Căutare paralelă.

Ciprian Dobre
ciprian.dobre@cs.pub.ro

Dezvoltarea aplicaţiilor pentru SM SIMD

- Shared Memory SIMD = PRAM *(Parallel Random Access Machines)*
- Sistemele SM SIMD se subdivid în:
  - EREW *(Exclusive Read Exclusive Write)*
  - CREW *(Concurrent Read Exclusive Write)*
  - ERCW *(Exclusive Read Concurrent Write)*
  - CRCW *(Concurrent Read Concurrent Write)*
- Caracteristici ale algoritmilor:
  - indicarea acţiunilor executate de procesoare
  - explicitarea separată a acţiunilor de citire şi de scriere
  - lansarea şi terminarea proceselor au "overhead" redus
  - algoritmii se încadrează, de obicei, în clasa "fine grain".

Proprietăţi (dorite) ale algoritmilor paraleli

- **Număr de procesoare**
  - **p(n) < n** – uzual, p(n) este funcţie sublineară de n
  - **p(n) adaptiv** - algoritmul să nu depindă de un număr fix de procesoare
- **Timp de execuţie**
  - **t(n) mic** - semnificativ mai mic decât al variantei secvenţiale
  - **t(n) adaptiv** - invers proporțional cu p(n)
- **Cost**
  - **c(n) minim** – deci, algoritmul să fie cost-optimal

Căutarea paralelă

- Problema:

  Căutarea unei valori **x** într-o secvenţă ordonată **S** de **n** valori

- Presupunere - elementele secvenţei sunt toate distincte între ele:

$$s[1] < s[2] < \ldots < s[n]$$

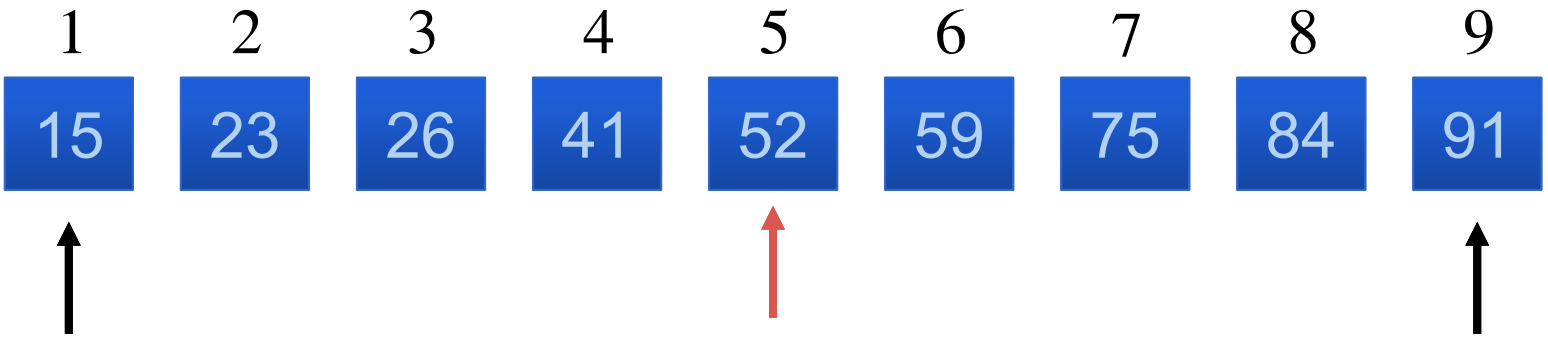
- Varianta secvenţială:
  - Căutare binară
  - Timp de executie: O(log n)

