Esercizio

- Si devono visitare 6 clienti A, B, C, D, E, F nell'arco della giornata lavorativa (dalle 9 alle 18).
- Due clienti non possono essere visitati contemporaneamente
- Si sa che i clienti C ed F devono essere visitati prima del cliente D.
- A è un cliente fuori città, mentre B, C, D, E ed F sono tutti in centro. Quindi, per spostarsi da A a ogni altro cliente si impiegano due ore, mentre qualunque spostamento in centro città viene effettuato a tempo trascurabile (=0).
- Il cliente C può essere visitato solo dalle 15 alle 17.
- Si modelli il problema in termini di variabili e vincoli. Si mostri l'albero di ricerca fino alla prima soluzione relativo alla strategia standard backtracking e quello relativo al forward checking e si commentino i risultati.

Soluzione

- Variabili: clienti
- Domini: possibili orari di visita A, B, C, D, E, F::[9..18]
- Vincoli:
  - Due clienti non possono essere visitati contemporaneamente:

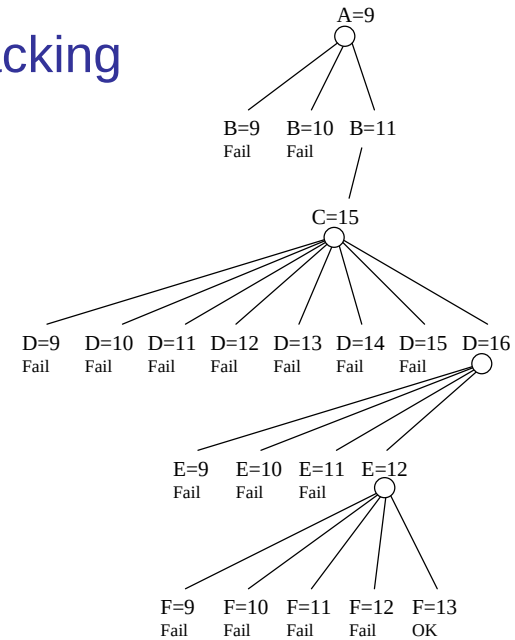
    $\forall X, Y \quad X \neq Y$
    - I clienti C ed F devono essere visitati prima del cliente D:

      $C < D \quad F < D$
  - A è un cliente fuori città, mentre B, C, D, E e F sono tutti in centro. Quindi, per spostarsi da A ad ogni altro cliente si impiegano due ore, mentre qualunque spostamento in centro città viene effettuato a tempo trascurabile (=0).

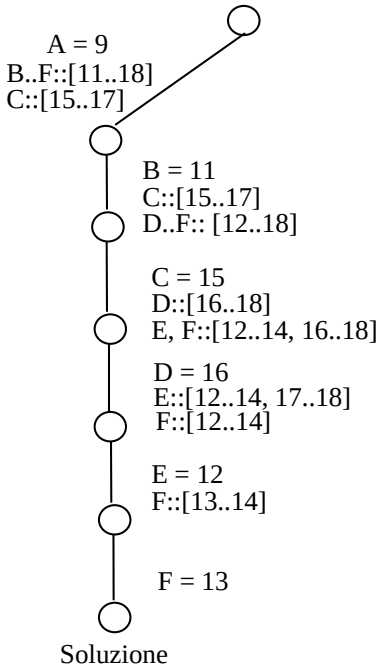
    $\forall X \in [B, C, D, E, F] \quad A \geq X + 2 \quad OR \quad X \geq A + 2$
  - Il cliente C può essere visitato solo dalle 15 alle 17. Vincolo unario su C che riduce il suo dominio

