Exercises: Artificial Intelligence

Planning & Logic: Blocks world

Planning & Logic: Blocks world

PROBLEM

Problem

- **Table:** T & **Blocks:** A,B,C,D
- **<u>Apply STRIPS</u>** on:
  - **Initial State, I:**
    - clear(A), clear(B), clear(C), clear(D), on(A,T), on(B,T), on(C,T), on(D,T)
  - **Final State, F:**
    - on(A,T), on(B,A), on(C,B), on(D,C), clear(D)
- **Indicate:** *Establish & Threaten*
- **Give:** *Before relation without loops*

Planning & Logic: Blocks world

INITIAL & FINAL STATE

Initial & Final State

If	
Add	on(A,T) on(B,T) on(C,T) on(D,T) clear(A) clear(B) clear(C) clear(D)
Del	

If	on(A,T) on(B,A) on(C,B) on(D,C) clear(D)
Add	
Del	

Planning & Logic: Blocks world

OPERATORS

Operators

Operator 1	
If	on(x,y)
	clear(x)
	clear(z)
Add	on(x,z)
	clear(y)
Del	on(x,y)
	clear(z)

Operator 2	
If	on(x,y)
	clear(x)
Add	on(x,T)
	clear(y)
Del	on(x,y)

Operator 3	
If	on(x,T)
	clear(x)
	clear(z)
Add	on(x,z)
Del	on(x,T)
	clear(z)

Actual operators are ground instances!

Planning & Logic: Blocks world

STRIPS

STRIPS

I	
If	
Add	on(A,T) on(B,T) on(C,T) on(D,T) clear(A) clear(B) clear(C) clear(D)
Del	

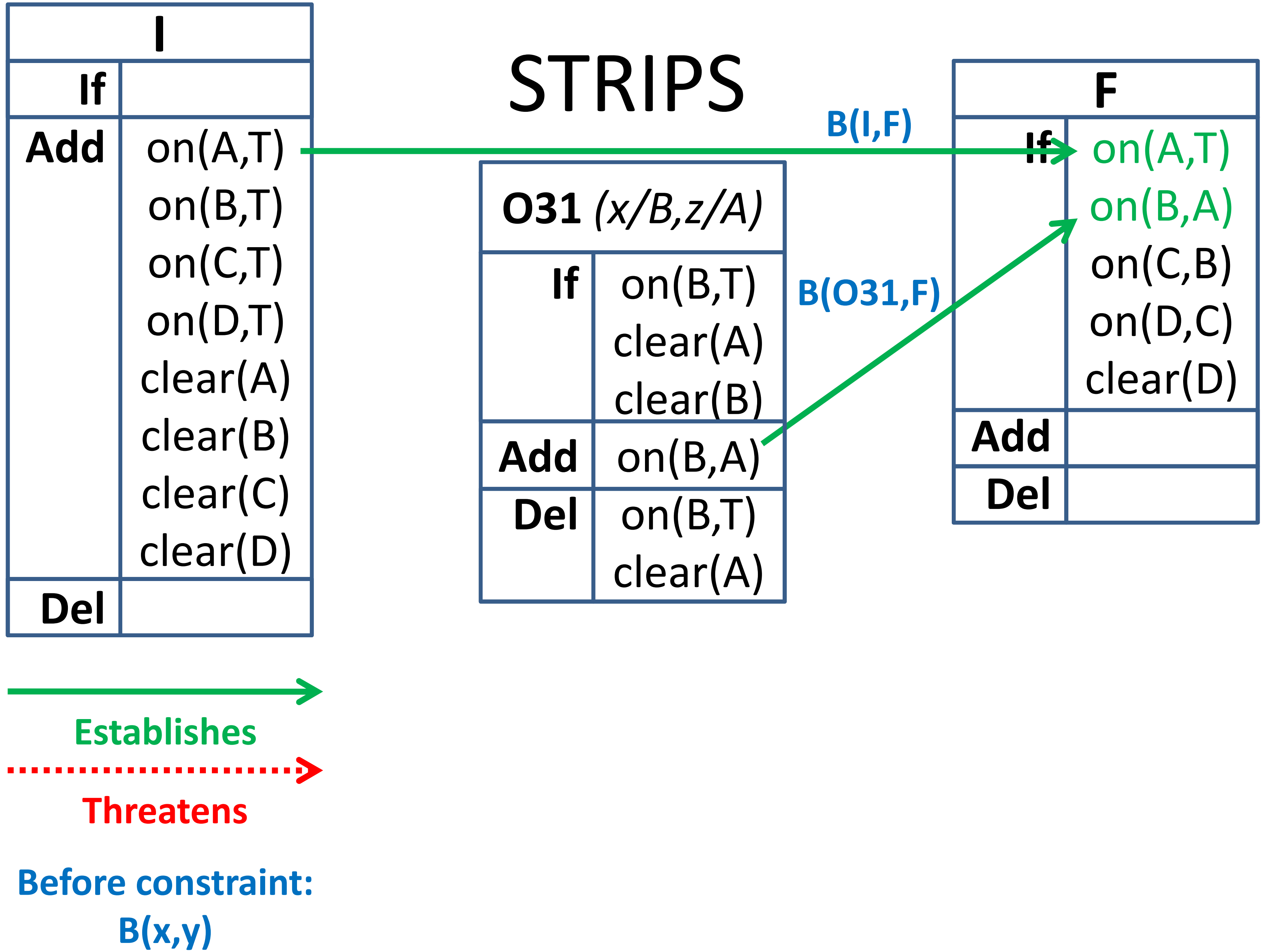
$\xrightarrow{B(I,F)}$

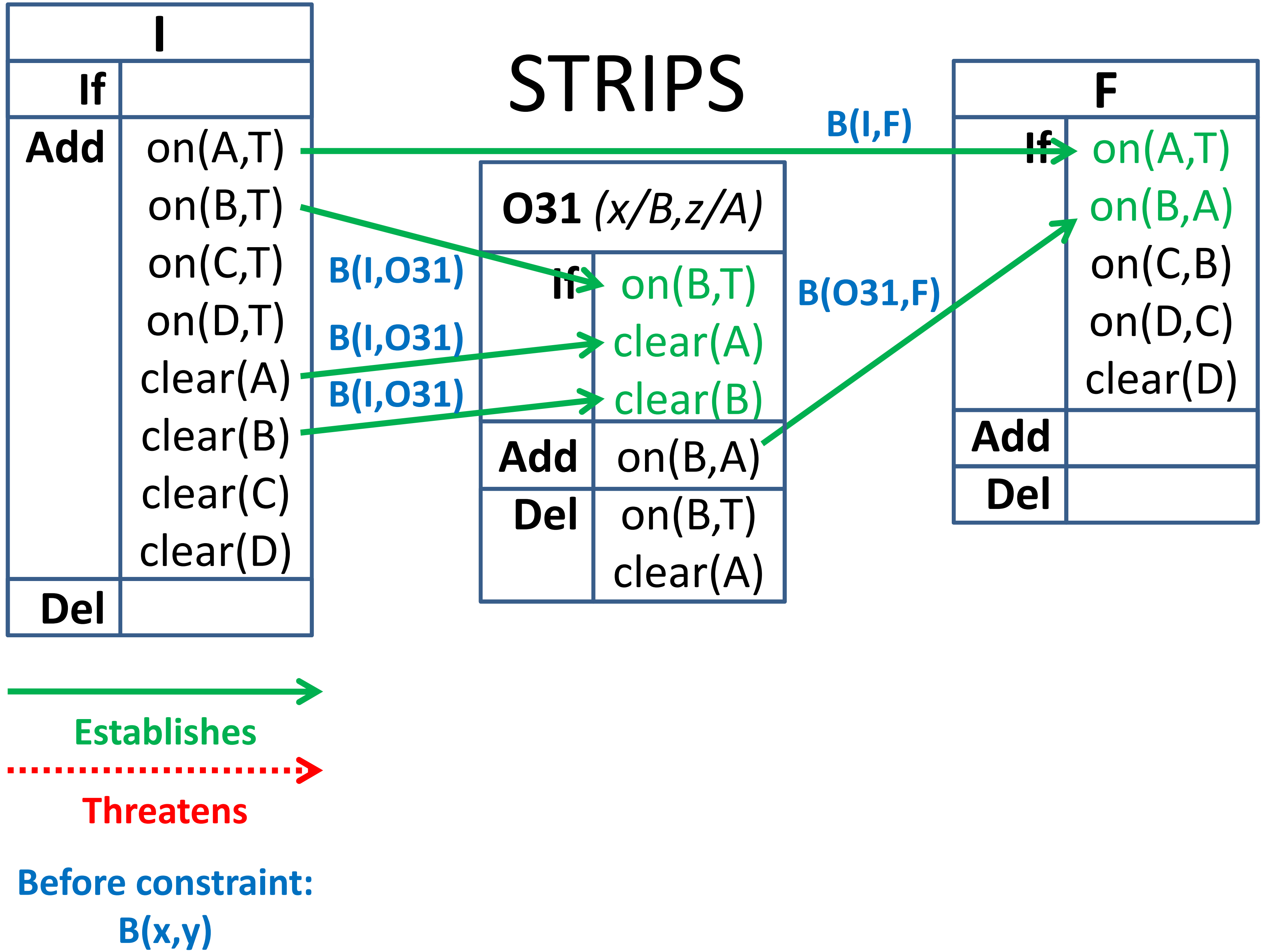
F	
If	on(A,T) on(B,A) on(C,B) on(D,C) clear(D)
Add	
Del	

$\longrightarrow$ Establishes

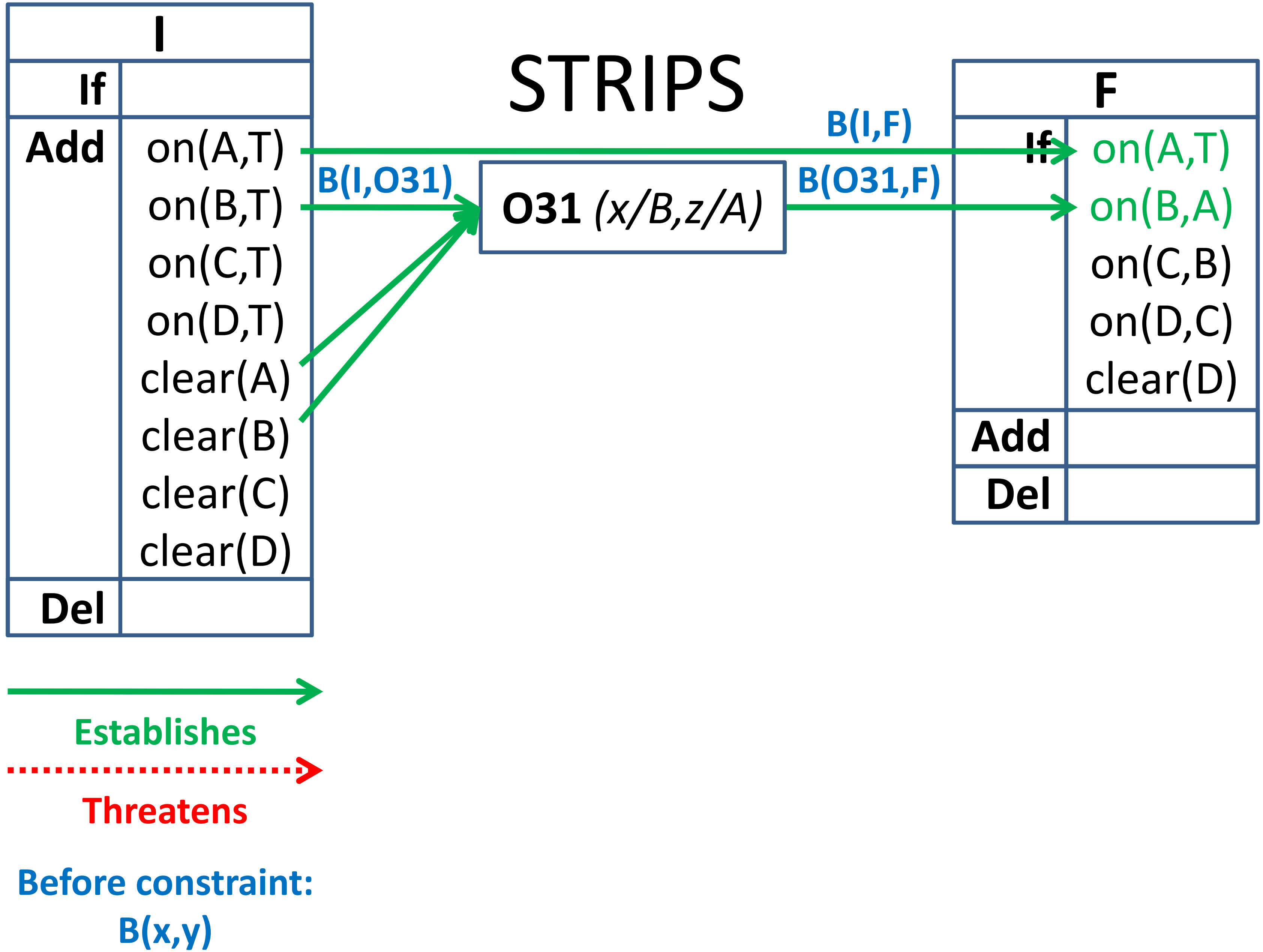
$\dashrightarrow$ Threatens

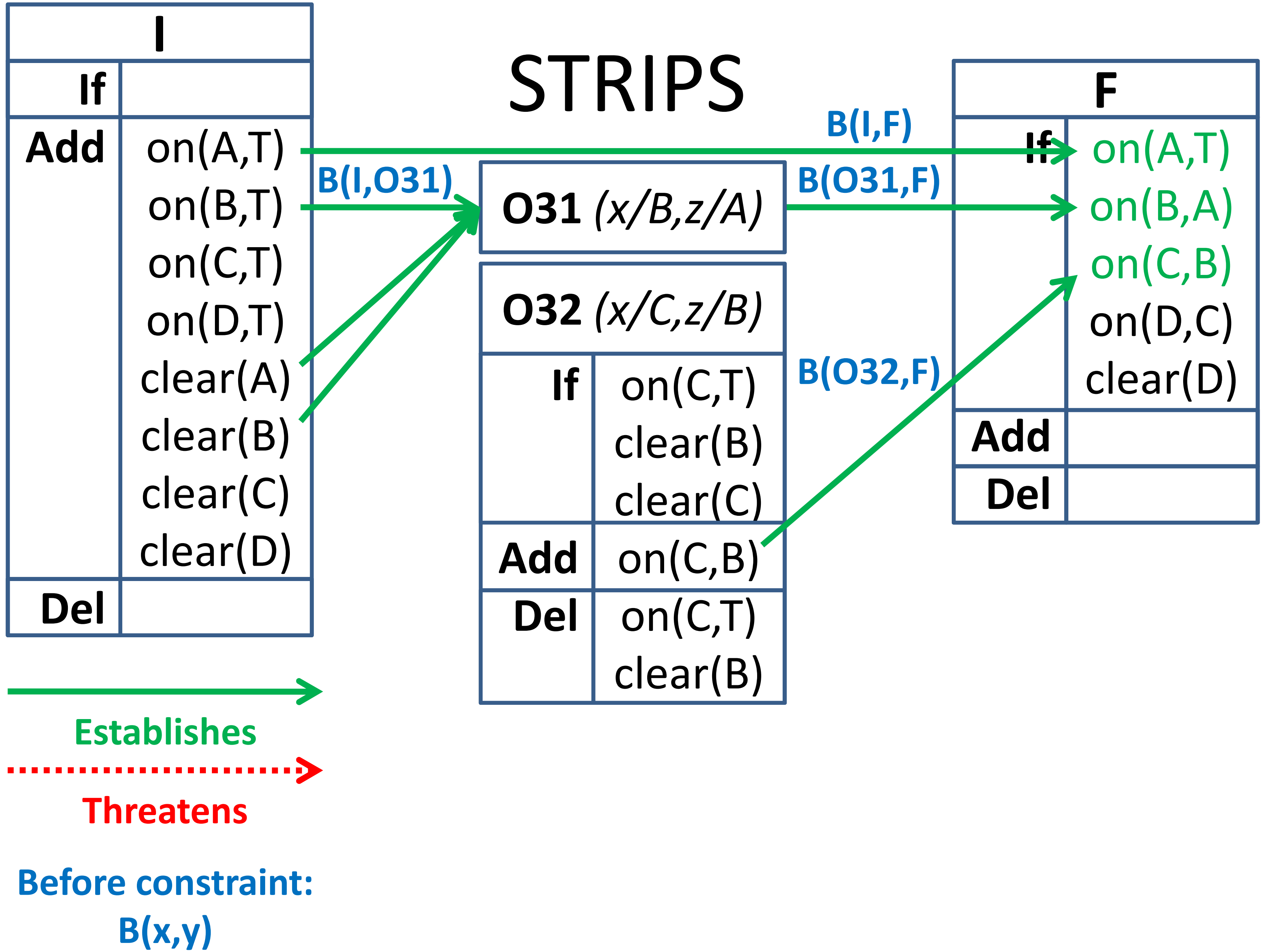
Before constraint:
B(x,y)


STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
O31 (x/B,z/A)
If
on(B,T)
clear(A)
clear(B)
Add
on(B,A)
Del
on(B,T)
clear(A)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
B(I,F)
B(O31,F)
Establishes
Threatens
Before constraint:
B(x,y)

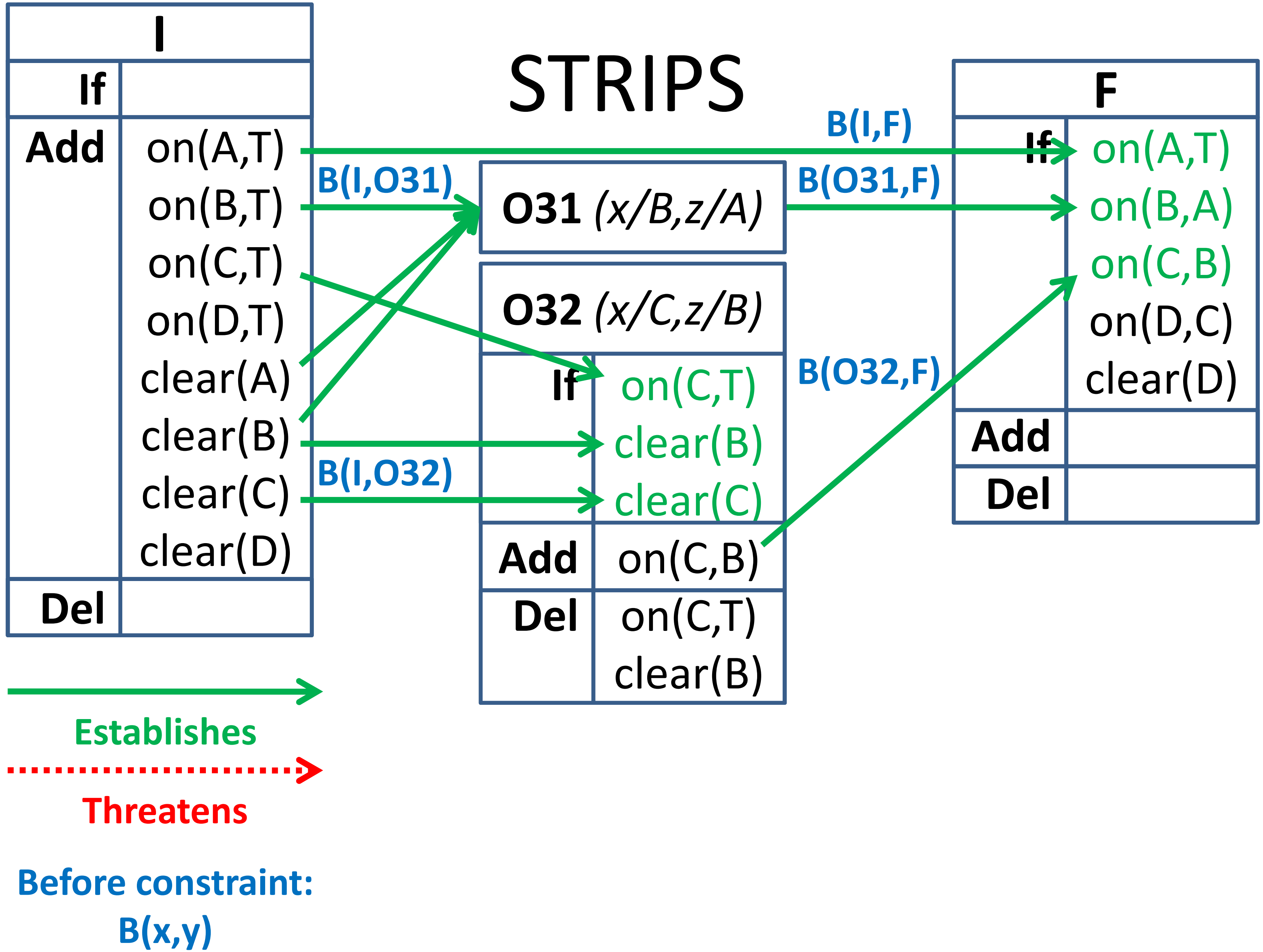


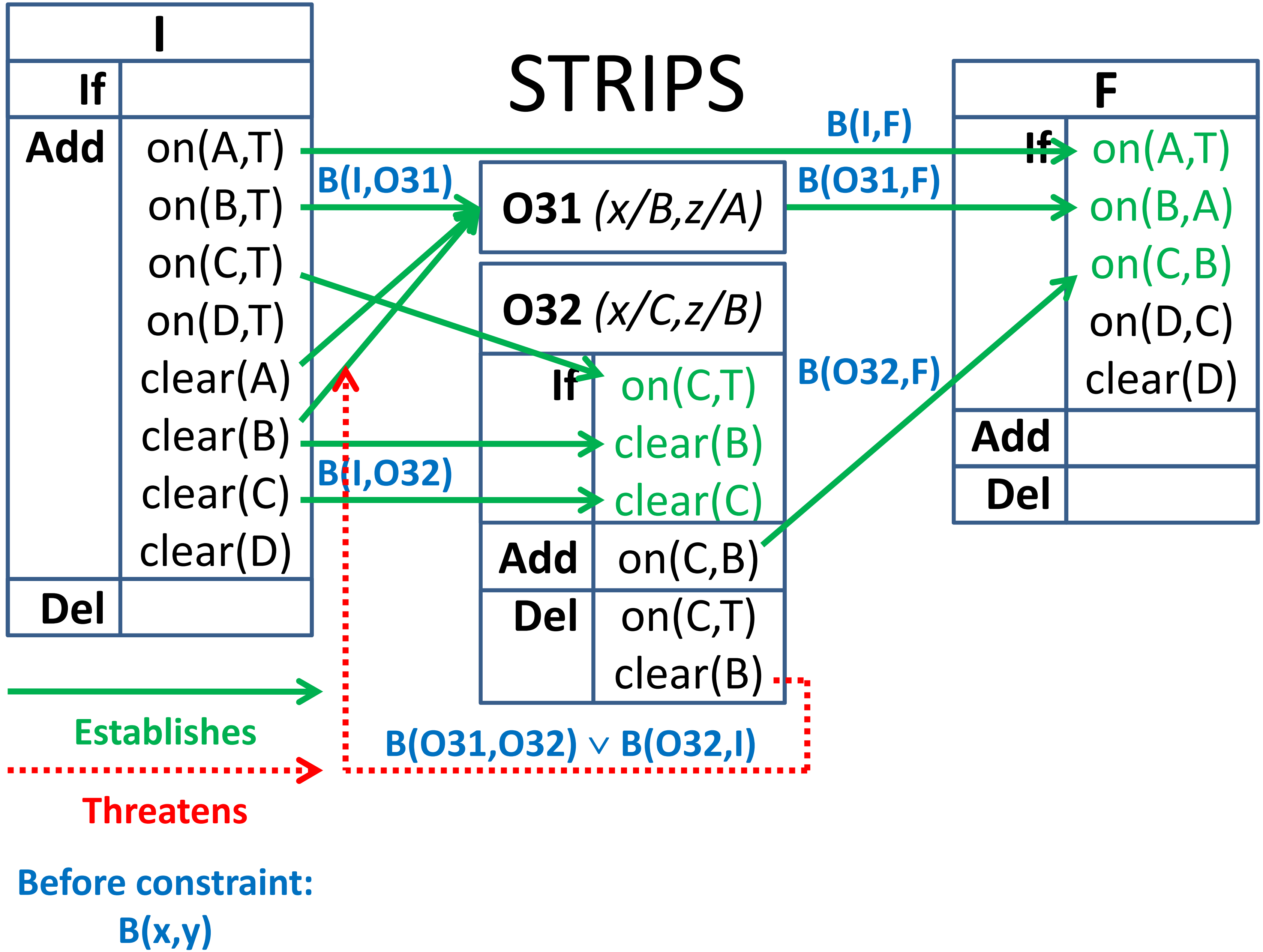
STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
O31 (x/B,z/A)
If
on(B,T)
clear(A)
clear(B)
Add
on(B,A)
Del
on(B,T)
clear(A)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
B(I,F)
B(I,O31)
B(I,O31)
B(I,O31)
B(O31,F)
Establishes
Threatens
Before constraint:
B(x,y)

