Constraint Satisfaction Problems

Le variabili sono legate tra loro da vincoli, specifici del problema

**Labeling**: Operatori, assegnamento di un valore ad una variabile della n-pla

La **DIM dell'albero decisionale** è data dal modello del problema:

* #variabili, determina il #operatori di labeling necessari per arrivare a una configurazione finale (profondità d)
* #valori possibili per una variabile, determina il fattore di ramificazione (branching, b)

Il livello di profondità massimo ha tutte le variabili istanziate, e le possibili configurazioni finali sono in numero bd.

**Per tagliare a-priori le foglie che non sono soluzione** si possono utilizzare i vincoli del problema:

* Staticamente , prima di effettuare alcun labeling (tecniche di consistenza)
* Dinamicamente (frammischiati al labeling), durante la ricerca (metodi generativi e algoritmi di propagazione)

Una **soluzione ad un CSP** prevede un **assegnamento di tutte le variabili** che **soddisfi tutti i vincoli**.

* Un possibile albero decisionale per un CSP si ottiene (dopo aver stabilito un ordinamento per le variabili) facendo corrispondere ad ogni livello dell'albero l'assegnamento di una variabile e ad ogni nodo la scelta di un possibile valore da dare alla variabile corrispondente al livello del nodo stesso.
* Ogni foglia dell'albero rappresenterà quindi un assegnamento di valori a tutte le variabili. Se tale assegnamento soddisfa tutti i vincoli, allora la foglia corrispondente rappresenterà una soluzione del problema, altrimenti rappresenterà un fallimento.
* La ricerca di una soluzione è equivalente all'esplorazione dell'albero decisionale per trovare una foglia-soluzione.

Escludere rami che non portano a soluzione è fondamentale:

* **Tecniche di consistenza**: Basati sulla propagazione dei vincoli per derivare un problema più semplice di quello (completo) originale.
* **Algoritmi di propagazione**: Basati sulla propagazione dei vincoli per eliminare a priori, ma durante la ricerca, porzioni dell'albero decisionale che porterebbero ad un sicuro fallimento (compatibilmente con le scelte già effettuate)

Uso dei vincoli a priori e posteriori

Consideriamo una ricerca depth-first in un albero decisionale (i livelli sono N, albero di ricerca finito e senza cicli, no loop)

Si tende a scendere di livello nell'albero fino a quando o si sono assegnate tutte le variabili, e quindi si è trovata una soluzione, oppure non è più possibile trovare un valore (la sequenza corrente non può portare a una soluzione ammissibile); quindi si esegue un'altra scelta sull'ultima variabile della sequenza stessa.

L'algoritmo ha **tre gradi di libertà**:

* La scelta nell'ordinamento delle variabili
* La scelta nell'ordine di selezione del valore da attribuire alla variabile corrente
* La propagazione effettuata in ciascun nodo.

La verifica (senza propagazione) dei vincoli può avvenire a posteriori, considerando le sole variabili già istanziate: Algoritmi (generativi)

* **Generate and Test**
* **Standard Backtracking**

La verifica e propagazione dei vincoli può avvenire a priori, anche su variabili non ancora istanziate: Algoritmi di Propagazione:

* **Forward Checking**
* **(Partial and Full) Look Ahead**

GENERATE AND TEST - GT

Generate&Test esplora depth-first l’albero decisionale, fino a una foglia, e solo allora verifica se è soluzione o meno.

STANDARD BACKTRACKING - SB

Anche questa tecnica prevede un utilizzo a posteriori dei vincoli, ma su porzioni di soluzione

* A ogni istanziazione di una variabile si preoccupa di verificare la consistenza della variabile appena istanziata con quelle assegnate già precedentemente.
* Quindi l'utilizzo dei vincoli è più efficace del Generate&Test perché non si prosegue la ricerca in rami che, già ai primi livelli dell'albero, presentano delle inconsistenze.

Gli **algoritmi di propagazione** sono metodi di ricerca più “intelligenti” (o meglio più efficaci nel ridurre lo spazio di ricerca esplorato) che **tentano di prevenire i fallimenti anziché recuperare fallimenti già avvenuti**. **Pruning a priori dell'albero** delle decisioni.

Utilizzano le relazioni tra le variabili del problema, ovvero i vincoli, per ridurre lo spazio di ricerca prima di arrivare al fallimento.

**Un modulo propaga i vincoli finché è possibile** (constrain); **alla fine della propagazione o si è giunti ad una soluzione** (od a un fallimento) **o sono necessarie nuove informazioni** sulle variabili libere (generate, fase di labeling di una variabile).

FORWARD CHECKING - FC

Si applica, **dopo ogni assegnamento**, la **propagazione dei vincoli** che consiste nell**'eliminazione dei valori incompatibili con quello appena istanziato** dai domini delle variabili non ancora istanziate.

