Problema del Bin Packing

Michele Monaci

Dipartimento di Ingegneria dell'Informazione, Università di Padova
Viale Gradenigo, 6/A - 35131 - Padova
monaci@dei.unipd.it

1 Introduzione

Dato un insieme $N = \{1, \ldots, n\}$ di oggetti, il k-esimo con peso $w_k > 0$ ed un insieme $M = \{1, \ldots, m\}$ di contenitori (*bin*), ciascuno con capacità W, il problema del bin packing (1BP) richiede di inserire tutti gli oggetti nei contenitori in modo che:

- ciascun oggetto venga inserito in un contenitore;
- il peso degli oggetti inseriti in ciascun contenitore non superi W;
- il numero di contenitori utilizzati sia minimizzato.

Una condizione necessaria effinchè possano esistere delle soluzioni ammissibili è che sia $w_k \leq W$ $(k \in N)$. Sotto questa condizione, è sempre possibile trovare una soluzione ammissibile nel caso in cui sia $m \geq n$ (al limite inserendo un oggetto in ciascun contenitore); viceversa, nel caso in cui sia $m < n$ il problema di verificare se esiste una soluzione ammissibile è NP-difficile. Noi considereremo sempre il caso in cui $m \geq n$.

2 Modello Matematico di tipo Descrittivo

Introducendo le seguenti variabili binarie

$$x_{ik} = \begin{cases} 1 & \text{se l'oggetto } k \text{ viene inserito nel contenitore } i \\ 0 & \text{altrimenti} \end{cases} \qquad (i \in M;\ k \in N)$$

$$y_i = \begin{cases} 1 & \text{se il contenitore } i \text{ è utilizzato} \\ 0 & \text{altrimenti} \end{cases} \qquad (i \in M)$$

si ottiene un possibile modello di PLI:

$$\min \sum_{i \in M} y_i \tag{1}$$

$$\sum_{k \in N} w_k\, x_{ik} \leq W\, y_i \qquad i \in M \tag{2}$$

$$\sum_{i \in M} x_{ik} = 1 \qquad k \in N \tag{3}$$

$$x_{ik} \in \{0, 1\} \qquad i \in M,\ k \in N \tag{4}$$

$$y_i \in \{0, 1\} \qquad i \in M \tag{5}$$

Il modello presenta un numero polinomiale di variabili ($n \cdot m + m$) e di vincoli ($m + n$).

Il vincolo (2) associato al contenitore i ($i \in M$) ha un duplice scopo: da un lato impone che non si ecceda la capacità del contenitore, dall'altro impone le seguenti relazioni logiche

(i) $y_i = 0 \quad \Rightarrow \quad x_{ik} = 0 \quad \forall k \in N$

(ii) $x_{ik} = 1$ per qualche $k \in N \quad \Rightarrow \quad y_i = 1$

È da notare che la relazione logica

(iii) $x_{ik} = 0 \quad \forall k \in N \quad \Rightarrow \quad y_i = 0$

non è imposta esplicitamente nel modello matematico, ma solo implicitamente tramite la funzione obiettivo.

Si noti che i vincoli (3) possono essere scritti anche nella forma

$$\sum_{i \in M} x_{ik} \geq 1 \qquad k \in N \tag{6}$$

richiedendo cioè che ciascun oggetto sia inserito in *almeno* un bin. Infatti, data una qualunque soluzione che soddisfa i vincoli (6) ma non i vincoli (3), nella quale cioè un qualche oggetto $k \in N$ è inserito in più di un bin, è possibile ottenere una soluzione ammissibile per il modello (1)–(5) rimuovendo l'oggetto k da tutti i bin tranne uno (a scelta). Per costruzione questa nuova soluzione soddisfa i vincoli (3) ed il suo costo non è superiore al costo della soluzione di partenza.

Un modello alternativo è ottenibile rimpiazzando i vincoli (2) con la seguente coppia di vincoli

$$\sum_{k \in N} w_k \; x_{ik} \leq W \qquad i \in M \tag{7}$$

$$x_{ik} \leq y_i \qquad i \in M, \; k \in N \tag{8}$$

3 Modello Matematico di tipo Set Partitioning

Indichiamo con $\mathcal{T}$ la collezione dei sottoinsiemi di oggetti che possono essere impaccati in un contenitore senza eccedere la capacità W. Formalmente:

$$\mathcal{T} := \left\{ S \subseteq N : \sum_{k \in S} w_k \leq W \right\} \tag{9}$$

Introducendo una variabile binaria per ciascun possibile riempimento di un contenitore

$$x_S = \begin{cases} 1 & \text{se il riempimento } S \text{ è assegnato ad un contenitore} \\ 0 & \text{altrimenti} \end{cases} \qquad (S \in \mathcal{T})$$

si ottiene il seguente modello matematico:

$$\min \sum_{S \in \mathcal{T}} x_S \tag{10}$$

$$\sum_{S \in \mathcal{T}: k \in S} x_S = 1 \qquad k \in N \tag{11}$$

$$x_S \in \{0,1\} \qquad S \in \mathcal{T} \tag{12}$$

Questo modello contiene un numero polinomiale di vincoli (n), ma un numero esponenziale di variabili ($O(2^n)$); il numero di variabili rischia di essere estremamente grande per problemi di dimensioni reali.

• **Esempio**
Consideriamo la seguente istanza:

$$n = 6,\ w_1 = 49,\ w_2 = 41,\ w_3 = 30,\ w_4 = 29,\ w_5 = 26,\ w_6 = 25,\ W = 100.$$

L'insieme dei riempimenti ammissibili risulta essere:

$$\mathcal{T} = \{s_1, \ldots, s_{31}\}$$

con

$$\begin{array}{llll}
s_1 = \{1\}, & s_2 = \{2\}, & s_3 = \{3\}, & s_4 = \{4\}, \\
s_5 = \{5\}, & s_6 = \{6\}, & s_7 = \{1,2\}, & s_8 = \{1,3\}, \\
s_9 = \{1,4\}, & s_{10} = \{1,5\}, & s_{11} = \{1,6\}, & s_{12} = \{2,3\}, \\
s_{13} = \{2,4\}, & s_{14} = \{2,5\}, & s_{15} = \{2,6\}, & s_{16} = \{3,4\}, \\
s_{17} = \{3,5\}, & s_{18} = \{3,6\}, & s_{19} = \{4,5\}, & s_{20} = \{4,6\}, \\
s_{21} = \{5,6\}, & s_{22} = \{1,5,6\}, & s_{23} = \{2,3,4\}, & s_{24} = \{2,3,5\}, \\
s_{25} = \{2,3,6\}, & s_{26} = \{2,4,5\}, & s_{27} = \{2,4,6\}, & s_{28} = \{2,5,6\}, \\
s_{29} = \{3,4,5\}, & s_{30} = \{3,4,6\}, & s_{31} = \{3,5,6\}, & s_{32} = \{4,5,6\}
\end{array}$$

