

Edényrendező, Radix sort, Összefésülő rendező

10. előadás

Edényrendezés

Edényrendezések

- Tegyük fel, hogy tudjuk, hogy a bemenő elemek ($A[1..n]$ elemei) egy m elemű U halmazból kerülnek ki
 - Például $\forall A[i]$ -re igaz, hogy $A[i] \in [1..m]$.
- Lefoglalunk egy U elemeivel indexelt B tömböt (m db ládát), először mind üres
- B segédtömb elemei lehetnek láncolt listák például

Edényrendezések

- Első fázis: végigolvassuk az A -t, és az $s = A[i]$ elemet a $B[s]$ lista végére fűzzük.
- Például tegyük fel, hogy a rendezendő $A[1..7]$ tömb elemei 0 és 9 közötti egészek

- A:

5	3	1	5	6	9	6
---	---	---	---	---	---	---

- B:

	1		3		5	6			9	
					5	6				

Edényrendezések

- Második fázis: elejétől a végéig növvő sorrendben végigmegyünk B -n, és a $B[i]$ listák tartalmát visszaírjuk A -ba.

- B :

	1		3		5	6			9	
					5	6				

- A :

1	3	5	5	6	6	9
---	---	---	---	---	---	---

Edényrendezések

- Lépésszám:
 - B létrehozása: $\mathcal{O}(m)$,
 - első fázis $\mathcal{O}(n)$,
 - második fázis $\mathcal{O}(n + m)$, összesen $\mathcal{O}(n + m)$.
- Ez gyorsabb, mint az általános alsó korlát, ha:
 - $m \leq c * n$

Edényrendezések

- Általánosabban:
 - Legyen K az S sorozat elemeinek típusértékhalmaza, $\varphi: K \rightarrow [0 \dots M - 1]$ olyan függvény, amire igaz, hogy ha $\varphi(k_1) < \varphi(k_2)$, akkor $k_1 < k_2$.
 - Legyenek $E_0, E_1, \dots, E_{M-1}$ edények, melyek éppen olyan sorozatok, mint S .
 - Az egyes edényekben megmarad az S -beli elemek ottani relatív sorrendje.

Edényrendezések

- Edényrendező(S)

$E_0, E_1, \dots, E_{M-1} \leftarrow \varepsilon, \varepsilon, \dots, \varepsilon$	
$S \neq \varepsilon$	
	Out(S, x) In($E_{\varphi(x)}, x$)
$S \leftarrow \text{Összefűzés}(E_0, E_1, \dots, E_{M-1})$	

Edényrendezések

- Ahhoz, hogy $\text{Edényrendező}(S)$ egyszeri szétrakással rendezzen, elégséges, ha:
 - minden edényben legfeljebb egy elem van, $(\forall i \in [0, M - 1]: |E_i| \leq 1)$ vagy:
 - az egyes edényekben csak azonos elemek vannak, vagy:
 - az egyes edények rendezettek.
- Ha az előző feltételek valamelyike teljesül, akkor **tökéletes** az edényrendezés.
 - Ennek műveletigénye: $\Theta(n)$, ahol $n = |S|$.
 - Példa: nagyon sok embert kell magasság szerint sorba rakni
 - megéri minden egyes testmagasság számára egy edényt létrehozni

Radix rendezés

- Tegyük fel, hogy a kulcsok összetettek, több komponensből állnak, $(t_1, \dots, t_k)$ alakú szavak, ahol a t_i komponens az L_i rendezett típusból való, a rendezés lexikografikus
- Példa: Legyen $(U; <)$ a huszadik századi dátumok összessége az időrendnek megfelelő rendezéssel:
 - $L_1 = \{1900; 1901; \dots; 1999\}$ $n_1 = 100$
 - $L_2 = \{\text{január, február, } \dots, \text{december}\}$ $n_2 = 12$
 - $L_3 = \{1; 2; \dots; 31\}$ $n_3 = 31$
 - A dátumok rendezése éppen az L_i típusokból származó lexikografikus rendezés lesz

Radix rendezés

1. Rendezzük a sorozatot az utolsó, a k -adik komponensek szerint edényrendezéssel!
 2. A kapott eredményt rendezzük a $k - 1$ -edik komponensek szerint edényrendezéssel, stb.
- Fontos, hogy az edényrendezésnél az elemeket a ládában mindig a lista végére tettük.
 - Így, ha két azonos kulcsú elem közül az egyik megelőzi a másikat, akkor a rendezés után sem változik a sorrendjük
 $\Rightarrow$ **stabil rendezés**.

