

Constraint Satisfaction Problems: Formulation, Arc Consistency & Propagation

Brian C. Williams
16.410-13
Session 10

Slides draw material from:
6.034 notes, by Tomas Lozano Perez
AIMA, by Stuart Russell & Peter Norvig
Constraint Processing, by Rina Dechter

1

Reading Assignments: Constraints

Readings:

- Lecture Slides (most material in slides only, READ ALL).
- AIMA Ch. 5 – Constraint Satisfaction Problems (CSPs)
- AIMA Ch. 24.4 pp. 881-884 – Visual Interpretation of line drawings as solving CSPs.

To find out more (optional reading):

- Constraint Processing, by Rina Dechter, Morgan-Kaufman, 2003.
 - Chapter 2 Constraint Networks
 - Chapter 3 Consistency-Enforcing and Constraint Propagation.

2

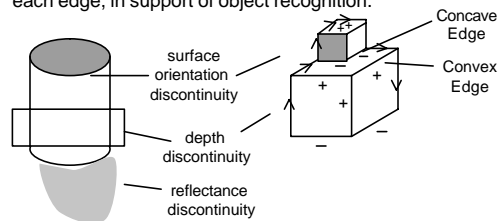
Outline

- Interpreting line diagrams
- Constraint satisfaction problems (CSP)
- Solving CSPs
 - Arc-consistency and propagation
 - Analysis of constraint propagation
 - Search (next lecture)
- Case study: Scheduling (appendix)

3

Line Labeling In Visual Interpretation

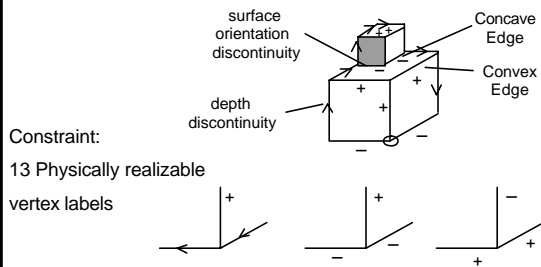
Problem: Given line drawing, assign consistent types to each edge, in support of object recognition.



Huffman Clowes (1971): Opaque, trihedral solids.
No surface marks.

4

Line Labeling In Visual Interpretation



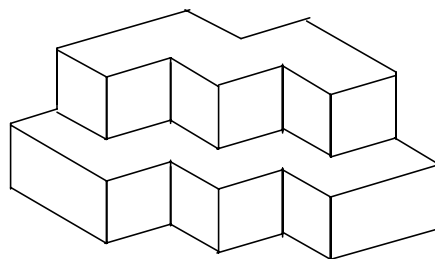
Constraint:
13 Physically realizable
vertex labels

Huffman Clowes (1971): Opaque, trihedral solids.
No surface marks.

Waltz (1972): labeling through local propagation.

5

Labeling must extend to complex objects



6

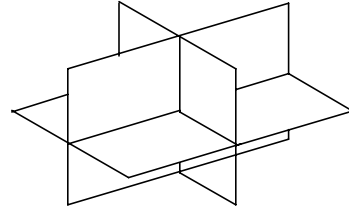
Line Interpretation Simplifying Assumptions

1. Limited line interpretations:
No shadows or cracks.
2. Three-faced vertices:
Intersection of exactly three object faces (e.g., no pyramid tops).
3. General position:
Small perturbations of selected viewing points can not lead to a change in junction type.

7

Deriving all possible junction types

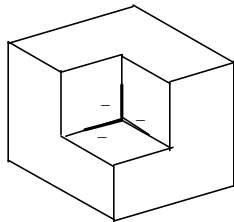
- Three face vertex divide space into octants.
 - May not be at right angles.
- Consider all possible fillings of octants, viewed from all empty octants.