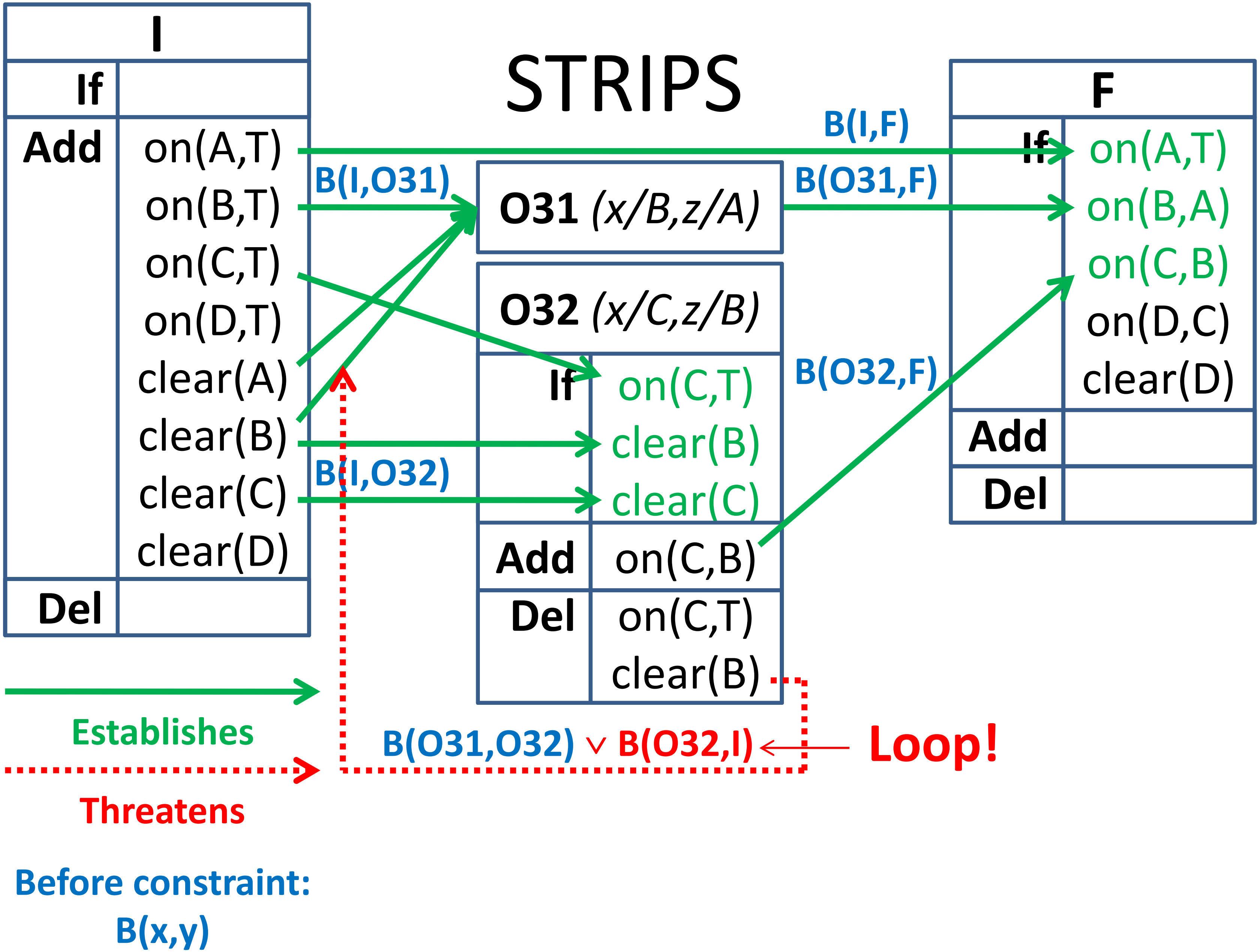

I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
STRIPS
B(I,F)
B(I,O31)
O31 (x/B,z/A)
B(O31,F)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

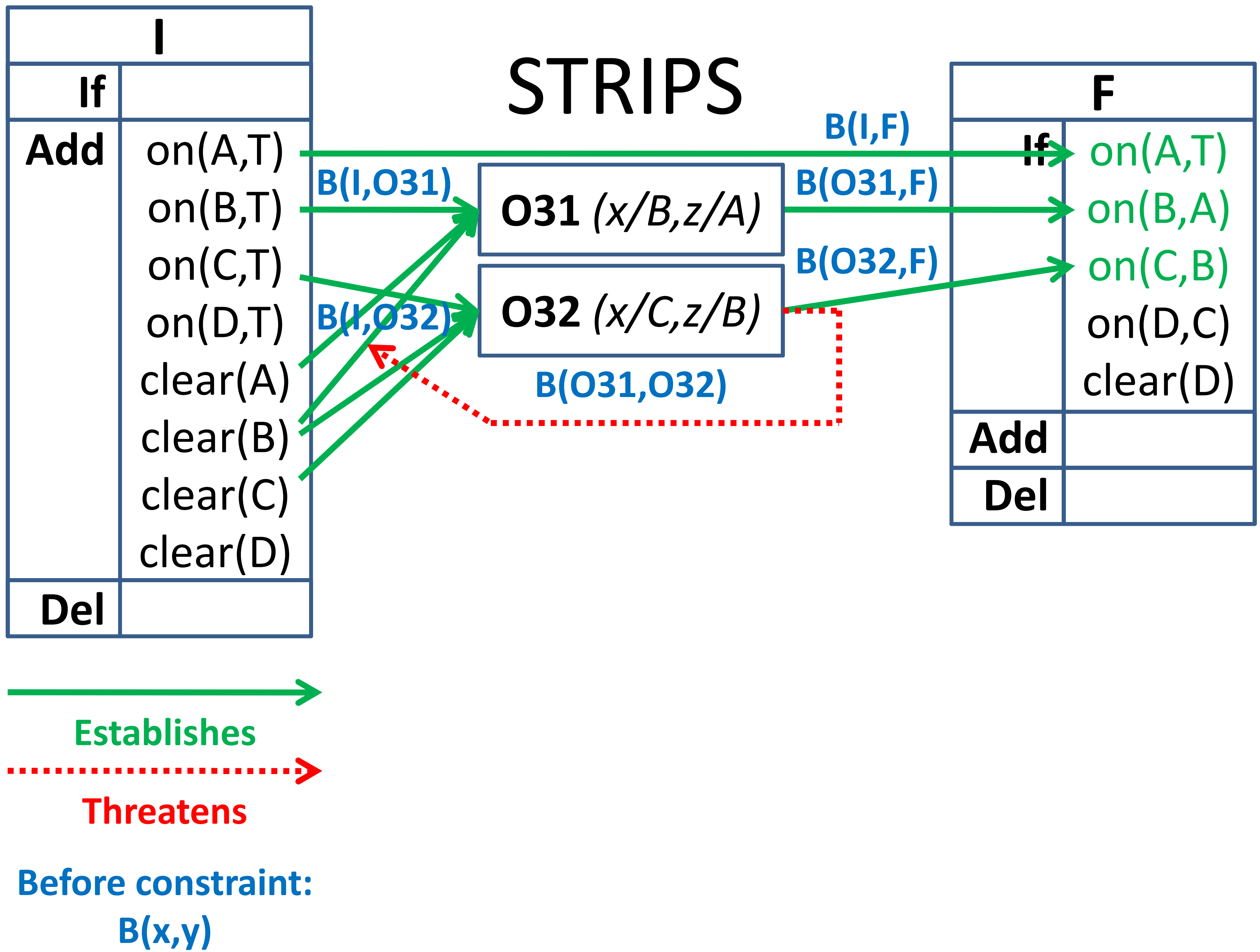


STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,O31)
O31 (x/B,z/A)
O32 (x/C,z/B)
If
on(C,T)
clear(B)
clear(C)
Add
on(C,B)
Del
on(C,T)
clear(B)
B(I,F)
B(O31,F)
B(O32,F)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

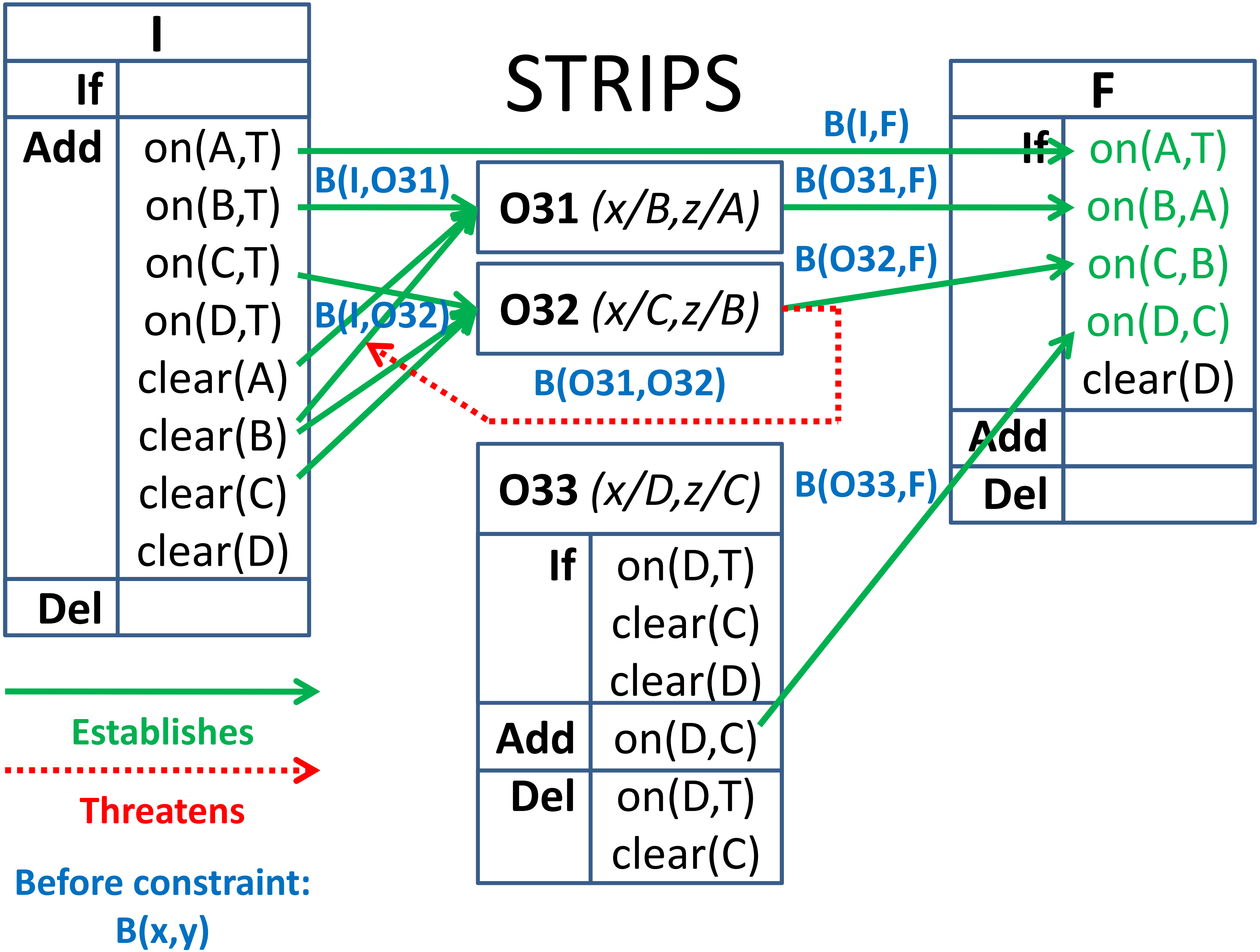

STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,O31)
B(I,O32)
O31 (x/B,z/A)
O32 (x/C,z/B)
If
on(C,T)
clear(B)
clear(C)
Add
on(C,B)
Del
on(C,T)
clear(B)
B(I,F)
B(O31,F)
B(O32,F)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

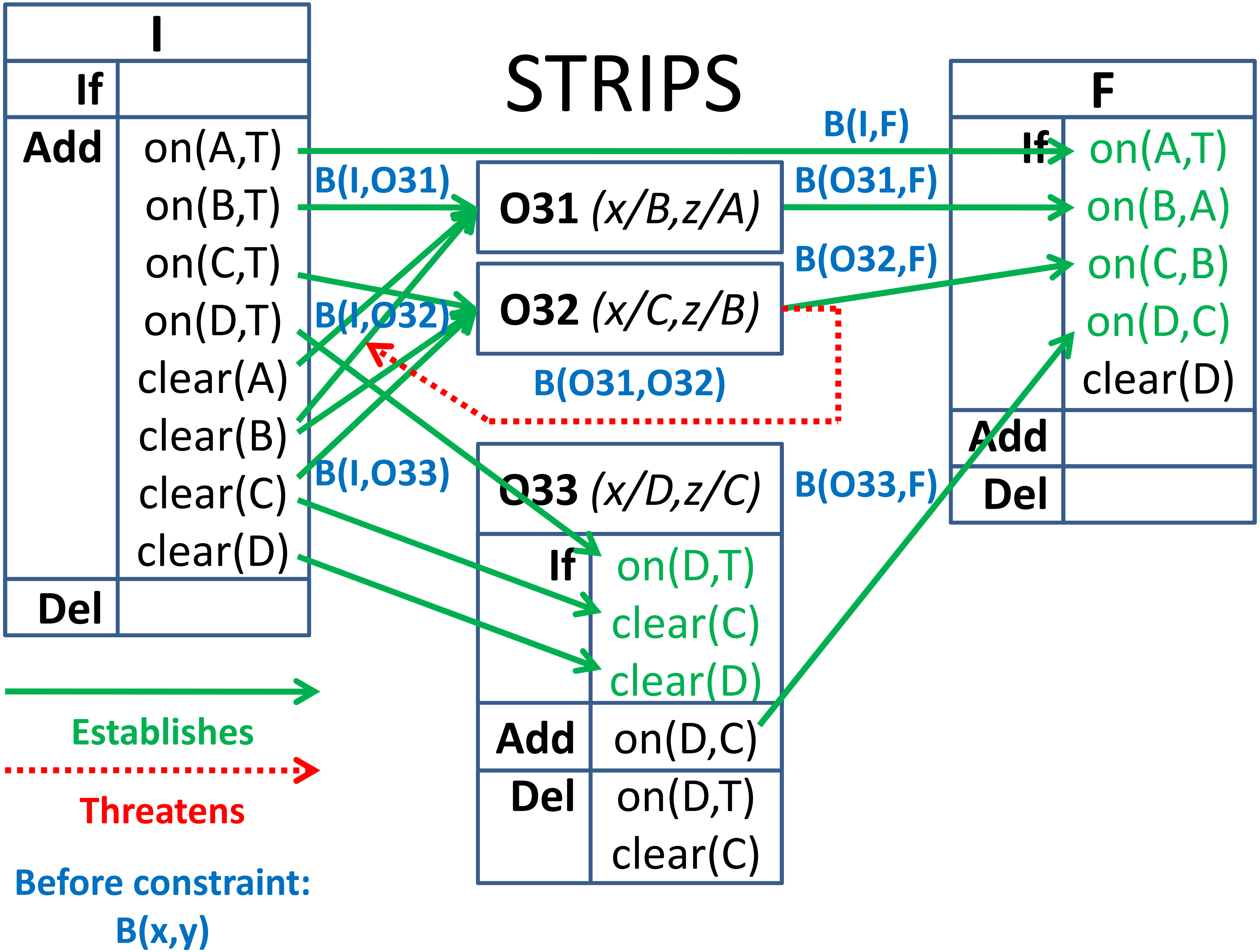

STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,F)
B(I,O31)
O31 (x/B,z/A)
B(O31,F)
O32 (x/C,z/B)
If
on(C,T)
clear(B)
clear(C)
Add
on(C,B)
Del
on(C,T)
clear(B)
B(I,O32)
B(O32,F)
B(O31,O32) ∨ B(O32,I)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

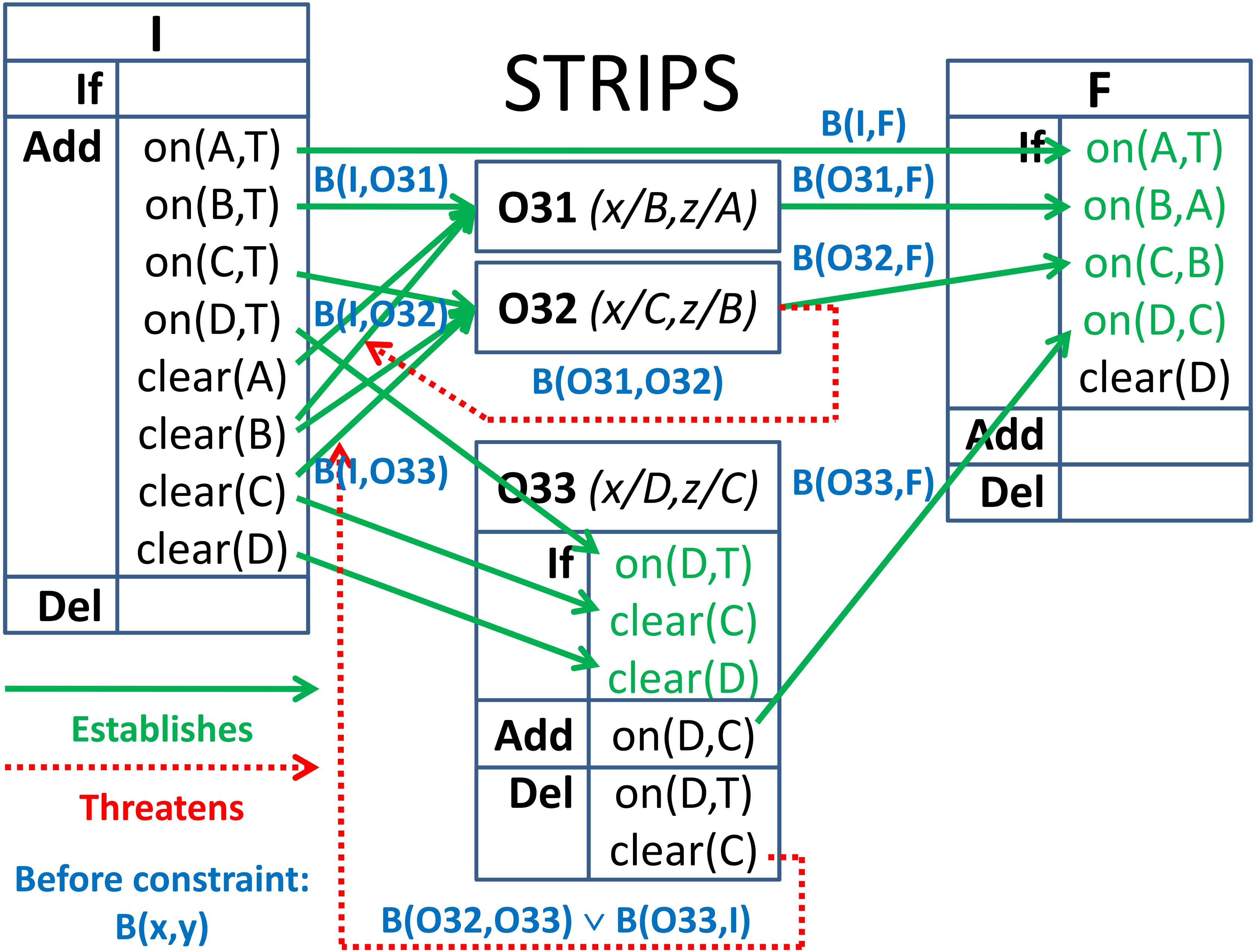

STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,F)
B(I,O31)
O31 (x/B,z/A)
B(O31,F)
O32 (x/C,z/B)
If
on(C,T)
clear(B)
clear(C)
Add
on(C,B)
Del
on(C,T)
clear(B)
B(O32,F)
B(I,O32)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
B(O31,O32) ∨ B(O32,I)
Loop!
Establishes
Threatens
Before constraint:
B(x,y)