    $C \geq 15 \quad e \quad C \leq 17$

Std backtracking

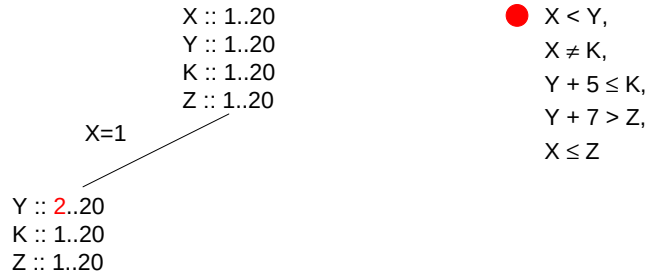


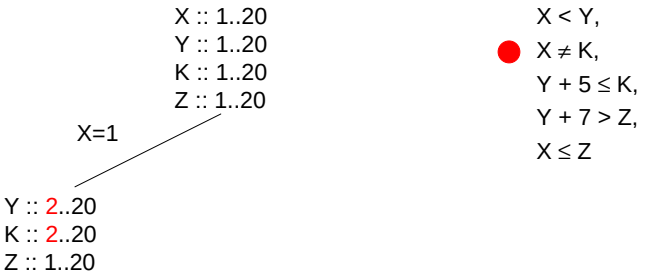
Forward Checking



ESERCIZIO

- Si supponga di avere a disposizione i seguenti vincoli:
- $X < Y, X \neq K, Y + 5 \leq K, Y + 7 > Z, X \leq Z$
- definiti sulle variabili X, Y, Z, K il cui dominio di definizione è [1..20].
- Si risolva il problema applicando la strategia di full look ahead.





