

# Addition to Chapter 6

## Temporal Constraint Satisfaction Problems

CS5811 - Artificial Intelligence

Nilufer Onder  
Department of Computer Science  
Michigan Technological University

# Outline

Temporal CSP problem definition

Temporal reasoning tasks

Qualitative networks

- The interval algebra

- The point algebra

Quantative networks

# Temporal constraint satisfaction problems (CSPs)

A *temporal constraint satisfaction problem* consists of

- ▶ a finite set of *variables*, where each variable takes on time values
- ▶ a set of *constraints* that show temporal ordering or duration constraints

# Qualitative reasoning example

*John was not in the room when I touched the switch to turn on the light, but John was in the room later when the light went out.*

Represent the events as time intervals:

*Switch:* time of touching the switch

*Light:* time the light was on

*Room:* time that John was in the room

Reasoning tasks:

Is this information consistent?

If it is consistent, what are the possible scenarios?

## Quantitative reasoning example

*Let's schedule an hour meeting before or after lunch. I go to lunch before my 1:00 o'clock class. The lunch period starts at 12:00. Eating lunch takes half an hour to an hour. I have class at 11:00.*

Have variables represent time points, usually the beginning or ending of an event. The constraints show the lower bound and upper bound of the time interval between two time points. For example, the constraint between  $\text{lunch}_b$  and  $\text{lunch}_e$  is  $[30, 60]$  minutes.

Reasoning tasks:

Is it possible that a proposition P holds at time  $t_1$ ?

What are the possible times at which a proposition P holds?

What are the possible temporal relationships between two propositions P and Q?

# Temporal representation and reasoning framework

- ▶ Temporal knowledge base
  - ▶ Temporal objects: points or intervals
  - ▶ Temporal constraints: qualitative or quantitative
- ▶ Temporal inference
  - ▶ Consistency check routines
  - ▶ Inference routines
  - ▶ Query answering mechanisms

# Interval algebra example

*John was not in the room when I touched the switch to turn on the light, but John was in the room later when the light went out.*

Represent the events as time intervals:

*Switch*: time of touching the switch

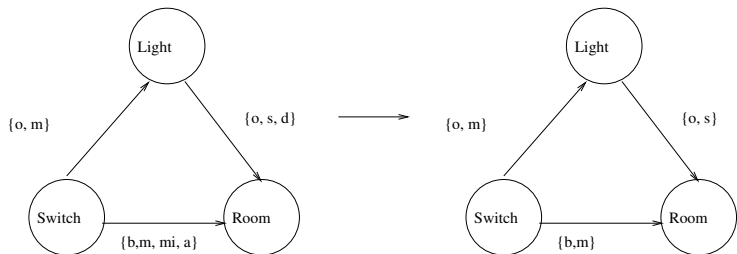
*Light*: time the light was on

*Room*: time that John was in the room

Represent the constraints:

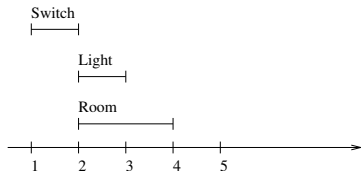
- ▶ *Switch* overlaps or meets *Light*
- ▶ *Light* overlaps, starts, or is during *Room*
- ▶ *Switch* is before, meets, meets-inverse, or starts *Room*

# IA constraint graph (network), minimal network, solution



Original constraints: (in solution)

- Switch overlaps or meets Light (m)
- Light overlaps, starts, or is during room (s)
- Switch is before, meets, meets-inverse, or starts Room (m)










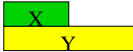

# Reasoning tasks for IA networks

- ▶ decide consistency
- ▶ find one or more solutions
- ▶ compute the minimal network

All are generally intractable, so

- ▶ improve exponential search algorithms such as backtracking,  
or
- ▶ resort to local inference procedures

# Interval algebra constraints

Relation	Symbol	Inverse	Example
X before Y	b	bi	
X equal Y	=	=	
X meets Y	m	mi	
X overlaps Y	o	oi	
X during Y	d	di	
X starts Y	s	si	
X finishes Y	f	fi	

# Representation

$I\{r_1, \dots, r_k\}$   $J$  represents  $(I r_1 J) \vee \dots \vee (I r_k J)$

For example  $I\{s, si, d, di, f, fi, o, oi, =\}$   $J$  expresses the fact that intervals  $I$  and  $J$  intersect (it excludes  $b, bi, m, mi$ ).