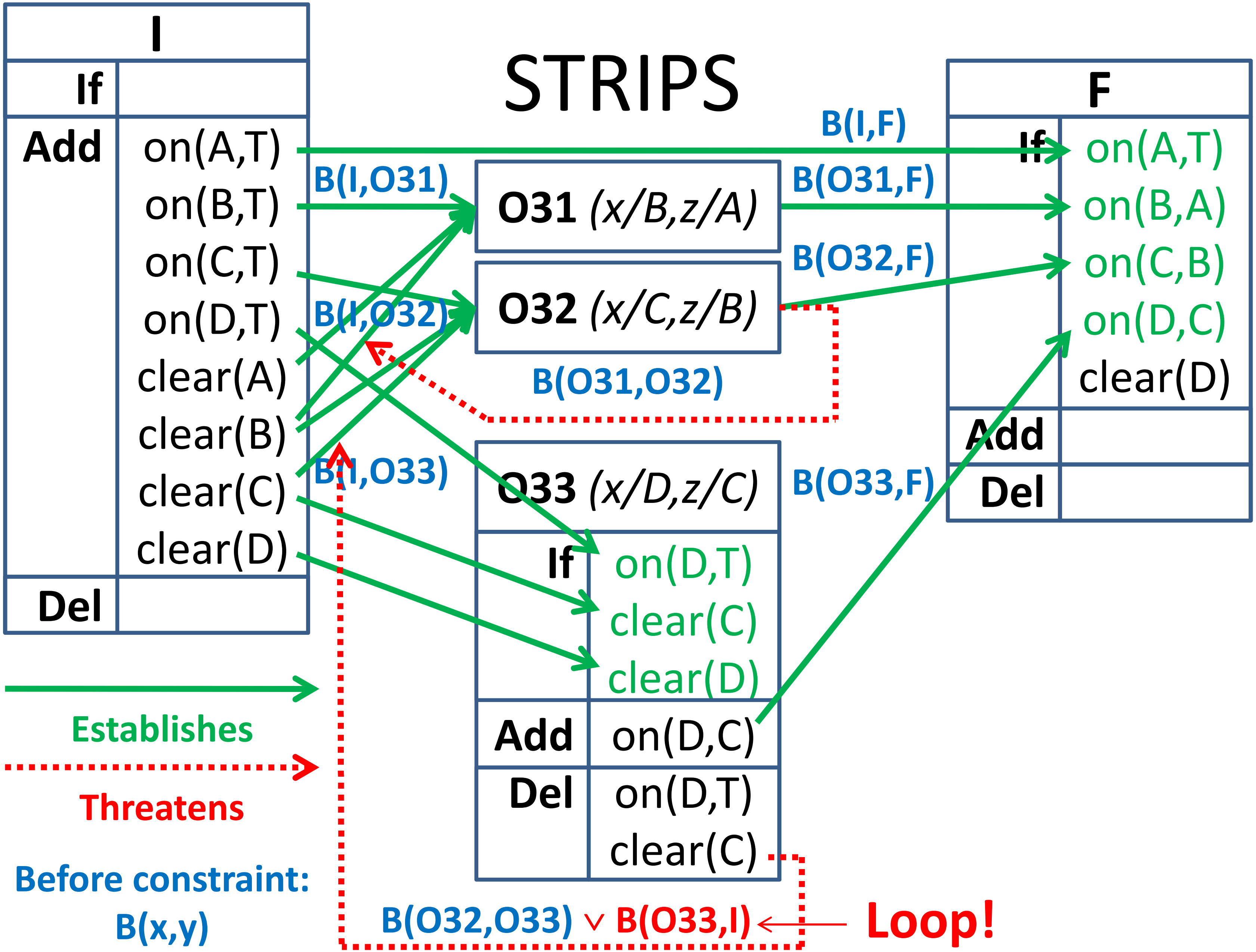


I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
STRIPS
B(I,F)
B(I,O31)
O31 (x/B,z/A)
B(O31,F)
B(O32,F)
B(I,O32)
O32 (x/C,z/B)
B(O31,O32)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

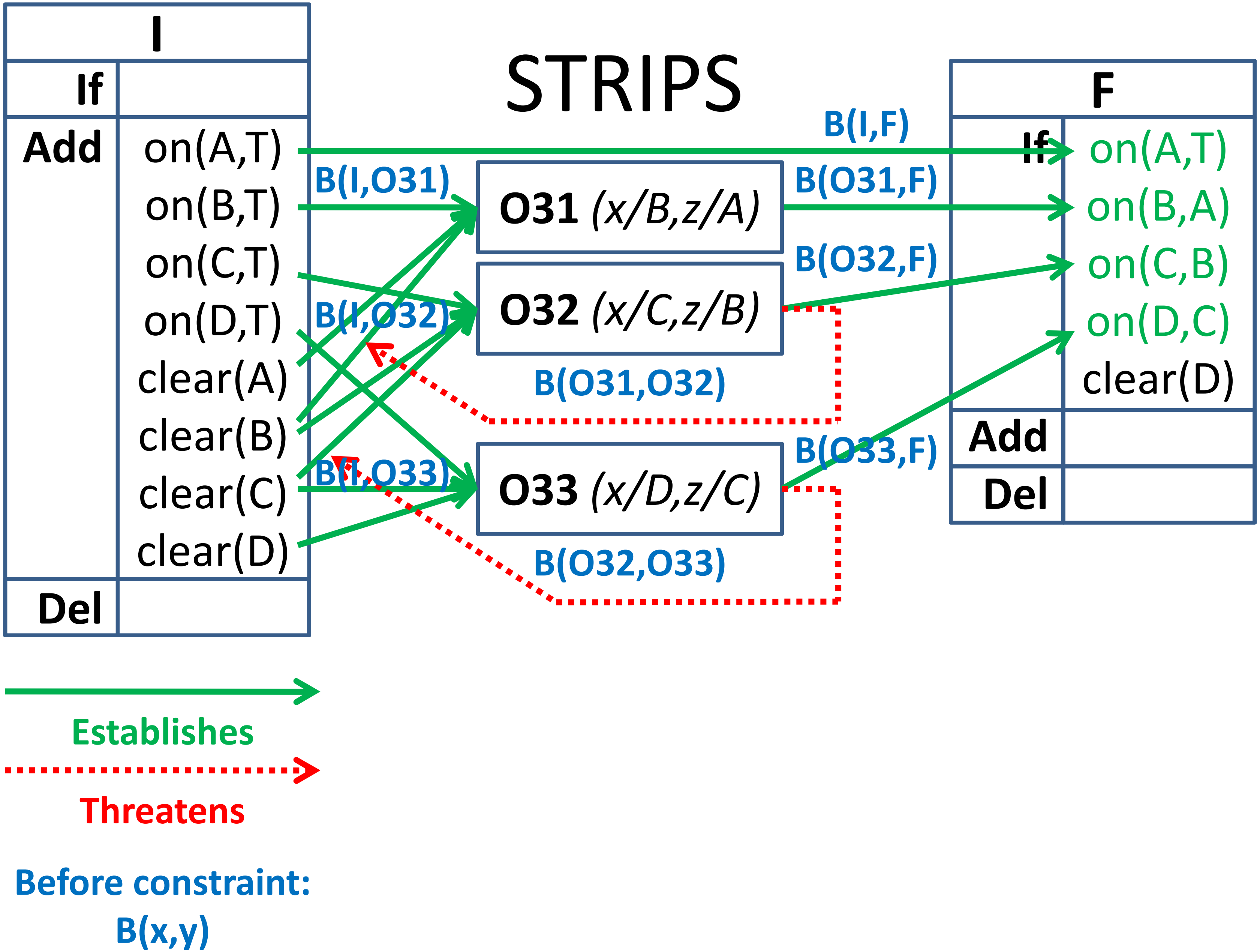

STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,O31)
B(I,O32)
O31 (x/B,z/A)
O32 (x/C,z/B)
B(O31,O32)
O33 (x/D,z/C)
If
on(D,T)
clear(C)
clear(D)
Add
on(D,C)
Del
on(D,T)
clear(C)
B(I,F)
B(O31,F)
B(O32,F)
B(O33,F)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)


STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
O31 (x/B,z/A)
O32 (x/C,z/B)
O33 (x/D,z/C)
If
on(D,T)
clear(C)
clear(D)
Add
on(D,C)
Del
on(D,T)
clear(C)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
B(I,F)
B(I,O31)
B(O31,F)
B(O32,F)
B(I,O32)
B(O31,O32)
B(I,O33)
B(O33,F)
Establishes
Threatens
Before constraint:
B(x,y)

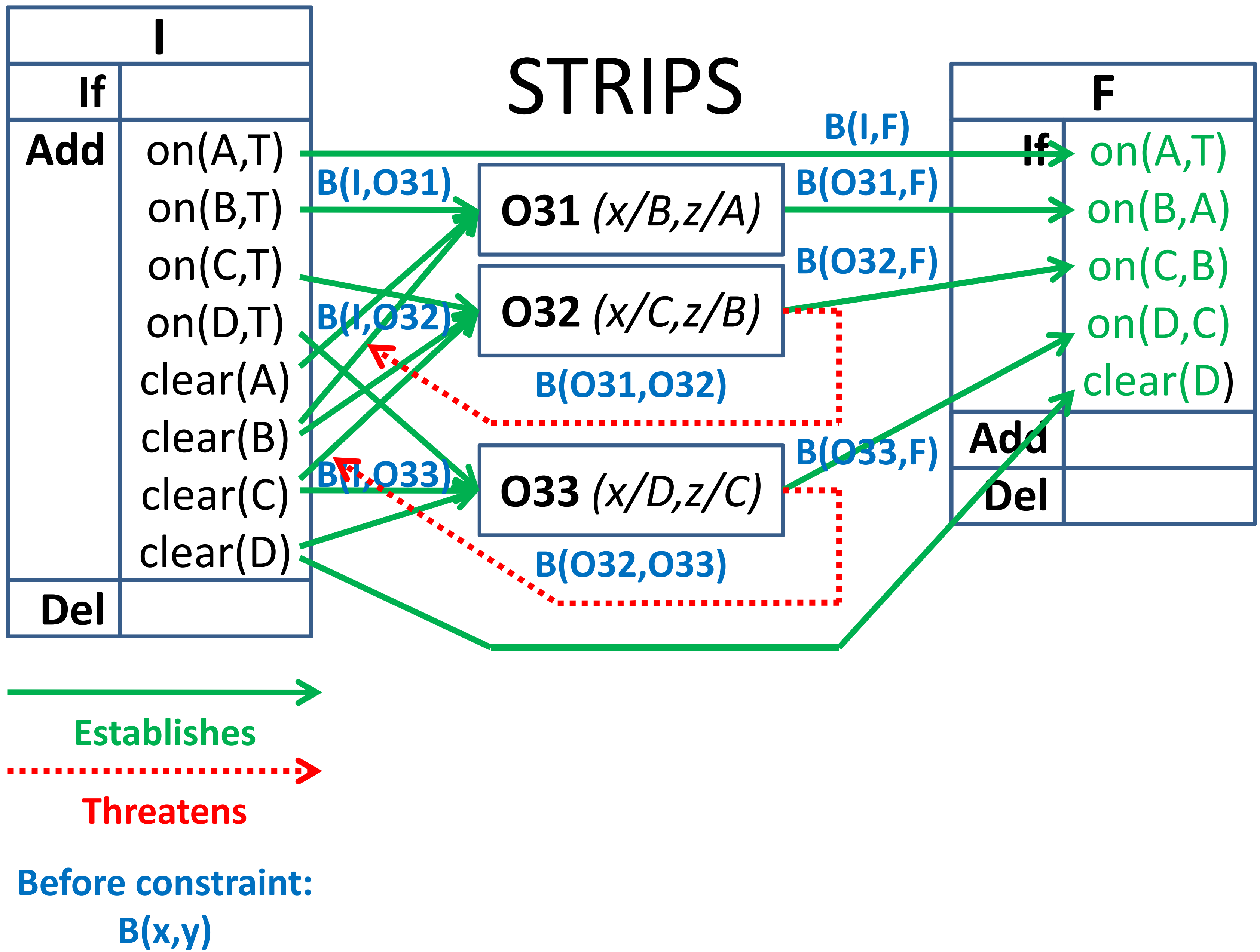

STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,F)
B(I,O31)
B(O31,F)
O31 (x/B,z/A)
B(O32,F)
B(I,O32)
O32 (x/C,z/B)
B(O31,O32)
B(I,O33)
O33 (x/D,z/C)
B(O33,F)
If
on(D,T)
clear(C)
clear(D)
Add
on(D,C)
Del
on(D,T)
clear(C)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)
B(O32,O33) ∨ B(O33,I)

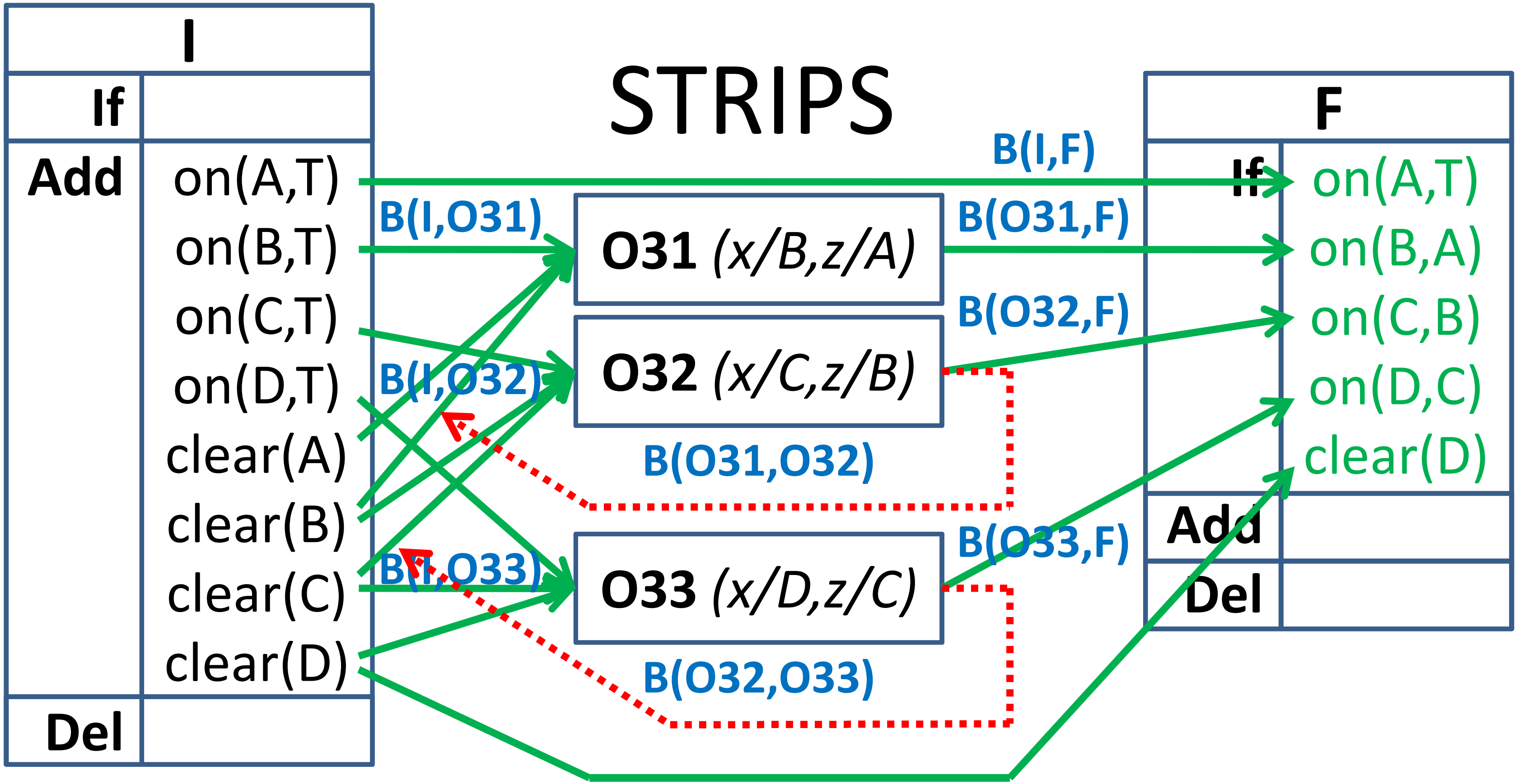


STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
B(I,F)
B(I,O31)
B(O31,F)
O31 (x/B,z/A)
B(O32,F)
O32 (x/C,z/B)
B(I,O32)
B(O31,O32)
B(I,O33)
O33 (x/D,z/C)
B(O33,F)
If
on(D,T)
clear(C)
clear(D)
Add
on(D,C)
Del
on(D,T)
clear(C)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)
B(O32,O33) ∨ B(O33,I)
Loop!



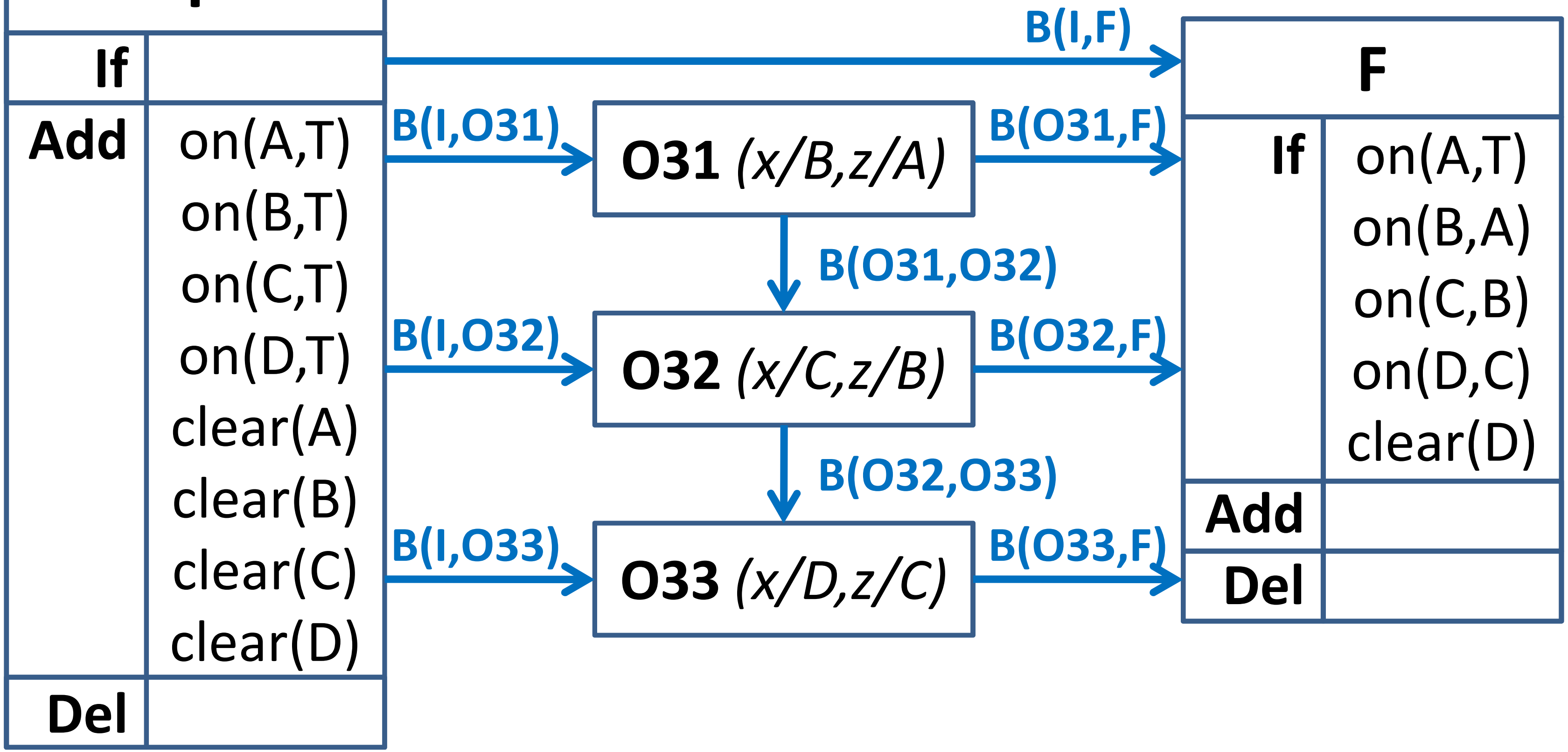
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
STRIPS
B(I,F)
B(I,O31)
B(O31,F)
O31 (x/B,z/A)
B(O32,F)
B(I,O32)
O32 (x/C,z/B)
B(O31,O32)
B(I,O33)
O33 (x/D,z/C)
B(O33,F)
B(O32,O33)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)


STRIPS
I
If
Add
on(A,T)
on(B,T)
on(C,T)
on(D,T)
clear(A)
clear(B)
clear(C)
clear(D)
Del
O31 (x/B,z/A)
O32 (x/C,z/B)
O33 (x/D,z/C)
F
If
on(A,T)
on(B,A)
on(C,B)
on(D,C)
clear(D)
Add
Del
B(I,F)
B(I,O31)
B(O31,F)
B(O32,F)
B(I,O32)
B(O31,O32)
B(I,O33)
B(O33,F)
B(O32,O33)
Establishes
Threatens
Before constraint:
B(x,y)



Are the before constraints satisfiable?

STRIPS



Are the before constraints satisfiable?