X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20

X=1

Y :: 2..20
K :: 2..20
Z :: 1..20

X < Y,
X ≠ K,
Y + 5 ≤ K,
Y + 7 > Z,
● X ≤ Z

X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20

X=1

Y :: 2..15
K :: 7..20
Z :: 1..20

X < Y,
X ≠ K,
● Y + 5 ≤ K,
Y + 7 > Z,
X ≤ Z

X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20

X=1

Y :: 2..15
K :: 7..20
Z :: 1..20

X < Y,
X ≠ K,
Y + 5 ≤ K,
● Y + 7 > Z,
X ≤ Z

X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20

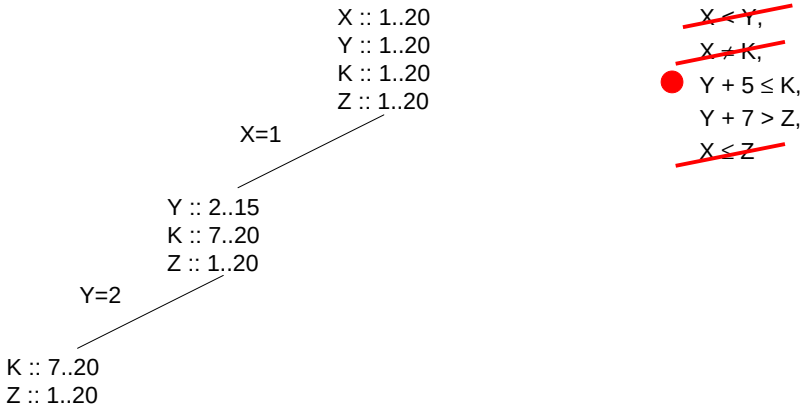
X=1

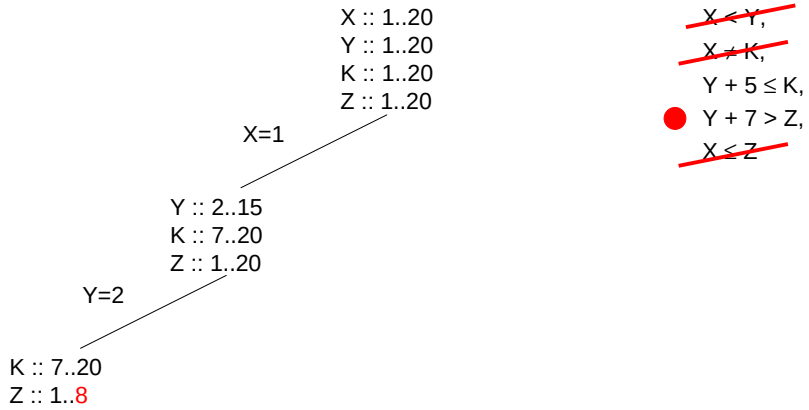
Y :: 2..15
K :: 7..20
Z :: 1..20

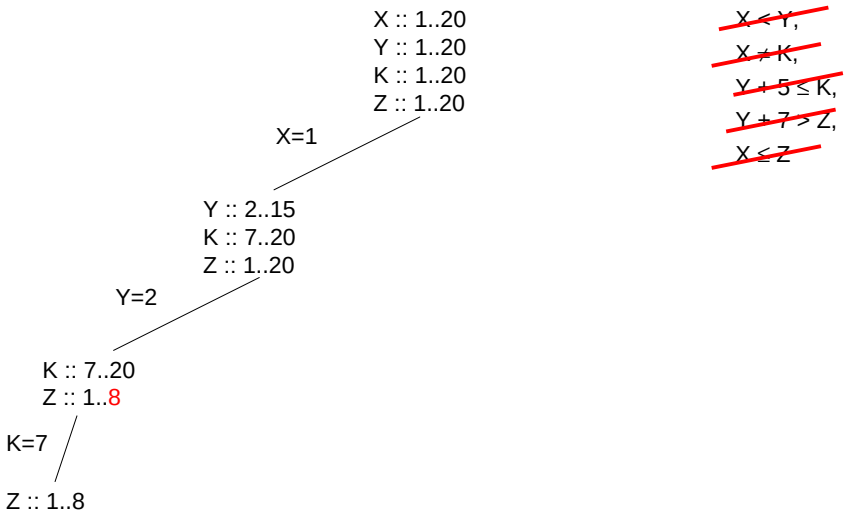
Y=2

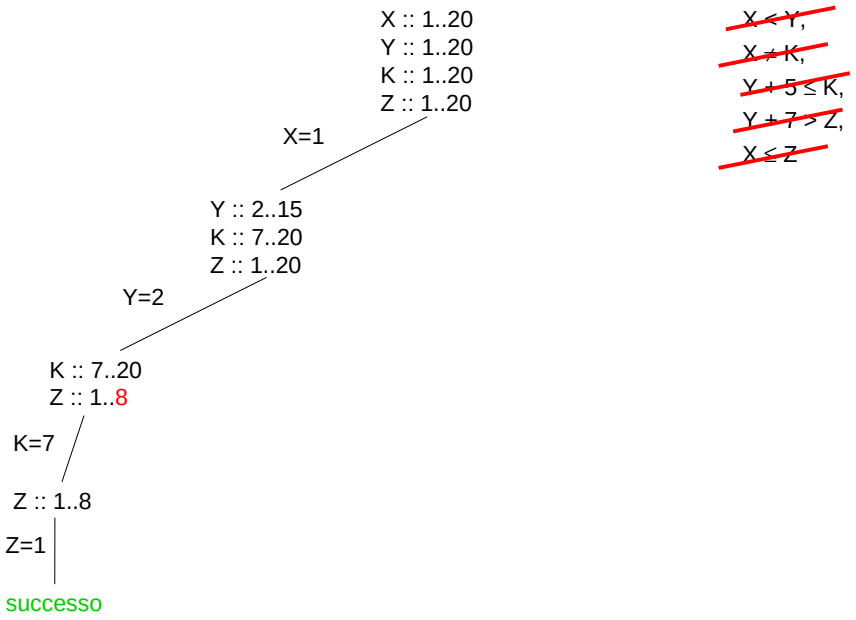
K :: 7..20
Z :: 1..20

~~X < Y,~~
~~X ≠ K,~~
Y + 5 ≤ K,
Y + 7 > Z,
~~X ≤ Z~~


X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20
X=1
Y :: 2..15
K :: 7..20
Z :: 1..20
Y=2
K :: 7..20
Z :: 1..20
X < Y,
X ≠ K,
Y + 5 ≤ K,
Y + 7 > Z,
X ≤ Z


X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20
X=1
Y :: 2..15
K :: 7..20
Z :: 1..20
Y=2
K :: 7..20
Z :: 1..8
X < Y,
X ≠ K,
Y + 5 ≤ K,
Y + 7 > Z,
X ≤ Z


