VREMENSKI RASPORED INTERVALA S TEŽINAMA

DINAMIČKO PROGRAMIRANJE

Loredana Musap

Definicija problema

ulaz:

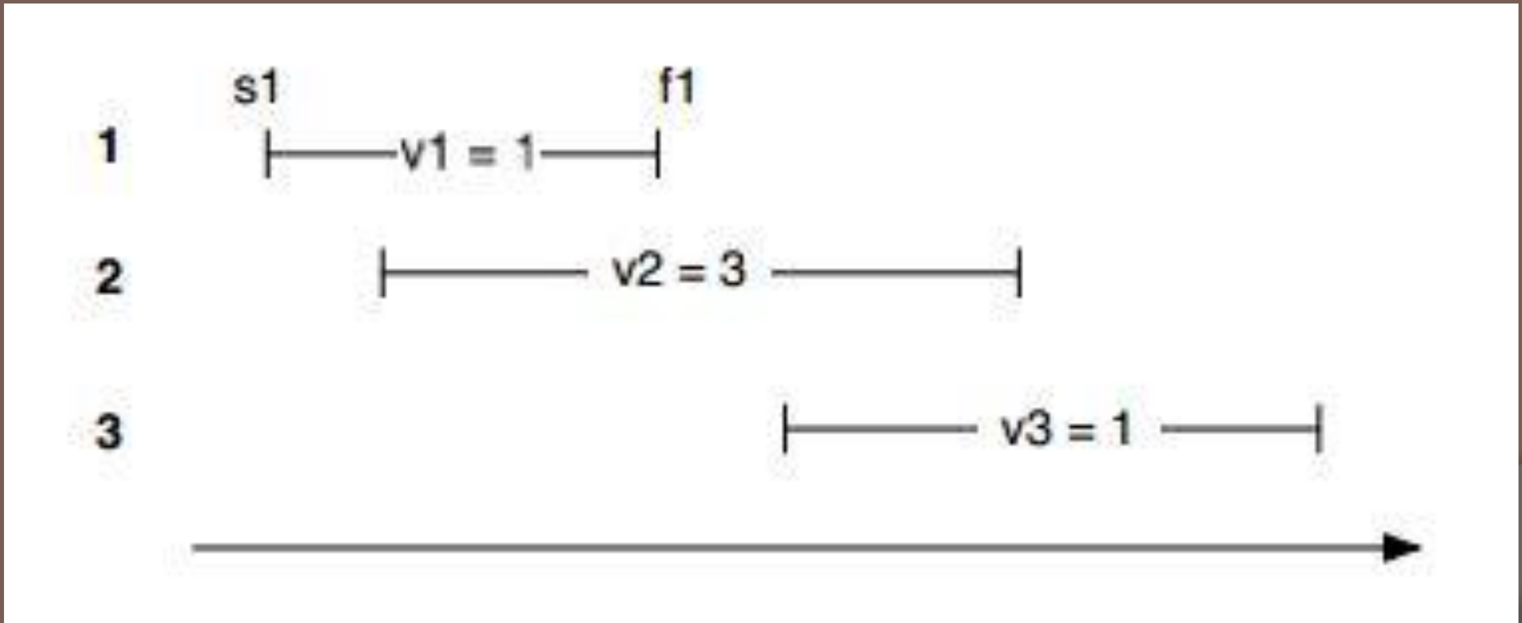
- skup N intervala
- interval i -> s_i, f_i, v_i

izlaz:

- skup kompatibilnih intervali
- suma težina maksimalna

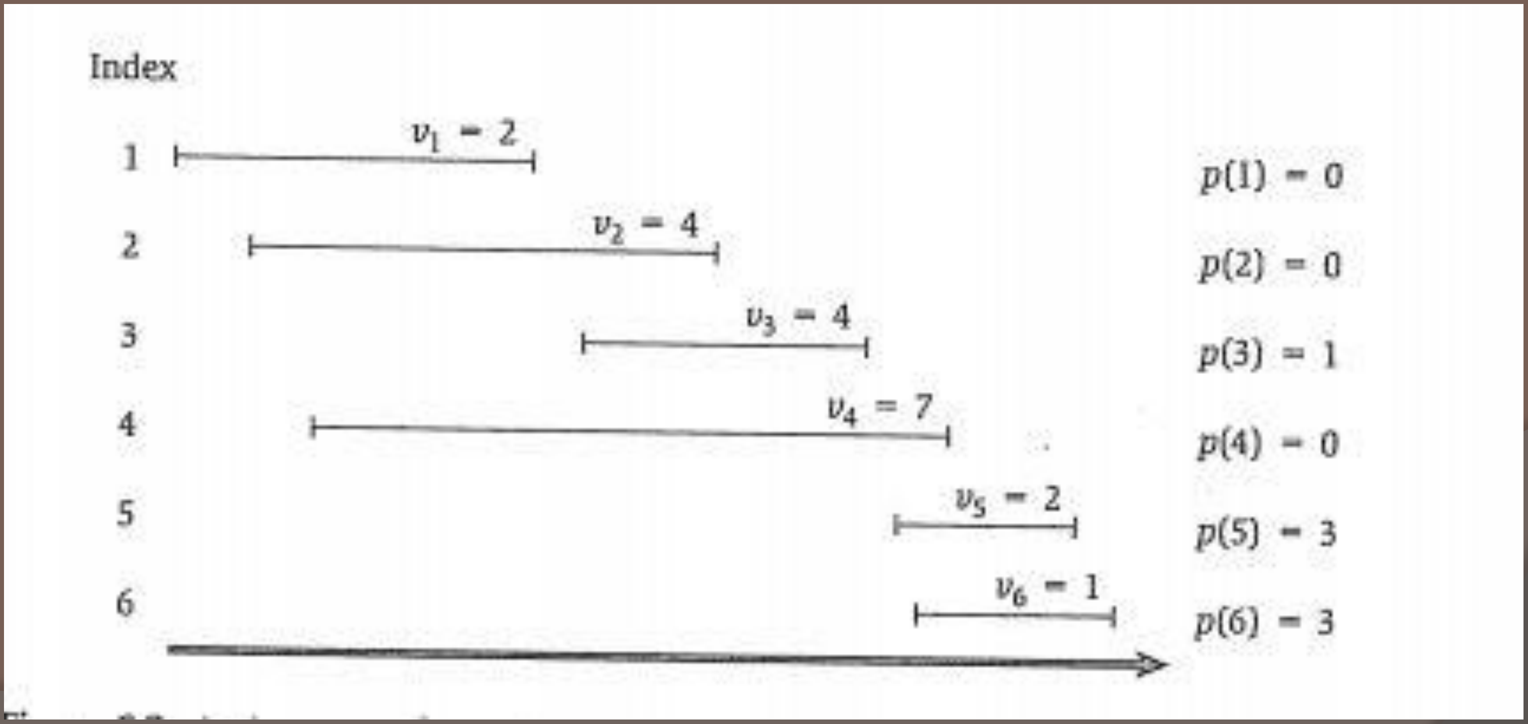
Greedy pristup

- problem rasporeda intervala “bez težina”
- kriterij: prema vremenima kraja



Skica rješenja

- intervali sortirani prema vremenu kraja
- p[j] = najveći indeks i < j takav da su intervali i, j kompatibilni



Skica rješenja

- O – optimalano rješenje
- vrijedi:
  - $n \in O$
    - {p(n)+1, ..., n} – nisu u optimalnom skupu rješenja
    - u O se nalazi optimalno rješenje od {1, ..., p(n)}
  - $n \notin O$
    - O = skupu optimalnih rješenja za {1, ..., n-1}

Skica rješenja

- traženje rješenja na podskupovima {1, ..., j}
  - O_j - skup intervala optimalnog rješenja
  - OPT(j) – vrijednost optimalnog rješenja
- $j \in O_j$, tada $\mathrm{OPT(j)} = \mathrm{v_j} + \mathrm{OPT(p(j))}$
- $j \notin O_j$, tada $\mathrm{OPT(j)} = \mathrm{OPT(j\text{-}1)}$

Skica rješenja

- vrijednost optimalnog rješenja:

$$OPT(j) = \max(v_j + OPT(p(j)), OPT(j-1))$$

- j je u skupu optimalih rješenja akko

$$v_j + OPT(p(j)) \geq OPT(j-1)$$

Rekurzivni pristup (brute force)

```
Rek_WIS(j)
    If j=0 then
        Return 0
    Else
        Return max (v_j + Rek_WIS(p(j)), Rek_WIS(j − 1))
    Endif
```