Căutarea binară în varianta secvenţială

Căutăm X = 59

Căutarea binară în varianta secvenţială

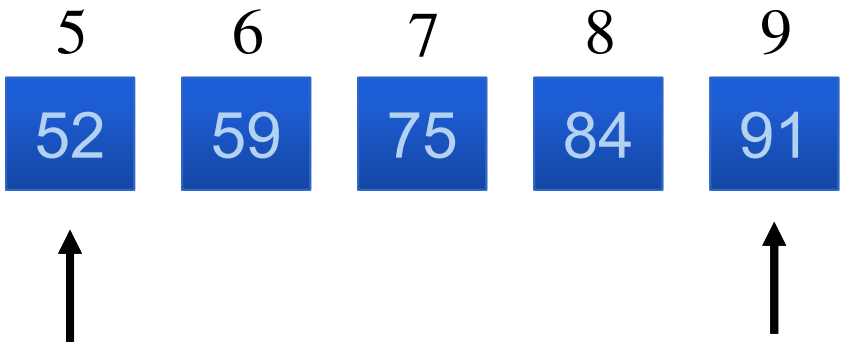
Căutăm X = 59



Verificăm valoarea de pe poziția (1 + 9) / 2 = 5

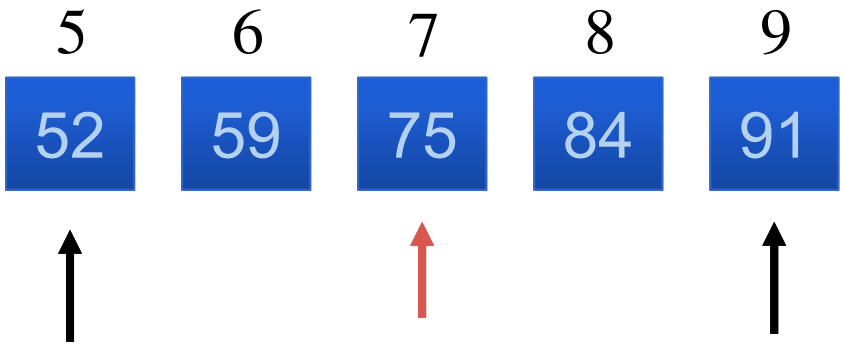
Căutarea binară în varianta secvenţială

Căutăm X = 59



Căutarea binară în varianta secvenţială

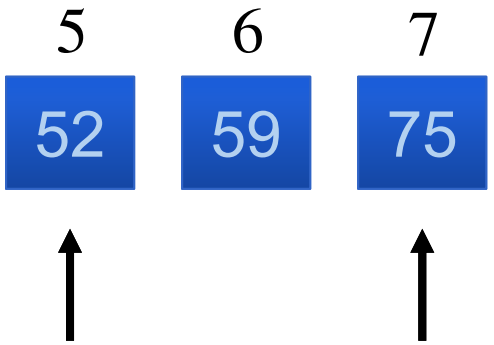
Căutăm X = 59



59 < 75 ⇒ ignorăm intervalul din dreapta

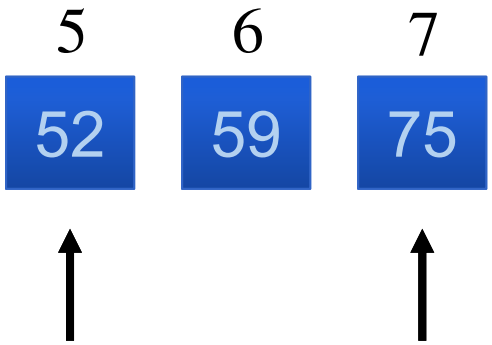
Căutarea binară în varianta secvenţială

Căutăm X = 59



Căutarea binară în varianta secvenţială

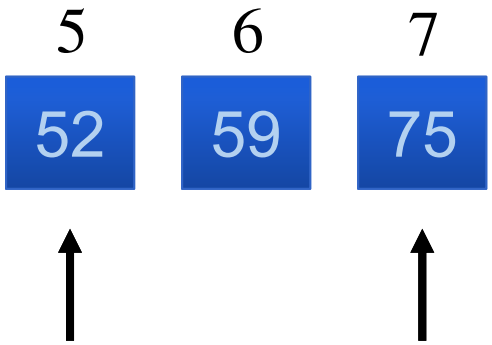
Căutăm X = 59



Verificăm valoarea de pe poziția (5 + 7) / 2 = 6

Căutarea binară în varianta secvenţială

Căutăm X = 59



Am găsit valoarea !!!

Sistem EREW

- Sistem cu P procesoare, $1 \leq P \leq n$
  - broadcast x are complexitatea $\boldsymbol{O(\log P)}$.
  - secvenţa S este divizată în N subsecvenţe de lungime n/N, fiecare secvență atribuită unui procesor :

$$P_i: \quad s\left[[(i-1)\frac{n}{P}+1\right] \ldots s\left[i\frac{n}{P}\right]$$

  - *un* procesor găseşte rezultatul folosind căutarea secvențiala în $\boldsymbol{O\left(\log \frac{n}{P}\right)}$ în cazul cel mai defavorabil.

- Timp de execuție:

$$O(\log P) + O\left(\log \frac{n}{P}\right) = \boldsymbol{O(\log n)}$$

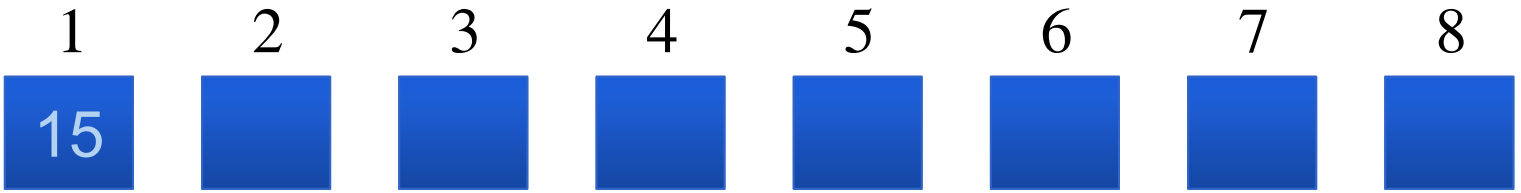
  = timpul necesar căutării binare secvenţiale.

Difuzarea unei valori

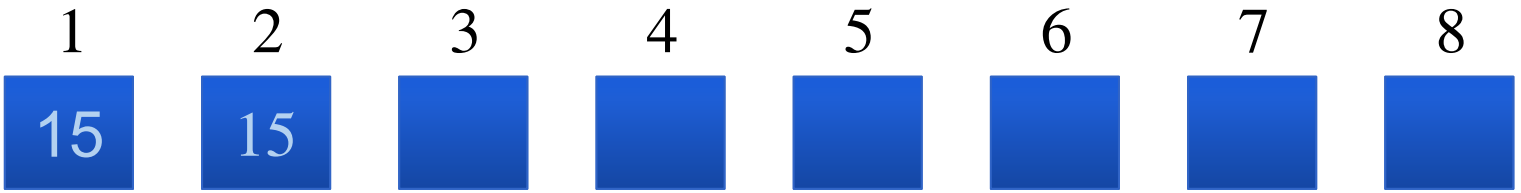
- D – celula din memoria comună ce trebuie difuzată
- Folosim un tablou A[1:N]

procedure BROADCAST (D, N, A)
Pas 1: Procesorul P1
1.1. citește valoarea din D
1.2. o memorează în propria memorie
1.3. o scrie în A[1]
Pas 2:
fa i := 0 to (log N -1) ->
fa j := 2^i+1 to $2^{(i+1)}$ do in parallel
Procesor Pj
2.1. citește valoarea din A[j-2^i]
2.2. o memorează în propria memorie
2.3. o scrie în A[j]
af
af

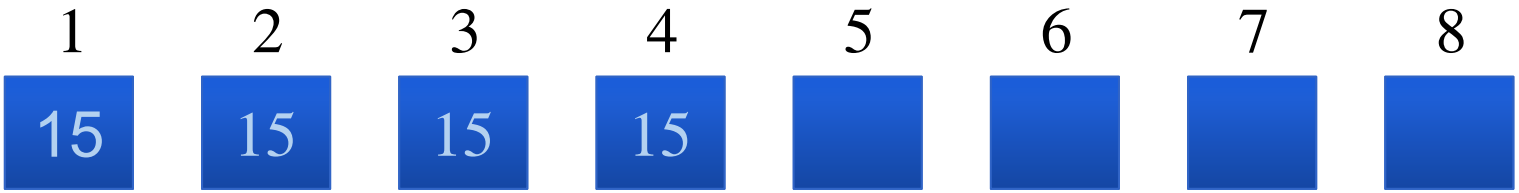
Difuzarea unei valori



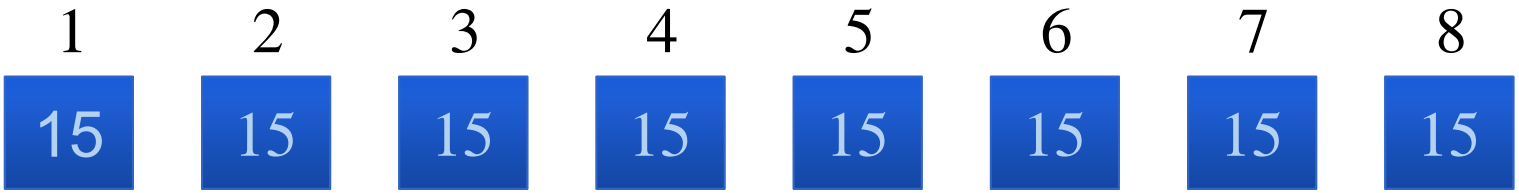
Difuzarea unei valori



Difuzarea unei valori



Difuzarea unei valori



Sisteme CREW

- Difuzarea valorii căutate = un pas.
- Algoritmul:

  - **g** : numărul de pași
  - **N** : numărul de procesoare
  - **n** : numărul de elemente ale vectorului
  - P_i : procesorul cu indicele i

- Deoarece fiecare etapă se aplică unei secvenţe de dimensiune $\frac{1}{N+1}$ din secvenţa precedentă, sunt necesare:

$$O(\log_{N+1}(n+1))$$ etape.

Sisteme CREW

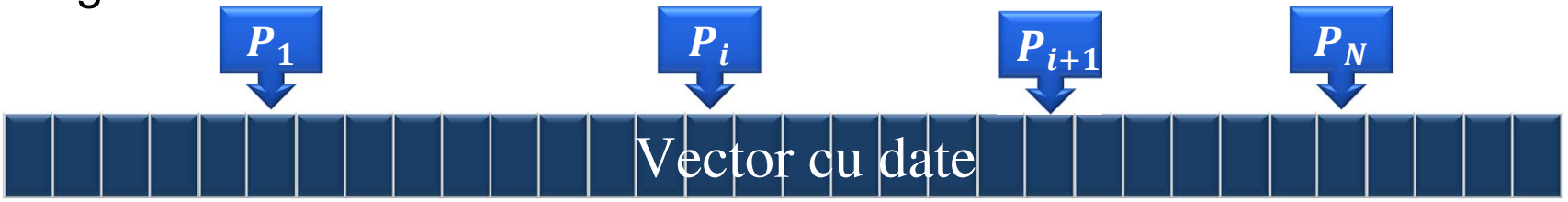
- Difuzarea valorii căutate = un pas.
- Algoritmul:

  - **g** : numărul de pași
  - **N** : numărul de procesoare
  - **n** : numărul de elemente ale vectorului
  - P_i : procesorul cu indicele i
- Deoarece fiecare etapă se aplică unei secvenţe de dimensiune $\frac{1}{N+1}$ din secvenţa precedentă, sunt necesare:

$$O(\log_{N+1}(n+1))$$ etape.

Sisteme CREW

- Difuzarea valorii căutate = un pas.
- Algoritmul:

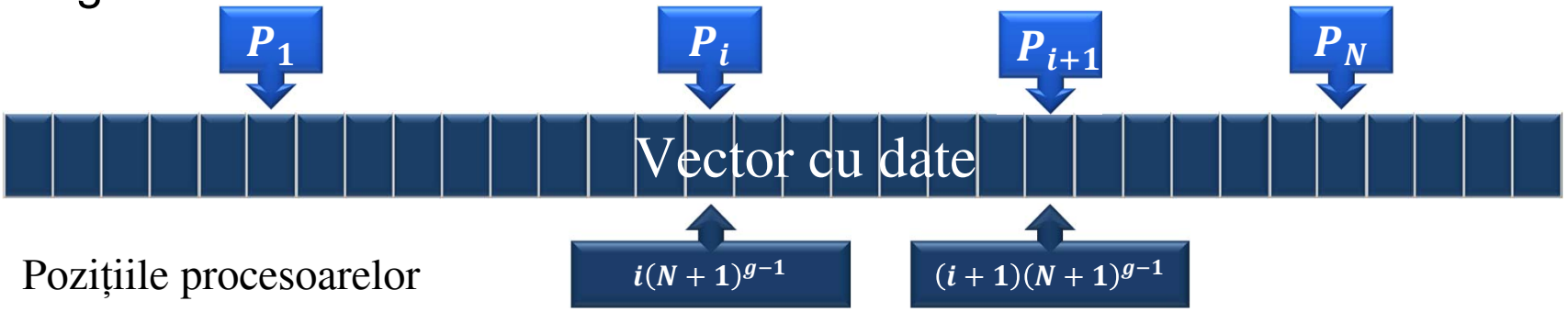


  - **g** : numărul de pași
  - **N** : numărul de procesoare
  - **n** : numărul de elemente ale vectorului
  - P_i : procesorul cu indicele i
- Deoarece fiecare etapă se aplică unei secvenţe de dimensiune $\frac{1}{N+1}$ din secvenţa precedentă, sunt necesare:

$$O(\log_{N+1}(n+1))$$ etape.

Sisteme CREW

- Difuzarea valorii căutate = un pas.
- Algoritmul:

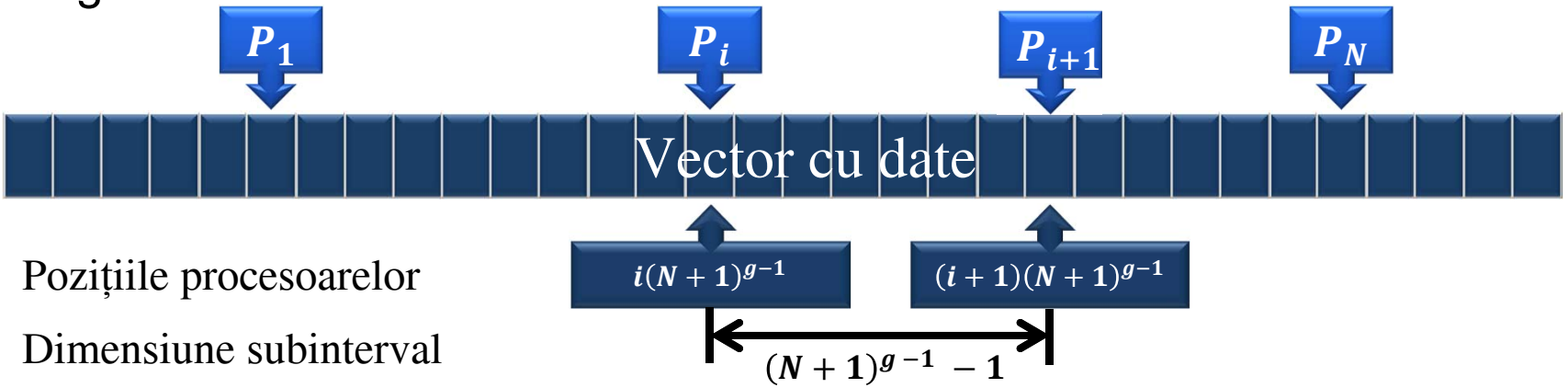


  - **g** : numărul de paşi
  - **N** : numărul de procesoare
  - **n** : numărul de elemente ale vectorului
  - P_i : procesorul cu indicele i
- Deoarece fiecare etapă se aplică unei secvenţe de dimensiune $\frac{1}{N+1}$ din secvenţa precedentă, sunt necesare:

$$O(\log_{N+1}(n+1))$$ etape.