John was not in the room when I touched the switch to turn on the light, but John was in the room later when the light went out.

1. *Switch*  $\{o, m\}$  *Light*
2. *Switch*  $\{b, m, mi, a\}$  *Room*
3. *Light*  $\{o, s, d\}$  *Room*

# IA Constraint graph terms

- ▶ In a *constraint graph*, the nodes represent the variables and an edge represents a direct constraint (coming from the IA relation set)
- ▶ A *universal constraint* permits all relationships between two variables and is represented by the lack of an edge between the variables.
- ▶ A constraint  $C'$  can be *tighter* than constraint  $C''$ , denoted by  $C' \subseteq C''$ , yielding a partial order between IA networks. A network  $N''$  is tighter than network  $N'$  if the partial order  $\subseteq$  is satisfied for all the corresponding constraints.
- ▶ The *minimal network* of  $M$  is the unique equivalent network of  $M$  which is minimal with respect to  $\subseteq$ .

# Path Consistency in CSPs

- ▶ Given a constraint network  $R = (X, D, C)$ , a *two-variable set*  $\{x_i, x_j\}$  is *path-consistent* relative to variable  $x_k$  iff for every consistent assignment  $(\langle x_i, a_i \rangle, \langle x_j, a_j \rangle)$  there is a value  $a_k \in D_k$  such that the assignment  $(\langle x_i, a_i \rangle, \langle x_k, a_k \rangle)$  is consistent and  $(\langle x_k, a_k \rangle, \langle x_j, a_j \rangle)$  is consistent.
- ▶ Alternatively, a *binary constraint*  $R_{ij}$  is *path-consistent* relative to  $x_k$  iff for every pair  $(a_i, a_j) \in R_{ij}$  where  $a_i$  and  $a_j$  are from their respective domains, there is a value  $a_k \in D_k$  such that  $(a_i, a_k) \in R_{ik}$  and  $(a_k, a_j) \in R_{kj}$ .

## Path-consistency in CSPs (cont'd)

- ▶ A subnetwork over three variables  $\{x_i, x_j, x_k\}$  is *path-consistent* iff for any permutation of  $(i, j, k)$ ,  $R_{ij}$  is path-consistent relative to  $x_k$ .
- ▶ A network is *path-consistent* iff for every  $R_{ij}$  (including universal binary relations) and for every  $k \neq i, j$ ,  $R_{ij}$  is path-consistent relative to  $x_k$ .

# Path-consistency in IA

- ▶ An IA network is *path-consistent* if for every three variables  $x_i, x_j, x_k$ ,  $C_{ij} \subseteq C_{ik} \otimes C_{kj}$ .
- ▶ The *intersection* of two IA relations  $R'$  and  $R''$ , denoted by  $R' \oplus R''$ , is the set-theoretic intersection  $R' \cap R''$ .
- ▶ The *composition* of two IA relations,  $R' \otimes R''$ ,  $R'$  between intervals  $I$  and  $K$  and  $R''$  between intervals  $K$  and  $J$ , is a new relation between intervals  $I$  and  $J$ , induced by  $R'$  and  $R''$  as follows.

