University of Malta
BSc(Hons.) IT Year II

Problem Solving Techniques in Artificial Intelligence

Lecture 10

Search - Constraint Propagation

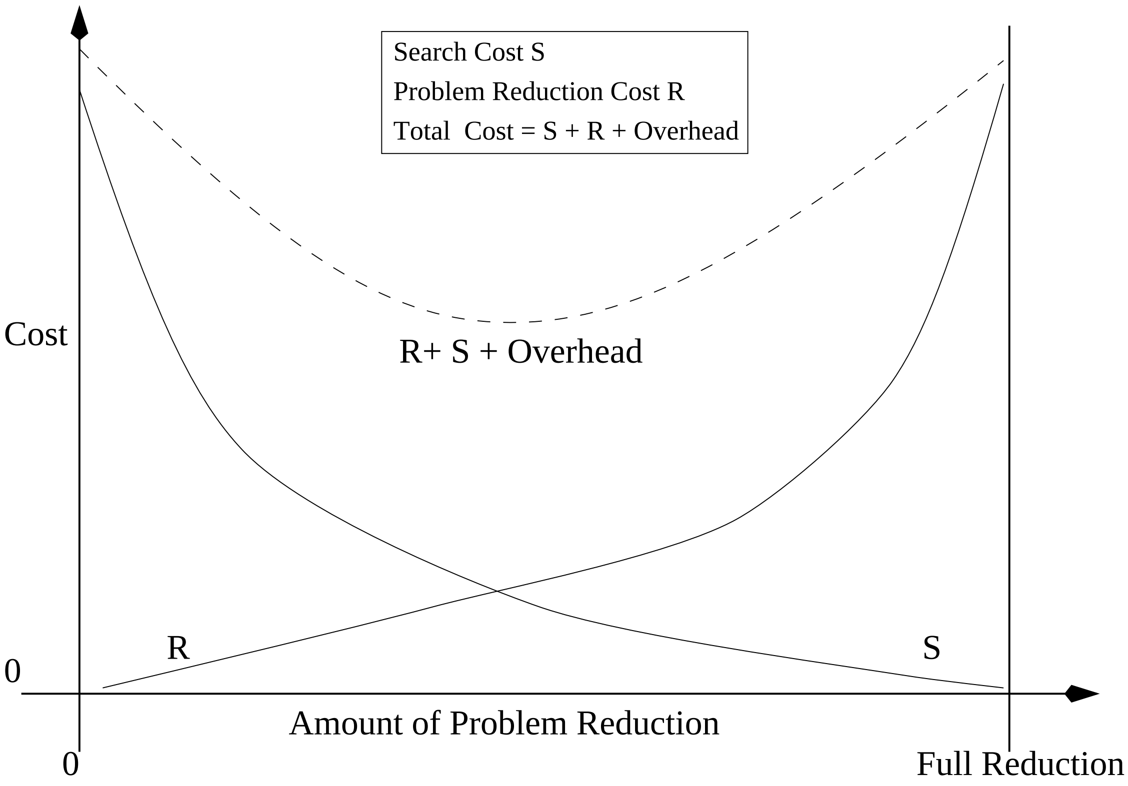
- Solution Costs
- K Consistency
- 3 Consistency, Temporal Constraints
- Allen's Algorithm

Department of Computer Science and AI

Story So Far

- **Concepts**
  - CSP, Constraint Network
  - Solutions to a CSP
  - Node Consistency
  - Path Consistency
- **Algorithms**
  - *Search*
    * Generate and Test
    * Backtracking
  - *Constraint Propagation*
    * AC-1
    * AC-3

Solution Costs



- A reduced problem is one that is K consistent for some K $0 < K \leq N$ where N is the number nodes in the constraint network.
- Reduction of the problem reduces the search cost.
- The total cost is the sum of the reduction cost and the search cost.

Arc Consistency and Solutions

Does achievement of arc consistency

- Eliminate the need for search?