Sisteme CREW

- Difuzarea valorii căutate = un pas.
- Algoritmul:



  - **g** : numărul de paşi
  - **N** : numărul de procesoare
  - **n** : numărul de elemente ale vectorului
  - P_i : procesorul cu indicele i
- Deoarece fiecare etapă se aplică unei secvenţe de dimensiune $\frac{1}{N+1}$ din secvenţa precedentă, sunt necesare:

$$O(\log_{N+1}(n+1)) \text{ etape.}$$

Justificare

în ultimul pas N procesoare acoperă N elemente

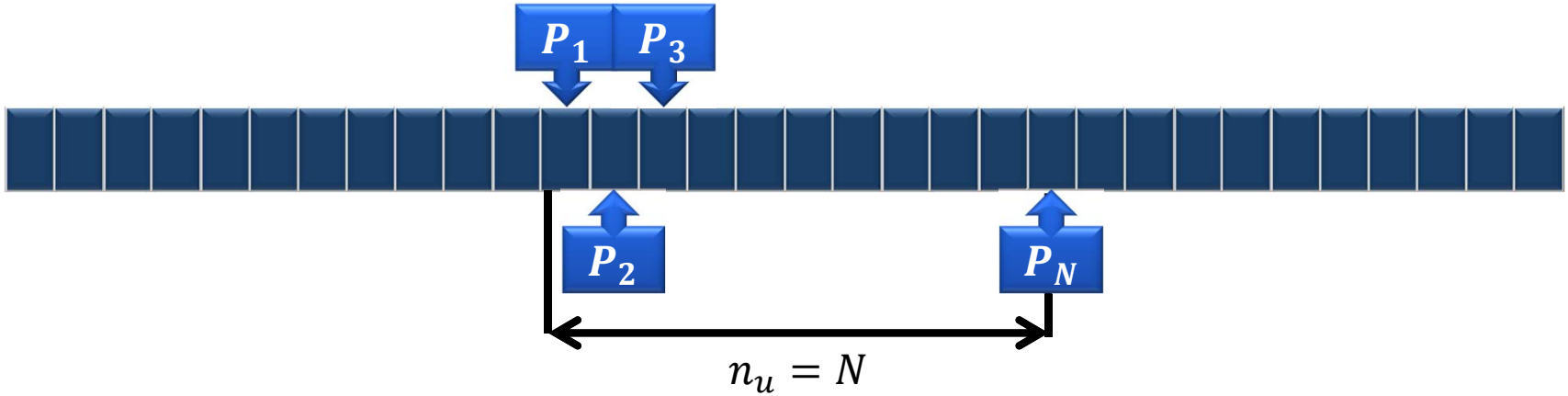
în pasul anterior $(N+1)^2 - 1$

în g paşi $(N+1)^g - 1$

din $(N+1)^g - 1 \geq n$ *(n = nr elemente şir)*

rezultă $\boldsymbol{g = sup\left(\frac{\log(n+1)}{\log(N+1)}\right)}$

Justificare



în ultimul pas N procesoare acoperă N elemente

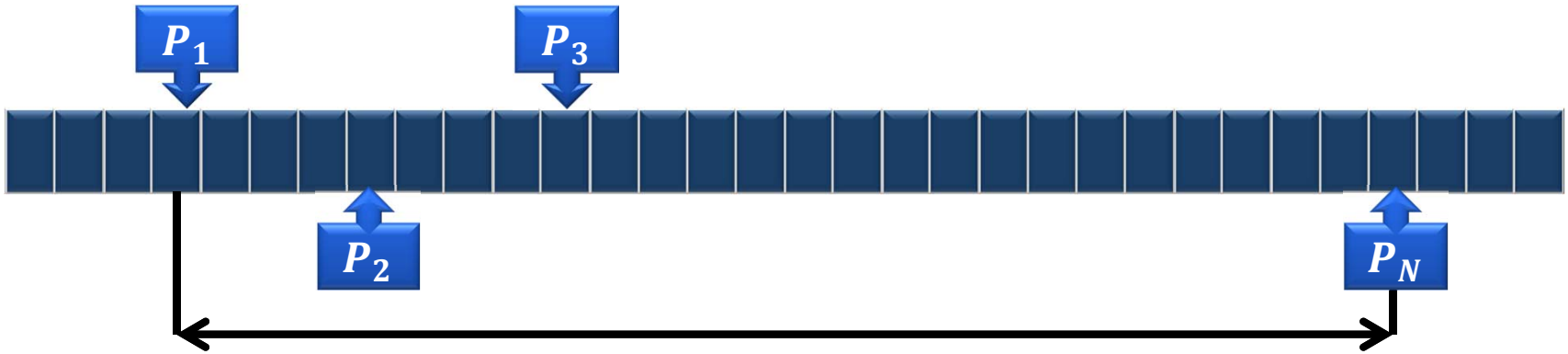
în pasul anterior $(N+1)^2 - 1$

în g paşi $(N+1)^g - 1$

din $(N+1)^g - 1 \geq n$ *(n = nr elemente şir)*

rezultă $\boldsymbol{g = sup\left(\frac{\log(n+1)}{\log(N+1)}\right)}$

Justificare



$$n_u = (N+1)(N+1)$$

în ultimul pas N procesoare acoperă N elemente

în pasul anterior $(N+1)^2 - 1$

în g paşi $(N+1)^g - 1$

din $(N+1)^g - 1 \geq n$ *(n = nr elemente şir)*

rezultă $\boldsymbol{g = sup\left(\frac{\log(n+1)}{\log(N+1)}\right)}$

Exemplu 1

1	4	6	9	10	11	13	14	15	18	20	25	32	45	51
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

$= 2$

- $\geq n + 1 \rightarrow \boldsymbol{g = 2}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.

