فصل ۲
انتخاب موازی

سید ناصر رضوی

rezavi@iust.ac.ir

۱۳۹۰

معرفی مسأله انتخاب

- **مسأله انتخاب**:
  - **ورودی**:
    - دنباله S متشکل از n عدد و عدد صحیح k به طوری که $1 \le k \le n$
  - **خروجی**:
    - k اُمین کوچکترین عنصر در S
- مثال:
  - ورودی:
    - $S = \{8, -3, 2, -5, 6, 0\}, k = 4$
  - خروجی:
    - Answer = 2

ترتیب خطی

- **ترتیب خطی**:
  - عناصر مجموعه A دارای ترتیب خطی $<$ هستند، اگر و فقط اگر:

i. به ازای هر دو عنصر مانند a و b متعلق به A داشته باشیم:

$a < b$ یا $a = b$ یا $a > b$

ii. به ازای هر سه عنصر مانند a، b و c متعلق به A داشته باشیم:

$(a < b)$ و $(b < c) \rightarrow (a < c)$

رتبه

- **رتبه**:
  - در دنباله $S = \{s_1, s_2, \ldots, s_n\}$ که عناصر آن از یک مجموعه دارای ترتیب خطی گرفته شده است، رتبه عنصر s_i به صورت تعداد عناصر کوچکتر از s_i در S به علاوه یک تعریف می شود.

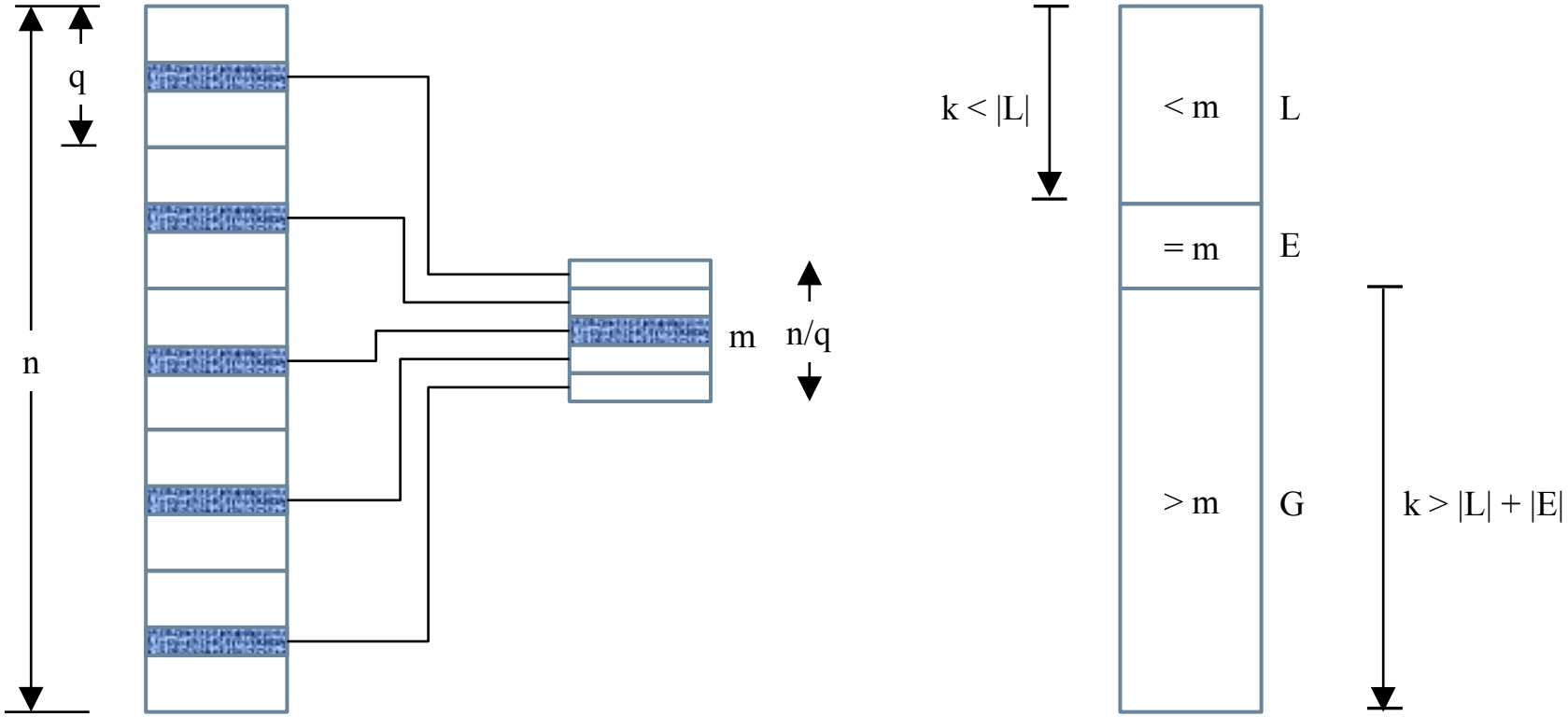
- مثال:
  - $S = \{8, -3, 2, -5, 6, 0\}$
  - پس از مرتب سازی دنباله فوق، خواهیم داشت:
  - $\{-5, -3, 0, 2, 6, 8\}$