8

Deriving all possible junction types

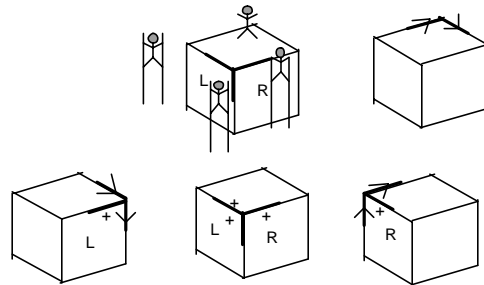
- Viewing seven filled octants.



9

Deriving all possible junction types

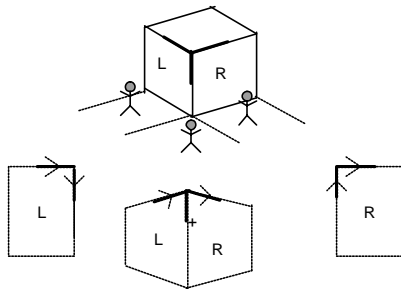
- Viewing one filled octant from empty the upper octants.



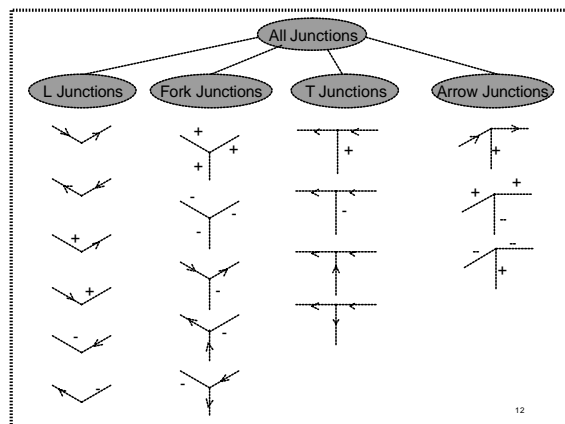
10

Deriving all possible junction types

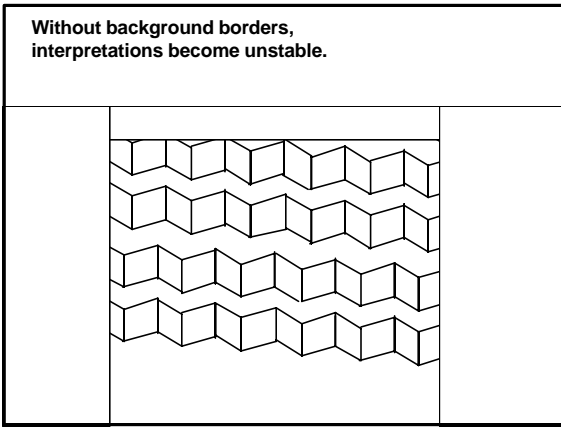
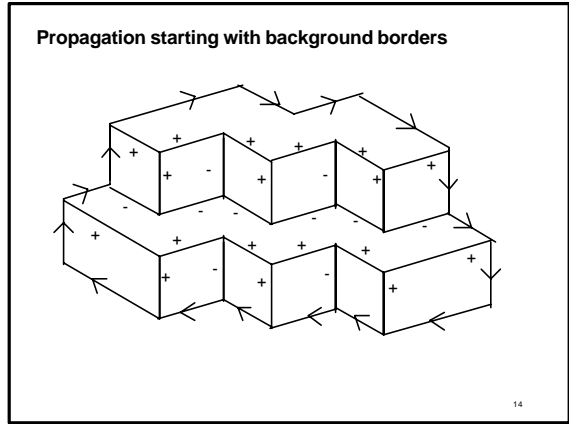
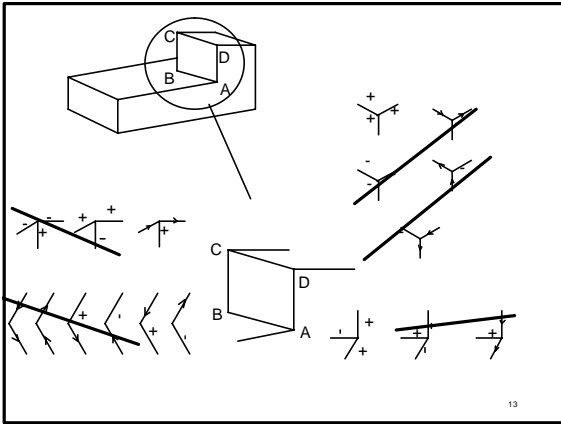
- Viewing one filled octant from the empty lower octants.