X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20
X=1
Y :: 2..15
K :: 7..20
Z :: 1..20
Y=2
K :: 7..20
Z :: 1..8
K=7
Z :: 1..8
X < Y,
X ≠ K,
Y + 5 ≤ K,
Y + 7 > Z,
X ≤ Z


X :: 1..20
Y :: 1..20
K :: 1..20
Z :: 1..20
X=1
Y :: 2..15
K :: 7..20
Z :: 1..20
Y=2
K :: 7..20
Z :: 1..8
K=7
Z :: 1..8
Z=1
successo
X < Y,
X ≠ K,
Y + 5 ≤ K,
Y + 7 > Z,
X ≤ Z

16 gennaio 2001

- Nella prima settimana di Gennaio 2001, sei CD si trovavano in classifica rispettivamente nelle posizioni 1,2,3,4,5,6. Durante la seconda settimana, si trovano in nuove posizioni secondo i seguenti vincoli:
- i CD di posizione dispari sono CALATI in classifica (cioè sono in una posizione peggiore);
- i CD di posizione pari sono CRESCIUTI in classifica (cioè sono in posizione migliore);
- il CD 6 ha superato in classifica il 3, ma non l'1;
- il CD 4 non ha superato in classifica l'1;
- Si formuli il problema come Problema di Soddisfacimento di Vincoli, si applichi l'arc-consistenza alla rete risultante e si dica se il problema ammette una soluzione, più soluzioni o nessuna soluzione.

Soluzione

- Variabili: CD1, CD2, CD3, CD4, CD5 e CD6.
- Domini: {1,2,3,4,5,6}
- Vincoli:
- CD1 > 1, CD3 > 3, CD5 > 5
- CD2 < 2, CD4 < 4, CD6 < 6
- CD6 < CD3, CD6 > CD1
- CD4 > CD1
- alldifferent([CD1,CD2,CD3,CD4,CD5,CD6,CD7])
- **Arc-Consistenza:**
- CD1 :: {2}
- CD2 :: {1}
- CD3 :: {5}
- CD4 :: {3}
- CD5 :: {6}
- CD6 :: {4}
- Esiste quindi una ed una sola soluzione.

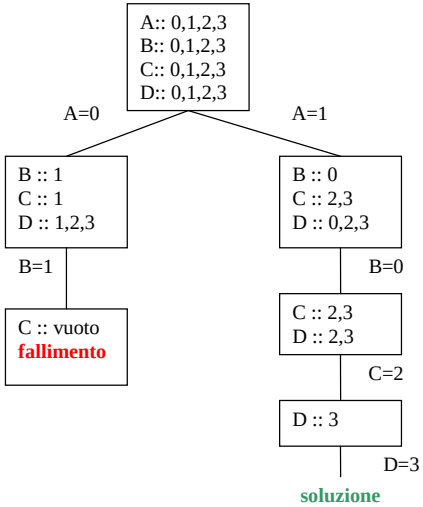
13 settembre 2006

- Per decidere le posizioni a bridge quattro amici, Aldo, Barbara, Claudio e Dino hanno distribuito i quattro assi. Ho chiesto di dirmi quale asso gli fosse capitato e mi hanno risposto:
- Aldo: Barbara ha un asso dello stesso colore del mio.
- Barbara: Claudio ha un asso nero.
- Claudio: Aldo ha l'asso di cuori.
- Dino: io ho l'asso di fiori.
- Io so che quelli che hanno una carta rossa dicono sempre la verità, mentre quelli che hanno una carta nera mentono sempre. Chi aveva l'asso di cuori?
- Si modelli il problema come CSP e lo si risolva con Forward Checking.

