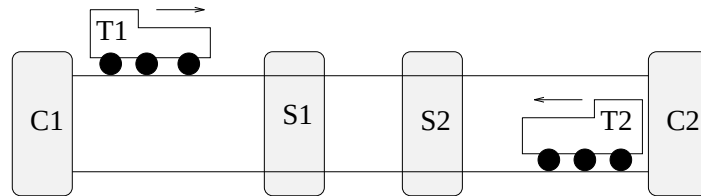


Ragionamento Temporale basato su Vincoli

Prof. Alfonso Gerevini

Ragionamento Temporale: un esempio



- (1) *Durante il suo viaggio da C1 a C2, T1 si è fermato prima a S1 e poi a S2:*

`at(T1,S1) {during} travel(T1,C1,C2)`

`at(T1,S2) {during} travel(T1,C1,C2)`

- (2) *T2 ha viaggiato in direzione opposta a T1:*

`at(T2,S1) {during} travel(T2,C2,C1)`

`at(T2,S2) {during} travel(T2,C2,C1)`

- (3) *T2 si è fermato prima a S2 e poi a S1:*

`at(T1,S1) {before} at(T1,S2)`

`at(T2,S2) {before} at(T2,S1)`

(4) *Quando T2 è arrivato a S1, T1 era a S1:*

$\text{at}(T1, S1) \quad \{\text{overlaps, contains, finished-by}\} \quad \text{at}(T2, S1)$

(5) *T1 e T2 hanno lasciato S1 in tempi diversi:*

$\text{at}(T1, S1) \quad \mathbf{B} - \{\text{equal, finishes, finished-by}\} \quad \text{at}(T2, S1)$

(6) *T1 è arrivato a C2 prima che T2 fosse arrivato a C1:*

$\text{Travel}(T1, C1, C2) \quad \{\text{b, m, o, s, d}\} \quad \text{Travel}(T2, C2, C1)$

(7) *Alla stazione S2 non può sostare più di un treno contemporaneamente:*

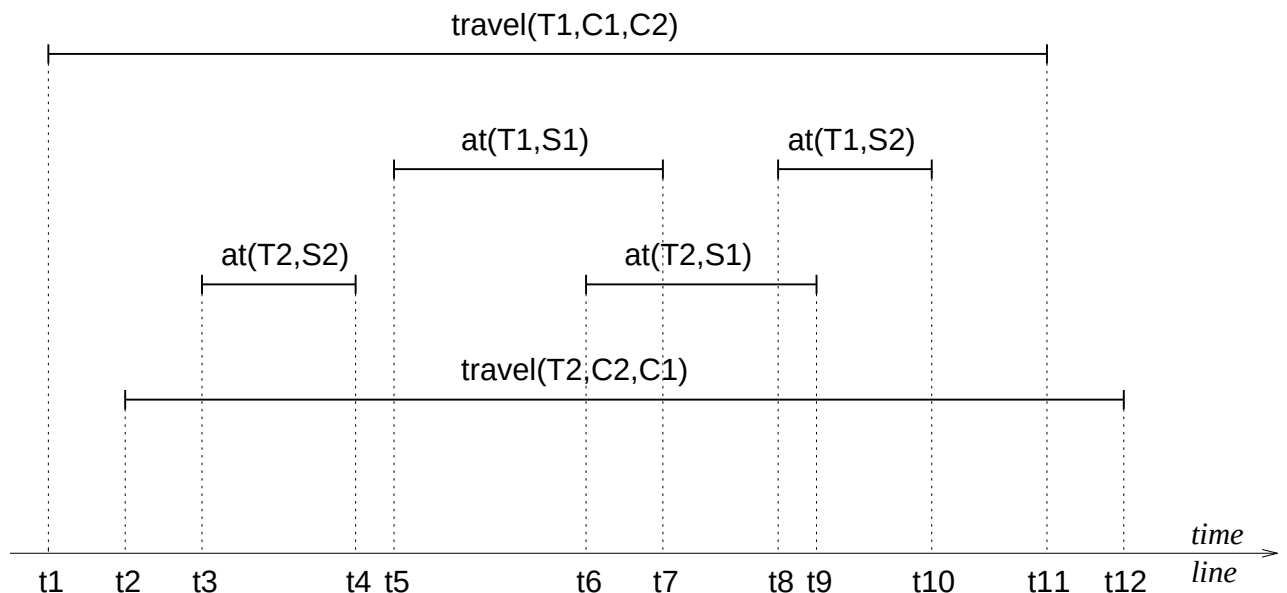
$\text{at}(T2, S2) \quad \{\text{before, after}\} \quad \text{at}(T1, S2)$