YES:

⟶ O31 ⟶ O32 ⟶ O33 ⟶

Exercises: Artificial Intelligence

Planning & Logic: Buying milk

Planning & Logic: Buying milk

PROBLEM

Problem

- **Apply STRIPS** on:
  - **Initial State, I:**
    - at(home)
  - **Final State, F:**
    - have(milk), at(home)
- **Indicate:** *Establish & Threaten*
- **Give:** *Before relation without loops (check!)*
- **Give:** *Possible linearisations*

Planning & Logic: Blocks world

INITIAL & FINAL STATE

Initial & Final State as Operator

I	
If	
Add	at(home)
Del	

F	
If	at(home) have(milk)
Add	
Del	

Planning & Logic: Blocks world

OPERATORS

Operators

Operator 1	
If	at(home)
Add	have(money)
Del	

Operator 2	
If	at(home)
Add	at(shop)
Del	at(home)

Operator 3	
If	at(shop)
Add	have(milk)
Del	

Operator 4	
If	at(shop) paid
Add	at(home)
Del	at(shop)

Operator 5	
If	at(shop) have(money) have(milk)
Add	paid
Del	have(money)

Planning & Logic: Blocks world

STRIPS

STRIPS

I	**If**	
	Add	at(home)
	Del	

Establishes

Threatens

Before constraint:
B(x,y)

F	
If	have(milk) at(home)
Add	
Del	

STRIPS

I		
	If	
	Add	at(home)
	Del	

Operator 3	
If	at(shop)
Add	have(milk)
Del	

B(O3,F)

F	
If	have(milk) at(home)
Add	
Del	

Establishes

Threatens

Before constraint:
B(x,y)

STRIPS

I		
	If	
	Add	at(home)
	Del	

Operator 2	
If	at(home)
Add	at(shop)
Del	at(home)

B(O2,O3)

Operator 3	
If	at(shop)
Add	have(milk)
Del	

B(O3,F)

F	
If	have(milk) at(home)
Add	
Del	

Establishes

Threatens

Before constraint:
B(x,y)

STRIPS

I		
	If	
	Add	at(home)
	Del	

B(I,O2)

Operator 2	
If	at(home)
Add	at(shop)
Del	at(home)

B(O2,O3)

Operator 3	
If	at(shop)
Add	have(milk)
Del	

B(O3,F)

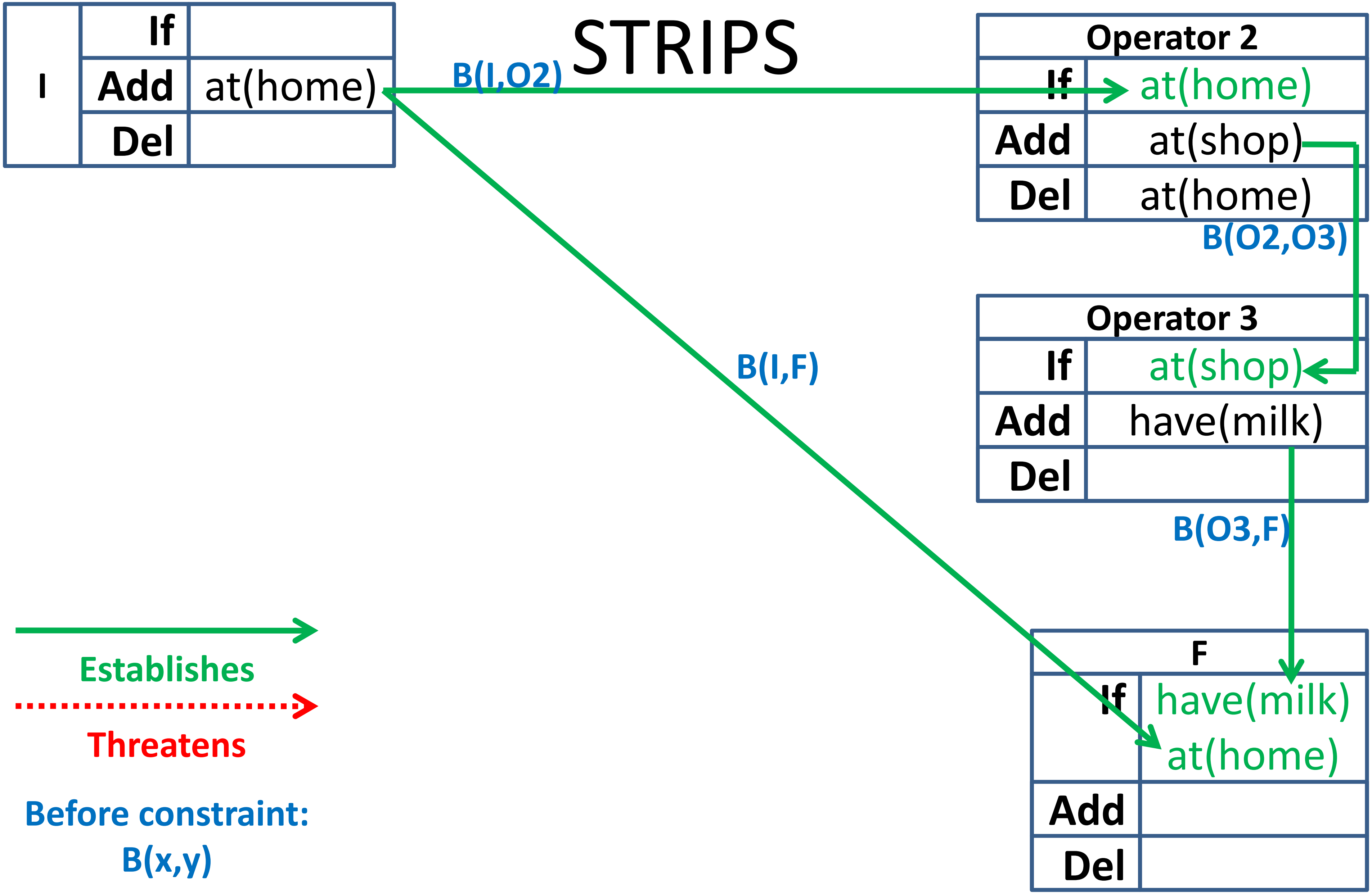
F	
If	have(milk) at(home)
Add	
Del	

Establishes

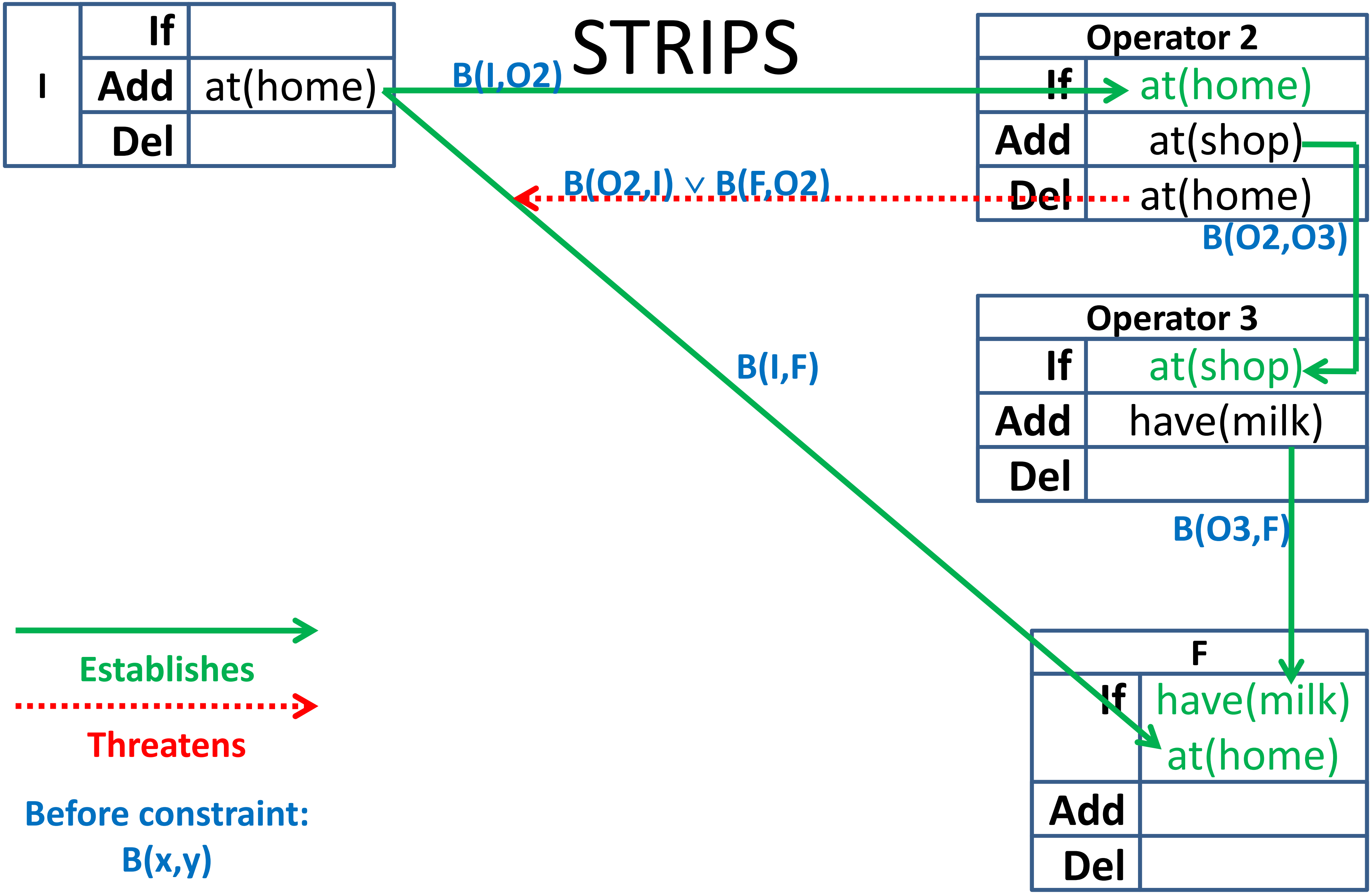
Threatens

Before constraint:
B(x,y)

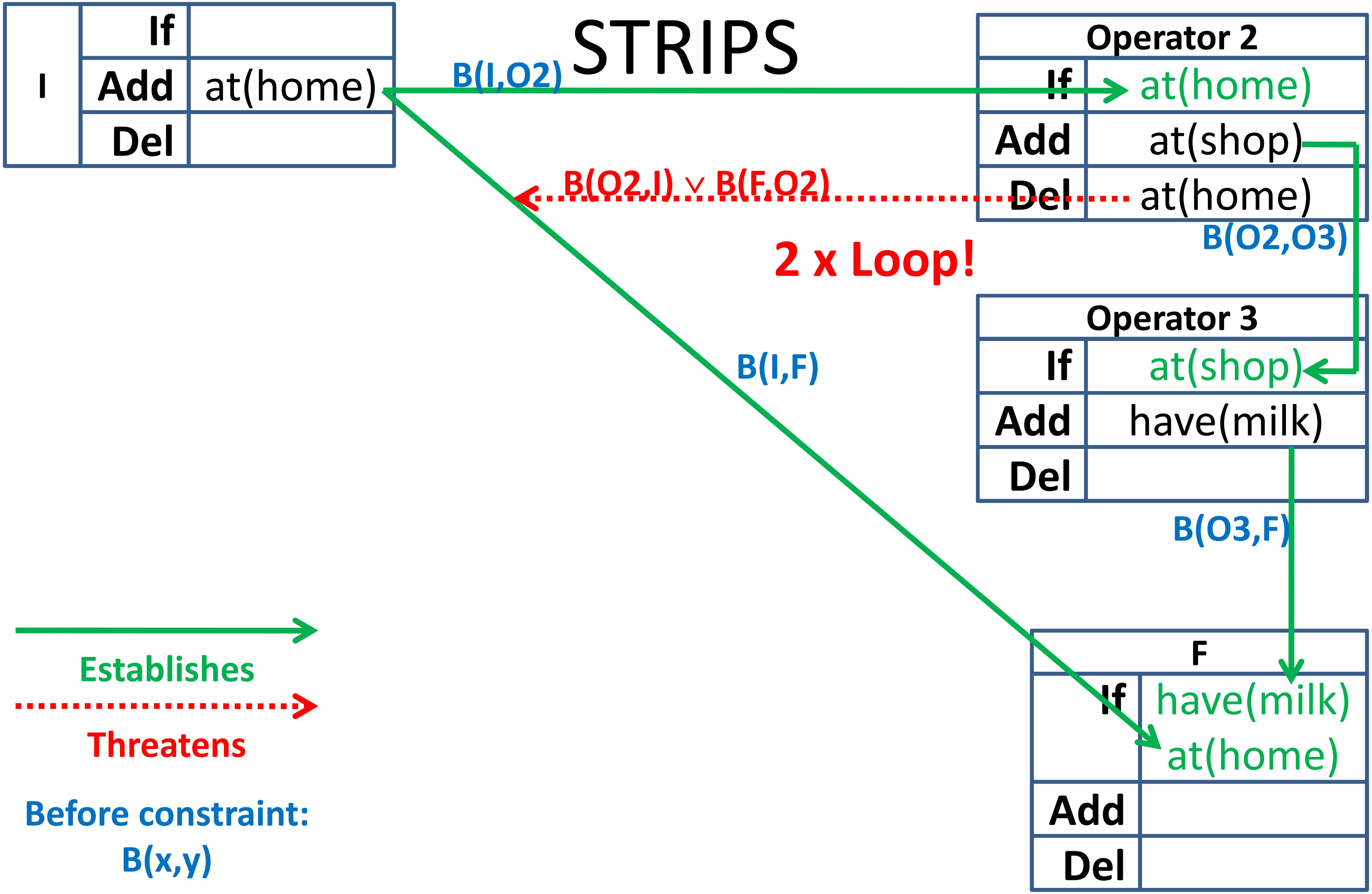
STRIPS


I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(I,F)
B(O3,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

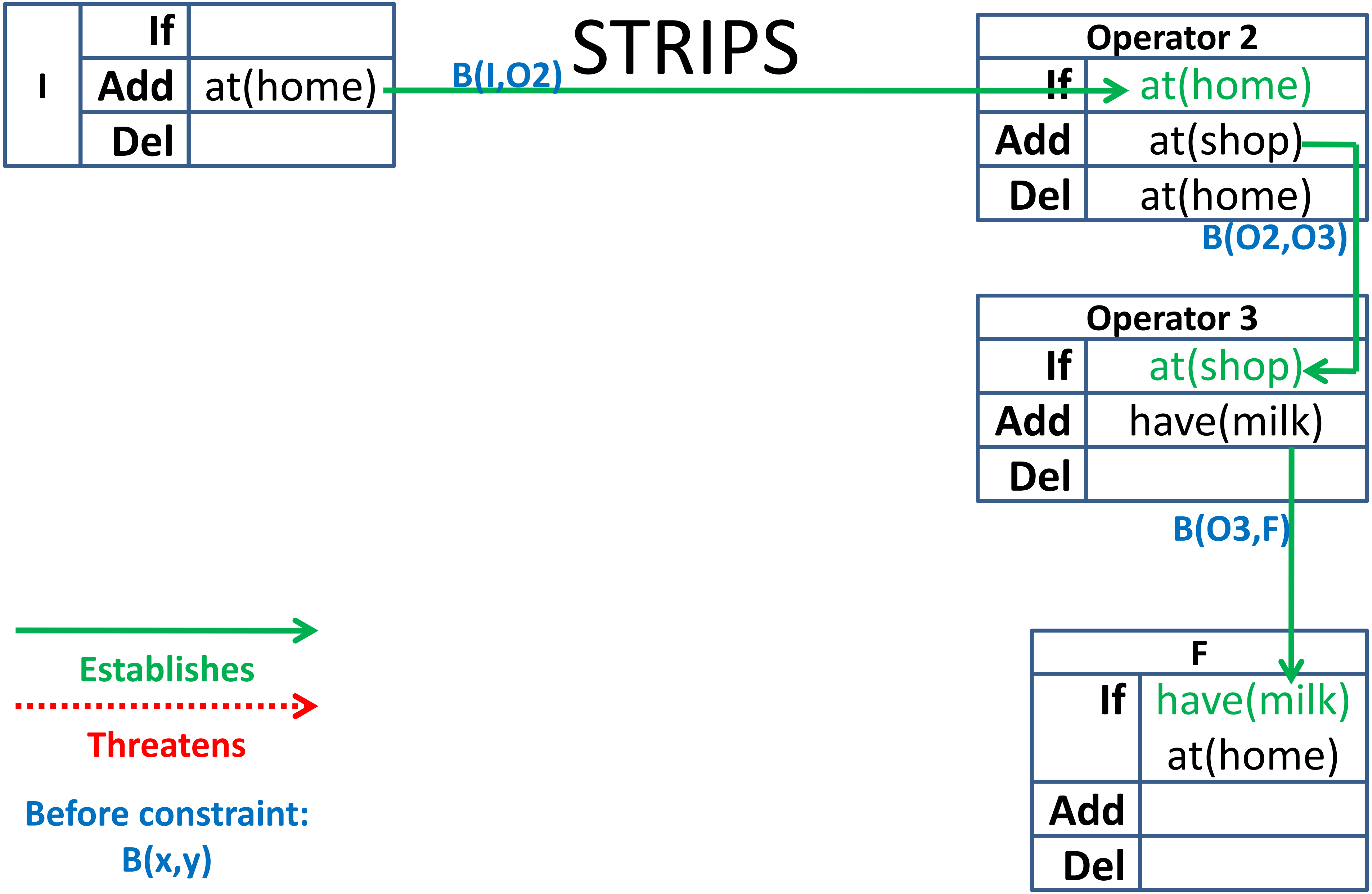
STRIPS


I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,I) ∨ B(F,O2)
B(O2,O3)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(I,F)
B(O3,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS

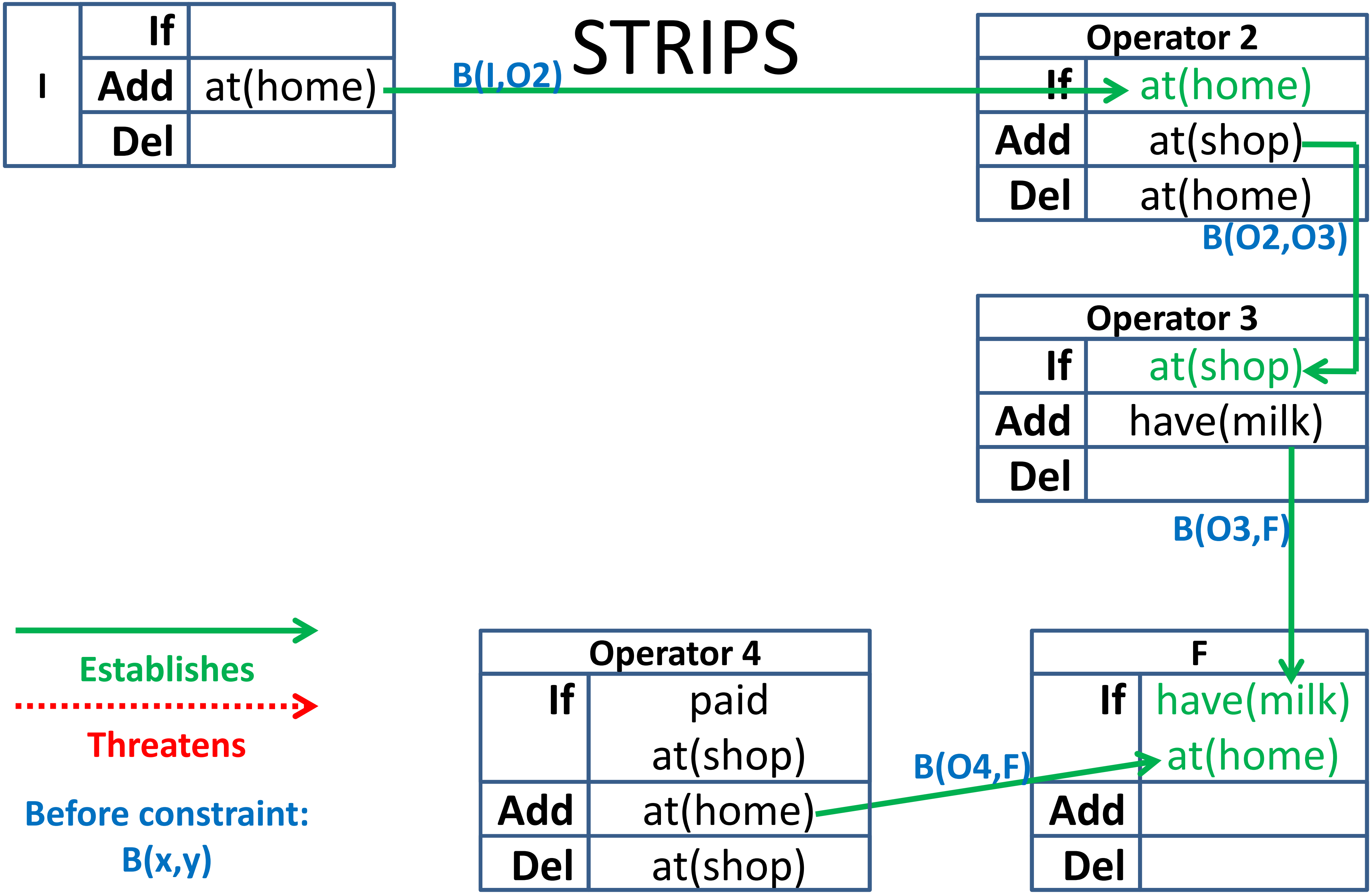

I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,I) ∨ B(F,O2)
B(O2,O3)
2 x Loop!
Operator 3
If
at(shop)
Add
have(milk)
Del
B(I,F)
B(O3,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS



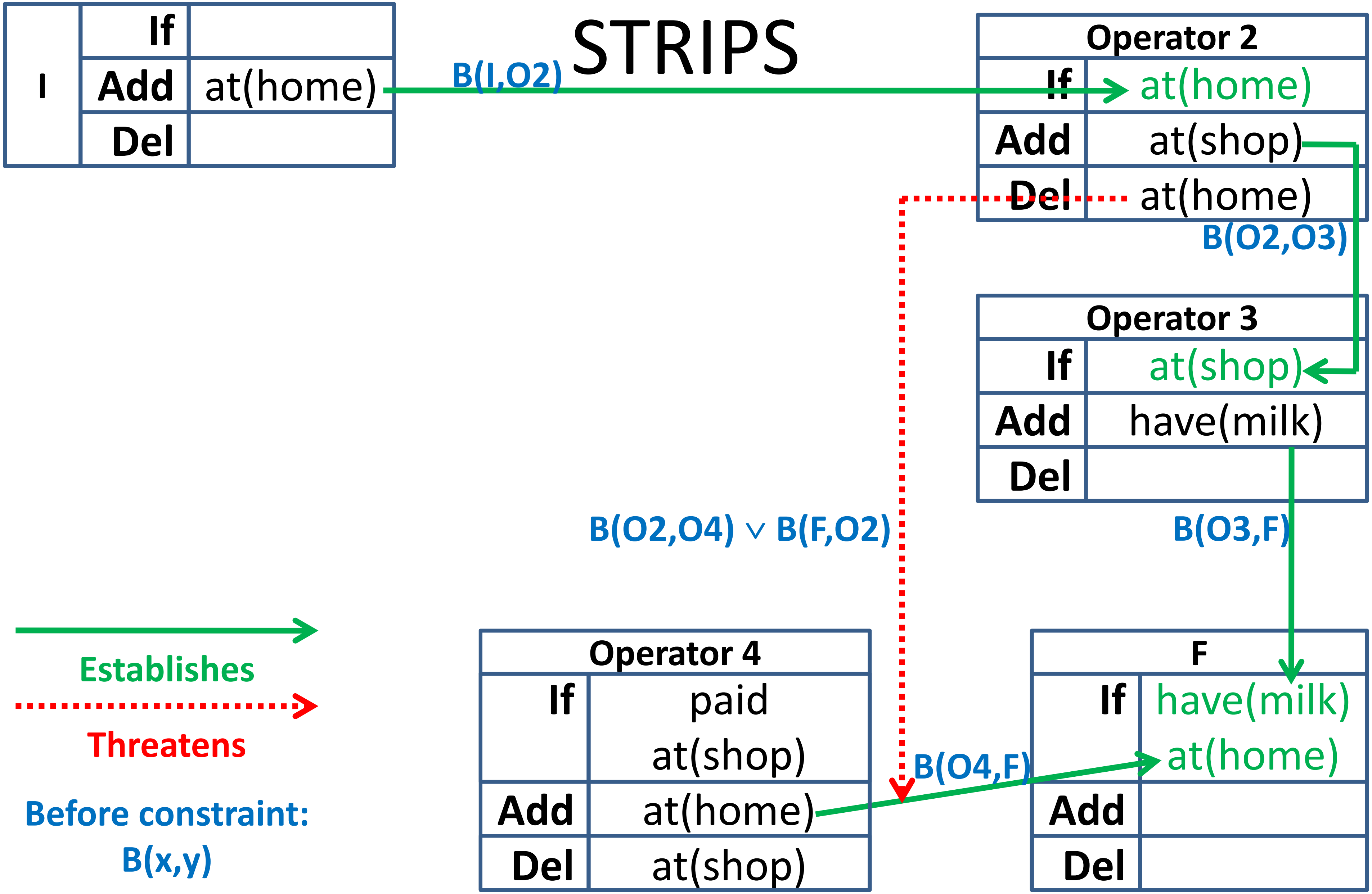
I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O3,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS

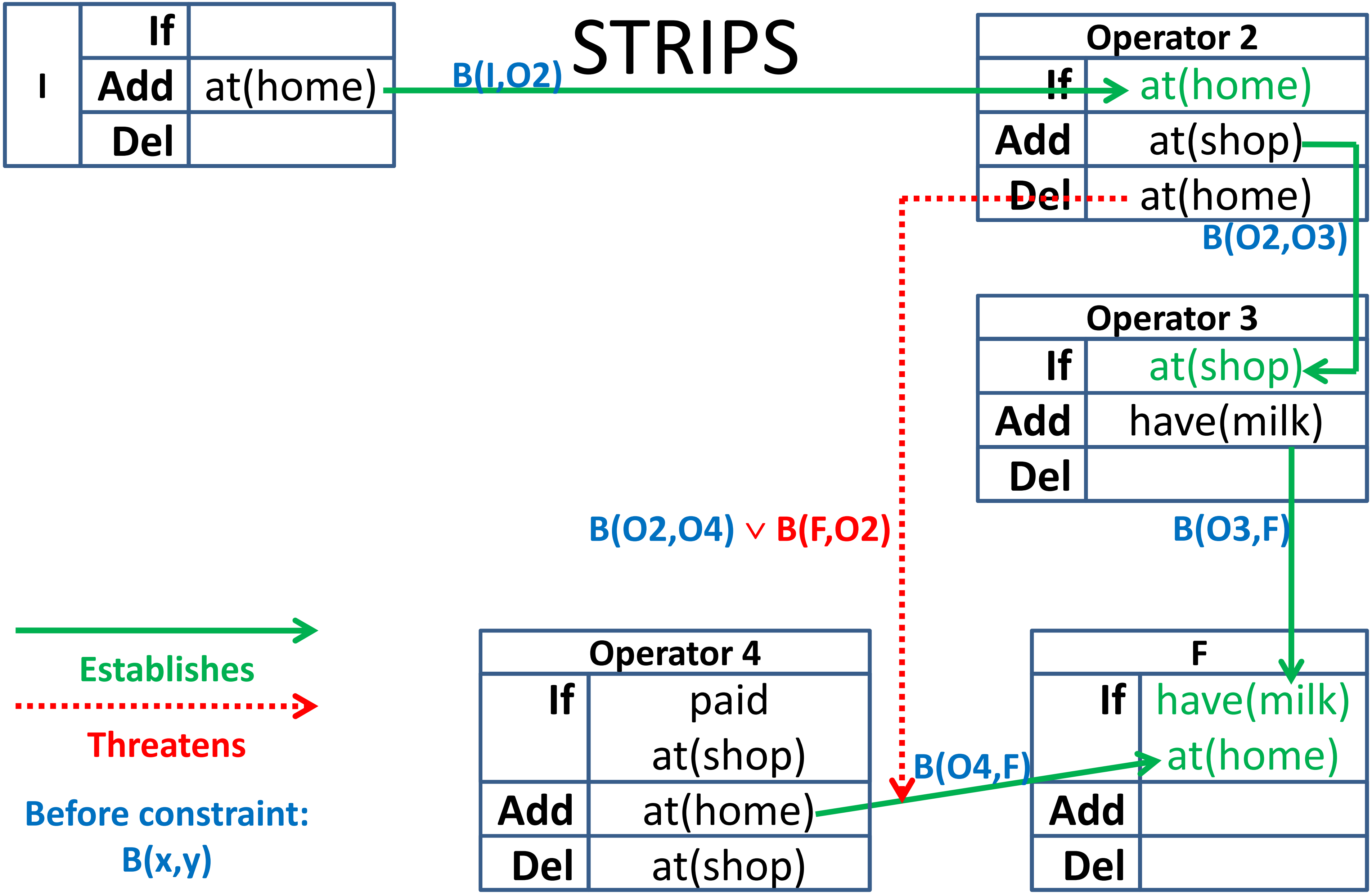


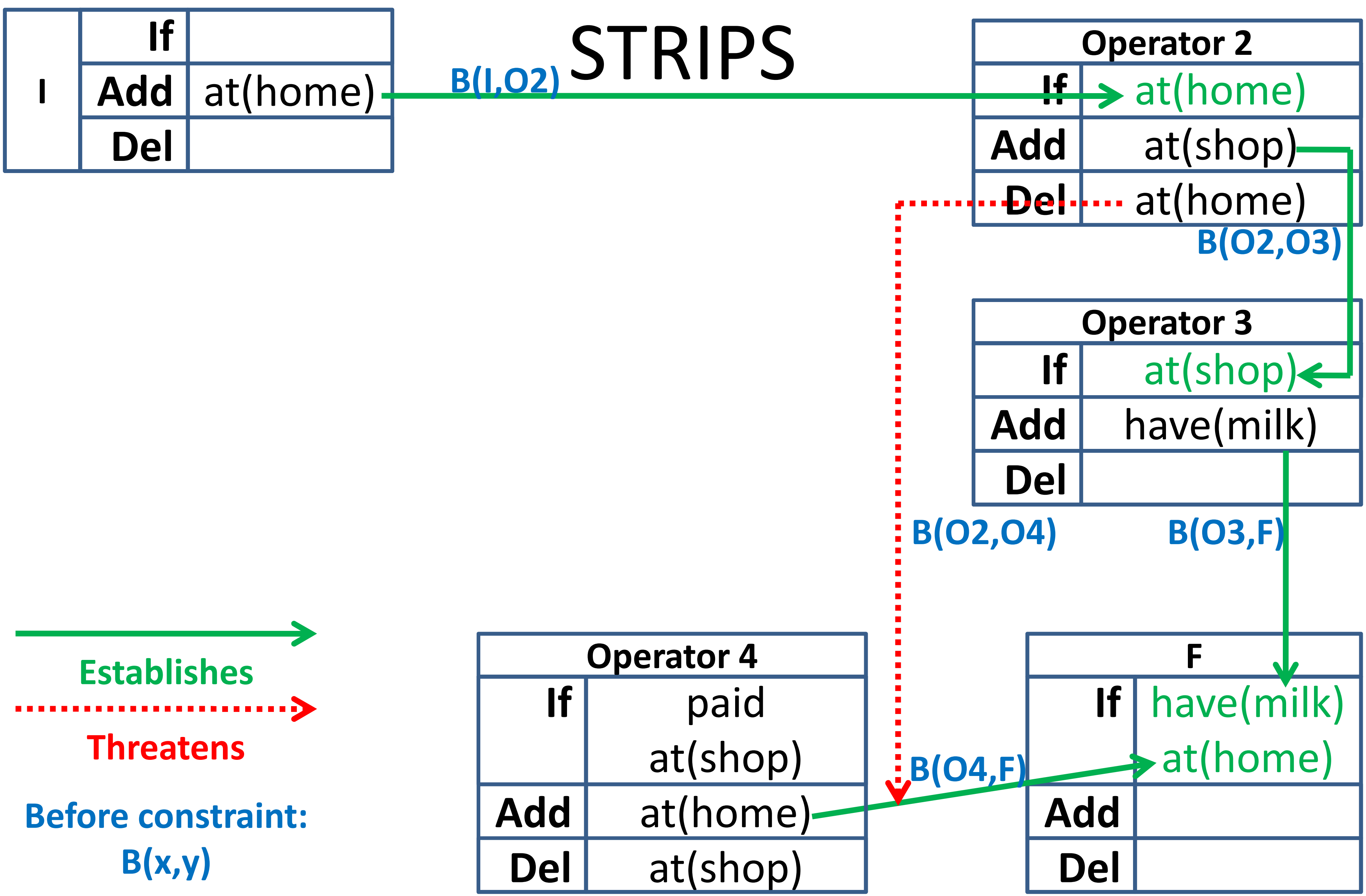
I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O3,F)
Establishes
Threatens
Before constraint:
B(x,y)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del

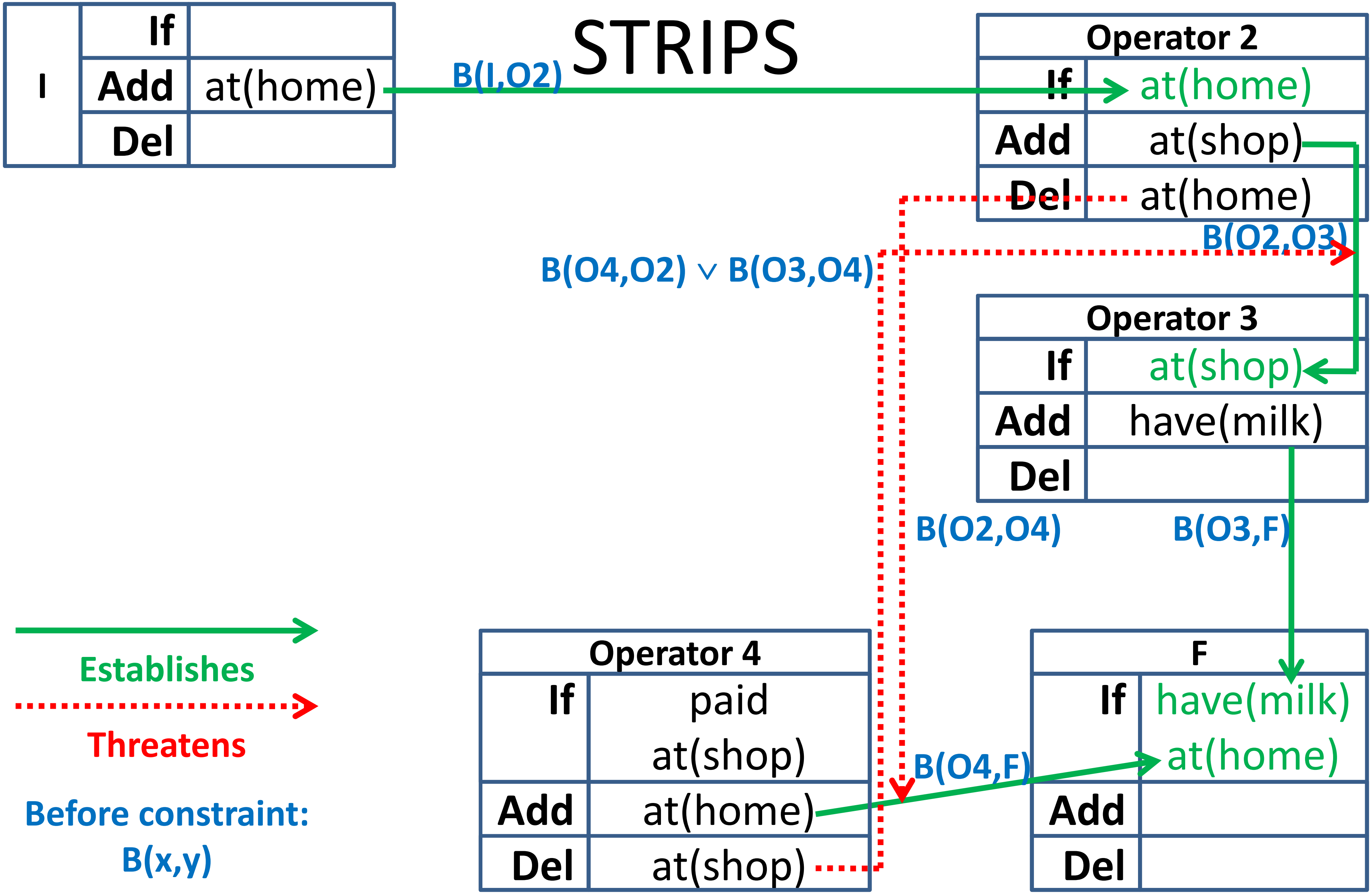
STRIPS


I
If
Add at(home)
Del
B(I,O2)
Operator 2
If at(home)
Add at(shop)
Del at(home)
B(O2,O3)
Operator 3
If at(shop)
Add have(milk)
Del
B(O3,F)
B(O2,O4) ∨ B(F,O2)
Operator 4
If paid
at(shop)
Add at(home)
Del at(shop)
B(O4,F)
F
If have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

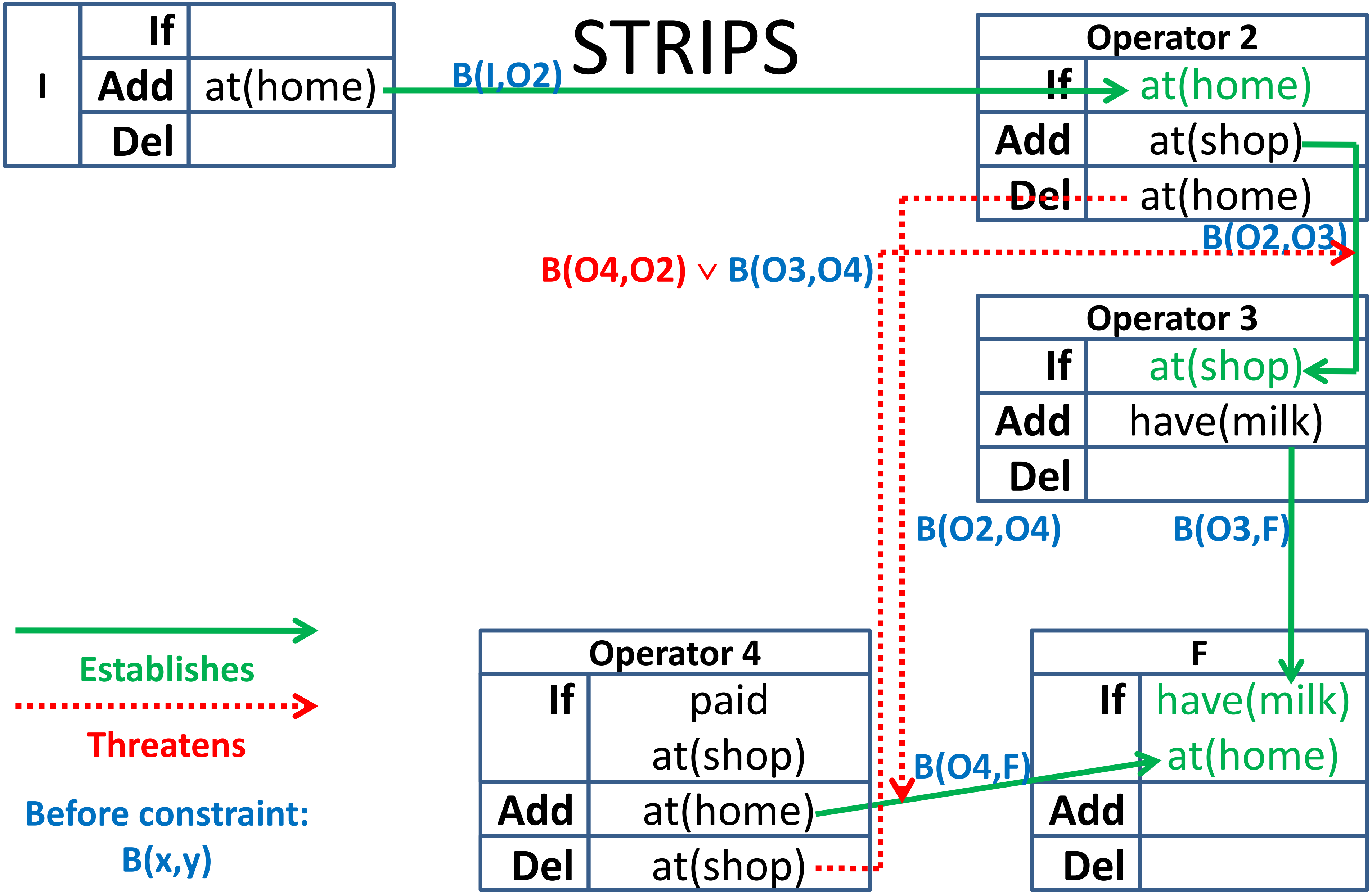
STRIPS


I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O2,O4) ∨ B(F,O2)
B(O3,F)
Establishes
Threatens
Before constraint:
B(x,y)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del


STRIPS
I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O2,O4)
B(O3,F)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

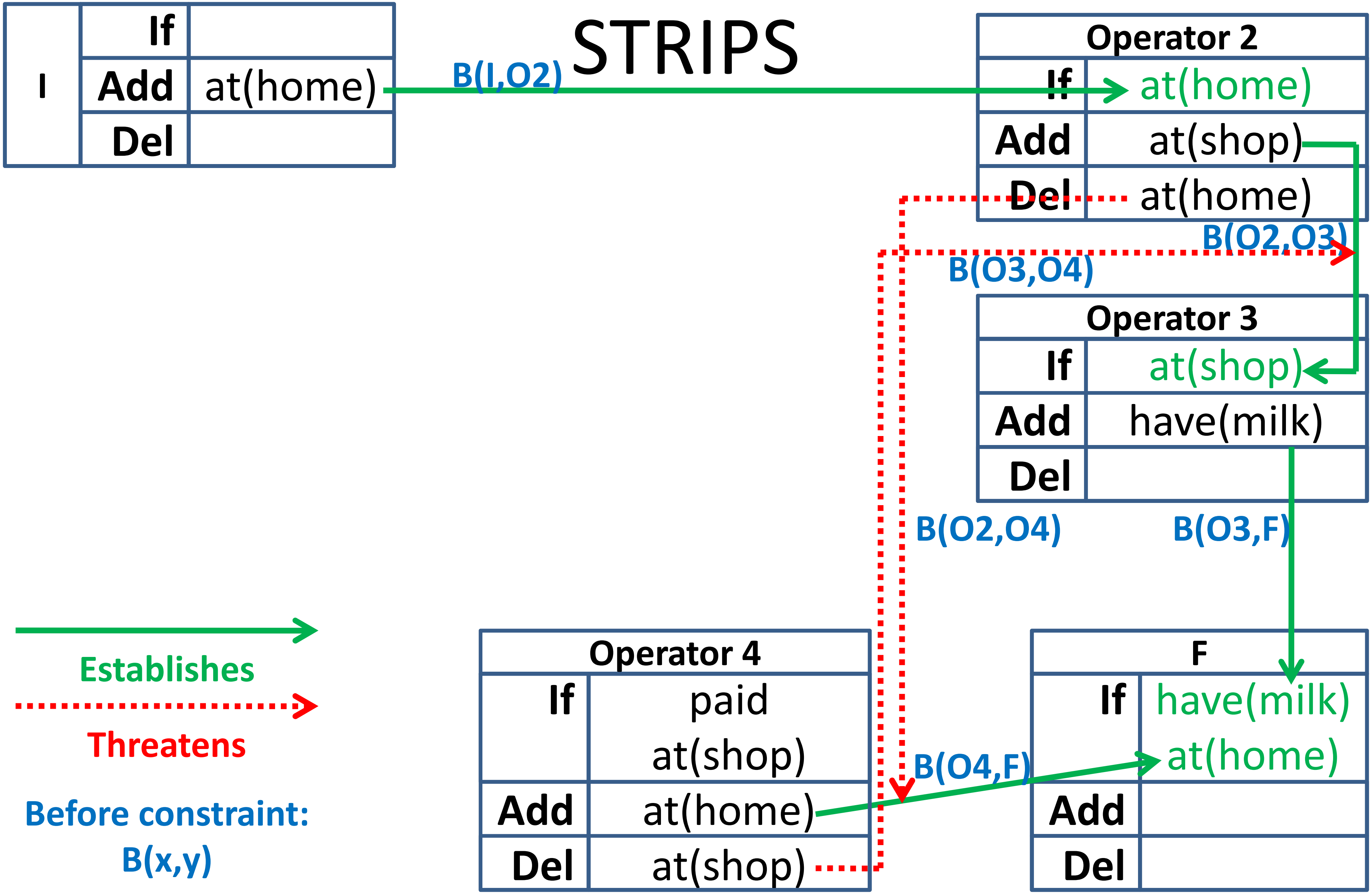

STRIPS
I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
B(O4,O2) ∨ B(O3,O4)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O2,O4)
B(O3,F)
Establishes
Threatens
Before constraint:
B(x,y)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del