ed il modello associato è

$$\min \sum_{S=1}^{32} x_S$$

$$\begin{array}{ccccccccccccccccccccccccc}
x_1 & + & x_7 & + & x_8 & + & x_9 & + & x_{10} & + & x_{11} & + & x_{22} & & & & & & & & & & & & = 1 \\
x_2 & + & x_7 & + & x_{12} & + & x_{13} & + & x_{14} & + & x_{15} & + & x_{23} & + & x_{24} & + & x_{25} & + & x_{26} & + & x_{27} & + & x_{28} & & = 1 \\
x_3 & + & x_8 & + & x_{12} & + & x_{16} & + & x_{17} & + & x_{18} & + & x_{23} & + & x_{24} & + & x_{25} & + & x_{29} & + & x_{30} & + & x_{31} & & = 1 \\
x_4 & + & x_9 & + & x_{13} & + & x_{16} & + & x_{19} & + & x_{20} & + & x_{23} & + & x_{26} & + & x_{27} & + & x_{29} & + & x_{30} & + & x_{32} & & = 1 \\
x_5 & + & x_{10} & + & x_{14} & + & x_{17} & + & x_{19} & + & x_{21} & + & x_{22} & + & x_{24} & + & x_{26} & + & x_{28} & + & x_{29} & + & x_{31} & + & x_{32} & = 1 \\
x_6 & + & x_{11} & + & x_{15} & + & x_{18} & + & x_{20} & + & x_{21} & + & x_{22} & + & x_{25} & + & x_{27} & + & x_{28} & + & x_{30} & + & x_{31} & + & x_{32} & = 1
\end{array}$$

$$x_S \in \{0,1\} \qquad S = 1, \ldots, 32$$

4 Modello Matematico di tipo Set Covering

La formulazione Set Covering è analoga al modello (10)–(12) ma richiede un numero minore (seppur ancora esponenziale) di variabili. Indichiamo con $\mathcal{S}$ la collezione dei sottoinsiemi *massimali* di oggetti che possono essere impaccati in un contenitore senza eccedere la capacità W. Un insieme di oggetti è detto massimale se non è possibile aggiungere nessun altro oggetto senza eccedere la capacità del contenitore. Formalmente si ha:

$$\mathcal{S} := \left\{ S \subseteq N : \sum_{k \in S} w_k \leq W,\ \sum_{k \in S} w_k + w_l > W \quad \forall l \notin S \right\}$$

Introducendo una variabile decisionale per ciascun riempimento ammissibile e massimale

$$x_S = \begin{cases} 1 & \text{se il riempimento } S \text{ è assegnato ad un contenitore} \\ 0 & \text{altrimenti} \end{cases} \qquad (S \in \mathcal{S})$$

si ottiene il seguente modello matematico:

$$\min \sum_{S \in \mathcal{S}} x_S \tag{13}$$

$$\sum_{S \in \mathcal{S}: k \in S} x_S \geq 1 \qquad k \in N \tag{14}$$

$$x_S \in \{0,1\} \qquad S \in \mathcal{S} \tag{15}$$

Ovviamente ogni soluzione ammissibile del modello (10)–(12) è ammissibile anche per il modello (13)–(15), salvo eventualmente rendere massimale gli insiemi $S \in \mathcal{T}$ che non lo sono e che hanno $x_S > 0$. È facile poi verificare che ogni soluzione ammissibile del modello (13)–(15) può essere trasformata in una soluzione ammissibile (di valore non superiore) del modello (10)–(12). In realtà si ha l'equivalenza completa anche per le soluzioni ottime dei rilassamenti continui dei due modelli.

Il modello (13)–(15) è quindi preferibile rispetto al modello (10)–(12) perchè:

- il suo numero di variabili è più piccolo;
- i valori dei rilassamenti continui coincidono;
- generalmente sono più facili da risolvere LP con disuguaglianze piuttosto che con equazioni.

• Esempio

Si consideri nuovamente l'istanza

$$n = 6,\ w_1 = 49,\ w_2 = 41,\ w_3 = 30,\ w_4 = 29,\ w_5 = 26,\ w_6 = 25,\ W = 100.$$

L'insieme dei riempimenti massimali ammissibili risulta essere:

$$\mathcal{S} = \{s_7, s_8, s_9, s_{22}, s_{23}, s_{24}, s_{25}, s_{26}, s_{27}, s_{28}, s_{29}, s_{30}, s_{31}, s_{32}\}$$

con

$$\begin{array}{lllll} s_7 = \{1,2\}, & s_8 = \{1,3\}, & s_9 = \{1,4\}, & s_{22} = \{1,5,6\}, & s_{23} = \{2,3,4\}, \\ s_{24} = \{2,3,5\}, & s_{25} = \{2,3,6\}, & s_{26} = \{2,4,5\}, & s_{27} = \{2,4,6\}, & s_{28} = \{2,5,6\}, \\ s_{29} = \{3,4,5\}, & s_{30} = \{3,4,6\}, & s_{31} = \{3,5,6\}, & s_{32} = \{4,5,6\} & \end{array}$$

ed il modello associato è

$$\begin{array}{lccccccccccccccccc} \min & x_7 & +x_8 & +x_9 & +x_{22} & +x_{23} & +x_{24} & +x_{25} & +x_{26} & +x_{27} & +x_{28} & +x_{29} & +x_{30} & +x_{31} & +x_{32} & & \\ & x_7 & +x_8 & +x_9 & +x_{22} & & & & & & & & & & & \geq & 1 \\ & x_7 & & & & +x_{23} & +x_{24} & +x_{25} & +x_{26} & +x_{27} & +x_{28} & & & & & \geq & 1 \\ & & x_8 & & & +x_{23} & +x_{24} & +x_{25} & & & & +x_{29} & +x_{30} & +x_{31} & & \geq & 1 \\ & & & x_9 & & +x_{23} & & & +x_{26} & +x_{27} & & +x_{29} & +x_{30} & & +x_{32} & \geq & 1 \\ & & & & x_{22} & & +x_{24} & & +x_{26} & & +x_{28} & +x_{29} & & +x_{31} & +x_{32} & \geq & 1 \\ & & & & x_{22} & & & +x_{25} & & +x_{27} & +x_{28} & & +x_{30} & +x_{31} & +x_{32} & \geq & 1 \\ & x_7, & x_8, & x_9, & x_{22}, & x_{23}, & x_{24}, & x_{25}, & x_{26}, & x_{27}, & x_{28}, & x_{29}, & x_{30}, & x_{31}, & x_{32} & \in & \{0,1\} \end{array}$$

4.1 Soluzione del rilassamento continuo del modello

Il rilassamento continuo del modello (13)–(15) è il seguente

$$z_P = \min \sum_{S \in \mathcal{S}} x_S \tag{16}$$

$$\sum_{S \in \mathcal{S}: k \in S} x_S \geq 1 \qquad k \in N \tag{17}$$

$$x_S \geq 0 \qquad S \in \mathcal{S} \tag{18}$$

Si noti che i vincoli $x_S \leq 1$ ($S \in \mathcal{S}$) sono stati omessi in quanto ridondanti.

