Uvod u računarske algoritme

dr.sc. Edin Pjanić

Pregled predavanja

- Pohlepni (greedy) algoritmi
- Backtracking

Pohlepni algoritmi (greedy algorithms)

- Pohlepni algoritmi su mnogo jednostavniji od DP.
- Pohlepni algoritam uvijek ide onim putem (bira onaj korak) koji izgleda najbolji u datom trenutku.
- Bira se lokalni optimum sa nadom da će dovesti do globalnog optimuma.
- Ovi algoritmi ne vode uvijek do optimalnog rješenja.
- Za mnoge probleme vode ka optimalnom rješenju.

- Primjer:
- Problem: ruksak ← greedy
- Problem: 0-1 ruksak ← DP (ne može greedy)

Problem ruksaka: Natovariti ruksak ograničenog kapaciteta stvarima tako da u ruksaku bude najveća moguća vrijednost.

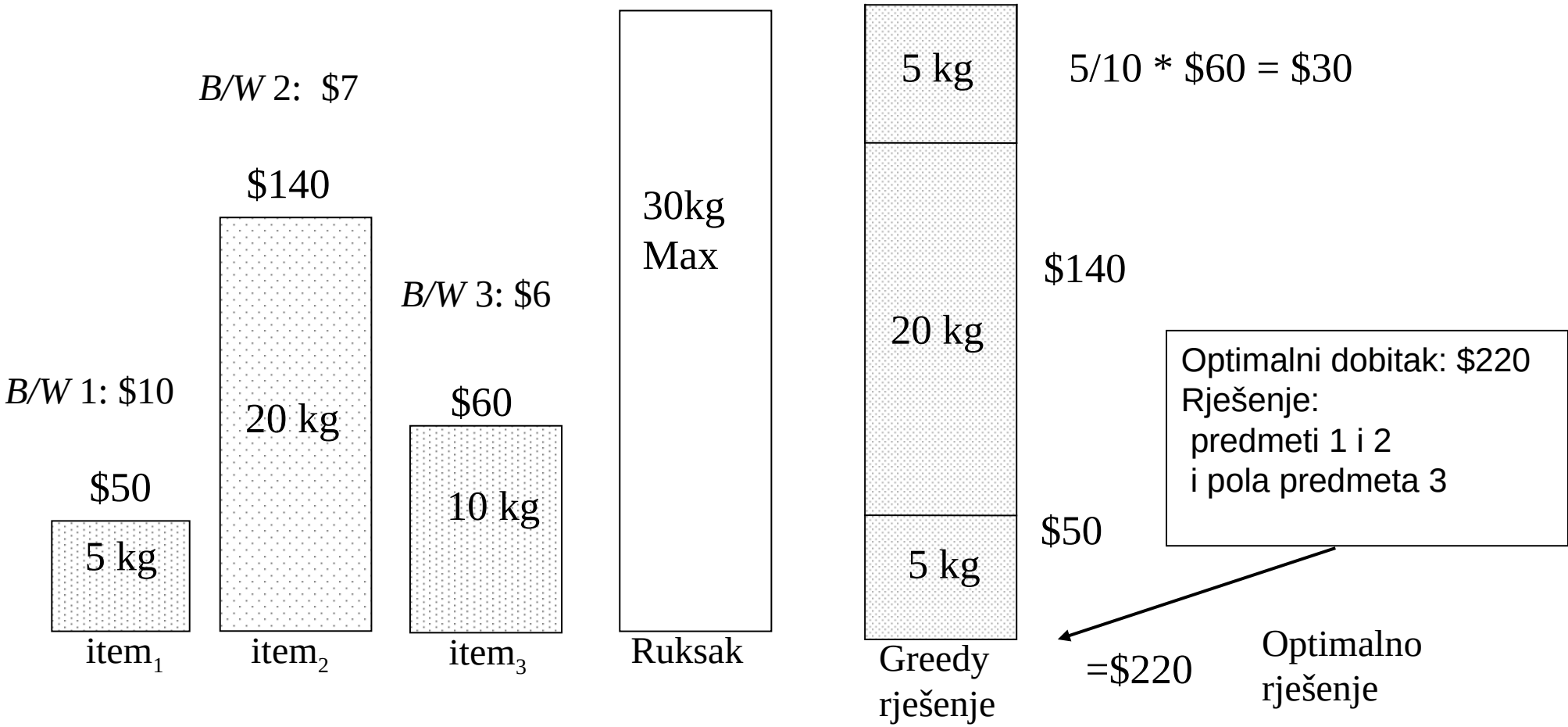
Pohlepni algoritam - primjer: ruksak

Na raspolaganju su različiti materijali ograničene količine.