11



12



- Outline**
- Interpreting line diagrams
 - Constraint satisfaction problems (CSP)
 - Solving CSPs
 - Arc-consistency and propagation
 - Analysis of constraint propagation
 - Search (next lecture)
 - Case study: Scheduling (appendix)

Constraint Satisfaction Problems

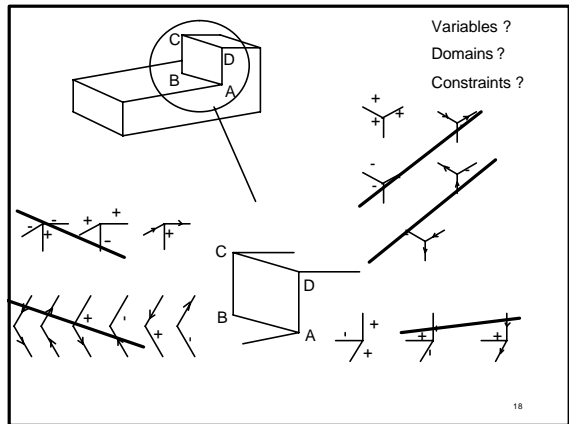
4 Queens Problem:
Place 4 queens on a 4x4 chessboard so that no queen can attack another.

How do we formulate?

Variables Chessboard positions

Domains Queen 1-4 or blank

Constraints Two positions on a line (vertical, horizontal, diagonal) cannot both be Q



Constraint Satisfaction Problem (CSP)

Input: A Constraint Satisfaction Problem is a triple $\langle V, D, C \rangle$, where:

- V is a set of variables V_i
- D is a set of variable domains,
 - The domain of variable V_i is denoted D_i
- C is a set of constraints on assignments to V
 - Each constraint $C_i = \langle S_i, R_i \rangle$ specifies allowed variable assignments.
 - S_i , the constraint's scope, is a subset of variables V .
 - R_i the constraint's relation, is a set of assignments to S_i .

Output: A full assignment to V , from elements of its V 's domain, such that all constraints in C are satisfied.

Example: "Provide one A and two B's."

- $V = \{A, B\}$, each with domain $D_i = \{1, 2\}$
- $C = \{ \langle \{A, B\}, \{ \langle 1, 2 \rangle, \langle 1, 1 \rangle \} \rangle, \langle \{A, B\}, \{ \langle 1, 2 \rangle, \langle 2, 2 \rangle \} \rangle \}$
- Output: $\langle 1, 2 \rangle$ (for example)

19

Constraint Satisfaction Problem (CSP)

Input: A Constraint Satisfaction Problem is a triple $\langle V, D, C \rangle$, where:

- V is a set of variables V_i
- D is a set of variable domains,
 - The domain of variable V_i is denoted D_i
- C is a set of constraints on assignments to V
 - Each constraint $C_i = \langle S_i, R_i \rangle$ specifies allowed variable assignments.
 - S_i , the constraint's scope, is a subset of variables V .
 - R_i the constraint's relation, is a set of assignments to S_i .

Output: A full assignment to V , from elements of its domain, such that all constraints in C are satisfied.

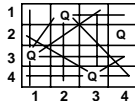
Example: "Provide one A and two B's."

- $V = \{A, B\}$, each with domain $D_i = \{1, 2\}$
 - $C = \{C_{AB}\}$
 - $C_{AB} = \{ \langle 1, 2 \rangle, \langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 2, 2 \rangle \}$
 - Output: $\langle 1, 2 \rangle$
- Conventions:
Scope in subscript
One constraint per scope

20

Good Encodings Are Essential: 4 Queens

4 Queens Problem:
Place 4 queens on a 4x4 chessboard so that no queen can attack another.



How big is the encoding?