Il duale di questo modello è dato da

$$z_D = \max \sum_{k \in N} \pi_k \tag{19}$$

$$\sum_{k \in S} \pi_k \leq 1 \qquad S \in \mathcal{S} \tag{20}$$

$$\pi_k \geq 0 \qquad k \in N \tag{21}$$

Dato che il problema (16)–(18) ha una soluzione ottima finita, il teorema di dualità forte ci assicura che $z_P = z_D$. Pertanto risolvendo il problema (19)–(21) si ottiene un lower bound sul valore della soluzione ottima del problema di bin packing. Il modello (19)–(21) è un problema di programmazione lineare nel quale il numero di vincoli (20) è esponenziale, come era per il rilassamento continuo del modello classico per l'ATSP. Quindi possiamo utilizzare anche in questo caso una procedura di tipo cutting-plane che (i) risolve un rilassamento del problema, contenente solo un sottoinsieme dei vincoli (20), (ii) individua un insieme di vincoli violati (se esistono) ed eventualmente (iii) li aggiunge alla formulazione corrente. Ad ogni iterazione quindi si risolve il rilassamento del problema duale definito considerando solo un sottoinsieme $\mathcal{S}' \subset \mathcal{S}$ di vincoli (20), ossia si risolve un *problema primale ristretto* nel quale si considerano solo le variabili x_S associate ai riempimenti di $\mathcal{S}' \subset \mathcal{S}$. Il procedimento termina quando non esistono più vincoli violati, cioè quando tutti i vincoli del duale sono soddisfatti (pur non essendo esplicitamente presenti nella formulazione). Questo significa anche che non esistono variabili del problema primale con costo ridotto negativo, ossia che il primale ristretto, pur non contenendo tutte le variabili del problemna (ma solo un sottoinsieme) fornisce la soluzione ottima del rilassamento continuo.

Il problema di separazione di vincoli per il duale è formulato nel seguente modo: data una soluzione $\tilde{\pi}$ ottima per il rilassamento corrente del duale, determinare, se esiste, un riempimento $S^* \in \mathcal{S}$ il cui corrispondente vincolo duale è violato da $\tilde{\pi}$. Quindi il problema di separazione consiste nel trovare un insieme S^* di oggetti tale che

(a) $\sum_{k \in S^*} w_k \leq W$

(b) $\sum_{k \in S^*} w_k + w_l > W \quad \forall l \notin S$

(c) $\sum_{k \in S^*} \tilde{\pi}_k > 1$

Le condizioni (a) e (b) impongono che il riempimento S^* sia ammissibile e massimale, rispettivamente, ossia che sia $S^* \in \mathcal{S}$. La condizione (c) richiede che il corrispondente vincolo (20) sia violato dalla soluzione duale $\tilde{\pi}$. Nella ricerca di vincoli duali violati possiamo trascurare la condizione (b); un insieme S^* per il quale valgono la (a) e la (c) può poi essere reso massimale a posteriori (aggiungendo in modo greedy degli oggetti) senza modificare la violazione del vincolo (20) associato (dato che $\tilde{\pi}_k \geq 0 \ \forall k$). Il problema di determinare vincolo (20) maggiormente violato può essere formulato nel seguente modo:

$$v = \max \sum_{k \in N} \tilde{\pi}_k \ \psi_k \tag{22}$$

$$\sum_{k \in N} w_k \ \psi_k \leq W \tag{23}$$

$$\psi_k \in \{0, 1\} \qquad k \in N \tag{24}$$

Quindi ad ogni iterazione bisogna risolvere un KP01 definito con capacità pari a W e con n oggetti, il k-esimo dei quali ha peso w_k e profitto $\tilde{\pi}_k$. (Nota: i profitti degli oggetti sono numeri reali e non interi - ma non negativi).

Il vincolo (23) corrisponde alla condizione (a), mentre la funzione obiettivo (22) massimizza il termine sinistro del vincolo (20) cercato, ossia individua il vincolo cui corrisponde la massima violazione v (se esiste). A ciascun oggetto $k \in N$ è associate una variabile ψ_k che vale 1 se e solo se $k \in S^*$. Se $v > 1$ allora il riempimento $S^* = \{k \in N : \psi_k = 1\}$ corrisponde al vincolo (23) maggiormente violato. Viceversa, se $v \leq 1$ allora tutti i vincoli (23), seppur non presenti esplicitamente nel modello, sono soddisfatti e le soluzioni primale e duale correnti sono entrambe ottime.

• Esempio

Consideriamo nuovamente l'istanza

$$n = 6,\ w_1 = 49,\ w_2 = 41,\ w_3 = 30,\ w_4 = 29,\ w_5 = 26,\ w_6 = 25,\ W = 100.$$

Iterazione 1

Partiamo dalla soluzione ammissibile definita dai seguenti riempimenti ammissibili:

$$S_1 = \{1, 2\},\ S_2 = \{3, 4, 5\},\ S_3 = \{1, 5, 6\}.$$

Il modello di partenza corrispondente alla sottofamiglia $\mathcal{S}' = \{S_1, S_2, S_3\}$ è:

$$\begin{array}{rrcccccc}
 & \tilde{z} = \min & x_1 & + & x_2 & + & x_3 & \\
k = 1: & & x_1 & & & + & x_3 & \geq 1 \\
k = 2: & & x_1 & & & & & \geq 1 \\
k = 3: & & & & x_2 & & & \geq 1 \\
k = 4: & & & & x_2 & & & \geq 1 \\
k = 5: & & & & x_2 & + & x_3 & \geq 1 \\
k = 6: & & & & & & x_3 & \geq 1 \\
 & & x_1 & , & x_2 & , & x_3 & \geq 0
\end{array}$$

per il quale la soluzione ottima è $\tilde{x}_1 = \tilde{x}_2 = \tilde{x}_3 = 1$ e vale $\tilde{z} = 3$.

Il duale corrispondente è

$$\begin{array}{rrccccccccccc} & \tilde{w} = \max & y_1 & + & y_2 & + & y_3 & + & y_4 & + & y_5 & + & y_6 & \\ S=1: & & y_1 & + & y_2 & & & & & & & & \leq & 1 \\ S=2: & & & & & & y_3 & + & y_4 & + & y_5 & & \leq & 1 \\ S=3: & & y_1 & & & & & & & + & y_5 & + & y_6 & \leq & 1 \\ & & y_1 & , & y_2 & , & y_3 & , & y_4 & , & y_5 & , & y_6 & \geq & 0 \end{array}$$

per il quale la soluzione ottima è $\tilde{y}_2 = \tilde{y}_4 = \tilde{y}_6 = 1$.

Il problema di generazione è dunque il seguente: esiste un riempimento $S^* \in \mathcal{S} \setminus \mathcal{S}'$ tale che $\sum_{k \in S^*} y_k > 1$? La risposta a questo problema è affermativa; ad esempio il riempimento $S_4 = \{2, 4, 6\}$ è tale che $\sum_{k \in S_4} w_k = 41 + 29 + 25 \leq W$ e $\sum_{k \in S_4} y_k = 1 + 1 + 1 > 1$.