Može se uzeti proizvoljna količina (%) svakog materijala.

S = { (item1 , 5, $50), (item2, 20, $140) (item3 ,10, $60), }

Optimalni kriterij izbora: Najvredniji predmet po jedinici mase



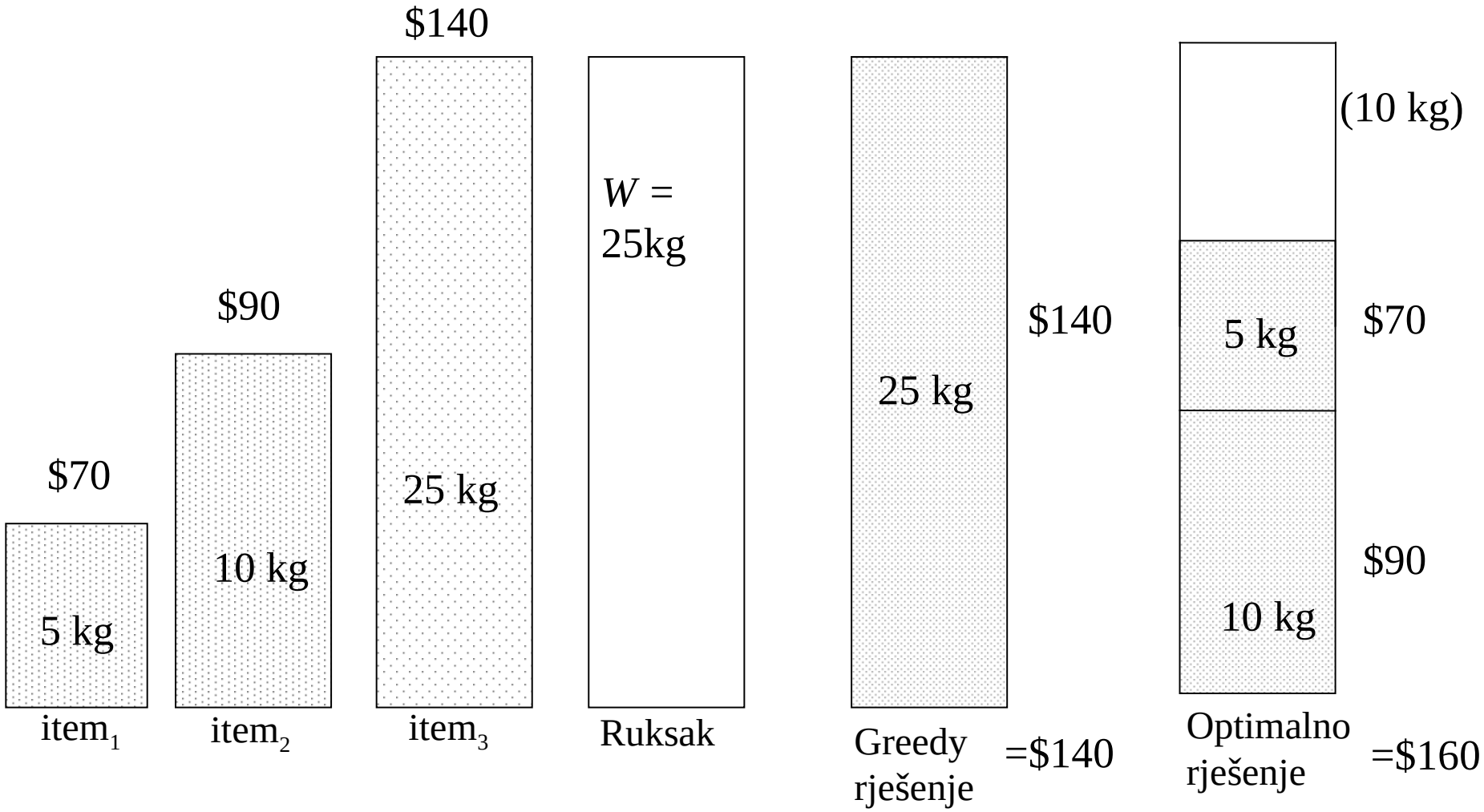
B/W: benefit per weight

Kontraprimjeri

primjeri problema u kojima se ne može koristiti pohlepni algoritam