Soluzione

- Variabili: A,B,C,D
- Domini: 0,1,2,3 (convenzione 0= ♥, 1=♦, 2=♣, 3=♠)
- Con questa convenzione abbiamo che A < 2 indica che Aldo ha un asso rosso (e quindi dice la verità)
- Vincoli:
- A ≠ B, A ≠ C, A ≠ D, B ≠ C, B ≠ D, C ≠ D
- Aldo: Barbara ha un asso dello stesso colore del mio.
  B/2 = A/2 ⇔ A < 2
- Barbara: Claudio ha un asso nero.
  C > 1 ⇔ B < 2
- Claudio: Aldo ha l'asso di cuori.
  A = 0 ⇔ C < 2
- Dino: io ho l'asso di fiori.
  D = 2 ⇔ D < 2

Forward Checking



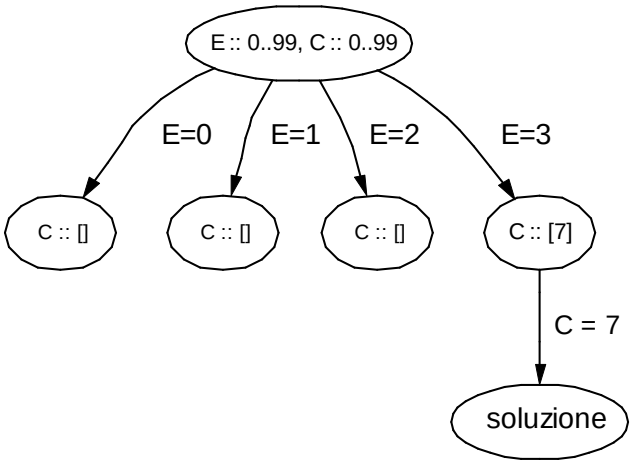
25 Mar 2002

- Formalizzare il seguente problema come Problema di Soddisfacimento di Vincoli:
- *"Ieri ho pagato un conto. Poi mi sono accorto di aver dato in Euro quello che dovevo in centesimi, e in centesimi il dovuto in Euro. Acc.. mi è costato più del doppio di quello che avrei dovuto, con una differenza di 89 centesimi. Qual era la cifra dovuta?"*
- Si risolva poi il problema utilizzando il Forward Checking.

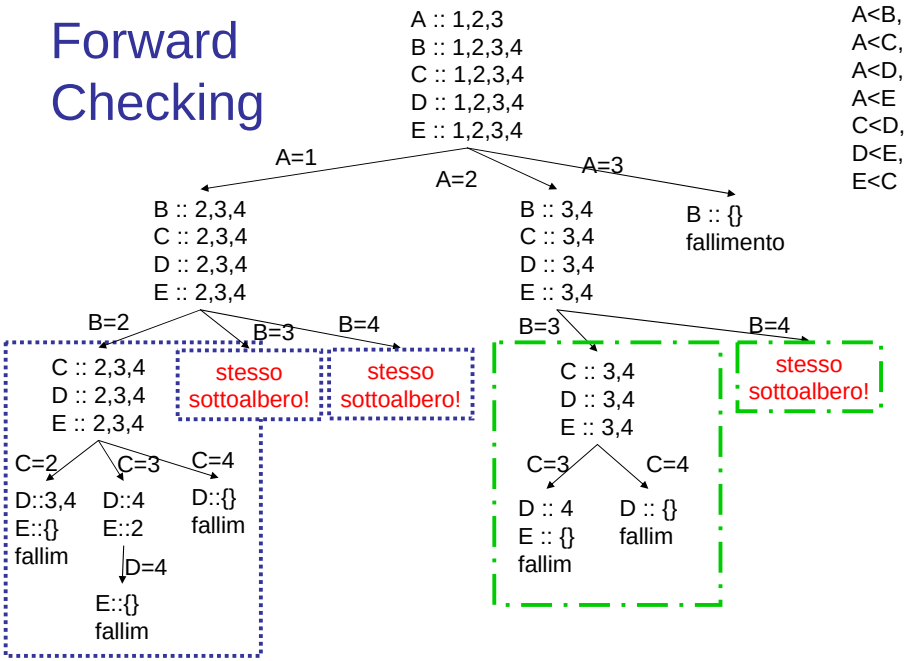
Formalizzazione CSP

- Abbiamo due variabili, che vanno da 0 a 99 (in quanto i centesimi possono andare solo da 0 a 99 e gli Euro sono scambiati con i centesimi nella cifra effettivamente pagata)
- Variabili: E, C
- Domini: 0..99
- Vincoli: 100*C+E = 2*(100*E + C)+89
- Visto che ho pagato più del dovuto, possiamo dire anche che C>E (vincolo ridondante).