# Composition ( $\otimes$ )

- ▶ The composition of two basic relations  $r'$  and  $r''$  is defined by a *transitivity table* (see a portion of it on the next slide).
- ▶ The composition of two *composite* relations  $R'$  and  $R''$ , denoted by  $R' \otimes R''$ , is the composition of the constituent basic relations:

$$R' \otimes R'' = \{r' \otimes r'' \mid r' \in R', r'' \in R''\}$$

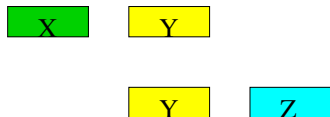


# Composition of basic relations

	b	s	d	o	m
b	b	b	b o m d s	b	b
s	b	s	d	b o m	b
d	b	d	d	b o m d s	b
o	b	o	o d s	b o m	b
m	b	m	o d s	b	b

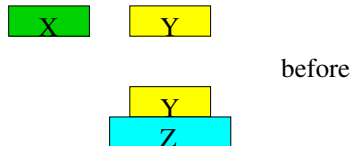
# Composition examples

X before Y, Y before Z

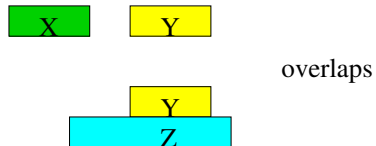


X before Z

X before Y, Y during Z



X {b, o, m, d, s} Z



# Qualitative Path Consistency (QPC) Algorithm

**function** QPC-1 ( $T$ )

**returns** a path consistent IA network

**input:**  $T$ , an IA network with  $n$  variables

**repeat**

$S \leftarrow T$

**for**  $k \leftarrow 1$  to  $n$  **do**

**for**  $i, j \leftarrow 1$  to  $n$  **do**

$C_{ij} \leftarrow C_{ij} \oplus C_{ik} \otimes C_{kj}$

**until**  $S = T$

**return**  $T$

# Example

$$\text{Apply } C_{SR} \leftarrow C_{SR} \oplus (C_{SL} \otimes C_{LR})$$

$$C_{SR} \leftarrow \{b, m, i, a\} \oplus (\{o, m\} \otimes \{o, s, d\})$$

$$C_{SR} \leftarrow \{b, m, i, a\} \oplus \{b, o, m, d, s\}$$

$$C_{SR} \leftarrow \{b, m\}$$

$$o \otimes o = b, o, m$$

$$o \otimes s = o$$

$$o \otimes d = o, d, s$$

$$m \otimes o = b$$

$$m \otimes s = m$$

$$m \otimes d = o, d, s$$

# Minimizing networks using path-consistency

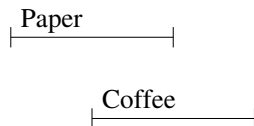
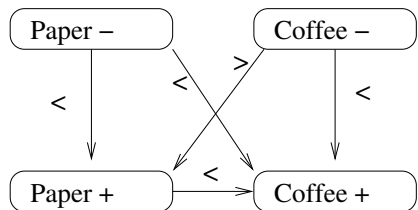
- ▶ In some cases, path-consistency algorithms are *exact*—they are guaranteed to generate the minimal network and therefore decide consistency.
- ▶ In general, IA networks are NP-complete, backtracking search is needed to generate a solution.
- ▶ Even when the minimal network is available, it is not guaranteed to be globally consistent to allow backtrack-free search.
- ▶ Path-consistency can be used for forward checking.

# The point algebra (PA)

- ▶ It is a model alternative to IA: the nodes represent time points rather than intervals
- ▶ It is less expressive: there are three basic types of constraints between points  $P$  and  $Q$ :
  - ▶  $P < Q$
  - ▶  $P = Q$
  - ▶  $P > Q$
- ▶ Reasoning tasks over PAs are polynomial

# Example

Fred put the paper down and drank the last of his coffee.



# Interval algebra vs. point algebra constraints

- ▶  $I\{s, d, f, =\}J$  where  $I = [x, y]$  and  $J = [z, t]$  can be represented with

$$x < y, z < t, x < t, x \geq z, y \leq t, y > z$$

- ▶ However,  $I\{b, a\}J$  where  $I = [x, y]$  and  $J = [z, t]$  cannot be represented with a PA network



# Composition in the PA

	$<$	$=$	$>$
$<$	$<$	$<$	$?$
$=$	$<$	$=$	$>$
$>$	$?$	$=$	$>$

“?” expresses the universal relation.

