# Capitolo 2: Semplificazione, Ottimizzazione e Implicazione



## Semplificazione, Ottimizzazione e Implicazione

- Semplificazione di vincoli
- Proiezione
- Semplificatori di vincoli
- Ottimizzazione
- Implicazione ed equivalenza



## Semplificazione di vincoli

- Due vincoli equivalenti rappresentano la stessa informazione, ma …
- Uno può essere più semplice dell'altro

$$
\begin{aligned}
X \geq 1 \wedge X \geq 3 \wedge 2 = Y + X \\
\leftrightarrow X \geq 3 \wedge 2 = Y + X \\
\leftrightarrow 3 \leq X \wedge X = 2 - Y \\
\leftrightarrow X = 2 - Y \wedge 3 \leq X \\
\leftrightarrow X = 2 - Y \wedge 3 \leq 2 - Y \\
\leftrightarrow X = 2 - Y \wedge Y \leq -1
\end{aligned}
$$

Rimuovere vincoli ridondanti, riscrivere un vincolo primitivo, cambiare l'ordine, sostituire usando una equazione, tutti preservano l'equivalenza



## Vincoli ridondanti

- Un vincolo *C1* **implica** un altro vincolo *C2* se le soluzioni di *C1* sono un sottoinsieme di quelle di *C2*
- *C2* è detto **ridondante** rispetto a *C1*
- Scriviamo:

$$
\begin{aligned}
X \geq 3 \rightarrow X \geq 1 \\
Y \leq X + 2 \wedge Y \geq 4 \rightarrow X \geq 1 \\
cons(X, X) = cons(Z, nil) \rightarrow Z = nil
\end{aligned}
$$



## Vincoli ridondanti

- Possiamo rimuovere un vincolo primitivo che è ridondante rispetto al resto dei vincoli

$$
\begin{aligned}
\underline{\underline{X \geq 1}} \wedge X \geq 3 \leftrightarrow X \geq 3 \\
Y \leq X + 2 \wedge \underline{\underline{X \geq 1}} \wedge Y \geq 4 \leftrightarrow Y \leq X + 2 \wedge Y \geq 4 \\
cons(X, X) = cons(Z, nil) \wedge \underline{\underline{Z = nil}} \leftrightarrow cons(X, X) = cons(Z, nil)
\end{aligned}
$$

Così otteniamo un vincolo più semplice



## Risolutori a forma risolta

- Un risolutore a forma risolta crea vincoli equivalenti ➔ può essere visto come un semplificatore

Per esempio usando il risolutore per equazioni di termini

$$
\begin{aligned}
& cons(X, X) = cons(Z, nil) \wedge Y = succ(X) \wedge succ(Z) = Y \wedge Z = nil \\
& \leftrightarrow X = nil \wedge Z = nil \wedge Y = succ(nil)
\end{aligned}
$$

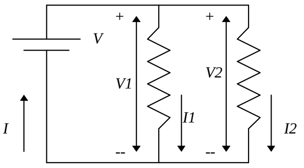
O usando il risolutore di Gauss-Jordan

$$
\begin{aligned}
& X = 2 + Y \wedge 2Y + X - T = Z \wedge X + Y = 4 \wedge Z + T = 5 \\
& \leftrightarrow X = 3 \wedge Y = 1 \wedge Z = 5 - T
\end{aligned}
$$

# Proiezione

Diventa anche più importante semplificare quando siamo solo interessati ad alcune variabili nel vincolo

$$
\begin{aligned}
V1 &= I1 \times R1 \\
V2 &= I2 \times R2 \\
V - V1 &= 0 \\
V - V2 &= 0 \\
V1 - V2 &= 0 \\
I - I1 - I2 &= 0 \\
-I + I1 + I2 &= 0 \\
R1 &= 5 \\
R2 &= 10
\end{aligned}
$$




Semplificato rispetto a $V$ e $I$: $V = \frac{10}{3} I$



# Proiezione

- La **proiezione** di un vincolo *C* sulle variabili *V* è un vincolo *C1* tale che
  - *C1* ha solo le variabili in *V*
  - Ogni soluzione di *C* è una soluzione di *C1*
  - Una soluzione di *C1* può essere estesa per ottenere una soluzione di *C*

| | |
|---|---|
| $X \geq Y \wedge Y \geq Z \wedge Z \geq 0$ | $X \geq 0$ |
| $\{X \mapsto 0, Y \mapsto 0, Z \mapsto 0\}$ | $\{X \mapsto 0\}$ |
| $\{X \mapsto 4, Y \mapsto 3, Z \mapsto 1\}$ | $\{X \mapsto 4\}$ |