Exemplu 1

1	4	6	9	10	11	13	14	15	18	20	25	32	45	51
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

- X=245 *(numărul căutat)*

- $\geq n + 1 \rightarrow \boldsymbol{g = 2}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.

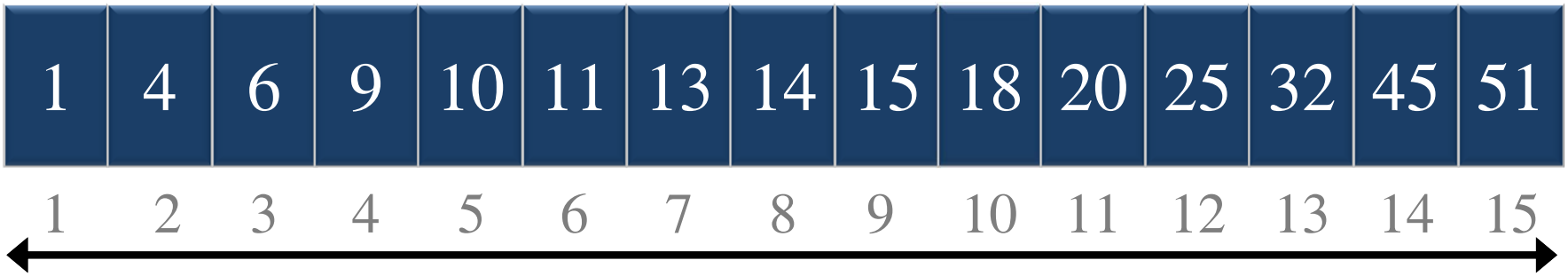
Exemplu 1

1	4	6	9	10	11	13	14	15	18	20	25	32	45	51
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

- (N=1) $g=1$ (*NN+1) (N+1) g=1 gg=1 (N+1) g=1 =**33.***
- ***NN**+1) (N+1) g gg (N+1) g ≥nn+1 →**gg=22.***
- $X = 45$ *(numărul căutat)*
- N = 3 *(numărul de procesoare)*
- n = 15 *(dimensiunea vectorului)*

- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.

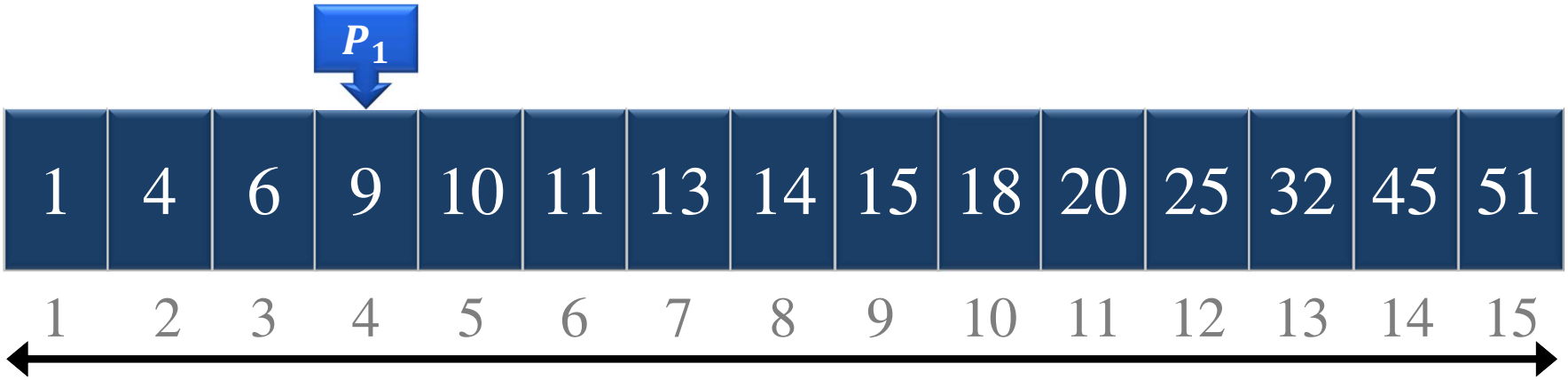
Exemplu 1



- $2g == 2$
- N = 3 *(numărul de procesoare)*

- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.

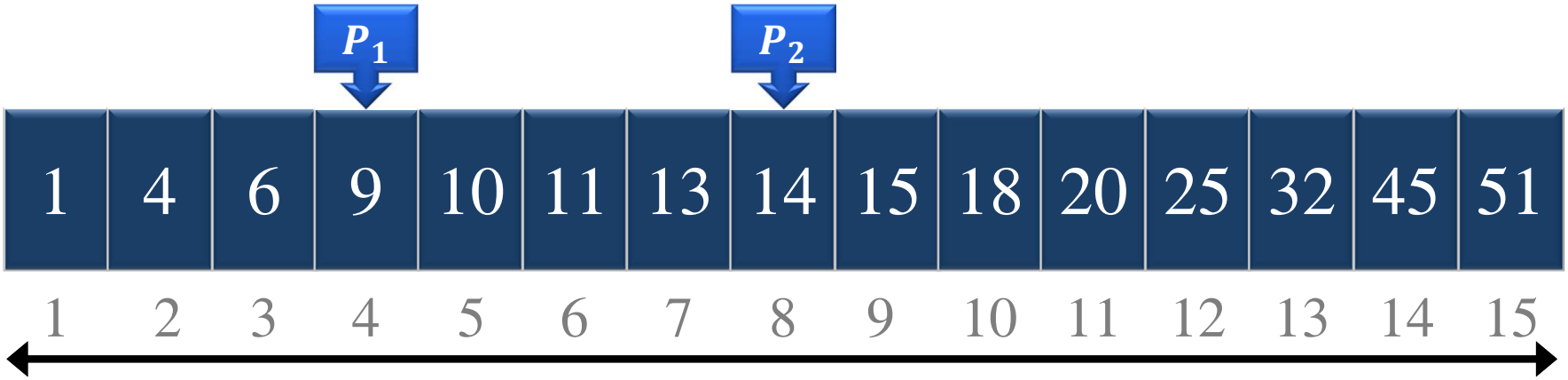
Exemplu 1



- *NN+1) i(N+1) g−1 gg−1 i(N+1) g−1 ⇒ P 1 PP P 1 1 P 1* va ocupa poziția 4 din vector
- 1
- N = 3 *(numărul de procesoare)*
  - Din $i(N+1)\ g-1 \Rightarrow P_1$ va ocupa poziția 4 din vector