    1 2 3 4 5 6

مسأله انتخاب: تعریف رسمی

- **انتخاب** (تعریف رسمی)
  - دنباله $S = \{s_1, s_2, \ldots, s_n\}$ که عناصر آن از یک مجموعه دارای ترتیب خطی گرفته شده اند و عدد صحیح k به طوری که $1 \leq k \leq n$ داده شده اند. می خواهیم عنصری که در S دارای رتبه k است را تعیین کنیم.
  - از این پس، عنصر دارای رتبه k را به صورت $s_{(k)}$ نشان می دهیم.
- مسأله انتخاب دارای حد پایین برابر $\Omega(n)$ با است:
  - زیرا هر عنصر S حداقل یک بار بررسی می شود.

یک الگوریتم ترتیبی



$$max(|L|, |G|) \leq 3n/4$$

یک الگوریتم ترتیبی

procedure SEQUENTIAL-SELECT(S, k)

step 1: **if** $|S| < q$ **then** sort S and **return** the kth element directly

else subdivide S into $|S|/q$ subsequences of q elements each

step 2: sort each subsequence and determine its median

step 3: call SEQUENTIAL-SELECT to find *m*, the median of the $|S|/q$ medians of step 2

step 4: create three subsequences S_1, S_2, and S_3 of elements of S smaller than, equal to, and larger than *m*, respectively

step 5: **if** $|S_1| \geq k$ **then return** SEQUENTIAL-SELECT(S_1, k)

else if $|S_1| + |S_2| \geq k$ **then return** m

else return SEQUENTIAL-SELECT(S_3, $k - |S_1| - |S_2|$)

تحلیل پیچیدگی الگوریتم ترتیبی

$T(n) = T(n/q) + T(3n/4) + cn$

- $T(n) = O(n)$
- در نتیجه الگوریتم ترتیبی ارائه شده از نظر زمانی بهینه است.

مثال: الگوریتم انتخاب ترتیبی


```
S    6 4 5 6 7   1 5 3 8 2   1 0 3 4 5   6 2 1 7 1   4 5 4 9 5
     ---------   ---------   ---------   ---------   ---------
T        6           3           3           2           5
m                                3

     1 2 1 0 2 1 1    3 3    6 4 5 6 7 5 8 4 5 6 7 4 5 4 9 5
     -------------    ---    -------------------------------
          L             E                     G
         |L| = 7       |E| = 2               |G| = 16
```

To find the 5th smallest element in *S*, select the 5th smallest element in *L*

```
S    1 2 1 0 2   1 1
     ---------   ---
T        1         1
m                  1
     0    1 1 1 1    2 2
     -    -------    ---
     L       E        G
```

Answer: 1

ویژگی های مطلوب یک الگوریتم موازی

- **تعداد پردازنده ها $p(n)$:**
  - **تعداد پردازنده ها باید کمتر از n باشد:**
    - $p(n)$ باید یک تابع زیرخطی از n باشد مانند n^x به طوری که $0 < x < 1$.
  - **تعداد پردازنده ها باید تطبیقی باشد:**
    - یک الگوریتم موازی باید خود را با تعداد پردازنده های موجود تطبیق دهد.
- **زمان اجرا $t(n)$:**
  - **زمان اجرا باید کم باشد:**
    - یک الگوریتم موازی باید به طور قابل ملاحظه ای از بهترین الگوریتم ترتیبی نظر سریع تر باشد.
  - **زمان اجرا باید تطبیقی باشد:**
    - زمان اجرای $t(n)$ باید با تعداد پردازنده ها رابطه عکس داشته باشد (یعنی، کاهش زمان اجرا با افزایش تعداد پردازنده ها)
- **هزینه $c(n)$:**
  - هزینه باید با حد پایین تعداد عملیات ترتیبی مورد نیاز در بدترین حالت مطابقت نماید. یعنی الگوریتم موازای باید هزینه بهینه باشد.