  Only if each variable is left with one value in its domain

- Guarantee that CSP has a solution?

  Not in general

- Eliminate the need for backtracking?

  Not in general

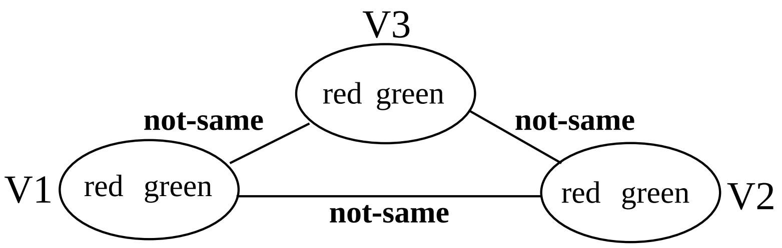
Backtracking-free search

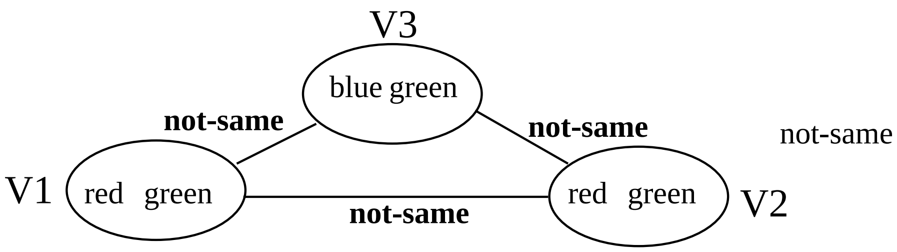
A search in a CSP is backtrack-free under a given ordering of its variables, if for every variable that is to be labelled, one can always find a value which is compatible with all the labels committed so far.

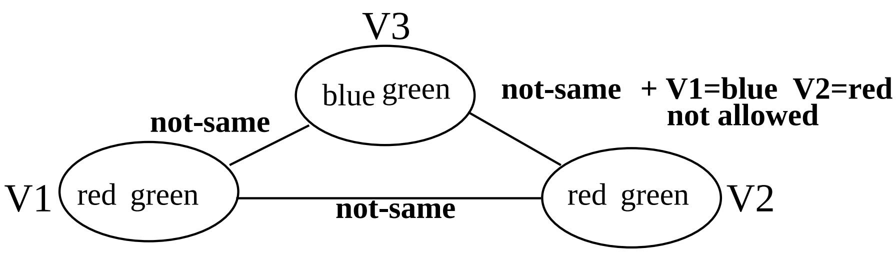
Arc Consistency and Solutions

An arc-consistent graph can have

- No solutions
- $>$ 1 solutions
- Exactly 1 solution







K Consistency

A constraint graph is **K consistent** if, for any set of K-1 variables satisfying all constraints, there exists a value for any Kth variable that satisfying all constraints among the K variables.

A constraint graph is **strongly K consistent** if it is J consistent for all J $\leq$ K.

- Node consistency is equivalent to strong 1 consistency

- Arc consistency is equivalent to strong 2 consistency

If a constraint-graph containing N nodes is strongly N consistent then solutions to the CSP can be found without backtracking.

The Case of the Clumsy Robot

1. The robot started cleaning the office after everyone and finished before anyone returned.
2. While cleaning the office, the robot accidentally disconnected the fax machine.
3. Is the fax machine is disconnected when no human is around?