Principali Problemi di Ragionamento

- Determinare la **consistenza** (soddisfacibilità) dell' insieme di vincoli;
- Trovare uno **scenario consistente** (un ordinamento di tutte le variabili temporali che sia consistente con i vincoli forniti);
- Trovare una **soluzione** (una interpretazione di tutte le variabili temporali coinvolte che sia consistente con i vincoli forniti)
- Dedurre nuovi vincoli da quelli già conosciuti e, in particolare: colcolare la **relazione implicata più forte**
 - tra tutte le coppie di variabili (rete di vincoli *minima*);
 - tra una particolare variabile e tutte le altre;
 - tra due specifiche variabili.

Esempio

- I vincoli sono consistenti;
- Scenario e soluzione:



- Esempi di vincoli impliciti che sono deducibili:

$\text{travel}(T1, C1, C2) \text{ } \mathbf{B} - \{b, a, m, mi\} \text{ } \text{travel}(T2, C2, C1)$
 $\text{at}(T2, S2) \text{ } \{\text{before}\} \text{ } \text{at}(T1, S2)$

Algebra dei Punti (PA)

Insieme di relazioni derivato da tre relazioni base (“<”, “>” and “=”) attraverso le operazioni di **inverso** (denotato con $\smile$), **unione** ($\cup$), **intersezione** ($\cap$) e **composizione** ($\circ$):

$$\forall x, y : \quad xR\smile y \leftrightarrow yRx$$

$$\forall x, y : \quad x(R \cup S)y \leftrightarrow xRy \vee xSy$$

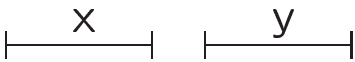
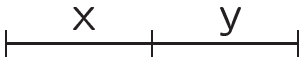
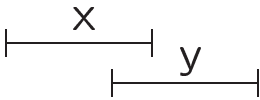
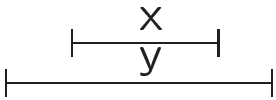
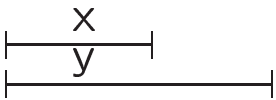
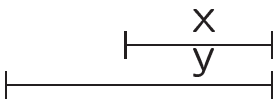
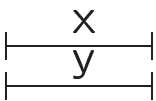
$$\forall x, y : \quad x(R \cap S)y \leftrightarrow xRy \wedge xSy$$

$$\forall x, y : \quad x(R \circ S)y \leftrightarrow \exists z : (xRz) \wedge (zSy).$$

$\circ$	<	>	=	$\leq$	$\geq$	$\neq$	?
<	<	?	<	<	?	?	?
>	?	>	>	?	>	?	?
=	<	>	=	$\leq$	$\geq$	$\neq$	?
$\leq$	<	?	$\leq$	$\leq$	?	?	?
$\geq$	?	>	$\geq$	?	$\geq$	?	?
$\neq$	?	?	$\neq$	?	?	?	?
?	?	?	?	?	?	?	?

Algebra degli Intervalli (IA)

Insieme di relazioni derivato da tredici relazioni base attraverso le operazioni di **inverso** (denotato con $\cdot^{\smile}$), **unione** ($\cup$), **intersezione** ($\cap$) e **composizione** ($\circ$):

Relaz.	Abbr.	Inv.	Significato
x before y	b	bi	
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	
x equal y	eq	eq	

Notazione:

$$(I \text{ before } J) \vee (I \text{ meets } J) \equiv I \{before, meets\} J.$$

IA contiene 2^{13} relazioni.