# Path consistency

- ▶ It is defined using composition and the transitivity table
- ▶ Path consistency decides the consistency of a PA network in  $O(n^3)$  steps.
- ▶ Consistency and solution generation of PA networks can also be accomplished in  $O(n^2)$ .
- ▶ The minimal network of a PA consistent network can be obtained using 4-consistency in  $O(n^4)$  steps.
- ▶ The minimal network of CPA networks can be obtained by path-consistency in  $O(n^3)$ .  
*Convex PA (CPA)* networks have only  $\{<, \leq, =, \geq, >\}$  and not  $\neq$ .

# Quantitative Temporal Networks

- ▶ Ability to express metric information on duration and timing of events
- ▶ John travels to work either by car (30–40 minutes) or by bus (at least 60 minutes). Fred travels to work either by car (20–30 minutes) or in a carpool (40–50 minutes). Today John left home between 7:10 and 7:20A.M., and Fred arrived at work between 8:00 and 8:10A.M. We also know that John arrived at work 10-20 minutes after Fred left home.
- ▶ Is the information in the story consistent?
- ▶ Is it possible that John took the bus and Fred used the carpool?
- ▶ What are the possible times at which Fred left home?

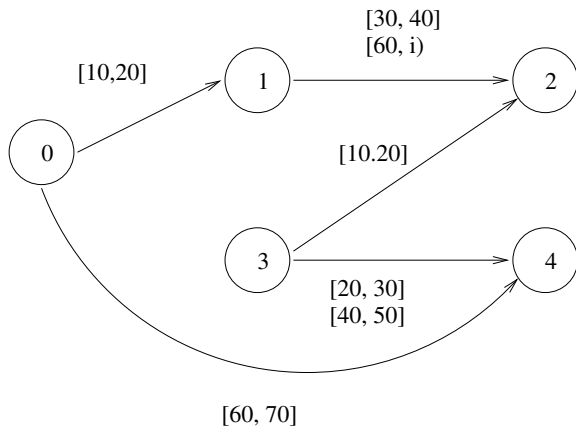
# Representation

- ▶ Proposition  $P_1$ : John was traveling to work ( $[x_1, x_2]$ )
- ▶ Proposition  $P_2$ : Fred was traveling to work ( $[x_3, x_4]$ )
- ▶ John travels to work either by car (30–40 minutes) or by bus (at least 60 minutes).  
 $30 \leq x_2 - x_1 \leq 40$  or  $x_2 - x_1 \geq 60$
- ▶ Fred travels to work either by car (20–30 minutes) or in a carpool (40–50 minutes).  
 $20 \leq x_4 - x_3 \leq 30$  or  $40 \leq x_4 - x_3 \leq 50$

## Representation (cont'd)

- ▶ Proposition  $P_1$ : John was traveling to work ( $[x_1, x_2]$ )
- ▶ Proposition  $P_2$ : Fred was traveling to work ( $[x_3, x_4]$ )
- ▶ Today John left home between 7:10 and 7:20AM  
(Assign  $x_0 = 7:00\text{AM}$ )  
 $10 \leq x_1 - x_0 \leq 20$
- ▶ Fred arrived at work between 8:00 and 8:10AM  
 $60 \leq x_4 - x_0 \leq 70$
- ▶ John arrived at work 10-20 minutes after Fred left home.  
 $10 \leq x_4 - x_0 \leq 20$

# The constraint graph



# Temporal Constraint Satisfaction Problem (TCSP)

A *temporal constraint satisfaction problem (TCSP)* involves a set of variables  $\{x_1, \dots, x_n\}$  having continuous domains; each variable represents a time point. Each constraint is represented by a set of intervals  $\{I_1, \dots, I_k\} = \{[a_1, b_1], \dots, [a_k, b_k]\}$ .