$Rek\_WIS(j)$

If $j=0$ then

Return 0

Else

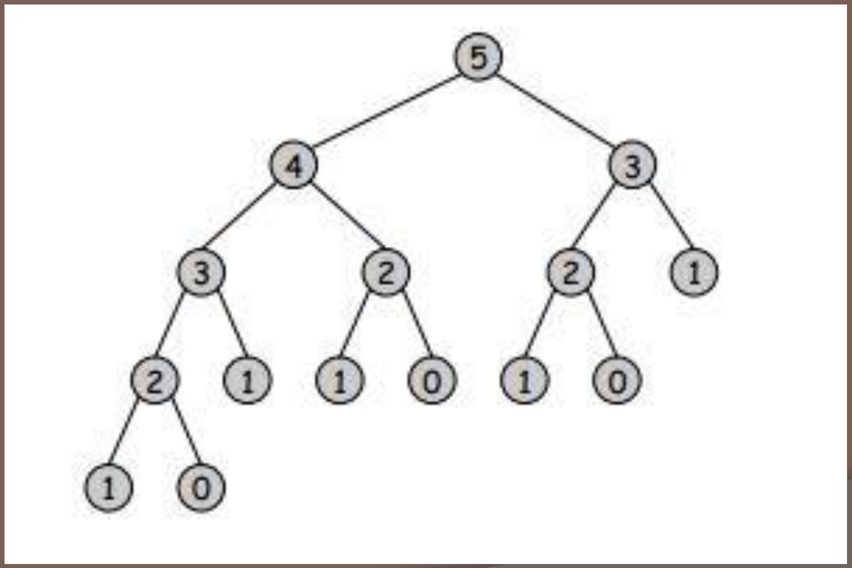
Return $\max\left(v_j + Rek\_WIS(p(j)), Rek\_WIS(j-1)\right)$

Endif

Rekurzivni pristup (brute force)

- složenost eksponencijalna

j	0	1	2	3	4	5	6	7	8	…	20	30	50
$T(j)$	1	2	3	5	8	13	21	34	55	…	17,711	2,178,309	32,951,280,099



Rekurzivni pristup (pamćenje)

```
M-Rek_WIS(j)
    If j = 0 then
        Return 0
    Else if M[j] is not empty
        Return M[j]
    Else
        M[j] = max (v_j + M − Rek_WIS(p(j)) , M − Rek_WIS(j − 1) )
        Return M[j]
    Endif
```

Rekurzivni pristup (pamćenje)

- Složenost: O(n)
- Složenost čitavog algoritma:
  - sortiranje: O(n log n)
  - izračun p(1), ..., p(n) : O(n log n)
  - M-Rek_WIS : O(n)
  - nalaženje skupa optimalnog rješenja: O(n)

  - ukupno: O(n log n)

Iterativni pristup

- jednake složenosti
- iterativno računanje M[1], ..., M[n]

Iter_WIS

M[0] = 0

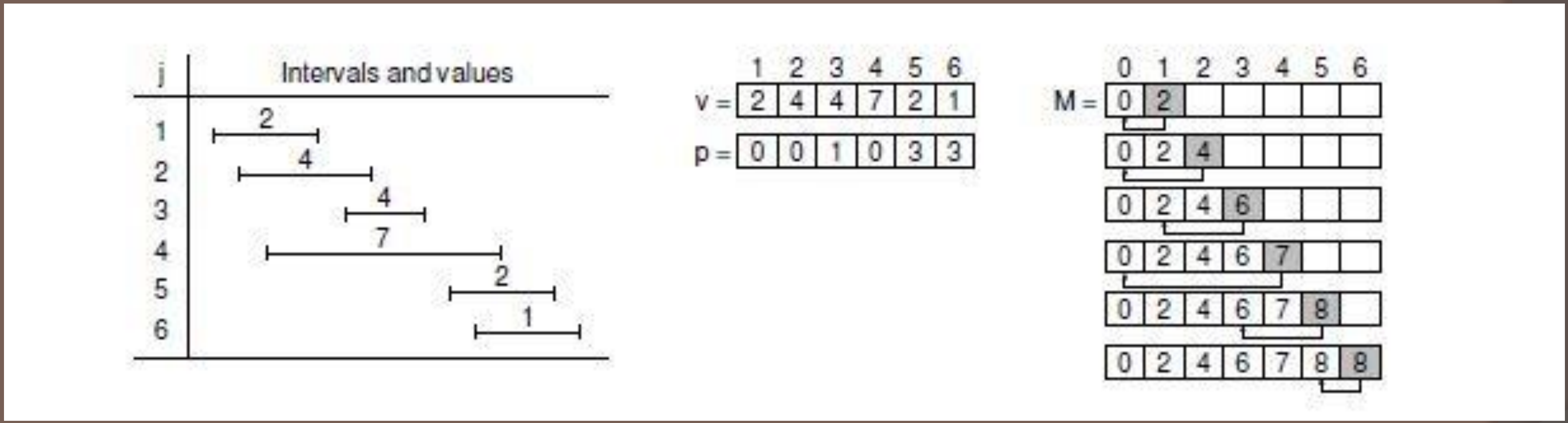
For j=1, ... , n

$M[j] = \max(v_j + M[p(j)], M[j-1])$

Endfor

Iterativni pristup

- primjer



Optimalan skup intervala

```
Opt_Rjes(j)
    If j = 0 then
        Output nothing
    Else
        If v_j + M[p(j)] ≥ M[j − 1] then
            Output j together with result of Opt_Rjes(p(j))
        Else
            Output Opt_Rjes(j − 1)
        Endif
    Endif
```