# Algoritmo di Fourier

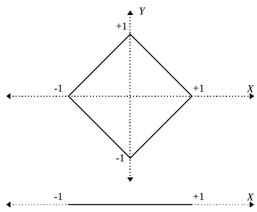
- Elimina una variabile *y* da disequazioni lineari *C*
- Scrive ogni diseq. con *y* su un lato:
  $t_1 \leq y \qquad y \leq t_2$
- Per ogni coppia $t_1 \leq y \quad y \leq t_2$
  - produce una nuova diseq. $t_1 \leq y \otimes y \leq t_2 \Rightarrow t_1 \leq t_2$
- Il risultato è un insieme di nuove diseq. e quelle diseq. in *C* che non riguardano *y*



# Esempio

Proiettiamo fuori *Y:*

$$
\begin{aligned}
&X - 1 \leq Y \\
&-1 - X \leq Y \\
&Y \leq 1 - X \\
&Y \leq 1 + X
\end{aligned}
$$




$$
\begin{aligned}
&X - 1 \leq Y \otimes Y \leq 1 - X \quad X \leq 1 \\
&X - 1 \leq Y \otimes Y \leq 1 + X \quad 0 \leq 2 \\
&-1 - X \leq Y \otimes Y \leq 1 - X \quad 0 \leq 2 \\
&-1 - X \leq Y \otimes Y \leq 1 + X \quad -1 \leq X
\end{aligned}
$$

Il risultato contiene solo *X:*

$$X \leq 1 \wedge -1 \leq X$$



# Proiettare i vincoli sugli alberi

- Possiamo proiettare vincoli sui termini
  $cons(Y,Y) = cons(X,Z) \wedge succ(Z) = succ(T)$
- Proiettato su *{X,Z}* è
- Ma cosa è *X = cons(Y,Z)* proiettato su *X*?
- Risposta: non c'è un tale vincolo!



# Semplificatori di vincoli

- vincoli *C1* and *C2* sono **equivalenti rispetto alle variabili in** *V* se
  - Prendendo una soluzione di uno dei due, e restringendola alle variabili in *V*, questa soluzione ristretta può essere estesa ad una soluzione dell'altro
  - Example *X=succ(Y)* and *X=succ(Z)* wrt *{X}*

| $X = succ(Y)$ | $\{X\}$ | $X = succ(Z)$ |
|---|---|---|
| $\{X \mapsto succ(a), Y \mapsto a\}$ | $\{X \mapsto succ(a)\}$ | $\{X \mapsto succ(a), Z \mapsto a\}$ |
| $\{X \mapsto succ(nil), Y \mapsto nil\}$ | $\{X \mapsto succ(nil)\}$ | $\{X \mapsto succ(nil), Z \mapsto nil\}$ |

## Definizione di semplificatore

- Un **semplificatore di vincoli** è una funzione *simpl* che prende un vincolo *C* e un insieme di variabili *V* e ritorna un vincolo *C1* che è equivalente a *C* rispetto a *V*
- Possiamo creare un semplificatore per disequazioni lineari usando l'algoritmo di Fourier



## Semplificatore per vincoli sugli alberi

- Applicare il risolutore per equazioni di termini a *C* e ottenere *C1*
- se *C1* è *false* allora return *false*
- Per ogni equazione *x=t* in *C1*
  - se *x* è in *V* allora
    - se *t* è una variabile non in *V*
      - sostituire *x* per *t* in *C1* e nel risultato
    - altrimenti aggiungere *x=t* al risultato
- return risultato



## Esempio di semplificazione sugli alberi

Vincolo sugli alberi da semplificare rispetto a *{Y,T}*:

$$h(f(X,Y),Z,g(T)) = h(f(g(T),X), f(X,X), g(U))$$

Vincolo equivalente dal risolutore sugli alberi:

$$Z = f(g(U),g(U)) \wedge X = g(U) \wedge Y = g(U) \wedge T = U$$

Eliminare le prime due equazioni, tenere la terza e usare l'ultima per sostituire T con U:

$$Y = g(T)$$



## Proprieta' dei semplificatori

- Proprietà desiderabili di un semplificatore:
  - **proiettante**: $vars(simpl(C,V)) \subseteq V$
  - **Debolmente proiettante:** per tutti I vincoli *C2* che sono equivalenti a *C1* rispetto a *V*
    $$|vars(simpl(C1,V)) - V| \leq |vars(C2) - V|$$
    - Un risolutore debolmente proiettante non usa mai più variabili di quelle necessarie
  - Entrambe le proprietà permettono ad un semplificatore di essere usato come un risolutore



## Ottimizzazione

- Spesso dato un problema modellato con vincoli, non vogliamo una qualsiasi soluzione, ma una soluzione ottima
- Quindi si ha un **problema di ottimizzazione**
- Abbiamo bisogno di una **funzione obbiettivo** in modo da poter paragonare soluzioni, cioè un mapping da soluzioni a valori reali



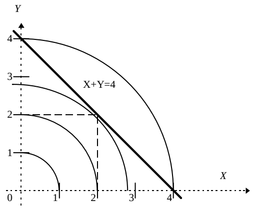
## Problema di ottimizzazione

- Un **probleema di ottimizzazione** *(C,f)* consiste di un vincolo *C* e una funzione obbiettivo *f*
- Una valutazione *v1* è **preferita** alla valutazione *v2* se *f(v1) < f(v2)*
- Una **soluzione ottima è** una soluzione di *C* tale che non esiste nessun'altra soluzione di *C* che è preferita a lei.