A unary constraint  $T_i$  restricts the domain of a variable  $x_i$  to the given set of intervals; that is, it represents the disjunction

$$(a_1 \leq x_i \leq b_1) \vee \dots \vee (a_k \leq x_i \leq b_k)$$

A binary constraint  $T_{ij}$  constrains the permissible values for the distance  $x_j - x_i$ ; it represents the disjunction

$$(a_1 \leq x_j - x_i \leq b_1) \vee \dots \vee (a_k \leq x_j - x_i \leq b_k)$$

## TCSP (cont'd)

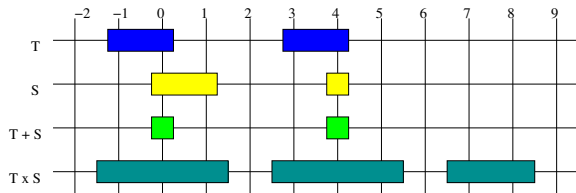
- ▶ Assume that constraints are given in a *canonical form* in which all intervals are pair-wise disjoint.
- ▶ A special time point,  $x_0$ , represents the “beginning of the world.” Each unary constraint can be represented as a binary constraint relative to  $x_0$ .
- ▶ A tuple  $x = \{a_1, \dots, a_n\}$  is called a *solution* if the assignment  $\{x_1 = a_1, \dots, x_n = a_n\}$  does not violate any constraint.



# Minimal and binary decomposable networks

- ▶ Given a TCSP, a value  $v$  is a *feasible value* for variable  $x_i$  if there exists a solution in which  $x_i = v$ .
- ▶ The set of all feasible values of a variable is called the *minimal domain*.
- ▶ A minimal constraint  $T_{ij}$  between  $x_i$  and  $x_j$  is the set of all feasible values for  $x_i - x_j$ .
- ▶ A network is minimal iff its domains and constraints are minimal.
- ▶ A network is *binary decomposable* if every consistent assignment of values to a set of variables  $S$  can be extended to a solution.

# Binary operators on constraints



$$T = \{[-1.25, 0.25]\}, [2.75, 4.25]\}$$

$$S = \{[-0.25, 1.25]\}, [3.75, 4.25]\}$$

$$T \oplus S = \{[-0.25, 0.25]\}, [3.75, 4.25]\}$$

$$T \otimes S = \{[-1.50, 1.50], [2.50, 5.50], [6.50, 8.50]\}$$

## Binary operators on constraints (cont'd)

Let  $T = \{I_1, \dots, I_l\}$  and  $S = \{J_1, \dots, J_m\}$  be two constraints. Each is a set of intervals of a temporal variable or a temporal binary constraint.

- ▶ The union of  $T$  and  $S$ , denoted by  $T \cup S$ , only admits values that are allowed by either  $T$  or  $S$ , that is,  $T \cup S = \{I_1, \dots, I_l, J_1, \dots, J_m\}$ .
- ▶ The intersection of  $T$  and  $S$ , denoted by  $T \oplus S$ , admits only values that are allowed by both  $T$  and  $S$ , that is,  $T \oplus S = \{K_1, \dots, K_n\}$  where  $K_k = I_i \cap J_j$  for some  $i$  and  $j$ . Note that  $n \leq l + m$ .

## Binary operators on constraints (cont'd)

- ▶ The composition of  $T$  and  $S$ , denoted by  $T \otimes S$ , admits only values  $r$  for which there exist  $t \in T$  and  $s \in S$ , such that  $t + s = r$ , that is  $T \otimes S = \{K_1, \dots, K_n\}$ , where  $K_k = [a + c, b + d]$  for some  $I_i = [a, b]$ , and  $J_j = [c, d]$ . Note that  $n \leq l \times m$ .