Radix rendezés

- Miért működik a radix jól?
 - Ha $X < Y$, az első $i - 1$ tag megegyezik, de $x_i < y_i$, akkor az i -edik komponens rendezésekor X előre kerül.
 - Stabil rendezés $\Rightarrow$ később már nem változik a sorrendjük

Radix rendezés

- Példa:

1969. jan. 18. 1969. jan. 1. 1955. dec. 18. 1955. jan. 18. 1918. dec. 18.

1. menet után:

1969. jan. 1. 1969. jan. 18. 1955. dec. 18. 1955. jan. 18. 1918. dec. 18.

2. menet után:

1969. jan. 1. 1969. jan. 18. 1955. jan. 18. 1955. dec. 18. 1918. dec. 18.

3. menet után:

1918. dec. 18. 1955. jan. 18. 1955. dec. 18. 1969. jan. 1. 1969. jan. 18.

Radix rendezés

- Példa – ha fordítva lenne

1969. jan. 18. 1969. jan. 1. 1955. dec. 18. 1955. jan. 18. 1918. dec. 18.

1. menet után:

1918. dec. 18. 1955. dec. 18. 1955. jan. 18. 1969. jan. 18. 1969. jan. 1.

2. menet után:

1955. jan. 18. 1969. jan. 18. 1969. jan. 1. 1918. dec. 18. 1955. dec. 18.

3. menet után:

1969. jan. 1. 1955. jan. 18. 1969. jan. 18. 1918. dec. 18. 1955. dec. 18.

- Helytelen a sorrend

Radix rendezés

- Általánosan
 - Az $e = e_d e_{d-1} \dots e_2 e_1$ számot jobbról balra, az alacsony helyiértékek felől indulva pozíciónként szétrakja edényekbe, majd összefűzi az edények tartalmát
 - Az i . pozíción a φ_i függvényt alkalmazzuk: $\varphi_i(e) = e_i$
 - Az i . pozíción végrehajtott szétrakás és összefűzés után S „ i -rendezett” lesz.

Radix rendezés

- Definíció
 - S „ i -rendezett” (jelölés: $x \leq_i y$), ha minden
 - $x = x_d x_{d-1} \dots x_2 x_1$ és $y = y_d y_{d-1} \dots y_2 y_1$ -ra: $x <_0 y$
 - $x \leq_i y \Leftrightarrow x_i < y_i$ vagy $x_i = y_i$ és $x <_{i-1} y$ ($i > 0$)
 - Ekkor a „ d -rendezés” a közönséges rendezés
- Hatékonyság:
 - $2 * d$ -szer megyünk végig az S sorozaton, így
$$T(n) = \Theta(d * |S|)$$

Radix rendezés

- Radix(S)

$i \leftarrow 1$	
$i < d$	
	$\text{Edenyrendezo}(S, \varphi_i)$
	$i \leftarrow i + 1$

