ECE 566

Interval Constraint Networks

Constraint Networks with interval and non-interval values:

In the CSP techniques we discussed,We assumed that the values of the variables are discrete or can be instantiated discretely.

Interesting problems - When the variables can take real values.

Instantiation impossible - Infinite values.

We use the concepts developed from Interval Mathematics.

Example Applications:

(1) Design of a Helical Spring:

Method is used to get feasible solutions by changing the parameters like Active number of coils,Shear modulus which do not have interval values and keeping the parameters like wire diameter and coil diameter which have interval values constant.

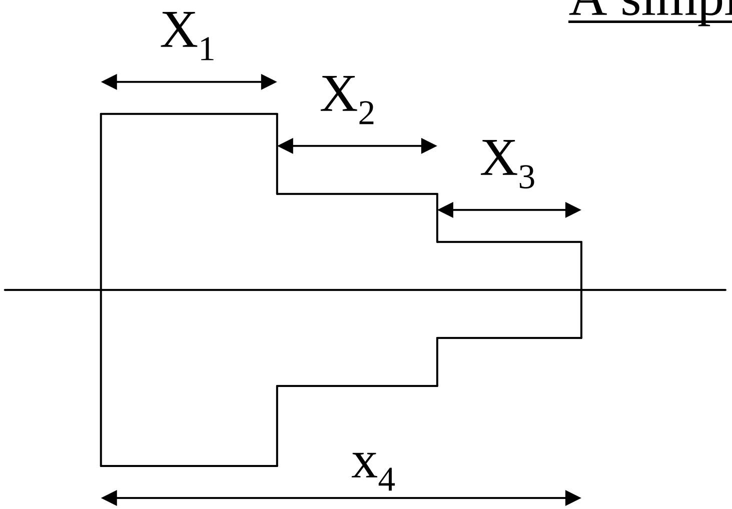
(2) Robotics:

To determine a path with a given final uncertainty in the final orientation and location of the end-effector.

(3) Active Vision:To determine the position of the active vision head so that the displacement errors are within the predetermined limits.

Example: tolerance design.

A simple problem:



X_1 , X_2 and X_3 may be the individual part tolerances and X_4 can be the tolerance on the total length.

Given that dimension X_4 should be within [62,68], can the range for the other tolerances be

X_1 =[1,5] (i.e. 3±2), X_2 =[8,10] and X_3 =[50,55]?

This can be answered by asking, is the tolerance of X_4 correct?

If the resulting tolerance for X_4 is incorrect, what should we assign X_1 , X_2 , and$_{,}$ X_3, such that X_4 lies within the given tolerance(functional requirement)?

Interval Arithmetic

Notation:

X,Y denote Interval valued variables.

$X \Rightarrow [x_{low}, x_{up}]$

Operations on Interval valued variables: Z= X ¦ Y, ¦ is a mathematical operation

(1) Addition:

$X+Y \Rightarrow [x_{low}, x_{up}] + [y_{low}, y_{up}] = [x_{low}+ y_{low} , x_{up}+ y_{up}]$

Example:

[2,3] + [1,2] = [3,5]

(2) Subtraction

$X-Y \Rightarrow [x_{low}, x_{up}] - [y_{low}, y_{up}] = [x_{low}- y_{up} , x_{up}- y_{low}]$

Example: [2,3] - [1,2] = [0,2]

(3) Multiplication:

$X*Y \Rightarrow [x_{low}, x_{up}] * [y_{low}, y_{up}] = [\mathrm{Min}\ \{x_{low} * y_{low} , x_{low} * y_{up}, x_{up} * y_{low}, x_{up} * y_{up}\},$

$\mathrm{Max}\{x_{low} * y_{low} , x_{low} * y_{up}, x_{up} * y_{low}, x_{up} * y_{up} \}]$

Example: [-2,3]*[1,2] = [-4, 6]

(4) Division

X / Y = X * [1/Y]

Example: [-2,3] / [1,2] =[-2,3]

