Topic 4: Propagation[1]
(Version of 14th September 2017)

Pierre Flener

ASTRA Research Group
on Combinatorial Optimisation
Uppsala University
Sweden

Course 1DL441:
Combinatorial Optimisation using
Constraint Programming

[1] Based also on some material by Christian Schulte

Outline

UPPSALA UNIVERSITET

Outline

1. **Intuition**
   - Example 1
   - Example 2
   - Example 3

2. **Theory**
   - Propagator for One Constraint
   - Fixpoint of Multiple Propagators

Outline

Example (Agricultural experiment design, AED)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–	✓	–	✓
spelt	–	–	✓	✓	–	–	✓
wheat	–	–	✓	–	✓	✓	–

Constraints to be **satisfied**:

1. Equal sample size: Every grain is grown in 3 plots.
2. Equal growth load: Every plot grows 3 grains.
3. Balance: Every grain pair has 1 plot in common.

Instance: 7 grains, 7 plots, 3 plots/grain, 3 grains/plot, balance 1.
General term: balanced incomplete block design (BIBD).

Example (Agricultural experiment design, AED)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	1	1	1	0	0	0	0
corn	1	0	0	1	1	0	0
millet	1	0	0	0	0	1	1
oats	0	1	0	1	0	1	0
rye	0	1	0	0	1	0	1
spelt	0	0	1	1	0	0	1
wheat	0	0	1	0	1	1	0

Constraints to be **satisfied**:

1. Equal sample size: Every grain is grown in 3 plots.
2. Equal growth load: Every plot grows 3 grains.
3. Balance: Every grain pair has 1 plot in common.

Instance: 7 grains, 7 plots, 3 plots/grain, 3 grains/plot, balance 1.
General term: balanced incomplete block design (BIBD).

Example (BIBD *integer* model: ✓ $\rightsquigarrow$ 1 and – $\rightsquigarrow$ 0)

int: $nVarieties, nBlocks, sampleSize, blockSize, balance$
set of int: $Varieties = 1..nVarieties, Blocks = 1..nBlocks$
array$[Varieties, Blocks]$ **of var** $0..1 : BIBD$
satisfy
forall$(v$ **in** $Varieties)(sampleSize = $ **sum**$(BIBD[v, *]))$
forall$(b$ **in** $Blocks)(blockSize = $ **sum**$(BIBD[*, b]))$
forall$(v_1, v_2$ **in** $Varieties$ **where** $v_1 < v_2)$
$(balance = $ **sum**$(b$ **in** $Blocks)(BIBD[v_1, b] \cdot BIBD[v_2, b]))$

Example (Instance data for AED)

$nVarieties = 7, nBlocks = 7,$
$sampleSize = 3, blockSize = 3, balance = 1$

Example (Idea for another BIBD model)

barley	{plot1, plot2, plot3}
corn	{plot1, plot4, plot5}
millet	{plot1, plot6, plot7}
oats	{plot2, plot4, plot6}
rye	{plot2, plot5, plot7}
spelt	{plot3, plot4, plot7}
wheat	{plot3, plot5, plot6}

Constraints to be **satisfied**:

1. Equal sample size: Every grain is grown in 3 plots.
2. Equal growth load: Every plot grows 3 grains.
3. Balance: Every grain pair is grown in 1 common plot.

Example (BIBD *set* model: ☞ see Topic 9 for details)

int: $nVarieties, nBlocks, sampleSize, blockSize, balance$
set of int: $Varieties = 1..nVarieties, Blocks = 1..nBlocks$
array[$Varieties$] **of var set of** $Blocks$: $BIBD$
satisfy
forall(v **in** $Varieties$)($sampleSize = |BIBD[v]|$)
forall(b **in** $Blocks$)($blockSize =$ **count**($BIBD, b$))
forall(v_1, v_2 **in** $Varieties$ **where** $v_1 < v_2$)
($balance = |BIBD[v_1] \cap BIBD[v_2]|$)

Example (Instance data for AED)

$nVarieties = 7$, $nBlocks = 7$,
$sampleSize = 3$, $blockSize = 3$, $balance = 1$

Store after filling the first four rows

Example (BIBD *integer* model)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	?						
spelt							
wheat							

Store after filling the first four rows

Example (BIBD *integer* model)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	?						
spelt							
wheat							

But plot1 cannot grow rye as that would violate the second constraint (every plot grows 3 grains).