Composizione di relazioni in IA

Relazioni di base (o atomiche): attraverso una *tabella di transitività* di 13×13 elementi.

$$I \{overlaps\} J \circ J \{during\} K = \\ I \{overlaps, during, starts\} K$$

Relazioni complesse:

$$I \{r_{11}, \dots, r_{1m}\} J \circ J \{r_{21}, \dots, r_{2n}\} K = \\ I \bigcup_{i,j=1}^{i=m, j=n} \{r_{1i} \circ r_{2j}\} K$$

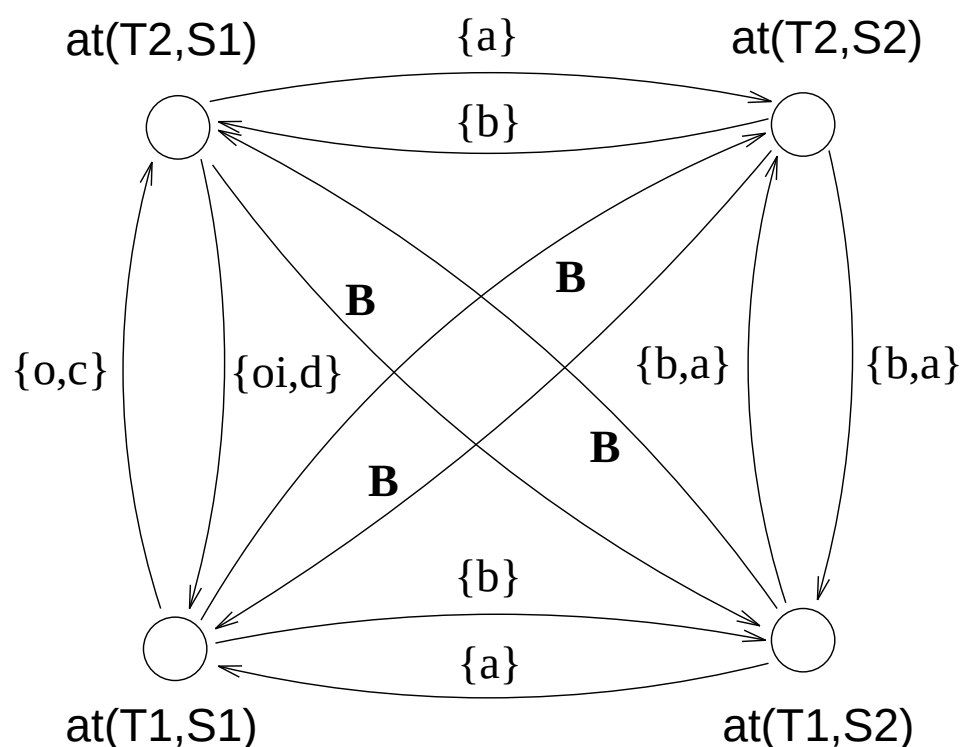
$$(1 \leq m, n \leq 13)$$

Esempio:

$$I \{during, finishes\} J \circ J \{before\} K = \\ I \{before\} K$$

Temporal Constraint Network (TCN)