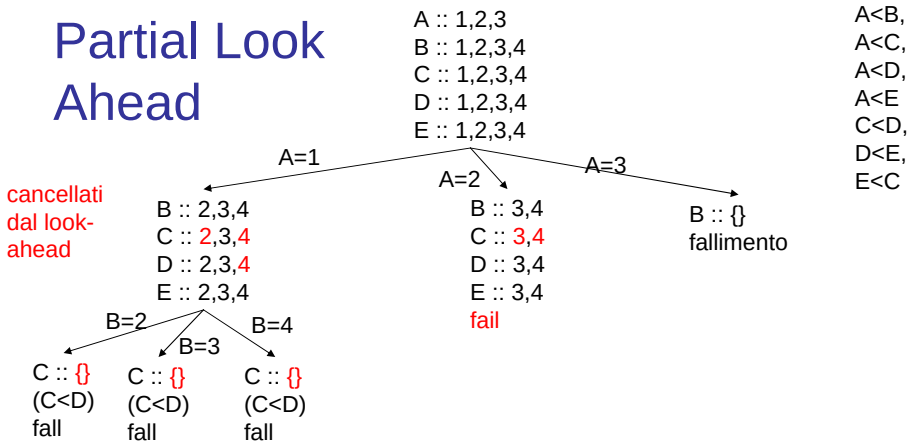
Forward Checking



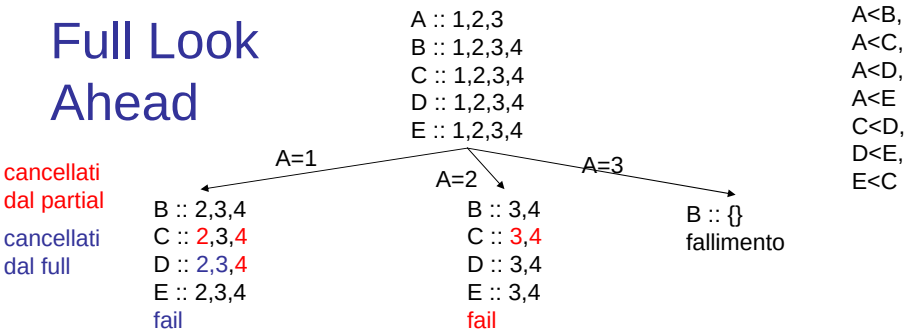
Forward Checking



Partial Look Ahead



Full Look Ahead



1 Aprile 2003

- Si consideri il seguente problema: si devono allocare 6 task t1, …, t6 le cui durate sono rispettivamente 3,4,5,4,7,6 su due risorse m1 e m2.
- Ad ogni task è associata la risorsa su cui viene eseguito:
  (t1, m2), (t2,m1), (t3,m2), (t4,m1), (t5,m1) (t6,m2).
- Su ogni risorsa può essere in esecuzione un solo task alla volta.
- Esistono dei vincoli di precedenza tra le seguenti coppie di task (t2,t4), (t1,t3), (t4,t5), (t1,t6), in cui il secondo task deve iniziare dopo o nello stesso istante in cui il primo finisce su una qualunque risorsa.
- Le risorse sono disponibili dall'istante 1. E' infine necessario che tutti i task terminino entro l'istante di tempo 16,
- Si modelli il problema come un problema di soddisfacimento di vincoli.
- Sul problema iniziale, si applichi la arc consistenza ai soli vincoli di precedenza.
- Infine lo si risolva utilizzando il forward checking con euristica first-fail

Modello

- Ad ogni task è associata una variabile che rappresenta l'inizio del task. Inoltre, ad ogni task è associata una variabile risorsa
- **Start1::[1..13] R1=m2**
- **Start2::[1..12] R2=m1**
- **Start3::[1..11] R3=m2**
- **Start4::[1..12] R4=m1**
- **Start5::[1..9] R5=m1**
- **Start6::[1..10] R6=m2**
- **Start2 + 4 ≤ Start4**
- **Start1 + 3 ≤ Start3**
- **Start4 + 4 ≤ Start5**
- **Start1 + 3 ≤ Start6**
- Se **Ri = Rj** $\rightarrow$ **Starti + di ≤ Startj** $\vee$ **Startj + dj ≤ Starti**