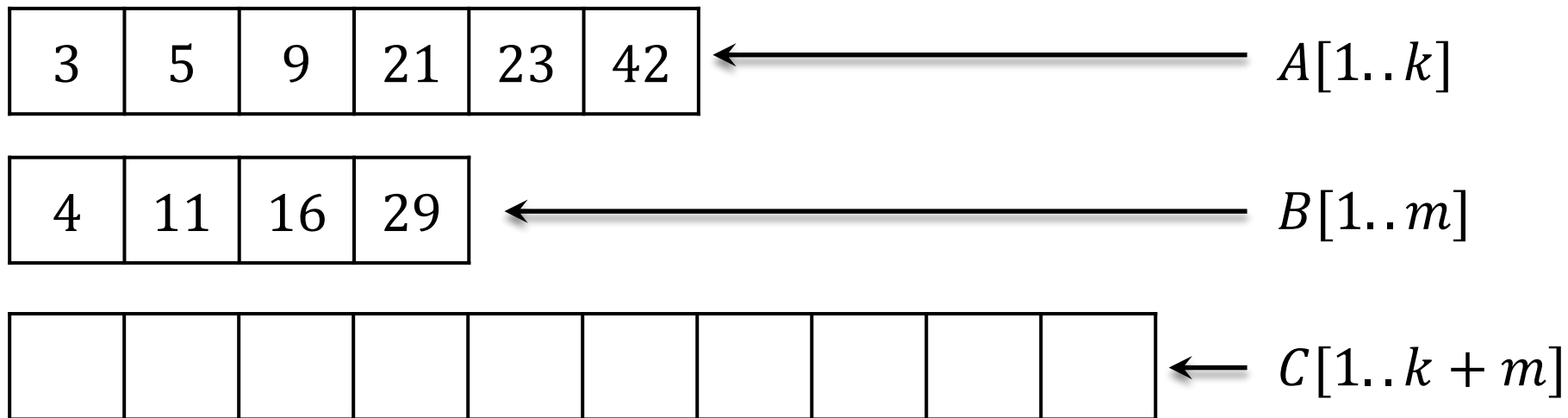
Radix rendezés

- Szokásos implementáció:
 - S fejelemes láncolt lista
 - Az edényeket egy „fej” és egy „vége” mutató ábrázolja.
 - A szétrakás és az összefűzés az elemek láncolásával megoldható.
 - Összefűzéskor nem kell az egyes edények részlistáit végigolvasni, hanem egy darabban lehet őket láncolni.

Összefuttatásos rendezés

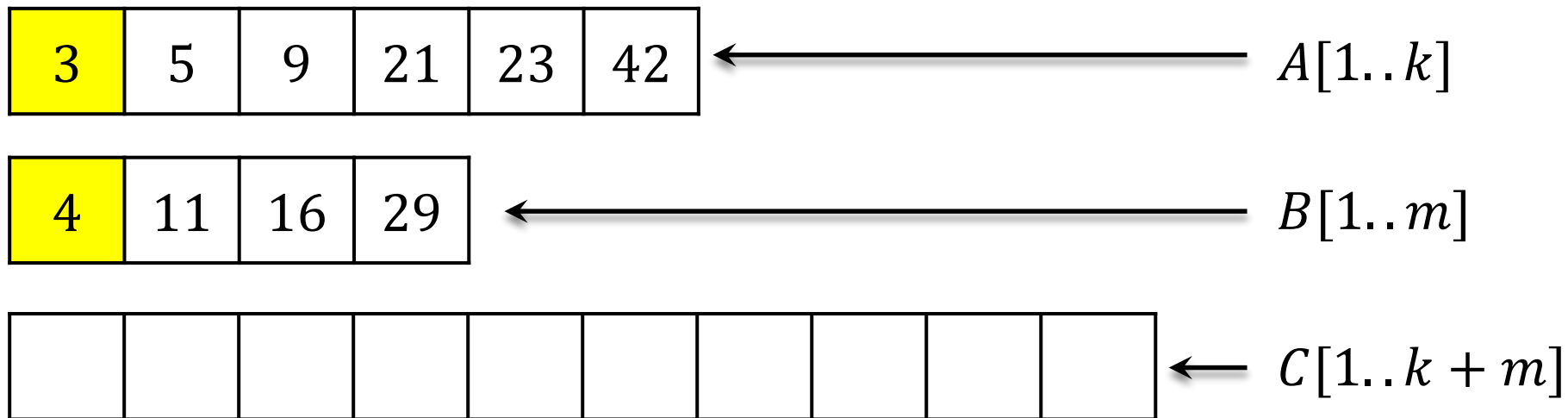
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



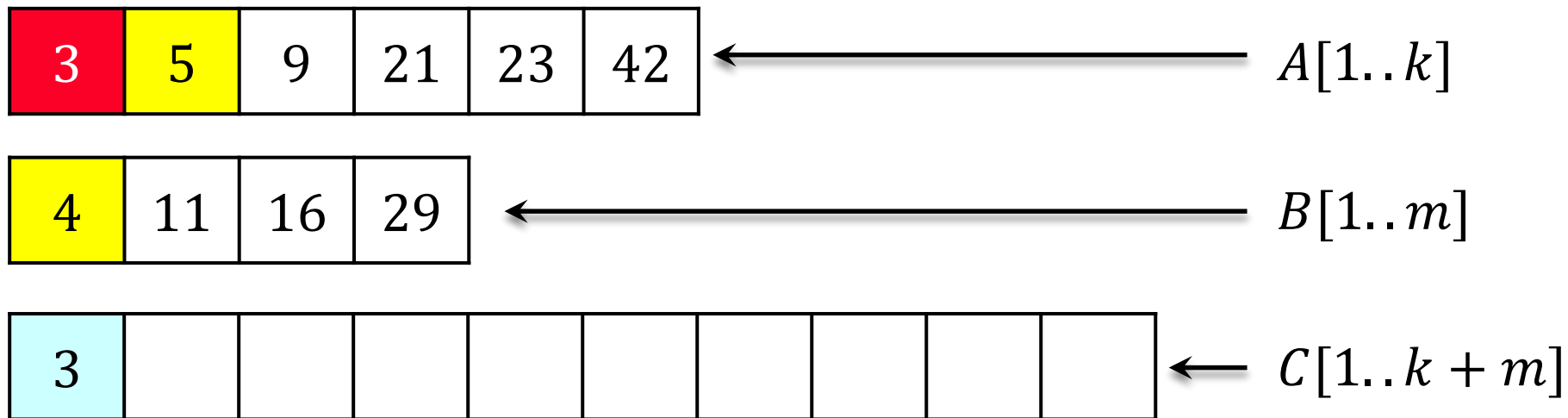
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