STRIPS

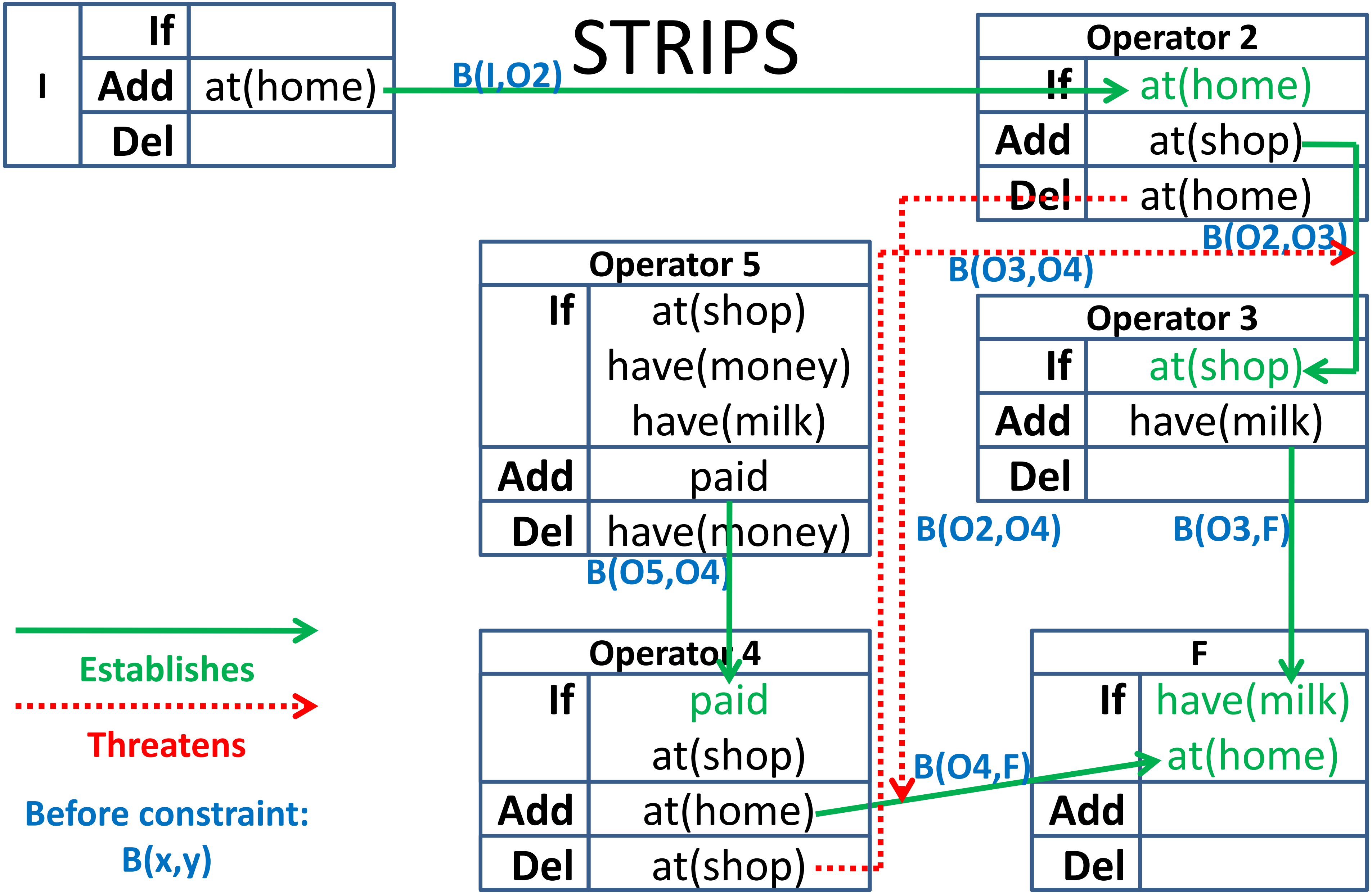


I
If
Add at(home)
Del
B(I,O2)
Operator 2
If at(home)
Add at(shop)
Del at(home)
B(O2,O3)
B(O4,O2) ∨ B(O3,O4)
Operator 3
If at(shop)
Add have(milk)
Del
B(O2,O4)
B(O3,F)
Establishes
Threatens
Before constraint:
B(x,y)
Operator 4
If paid
at(shop)
Add at(home)
Del at(shop)
B(O4,F)
F
If have(milk)
at(home)
Add
Del

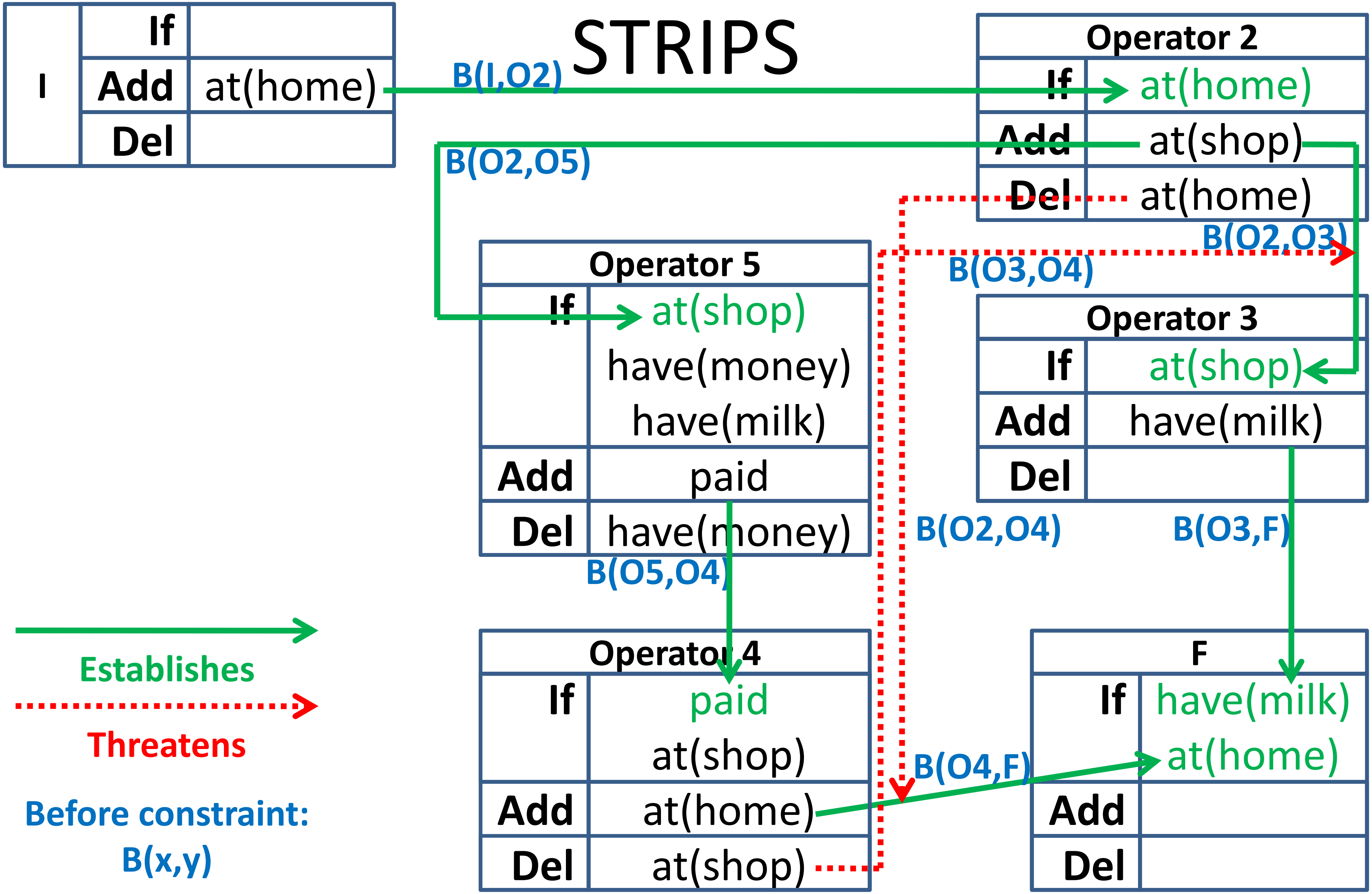
STRIPS

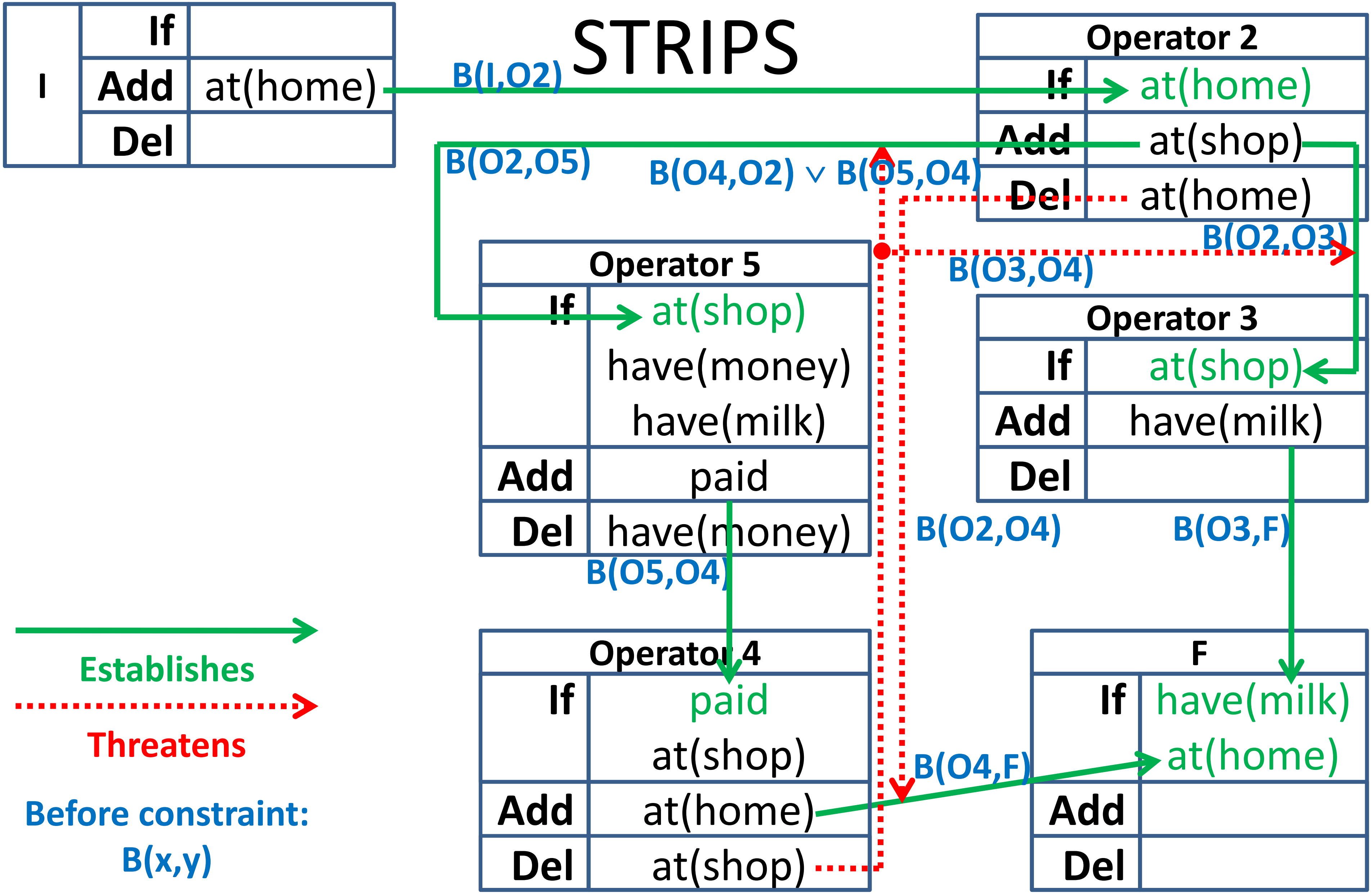

I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O3)
B(O3,O4)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O2,O4)
B(O3,F)
Establishes
Threatens
Before constraint:
B(x,y)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del

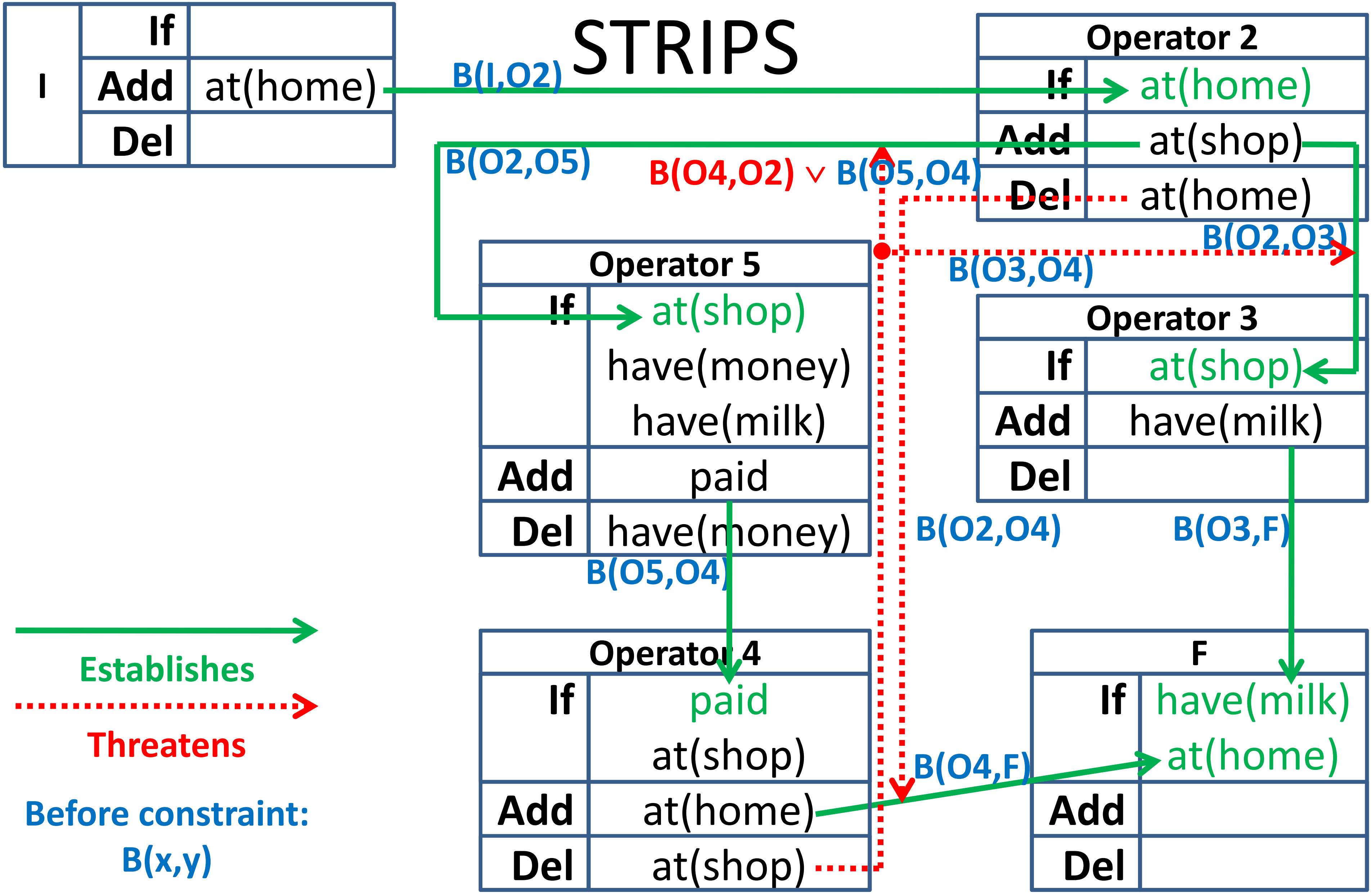
STRIPS



I
If
Add at(home)
Del
B(I,O2)
Operator 2
If at(home)
Add at(shop)
Del at(home)
B(O2,O3)
B(O3,O4)
Operator 5
If at(shop) have(money) have(milk)
Add paid
Del have(money)
B(O5,O4)
Operator 3
If at(shop)
Add have(milk)
Del
B(O2,O4)
B(O3,F)
Operator 4
If paid at(shop)
Add at(home)
Del at(shop)
B(O4,F)
F
If have(milk) at(home)
Add
Del
Establishes
Threatens
Before constraint: B(x,y)

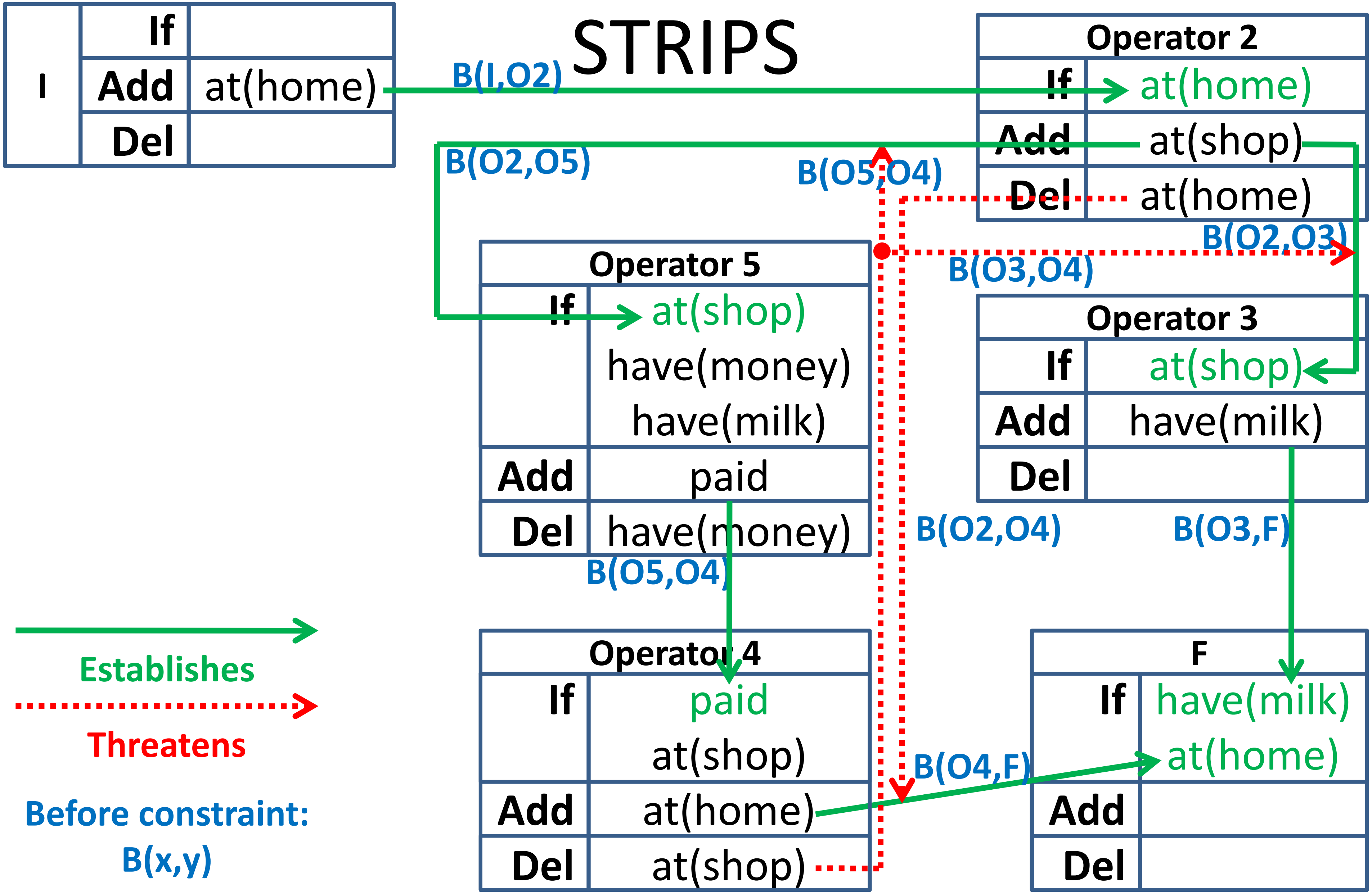

STRIPS
I
If
Add at(home)
Del
Operator 2
If at(home)
Add at(shop)
Del at(home)
Operator 5
If at(shop) have(money) have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
Operator 4
If paid at(shop)
Add at(home)
Del at(shop)
F
If have(milk) at(home)
Add
Del
B(I,O2)
B(O2,O5)
B(O2,O3)
B(O3,O4)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)


STRIPS
I
If
Add at(home)
Del
B(I,O2)
Operator 2
If at(home)
Add at(shop)
Del at(home)
B(O2,O5)
B(O4,O2) ∨ B(O5,O4)
B(O2,O3)
B(O3,O4)
Operator 5
If at(shop) have(money) have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
B(O2,O4)
B(O3,F)
B(O5,O4)
Operator 4
If paid at(shop)
Add at(home)
Del at(shop)
B(O4,F)
F
If have(milk) at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

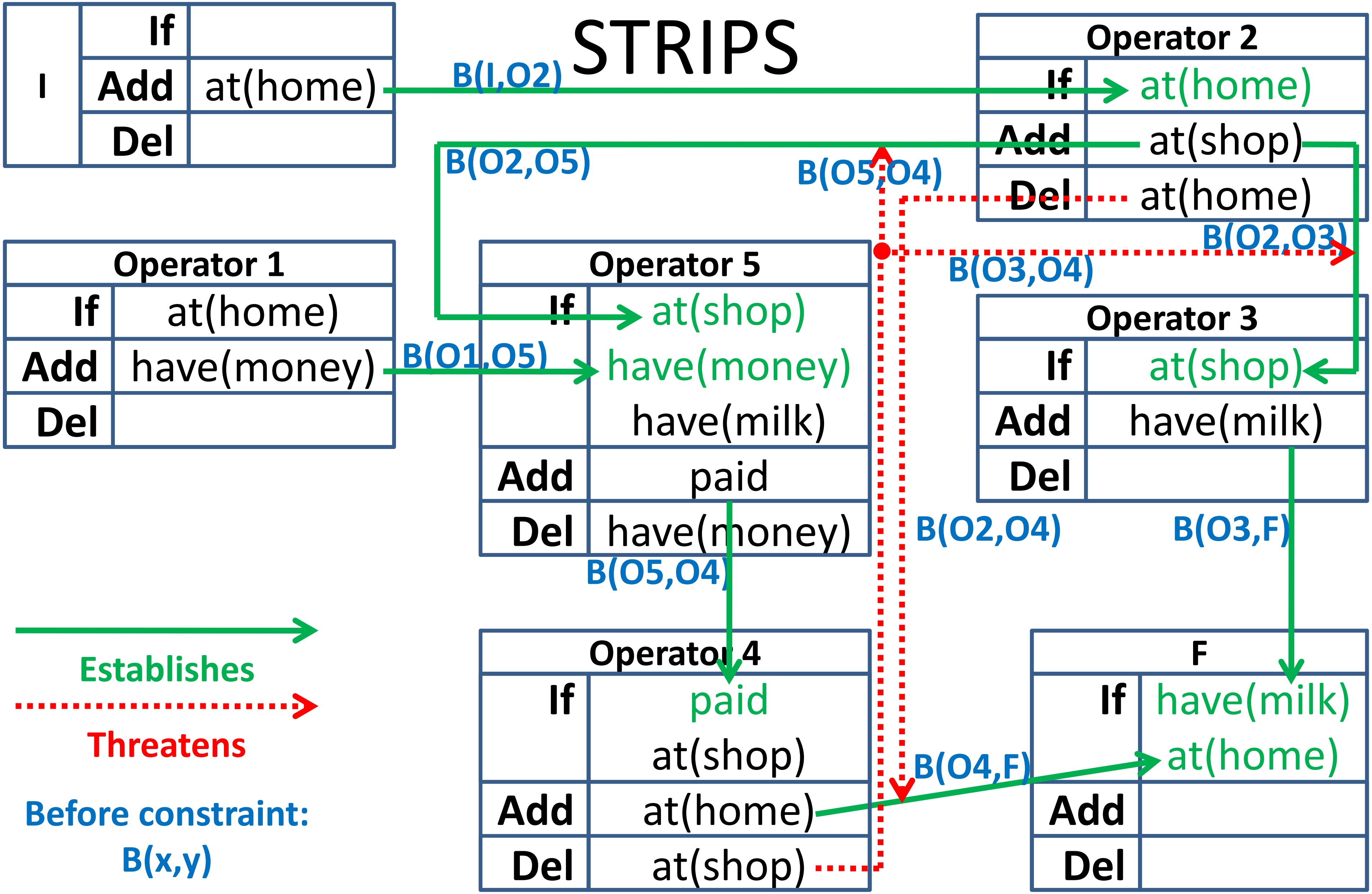


STRIPS
I
If
Add at(home)
Del
B(I,O2)
Operator 2
If at(home)
Add at(shop)
Del at(home)
B(O2,O5)
B(O4,O2) ∨ B(O5,O4)
B(O2,O3)
B(O3,O4)
Operator 5
If at(shop) have(money) have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
B(O2,O4)
B(O3,F)
B(O5,O4)
Operator 4
If paid at(shop)
Add at(home)
Del at(shop)
F
If have(milk) at(home)
Add
Del
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)