Store after filling the first four rows

Example (BIBD *integer* model)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–						
spelt							
wheat							

But plot1 cannot grow rye as that would violate the second constraint (every plot grows 3 grains).

UPPSALA UNIVERSITET

Store after filling the first four rows

Example (BIBD *integer* model)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–						
spelt							
wheat							

But plot1 cannot grow rye as that would violate the second constraint (every plot grows 3 grains). Actually, plot1 cannot grow oats, spelt, or wheat either, for the same reason, and this was already propagated when trying the search guess that plot1 grow millet!

Store after filling the first four rows

Example (BIBD *integer* model)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–						
spelt	–						
wheat	–						

But plot1 cannot grow rye as that would violate the second constraint (every plot grows 3 grains). Actually, plot1 cannot grow oats, spelt, or wheat either, for the same reason, and this was already propagated when trying the search guess that plot1 grow millet!

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	?					
spelt	–						
wheat	–						

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	?					
spelt	–						
wheat	–						

Guess: Let plot2 grow rye. (Strategy: ✓ guesses first.)

Continuing ...

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓					
spelt	–						
wheat	–						

Guess: Let plot2 grow rye. (Strategy: ✓ guesses first.)

Continuing ...

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓					
spelt	–						
wheat	–						

Propagation: plot2 cannot grow spelt and wheat as otherwise the second constraint (every plot grows 3 grains) would be violated for plot2.

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓					
spelt	–	–					
wheat	–	–					

Propagation: plot2 cannot grow spelt and wheat as otherwise the second constraint (every plot grows 3 grains) would be violated for plot2.

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓					
spelt	–	–					
wheat	–	–					

Propagation: plot3, plot4, and plot6 cannot grow rye as otherwise the third constraint (every grain pair is grown in 1 common plot) would be violated.

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–		–	
spelt	–	–					
wheat	–	–					

Propagation: plot3, plot4, and plot6 cannot grow rye as otherwise the third constraint (every grain pair is grown in 1 common plot) would be violated.

Continuing ...

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–		–	
spelt	–	–					
wheat	–	–					

Propagation: plot5 and plot7 must grow rye as otherwise the first constraint (every grain is grown in 3 plots) would be violated for rye.

UPPSALA UNIVERSITET

Continuing ...

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–	✓	–	✓
spelt	–	–					
wheat	–	–					

Propagation: plot5 and plot7 must grow rye as otherwise the first constraint (every grain is grown in 3 plots) would be violated for rye.

Continuing ...

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–	✓	–	✓
spelt	–	–					
wheat	–	–					

Propagation: plot3 must grow spelt and wheat as otherwise the second constraint (every plot grows 3 grains) would be violated for plot3.

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–	✓	–	✓
spelt	–	–	✓				
wheat	–	–	✓				

Propagation: plot3 must grow spelt and wheat as otherwise the second constraint (every plot grows 3 grains) would be violated for plot3.

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–	✓	–	✓
spelt	–	–	✓				
wheat	–	–	✓				

Common fixpoint reached: No more propagation possible.

Continuing . . .

Example (BIBD: AED partial assignment)

	plot1	plot2	plot3	plot4	plot5	plot6	plot7
barley	✓	✓	✓	–	–	–	–
corn	✓	–	–	✓	✓	–	–
millet	✓	–	–	–	–	✓	✓
oats	–	✓	–	✓	–	✓	–
rye	–	✓	–	–	✓	–	✓
spelt	–	–	✓	✓			
wheat	–	–	✓				

Guess: Let plot4 grow spelt. (Strategy: ✓ guesses first.)

Propagation: etc.

Outline

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a		1	2	3	4	5	6	7	8	9
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of a:

a		1	2	3	4	5	6	7	8	9
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of a:

a		1	2	3	4	5	6	7	8	9
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of b:

a			2		4		6		8	
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of b:

a			2		4		6		8	
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a			2		4		6		8	
b			2	3	4	5				

Keep propagator for $2 \cdot a + 4 \cdot b = 24$, as not subsumed: its constraint is not definitely true under the current store.

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $a + b = 9$: prune unsupported values of a:

a			2		4		6		8	
b			2	3	4	5				

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $a + b = 9$: prune unsupported values of a:

a			2		4		6		8	
b			2	3	4	5				

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $a + b = 9$: prune unsupported values of b:

a					4		6			
b			2	3	4	5				

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

State $a + b = 9$: prune unsupported values of b:

a					4		6			
b			2	3	4	5				

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a					4		6			
b				3		5				

Keep propagator for $a + b = 9$, as not subsumed:
its constraint is not definitely true under the current store.