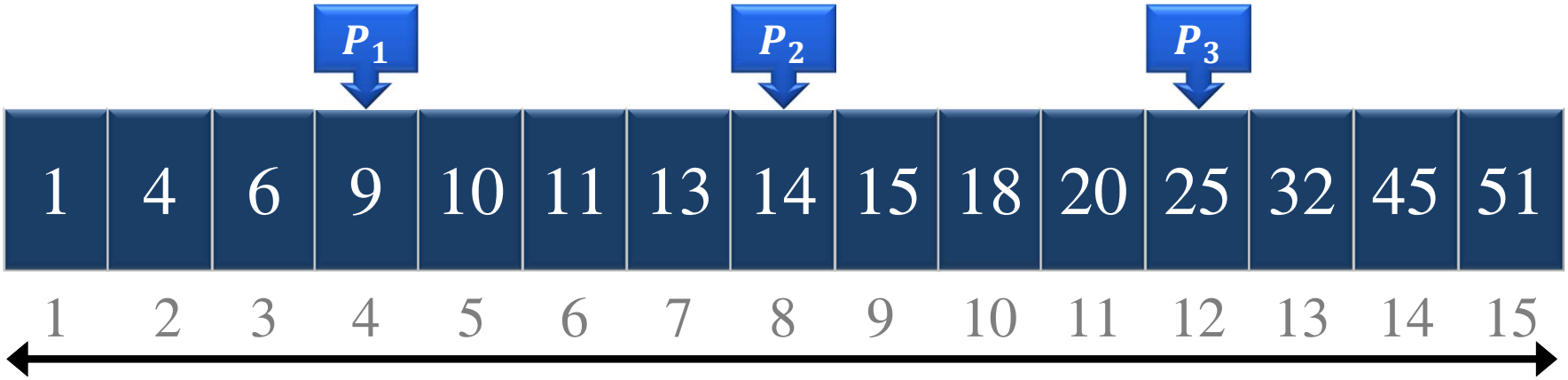
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{3}$.
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{3}$.

Exemplu 1



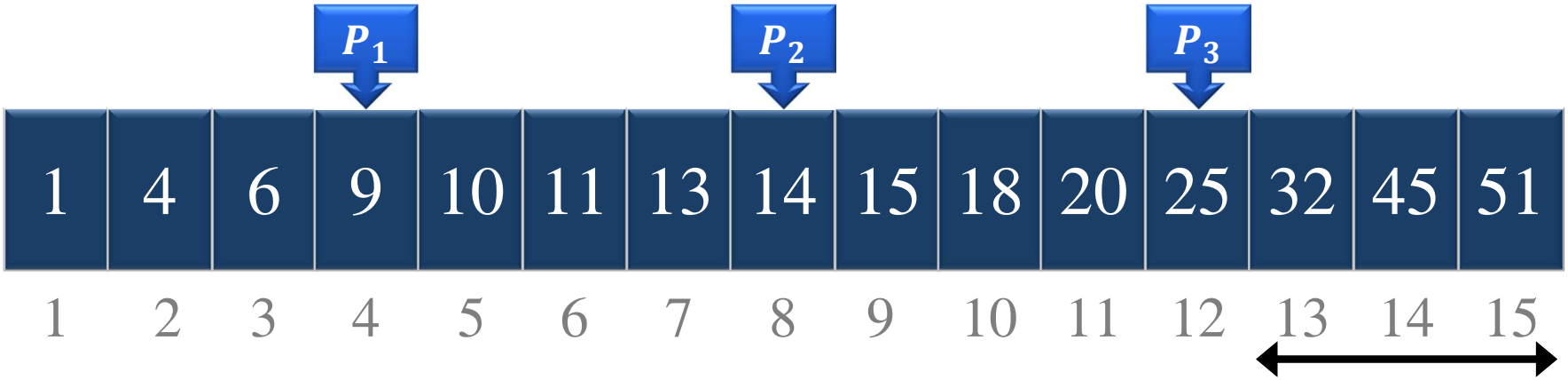
- $NN+1)\ i(N+1)\ g-1\ gg-1\ i(N+1)\ g-1 \Rightarrow P\ 2\ PP\ P\ 2\ 2\ P\ 2$ va ocupa poziția 8 din vector
- 2
- $NN+1)\ i(N+1)\ g-1\ gg-1\ i(N+1)\ g-1 \Rightarrow P\ 1\ PP\ P\ 1\ 1\ P\ 1$ va ocupa poziția 4 din vector
- 1
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{3}$.
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{3}$.

Exemplu 1



- $NN+1)\ i(N+1)\ g-1\ gg-1\ i(N+1)\ g-1 \Rightarrow\ P\ 3\ PP\ P\ 3\ 3\ P\ 3$ va ocupa poziția 12 din vector
- 3
- $NN+1)\ i(N+1)\ g-1\ gg-1\ i(N+1)\ g-1 \Rightarrow\ P\ 2\ PP\ P\ 2\ 2\ P\ 2$ va ocupa poziția 8 din vector
- 2
- Lungimea unei subsecvențe în primul pas este $(N+1)^{g-1}=\mathbf{3}$.
- Lungimea unei subsecvențe în primul pas este $(N+1)^{g-1}=\mathbf{3}$.
- 1

Exemplu 1



- $g = 1$
- N = 3 *(numărul de procesoare)*
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.

Exemplu 1

- $g = 1$
- N = 3 *(numărul de procesoare)*
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{3}$.