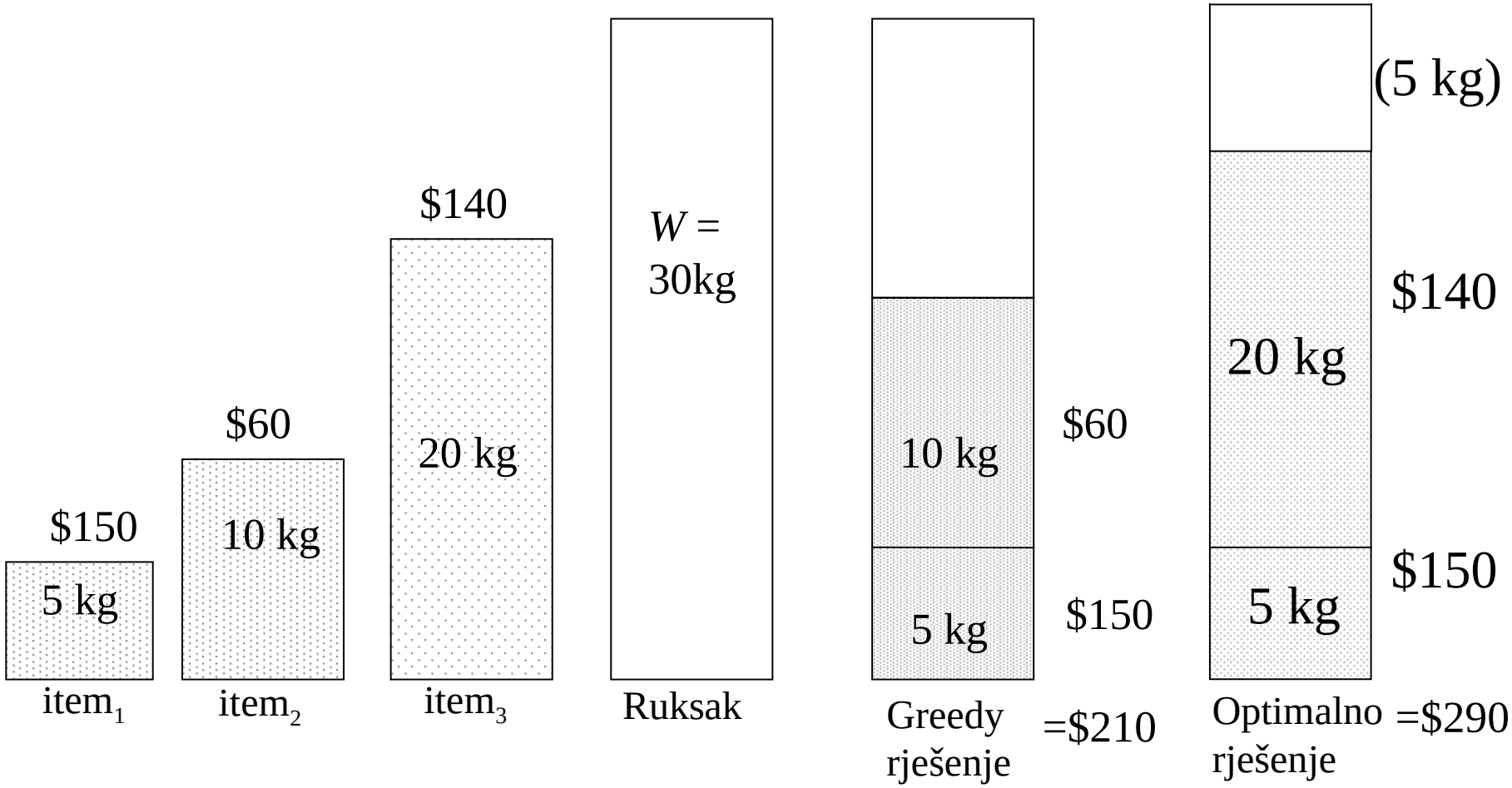
Kontraprimjer 1: ruksak 0/1 (uzmi ili ostavi)
Kriterij izbora: Najvredniji predmet.

Na raspolaganju su različiti predmeti. Može se uzeti samo cijeli predmet.
S = { ($item_1$, 5, \$70), ($item_2$,10, \$90), ($item_3$, 25, \$140) }