ویژگی های مطلوب یک الگوریتم موازی

- مثال: الگوریتم انتخاب موازی همه این ویژگی های مطلوب را دارد:

- $p(n) = O(n^{1-x})$,
- تعداد پردازنده ها یک تابع زیرخطی از n و تطبیقی است.
- $t(n) = O(n^{x})$,
- الگوریتم موازی بسیار سریعتر از الگوریتم ترتیبی است و همچنین زمان اجرای الگوریتم موازی با افزایش تعداد پردازنده ها کاهش می یابد.
- $c(n) = t(n) \times p(n) = O(n)$
- الگوریتم موازی هزینه بهینه است.

دو رویه مفید در الگوریتم انتخاب موازی

procedure BROADCAST(D, N, A)

Step 1: Processor P_1 ,

(i) reads the value in D

(ii) stores it in its own memory

(iii) writes it in A(1)

Step 2: **for** i = 0 **to** (Log N – 1) **do**

for j = $2^i + 1$ **to** 2^{i+1} **do in parallel**

Processor P_j

(i) reads the value in A(j – 2^i)

(ii) stores it in its own memory

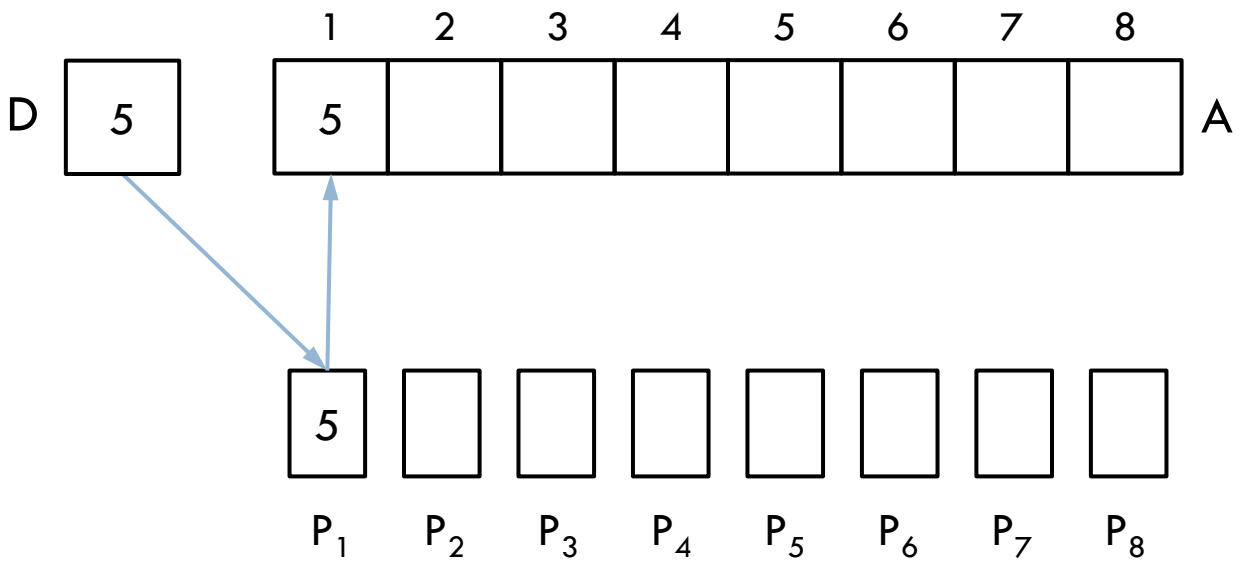
(iii) writes it in A(j)

end for

end for.