## *Esempio di ottimizzazione*

Un problema di ottimizzazione:
$(C \equiv X + Y \geq 4, \quad f \equiv X^2 + Y^2)$

Trovare il punto più vicino all'origine che soddisfi *C*. Alune soluzioni e il valore di $f$:




| | |
|---|---|
| $\{X \mapsto 0, Y \mapsto 4\}$ | 16 |
| $\{X \mapsto 3, Y \mapsto 3\}$ | 18 |
| $\{X \mapsto 2, Y \mapsto 2\}$ | 8 |

Soluzione ottima:
$\{X \mapsto 2, Y \mapsto 2\}$

19

## *Ottimizzazione*

- Alcuni problemi di ottimizzazione non hanno soluzioni
  - Il vincolo non ha soluzioni:
    $(X \geq 2 \wedge X \leq 0, \quad X^2)$
  - Il problema non ha ottimo:
    $(X \leq 0, \quad X)$
    - Per ogni soluzione, c'è n'è sempre un'altra migliore

20

## *Algoritmo del simplesso*

- L'algoritmo più usato per l'ottimizzazione
- Ottimizza un funzione lineare rispetto a dei vincoli lineari
- Collegato all'eliminazione di Gauss-Jordan

21

## *Algoritmo del simplesso*

- Un problema di ottimizzazione (*C, f*) è in **forma simplesso** se:
  - *C* è la congiunzione di *CE* e *CI*
  - *CE* è una congiunzione di equazioni lineari
  - *CI* vincola tutte le variabili in *C* ad essere non negative
  - *f* è una espressione lineare sulle variabili in *C*

22

## *Esempio del simplesso*

Un problema di ottimizzazione in forma simplesso

minimize $3X+2Y-Z+1$ subject to

$$\begin{array}{lllll} X & +Y & & & = 3 \wedge \\ -X & -3Y & +2Z & +T & = 1 \wedge \end{array}$$
$$X \geq 0 \wedge Y \geq 0 \wedge Z \geq 0 \wedge T \geq 0$$

- Un problema arbitrario può essere messo in forma simplesso:
  - rimpiazzando ogni variabile non vincolata *X* con nuove variabili $X^+ - X^-$
  - Rimpiazzando ogni disequazione $e \leq r$ con una nuova variabile *s* e $e + s = r$

23

## *Implicazione ed Equivalenza*

- Altre importanti operazioni sui vincoli:
- **implicazione**: controlla se *C1* implica *C2*
  - *impl(C1, C2)* risponde *true, false* o *unknown*
- **equivalenza**: controlla se *C1* e *C2* sono equivalenti
  - *equiv(C1, C2)* risponde *true, false* o *unknown*

33

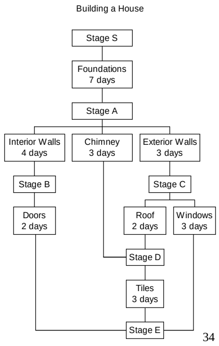
## Esempio di implicazione

Per I vincoli della casa *CH*, lo stadio B sarà raggiunto dopo lo stadio C?

$$CH \rightarrow T_B \geq T_C$$

Per questa domanda la risposta è *false*, ma se richiediamo che la casa sia finita in 15 giorni la risposta è *true*

$$CH \wedge T_E = 15 \rightarrow T_B \geq T_C$$






## Implicazione ed Equivalenza

- Possiamo usare *impl* per definire *equiv* e viceversa:

  $impl(C1, C2) = equiv(C1, C1 \wedge C2)$

  $equiv(C1, C2) = impl(C1, C2) \wedge impl(C2, C1)$
- Possiamo usare un risolutore per testare *impl:* $impl(C1, C2) = \neg solv(C1 \wedge \neg C2)$
- Es.: $impl(CH, T_B \geq T_C) = \neg solv(CH \wedge T_B < T_C)$



## Sommario su semplificazione, ottimizzazione e implicazione

- Vincoli equivalenti possono essere scritti in varie forme, quindi serve la semplificazione
- Soprattutto se siamo solo interessati all'interazione tra alcune delle variabili
- Molti problemi necessitano di soluzioni ottime, ci sono algoritmi (es.: simplesso) per trovarle
- Posssiamo anche voler fare domande che riguardano l'implicazione