# Simple temporal problems (STPs)

- ▶ It is a subclass of TCSPs where all constraints specify a single interval (no disjunctions).
- ▶ Each edge  $i \rightarrow j$  is labeled by a single interval  $[a_{ij}, b_{ij}]$  that represents the constraint

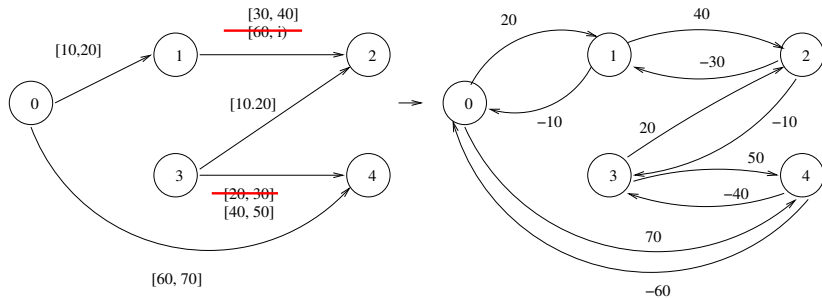
$$a_{ij} \leq x_j - x_i \leq b_{ij}$$

or

$$x_j - x_i \leq b_{ij} \text{ and } x_i - x_j \leq -a_{ij}$$

- ▶ Can be represented and solved as a system of linear inequalities but a better graph algorithm exists: first convert the graph into a *distance graph*

# Distance graph example



# Distance graph

- ▶ An STP can be associated with a directed-edge weighted graph  $G_d = (v, E_d)$ , called the *distance graph*. It has the same node set as  $G$ , and each edge  $i \rightarrow j \in E_d$  is labeled by a weight  $a_{ij}$  representing the linear inequality  $x_j - x_i \leq a_{ij}$ .
- ▶ Each path from  $i$  to  $j$  in  $G_d$ ,  $i_0 = i, i_1, \dots, i_k = j$ , induces the following constraint on the distance  $x_j - x_i$ :

$$x_j - x_i \leq \sum_{j=1}^k a_{i_{j-1}, i_j}$$

## Distance graph (cont'd)

- ▶ If there is more than one path from  $i$  to  $j$ , then it can easily be verified that the intersection of all the induced path constraints yields

$$x_j - x_i \leq d_{ij}$$

where  $d_{ij}$  is the length of the shortest path from  $i$  to  $j$ .

- ▶ Theorem: An STP  $T$  is consistent iff its distance graph  $G_d$  has no negative cycles
- ▶ For any pair of connected nodes  $i$  and  $j$ , the shortest paths satisfy  $d_{oj} \leq d_{oi} + a_{ij}$ ; thus,

$$d_{oj} - d_{oi} \leq a_{ij}$$



## Distance graph (cont'd)

- ▶ Let  $G_d$  be the distance graph of a consistent STP. Two consistent scenarios are given by

$$S_1 = (d_{01}, \dots, d_{0n}) \text{ and } S_2 = (-d_{10}, \dots, -d_{n0})$$

which assign to each variable its latest and earliest possible times, respectively.

- ▶ A given STP can be effectively specified by a complete directed graph, called *d-graph*, where each edge is labeled by the shortest-path length  $d_{ij}$  in  $G_d$ .
- ▶ Decomposability theorem: Any consistent STP is backtrack-free (decomposable) relative to the constraints in its *d-graph*.

## Lengths of shortest paths ( $d_{ij}$ )

	0	1	2	3	4
0	0	20	50	30	70
1	-10	0	40	20	60
2	-40	-30	0	-10	30
3	-20	-10	20	0	50
4	-60	-50	-20	-40	0

# The minimal network

	0	1	2	3	4
0	[0]	[10,20]	[40,50]	[20,30]	[60,70]
1	[-20,-10]	[0]	[30,40]	[10,20]	[50,60]
2	[-50,-40]	[-40,-30]	[0]	[-20,-10]	[20,30]
3	[-30,-20]	[-20,-10]	[10,20]	[0]	[40,50]
4	[-70,-60]	[-60,-50]	[-20,-30]	[-50,-40]	[0]