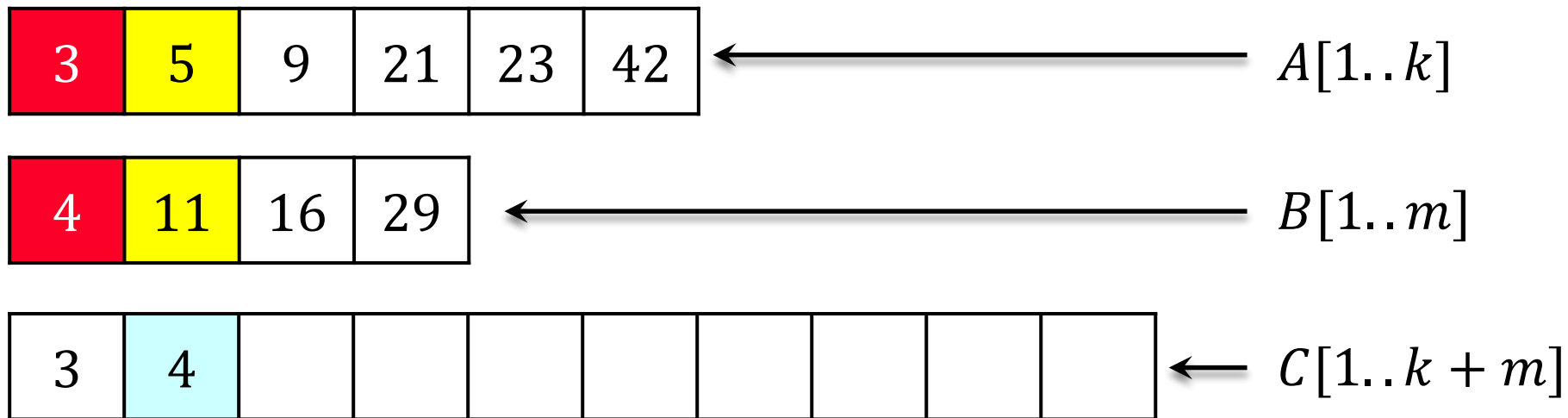
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



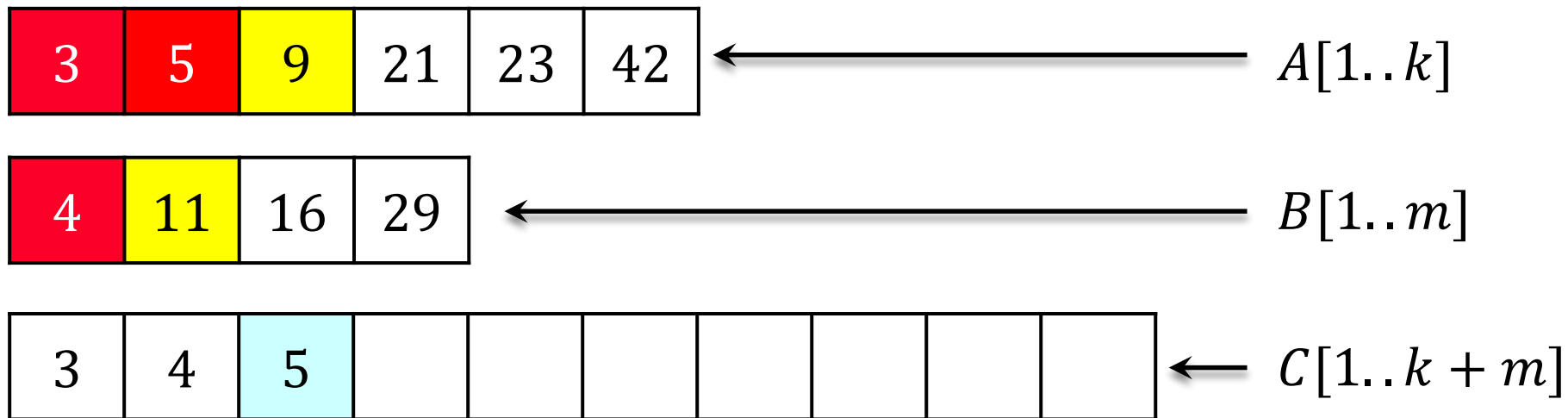
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