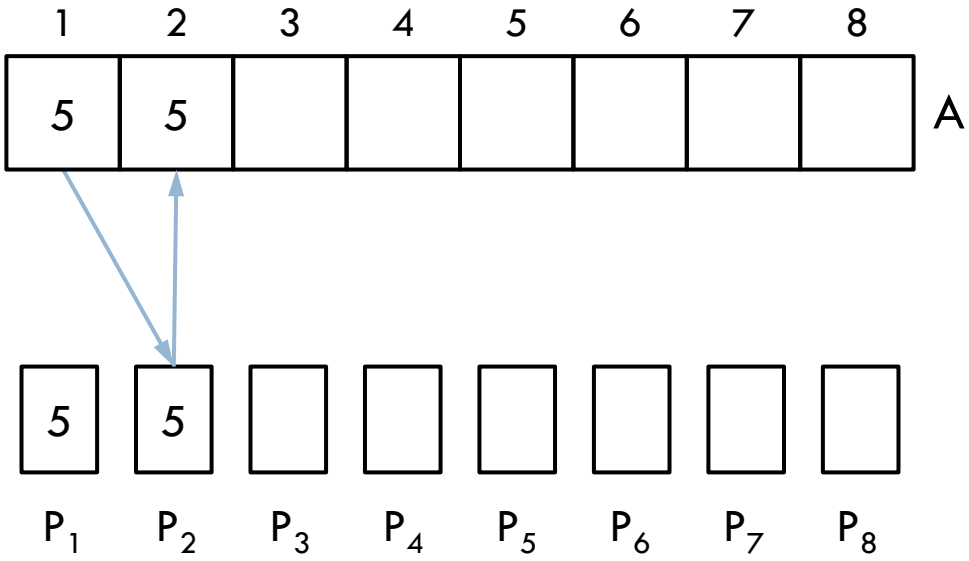
رویه انتشار

STEP 1



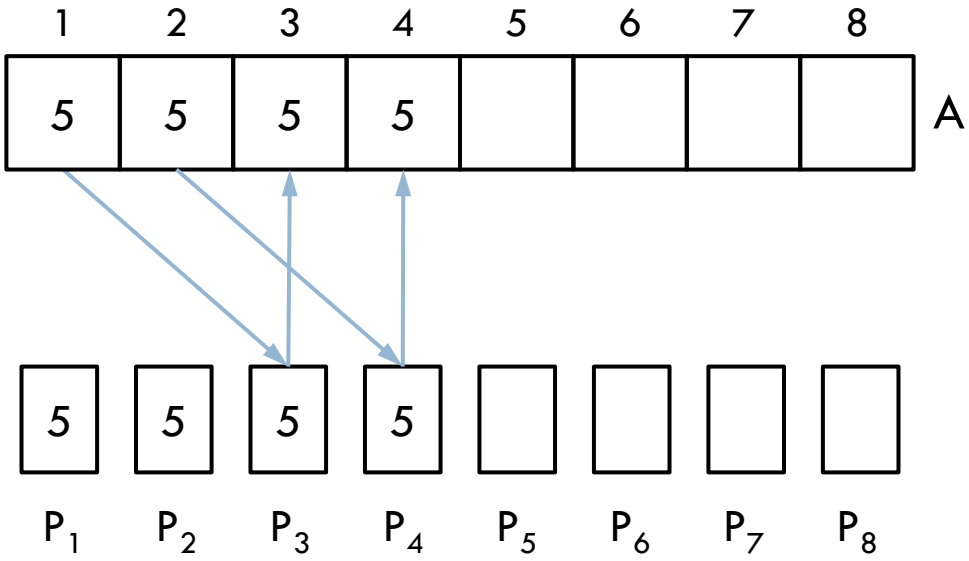
رویه انتشار

STEP 2 (i = 0)



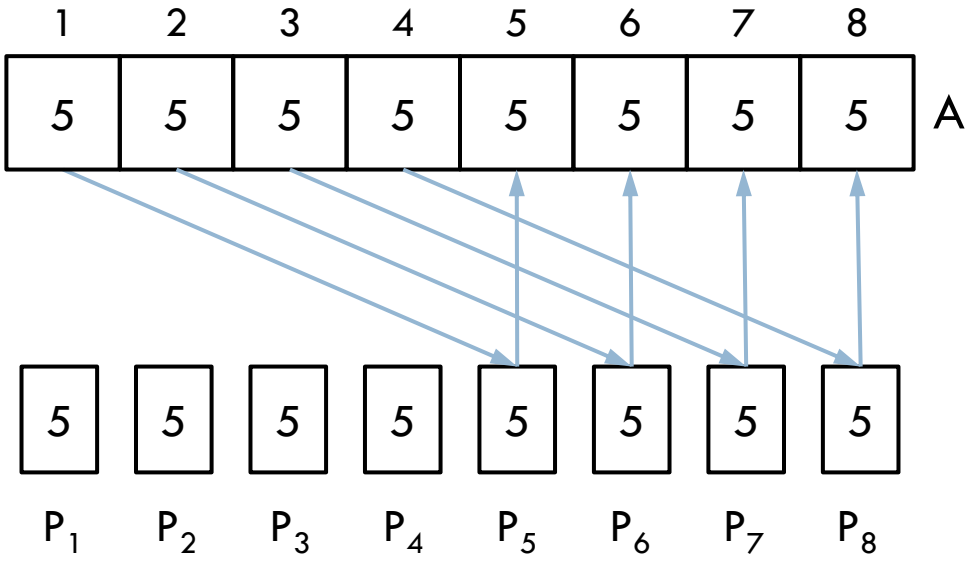
رویه انتشار

STEP 2 (i = 1)