Implementacija

- c++, Visual studio

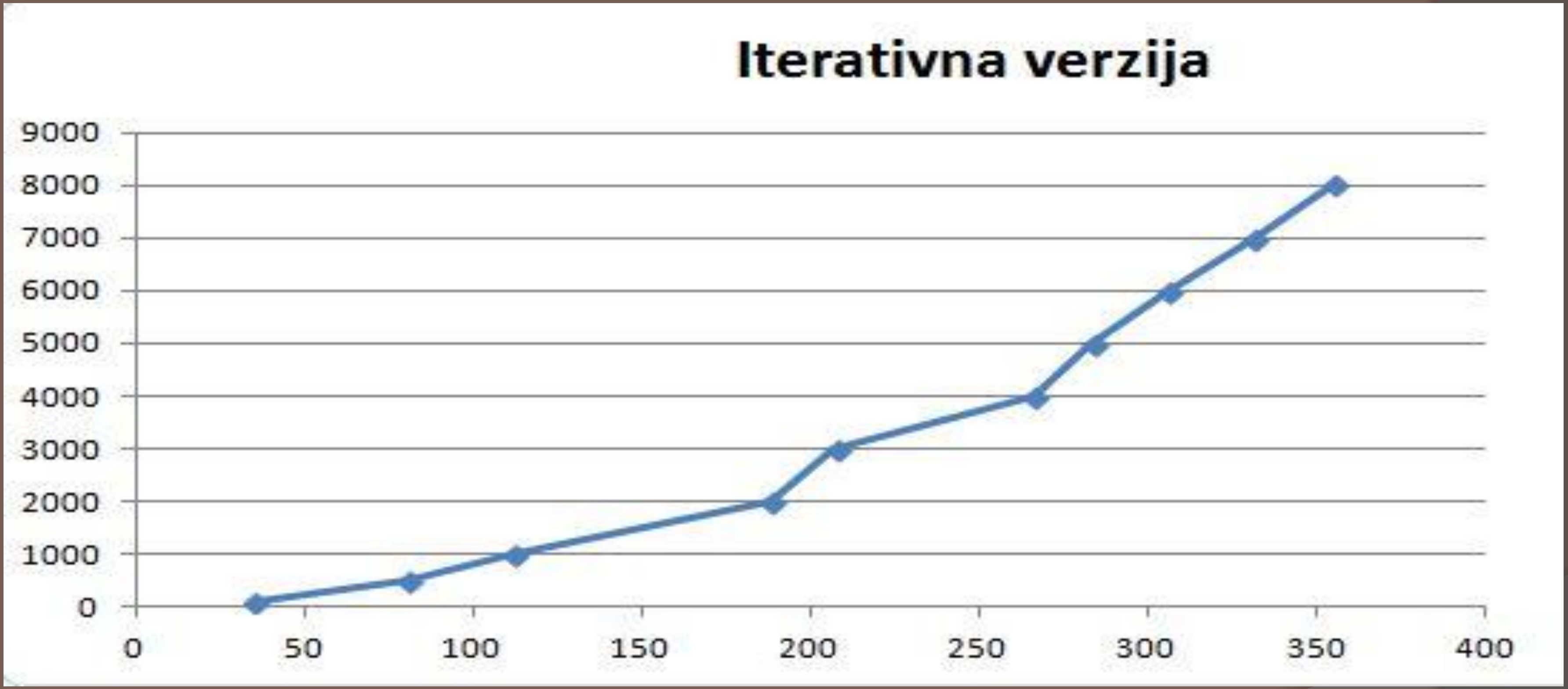
```
int PosljednjiNepreklapajuciInterval(Interval *intervali, int i)
{
    int S;
    int pocetak = intervali[i].pocetak;
    int i1 = 0, i2 = i-1;

    while(i1 < i2)
    {
        S = i1 + (i2 - i1 + 1) / 2;

        if(intervali[S].kraj <= pocetak) i1 = S;
        else i2 = S - 1;
    }

    if(intervali[i1].kraj > pocetak) return 0;
    return i1;
}
```

Testiranje – iterativna verzija



Testiranje – rekurzivna verzija

Testiranje

- očekivani rezultati
- iterativna verzija u implementaciji bolja

- procesor:
  - Intel Core i5-3337U 1.8 GHz
  - 4 GB ram

Literatura

[1] J. Kleinberg, E. Tardos – Algorithm Design, Cornell University

[2] David M. Mount – Design and Analysis of Computer Algorithms, CMSC 451