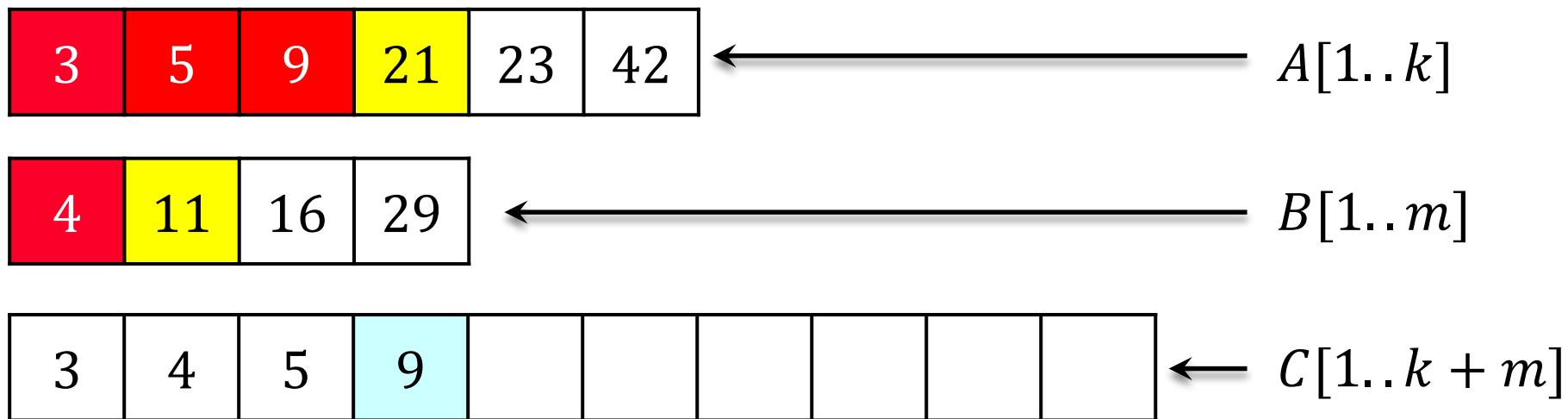
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



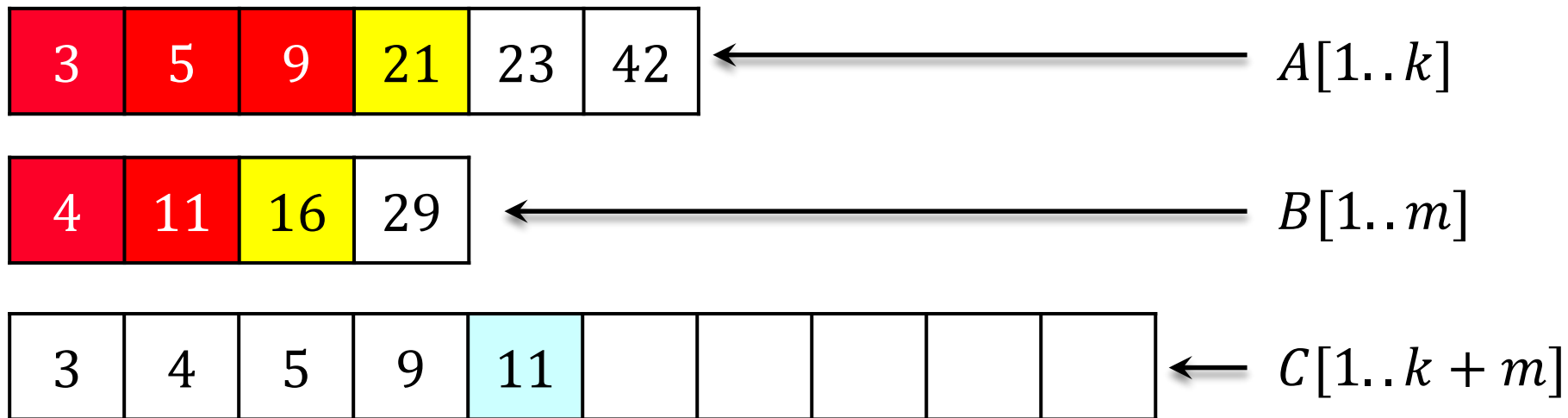
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