رویه انتشار

STEP 2 (i = 2)



محاسبه پیشوند مجموع

- فرض می کنیم پردازنده P_i در حافظه محلی خود مقدار عددی a_i را نگهداری می کند (به ازای $1 \le i \le N$)

- اکنون می خواهیم به ازای هر پردازنده مانند P_i مجموع زیر را محاسبه کنیم:

$$a_1 + a_2 + \ldots + a_i$$

- **ایده**: مشغول نگاه داشتن پردازنده ها تا حد ممکن و استفاده از خاصیت شرکت پذیری در عمل جمع

محاسبه پیشوند مجموع

procedure ALLSUMS(a_1, a_2, … , a_n)

  for j = 0 **to** Log N – 1 **do**

    for i = 2^j + 1 **to** N **do** in **parallel**

      Processor P_i

        (i) obtains a_{i-2^j} from P_{i-2^j} , through shared memory

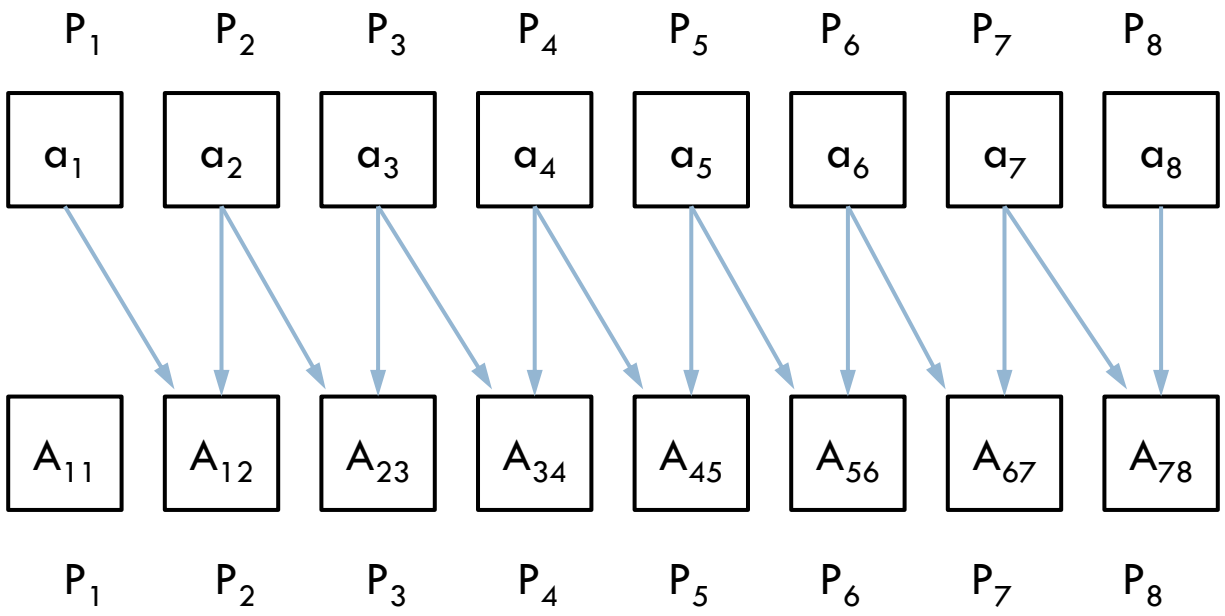
        (ii) replaces a_i with $a_{i-2^j} + a_i$

    end for

  end for.