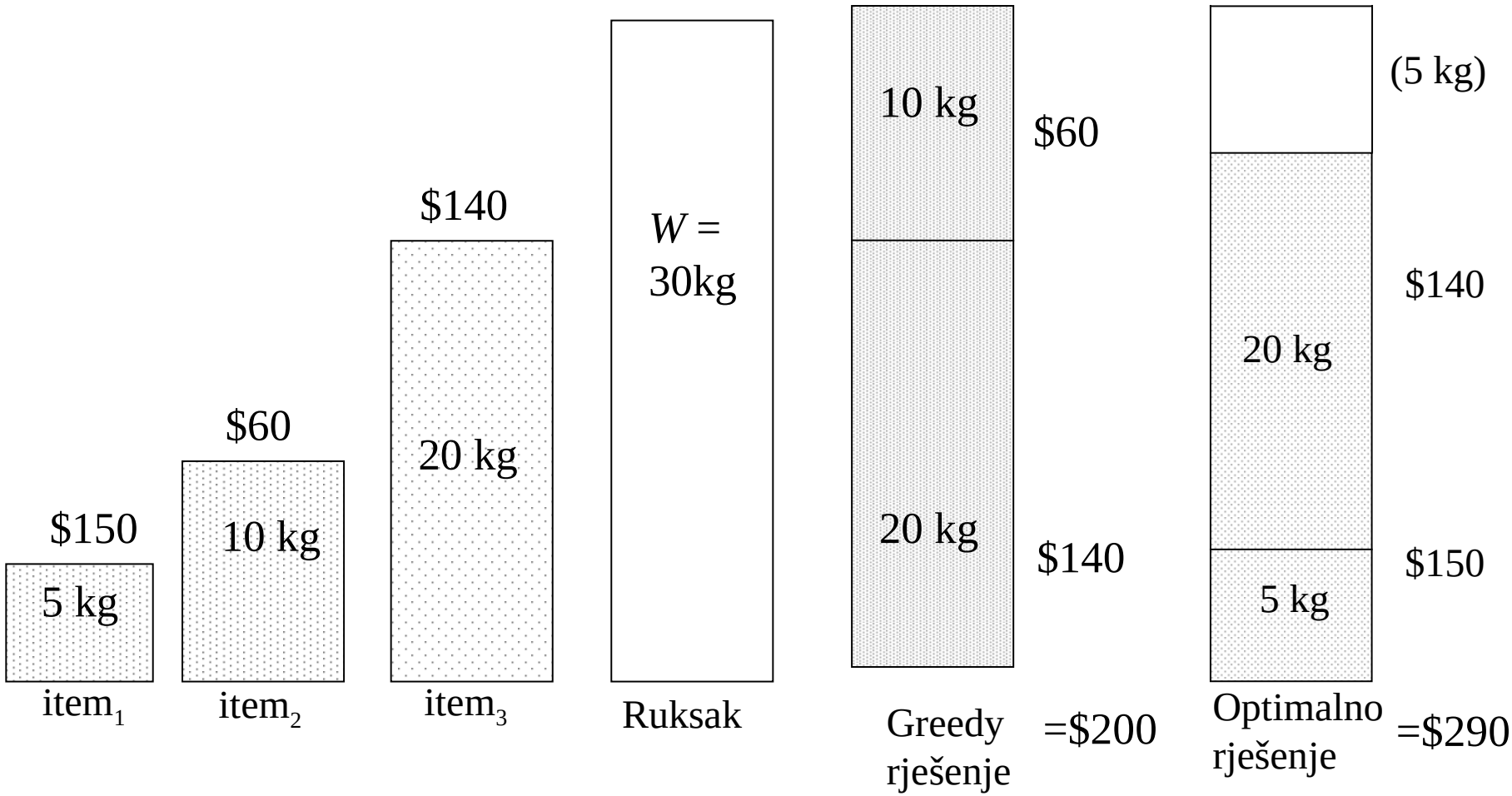
Kontraprimjer 2: ruksak 0/1
Kriterij izbora: Najlakši predmet.