# Floyd-Warshall's Algorithm (apsp)

**function** ALL-PAIRS-SHORTEST-PATHS ( $G$ )

**returns** a  $d$ -graph

**input:** Distance graph  $G = (V, E)$  with weights  $a_{ij}$  for  $(i, j) \in E$ .

**for**  $i \leftarrow 1$  to  $n$  **do**

$d_{ij} \leftarrow 0$

**for**  $i, j \leftarrow 1$  to  $n$  **do**

$d_{ij} \leftarrow a_{ij}$

**for**  $k \leftarrow 1$  to  $n$  **do**

**for**  $i, j \leftarrow 1$  to  $n$  **do**

$d_{ij} \leftarrow \min \{d_{ij}, d_{ik} + d_{kj}\}$

# Summary

- ▶ Floyd-Warshall's algorithm runs in  $O(n^3)$  and detects negative cycles simply by examining the sign of the diagonal elements  $d_{ii}$ .
- ▶ Once the  $d$ -graph is available, assembling a solution takes only  $O(n^2)$  time, because each successive assignment only needs to be checked against previous assignments and is guaranteed to remain unaltered.
- ▶ Thus, finding a solution takes  $O(n^3)$  time.
- ▶ Note that in TCSP, path consistency can be checked in polynomial time but does not guarantee minimality.

## Summary (cont'd)

- ▶ Any constraint network in PA is a special case of a TCSP lacking metric information.
- ▶ A PA can be translated into a TCSP in a straightforward manner.
  - ▶  $x_j < x_i$  translates to  $T_{ij} = \{(-\infty, 0)\}$
  - ▶  $x_j \leq x_i$  translates to  $T_{ij} = \{(-\infty, 0]\}$
  - ▶  $x_j = x_i$  translates to  $T_{ij} = \{[0]\}$
  - ▶  $x_j \neq x_i$  translates to  $T_{ij} = \{(-\infty, 0), (0, \infty)\}$
- ▶ IA networks cannot always be translated into binary TCSPs because such a translation may require nonbinary constraints:  
 $X \{b, bi\} Y \equiv X_e < Y_s \vee Y_e < X_s$

## Example: Autominder

- ▶ To assist people with memory impairment.
- ▶ Model their daily activities, including temporal constraints on their performance
- ▶ Monitor the execution of those activities
- ▶ Decide whether and when to issue reminders

## Example: Autominder (cont'd)

ACTION	TARGET TIME
Start laundry	Before 10 a.m.
Put clothes in dryer	Within 20 minutes of washer ending
Fold clothes	Within 20 minutes of dryer ending
Prepare lunch	Between 11:45 and 12:15
Eat lunch	At end of prepare lunch
Check pulse	Between 11:00 and 1:00, and between 3:00 and 5:00
<i>depending on pulse</i> take medication	at end of check pulse



# Other examples

- ▶ US NINDS (National Institute of Neurological Disorders and Stroke) guidelines for treatment of potential stroke (thrombolytic) patient
  - ▶ hospital door to doctor: 10 minutes
  - ▶ door to neurological expert: 15 minutes
  - ▶ door to CT scan completion: 25 minutes
  - ▶ ...
- ▶ Space facility crew activity planning
- ▶ Control of spacecraft on another planet

# Sources for the slides

- ▶ AIMA textbook (3<sup>rd</sup> edition)
- ▶ Bartak, Roman; Morris, Robert A.; Venable, K. Brent. ICAPS-14 Constraint-Based Temporal Reasoning. 2014.
- ▶ Dechter, Rina. Constraint Processing. Chapter 12 (Temporal Constraint Processing). Morgan Kaufmann Publishers (Elsevier Science), 2003.
- ▶ Dechter, Rina; Meiri, Itay; and Pearl, Judea. Temporal Constraint Networks. Artificial Intelligence, 49 (1991), pp. 61-95.