**Riassunto Completo: Features and Constraints**

**4.1 Introduzione a Features and Constraints**

* **Concetti chiave**:
  + Ogni attività implica **vincoli**, che possono essere:
    - **Vincoli rigidi (hard constraints)**: Devono essere rispettati (es. restrizioni obbligatorie).
    - **Vincoli flessibili (soft constraints)**: Definiscono preferenze, con possibilità di violazione a un costo.
  + **Features** (caratteristiche): Rappresentano stati tramite variabili, facilitando il ragionamento sulle loro combinazioni.
* **Applicazioni comuni**:
  + **Scheduling**: Pianificazione rispettando orari rigidi e preferenze.
  + **Cruciverba**: Assegnazione di parole o lettere rispettando vincoli di coerenza.

**4.2 Possibili Mondi, Variabili e Vincoli**

* **Mondo possibile**:
  + Configurazione di valori assegnati alle variabili che soddisfa tutti i vincoli.
  + Esempio: Cruciverba con parole assegnate a quadrati numerati.
* **Tipologie di variabili**:
  + **Discrete**: Con domini finiti o numerabili (es. variabili Booleane con {true, false}).
  + **Continue**: Con domini rappresentati da intervalli reali.
* **Vincoli (Constraints)**:
  + **Hard constraints**: Combinazioni legali obbligatorie tra valori.
  + **Scope**: Set di variabili coinvolte in un vincolo.
  + **Rappresentazione**:
    - **Intensionale**: Definiti tramite predicati logici o funzioni booleane.
    - **Estensionale**: Definiti elencando esplicitamente combinazioni valide.

**4.3 Constraint Satisfaction Problems (CSP)**

* **Definizione di CSP**:
  + Include:
    1. Un set di variabili.
    2. Domini per ciascuna variabile.
    3. Un set di vincoli.
  + Obiettivo: Trovare un’assegnazione valida che soddisfi tutti i vincoli.
* **Compiti principali**:
  + Verificare se esiste una soluzione.
  + Contare o enumerare tutte le soluzioni.
  + Valutare la qualità di una soluzione (es. minimizzare i costi).

**4.4 Algoritmi di Ricerca per CSP**

**4.4.1 Generate-and-Test**

* **Metodo base**:
  + Genera tutte le configurazioni possibili.
  + Testa ogni configurazione rispetto ai vincoli.
* **Limiti**:
  + Complessità esponenziale: O(dn)O(d^n), dove dd è la dimensione del dominio e nn il numero di variabili.

**4.4.2 Backtracking**

* **Procedura**:
  + Esplora lo spazio degli stati con una ricerca DFS.
  + Elimina sottospazi inconsistenti tramite pruning.
* **Vantaggi**:
  + Riduzione significativa dello spazio di ricerca rispetto a "Generate-and-Test".
  + Verifica parziale dei vincoli durante la ricerca.

**4.5 Algoritmi di Consistenza**

* **Definizione**:
  + Rendono l’intero sistema consistente rimuovendo valori non validi dai domini.
* **Tipologie**:
  + **Domain Consistency**: Ogni valore di una variabile deve soddisfare i vincoli con le variabili connesse.
  + **Arc Consistency**:
    - Ogni valore di una variabile deve avere corrispondenze valide nei valori delle variabili connesse.
* **Algoritmo Generalized Arc Consistency (GAC)**:
  + Rimuove valori non consistenti dai domini.
  + Complessità: O(e⋅d3)O(e \cdot d^3), dove ee è il numero di archi e dd la dimensione del dominio.

**4.6 Suddivisione dei Domini**

* **Procedura**:
  + Divide il dominio di una variabile in sottoinsiemi disgiunti.
  + Risolve il problema per ciascun sottoinsieme separatamente.
* **Approccio combinato**:
  + Applica consistenza di arco per ridurre i domini.
  + Divide il dominio per variabili con più valori.

**4.7 Eliminazione delle Variabili**

* **Variable Elimination (VE)**:
  + Rimuove iterativamente variabili creando nuovi vincoli tra quelle rimanenti.
* **Benefici**:
  + Riduce la dimensione del problema.
  + Rende più gestibile il CSP per problemi complessi.

**4.8 Ricerca Locale**

* **Caratteristiche**:
  + Non garantisce una soluzione globale, ma efficace per problemi dove le soluzioni esistono.
  + Parte da un’assegnazione casuale e la migliora iterativamente.
* **Strategie**:
  + **Hill Climbing**:
    - Massimizza una funzione obiettivo.
    - Rischia di restare bloccata in minimi locali.
  + **Simulated Annealing**:
    - Accetta soluzioni peggiori con una probabilità che diminuisce nel tempo.
    - Usa un "schedule" di raffreddamento, es. decrescita geometrica.
  + **Tabu Search**:
    - Evita ripetizioni registrando soluzioni recenti.
  + **Two-Stage Choice**:
    - Seleziona prima la variabile, poi il valore.

**4.9 Metodi Basati sulla Popolazione**

* **Descrizione**:
  + Mantengono una popolazione di soluzioni invece di una singola.
* **Algoritmi**:
  + **Beam Search**: Conserva i kk migliori cammini.
  + **Genetic Algorithms**:
    - Ispirati alla selezione naturale.
    - Operazioni principali:
      * **Crossover**: Combina due soluzioni per generare una nuova.
      * **Mutazione**: Modifica casuale di alcuni valori.

**4.10 Ottimizzazione**

* **Problema di Ottimizzazione**:
  + Trova l’assegnazione che minimizza o massimizza una funzione obiettivo.
* **Problemi di Ottimizzazione Vincolati**:
  + Include sia vincoli rigidi che flessibili.
  + Obiettivo: Minimizzare il costo delle violazioni.

**4.10.2 Local Search for Optimization**

* **Strategie**:
  + **Gradient Descent**:
    - Muove verso il minimo seguendo la direzione di massima pendenza.
    - Richiede calcolo di derivate parziali.
  + **Simulated Annealing**:
    - Combina casualità e ottimizzazione per evitare minimi locali.

**4.11 Review**

* **Punti chiave**:
  + Gli algoritmi di ricerca e ottimizzazione risolvono CSP combinando rigore e flessibilità.
  + Tecniche avanzate come GAC, VE, e Local Search migliorano significativamente l’efficienza.
  + Algoritmi basati sulla popolazione offrono una prospettiva evolutiva per problemi complessi.

Se hai bisogno di approfondimenti o chiarimenti, fammi sapere! 😊