1. Questo metodo si rivela molto efficace soprattutto quando le ultime variabili ancora libere sono associate ad un insieme di valori ammissibili ridotto e perciò risultano molto vincolate e facilmente assegnabili.
   1. Se il dominio associato ad una variabile libera presenta un solo valore, l'assegnamento può essere effettuato senza costo computazionale.
   2. Se ad un certo punto della computazione ci si accorge che un dominio associato ad una variabile risulta vuoto il meccanismo del FC fallisce senza proseguire in tentativi, e si attua il backtracking.
2. **L'assegnazione di un valore ad una variabile ha ripercussioni sull'insieme dei valori disponibili per le variabili ancora libere**. In questo modo i vincoli agiscono in avanti (forward) e **limitano lo spazio delle soluzioni prima che siano effettuati tentativi** su di esso.

LOOK AHEAD

Ad ogni istanziazione **viene controllata**, come per il FC, la **compatibilità dei vincoli contenenti la variabile appena assegnata con le precedenti** (istanziate) **e le successive** (libere).

In più viene sviluppato il **look ahead** (sguardo in avanti) che controlla **l'esistenza, nei domini associati alle variabili ancora libere, di valori compatibili con i vincoli contenenti solo variabili non istanziate**.

I domini associati a ogni variabile vengono ridotti propagando anche le relazioni contenenti coppie di variabili non istanziate. Viene verificata quindi la possibilità di una futura assegnazione consistente fra le variabili libere.

PARTIAL LOOK AHEAD

Si ha una **propagazione dei vincoli contenenti la variabile Xh, non ancora istanziata e le variabili "future"**, ossia le variabili Xh+1,..., Xn.

* **Per ogni variabile non ancora assegnata** Xk+1,...,Xn, **deve esistere un valore** per il quale sia possibile trovare, **per tutte le altre variabili "successive"** non ancora assegnate, **almeno un valore compatibile con esso**.

FULL LOOK AHEAD

Se Vk è il valore appena assegnato alla variabile Xk, **si ha una propagazione dei**

**vincoli contenenti la variabile Xh, non ancora istanziata**, **e tutte le variabili non ancora assegnate**, ossia le variabili Xk+1,...Xh-1,Xh+1..,Xn.

* Per ogni variabile non ancora assegnata Xk+1,....,Xn deve esistere un valore per il quale sia possibile trovare, per tutte le variabili non ancora assegnate, almeno un valore compatibile con esso.

Classificazione delle euristiche

1. Euristiche Statiche: determinano l'ordine in cui le variabili (o i valori) vengono scelti prima di iniziare la ricerca; tale ordine rimane invariato durante tutta la ricerca
2. Euristiche Dinamiche: scelgono la prossima selezione da effettuare ogni volta che una nuova selezione viene richiesta (quindi ad ogni passo di labeling).
3. Euristiche per la selezione della variabile: determinano quale deve essere la prossima variabile da istanziare. Le due euristiche più comunemente usate sono il
   1. First-Fail (o MRV: Minimum Remaining Values) che sceglie la variabile con il dominio di cardinalità minore
   2. Most-Constrained Principle che sceglie la variabile legata a più vincoli. Entrambe queste euristiche decidono di istanziare prima le variabili più difficili da assegnare.
4. Euristiche per la selezione del valore: determinano quale valore assegnare alla variabile selezionata. Si segue in genere il principio di scegliere prima il valore che si ritiene abbia più probabilità di successo (least constraining principle).

Tecniche di consistenza

Riducono il problema originale eliminando dai domini delle variabili i valori che non possono comparire in una soluzione finale.

NODE CONSISTENCY

Un nodo di un grafo di vincoli è consistente se per ogni valore v∊Di il vincolo unario su Xi è soddisfatto.

ARC CONSISTENCY

Un arco A(Xi, Xj) è consistente se per ogni valore x∊Di esiste almeno un valore y∊Dj tale che il vincolo tra Xi e Xj P(Xi,Xj) sia soddisfatto.

PATH CONSISTENCY

Un cammino tra i nodi (Xi,Xj ,Xk) è path consistente se, per ogni valore x∊Di, e y∊Dj (che rispettano la node e la arc-consistenza) esiste un valore z∊Dk che soddisfa i vincoli P(Xi,Xk) e P(Xk,Xj).

K CONSISTENCY

Scelti valori per ogni (k-1)-pla di variabili consistenti con i vincoli imposti dal problema, si cerca un valore per ogni k-esima variabile che soddisfa i vincoli tra tutte le k variabili. Se tale valore esiste allora le k variabili sono k consistenti.

* In generale, se un grafo contenente n variabili è k-consistente con k<n, allora per trovare una soluzione è necessaria una ricerca nello spazio restante.