Iterazione 2

Aggiungendo la colonna x_4 si ottiene $\mathcal{S}' = \{S_1, S_2, S_3, S_4\}$ ed il primale diventa

$$\begin{array}{rrccccccccc} & \tilde{z} = \min & x_1 & + & x_2 & + & x_3 & + & x_4 & & \\ k=1: & & x_1 & & & + & x_3 & & & \geq & 1 \\ k=2: & & x_1 & & & & & + & x_4 & \geq & 1 \\ k=3: & & & & x_2 & & & & & \geq & 1 \\ k=4: & & & & x_2 & & & + & x_4 & \geq & 1 \\ k=5: & & & & x_2 & + & x_3 & & & \geq & 1 \\ k=6: & & & & & & x_3 & + & x_4 & \geq & 1 \\ & & x_1 & , & x_2 & , & x_3 & , & x_4 & \geq & 0 \end{array}$$

per il quale la soluzione ottima è $\tilde{x}_1 = \tilde{x}_3 = \tilde{x}_4 = 1/2$, $\tilde{x}_2 = 1$, $\tilde{z} = 5/2$.

Il duale corrispondente è

$$\begin{array}{rrccccccccccc} & \tilde{w} = \max & y_1 & + & y_2 & + & y_3 & + & y_4 & + & y_5 & + & y_6 & \\ S=1: & & y_1 & + & y_2 & & & & & & & & \leq & 1 \\ S=2: & & & & & & y_3 & + & y_4 & + & y_5 & & \leq & 1 \\ S=3: & & y_1 & & & & & & & + & y_5 & + & y_6 & \leq & 1 \\ S=4: & & & & y_2 & & & + & y_4 & & & + & y_6 & \leq & 1 \\ & & y_1 & , & y_2 & , & y_3 & , & y_4 & , & y_5 & , & y_6 & \geq & 0 \end{array}$$

per il quale la soluzione ottima è $\tilde{y}_1 = \tilde{y}_2 = \tilde{y}_6 = 1/2$, $\tilde{y}_3 = 1$.

La procedura di generazione fornisce il riempimento $S_5 = \{2, 3, 6\}$, per il quale si ha $\sum_{k \in S_5} y_k = 1/2 + 1 + 1/2 > 1$.

Iterazione 3

Il primale corrente diventa

$$\begin{array}{rrccccccccccc} & \tilde{z} = \min & x_1 & + & x_2 & + & x_3 & + & x_4 & + & x_5 & & \\ k=1: & & x_1 & & & + & x_3 & & & & & \geq & 1 \\ k=2: & & x_1 & & & & & + & x_4 & + & x_5 & \geq & 1 \\ k=3: & & & & x_2 & & & & & + & x_5 & \geq & 1 \\ k=4: & & & & x_2 & & & + & x_4 & & & \geq & 1 \\ k=5: & & & & x_2 & + & x_3 & & & & & \geq & 1 \\ k=6: & & & & & & x_3 & + & x_4 & + & x_5 & \geq & 1 \\ & & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \geq & 0 \end{array}$$

con soluzione ottima $\tilde{x}_1 = \tilde{x}_3 = 1/2$, $\tilde{x}_2 = 3/4$, $\tilde{x}_4 = \tilde{x}_5 = 1/4$, $\tilde{z} = 9/4$.

Il duale corrispondente è

$$\begin{array}{rccccccccccccc}
\tilde{w} = \max & y_1 & + & y_2 & + & y_3 & + & y_4 & + & y_5 & + & y_6 & & \\
S=1: & y_1 & + & y_2 & & & & & & & & & \leq & 1 \\
S=2: & & & & & y_3 & + & y_4 & + & y_5 & & & \leq & 1 \\
S=3: & y_1 & & & & & & & + & y_5 & + & y_6 & \leq & 1 \\
S=4: & & & y_2 & & & + & y_4 & & & + & y_6 & \leq & 1 \\
S=5: & & & y_2 & + & y_3 & & & & & + & y_6 & \leq & 1 \\
 & y_1 & , & y_2 & , & y_3 & , & y_4 & , & y_5 & , & y_6 & \geq & 0
\end{array}$$

con soluzione ottima $\tilde{y}_1 = 3/4$, $\tilde{y}_2 = \tilde{y}_6 = 1/4$, $\tilde{y}_3 = \tilde{y}_4 = 1/2$.

La procedura di generazione fornisce il riempimento $S_6 = \{2, 3, 4\}$ per il quale $\sum_{k \in S_5} y_k = 1/4 + 1/2 + 1/2 > 1$.

Iterazione 4

Il primale corrente è

$$\begin{array}{rccccccccccccc}
\tilde{z} = \min & x_1 & + & x_2 & + & x_3 & + & x_4 & + & x_5 & + & x_6 & & \\
k=1: & x_1 & & & + & x_3 & & & & & & & \geq & 1 \\
k=2: & x_1 & & & & & + & x_4 & + & x_5 & + & x_6 & \geq & 1 \\
k=3: & & & x_2 & & & & & + & x_5 & + & x_6 & \geq & 1 \\
k=4: & & & x_2 & & & + & x_4 & & & + & x_6 & \geq & 1 \\
k=5: & & & x_2 & + & x_3 & & & & & & & \geq & 1 \\
k=6: & & & & & x_3 & + & x_4 & + & x_5 & & & \geq & 1 \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & , & x_6 & \geq & 0
\end{array}$$

con soluzione ottima (intera) $\tilde{x}_3 = \tilde{x}_6 = 1$, $\tilde{z} = 2$.

Il duale corrispondente è

$$\begin{array}{rccccccccccccc}
\tilde{w} = \max & y_1 & + & y_2 & + & y_3 & + & y_4 & + & y_5 & + & y_6 & & \\
S=1: & y_1 & + & y_2 & & & & & & & & & \leq & 1 \\
S=2: & & & & & y_3 & + & y_4 & + & y_5 & & & \leq & 1 \\
S=3: & y_1 & & & & & & & + & y_5 & + & y_6 & \leq & 1 \\
S=4: & & & y_2 & & & + & y_4 & & & + & y_6 & \leq & 1 \\
S=5: & & & y_2 & + & y_3 & & & & & + & y_6 & \leq & 1 \\
S=6: & & & y_2 & + & y_3 & + & y_4 & & & & & \leq & 1 \\
 & y_1 & , & y_2 & , & y_3 & , & y_4 & , & y_5 & , & y_6 & \geq & 0
\end{array}$$

con soluzione ottima $\tilde{y}_1 = 2/3$, $\tilde{y}_2 = \tilde{y}_3 = \tilde{y}_4 = \tilde{y}_6 = 1/3$. Con questi valori delle variabili duali non esiste nessun riempimento S^* tale che $\sum_{k \in S^*} w_k \leq W$ e $\sum_{k \in S^*} y_k > 1$. Questo è verificabile risolvendo il seguente problema di knapsack

$$\begin{array}{rccccccccccccc}
v = \max & \frac{2}{3}\psi_1 & + & \frac{1}{3}\psi_2 & + & \frac{1}{3}\psi_3 & + & \frac{1}{3}\psi_4 & & & + & \frac{1}{3}\psi_6 & & \\
 & 49\psi_1 & + & 41\psi_2 & + & 30\psi_3 & + & 29\psi_4 & + & 26\psi_5 & + & 25\psi_6 & \leq & 100 \\
 & \psi_1 & , & \psi_2 & , & \psi_3 & , & \psi_4 & , & \psi_5 & , & \psi_6 & \in & \{0,1\}
\end{array}$$

e verificando che $v = 1$ (ed esempio la soluzione $\psi_2 = \psi_3 = \psi_4 = 1$ è ottima e vale 1).