We can derive such expressions for other piecewise monotonic functions

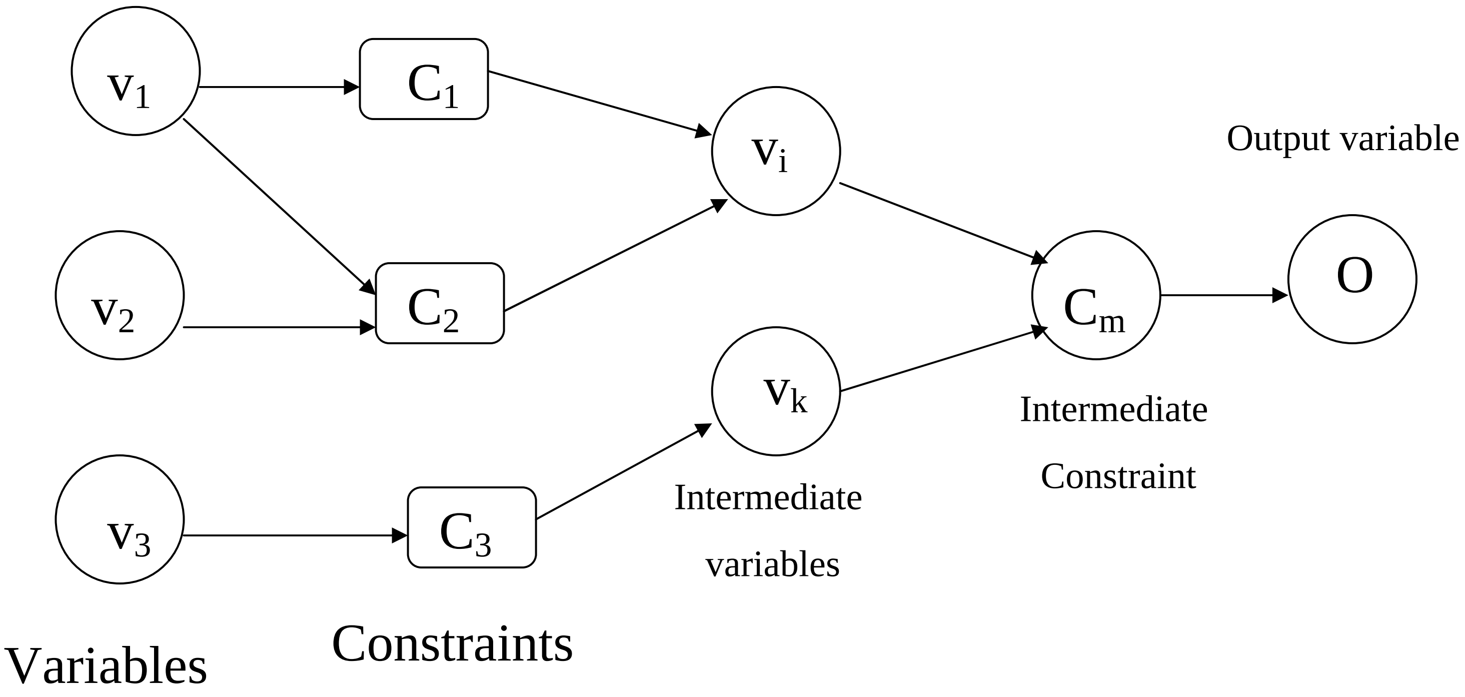
like logarithmic, exponential, etc.

Consistency of a constraint:For every value of the input variables, There are values for the output variables such that the constraint is satisfied. Input and Output variables are differentiated here.

Example: Start with the allowed values $C=[1,5]$, $F=[27,35]$, given constraint equation: $F=1.8*C+32$.

The constraint will be consistent if $C=[1,1.666]$ $F=[27,35]$
Satisfaction of the CSP: The CSP is satisfied if all constraints are consistent.

New Constraint network representation: Here the constraints are also represented by nodes in addition to the variables,each constraint node has multiple inputs and a single output. There is Differentiation between input and output.



New Constraint network

Notation in the new Constraint Network representation:

Representation of the Constraint Network:

A constraint network is a double, CN(**X**,C) where **X** represents a set of variables, $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n\}$ and C represents a set of constraints, $\{C_1, C_2, \ldots, C_m\}$.

The nodes in the constraint network represent variables or constraints.