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

Run $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of a:

a					4		6			
b				3		5				

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

Run $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of a:

a					4		6			
b				3		5				

UPPSALA UNIVERSITET

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

Run $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of b:

a							6			
b				3		5				

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

Run $2 \cdot a + 4 \cdot b = 24$: prune unsupported values of b:

a							6			
b				3		5				

UPPSALA UNIVERSITET

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a							6			
b				3						

Dispose of propagator for $2 \cdot a + 4 \cdot b = 24$, as subsumed: its constraint is definitely true under the current store.

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

Run $a + b = 9$: prune unsupported values of a:

a							6			
b				3						

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

Run $a + b = 9$: prune unsupported values of b:

a							6			
b				3						

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a							6			
b				3						

Dispose of propagator for $a + b = 9$, as subsumed:
its constraint is definitely true under the current store.

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a							6			
b				3						

No propagators are left: all solutions are found. No search!

Problem, Model, and Propagation

Example (Propagation to *Domain* Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$\begin{aligned} 2 \cdot a + 4 \cdot b &= 24 \\ a + b &= 9 \end{aligned}$$

a							6			
b				3						

This general propagation method works for all systems of constraints (linear or not, equalities or inequalities, etc), no matter how many constraints and decision variables.

Outline

Problem, Model, and Propagation

Example (Propagation to *Bounds*($*$) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

a		1	2	3	4	5	6	7	8	9
b	0	1	2	3	4	5	6	7	8	

UPPSALA UNIVERSITET

Problem, Model, and Propagation

Example (Propagation to *Bounds*($*$) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported bounds of a:

a		1	2	3	4	5	6	7	8	9
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Bounds*(∗) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported bounds of a:

a		1	2	3	4	5	6	7	8	9
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Bounds*($*$) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported bounds of b:

a			2	3	4	5	6	7	8	
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Bounds*($*$) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

State $2 \cdot a + 4 \cdot b = 24$: prune unsupported bounds of b:

a			2	3	4	5	6	7	8	
b	0	1	2	3	4	5	6	7	8	

Problem, Model, and Propagation

Example (Propagation to *Bounds*($*$) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

a			2	3	4	5	6	7	8	
b			2	3	4	5				

Keep the propagator for $2 \cdot a + 4 \cdot b = 24$, as not subsumed.

UPPSALA UNIVERSITET

Problem, Model, and Propagation

Example (Propagation to *Bounds*($*$) Consistency)

Find $a \in \{1, 2, \ldots, 9\}$ and $b \in \{0, 1, \ldots, 8\}$ such that

$$2 \cdot a + 4 \cdot b = 24$$

a			2	3	4	5	6	7	8	
b			2	3	4	5				

Keep the propagator for $2 \cdot a + 4 \cdot b = 24$, as not subsumed.

Some propagators are left: no solutions found yet. Search!

UPPSALA UNIVERSITET

Outline

1. Intuition
   - Example 1
   - Example 2
   - Example 3

2. **Theory**
   - Propagator for One Constraint
   - Fixpoint of Multiple Propagators

Solving

Systematic search, for a satisfaction problem:

1: propagate all constraints; **backtrack** if empty domain
2: **if** only fixed variables, **then** show solution & **backtrack**
3: **while** there is at least one scheduled propagator **do**
4: select unfixed variable, v, of current domain dom(v)
5: partition dom(v) using guesses (say $v = d$ & $v \neq d$, or $v > d$ & $v \leq d$, for a picked value $d \in$ dom(v))
6: **for each** guess: **recurse** upon adding it as constraint

For an optimisation problem: before backtracking at line 2 add the constraint that any next solution must be better.

Strategies:

- Line 4: variable selection: smallest domain, ...
- Line 5: value selection: maximum, median, ...
- Line 5: guess selection: equality, bisection, ...
- Tree exploration: depth-first search, ...

UPPSALA UNIVERSITET

Strength of Stores

Definition (Store strength comparison, denoted $s \prec t$)

Store s is (strictly) stronger than store t iff $s(v) \subseteq t(v)$ for every decision variable v, and $s(v) \subset t(v)$ for at least one decision variable v. (☞ Well-founded order over stores.)