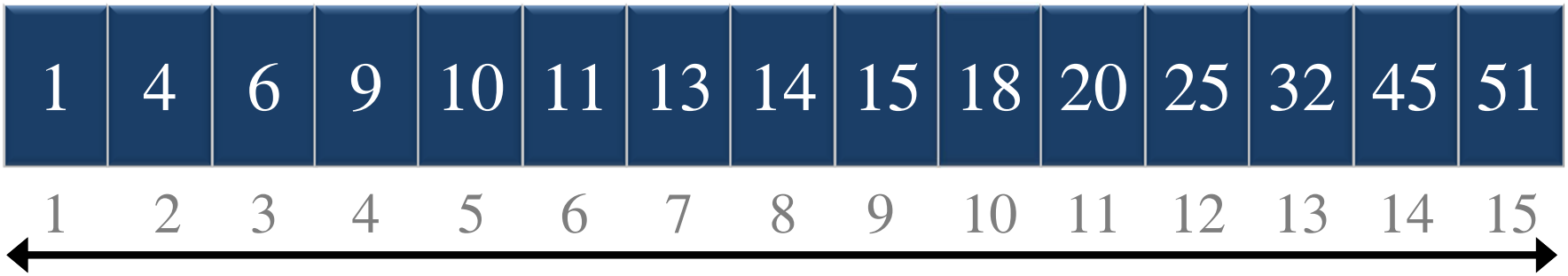
Exemplu 2

1	4	6	9	10	11	13	14	15	18	20	25	32	45	51
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

- (N3-1) $g-1$ (*NN+1) (N+1)* $g-1$ *gg*-1 *(N+1)* $g-1$ *=**88.***
- ***NN**+1) (N+1) g gg (N+1) g ≥nn+1 →**gg=33.***
- $X = 21$ *(numărul căutat)*
- N = 2 *(numărul de procesoare)*
- n = 15 *(dimensiunea vectorului)*

- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{8}$.
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{8}$.

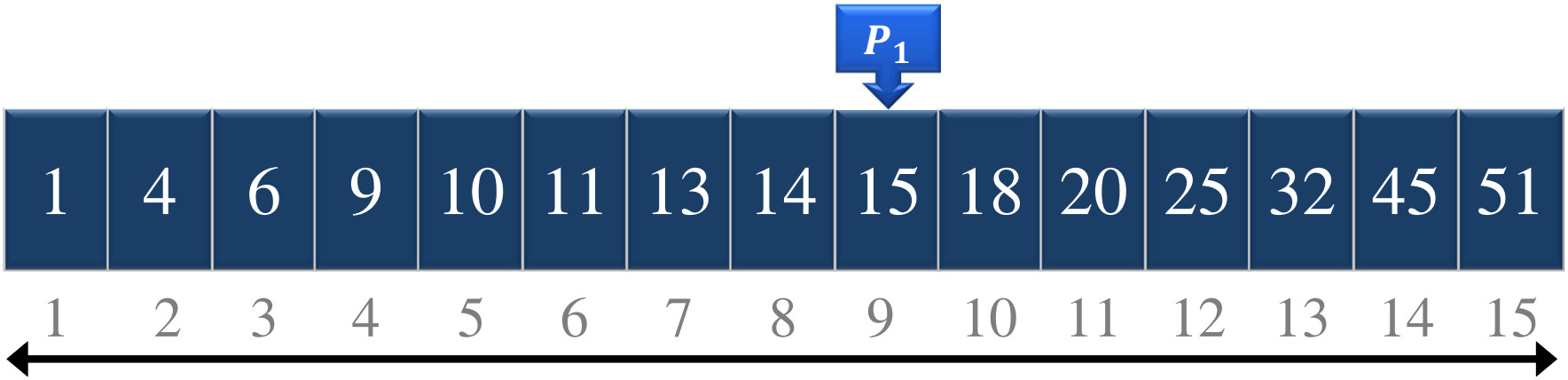
Exemplu 2



- $3g == 3$
- N = 2 *(numărul de procesoare)*

- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{8}$.
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{8}$.

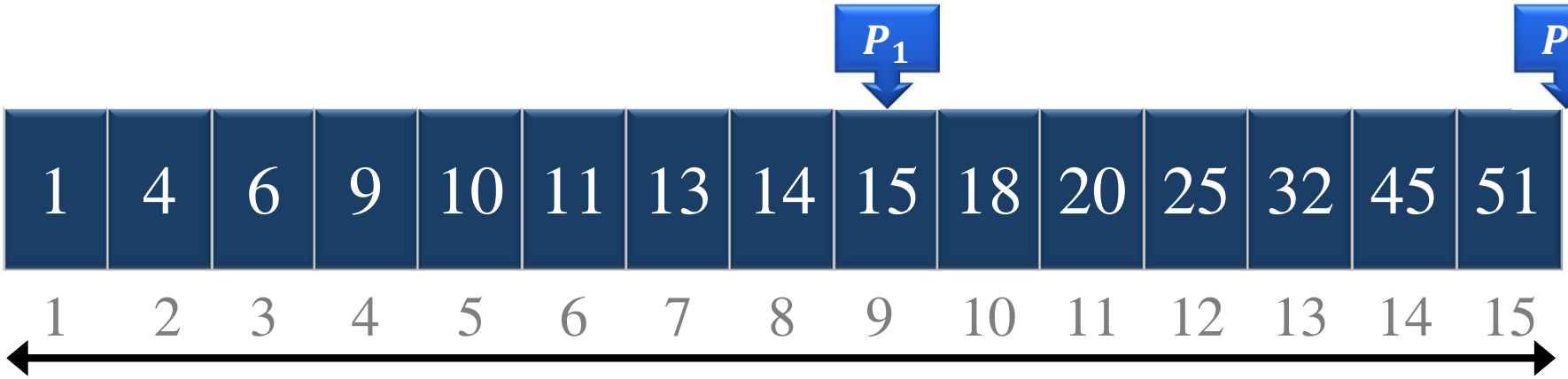
Exemplu 2



- *NN+1) i(N+1) g−1 gg−1 i(N+1) g−1 ⇒ P 1 PP P 1 1 P 1* va ocupa poziția 9 din vector
- 1
- N = 2 *(numărul de procesoare)*
  - Din $i(N+1)\ g-1 \Rightarrow P_1$ va ocupa poziția 9 din vector

