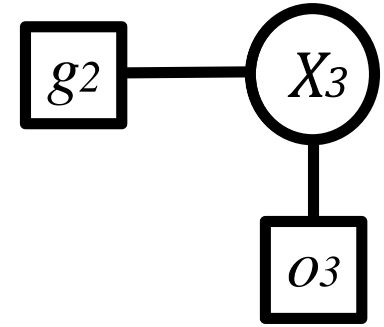


- $g_2(x_3) = \max_{x_2 \in \{0,1,2\}} g_1(x_2) \cdot o_2(x_2) \cdot t_2(x_2, x_3)$



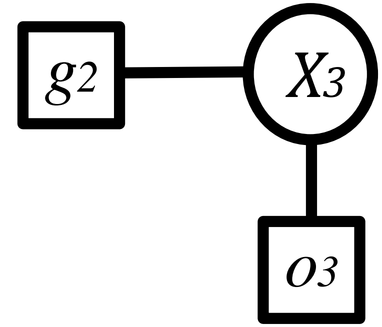
x_3	x_2	$g_1(x_2)$	$o_2(x_2)$	$t_2(x_2, x_3)$	$g_1(x_2)\ o_2(x_2)\ t_2(x_2, x_3)$	$g_2(x_3)$
0	0	4: $\{x_1{:}0\}$	0	2	0	2: $\{x_1 : 1, x_2 : 1\}$
0	1	2: $\{x_1{:}\ 1\}$	1	1	2	
0	2	1: $\{x_1{:}\ 1\}$	2	0	0	
1	0	4: $\{x_1{:}0\}$	0	1	0	4: $\{x_1 : 1, x_2 : 1\}$
1	1	2: $\{x_1{:}\ 1\}$	1	2	4	
1	2	1: $\{x_1{:}\ 1\}$	2	1	2	
2	0	4: $\{x_1{:}0\}$	0	0	0	4: $\{x_1 : 1, x_2 : 2\}$
2	1	2: $\{x_1{:}\ 1\}$	1	1	2	
2	2	1: $\{x_1{:}\ 1\}$	2	2	4	

- We are left with:



$$\max_{x_3 \in \{0,1,2\}} g_2(x_3) \cdot o_3(x_3)$$

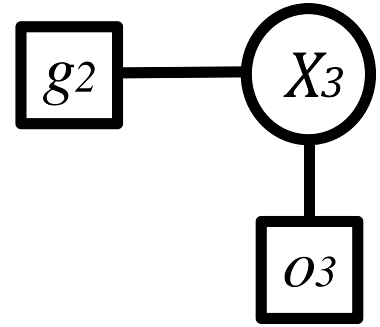
- We are left with:



$$\max_{x_3 \in \{0,1,2\}} g_2(x_3) \cdot o_3(x_3)$$

x3	*g2(x3)*	*o3(x3)*	*g2(x3) o3(x3)*	***Optimal Weight***
0				
1				
2				

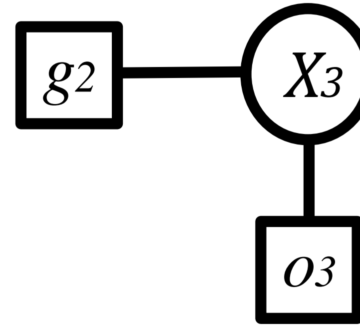
- We are left with:



$$\max_{x_3 \in \{0,1,2\}} g_2(x_3) \cdot o_3(x_3)$$

x3	*g2(x3)*	*o3(x3)*	*g2(x3) o3(x3)*	*Optimal Weight*
0	2: $\{x_1 : 1, x_2 : 1\}$	0		
1	4: $\{x_1 : 1, x_2 : 1\}$	1		
2	4: $\{x_1 : 1, x_2 : 2\}$	2		

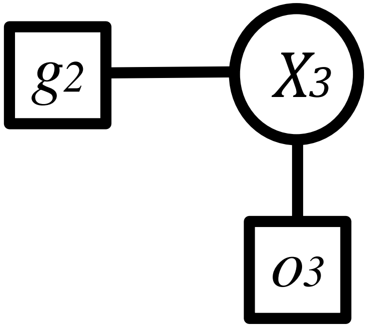
- We are left with:



$$\max_{x_3 \in \{0,1,2\}} g_2(x_3) \cdot o_3(x_3)$$

x3	***g2(x3)***	***o3(x3)***	***g2(x3) o3(x3)***	***Optimal Weight***
0	2: $\{x_1 : 1, x_2 : 1\}$	0	0	
1	4: $\{x_1 : 1, x_2 : 1\}$	1	4	
2	4: $\{x_1 : 1, x_2 : 2\}$	2	8	

- We are left with:



$$\max_{x_3 \in \{0,1,2\}} g_2(x_3) \cdot o_3(x_3)$$

x_3	$g_2(x_3)$	$o_3(x_3)$	$g_2(x_3)\ o_3(x_3)$	***Optimal Weight***
0	2: $\{x_1 : 1, x_2 : 1\}$	0	0	8: $\{x_1 : 1, x_2 : 2, x_3 : 2\}$
1	4: $\{x_1 : 1, x_2 : 1\}$	1	4	
2	4: $\{x_1 : 1, x_2 : 2\}$	2	8	