Arc-Consistenza sui vincoli di precedenza

- **Start1::[1..13] R1=m2**
- **Start2::[1..12] R2=m1**
- **Start3::[1..11] R3=m2**
- **Start4::[1..12] R4=m1**
- **Start5::[1..9] R5=m1**
- **Start6::[1..10] R6=m2**
- **Start2 + 4 ≤ Start4**
- **Start1 + 3 ≤ Start3**
- **Start4 + 4 ≤ Start5**
- **Start1 + 3 ≤ Start6**
- Arc consistenza su vincoli di precedenza:
- **Start2 + 4 ≤ Start4** $\rightarrow$ **Start2::[1..8] Start4::[4..12]**
- **Start4 + 4 ≤ Start5** $\rightarrow$ **Start4::[5] Start5::[9]**
- Risveglio
- **Start2 + 4 ≤ Start4** $\rightarrow$ **Start2::[1] Start4::[5]**
- **Start1 + 3 ≤ Start3** $\rightarrow$ **Start1::[1..8] Start3::[4..11]**
- **Start1 + 3 ≤ Start6** $\rightarrow$ **Start1::[1..7] Start6::[4..10]**

Forward Checking

- **Arc Consistency**
  - **Start1::[1..7]**
  - **Start2::[1]**
  - **Start3::[4..11]**
  - **Start4::[5]**
  - **Start5::[9]**
  - **Start6::[4..10]**
- **Forward checking**
  - **Start2 = 1 ➔ Domini non cambiano**
  - **Start4 = 5 ➔ Domini non cambiano**
  - **Start5= 9 ➔ Domini non cambiano**
  - **Start1 = 1 ➔ Domini non cambiano**
  - **Start3 = 4 ➔ Start6::[9..10]**
  - **Start6= 9 ➔ soluzione**

Forward Checking vs Look Ahead

Si consideri il seguente CSP:

- variabili e domini
  A :: 1..3
  B,C,D,E :: 1..4
- vincoli
  A<B, A<C, A<D, A<E
  C<D, D<E, E<C

Si mostri l'albero esplorato dal Forward Checking dal Partial Look Ahead e da Full Look Ahead, istanziando le variabili in ordine lessicografico

16 Settembre 2005

- Individuare due pozioni (una che consente di avanzare tra le fiamme e una per tornare indietro sani e salvi) tra una serie di sette bottiglie dal contenuto sconosciuto allineate sul tavolo. Il prof. Snape ha fornito una serie di indizi
- Le pozioni servono per avanzare o per tornare indietro oppure contengono vino d'ortica o contengono veleno

1. C'è sempre del veleno a sinistra del vino d'ortica.
2. Le bottiglie alle estremità hanno contenuti diversi, ma nessuna di queste serve per andare avanti.
3. Né la bottiglietta più piccola, né quella gigante contengono veleno.
4. La seconda da sinistra e la seconda da destra hanno lo stesso contenuto.
5. La prima bottiglia non contiene ortica.

- Si formalizzi il problema come CSP

Modello CSP

- *Suppongo di numerare le bottiglie con 1, 2 … 7 da sinistra a destra*
- *Variabili: V1, V2 … V7*
- *Dom(Vi) = {A, I, O, V} dove A=avanti, I=indietro, O=vino d'ortica, V=veleno*
- *Vincoli:*
  - *Vi = O => Vi-1 = V (i=2, … 7)*
  - *V1≠V7; V1≠A; V7≠A;*
  - *V4≠V; V6≠V*
  - *V2=V6*
  - *V1≠O*