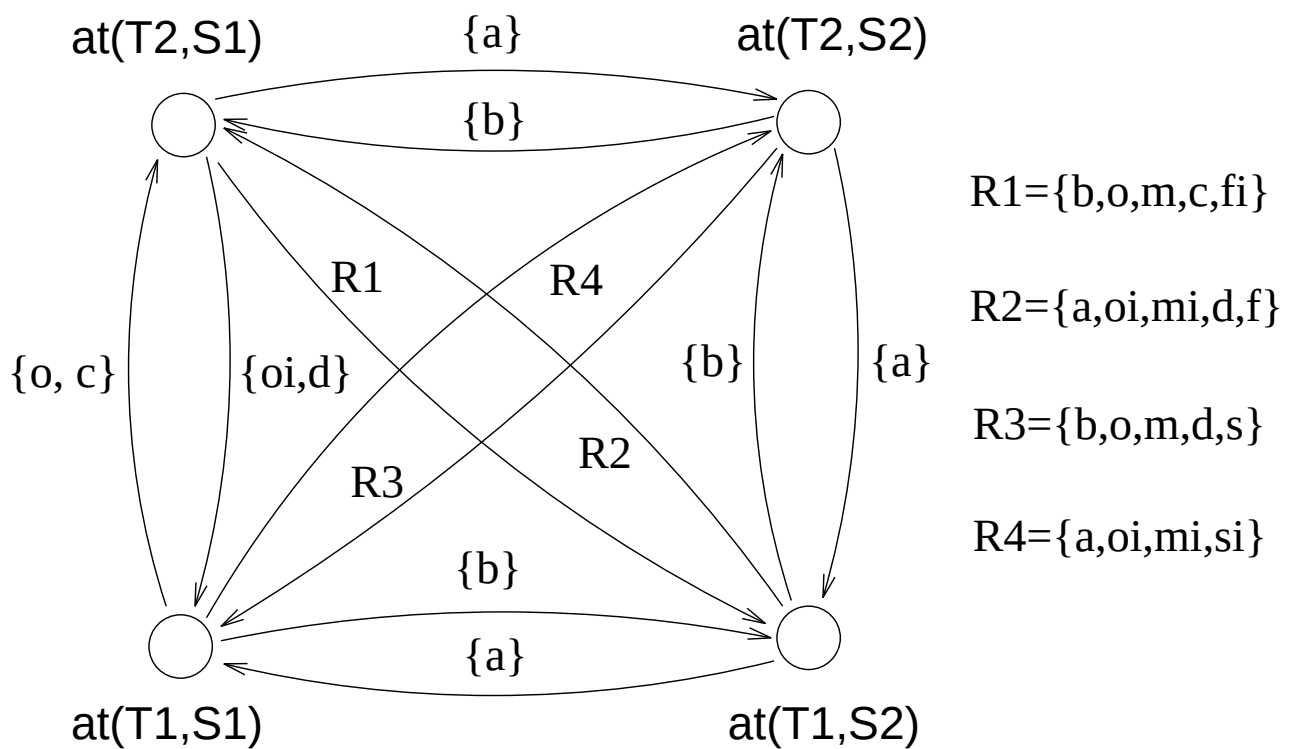
- *temporal constraint network*: un insieme di vincoli può essere rappresentato attraverso una **rete di vincoli** (Montanari 1974).
- I vertici di una TCN rappresentano variabili-intervallo (o variabili-punto); gli archi sono etichettati dalle relazioni che valgono tra queste variabili.
- Una porzione della rete di vincoli temporali per l'esempio dei treni (**B** è la relazione “universale” = insieme delle 13 relazioni di base):



Proprietà Reti di Vincoli

- Due reti sono **equivalenti** se gli insiemi di vincoli rappresentati ammettono le stesse soluzioni (scenari consistenti).
- Una relazione (vincolo) xRy è **più forte** di xSy se $xRy \models xSy$, ma $xSy \not\models xRy$ (es.: $x < y$ é più forte di $x \leq y$).
- Una rete di vincoli $\mathcal{T}$ é più forte di una rete $\mathcal{S}$ se i vincoli di $\mathcal{T}$ sono più forti dei vincoli di $\mathcal{S}$.
- Data una rete $\mathcal{T}$, la **rete minima** di $\mathcal{T}$ è la rete più forte equivalente a $\mathcal{T}$.
- In una rete minima per ogni coppia di valori v_1 e v_2 ammessi da un vincolo xRy (per ogni relazione atomica r in R), esiste una soluzione (scenario consistente) in cui $x = v_1$ e $y = v_2$ (xry).