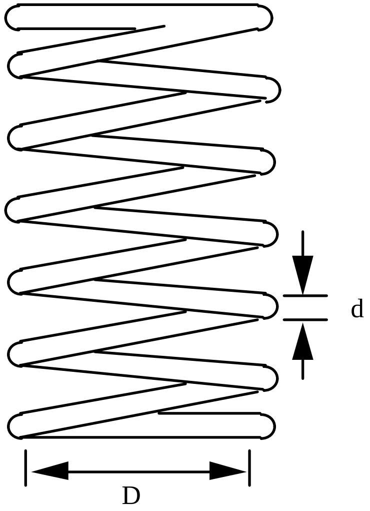
Representation of Constraints:

A constraint has multiple inputs and a single output (MISO) and can be represented as a triple, C_i(U,k, f()). U is the set of input variables and k is the output variable for the constraint C_i. f() represents the *constraint function* for C_i. For example, if U = {X1,X2} and k = X3, and the constraint function of C_i is add(), then C_i represents $\mathbf{X}_3$ = add($\mathbf{X}_1$,$\mathbf{X}_2$) (or equivalently $\mathbf{X}_3 = \mathbf{X}_1 + \mathbf{X}_2$).

Design of a Helical Spring:

Consider a Spring, the governing equation for a spring is:

$$K = \frac{Gd^4}{8(D+d)^3 N}$$



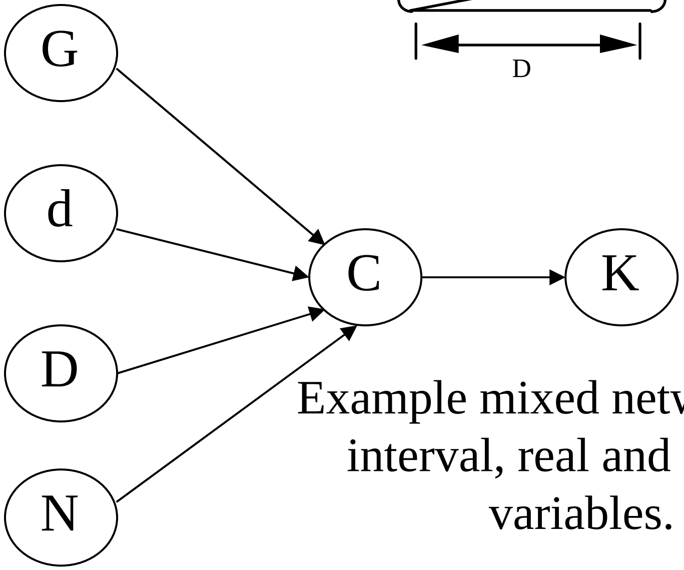
Where: K is the Spring Constant.

G : Shear modulus of the material of the spring.

D: Coil diameter of the spring.

d: wire diameter of the spring.

N: Number of coils.



Example mixed network with interval, real and integer variables.

Problem:

What is the right number of coils?

Is the material right?

Forward Propagation in the Interval constraint network:

Based on the values of the input variables, we calculate the values of the output variables and check whether all the constraints are consistent.

Backward Propagation in the Interval Constraint network:

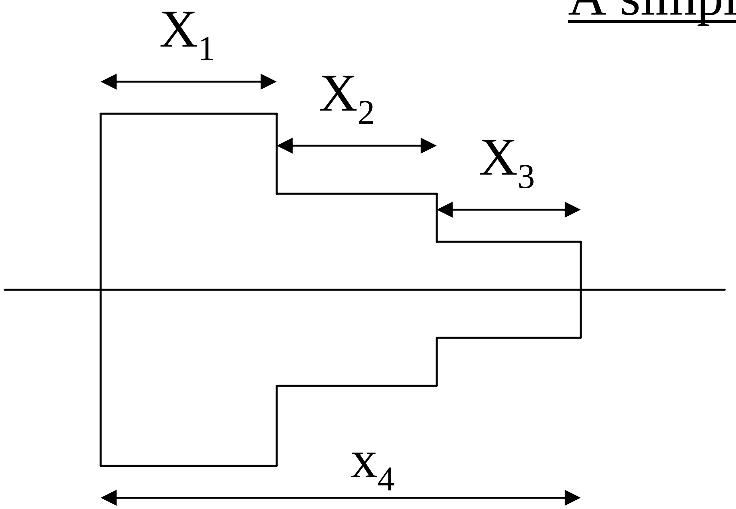
If any of the constraints are inconsistent, we change the values of the input variables based on the magnitude of the inconsistency. We may restrict changes to only some particular types of input variables.