12 Gennaio 2006

- Un impiegato del comune, in fase di censimento, ha qualche informazione sugli abitanti di una palazzina: i signori Alberti, Bianchi, Carli, Doni, Elmi e Ferri che abitano (non necessariamente in questo ordine) negli appartamenti 1,2,3,4,5,6. Gli appartamenti si trovano su tre piani, due per piano. Gli appartamenti 1 e 2 al primo piano, il 3 e 4 al secondo piano e il 5 e 6 al terzo piano.
- L'impiegato sa che i signori Alberti e Carli non abitano sullo stesso piano, i signori Bianchi e Ferri abitano a un piano più alto (maggiore) rispetto a quello in cui abita il signor Elmi. Doni abita a un piano più alto (maggiore) rispetto e Carli. Infine, Alberti e Elmi abitano allo stesso piano.
- Può l'impiegato del comune in possesso di queste informazione dire a priori se esiste una sola soluzione? E se no come può trovarne una ammissibile?
- Si modelli il problema come un CSP applichi alla rete iniziale la tecnica della arc consistenza. Si proceda poi alla ricerca di una soluzione applicando il forward checking. Come euristica di selezione delle variabili si usi il first fail.

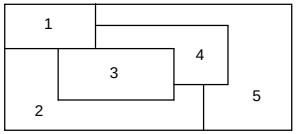
Soluzione

- Per il problema non esiste mai un'unica soluzione. Infatti, o il problema è insoddisfacibile oppure esistono soluzioni simmetriche. Non è specificato tra i vincoli se un inquilino abita nel primo o nel secondo appartamento di un piano. Pertanto il problema può essere modellato usando all'interno dei domini i piani. In tal modo, è possibile che il problema abbia una unica soluzione (riferendosi ai piani) e una volta individuati il piano di ciascun individuo, è possibile generare tutte le soluzioni ammissibili del problema.
- Si modelli il problema con sei variabili A,B,C,D,E,F ciascuna inizialmente con dominio contenente i piani [1..3]
- I vincoli sono i seguenti
- Alberti e Carli non abitano sullo stesso piano: A ≠ C
- Bianchi abita a un piano più alto rispetto a quello in cui abita il signor Elmi: B > E
- Ferri abita a un piano più alto rispetto a quello in cui abita il signor Elmi: F > E
- Doni abita a un piano più alto rispetto e Carli: D > C
- Alberti e Elmi abitano allo stesso piano: A = E
- Due inquilini per piano a posteriori ossia appena due variabili assumono lo stesso valore questo valore e' cancellato dai domini delle altre variabili

- Si consideri il problema di colorare la mappa seguente usando solo 4 colori in modo che nessuna regione adiacente condivida lo stesso colore.

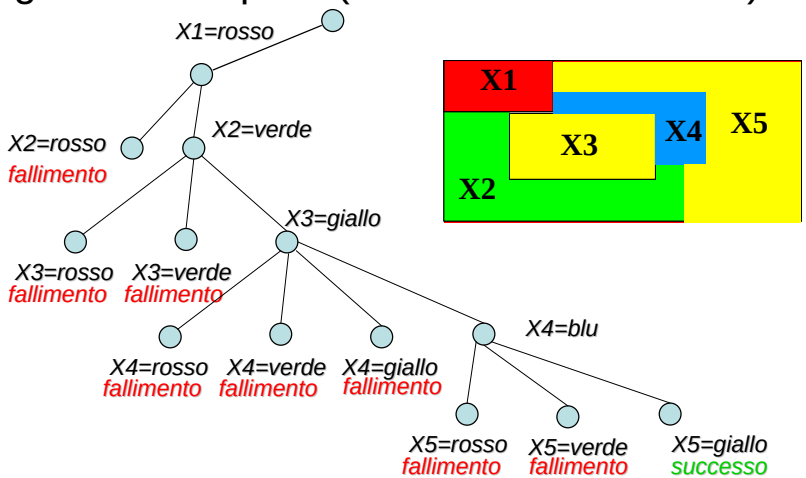
Problemi di soddisfacimento di vincoli

- trovare un assegnamento di valori alle variabili che soddisfa tutti i vincoli
- Es: Colorazione di mappe
  - Ho a disposizione 4 colori per colorare una mappa politica ➔ stati confinanti devono avere diverso colore



Backtracking

- Algoritmo semplice (ma a volte inefficiente)



Propagazione di vincoli

- Eliminazione a priori dei valori inconsistenti