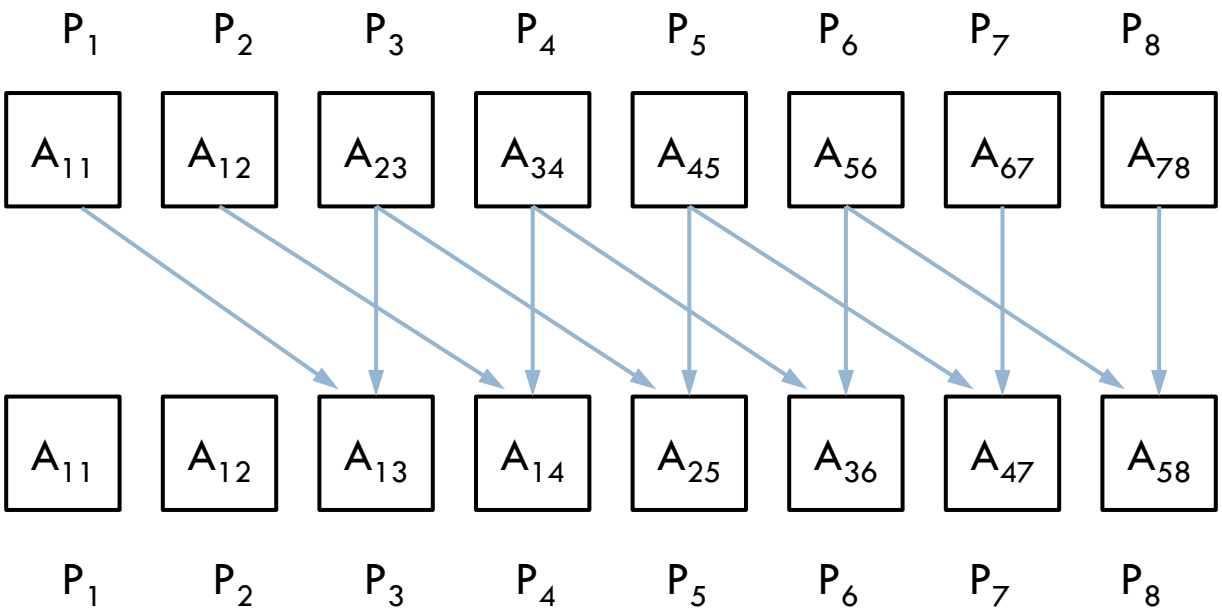
محاسبه پیشوند مجموع

$(j = 0)$



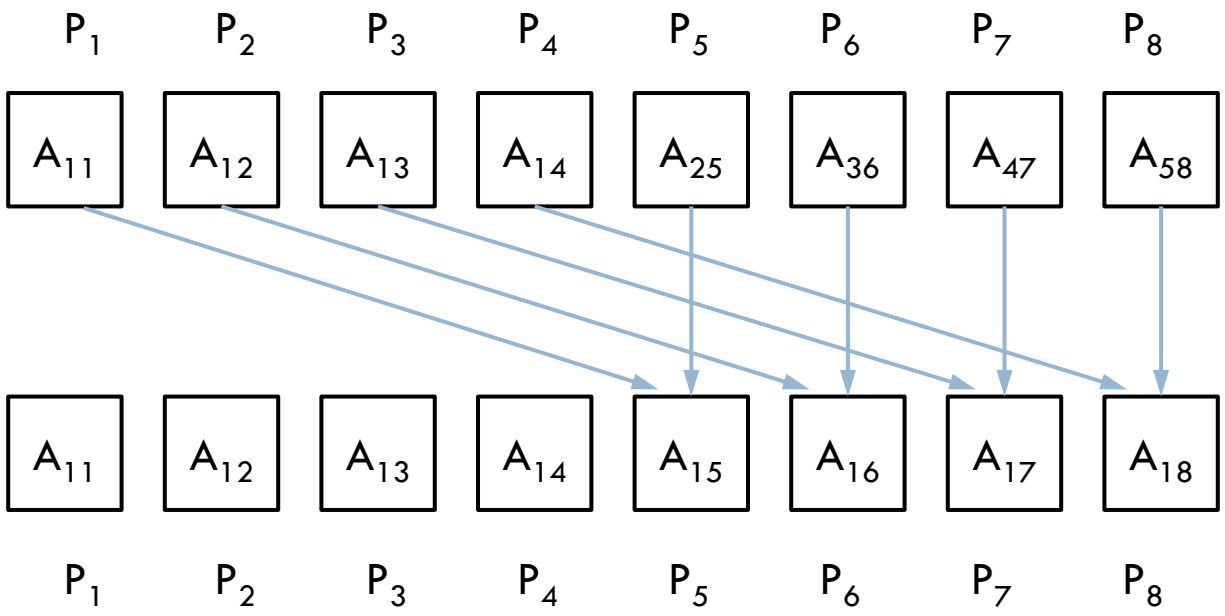
محاسبه پیشوند مجموع

$(j = 1)$



محاسبه پیشوند مجموع

$(j = 2)$



محاسبه پیشوند مجموع

- در الگوریتم قبل به جای عمل جمع از هر عمل شرکت پذیر دیگری نیز می توان استفاده نمود.