For example: Only the real or integer valued variables.

This is dictated by the application we have on hand.

Example: tolerance design.

A simple problem:



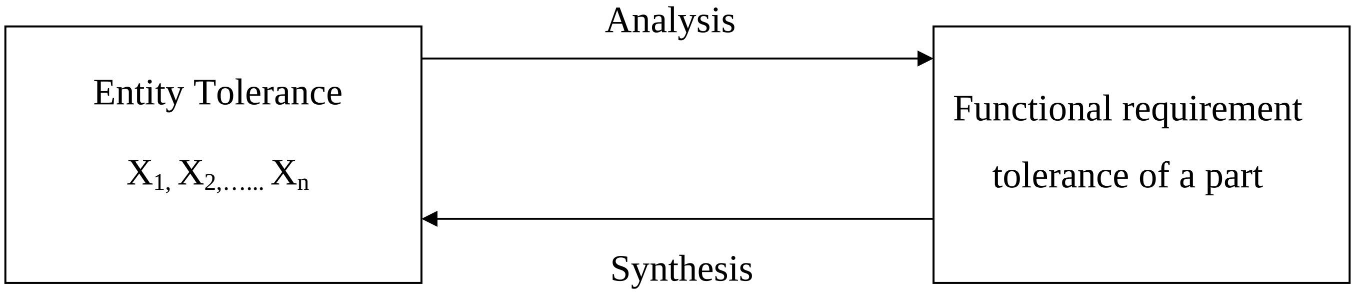
X_1 , X_2 and X_3 may be the individual part tolerances and X_4 can be the tolerance on the total length.

Given that dimension X_4 should be within [62,68], can the other tolerances be X_1 =[1,5] (i.e. 3±2), X_2 =[8,10] and X_3 =[50,55]?

This can be answered by asking, is the tolerance of X_4 correct? (Analysis)

If the resulting tolerance for X_4 is incorrect, what should we assign X_1 , X_2 and X_3, such that X_4 lies within the given tolerance(functional requirement)? (Synthesis)

Analysis and Synthesis:



Analysis

Entity Tolerance

$X_{1,}\ X_{2,\ldots\ldots}\ X_n$

Functional requirement

tolerance of a part

Synthesis

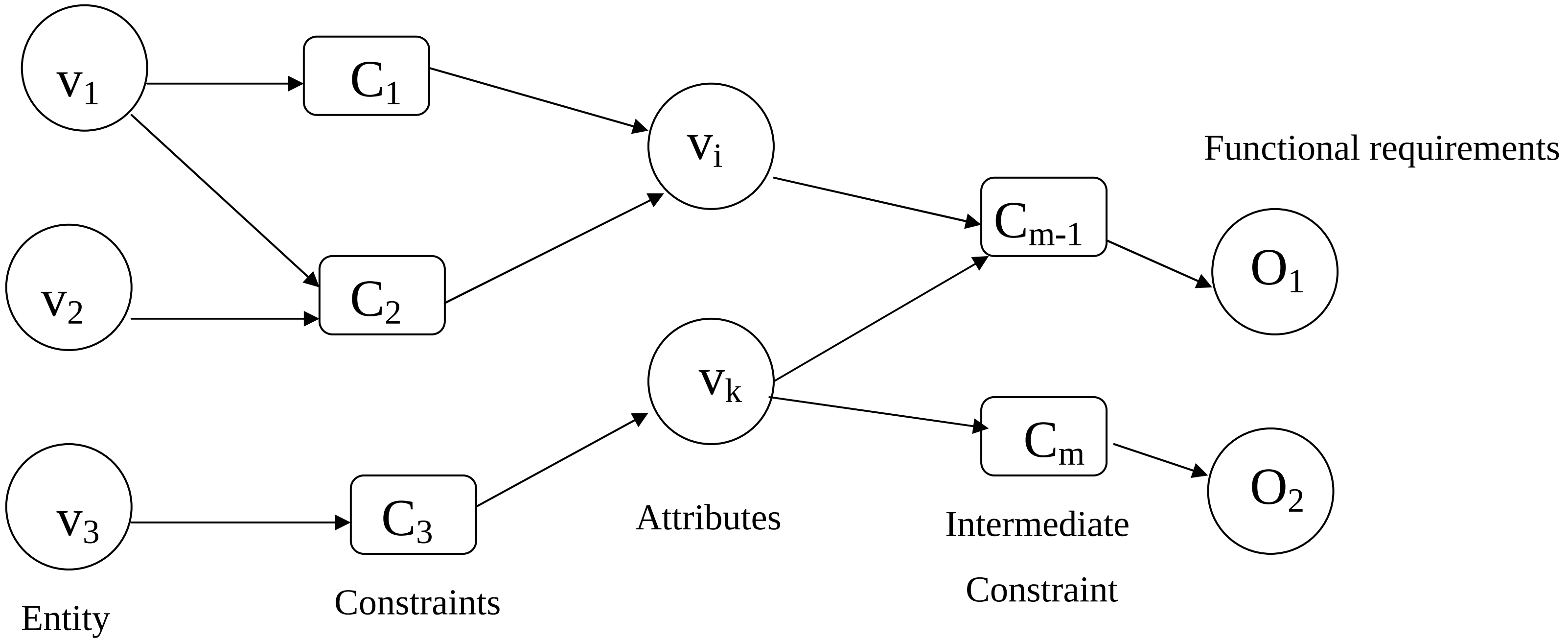
Analysis: Given the entity tolerances, the problem addressed by analysis is to make sure that the functional requirement of the design is met.