S = { ($item_1$, 5, \$150), ($item_2$,10, \$60), ($item_3$, 20, \$140) }



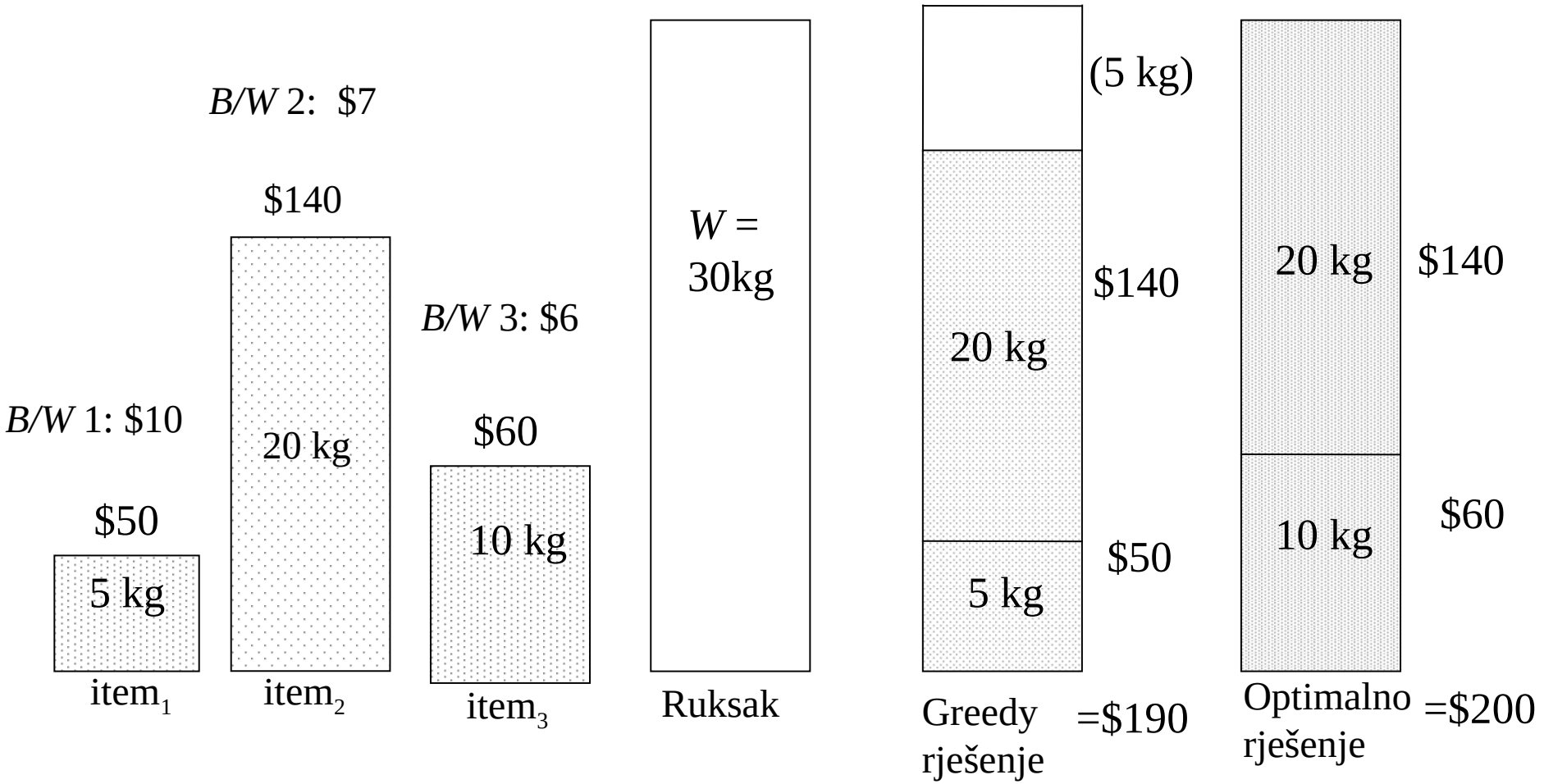
Kontraprimjer 3: ruksak 0/1
Kriterij izbora: Najteži predmet.

$S = \{ (item_1 , 5, \$150), (item_2 ,10, \$60), (item_3, 20, \$140) \}$



Kontraprimjer 4: ruksak 0/1
Kriterij izbora: Najvredniji predmet po jedinici mase.

S = { ($item_1$, 5, \$50), ($item_2$, 20, \$140) ($item_3$,10, \$60), }



Strategija pri rješavanju problema pohlepnim algoritmom

- Optimizacioni problem napisati u obliku u kojem nam nakon prvog izbora (pohlepnog) ostaje jedan podproblem.
- Dokazati da uvijek postoji optimalno rješenje originalnog problema koje sadrži pohlepni izbor.
- Dokazati da se kombinacijom optimalnih rješenja podproblema pohlepnim izborom dolazi do optimalnog rješenja originalnog problema.

Backtracking

- ***Backtrack*** – vraćanje jedan korak unazad.
- Ovo je način pronalaska rješenja problema na način da se inkrementalno gradi (bira) lista kandidata rješenja (rekurzivno)
- Čim se utvrdi da neki kandidat za rješenje ne može dati ispravno rješenje odbacuje se taj kandidat, odnosno cijela familija kandidata koja bi slijedila iz tog kandidata, i vraćamo se unazad (uradimo “backtrack”)
- Ovaj način se uglavnom koristi za probleme koji imaju jasno utvrđena ograničenja
- Primjeri:
    - Raspored kraljica na šahovskoj ploči
    - Ukrštenica (križaljka)
    - Sudoku
    - Parsiranje
    - itd.