Variables Chessboard positions

Domains Queen 1-4 or blank

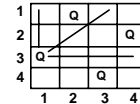
Constraints Two positions on a line (vertical, horizontal, diagonal) cannot both be Q

What is a better encoding?

21

Good Encodings Are Essential: 4 Queens

Place queens so that no queen can attack another.



What is a better encoding?

- Assume one queen per column.
- Determine what row each queen should be in.

Variables Q_1, Q_2, Q_3, Q_4

Domains $\{1, 2, 3, 4\}$

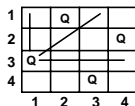
Constraints $Q_i \neq Q_j$ On different rows
 $|Q_i - Q_j| \neq |i - j|$ Stay off the diagonals

Example: $C_{1,2} = \{(1,3) (1,4) (2,4) (3,1) (4,1) (4,2)\}$

22

Good Encodings Are Essential: 4 Queens

Place queens so that no queen can attack another.



Variables Q_1, Q_2, Q_3, Q_4

Domains $\{1, 2, 3, 4\}$

Constraints $Q_i \neq Q_j$ On different rows
 $|Q_i - Q_j| \neq |i - j|$ Stay off the diagonals

Example: $C_{1,2} = \{(1,3) (1,4) (2,4) (3,1) (4,1) (4,2)\}$

What is C_{13} ?

23

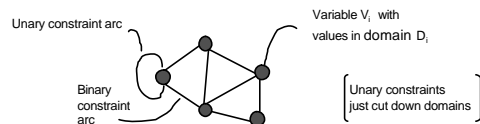
A general class of CSPs

Finite Domain, Binary CSPs

- each constraint relates at most two variables.
- each variable domain is finite.
- all n-ary CSPs reducible to binary CSPs.

Depict as a Constraint Graph

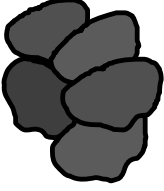
- Nodes are variables.
- Arcs are binary constraints.



24

Example: CSP Classic - Graph Coloring

Pick colors for map regions, without coloring adjacent regions with the same color



Variables regions

Domains allowed colors

Constraints adjacent regions must have different colors

25

Outline

- Constraint satisfaction problems (CSP)
- Solving CSPs
 - Arc-consistency and propagation
 - Analysis of constraint propagation
 - Search (next lecture)
- Case study: Scheduling (appendix)

26

Good News / Bad News

Good News

- very general & interesting family of problems.
- Problem formulation extensively used in autonomy and aerospace applications.

Bad News includes NP-Hard (intractable) problems

27

Solving CSPs

Solving CSPs involves some combination of:

1. Inference (e.g., Constraint propagation)
 - Partially solves, by eliminating values that can't be part of any solution.
 - By making explicit implicit constraints.
2. Search
 - Explores alternate valid assignments.

The problem that Waltz constraint propagation solves for visual interpretation is, in general, called arc-consistency and the algorithm is called AC-3.

28

Directed Arc Consistency

Directed arc consistency eliminates values of each variable domain that can never satisfy a particular constraint (an arc).

$$\begin{array}{ccc}
 x_i & \text{---} & x_j \\
 = & & \\
 \{1,2,3\} & & \{1,2\}
 \end{array}$$

- **Definition:** A directed arc $\langle x_i, x_j \rangle$ is arc consistent if
 - For every a_i in D_i , there exists some a_j in D_j such that assignment $\langle a_i, a_j \rangle$ satisfies constraint C_{ij} .
 - That is, " $\forall a_i \in D_i, \exists a_j \in D_j$ such that $\langle a_i, a_j \rangle \in C_{ij}$ " where
 - " \forall " denotes "for all"
 - " \exists " denotes "there exists"
 - " \in " denotes "in"