Example: Given the tolerances of a pipe’s inner and outer diameters(entity tolerances), tolerance analysis ensures that the pipe can withstand the maximum and minimum pressure due to the flow through the pipe.

Synthesis: The problem here is the inverse of analysis, From the functional requirement, we try to derive the the tolerances for the participating entities.

Example: In the pipe example, from the tolerances on the acceptable pressure inside the pipe we derive the maximum and minimum pipe diameter.

Synthesis can be a harder problem to solve than analysis,since we need to determine the tolerances for the entities and the number of entities are usually more than the number of functional requirements. In analysis we determine one functional requirement tolerance based on many entity tolerances.



Constraint Network for Analysis and Synthesis

We can use a constraint network to model this problem. The lowest level corresponds to the entities and the highest level corresponds to the functional requirement.

Forward Propagation: (Analysis)

If the Interval Propagated from the input intervals is not a subset of the output interval,The output interval should be updated. Otherwise the constraint is consistent and we do not change anything.

Algorithm for Forward Propagation for a Single Constraint (Analysis)

The forward propagation is based on the constraint function such that the intervals of the input variables are propagated to the interval of the single output variable. In forward updating, the interval for the output variable is updated (relaxed) to the union of the propagated interval and the original assigned output interval.

Forward Propagation for constraint, C({1,2, ..., n}, k, f()), FP($\mathbf{X_1}$,$\mathbf{X_2}$, ...,$\mathbf{X_n}$; $\mathbf{X_k}$, **f()**)

1. Propagate from Inputs to the Upper Limit of the Output

$x_{kup}' = f(\, x_{1\varphi}, \ldots, x_{n\varphi}\,)$

where $x_{i\varphi} = x_{iup}$ if $\mathbf{X_k}$ is monotonically increasing with respect to $\mathbf{X_i}$.
$x_{i\varphi} = x_{ilow}$ if $\mathbf{X_k}$ is monotonically decreasing with respect to $\mathbf{X_i}$.

2. Propagate from Inputs to the Lower Limit of the Output

$x_{klow}' = f(\, x_{1\kappa}, \ldots, x_{n\kappa}\,)$

where $x_{i\kappa} = x_{ilow}$ if $\mathbf{X_k}$ is monotonically increasing with respect to $\mathbf{X_i}$.
$x_{i\kappa} = x_{iup}$ if $\mathbf{X_k}$ is monotonically decreasing with respect to $\mathbf{X_i}$.