Example (Store strength comparison)

Consider these stores for variables $\{x, y\}$ over $\{1, 2, 3\}$:

$$\begin{aligned} s_1 &= \{x \mapsto \{1, 2\}, y \mapsto \{2, 3\}\} \\ s_2 &= \{x \mapsto \{2\}, y \mapsto \{2, 3\}\} \\ s_3 &= \{x \mapsto \{2, 3\}, y \mapsto \{1, 2, 3\}\} \end{aligned}$$

Note: $s_2 \prec s_1$ and $s_2 \prec s_3$, but s_1 and s_3 are incomparable.

Outline

Constraint Propagator

Definition (Propagator)

A propagator p_c for a constraint c modifies a store so that:

- **Contraction:** The result store is stronger than or equal to ($\preceq$) the input store: $p_c(s) \prec s$ or $p_c(s) = s$, for any s.
- **Monotonicity:** Strength-ordered stores remain ordered: $s_1 \preceq s_2 \Rightarrow p_c(s_1) \preceq p_c(s_2)$, for any s_1 and s_2.
- **Solution identification:** For a solution to c, no domain is shrunk: $p_c(s) = s$, for any solution store s to c: fixpt!

Example (Domain-consistency propagator for $x \leq y$)

$$p_{x \leq y}(s) = \left\{ \begin{array}{l} x \mapsto \{n \in s(x) \mid n \leq \max(s(y))\}, \\ y \mapsto \{n \in s(y) \mid n \geq \min(s(x))\} \end{array} \right\}$$

$$p_{x \leq y}(\{x \mapsto \{1,3,5\}, y \mapsto \{0,2,4\}\}) = \{x \mapsto \{1,3\}, y \mapsto \{2,4\}\}$$

Justification for Monotonicity

Counter-example

Consider the non-monotonic propagator for constraint c

$$p_c(s) = \textbf{if } s(x) = \{1,2,3\} \textbf{ then } \{x \mapsto \{1\}\} \textbf{ else } s$$

and the stores $s_1 = \{x \mapsto \{1,2\}\}$ and $s_2 = \{x \mapsto \{1,2,3\}\}$:

$$s_1 \preceq s_2 \quad \text{but} \quad p_c(s_2) = \{x \mapsto \{1\}\} \preceq \{x \mapsto \{1,2\}\} = p_c(s_1)$$

The result stores could also be incomparable;
note that $\prec$ and $\preceq$ are partial ordering relations.

But propagation would be propagator-order-dependent:

$$p_c(p_{x<3}(s_2)) = \{x \mapsto \{1,2\}\} \neq \{x \mapsto \{1\}\} = p_{x<3}(p_c(s_2))$$

Consequences of Propagator Definition

- Property of propagation:
  - **Order independence:** Propagators may be invoked in any order: their weakest common fixpoint is unique. E.g., from $\{x, y \mapsto \{3, 4, 5\}\}$, the weakest fixpoint of $p_{x \geq y}$ and $p_{y>3}$ is $\{x, y \mapsto \{4, 5\}\}$, whereas a strongest fixpoint is a solution store, such as $\{x, y \mapsto \{5\}\}$.
- Properties of a propagator p_c for a constraint c:
  - **Solution preservation:** No solution is lost: if a solution to c is in a store before propagation, then it is in the result store after propagation of c: $d \in s \Rightarrow d \in p_c(s)$, for any store s and solution d to c.
  - **Non-solution identification:** For a non-solution to c, the domain of some decision variable becomes empty.

Idempotency of propagators is *not* required:

Every DC propagator is idempotent; a BC propagator may be non-idempotent: see Ex. 2.9 on p. 19 of Course Notes.

Terminology:

The objective of a propagator is to delete ~~the~~ unsupported values, according to a chosen consistency, from the domains of decision variables. In the literature, this deletion is also called pruning, filtering, contracting, or narrowing. If a domain loses its last value, then we say that there was a domain wipe-out and the propagator must fail.

Definition (Model)

A model of a CSP $\langle V, U, C \rangle$ is a tuple $\langle V, U, P \rangle$, where P is the set of propagators chosen for the constraints C.

Outline

Naïve Fixpoint Algorithm

Let $\langle V, U, P \rangle$ be a model where, without loss of generality, there is a common domain U for all decision variables of V.

Let $s_0 = \{v \mapsto U \mid v \in V\}$ be the initial store, where every decision variable v of V is mapped to the universe U.