Dal punto di vista del duale, il fatto che tutti i vincoli (20) siano soddisfatti significa che la soluzione $\tilde{y}$ corrente è ammissibile (e ottima). Dal punto di vista del primale questo significa invece che tutte le variabili che non compaiono esplicitamente nel modello (cioè quelle di $\mathcal{S} \setminus \mathcal{S}'$) hanno costo ridotto ≥ 0, e che quindi non serve considerare queste variabili per ottenere la soluzione ottima del rilassamento continuo.

È da notare che solo all'ultima iterazione il valore $\tilde{z}$ è un lower bound valido sul valore della soluzione ottima (intera). Quindi se il procedimento viene interrotto prima della fine (ad esempio per time limit), il tempo impiegato è sprecato dal punto di vista dell'ottenimento di un lower bound (può darsi che invece questo abbia portato a soluzioni ammissibili).
Però, ad ogni iterazione, è possibile ricavare una soluzione $\tilde{u}$ ammissibile per il duale a partire dalla soluzione $\tilde{y}$ corrente (non ammissibile) per il duale e dalla soluzione ottima v del problema di separazione per il duale. La soluzione $\tilde{u}$ definita come $\tilde{u}_i = \tilde{y}_i / v$ ($i = 1, \ldots, m$) è ammissibile per il duale (per costruzione) e fornisce quindi un lower bound sul valore della soluzione ottima del problema primale (pertanto questo è anche un lower bound sul valore della soluzione ottima del problema di bin packing).

Affinchè il procedimento funzioni e converga, non occorre risolvere esattamente KP01 ad ogni iterazione, ma basta aggiungere una qualsiasi colonna a costo ridotto negativo, anche ottenuta tramite euristici per il problema knapsack. L'importante è che all'ultima iterazione il problema knapsack sia risolto in modo esatto per avere un lower bound valido sul valore della soluzione ottima intera.

4.2 Formulazione compatta per 1BP

Esiste una formulazione compatta per il rilassamento continuo (16)–(18) della formulazione set-covering del problema del bin packing. Questo deriva dal fatto che il problema di separazione del duale è un problema knapsack, che può essere formulato come modello di programmazione lineare nel quale la soluzione corrente compare solo nei coefficienti della funzione obiettivo.
Per ottenere questa formulazione bisogna partire dal modello (lineare) del knapsack definito su grafo, scriverne il duale ed inserirlo nella formulazione (19)–(21). Facendo il duale di questo duale, si ottiene la seguente formulazione, che è equivalente a (16)–(18) ma possiede un numero polinomiale di variabili e vincoli:

$$\min \sum_{a \in \delta^+(s)} z_a \tag{25}$$

$$\sum_{a \in \delta^+(i)} z_a - \sum_{a \in \delta^-(i)} z_a = 0 \qquad i \in V \setminus \{s, t\} \tag{26}$$

$$\sum_{a \in A_j} z_a \geq 1 \qquad j \in N \tag{27}$$

$$z_a \geq 0 \qquad a \in A \tag{28}$$

Questa formulazione è basata sullo stesso grafo $G = (V, A)$ usato per definire il problema knapsack. Per ogni oggetto $j \in N$ l'insieme A_j rappresenta l'insieme di archi in A associati alla scelta di inserire l'oggetto j (archi obliqui). In questa formulazione quindi, si vuole

determinare un insieme di cammini da s a t in modo che (i) ciascun oggetto sia presente in almeno un cammino, e (ii) il numero dei cammini utilizzato sia minimo.
In alternativa, il problema può esser visto come quello di inviare il minimo flusso da s a t che garantisca che, per ogni oggetto j, il flusso sugli archi corrispondenti alla scelta dell'oggetto j sia almeno 1.

Imponendo l'ulteriore condizione di interezza sulle variabili x_a ($a \in A$) otteniamo un flusso intero, cioè il problema coincide con la ricerca del numero minimo di cammini da s a t che coprano tutti gli oggetti.

5 Programmazione dinamica

Consideriamo il caso particolare in cui esistono r valori distinti $s_1, \ldots, s_r$ di peso degli oggetti, ed esistono n_j esemplari di ciascun peso s_j ($j = 1, \ldots, r$); ovviamente deve essere $\sum_{j=1}^{r} n_j = n$. In questo caso esiste un algoritmo di programmazione dinamica che risolve il bin packing in modo esatto e che richiede un tempo di calcolo polinomiale in n ed esponenziale in r.

Qualunque insieme di oggetti può essere rappresentato come una r-pla $(i_1, i_2, \ldots, i_r)$ dove ogni componente i_j ($j = 1 \ldots, r$) è tale che $0 \leq i_j \leq n_j$. Dato che ciascuna componente i_j della r-pla non può eccedere n, il numero di insiemi di oggetti che ha senso considerare non può essere superiore a n^r. Tra tutti questi insiemi di oggetti, quelli che soddisfano la condizione $\sum_{j=1}^{r} i_j s_j \leq W$ sono quelli che corrispondono ad un insieme di oggetti che possono essere inseriti in un solo bin; questi sono i riempimenti ammissibili $S \in \mathcal{T}$ secondo la definizione (9).

Indicando con $B(i_1, i_2, \ldots, i_r)$ il numero minimo di contenitori necessari per impaccare i_j esemplari di oggetti di peso s_j per ogni $j = 1, \ldots, r$, lo schema di principio dell'algoritmo di programmazione dinamica è il seguente:

1. $B(0, 0, \ldots, 0) = 0$;
2. $B(a_1, a_2, \ldots, a_r) = 1 \quad \forall (a_1, a_2, \ldots, a_r) = S \in \mathcal{T}$;
3. **for** $i_1 = 0, \ldots, n_1$ **do**
4. **for** $i_2 = 0, \ldots, n_2$ **do**
5. ...
6. **for** $i_r = 0, \ldots, n_r$ **do**
7. $M = \infty$;
8. **for each** $(a_1, a_2, \ldots, a_r) \in \mathcal{T}$ such that $i_j \geq a_j \; \forall j = 1, \ldots, r$ **do**
9. **if** $M > B(i_1 - a_1, i_2 - a_2, \ldots, i_r - a_r) + 1$ **then**
10. $M = B(i_1 - a_1, i_2 - a_2, \ldots, i_r - a_r) + 1$
11. **endif**
12. **endfor**
13. $B(i_1, i_2, \ldots, i_r) = M$
14. **endfor**