الگوریتم انتخاب موازی

procedure PARALLEL-SELECT(S, k)

Step 1: **if** $|S| \leq 4$ **then** P_1 uses at most 5 comparisons to return the kth element

$O(\log n)$ **else**

(i) S is subdivided into $|S|^{1-x}$ subsequences S_i of length $|S|^x$ each, where $1 \leq i \leq |S|^{1-x}$

(ii) subsequence S_i is assigned to processor P_i

end if

Step 2: **for** i = 1 **to** $|S|^{1-x}$ **do** in **parallel**

$O(n^x)$ (2.1) SEQUENTIAL-SELECT(S_i, $\lceil |S_i|/2 \rceil$)

(2.2) P_i stores m_i in M(i)

end for

الگوریتم انتخاب موازی (ادامه)

Step 3: PARALLEL-SELECT(M, ⌈|M|/2⌉)

$T(n^{1-x})$

Step 4: The sequence S is subdivided into three subsequences

$L = \{s_i \in S;\ s_i < m\}$

$E = \{s_i \in S;\ s_i = m\}$

$G = \{s_i \in S;\ s_i > m\}$

$O(n^x)$

Step 5: **if** $|L| \geq k$ **then** PARALLEL-SELECT(L, k)

else if $|L| + |E| \geq k$ **then return** m

else PARALLEL-SELECT(G, k - |L| - |E|)

end if

end if

$T(3n/4)$

توضیح مرحله ۴

- مرحله ۴:
  - هر پردازنده مقدار m را دریافت می کند (الگوریتم انتشار)
  - هر پردازنده به صورت موازی با پردازنده های دیگر، زیرآرایه خود را افراز می کند.
  - پس از افراز به ازای هر پردازنده مانند i سه لیست خواهیم داشت:
    - L_i, E_i, G_i,
  - با ادغام L_i ها لیست L، با ادغام E_i ها لیست E و با ادغام G_i ها لیست G به دست می آید.
  - برای ادغام هر یک از این لیست ها با یکدیگر از الگوریتم پیشوند مجموع استفاده می شود.

الگوریتم انتخاب موازی: تحلیل

$T(n,p) = T(n^{1-x},p) + T(3n/4,p) + cn^x$

پاسخ برابر $O(n^x)$ است. با استفاده از جایگزینی پاسخ قابل تصدیق است

Speedup = $\Theta(n) / O(n^x) = \Omega(n^{1-x}) = \Omega(p)$

Efficiency = $\Omega(1)$

Cost(n, p) = $pT(n, p) = \Theta(n^{1-x})\ \Theta(n^x) = \Theta(n)$

$p = O(n^{1-x})$

What happens if we set x to 1? (i.e., use one processor)

$T(n, 1) = O(n^x) = O(n)$

What happens if we set x to 0? (i.e., use n processors)

$T(n, n) = O(n^x) = O(1)$?

خیر، زیرا در تحلیل مجانبی چندین جمله با مرتبه لگاریتمی در مقایسه با $O(n^x)$ حذف شدند.

الگوریتم انتخاب موازی: مثال (n = 29, k = 21)