29

Directed Arc Consistency

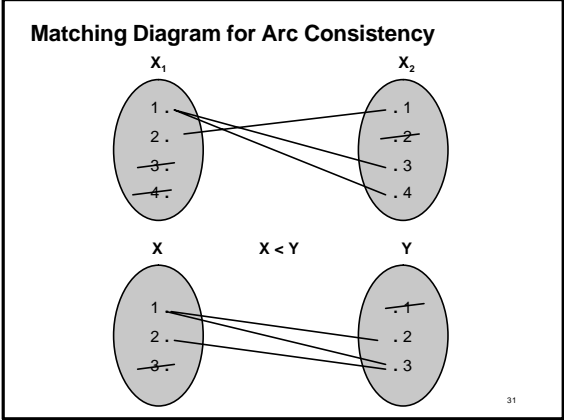
Directed arc consistency eliminates values of each variable domain that can never satisfy a particular constraint (an arc).

$$\begin{array}{ccc}
 x_i & \text{---} & x_j \\
 = & & \\
 \{1,2,3\} & & \{1,2\}
 \end{array}$$

- **Definition:** Directed arc $\langle x_i, x_j \rangle$ is arc consistent if
 - " $\forall a_i \in D_i, \exists a_j \in D_j$ such that $\langle a_i, a_j \rangle \in C_{ij}$ "

Example: Given: Variables x_1 and x_2 , each with Domain $\{1,2,3,4\}$
 Constraint: $C_{12} = \{(1,3) (1,4) (2,1)\}$
 What is the result of arc consistency?

30



Directed arc consistency procedure Revise

Definition: Directed arc $\langle x_i, x_j \rangle$ is arc consistent if

- " $a_i \in D_i, \exists a_j \in D_j$ such that $\langle a_i, a_j \rangle \in C_{ij}$

Revise (V_i, V_j)

Input: A subnetwork of a CSP, defined by two variables $X = \{x_i, x_j\}$, a distinguished variable x_i , domains D_i and D_j , and constraint R_{ij} .

Output: D_i , such that x_i is arc-consistent relative to x_j .

1. **for** each $a_i \in D_i$
2. **if** there is no $a_j \in D_j$ such that $\langle a_i, a_j \rangle \in R_{ij}$
3. **then** delete a_i from D_i .
4. **endif**
5. **Endfor**

Complexity of Revise?
 $= O(k^2)$ where $k = \max_i |D_i|$

32

Full Arc Consistency via Constraint Propagation

Directed arc consistency eliminates values of each variable domain that can never satisfy a particular constraint (an arc).

- Directed arc $\langle x_i, x_j \rangle$ is arc consistent if
" $a_i \in D_i, \exists a_j \in D_j$ such that $\langle a_i, a_j \rangle \in C_{ij}$

Constraint propagation: To achieve directed arc consistency over CSP:

1. For every arc C_{ij} in CSP, with tail domain D_i , call Revise:
(delete every value $a_i \in D_i$ that fails directed arc consistency on C_{ij} .)
2. Repeat until quiescence:
 - If an element was deleted from D_i , then
 - Repeat Step 1 (AC-1)

33

Full Arc-Consistency via AC-1

AC-1(CSP)

Input: A network of constraints CSP = $\langle X, D, C \rangle$.

Output: CSP', the largest arc-consistent subset of CSP.

1. **repeat**
2. **for** every $C_{ij} \in C$,
3. Revise(x_i, x_j)
4. Revise(x_j, x_i)
5. **endif**
6. **until no domain is changed.**

Complexity of AC-1?
 $= O(nk^2 \cdot e^k) = O(enk^3)$ where $k = \max_i |D_i|$

34

Full Arc Consistency via Constraint Propagation

Directed arc consistency eliminates values of each variable domain that can never satisfy a particular constraint (an arc).