15. **endfor**
16. **endfor**

Le istruzioni 1. e 2. inizializzano gli elementi della matrice di programmazione dinamica corrispondenti ai riempimenti ammissibili e richiedono un tempo di calcolo $O(n^r)$.
Il loop interno corrispondente ai passi 7.-12. definisce i restanti elementi della matrice: per ogni insieme $(i_1, i_2, \ldots, i_r)$ di oggetti, una soluzione S_1 che impacca tutti gli oggetti è ottenibile aggiungendo un riempimento ammissibile $(a_1, a_2, \ldots, a_r)$ ad una una soluzione S_2 che impacca $i_j - a_j$ oggetti di dimensione s_j per ogni $j = 1, \ldots, r$. Ovviamente questa soluzione S_2 esiste solo se $i_j - a_j \geq 0 \ \forall j = 1, \ldots, r$; in tal caso numero minimo di bin per impaccare $i_j - a_j$ oggetti di dimensione s_j (per ogni $j = 1, \ldots, r$) è già stato calcolato in un loop precedente. Il valore della soluzione S_1 dipende dal particolare riempimento ammissibile $(a_1, a_2, \ldots, a_r)$ utilizzato; valutando tutte le soluzioni ottenibili da tutti i riempimenti ammissibili di un singolo bin, si ottiene il numero minimo di bin necessari ad impaccare i_j oggetti di dimensione s_j per ogni $j = 1, \ldots, r$. Ciascuna iterazione del loop interno richiede un tempo di calcolo $O(n^{r+1})$; dato che questo loop viene eseguito $O(n^r)$ volte, il tempo di calcolo complessivo dell'algoritmo è $O(n^{2r+1})$; questo è un tempo di calcolo "ragionevole" se il valore di r è "non troppo grande".

6 Lower bound combinatori

6.1 Lower bound continuo

Data una istanza I di bin packing, consideriamo il rilassamento ottenuto rimpiazzando ciascun oggetto k ($k \in N$) con w_k oggetti di peso unitario. La nuova istanza I' è un rilassamento dell'istanza di partenza, in quanto non è stata modificata la funzione obiettivo, mentre è stato allargato l'insieme delle soluzioni ammissibili: ogni soluzione ammissibile per I è ammissibile anche per I' mentre non è detto che valga il contrario. Quindi il valore della soluzione ottima dell'istanza rilassata I' è un lower bound sul valore della soluzione ottima dell'istanza I. La soluzione dell'istanza I' è triviale: dato che tutti gli oggetti hanno peso unitario, basta inserire gli oggetti (in qualunque ordine) nei bin fino a che ci stanno. Il valore della soluzione è quindi

$$L_c = \sum_{k \in N} w_k / W \tag{29}$$

Ovviamente anche $\lceil L_c \rceil$ è un lower bound sul numero minimo di bin necessari per impaccare tutti gli oggetti. Il tempo di calcolo necessario per valutare L_c è $O(n)$.

Teorema *L_c è uguale al valore della soluzione ottima del rilassamento continuo del modello (1)–(5).*

Dim: Indichiamo con z_D il valore della soluzione ottima del rilassamento continuo del modello (1)–(5) e dimostriamo che z_D non può essere peggiore (più basso) di L_c. Infatti si ha che

$$z_D = \sum_{i \in M} y_i \geq \sum_{i \in M} (\sum_{k \in N} \frac{w_k}{W} x_{ik}) = \sum_{k \in N} \frac{w_k}{W} (\sum_{i \in M} x_{ik}) = \sum_{k \in N} \frac{w_k}{W} = L_c$$

per cui $z_D \geq L_c \quad (*)$.

Ora mostriamo che $z_D \leq L_c$. Questo si verifica facilmente, in quanto la seguente soluzione

$$x_{ik} = \begin{cases} 1 & \text{se } i = k \\ 0 & \text{altrimenti} \end{cases} \qquad (i \in M;\ k \in N)$$

$$y_i = \begin{cases} w_i/W & \text{se } i \leq |N| \\ 0 & \text{altrimenti} \end{cases} \qquad (i \in M)$$

è ammissibile per il rilassamento continuo del modello (1)–(5) ed ha valore pari a L_c. Dato che il valore della soluzione ottima di un problema di minimo è minore (o uguale) al valore di una soluzione ammissibile, allora deve essere $z_D \leq L_c \quad (**)$.
Dalle disuguaglianze $(*)$ e $(**)$ segue che deve essere $z_D = L_c$. $\square$

6.2 Lower bound L_2

Data una istanza I di bin packing ed un valore $\alpha \in [0, W/2]$, definiamo i seguenti insiemi:

$$J_1 = \{k \in N : w_k > W - \alpha\}$$
$$J_2 = \{k \in N : W - \alpha \geq w_k > W/2\}$$
$$J_3 = \{k \in N : W/2 \geq w_k \geq \alpha\}$$

Teorema

$$L(\alpha) = |J_1| + |J_2| + \max\left(0, \left\lceil \frac{\sum_{k \in J_2 \cup J_3} w_k - |J_2|W}{W} \right\rceil\right) \tag{30}$$

è un lower bound sul valore della soluzione ottima dell'istanza I.

Dim: Consideriamo il rilassamento ottenuto sostituendo l'insieme di oggetti N con gli oggetti in $J_1 \cup J_2 \cup J_3$ (quindi trascurando gli oggetti di N con $w_k < \alpha$). Dato che $J_1 \cup J_2$ non dipende da α e che ciascun oggetto in $J_1 \cup J_2$ ha peso maggiore di $W/2$ (e quindi richiede un bin separato per essere impaccato), allora $|J_1 \cup J_2|$ è un lower bound valido sul valore della soluzione ottima. Inoltre nessun oggetto di J_3 può essere impaccato insieme ad un oggetto di J_1. La capacità residua dei bin che contengono gli oggetti di J_2 è pari a $\bar{c} = |J_2|W - \sum_{k \in J_2} w_k$. Questa capacità può essere utilizzata completamente per impaccare gli oggetti di J_3; nel caso in cui non tutti gli oggetti di J_3 possano essere inseriti in questa capacità (ossia se $\sum_{k \in J_3} w_k > \bar{c}$) allora sarà necessario utilizzare almeno altri $\lceil \frac{\sum_{k \in J_3} w_k - \bar{c}}{W} \rceil$ bin per gli oggetti residui di J_3. $\square$

Dalla (30) segue che

$$L_2 = \max_{0 \leq \alpha \leq W/2} L(\alpha)$$

è un lower bound valido per il bin packing. Si noti che $L(0) = \max\{L_c, |J_1 \cup J_2|\}$, quindi il bound L_2 domina il bound continuo.
Martello e Toth hanno dimostrato che gli unici valori di α che ha interesse considerare per ottenere il bound L_2 sono solo quelli corrispondenti a valori del peso di qualche oggetto $k \in N$; pertanto il numero di valori di α da considerare è dell'ordine di $O(n)$. Dato che il calcolo di $L(\alpha)$ richiede tempo $O(n)$, una banale implementazione di L_2 porta ad un tempo di calcolo di $O(n^2)$. Si può dimostrare che se gli oggetti sono ordinati per valori decrescenti di peso, il lower bound L_2 può essere calcolato in tempo lineare.