The Case of the Clumsy Robot

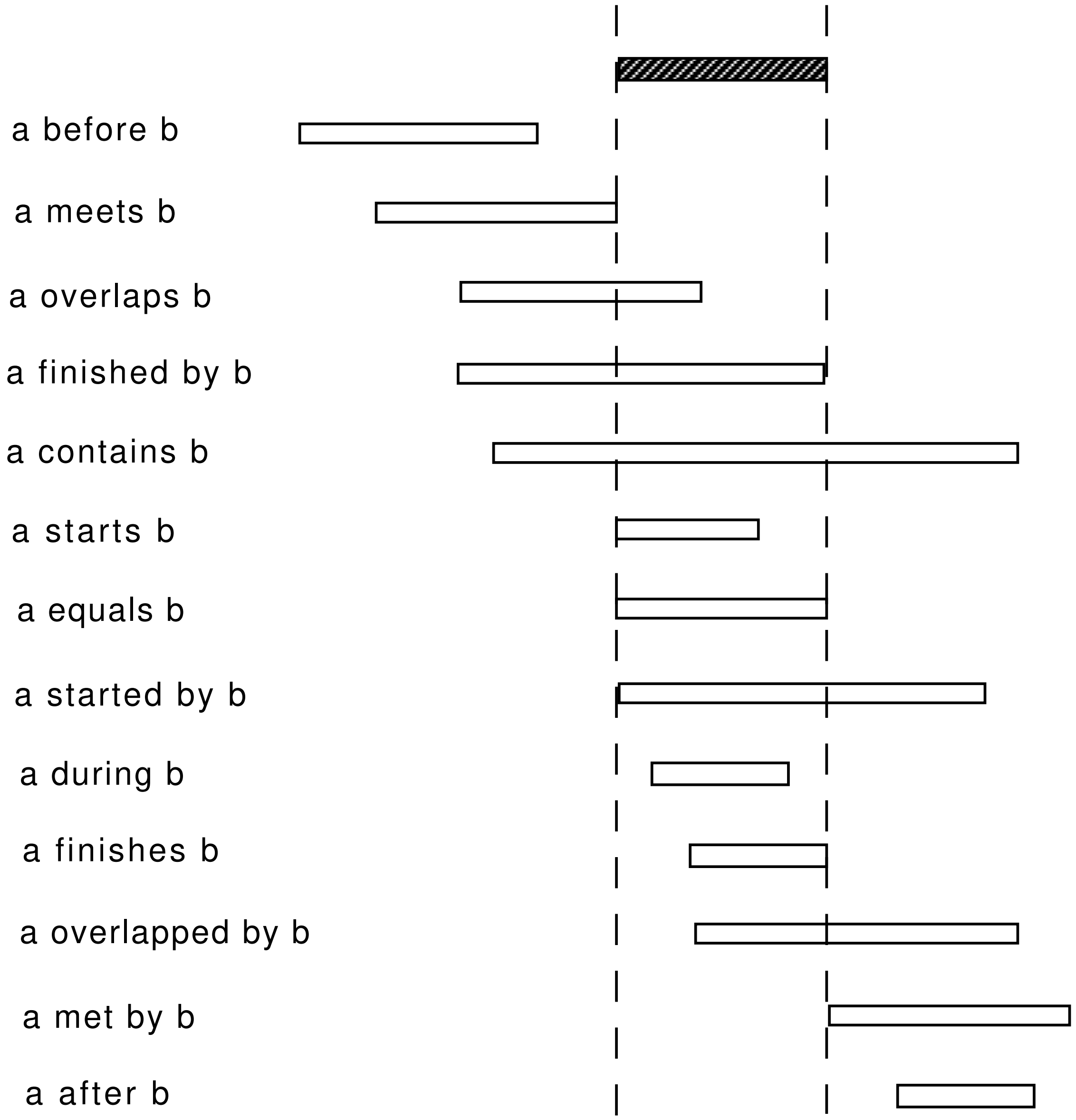
```
                  (-people;)
     people       cleaning              people
|-------------| |---------------| |-------------|
      C                B                 D

                  fax discon
                  |---------|
                       A
```