Porzione Rete Minima per Esempio Treni



Determinare la consistenza (soddisfacibilità)

- ISAT per IA è NP-completo $\Rightarrow$ non esiste un algoritmo completo polinomiale (se $P \neq NP$).
- PSAT per PA é polinomiale.
- ISAT é polinomiale per alcune sottoclassi di IA:
Algebra Semplice degli Intervalli (SIA), Algebra ORD-Horn.

SIA = 188 relazioni di IA che sono traducibili come congiunzioni di vincoli in PA tra estremità degli intervalli.

ORD-Horn = Relazioni di IA che sono traducibili come congiunzioni di disgiunzioni di vincoli in $\{=, \leq, \neq\}$, con *al massimo un disgiunto di tipo " $\leq$ " o " $=$ ".*

Path-consistenza

- Una rete (o insieme di vincoli) $\mathcal{T}$ è *path consistente* se

$$\forall i, j, k \quad R_{ij} \subseteq R_{ik} \circ R_{kj}$$

R_{ij} = relazione tra i e j in $\mathcal{T}$.

- Se una rete (insieme) di vincoli è path-consistente, allora tutte le sottoreti (sottoinsiemi) che coinvolgono tre vertici (variabili) sono minime(i).
- Data una rete (insieme) di vincoli in IA o in PA, è possibile calcolare una rete path-consistente equivalente in tempo $O(n^3)$ (n = numero di variabili).
- *Un algoritmo per calcolare la path-consistenza è sufficiente a decidere la consistenza per PA, SIA e ORD-Horn: l'insieme di input è inconsistente se e solo se viene generata la relazione vuota.*

Algoritmo Path-Consistency(C)

Input: una matrice C che rappresenta una rete con n var.

Output: una rete PC equivalente a C , oppure *false*.

```
1.   $L \leftarrow \{(i, j) \mid 1 \leq i < j \leq n\}$ 
2.  while ( $L$  is not empty) do
3.      select and delete an item  $(i, j)$  from  $L$ 
4.      for  $k \leftarrow 1$  to  $n$ ,  $k \neq i$  and  $k \neq j$ , do
5.           $t \leftarrow R_{ik} \cap R_{ij} \circ R_{jk}$ 
6.          if  $t$  is the empty relation then
7.              return false
8.          else
9.              if  $t \neq R_{ik}$  then
10.                   $R_{ik} \leftarrow t$ 
11.                   $R_{ki} \leftarrow \text{Inverse}(t)$ 
12.                   $L \leftarrow L \cup \{(i, k)\}$ 
13.                   $t \leftarrow R_{kj} \cap R_{ki} \circ R_{ij}$ 
14.                  if  $t$  is the empty relation then
15.                      return false
16.                  else
17.                      if  $t \neq R_{kj}$  then
18.                           $R_{kj} \leftarrow t$ 
19.                           $R_{jk} \leftarrow \text{Inverse}(t)$ 
20.                           $L \leftarrow L \cup \{(k, j)\}$ 
21.  return  $C$ 
```

R_{ij} = relazione tra la i -esima var e la j -esima var, memorizzate in $C[i, j]$ of C . Inverse = operazione di inverso.

Consistenza per relazioni in IA

Algoritmo IA-consistency(C)

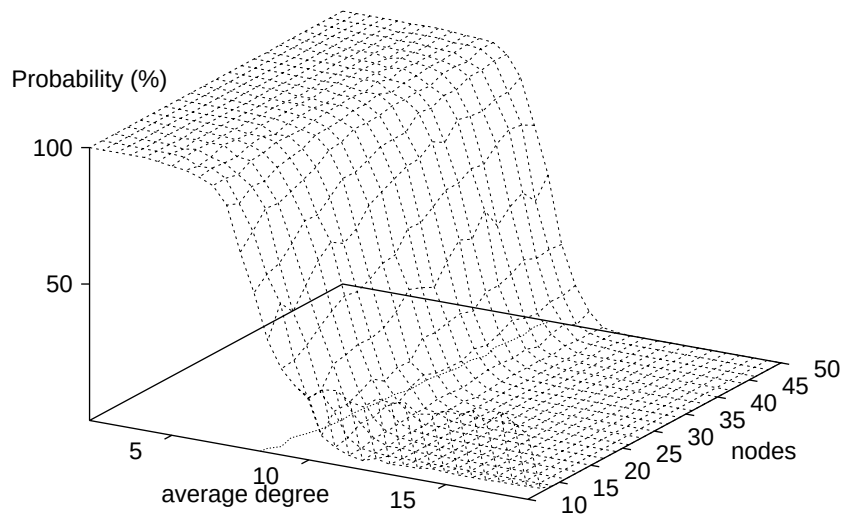
Input: Una matrice C che rappresenta un insieme Θ di vincoli