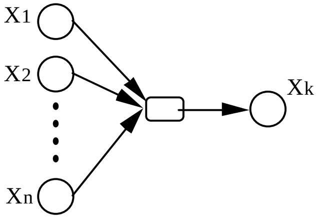
3. Relaxing the Output

If $x_{kup}' < x_{klow}$ or $x_{klow}' > x_{kup}$,
NO SOLUTION

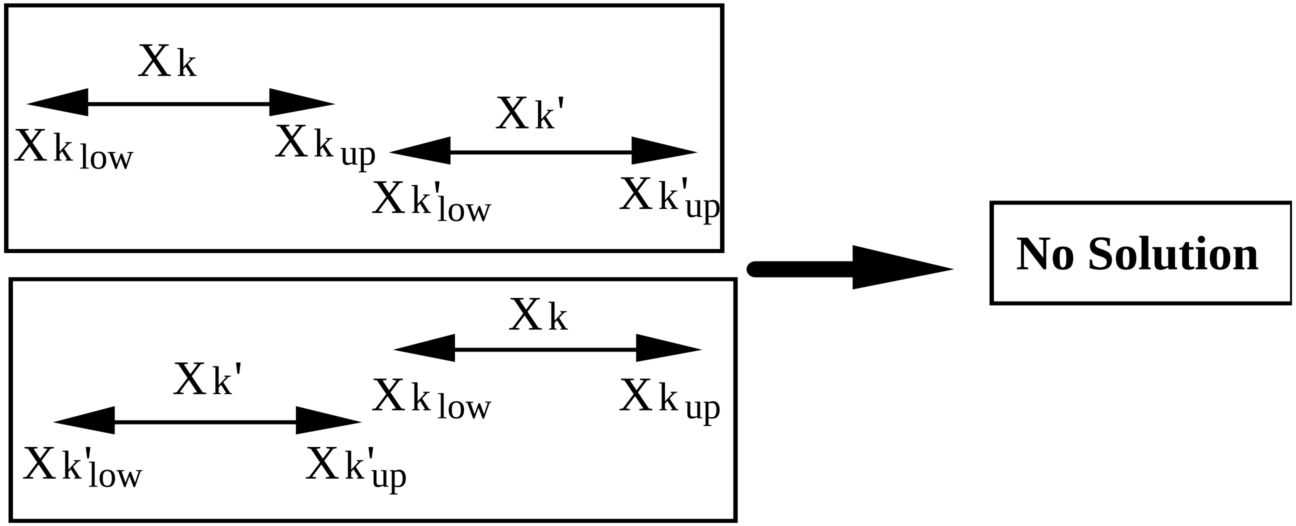
Otherwise,

$x_{kup} = x_{kup}'$ if $x_{kup}' > x_{kup}$.
$x_{klow} = x_{klow}'$ if $x_{klow}' < x_{klow}$.

Propagation and existence of solutions in analysis using the algorithm given.



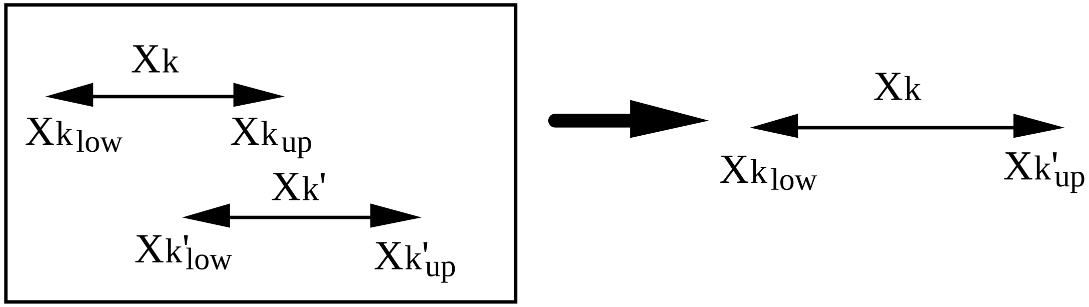
The Constraint network being considered



No Solution

Based on the algorithm, when we get intervals which do not intersect, we do not have a solution. Figure above shows two such cases.

Existence of the solution(continued)



When the two intervals we obtain are intersecting as shown above, if we are allowed to change the original output interval, the solution is the union of the two intersecting intervals.

Forward Propagation: (Analysis)

If the Interval Propagated from the input intervals is not a subset of the output interval,The output interval should be updated. Otherwise the constraint is consistent and we do not change anything.

Backward Propagation: (Synthesis)

If the constraint is not consistent, and the output interval is not to be changed, the output interval is propagated to the input intervals by tightening the input intervals.