GIVEN		UNKNOWN
A during B B after C B before D	$\vdash$	A ? C A ? D

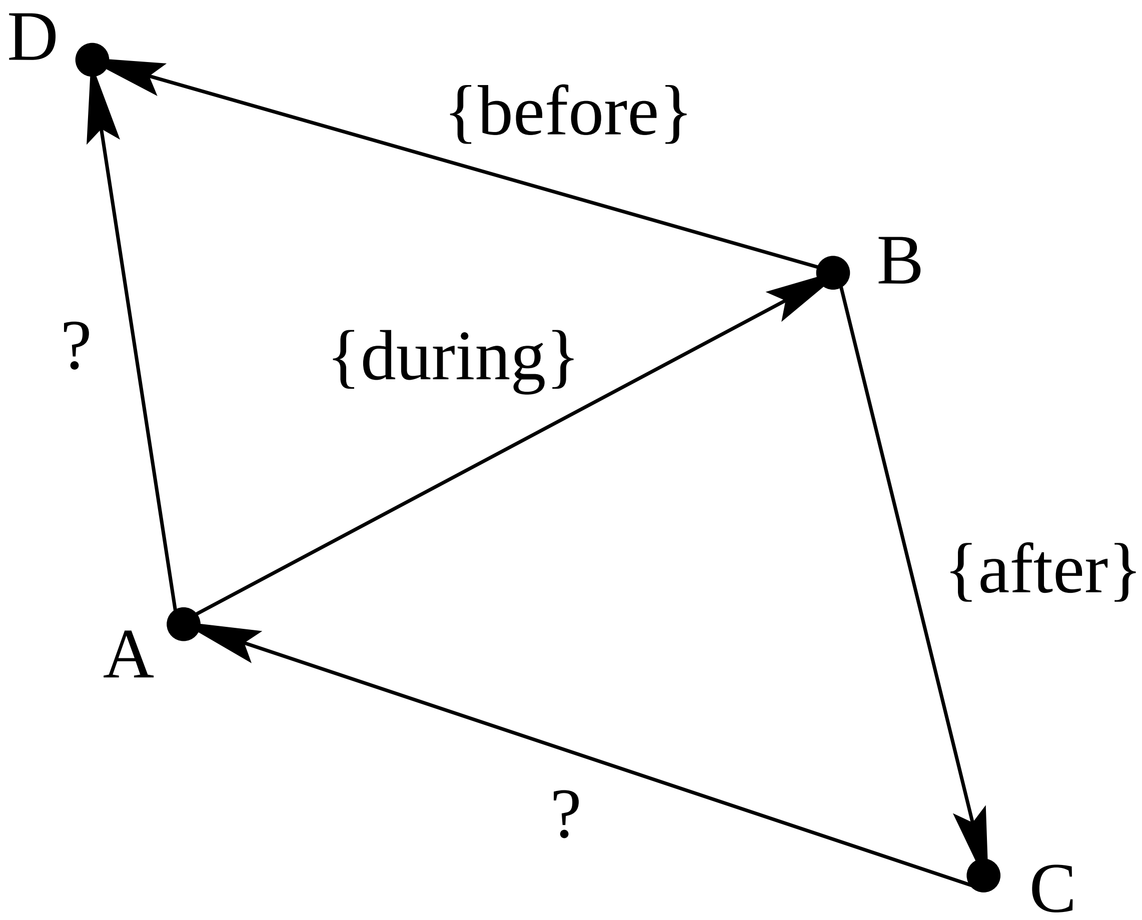
What are the *possible* relations between time intervals A, C and D?