Call to build the root of the search tree: Propagate(P, s_0).

function Propagate(R, s)
while $\exists q \in R : q(s) \precneqq s$ **do** // **variant:** s
  pick $q \in R : q(s) \precneqq s$
  $s := q(s)$
return s // **post:** s is the weakest common fixpoint of R

Toward More Realistic Propagation

Why is the previous algorithm naïve?

For the condition of its **while** loop:

- We do not maintain the set of propagators that are known to be at fixpoint.
- We may run a propagator that does not even depend in some sense on the propagator that was just run.

Hence we may run a propagator that cannot prune values.

Variables of a propagator:

Let var(p) denote the set of decision variables of the constraint implemented by propagator p:

- Running p has no effect on dom(v), for $v \in V \setminus \text{var}(p)$.
- Running p is independent of dom(v), for $v \in V \setminus \text{var}(p)$.

Variable-Directed Fixpoint Algorithm

Call to build the root of the search tree: Propagate(P, P, s_0).

function Propagate(R, Q, s)
while $Q \neq \emptyset$ **do** // **invariant:** every $p \in R \setminus Q$ is at fixpoint
// **variant:** $\langle s, |Q| \rangle$
  pick $q \in Q$ // prop.s of Q are possibly not at fixpoint
  $Q := Q \setminus \{q\}$
  $s' := q(s)$ // $s' \preceq s$
  $ModVars := \{v \in \text{var}(q) \mid s(v) \neq s'(v)\}$
  $DepProps := \{p \in R \mid \exists v \in \text{var}(p) : v \in ModVars\}$
  $Q := Q \cup DepProps$ // maybe $q \in Q$: optional idempot.
  $s := s'$
return s // **post:** s is the weakest common fixpoint of R

Toward Further Improved Propagation

Propagators signal status to avoid some useless runs:

- Propagator p is failed upon a domain wipe-out.
- Propagator p is subsumed (or entailed) by store s iff all stronger stores are fixpoints: $\forall s' \preceq s : p(s') = s'$. This status is an obligation when s is a solution store. Such a propagator can safely be disposed of in the model.
- Otherwise, if so, ideally signal that p is at fixpoint for s.
- It is always safe to signal that a propagator p is possibly not at fixpoint for the result store s.

Examples (Subsumption)

$p_{x \leq y}$ is subsumed by $\{x \mapsto \{1,3\}, y \mapsto \{3,5\}\}$, but not by $\{x \mapsto \{1,3,4\}, y \mapsto \{3,5\}\}$. A DC propagator of a unary constraint, like $x \in \{1,3,5\}$, is subsumed upon its first run.

Propagators with Status Message

Example (Domain-consistency propagator for $x \leq y$)

$$p_{x \leq y}(s) = \textbf{let } s' = \left\{ \begin{array}{l} x \mapsto \{n \in s(x) \mid n \leq \max(s(y))\}, \\ y \mapsto \{n \in s(y) \mid n \geq \min(s(x))\} \end{array} \right\} \textbf{ in}$$

$$\textbf{if } s'(x) = \emptyset \vee s'(y) = \emptyset \textbf{ then } \langle \text{Failed}, \emptyset \rangle$$

$$\textbf{else if } \max(s'(x)) \leq \min(s'(y)) \textbf{ then } \langle \text{Subsumed}, s' \rangle$$

$$\textbf{else } \langle \text{AtFixpt}, s' \rangle$$

Note that $\min(s(x))$ and $\max(s(y))$ do not change: hence s' is at least a fixpoint for $p_{x \leq y}$ and at best subsumes it!

Responsibility:

The burden of signalling, in reasonable runtime, a proper status message is on the programmer of a propagator.

UPPSALA UNIVERSITET

Propagator-Status-Directed Fixpoint Algo.

function Propagate(R, Q, s) // non-subsumed prop.s in R
while $Q \neq \emptyset$ **do** // **invariant: . . . ; variant: . . .**
  pick $q \in Q$
  $Q := Q \setminus \{q\}$
  $\langle m, s' \rangle := q(s)$ // $s' \preceq s$
  if $m =$ Failed **then return** $\langle R, \emptyset \rangle$ **end if**
  if $m =$ Subsumed **then** $R := R \setminus \{q\}$ **end if**
  $ModVars := \{v \in \text{var}(q) \mid s(v) \neq s'(v)\}$
  $DepProps := \{p \in R \mid \exists v \in \text{var}(p) : v \in ModVars\}$
  if $m =$ AtFixpt **then** $DepProps := DepProps \setminus \{q\}$ **end if**
  $Q := Q \cup DepProps$
  $s := s'$
return $\langle R, s \rangle$ // **post:** s is the weakest common fixpt of R

Toward Even Further Improved Propagation

Signalling *how* domains were modified:
Mutually exclusive modification events for each variable v:

1. None(v): the domain of v was not changed.
2. Failed(v): the domain of v was wiped out.
3. Fixed(v): the domain of v was pruned to a singleton.
4. Min(v): the lower bound of dom(v) was increased.
   Max(v): the upper bound of dom(v) was decreased.
5. Any(v): the domain of v was otherwise pruned.