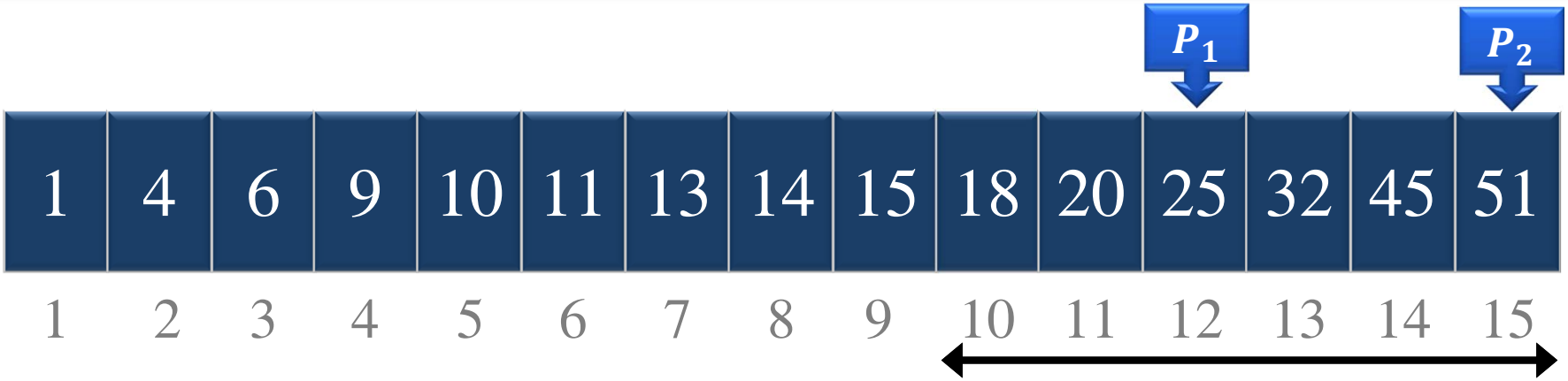
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{8}$.
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{8}$.

Exemplu 2



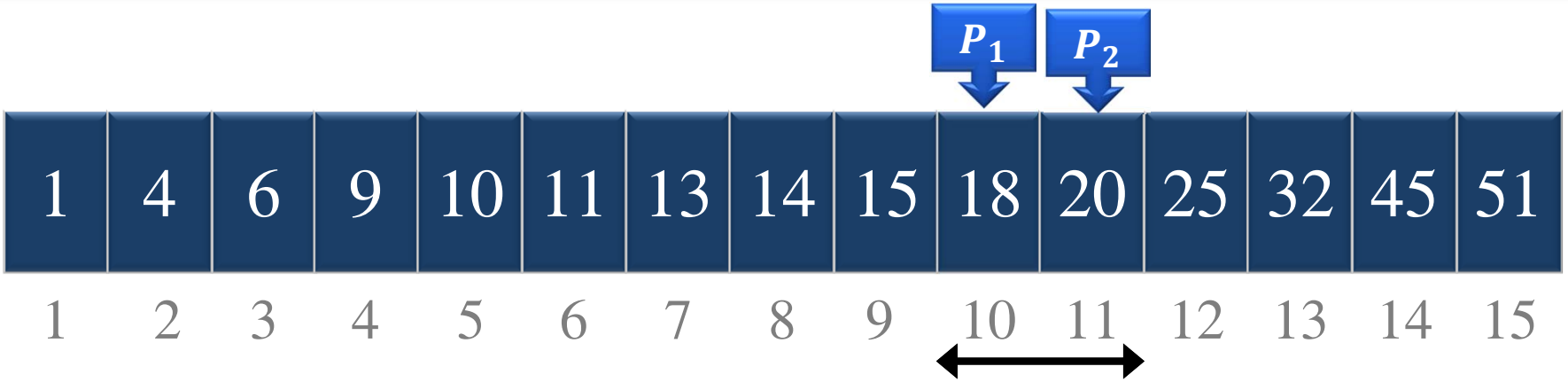
- $NN+1)\ i(N+1)\ g-1\ gg-1\ i(N+1)\ g-1 \Rightarrow\ P\ 2\ PP\ P\ 2\ 2\ P\ 2$ va ocupa poziția **18** din vector
- 2
- $NN+1)\ i(N+1)\ g-1\ gg-1\ i(N+1)\ g-1 \Rightarrow\ P\ 1\ PP\ P\ 1\ 1\ P\ 1$ va ocupa poziția 9 din vector
- 1
- Lungimea unei subsecvenţe în primul pas este $(N+1)^{g-1} = \mathbf{8}$.
- Lungimea unei subsecvenţe în P_2 va fi $(N+1)^{g-1} = \mathbf{8}$.
  - Deci ultimul element din P_2 va ocupa poziția **18** din vector

Exemplu 2



- $g = 2$
- N = 2 *(numărul de procesoare)*
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{8}$.
- Lungimea unei subsecvenţe în pasul g este $(N + 1)^{g-1} = \mathbf{8}$.
  - Deci, P_2 va poziționa **18** din vector

Exemplu 2



- $g = 1$
- N = 2 *(numărul de procesoare)*
- Lungimea unei subsecvenţe în primul pas este $(N + 1)^{g-1} = \mathbf{8}$.
- Lungimea unei subsecvenţe în pasul g este $(N + 1)^{g-1} = \mathbf{8}$.
  - Deci, P_1 va prelucra poziția 1 din vector
  - P_2 va prelucra poziția $\mathbf{18}$ din vector

Algoritm

```
/* Inițializarea indicilor secvenței de căutat */
    var q:int := 1;
    var r:int := n;

/* Inițializarea rezultatului și a numărului
 * maxim de pași */
   var c: array [0:N+1] of (stg,dr);
   var j: array [1:N] of integer;
   var k:int := 0;
   var g:int := sup(log(n+1)/log(N+1));

   c[0] := dr;
   c[N+1] := stg;
```

Algoritm (2)

```
do (q<=r) and (k=0) and (q>0) ->
    fa i := 1 to N do in parallel
        j_i := (q-1) + i * (N + 1)^(g-1)
            /* P_i compară x cu s[j_i] și determină partea de secv acceptată */
            if j_i <= r ->
               if s[j_i] = x -> k := j_i
                   [] s[j_i] > x -> c[i] := stg
                   [] s[j_i] < x -> c[i] := dr
               fi;
           [] j_i > r -> j_i := r + 1;
               c[i] := stg;
           fi;
           /* calculează indicii subsecvenței următoare */
           if c[i] <> c[i-1] -> q := j_(i-1) + 1;
                                r := j_i - 1;
           fi;
           if (i = N) and (c[i] <> c[i + 1]) -> q := j_i + 1 fi;
    af;
    g := g - 1;
od;
```

Observaţii

- Complexitatea

  $t(\mathrm{n}) = \mathrm{O}(\log_{N+1}(n+1))$ → algoritmul este *timp* optimal
- Lucrul

  $O(N\ \log_{N+1}(N+1))$ → algoritmul nu este *cost* optimal

- Notă
  - cerinţa este ca toate elementele secvenţei S să fie distincte

- În cazul în care șirul conține valori egale
  - Menţinerea eficienţei algoritmului → model CRCW.

Sumar

- Dezvoltarea aplicațiilor pentru SM SIMD
- Proprietăți dorite ale algoritmilor paraleli
- Exemplificare: căutare paralelă - justificare