7 Algoritmi approssimati

7.1 Algoritmo Next Fit

Il più semplice algoritmo euristico per il bin packing è l'algoritmo greedy noto come Next Fit (NF). Il primo oggetto è assegnato al primo bin, che viene detto il *bin corrente*. Ciascun oggetto k successivo è assegnato al bin corrente, se possibile; altrimenti l'oggetto k viene inserito in un nuovo bin, che diventa il bin corrente.

- **Algoritmo NF**

  1. Inizializza: $x_{ik} = 0 \ \ \forall i, k$ (soluzione),
     $i := 0$ (numero di bin utilizzati),
     $\bar{c} := 0$ (capacità residua del bin corrente).
  2. **For** $k = 1, \ldots, n$ **do**
  3. **if** $(w_k \leq \bar{c})$
  4. **then** $x_{ik} := 1, \ \ \bar{c} := \bar{c} - w_k$
  5. **else** $i := i + 1$, $x_{ik} := 1, \ \ \bar{c} := W - w_k$
  6. **endif**
  7. **endfor**

Teorema $r(NF) = 2$.

Dim: Sia I una istanza di bin packing e sia z^H il valore della soluzione prodotta dall'algoritmo NF. La somma dei pesi degli oggetti inseriti in un generico bin i dall'algoritmo NF può essere arbitrariamente piccola rispetto a W. Però, se consideriamo due bin successivi (i e $i + 1$) la somma dei pesi degli oggetti inseriti in questi bin è certamente maggiore di W (altrimenti l'algoritmo NF avrebbe inserito questi oggetti in un unico bin). Quindi, per ogni coppia $(i, i + 1)$ di bin consecutivi, si ha che

$$\sum_{k \in N} w_k (x_{i,k} + x_{i+1,k}) > W \geq W + \varepsilon$$

per qualche $\varepsilon > 0$; cioè ogni coppia di bin consecutivi è riempita almeno per metà della loro capacità.
Indicando con $t = \lfloor z^H/2 \rfloor$ si ha

$$\sum_{k \in N} w_k = \sum_{i=1}^{z^H} \sum_{k \in N} w_k \ x_{ik} \geq \sum_{p=1}^{t} \sum_{i=2p-1}^{2p} \sum_{k \in N} w_k \ x_{ik} \geq \sum_{p=1}^{t} (W + \varepsilon) = t \ (W + \varepsilon) \qquad (31)$$

Dato che $z^* \geq L_c = \dfrac{\sum_{k \in N} w_k}{W}$, la (31) equivale a dire che $z^* \geq \dfrac{t(W + \varepsilon)}{W} > t$ e quindi

$$z^* \geq t + 1 > \frac{z^H}{2}$$

Per provare che la ratio è effettivamente 2 basta considerare la seguente famiglia di istanze con $4n$ oggetti, capacità $W = 1$ e pesi definiti nel seguente modo: $w_k = 1/2$ se k è dispari

e $w_k = 1/(2n)$ se k è pari. La soluzione prodotta dall'algoritmo NF è composta da $2n$ bin, ciascuno contenente un oggetto di indice pari ed un oggetto di indice dispari. Viceversa la soluzione ottima prevede di utilizzare solo $n+1$ bin: ciascuno dei primi n contiene 2 oggetti di peso $1/2$, mentre l'ultimo bin contiene i $2n$ oggetti di peso $1/(2n)$. Quindi si ha $r(NF) = \dfrac{2\ n}{n+1} \to 2$ □

7.2 Limiti sull'approssimabilità del Bin Packing

Per il problema del bin packing non è possibile avere un algoritmo polinomiale che dia una approssimazione "buona" a piacere. Il limite sulla approssimabilità del problema è stabilito dal seguente teorema.

Teorema *Non esiste un algoritmo polinomiale A per 1BP con worst-case performance ratio $r(A) < 3/2$, a meno che non sia $P = NP$.*

Dim: Per dimostrare il teorema occorre introdurre il *Partition Problem* (PP), che è noto essere NP-completo: dato un insieme $N = \{1, \ldots, n\}$ di numeri non negativi a_k $(k \in N)$, determinare se esiste una partizione (N_1, N_2) di N tale che

$$\sum_{k \in N_1} a_k = \sum_{k \in N_2} a_k = b = \frac{\sum_{k \in N} a_k}{2}$$

Ragioniamo per assurdo supponendo che esista un algoritmo A per il quale si ha $r(A) < 3/2$. Data una istanza I di PP, definiamo una istanza di 1BP con n oggetti, il k-esimo con peso $w_k = a_k$ e con bin di capacità $b = \frac{\sum_{k \in N} w_k}{2}$. Se la soluzione ottima del problema di bin packing vale 2, allora la risposta al problema di partizionamento è si; altrimenti, la soluzione ottima richiede almeno 3 bin, per cui non esiste il partizionamento degli oggetti in 2 sottoinsiemi. Applicando l'algoritmo A con $r(A) < 3/2$ ed ottenendo una soluzione approssimata di valore z^A si riesce subito a valutare se la soluzione ottima dell'istanza di 1BP vale 2 oppure un valore maggiore di 2. Infatti se z^A vale 2, allora anche z^* vale 2, e la risposta del PP è si. Se invece $z^A \geq 3$, allora si ha anche che $z^* > 2$, per cui la risposta del PP è no. Quindi, se esistesse l'algoritmo A ipotizzato, potremmo risolvere in tempo polinomiale il partition problem, il che è impossibile a meno che non sia $P = NP$. □

8 PTAS per il bin packing?

Il risultato negativo della sezione 7.2 ci assicura che, per il problema del bin packing, non è possibile avere un algoritmo polinomiale con errore assoluto piccolo a piacere (a meno che non sia $P = NP$).
Utilizzando però l'algoritmo di programmazione dinamica descritto nella sezione 5 e l'algoritmo Next Fit della sezione 7.1 si ottiene un algoritmo che è asintoticamente un PTAS, cioè un algoritmo che trova una soluzione che asintoticamente è "vicina a piacere" al valore della soluzione ottima. Questo algoritmo non è propriamente un PTAS, nel senso che riesce a contenere (a piacere) l'errore asintotico (per $z^* \to \infty$) ma non l'errore assoluto (per valori piccoli di z^*).