- Directed arc $\langle x_i, x_j \rangle$ is arc consistent if
" $a_i \in D_i, \exists a_j \in D_j$ such that $\langle a_i, a_j \rangle \in C_{ij}$

Constraint propagation: To achieve directed arc consistency over CSP:

1. For every arc C_{ij} in CSP, with tail domain D_i , call Revise:
(delete every value $a_i \in D_i$ that fails directed arc consistency on C_{ij} .)
2. Repeat until quiescence:
 - If an element was deleted from D_i , then
 - Repeat Step 1 (AC-1) OR
 - Check directed arc consistency for each arc with head D_j (AC-3).
 - (Maintain arcs to be checked on FIFO queue, with no duplicates).

35

Full Arc-Consistency via AC-3

AC-3(CSP)

Input: A network of constraints CSP = $\langle X, D, C \rangle$.

Output: CSP', the largest arc-consistent subset of CSP.

1. **for** every $C_{ij} \in C$,
2. *queue* ? *queue* $\cup \{ \langle x_i, x_j \rangle, \langle x_j, x_i \rangle \}$
3. **endif**
4. **while** *queue* ? { }
5. select and delete arc $\langle x_i, x_j \rangle$ from *queue*
6. Revise(x_i, x_j)
7. **if** Revise(x_i, x_j) caused a change in D_i ,
8. **then** *queue* ? *queue* $\cup \{ \langle x_i, x_j \rangle \mid k ? i, k ? j \}$
9. **endif**
10. **endwhile**

Complexity of AC-3?
 $= O(ek^3)$ where $k = \max_i |D_i|$

36

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Each undirected constraint arc denotes two directed constraint arcs.

37

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted

Arcs to examine

$V_1 - V_2, V_1 - V_3, V_2 - V_3$

- Introduce queue of arcs to be examined.
- Start by adding all arcs to the queue.

38

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 > V_2$	

Arcs to examine

$V_2 > V_1, V_1 - V_3, V_2 - V_3$

- Delete unmentioned tail values
- $V_i - V_j$ denotes two arcs, between V_i and V_j .
- $V_i > V_j$ denotes an arc from V_j and V_i .

39

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 > V_2$	none

Arcs to examine

$V_2 > V_1, V_1 - V_3, V_2 - V_3$

- Delete unmentioned tail values
- $V_i - V_j$ denotes two arcs, between V_i and V_j .
- $V_i > V_j$ denotes an arc from V_j and V_i .

40

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 > V_2$	none
$V_2 > V_1$	

Arcs to examine

$V_1 - V_3, V_2 - V_3$

- Delete unmentioned tail values
- $V_i - V_j$ denotes two arcs, between V_i and V_j .
- $V_i > V_j$ denotes an arc from V_j and V_i .

41

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 > V_2$	none
$V_2 > V_1$	none

Arcs to examine

$V_1 - V_3, V_2 - V_3$

- Delete unmentioned tail values
- $V_i - V_j$ denotes two arcs, between V_i and V_j .
- $V_i > V_j$ denotes an arc from V_j and V_i .

42

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none

Arcs to examine
 $V_1 - V_3$ $V_2 - V_3$

• Delete unmentioned tail values • $V_i - V_j$ denotes two arcs, between V_i and V_j
 • $V_i > V_j$ denotes an arc from V_i and V_j

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 > V_3$	

Arcs to examine
 $V_3 > V_1$ $V_2 - V_3$

• Delete unmentioned tail values • $V_i - V_j$ denotes two arcs, between V_i and V_j
 • $V_i > V_j$ denotes an arc from V_i and V_j

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 > V_3$	$V_1(G)$

Arcs to examine
 $V_3 > V_1$ $V_2 - V_3$

IF An element of a variable's domain is removed,
 THEN add all arcs to that variable to the examination queue

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 > V_3$	$V_1(G)$

Arcs to examine
 $V_3 > V_1$ $V_2 - V_3$ $V_2 > V_1$ ~~$V_2 > V_1$~~

IF An element of a variable's domain is removed,
 THEN add all arcs to that variable to the examination queue

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 > V_3$	$V_1(G)$
$V_3 > V_1$	