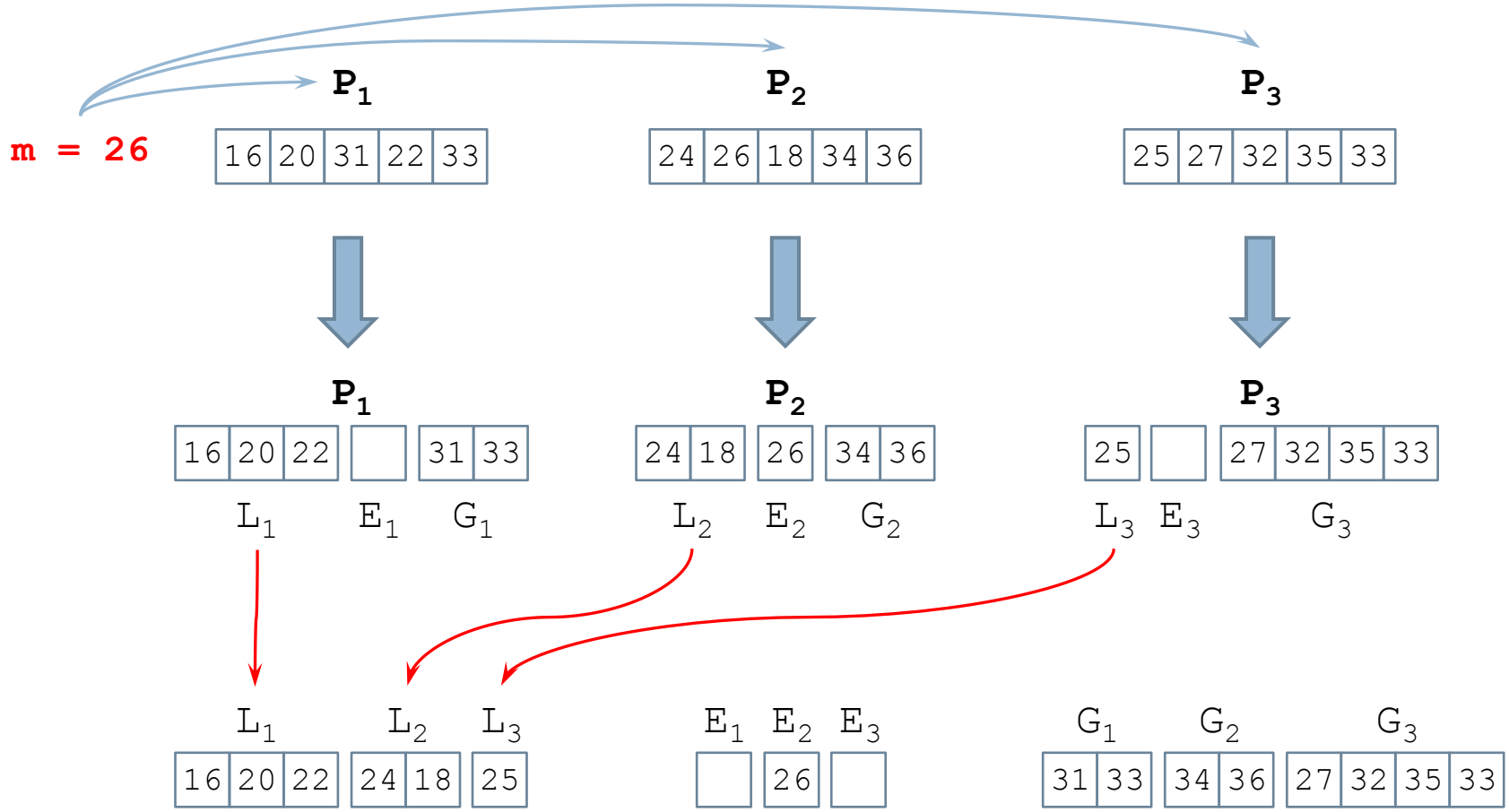

```
S   3 14 16 20 8 31 22 12 33 1 4 9 10 5 13 7 24 2 14 26 18 34 36 25 14 27 32 35 33

S   3 14 16 20 8 31  22 12 33 1 4 9  10 5 13 7 24 2 14 26 18 34 36 25 14 27 32 35 33
    ------------- ------------- ------------ -------------- -----------
M          14                9                7                 25              32
m                                            14

  3 8 12 1 4 9 10 5 13 7 2   14 14 14   16 20 31 22 33 24 26 18 34 36 25 27 32 35 33
 --------------------      -------   ------------------------------------
             L                  E                              G
            |L| = 11           |E| = 3                       |G| = 15
```

To find the 21th smallest element in *S*, select the 7th smallest element in *G*

```
S   16 20 31 22 33      24 26 18 34 36      25 27 32 35 33
    --------------      --------------      --------------
M         22                 26                  32
m                            26
    16 20 22 24 18 25     26     31 33 34 36 27 32 35 33
    -----------------     --     -----------------------
          L                E                   G
         |L| = 6          |E| = 1            |G| = 8
```

Answer: 26

مثال: مرحله ۴


m = 26
P1
16 20 31 22 33
P2
24 26 18 34 36
P3
25 27 32 35 33
P1
16 20 22 31 33
L1 E1 G1
P2
24 18 26 34 36
L2 E2 G2
P3
25 27 32 35 33
L3 E3 G3
L1 L2 L3
16 20 22 24 18 25
E1 E2 E3
26
G1 G2 G3
31 33 34 36 27 32 35 33

مثال: ادامه مرحله ۴

L_1			L_2		L_3
16	20	22	24	18	25

E_1	E_2	E_3
	26	

G_1		G_2		G_3			
31	33	34	36	27	32	35	33

محاسبه پیشوند مجموع $a_1=3$ $a_2=2$ $a_3=1$

$z_0=0$ $z_1=3$ $z_2=5$ $z_3=6$

16	20	22	24	18	25

L

26

E

31	33	34	36	27	32	35	33

G

الگوریتم انتخاب موازی: مثال

- مثال قبل بر روی یک EREW SM SIMD با ۵ پردازنده اجرا شده است.
- بنابراین:

$$|S|^{1-x} = 5 \Rightarrow 1 - x = 0.47796$$

- در تکرار دوم تعداد پردازنده ها به صورت زیر محاسبه می شود:

$$|G|^{1-x} = 15^{0.47796} = 3.6485$$