Output: *true* if Θ is consistente, *false* altrimenti

1. $C \leftarrow \text{path-consistency}(C)$
2. **if** $C = \text{false}$ **then**
3. **return** *false*
4. **else** scegli una relazione $C[i, j]$ non processata e
5. dividi $C[i, j]$ in $R_1, \dots, R_k$ tale che $R_l \in \text{Split}$
 ($1 \leq l \leq k$)
6. **if** nessuna relazione puo' essere divisa **then**
7. **return** *true*
8. **for** $l \leftarrow 1$ **to** k **do**
9. $C_{ij} \leftarrow R_l$
10. **if** IA-consistency(C) **then**
11. **return** *true*
12. **return** *false*

Split = sottoinsieme di relazioni “trattabili” (polinomiali) in IA (ad es., SIA o ORD-Horn).

Probability of satisfiability for label size 6.5

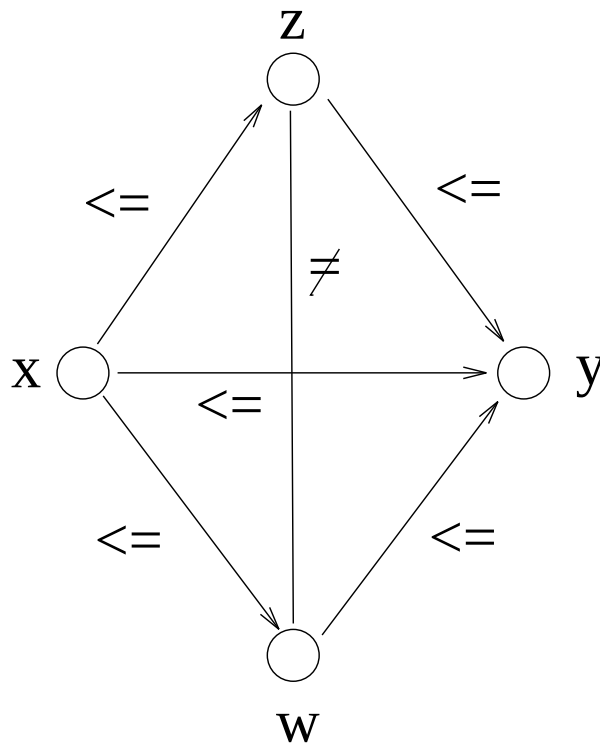


Nebel's Phase transition of ISAT for IA when the label size is 6.5

Rete Minima per PA

Una rete di vincoli in PA che è path-consistente é consistente, ma *non è minima*.

Teorema: *Se una rete path-consistente di vincoli in PA non è minima, allora contiene almeno un **grafo proibito**.*



La rete non è minima perchè $x = y$ non è possibile: $x \leq y$ deve essere $x < y$.