L'algoritmo riceve in input l'accuratezza ϵ richiesta e funziona nel seguente modo:

1. Definisci un valore di soglia $\overline{w} = \epsilon W$ e partiziona gli oggetti nell'insieme degli oggetti "grandi" $B = \{k \in N : w_k > \overline{w}/2\}$ e in quello degli oggetti "piccoli" $S = \{k \in N : w_k \leq \overline{w}/2\}$. Siano $n_B = |B|$ e $n_S = |S|$.
2. Impacca gli oggetti di B in b bin nel seguente modo:
   (a) ordina gli oggetti di B per valori non crescenti di peso;
   (b) definisci $\ell = \lceil \epsilon \frac{\sum_{k \in N} w_k}{W} \rceil$ e $p := \lceil \frac{n_B}{\ell} \rceil$ e raggruppa gli oggetti di B, a partire da quelli più grandi, in p gruppi $G_1, \ldots, G_p$, ciascuno con ℓ oggetti (tranne eventualmente l'ultimo gruppo);
   (c) definisci una istanza I' nel seguente modo:
      – trascura gli oggetti di G_1;
      – per ogni altro gruppo G_i $(i = 2, \ldots, p)$ definisci il peso di tutti gli oggetti di G_i come il peso del primo (più grande) oggetto del gruppo;
   (d) determina la soluzione ottima $z^*(I')$ dell'istanza I' utilizzando l'algoritmo di programmazione dinamica;
   (e) impacca gli oggetti di G_1 in ℓ bin separati.
3. Impacca gli oggetti di S con l'algoritmo Next Fit nei b bin inizializzati dagli oggetti di B, eventualmente utilizzando dei nuovi bin.

Teorema *Per ogni istanza I di bin packing si ha $z^H(I) \leq (1 + \epsilon)z^*(I) + 1$.*

Dim: Il passo 1. prevede di dividere gli oggetti in "grandi" e "piccoli"; gli oggetti di questi due sottoinsiemi saranno impaccati nei passi 2. e 3., rispettivamente.
Il passo 2. prevede di impaccare gli oggetti grandi utilizzando l'algoritmo di programmazione dinamica descritto nella sezione 5, il cui tempo di calcolo è polinomiale in n_B ma esponenziale nel numero r di pesi diversi presenti in B. Per ridurre il tempo di calcolo, definiamo una istanza I' ausiliaria e risolviamo questa nuova istanza invece di impaccare gli oggetti di B. In particolare, partizioniamo gli oggetti di B in p gruppi, a partire dagli oggetti più grandi: ciascun gruppo G_i $(i = 1, \ldots, p-1)$ comprende ℓ oggetti (con $\ell = \lceil \epsilon \frac{\sum_{k \in N} w_k}{W} \rceil$), mentre l'ultimo gruppo G_p è composto dagli utimi (più piccoli) $n_B - \ell(p-1)$ oggetti. L'istanza I' è definita dagli oggetti dei gruppi $G_2, \ldots, G_p$, il cui peso viene incrementato (o lasciato inalterato) con l'operazione di *grouping* degli oggetti. Nell'istanza I' vi sono al massimo $r = p - 1$ dimensioni distinte, una per ciascun gruppo G_i $(i = 2, \ldots, p)$. Quindi è trovare il valore $z^*(I')$ della soluzione ottima dell'istanza I' in tempo $O(n^{2p-1})$ applicando l'algoritmo di programmazione dinamica della sezione 5 a tale istanza.

Esempio: consideriamo il seguente insieme di 11 oggetti per l'insieme B dell'istanza I originale. Prendendo $\ell = 3$ (massima dimensione di ciascun gruppo di oggetti) si ha $p = 4$ (numero di gruppi); l'istanza I' ha 8 oggetti ma solo $p - 1 = 3$ valori distinti di pesi.

Insieme B	10	9	8	7	7	6	6	6	5	3	2
	G_1			G_2			G_3			G_4	
Istanza I'				7	7	7	6	6	6	3	3

È ovvio che $z^*(I') \leq z^*(B)$ in quanto: (i) l'istanza I' ha $n_B - \ell$ oggetti mentre l'insieme B è composto da n_B oggetti; (ii) il generico oggetto k ($k = 1, \ldots, n_B - \ell$) dell'istanza I' ha peso non superiore al peso del k-esimo oggetto dell'insieme B. Quindi applicando l'algoritmo di programmazione dinamica all'istanza I' ed impaccando separatamente gli oggetti di G_1 in ℓ bin (uno per ogni oggetto), si ottiene un impaccamento di tutti gli oggetti di B che utilizza b bin con

$$b = z^*(I') + \ell \leq z^*(B) + \ell \leq z^*(I) + \ell$$

Al passo 3., gli oggetti piccoli vengono inseriti nei b bin aperti tramite l'algoritmo Next Fit, eventualmente aprendo dei nuovi bin. In questa fase si possono verificare due eventualità.

(a) Nel caso in cui non sia necessario aprire dei nuovi bin, il valore della soluzione euristica è pari a b, per cui

$$z^H \leq z^* + \ell = z^* + \lceil \epsilon \frac{\sum_{k \in N} w_k}{W} \rceil \leq z^* + \epsilon \frac{\sum_{k \in N} w_k}{W} + 1 \leq z^* + \epsilon\ z^* + 1 = (1 + \epsilon)\ z^* + 1$$

da cui segue la tesi.

(b) Nel caso in cui sia necessario utilizzare dei nuovi bin, ciascun bin (tranne eventualmente l'ultimo) è pieno almeno per $W - \overline{w}/2$; se infatti esistesse un bin con riempimento minore di $W - \overline{w}/2$, allora l'algoritmo Next Fit avrebbe inserito in quel bin almeno un altro oggetto (dato che ogni oggetto di S ha peso non superiore a $\overline{w}/2$). Pertanto si ha che

$$z^H \leq \lceil \frac{\sum_{k \in N} w_k}{W - \frac{\overline{w}}{2}} \rceil \leq \frac{\sum_{k \in N} w_k}{W - \frac{\overline{w}}{2}} + 1 = \frac{\sum_{k \in N} w_k}{W\ (1 - \frac{\overline{w}}{2W})} + 1 = \frac{\sum_{k \in N} w_k}{W\ (1 - \frac{\epsilon}{2})} + 1 \leq z^*\ \frac{1}{1 - \frac{\epsilon}{2}} + 1$$

Per $\epsilon \leq 1$ si ha che $\dfrac{1}{1 - \epsilon/2} \leq 1 + \epsilon$, per cui

$$z^H \leq z^*\ (1 + \epsilon) + 1 = (1 + \epsilon)\ z^* + 1$$

e quindi anche in questo caso segue la tesi. □

Per quanto riguarda il tempo di calcolo dell'algoritmo, i passi 1. e 3. possono essere eseguiti in tempo lineare.
Al passo 2. l'istanza I' viene definita in tempo lineare e risolta, tramite programmazione dinamica, in tempo $O(n_B^{2r+1})$ con $r = p - 1 \leq n_B/\ell - 1$. Dato che

$$\frac{n_B}{\ell} \leq \frac{n_B}{\epsilon \frac{\sum_{k \in N} w_k}{W}} \leq \frac{n_B}{\epsilon \frac{n_B\ \overline{w}}{W}} \leq \frac{n_B}{\epsilon \frac{n_B\ \epsilon}{2}} = \frac{2}{\epsilon^2}$$

allora il tempo di calcolo del passo 2. è dell'ordine di $O(n^{4/\epsilon^2})$, cioè è polinomiale in n ed esponenziale in $1/\epsilon^2$.