Arcs to examine
 $V_2 - V_3$ $V_2 > V_1$

• Delete unmentioned tail values
 IF An element of a variable's domain is removed,
 THEN add all arcs to that variable to the examination queue

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 > V_3$	$V_1(G)$
$V_3 > V_1$	none

Arcs to examine
 $V_2 - V_3$ $V_2 > V_1$

• Delete unmentioned tail values
 IF An element of a variable's domain is removed,
 THEN add all arcs to that variable to the examination queue

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Different-color constraint

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 > V_3$	

Arcs to examine

$V_3 > V_3, V_2 > V_1$

- Delete unmentioned tail values

IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

49

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Different-color constraint

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 > V_3$	$V_2(G)$

Arcs to examine

$V_3 > V_3, V_2 > V_1$

- Delete unmentioned tail values

IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

50

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Different-color constraint

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 > V_3$	$V_2(G)$

Arcs to examine

$V_3 > V_2, V_2 > V_1, V_1 > V_2$

- Delete unmentioned tail values

IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

51

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Different-color constraint

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 > V_3$	$V_2(G)$
$V_3 > V_2$	

Arcs to examine

$V_2 > V_1, V_1 > V_2$

- Delete unmentioned tail values

IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

52

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Different-color constraint

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 > V_3$	$V_2(G)$
$V_3 > V_2$	none

Arcs to examine

$V_2 > V_1, V_1 > V_2$

- Delete unmentioned tail values

IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

53

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Different-color constraint

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 > V_1$	

Arcs to examine

$V_1 > V_2$

- Delete unmentioned tail values

IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

54

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 > V_1$	none

Arcs to examine

• Delete unmentioned tail values
IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

55

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 > V_1$	none
$V_1 > V_2$	

Arcs to examine

• Delete unmentioned tail values
IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

56

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 > V_1$	none
$V_1 > V_2$	$V_1(R)$

Arcs to examine

• Delete unmentioned tail values
IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

57

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 > V_1$	none
$V_1 > V_2$	$V_1(R)$

Arcs to examine

• Delete unmentioned tail values
IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

58

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 - V_1$	$V_1(R)$
$V_2 > V_1$	
$V_3 > V_1$	

Arcs to examine

• Delete unmentioned tail values
IF An element of a variable's domain is removed,
THEN add all arcs to that variable to the examination queue

59

Constraint Propagation Example AC-3

Graph Coloring
Initial Domains

Arc examined	Value deleted
$V_1 - V_2$	none
$V_1 - V_3$	$V_1(G)$
$V_2 - V_3$	$V_2(G)$
$V_2 - V_1$	$V_1(R)$
$V_2 > V_1$	none
$V_3 > V_1$	none

IF examination queue is empty
THEN arc (pairwise) consistent

60

Outline

- Constraint satisfaction problem (CSPs)
- Solving CSPs
 - Arc-consistency and propagation
 - Analysis of constraint propagation
 - Search (next lecture)
- Case study: Scheduling (appendix)

61

What is the Complexity of Constraint Propagation?

Assume:

- Domains are of size at most k
- There are e binary constraints.

Which is the correct complexity?

1. $O(k^e)$
2. $O(ek^e)$
3. $O(ek^3)$
4. $O(e^3)$

62

Complexity of Constraint Propagation

Assume:

- Domains are of size at most k
- There are e binary constraints.

Complexity:

- Straight forward arc consistency is $O(ek^3)$
 - There are $2 * e$ arcs to check
 - Verifying arc consistency takes $O(k^2)$ for each arc.
 - An arc is checked at most $O(k)$ times.

63

Full Arc-Consistency via AC-3

AC-3(CSP)

Input: A network of constraints $CSP = \langle X, D, C \rangle$.

Output: CSP' , the largest arc-consistent subset of CSP.