Eliminazione Grafi Proibiti

Algoritmo Elimina-Grafi-Proibiti(C)

Input: una matrice C che rappresenta una rete path-consistente

Output: una matrice che rappresenta una rete minima equivalente a C

1. **Per ogni** arco (v, w) tale che $w \in adj_{\neq}(v)$ **do**
2. $S \leftarrow (adj_{\geq}(w) \cap adj_{\geq}(w))$
3. $T \leftarrow (adj_{\leq}(w) \cap adj_{\leq}(w))$
4. **Per ogni** $s \in S, t \in T$ **do**
5. $C[s, t] \leftarrow "<"$
6. $C[t, s] \leftarrow ">"$

$$adj_R(v) = \{t \mid C[v, t] = R\}$$

Esempio Relazioni Temporal Quantitative

- La durata del viaggio dei treni T1 e T2 deve essere inferiore a 48 ore;
- Il viaggio di T1 da C1 a S1 dura 6-8 ore; il viaggio di T2 da S1 a C1 dura 6-8 ore.
- Il viaggio di T1 da S1 S2 dura 8-12 ore; il viaggio di T2 da S2 a S1 dura 8-12 ore.
- Il viaggio di T1 da S2 a C2 dura 14-20 ore; il viaggio di T2 da C2 a S2 dura 14-20 ore.
- T1 e T2 si fermano a ciascuna stazione per almeno 2 ore.
- T1 ha lasciato C1 alle 24:00 del 11/12/97. Si è fermato a S1 per 6 ore, ed è arrivato a C2 alle 16:00 del 12/12/97. T2 ha lasciato C2 alle 11:00 del 10/12/97, si è fermato 3 ore a S2 e 4 ore a S1, ed ha impiegato 9 ore per raggiungere C1 da S2.

Esempio (continuazione)

Esempi di interrogazioni che richiedono ragionamento:

- Le informazioni temporali sono consistenti ?
- Quali sono i possibili tempi di arrivo di T2 a C1?
- Quanto tempo ha impiegato T1 per andare da S2 a C2 ?
- T2 é arrivato a C2 prima che T1 fosse arrivato a C1 ?
- È possibile che T2 non fosse ancora arrivato a S1 alle 14:00 del 11/12/97 ?

Approccio di Dechter Meiri & Pearl (1990)

Vincoli sulle *distanze temporali*

$$y - x \in \{I_1, I_2, \dots, I_n\} \equiv$$

$$y - x \in I_1 \vee y - x \in I_2 \vee \dots \vee y - x \in I_n$$

$$I_i = [\min_i, \max_i] \rightarrow y - x \in I_i \equiv \min_i \leq y - x \leq \max_i$$

Se $n = 1$ i vincoli sono *semplici* e formano un *Simple Temporal Constraint Satisfaction Problem (STP)*.

Alcune estensioni

Intervalli aperti: ad es. $y - x \in (1, 10]$

inclusione di $\neq$:

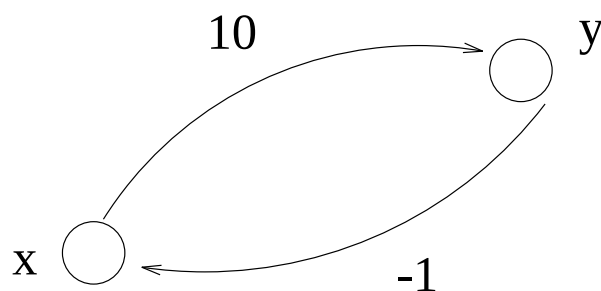
$$y - x \in [1, 10] - \{5, 7\} \equiv y - x \in \{[1, 5), (5, 7), (7, 10]\}$$

Grafo delle Distanze

Un STP S può essere rappresentato da un grafo delle distanze (GD): se $y - x \in I$ è in S , allora

- x e y sono vertici di GD;
- GD contiene un arco da x a y con etichetta I^+ (l'estremo superiore di I);
- GD contiene un arco da y a x con etichetta I^- (l'estremo inferiore di I).