Relations between Temporal Intervals
J.Allen



Setting Up Temporal Reasoning Problems as CSPs

GIVEN	UNKNOWN
A during B B after C B before D	A ? C A ? D



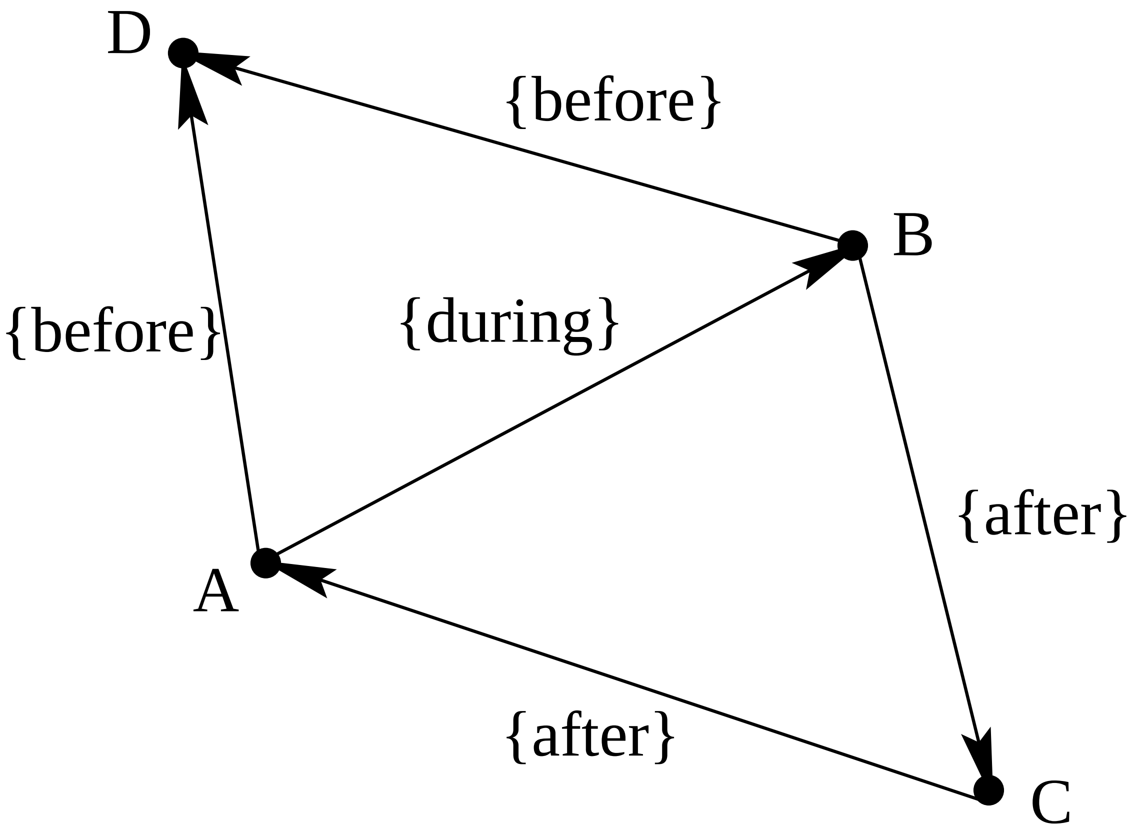
Transitive Inferences

We can determine the missing relationships by means of the following inferences.

(A during B) & (B before D) $\rightarrow$ (A before D)

(A during B) & (B after C) $\rightarrow$ (A after C)