Gecode: Min(v) and Max(v) are bundled into Bounded(v).

☞ It is often simple to decide whether a propagator remains at fixpoint depending on how another propagator prunes domains of decision variables they share: variable sharing is no longer the sole criterion for adding propagators to Q.

Propagator Conditions

Example (Domain-consistency propagator for $x \leq y$)

$$p_{x \leq y}(s) = \left\{ \begin{array}{l} x \mapsto \{n \in s(x) \mid n \leq \max(s(y))\}, \\ y \mapsto \{n \in s(y) \mid n \geq \min(s(x))\} \end{array} \right\}$$

☞ PropConds$(p_{x \leq y}) = \{\text{Min}(x), \text{Max}(y)\}$

Promise: If the propagator is at fixpoint, then it will remain at fixpoint, unless min(dom(x)) or max(dom(y)) changes.

Example (Domain-consistency propagator for $x \neq y$)

$$p_{x \neq y}(s) = \left\{ \begin{array}{l} x \mapsto s(x) \setminus \textbf{if } |s(y)| = 1 \textbf{ then } s(y) \textbf{ else } \emptyset, \\ y \mapsto s(y) \setminus \textbf{if } |s(x)| = 1 \textbf{ then } s(x) \textbf{ else } \emptyset \end{array} \right\}$$

☞ PropConds$(p_{x \neq y}) = \{\text{Fixed}(x), \text{Fixed}(y)\}$

Promise: If the propagator is at fixpoint, then it will remain at fixpoint, unless dom(x) or dom(y) becomes a singleton.

Assumptions

Responsibilities, under *Gecode*:

- The programmer of propagator p states PropConds(p).
- The solver computes as follows the set Conds(s, s') of propagator conditions raised by applying a propagator q to a store s, giving $s' = q(s)$:

Modification event	Conditions added to Conds(s, s')
Fixed(v)	Fixed(v), Bounded(v), Any(v)
Bounded(v)	Bounded(v), Any(v)
Any(v)	Any(v)
None(v)	(none)

- The solver schedules a propagator p (adds p to Q) if the conditions Conds(s, s') raised by propagator q intersect with the propagator conditions PropConds(p).

UPPSALA UNIVERSITET

Intuition

Example 1

Example 2

Example 3

Theory

Propagator for One Constraint

Fixpoint of Multiple Propagators

Status-and-Condition-Directed Fixpt Algo.

function Propagate(R, Q, s)
while $Q \neq \emptyset$ **do** // **invariant: . . . ; variant: . . .**
  pick $q \in Q$
  $Q := Q \setminus \{q\}$
  $\langle m, s' \rangle := q(s)$ // $s' \preceq s$
  if $m =$ Failed **then return** $\langle R, \emptyset \rangle$ **end if**
  if $m =$ Subsumed **then** $R := R \setminus \{q\}$ **end if**
  ~~$\mathit{ModVars} := \{v \in \mathrm{var}(q) \mid s(v) \neq s'(v)\}$~~
  $\mathit{DepProps} :=$
  $\{p \in R \mid \mathrm{Conds}(s, s') \cap \mathrm{PropConds}(p) \neq \emptyset\}$
  if $m =$ AtFixpt **then** $\mathit{DepProps} := \mathit{DepProps} \setminus \{q\}$ **end if**
  $Q := Q \cup \mathit{DepProps}$
  $s := s'$
return $\langle R, s \rangle$ // **post:** s is the weakest common fixpt of R

Yet Further Optimisations

Priorities: The set Q is implemented as a queue:

How to do "**pick** $q \in Q$"?

- According to cost: cheapest first
- According to expected impact: highest impact first
- In general: first-in first-out queue

Propagator rewriting:

Example

When all domain values for x are smaller than those for y, then the propagator for $\max(x, y) = z$ can be replaced by the propagator for $y = z$.

Further reading:

For a more formal treatment of all these issues, including proofs, see the Course Notes.