Esempio (rappresentazione di $y - x \in [1, 10]$)



Teorema: *Un STP S è consistente se e solo se il grafo delle distanze di S non contiene cicli negativi (la cui somma delle etichette è un valore negativo).*

Algoritmi per STP

- Gli algoritmi di *Bellman-Ford* e di *Floyd-Warshall* sono completi per determinare la consistenza di un STP e richiedono tempo cubico ($O(n^3)$ – n = numero variabili).
- **Bellman-Ford** = algoritmo per calcolare i cammini più brevi tra un vertice e tutti gli altri di un DG.
- **Floyd-Warshall** = algoritmo per calcolare i percorsi più brevi tra tutte le coppie di vertici di un DG.
- Bellman-Ford è più efficiente dal punto di vista spaziale (complessità $O(e)$ – e = numero di vincoli in input).
- Floyd-Warshall può essere usato per calcolare la rete minima.

Floyd-Warshall

(all-pairs-shortest-paths)

1. FOR $i = 1$ TO n DO $D_{ii} \leftarrow 0$;
2. FOR $i, j = 1$ TO n DO $D_{ij} \leftarrow a_{ij}$;
3. FOR $k = 1$ TO n DO
4. FOR $i, j = 1$ to n DO
5. $D_{ij} \leftarrow \min\{D_{ij}, D_{ik} + D_{kj}\}$

a_{ij} = etichetta sull'arco da i a j ne grafo delle distanze.

Se tra i e j non c'è arco, si assume un arco con etichetta "infinito".

Per ogni x, y vale $-\infty \leq y - x \leq +\infty$

D_{ij} rappresentato con una matrice.

Disjunctive Temporal Problems

A **DTP** (Stergiou & Koubarakis) is a pair $\langle \mathcal{P}, \mathcal{C} \rangle$, where

- $\mathcal{P}$ is a set of time point variables over $\mathbb{R}$
- $\mathcal{C}$ is a set of disjunctive constraints $c_1 \vee \dots \vee c_n$ such that
 - c_i is of form $y_i - x_i \leq k_i$,
 - $x_i, y_i \in \mathcal{P}$ and $k_i \in \mathbb{R}$ (for $i = 1 \dots n$).

Reasoning problems:

- *Decide satisfiability* (all constraints can be satisfied)
- *Find a solution* (consistent interpretation of the variables)
- *Find an optimal solution* (lowest time for the one/all variables)

In general NP-hard, but polynomial with unary constraints (**STPs**)

DTP and “Meta CSP”

Finding a solution for a DTP $\Rightarrow$ solving a meta CSP:

- *Meta variables*: constraints of the DTP
- *Meta variable values*: constraint disjuncts
- *Implicit meta constraint*: the values (constraint disjuncts) of the meta variables form a satisfiable STP

Solution of the meta CSP = consistent STP with exactly one disjunct for each constraint in the original DTP

In general, NP-hard task solvable by a *backtracking algorithm*.

Backtracking Algorithm

Input: The set X of meta-variables in the meta CSP of a DTP, a partial solution S of the meta CSP;

Output: Either a solution of the meta CSP or *fail*.

1. **if** $X = \emptyset$ **then stop** and **return** S ;
2. $x \leftarrow \text{SelectVariable}(X)$; $X' \leftarrow X - \{x\}$;
3. **while** $D(x) \neq \emptyset$ **do**
4. $d \leftarrow \text{SelectValue}(D(x))$;
5. $S' \leftarrow S \cup \{x \leftarrow d\}$; $D(x) \leftarrow D(x) - \{d\}$;
6. $D'(x) \leftarrow D(x)$; /* Saving the domain values */
7. **if** $\text{ForwardCheck-DTP}(X', S')$ **then**
8. $\text{Solve-DTP}(X', S')$;
9. $D(x) \leftarrow D'(x)$; /* Restoring the domain values */
10. **return fail**; /* backtracking */

$\text{ForwardCheck-DTP}(X, S)$

Input: The set X of meta-variables, a (partial) solution S ;

Output: Either *true* or *false*.

1. **forall** $x \in X$ **do**
2. **forall** $d \in D(x)$ **do**
3. **if not** $\text{Consistency-STP}(S \cup \{x \leftarrow d\})$ **then**
4. $D(x) \leftarrow D(x) - \{d\}$;
5. **if** $D(x) = \emptyset$ **then return false**; /* dead-end */
6. **return true**.