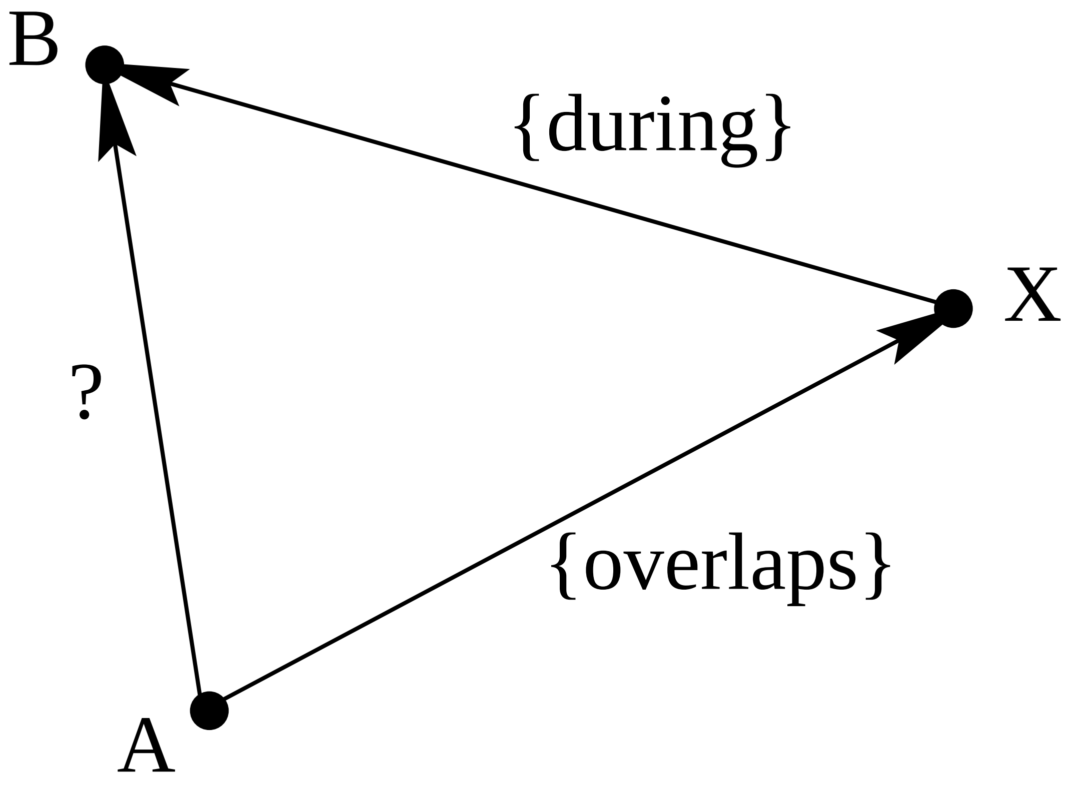
and thus fill in the complete constrant graph in a way that corresponds with our intuitions.



Incomplete Knowledge

In the last example, we either know everything about the relationship between intervals (e.g. {during}), or nothing (i.e. ?). However these are extremes, and intermediate states of uncertainty are also possible.

Suppose initially, we have no information about the interval pair (A,B), but we do know that for some other interval X it is known that (i) A overlaps X and (ii) X is during B, i.e.



Incomplete Knowledge

Given Allen's relations, the relation between A and B *must* be one of the following:

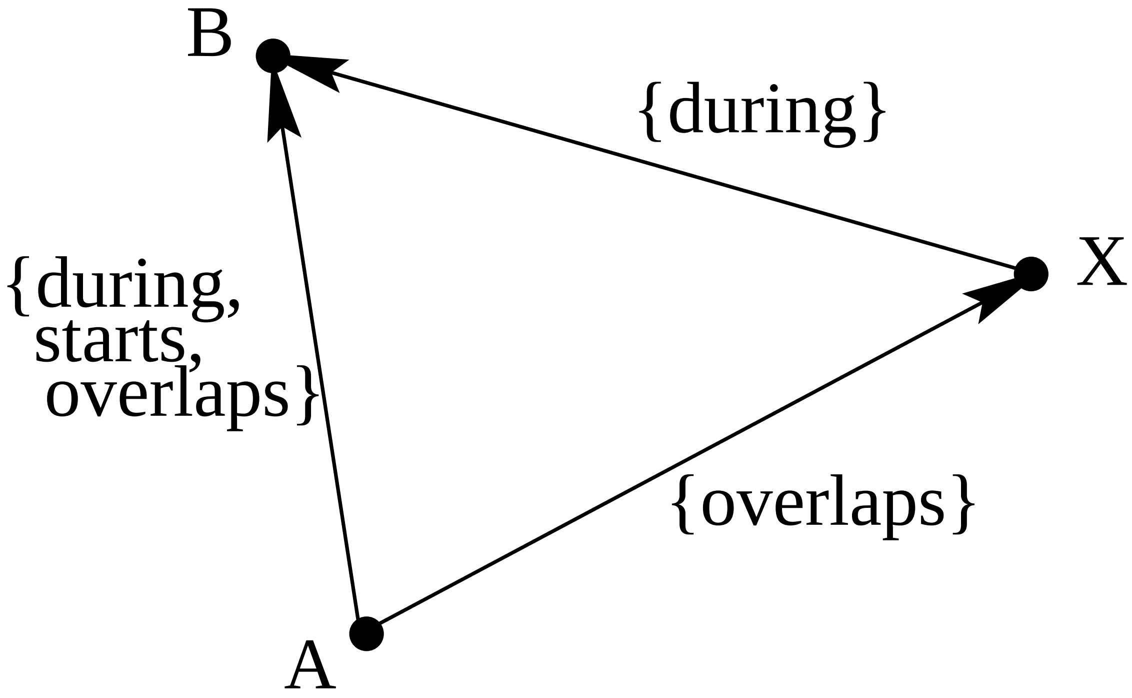
```
A |---------|
X     |-----------|           (i) A during B
B |----------------------|
```

```
A    |---------|
X        |-----------|        (ii) A starts B
B    |----------------------|
```

```
A |---------|
X         |-----------|      (iii) A overlaps B
B    |----------------------|
```