1. **for** every $c_i \in C$, $O(e)$
2. $queue \leftarrow queue \cup \{ \langle x_i, x_j \rangle \}$
3. **endfor**
4. **while** $queue \neq \{ \}$
5. select and delete arc $\langle x_i, x_j \rangle$ from $queue$
6. Revise(x_i, x_j) $O(k^2)$
7. **if** Revise(x_i, x_j) caused a change in D_j , $*O(ek)$
8. **then** $queue \leftarrow queue \cup \{ \langle x_i, x_k \rangle \mid k \neq i, k \neq j \}$
9. **endif**
10. **endwhile**

Complexity of AC-3?

$= O(ek^3)$ where $k = \max_i |D_i|$

64

Is arc consistency sound and complete?

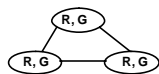
An *arc consistent solution* is any selection of values for every variable from the arc consistent domains.

Completeness: Does arc consistency rule out any valid solutions?

- Yes
- No

Soundness: Is every arc-consistent solution a solution to the CSP?

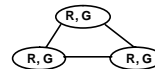
- Yes
- No



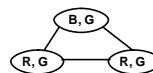
65

Arc consistency doesn't rule out all infeasible solutions

Graph Coloring



arc consistent, but NO solutions.



arc consistent, but 2 solutions, not 8.

B, R, G
B, G, R

66

Next Lecture: To Solve CSPs we combine arc consistency and search

1. Arc consistency (Constraint propagation),
 - Eliminates values that are shown locally to not be a part of any solution.
2. Search
 - Explores consequences of committing to particular assignments.

Methods Incorporating Search:

- Standard Search
- BackTrack search (BT)
- BT with Forward Checking (FC)
- Dynamic Variable Ordering (DV)
- Iterative Repair
- Backjumping (BJ)

67

Outline

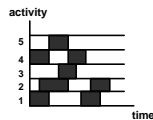
- Constraint satisfaction problem (CSPS)
- Solving CSPs
 - Arc-consistency and propagation
 - Analysis of constraint propagation
 - Search (next lecture)
- Case study: Scheduling (appendix)

68

Real World Example: Scheduling as a CSP

Choose time for activities:

- Observations on Hubble telescope.
- Jobs performed on machine tools.
- Terms to take required classes.



Variables are activities

Domains sets of possible start times (or “chunks” of time)

- Constraints**
1. Activities that use the same resource cannot overlap in time, and
 2. Preconditions are satisfied.

69

Case Study: Course Scheduling

Given:

- 40 required courses (8.01, 8.02, 6.840), and
- 10 terms (Fall 1, Spring 1, , Spring 5).

Find: a legal schedule.

- Constraints
- Pre-requisites satisfied,
 - Courses offered only on certain terms,
 - Limited number of courses taken per term (say 4), and
 - Avoid time conflicts.

Note, traditional CSPs are not for expressing (soft) preferences e.g. minimize difficulty, balance subject areas, etc.

But see recent work on semi-ring CSPs!

70

Alternative formulations for variables & values

VARIABLES

DOMAINS

- A. 1 var per Term
(Fall 1) (Spring 1)
(Fall 2) (Spring 2) . . .

All legal combinations of 4 courses,
all offered during that term.

- B. 1 var per Term-Slot
subdivide each term
into 4 course slots:
(Fall 1, 1) (Fall 1, 2)
(Fall 1, 3) (Fall 1, 4)

All courses offered during that term.

- C. 1 var per Course

Terms or term -slots.

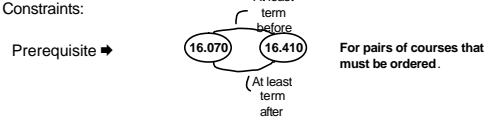
Term-slots make it easier to express the constraint limiting the number of courses per term.

71

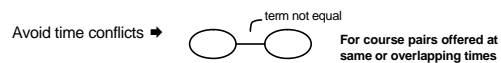
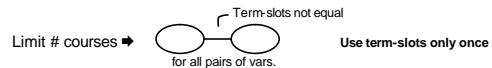
Encoding Constraints

Assume: Variables = Courses, Domains = term-slots

Constraints:



Courses offered only during certain terms → **Filter domain**



72