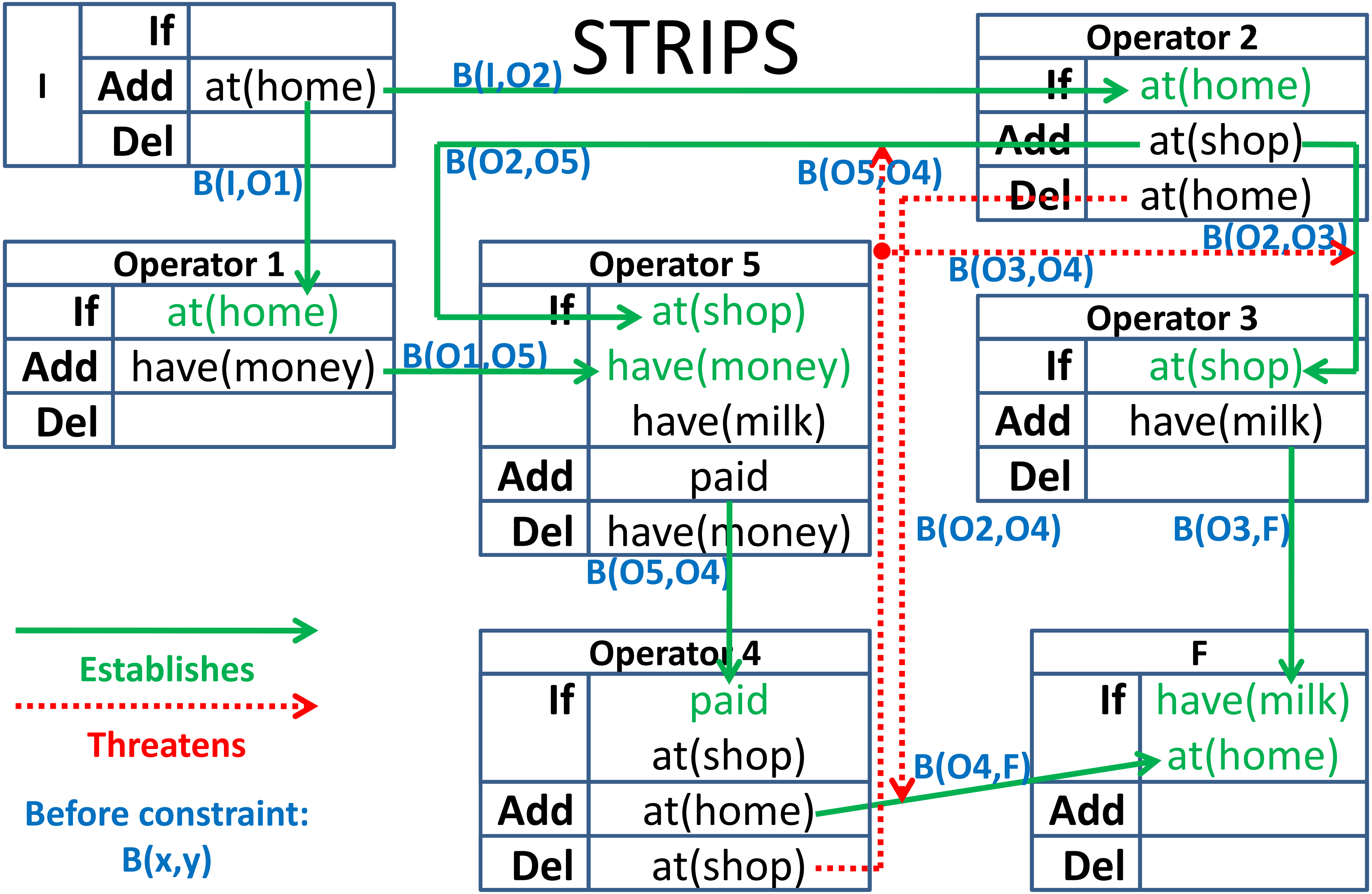
STRIPS



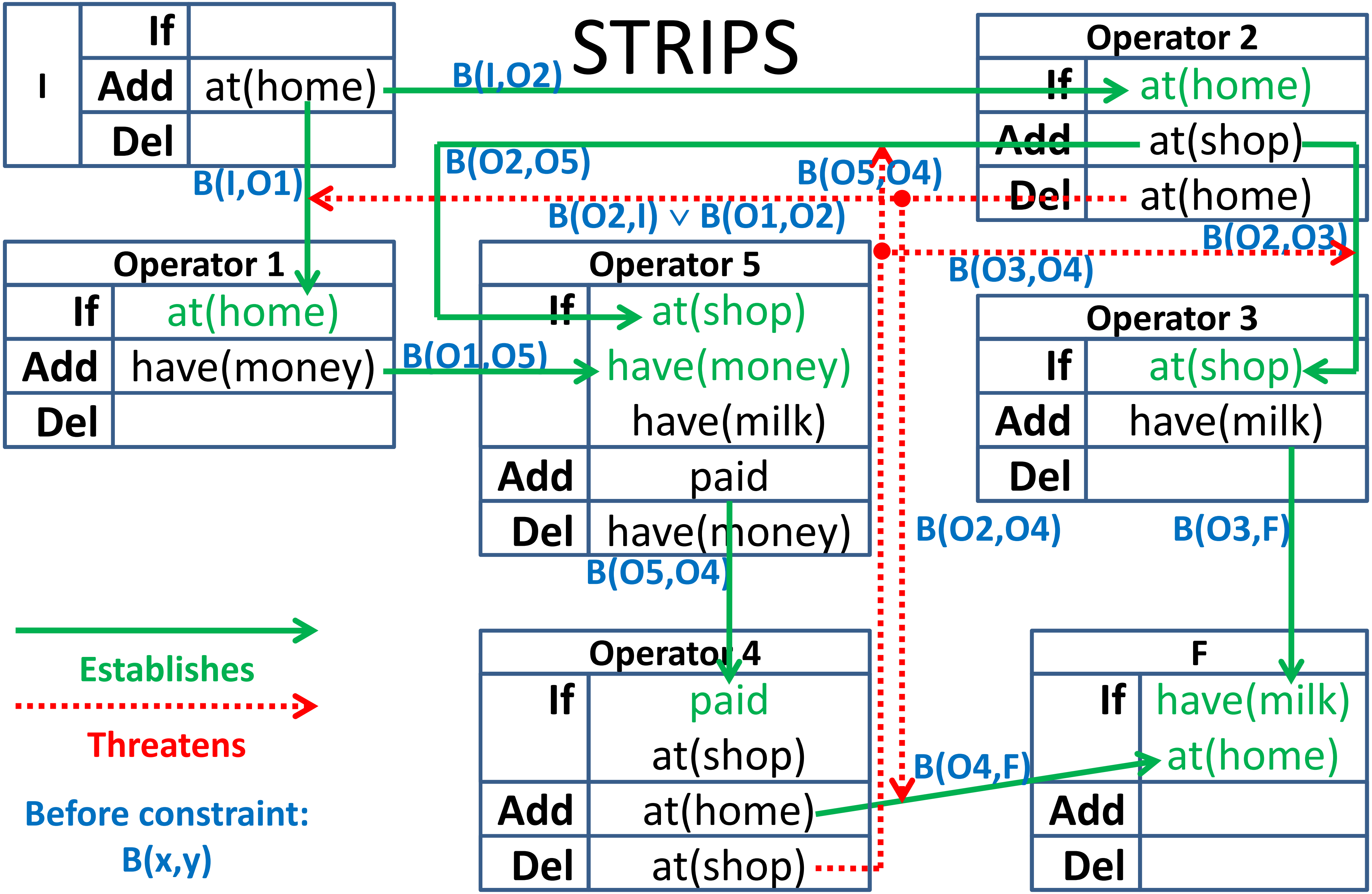
I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O5)
B(O5,O4)
B(O2,O3)
B(O3,O4)
Operator 5
If
at(shop)
have(money)
have(milk)
Add
paid
Del
have(money)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O2,O4)
B(O3,F)
B(O5,O4)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)


STRIPS
I
If
Add at(home)
Del
Operator 2
If at(home)
Add at(shop)
Del at(home)
Operator 1
If at(home)
Add have(money)
Del
Operator 5
If at(shop)
have(money)
have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
Operator 4
If paid
at(shop)
Add at(home)
Del at(shop)
F
If have(milk)
at(home)
Add
Del
B(I,O2)
B(O2,O5)
B(O5,O4)
B(O2,O3)
B(O3,O4)
B(O1,O5)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS

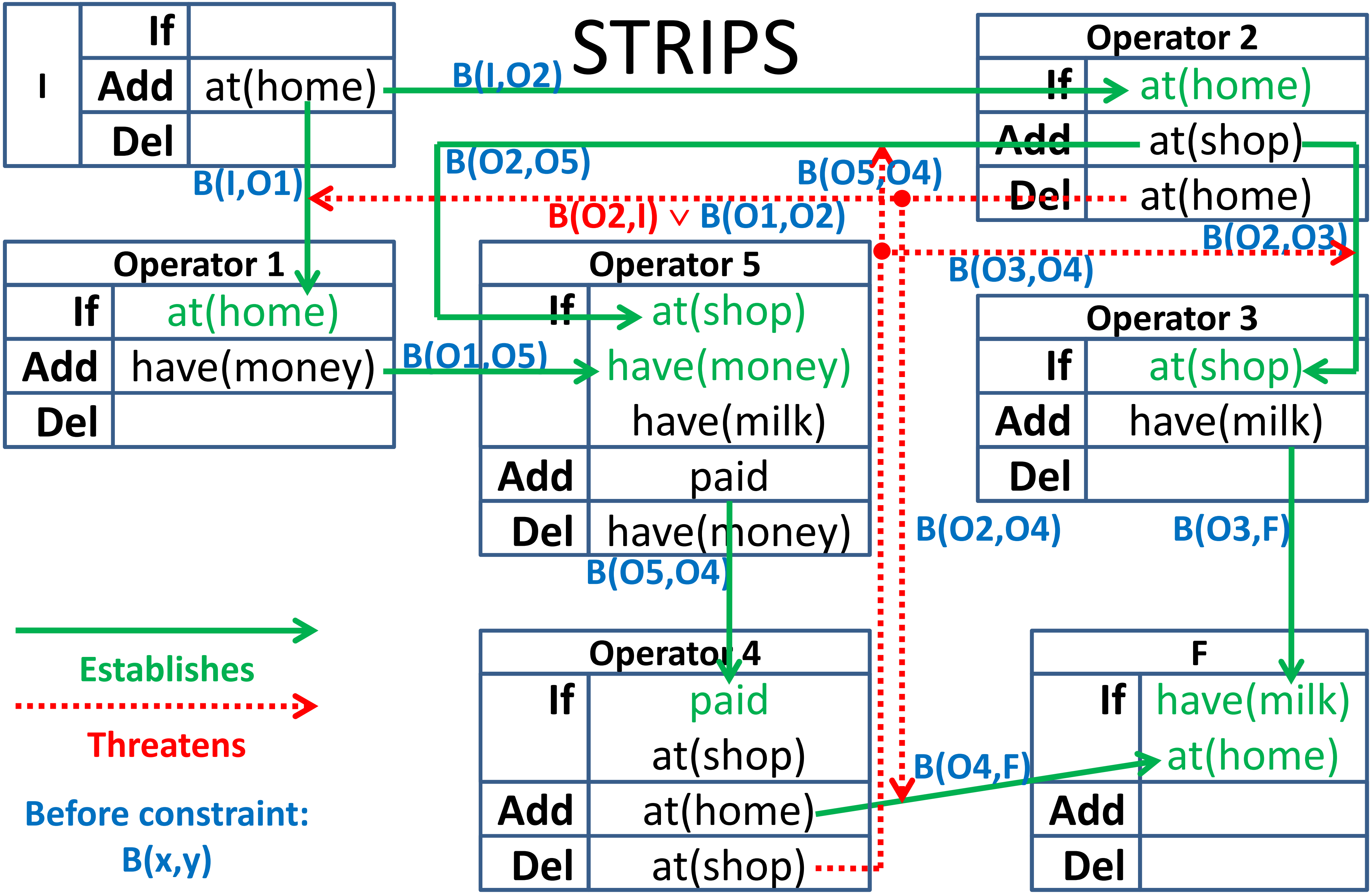


I
If
Add
at(home)
Del
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
Operator 1
If
at(home)
Add
have(money)
Del
Operator 5
If
at(shop)
have(money)
have(milk)
Add
paid
Del
have(money)
Operator 3
If
at(shop)
Add
have(milk)
Del
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
F
If
have(milk)
at(home)
Add
Del
B(I,O2)
B(I,O1)
B(O2,O5)
B(O5,O4)
B(O2,O3)
B(O3,O4)
B(O1,O5)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)



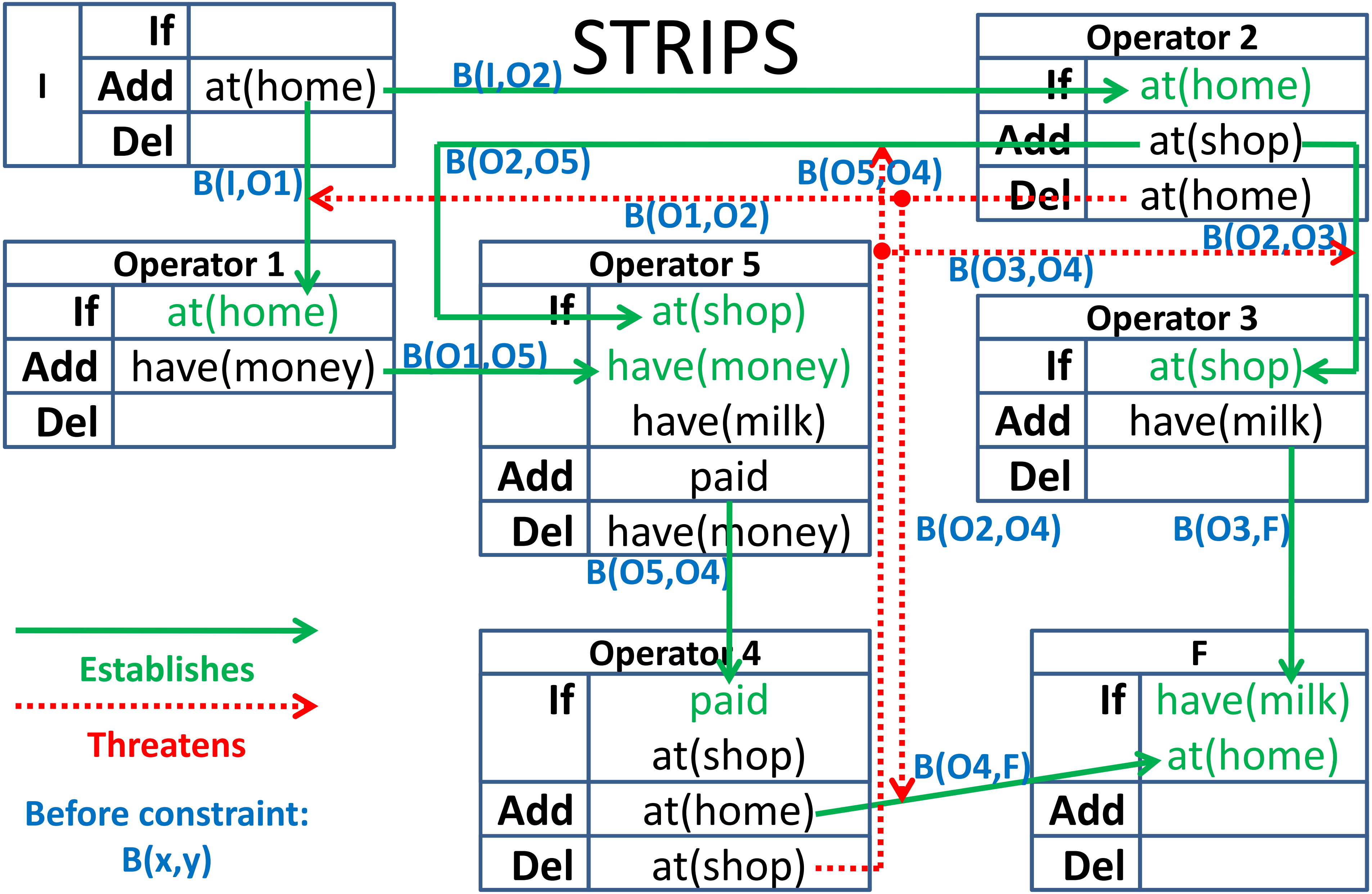
STRIPS
I
If
Add at(home)
Del
Operator 2
If at(home)
Add at(shop)
Del at(home)
Operator 1
If at(home)
Add have(money)
Del
Operator 5
If at(shop)
have(money)
have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
Operator 4
If paid
at(shop)
Add at(home)
Del at(shop)
F
If have(milk)
at(home)
Add
Del
B(I,O2)
B(O2,O5)
B(I,O1)
B(O5,O4)
B(O2,I) ∨ B(O1,O2)
B(O2,O3)
B(O3,O4)
B(O1,O5)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)

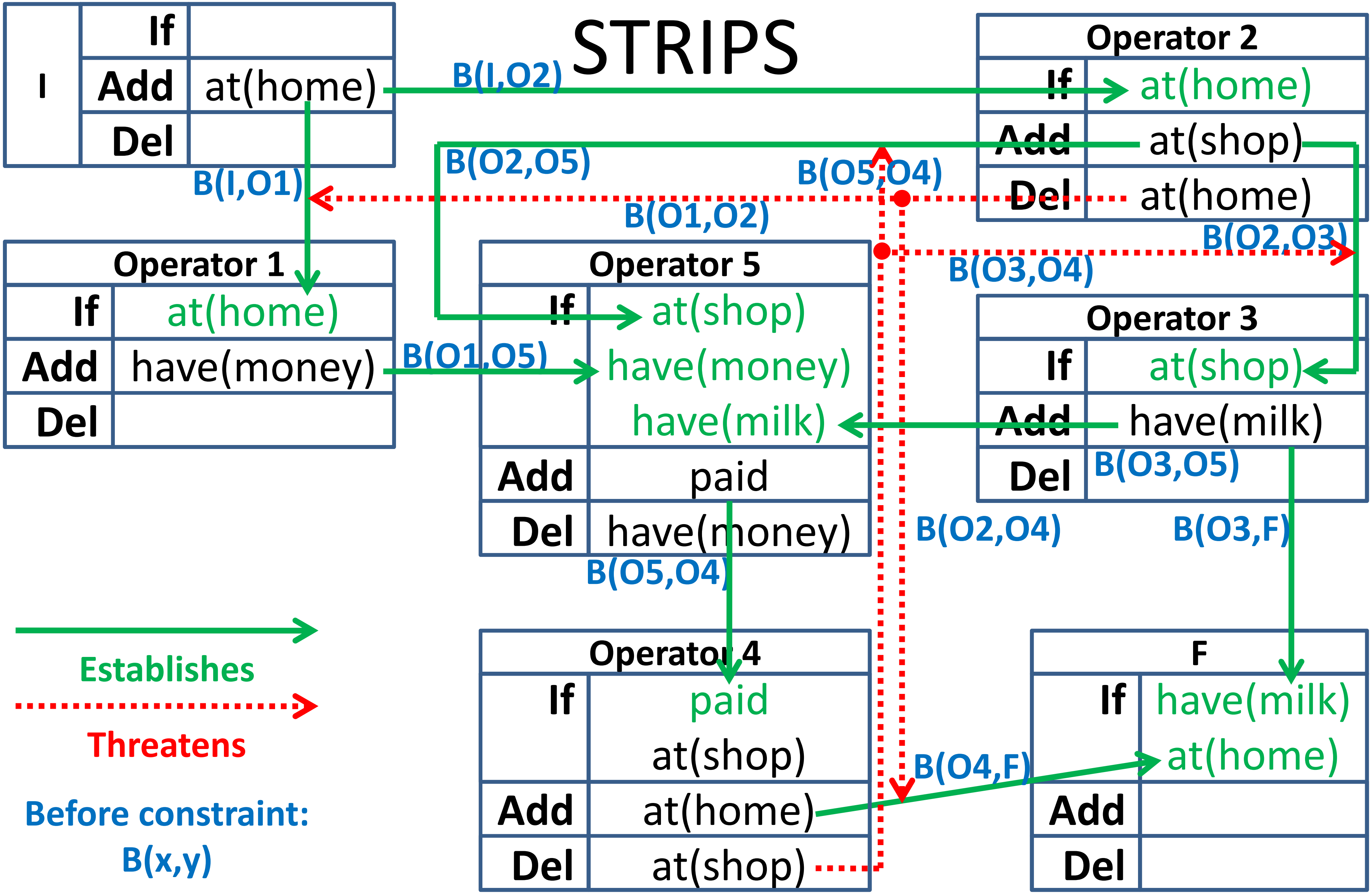
STRIPS



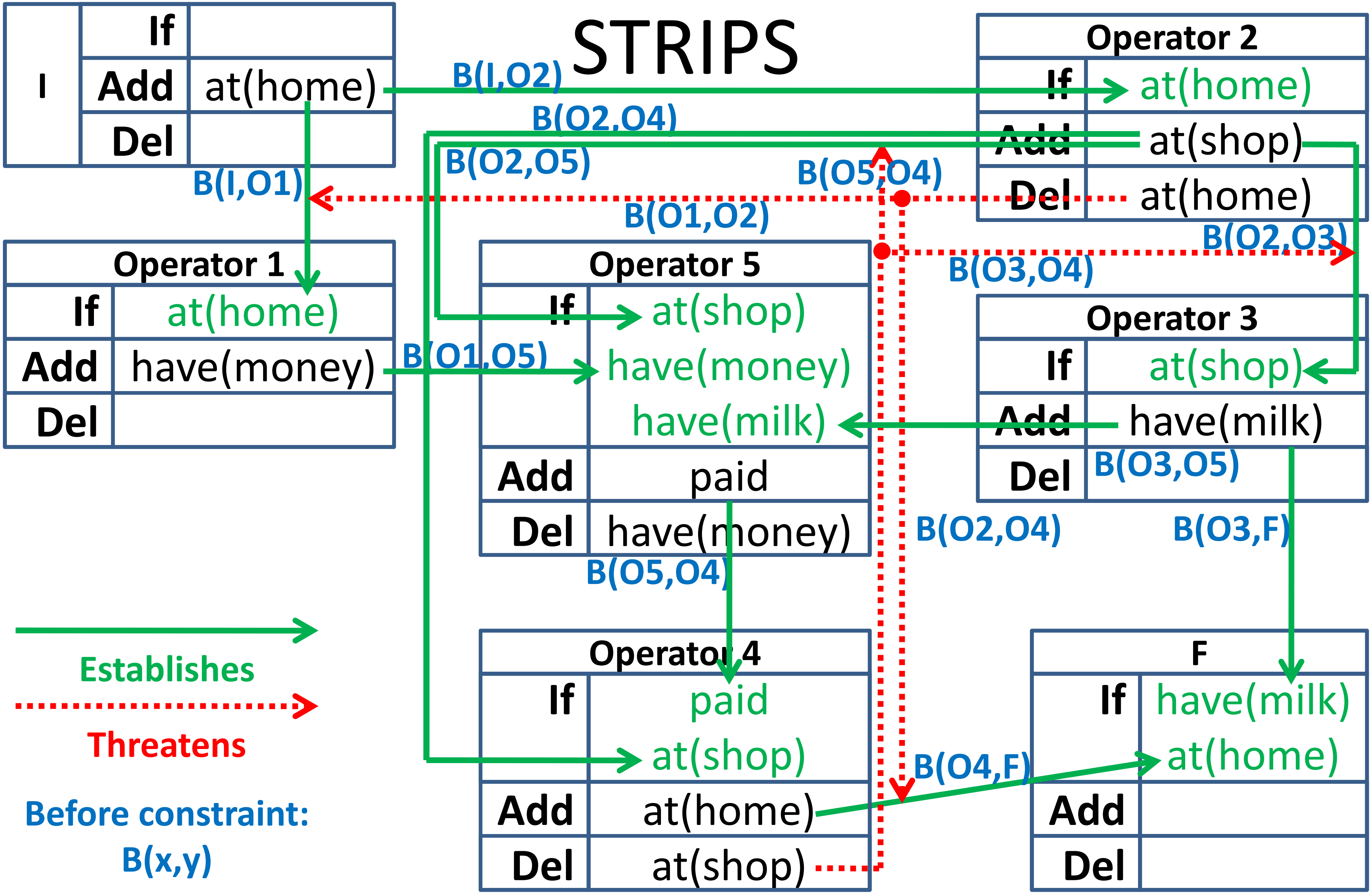
I
If
Add
at(home)
Del
B(I,O2)
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
B(O2,O5)
B(O5,O4)
B(I,O1)
B(O2,I) ∨ B(O1,O2)
B(O2,O3)
B(O3,O4)
Operator 1
If
at(home)
Add
have(money)
Del
Operator 5
If
at(shop)
have(money)
have(milk)
Add
paid
Del
have(money)
B(O1,O5)
Operator 3
If
at(shop)
Add
have(milk)
Del
B(O2,O4)
B(O3,F)
B(O5,O4)
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
B(O4,F)
F
If
have(milk)
at(home)
Add
Del
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS


I
If
Add at(home)
Del
Operator 2
If at(home)
Add at(shop)
Del at(home)
Operator 1
If at(home)
Add have(money)
Del
Operator 5
If at(shop) have(money) have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
Operator 4
If paid at(shop)
Add at(home)
Del at(shop)
F
If have(milk) at(home)
Add
Del
B(I,O2)
B(O2,O5)
B(O5,O4)
B(I,O1)
B(O1,O2)
B(O2,O3)
B(O3,O4)
B(O1,O5)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)


STRIPS
I
If
Add at(home)
Del
Operator 2
If at(home)
Add at(shop)
Del at(home)
Operator 1
If at(home)
Add have(money)
Del
Operator 5
If at(shop) have(money) have(milk)
Add paid
Del have(money)
Operator 3
If at(shop)
Add have(milk)
Del
Operator 4
If paid at(shop)
Add at(home)
Del at(shop)
F
If have(milk) at(home)
Add
Del
B(I,O2)
B(I,O1)
B(O2,O5)
B(O5,O4)
B(O1,O2)
B(O2,O3)
B(O3,O4)
B(O1,O5)
B(O3,O5)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS



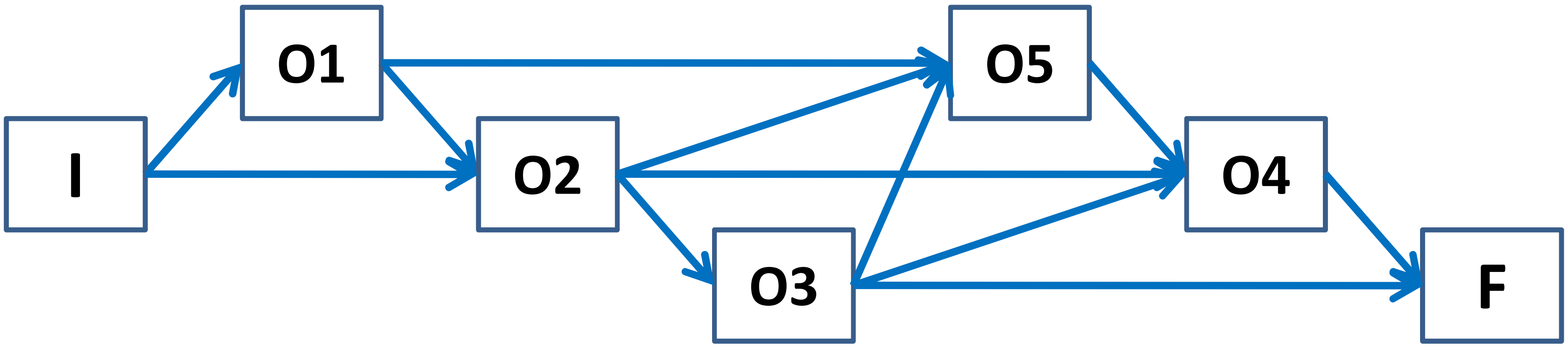
I
If
Add
at(home)
Del
Operator 2
If
at(home)
Add
at(shop)
Del
at(home)
Operator 1
If
at(home)
Add
have(money)
Del
Operator 5
If
at(shop)
have(money)
have(milk)
Add
paid
Del
have(money)
Operator 3
If
at(shop)
Add
have(milk)
Del
Operator 4
If
paid
at(shop)
Add
at(home)
Del
at(shop)
F
If
have(milk)
at(home)
Add
Del
B(I,O2)
B(O2,O4)
B(O2,O5)
B(I,O1)
B(O5,O4)
B(O1,O2)
B(O2,O3)
B(O3,O4)
B(O1,O5)
B(O3,O5)
B(O2,O4)
B(O3,F)
B(O5,O4)
B(O4,F)
Establishes
Threatens
Before constraint:
B(x,y)

STRIPS

B(I,O1)
B(I,O2)
B(O1,O2)
B(O1,O5)
B(O2,O3)
B(O2,O4)
B(O2,O5)
B(O3,O4)
B(O3,O5)
B(O3,F)
B(O4,F)
B(O5,O4)

Are the before constraints satisfiable?

STRIPS



Are the before constraints satisfiable?

YES:

$\longrightarrow$ **O1** $\longrightarrow$ **O2** $\longrightarrow$ **O3** $\longrightarrow$ **O5** $\longrightarrow$ **O4** $\longrightarrow$

Exercises: Artificial Intelligence

Planning & Logic: English to Logic

Planning & Logic: English to Logic

PROBLEM & SOLUTION

Problem & Solution

- *Not all students take both history and biology*

Problem & Solution

- *Not all students take both history and biology*

  $\neg \forall x\ [\text{student}(x) \Rightarrow \text{takes}(x,\text{hist}) \wedge \text{takes}(x,\text{bio})]$

Problem & Solution

- *Not all students take both history and biology*

$$\neg \forall x\ [\text{student}(x) \Rightarrow \text{takes}(x,\text{hist}) \wedge \text{takes}(x,\text{bio})]$$

$$\Leftrightarrow \ [A \Rightarrow B \Leftrightarrow \neg A \vee B]$$

$$\neg \forall x\ [\neg[\text{student}(x)] \vee [\text{takes}(x,\text{hist}) \wedge \text{takes}(x,\text{bio})]]$$

Problem & Solution

- *Not all students take both history and biology*

$$\neg \forall x\ [student(x) \Rightarrow takes(x,hist) \wedge takes(x,bio)]$$

$$\Leftrightarrow \quad [A \Rightarrow B \Leftrightarrow \neg A \vee B]$$

$$\neg \forall x\ [\neg[student(x)] \vee [takes(x,hist) \wedge takes(x,bio)]]$$

$$\Leftrightarrow \quad [\neg \forall x\ (F) \Leftrightarrow \exists x\ (\neg F)]$$

$$\exists x\ \neg[\neg[student(x)] \vee [takes(x,hist) \wedge takes(x,bio)]]$$

Problem & Solution

- *Not all students take both history and biology*

$$\neg\forall x\ [student(x) \Rightarrow takes(x,hist) \wedge takes(x,bio)]$$

$$\Leftrightarrow\ [A \Rightarrow B \Leftrightarrow \neg A \vee B]$$

$$\neg\forall x\ [\neg[student(x)] \vee [takes(x,hist) \wedge takes(x,bio)]]$$

$$\Leftrightarrow\ [\neg\forall x\ (F) \Leftrightarrow \exists x\ (\neg F)]$$

$$\exists x\ \neg[\neg[student(x)] \vee [takes(x,hist) \wedge takes(x,bio)]]$$

$$\Leftrightarrow\ [\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B],[\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B]$$

$$\exists x\ [student(x) \wedge [\neg takes(x,hist) \vee \neg takes(x,bio)]]$$

Problem & Solution

- *No person likes a smart vegetarian*

Problem & Solution

- *No person likes a smart vegetarian*

  $\forall x\ \forall y\ [\text{person}(x) \wedge \text{vegetarian}(y) \wedge \text{smart}(y) \Rightarrow \neg \text{likes}(x,y)]$

Problem & Solution

- *No person likes a smart vegetarian*

$$\forall x\ \forall y\ [person(x) \wedge vegetarian(y) \wedge smart(y) \Rightarrow \neg likes(x,y)]$$

$$\Leftrightarrow \quad [A \Rightarrow B \Leftrightarrow \neg A \ \vee B]$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge vegetarian(y) \wedge smart(y)] \vee \neg likes(x,y)]$$

Problem & Solution

- *No person likes a smart vegetarian*

$$\forall x\ \forall y\ [person(x) \land vegetarian(y) \land smart(y) \Rightarrow \neg likes(x,y)]$$

$$\Leftrightarrow \quad [A \Rightarrow B \Leftrightarrow \neg A \lor B]$$

$$\forall x\ \forall y\ [\neg[person(x) \land vegetarian(y) \land smart(y)] \lor \neg likes(x,y)]$$

$$\Leftrightarrow \quad [\neg A \lor \neg B \Leftrightarrow \neg(A \land B)]$$

$$\forall x\ \forall y\ \neg[person(x) \land vegetarian(y) \land smart(y) \land likes(x,y)]$$

Problem & Solution

- *No person likes a smart vegetarian*

$$\forall x\ \forall y\ [person(x) \wedge vegetarian(y) \wedge smart(y)\ \Rightarrow\ \neg likes(x,y)]$$

$$\Leftrightarrow\ [A \Rightarrow B \Leftrightarrow \neg A\ \vee B]$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge vegetarian(y) \wedge smart(y)] \vee \neg likes(x,y)]$$

$$\Leftrightarrow\ [\neg A\ \vee \neg B \Leftrightarrow \neg(A \wedge B)]$$

$$\forall x\ \forall y\ \neg[person(x) \wedge vegetarian(y) \wedge smart(y) \wedge likes(x,y)]$$

$$\Leftrightarrow\ [\forall x\ \neg(F) \Leftrightarrow \neg\exists x\ (F)]$$

$$\neg\ \exists x\ \exists y\ [person(x) \wedge vegetarian(y) \wedge smart(y) \wedge likes(x,y)]$$

Problem & Solution

- *There is a woman who likes all men who are not vegetarians.*

Problem & Solution

- *There is a woman who likes all men who are not vegetarians.*

  $\exists x[\text{woman}(x) \wedge [\forall y\ [\text{man}(y) \wedge \neg \text{vegetarian}(y) \Rightarrow \text{likes}(x,y)]]]$

Problem & Solution

- *The best score in history was better than the best score in biology.*

Problem & Solution

- *The best score in history was better than the best score in biology.*

  $\forall x\ \forall y\ [\text{bestscore}(\text{hist},x) \wedge \text{bestscore}(\text{bio},y) \Rightarrow \text{better}(x,y)]$

Problem & Solution

- *Every person who dislikes all vegetarians is smart.*

Problem & Solution

- *Every person who dislikes all vegetarians is smart.*

  $\forall x[\ person(x) \wedge [\forall y\ [vegetarian(y) \Rightarrow \neg likes(x,y)]] \Rightarrow smart(x)]$

Problem & Solution

- *There is a barber who shaves all men in town who do not shave themselves.*

Problem & Solution

- *There is a barber who shaves all men in town who do not shave themselves.*

  $\exists x\ [\text{barber}(x) \wedge [\forall y\ [\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y) \Rightarrow \text{shaves}(x,y)]]]$

Problem & Solution

- *There is a barber who shaves all men in town who do not shave themselves.*

$$\exists x\ [\text{barber}(x) \wedge [\forall y\ [\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y) \Rightarrow \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x\ [\text{barber}(x) \wedge [\forall y\ [\neg[\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y)] \vee \text{shaves}(x,y)]]]$$

Problem & Solution

- *There is a barber who shaves all men in town who do not shave themselves.*

$$\exists x\ [barber(x) \wedge [\forall y\ [townsman(y) \wedge \neg shaves\ (y,y) \Rightarrow shaves(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x\ [barber(x) \wedge [\forall y\ [\neg[townsman(y) \wedge \neg shaves\ (y,y)] \vee shaves(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x\ [barber(x) \wedge [\forall y\ \neg[townsman(y) \wedge \neg shaves\ (y,y) \wedge \neg shaves(x,y)]]]$$

Problem & Solution

- *There is a barber who shaves all men in town who do not shave themselves.*

$$\exists x\ [\text{barber}(x) \wedge [\forall y\ [\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y) \Rightarrow \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x\ [\text{barber}(x) \wedge [\forall y\ [\neg [\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y)] \vee \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x\ [\text{barber}(x) \wedge [\forall y\ \neg [\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y) \wedge \neg \text{shaves}(x,y)]]]$$

$$\Leftrightarrow$$

$$\exists x\ [\text{barber}(x) \wedge\ [\neg\ \exists y\ [\text{townsman}(y) \wedge \neg \text{shaves}\ (y,y) \wedge \neg \text{shaves}(x,y)]]]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x\ \forall y\ [\text{person}(x) \wedge \text{professor}(y) \Rightarrow [\text{likes}(x,y) \Rightarrow \text{smart}(y)]]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [likes(x,y) \Rightarrow smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [\neg likes(x,y) \vee smart(y)]]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [likes(x,y) \Rightarrow smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee [\neg likes(x,y) \vee smart(y)]]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [likes(x,y) \Rightarrow smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee \neg[likes(x,y) \wedge \neg smart(y)]]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [likes(x,y) \Rightarrow smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee \neg[likes(x,y) \wedge \neg smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ \neg[person(x) \wedge professor(y) \wedge likes(x,y) \wedge \neg smart(y)]$$

Problem & Solution

- *No person likes a professor unless the professor is smart.*

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [likes(x,y) \Rightarrow smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [person(x) \wedge professor(y) \Rightarrow [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee [\neg likes(x,y) \vee smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ [\neg[person(x) \wedge professor(y)] \vee \neg[likes(x,y) \wedge \neg smart(y)]] \Leftrightarrow$$

$$\forall x\ \forall y\ \neg[person(x) \wedge professor(y) \wedge likes(x,y) \wedge \neg smart(y)] \Leftrightarrow$$

$$\neg\ \exists x\ \exists y\ [person(x) \wedge professor(y) \wedge likes(x,y) \wedge \neg smart(y)]$$

Problem & Solution

- *Only one person failed both history and biology.*

Problem & Solution

- *Only one person failed both history and biology.*

$$\exists! x \; student(x) \land failed(x,hist) \land failed(x,bio)$$

Problem & Solution

- *Only one person failed both history and biology.*

$$\exists! x \text{ student}(x) \land \text{failed}(x,\text{hist}) \land \text{failed}(x,\text{bio})$$

Note that: $\exists! x\ p(x) \Leftrightarrow \exists x\ p(x) \land [\forall y\ [p(y) \Rightarrow x=y]]$

Problem & Solution

- *Politicians can fool some of the people all the time, and they can fool all of the people some of the time, but they can't fool all the people all of the time.*

Problem & Solution

- ***Politicians can fool some of the people all the time****, and they can fool all of the people some of the time, but they can't fool all the people all of the time.*

$$\forall x\ [\text{politician}(x) \Rightarrow [\exists y\ \text{people}(y) \wedge [\forall t\ \text{time}(t) \Rightarrow \text{fool}(x,y,t)]]]$$

Problem & Solution

- *Politicians can fool some of the people all the time, and* ***they can fool all of the people some of the time****, but they can't fool all the people all of the time.*

$$\forall x\ [\text{politician}(x) \Rightarrow [\exists y\ \text{people}(y) \wedge [\forall t\ \text{time}(t) \Rightarrow \text{fool}(x,y,t)]]]$$

$$\forall x\ [\text{politician}(x) \Rightarrow [\exists t\ \text{time}(t) \wedge [\forall y\ \text{people}(y) \Rightarrow \text{fool}(x,y,t)]]]$$

Problem & Solution

- *Politicians can fool some of the people all the time, and they can fool all of the people some of the time, but* ***they can't fool all the people all of the time****.*

$$\forall x\ [politician(x) \Rightarrow [\exists y\ people(y) \wedge [\forall t\ time(t) \Rightarrow fool(x,y,t)]]]$$

$$\forall x\ [politician(x) \Rightarrow [\exists t\ time(t) \wedge [\forall y\ people(y) \Rightarrow fool(x,y,t)]]]$$

$$\forall x\ [politician(x) \Rightarrow \neg[\forall y\ \forall t\ [people(y) \wedge time(t)] \Rightarrow fool(x,y,t)]]$$

Exercises: Artificial Intelligence

Planning & Logic: And-Or-If

Planning & Logic: And-Or-If

PROBLEM & SOLUTION

Problem & Solution

- *One more outburst like that and you are in contempt of court.*

Problem & Solution

- *One more outburst like that and you are in contempt of court.*

outburst $\Rightarrow$ court

Problem & Solution

- *One more outburst like that and you are in contempt of court.*

$$\text{outburst} \Rightarrow \text{court}$$

NOT: $\text{outburst} \wedge \text{court}$

Problem & Solution

- *Either the Red Sox win or I’m out ten dollars.*

Problem & Solution

- *Either the Red Sox win or I'm out ten dollars.*

$$\text{redSoxWin} \Leftrightarrow \neg \text{outTenDollars}$$

Problem & Solution

- *Either the Red Sox win or I'm out ten dollars.*

$$redSoxWin \Leftrightarrow \neg outTenDollars$$

NOT: $redSoxWin \vee outTenDollars$

Problem & Solution

- *Maybe I’ll come to the party and maybe I won’t.*

Problem & Solution

- *Maybe I'll come to the party and maybe I won't.*

$$\text{maybeComeToParty} \lor \neg\text{maybeComeToParty}$$

Problem & Solution

- *Maybe I'll come to the party and maybe I won't.*

$$\text{maybeComeToParty} \lor \neg\text{maybeComeToParty}$$

NOT: $\text{maybeComeToParty} \land \neg\text{maybeComeToParty}$

Exercises: Artificial Intelligence

Planning & Logic: Weird Logic

Planning & Logic: Weird Logic

PROBLEM & SOLUTION

Problem & Solution

- *I don't jump off the Empire State Building implies if I jump off the Empire State Building, then I float safely to the ground.*

Problem & Solution

- *I don't jump off the Empire State Building implies if I jump off the Empire State Building, then I float safely to the ground.*
  - *Translating the meaning of the sentence is not possible*

Problem & Solution

- *I don't jump off the Empire State Building implies if I jump off the Empire State Building, then I float safely to the ground.*
  - *Translating the meaning of the sentence is not possible*

$$\neg \text{jumpESB} \Rightarrow [\text{jumpESB} \Rightarrow \text{floatTTGround}]$$

Problem & Solution

- *I don't jump off the Empire State Building implies if I jump off the Empire State Building, then I float safely to the ground.*
  - *Translating the meaning of the sentence is not possible*

$$\neg\text{jumpESB} \Rightarrow [\text{jumpESB} \Rightarrow \text{floatTTGround}] \Leftrightarrow$$

$$\neg\text{jumpESB} \Rightarrow [\neg\text{jumpESB} \vee \text{floatTTGround}]$$

Problem & Solution

- *I don't jump off the Empire State Building implies if I jump off the Empire State Building, then I float safely to the ground.*
  - *Translating the meaning of the sentence is not possible*

$$\neg \text{jumpESB} \Rightarrow [\text{jumpESB} \Rightarrow \text{floatTTGround}] \Leftrightarrow$$

$$\neg \text{jumpESB} \Rightarrow [\neg \text{jumpESB} \vee \text{floatTTGround}] \Leftrightarrow$$

$$\text{jumpESB} \vee \neg \text{jumpESB} \vee \text{floatTTGround}$$

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*
  - *Translating the meaning of the sentence is not possible*

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*
  - *Translating the meaning of the sentence is not possible*

$$\neg[\text{attempt} \Rightarrow \text{getF}] \Rightarrow \text{attempt}$$

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*
  - *Translating the meaning of the sentence is not possible*

$$\neg[\text{attempt} \Rightarrow \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$\neg[\neg\text{attempt} \vee \text{getF}] \Rightarrow \text{attempt}$$

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*
  - *Translating the meaning of the sentence is not possible*

$$\neg[\text{attempt} \Rightarrow \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$\neg[\neg\text{attempt} \vee \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$[\text{attempt} \wedge \neg\text{getF}] \Rightarrow \text{attempt}$$

Problem & Solution

- *It is not the case that if you attempt this exercise you will get an F. Therefore, you will attempt this exercise.*
  - *Translating the meaning of the sentence is not possible*

$$\neg[\text{attempt} \Rightarrow \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$\neg[\neg\text{attempt} \vee \text{getF}] \Rightarrow \text{attempt} \Leftrightarrow$$

$$\neg\text{attempt} \vee \text{getF} \vee \text{attempt}$$