Incomplete Knowledge

We can represent this kind of uncertainty as follows:



In this representation, certainty is represented by a singleton set, total uncertainty by a “?” (which stands for a set of cardinality 13).

The corresponding CSP

Our state of information about these intervals These can be shown in the form of a constraint graph whose *variables* correspond to the directed relations between the intervals, and whose domains correspond to the current values of those variables, in this case

Variable	Domain
AX	{overlaps}
XB	{during}
AB	{during,starts,overlaps}

The Transitivity Table for Interval Relations

Inferences such as that just shown can be pre-computed and stored in the form of a "transitivity table" (a small section is shown below) whose X and Y coordinates represent R(X,B), the relation between X and B, and R(A,X), that between A and X respectively The cells contain inferred possible values for R(A,B), the relation between A and B.

$\rightarrow$ R(X,B) $\rightarrow$ / $\downarrow$ R(A,X) $\downarrow$	**b**	**a**	**d**	...
b	b	?	b,o,m,d,s	
a	?	a	a,oi,mi,d,e	
d	b	a	d	?
o	b	a,oi di,mi,si	o,d,s	
...				

For instance, if we have before(A,X) and during(X,B), we index the cell with coordinates (d,b) (b stands for before and d for during). The contents of that cell is the set {o,d,s} (overlap, during, starts) of possible relations between A and B.

The General Priniciple

In general, if A and X can stand in one of several relations, and the same is true of X and B, we obtain the possible relations between A and B by selecting possible pairs of relations between (A,X) and (X,B) and forming the union of all the inferred relations (using a table of the above kind).

Here is a specification of the algorithm , which makes use of a table T containing the transitivity relations between intervals, and for whih R1 and R2 are the possible relations on (A,X) and (X,B) respectively.

```
PROCEDURE CONSTRAINTS (R1, R2: constraints)
R := empty
for each r1 in R1 and r2 in R2 do
 R := R union T(r1,r2)
```

Allen's Algorithm

The input to the algorithm is a list of binary constraints on n intervals, given in the form of an nxn matrix N, so that N(i,j) contains the constraints between intervals i and j.

```
PROCEDURE
 PROPAGATE-INTERVAL-CONSTRAINTS(N:matrix)
Q := {(i,j) | 1 .le. i,j .le. n, i .ne. j}
repeat until Q is empty
{
 remove any (i,j) from Q
 for each interval k (k .ne. i,j)
 { N(i,k) := N(i,k) intersect
             CONSTRAINTS(N(i,j),N(j,k))
   If N(i,k) changed then add (i,k) to Q
   N(k,j) := N(k,j) intersect
             CONSTRAINTS(N(k,i),N(i,j))
   If N(k,j) changed then add (k,j) to Q

 }
}
```