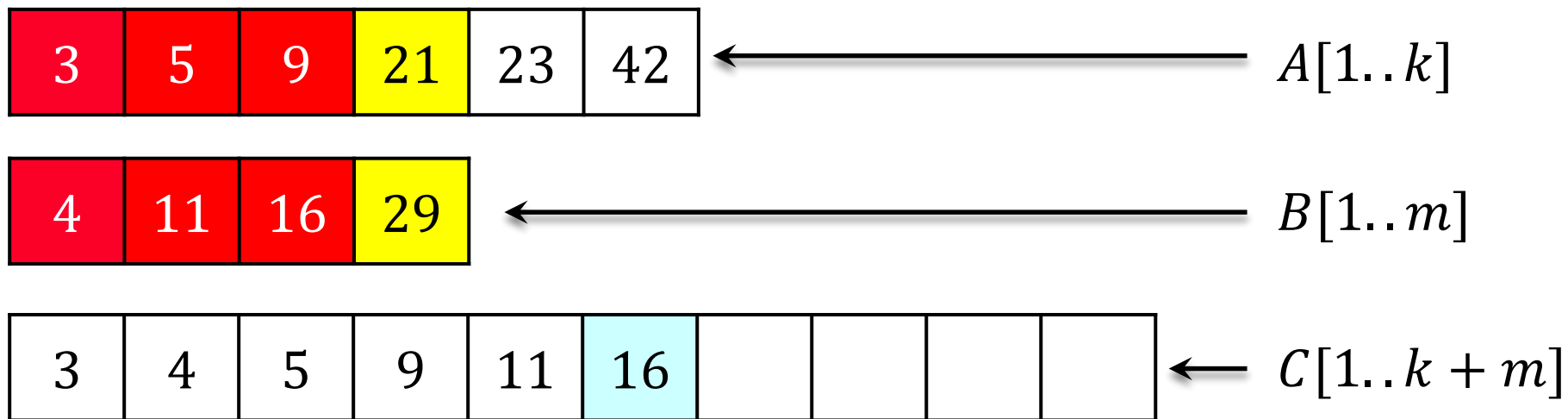
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



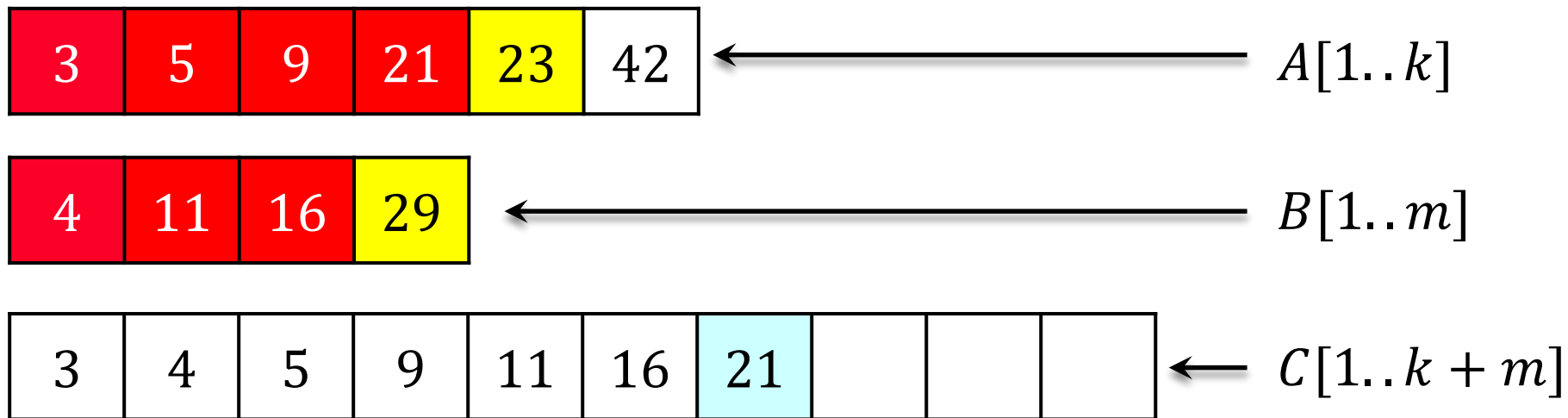
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



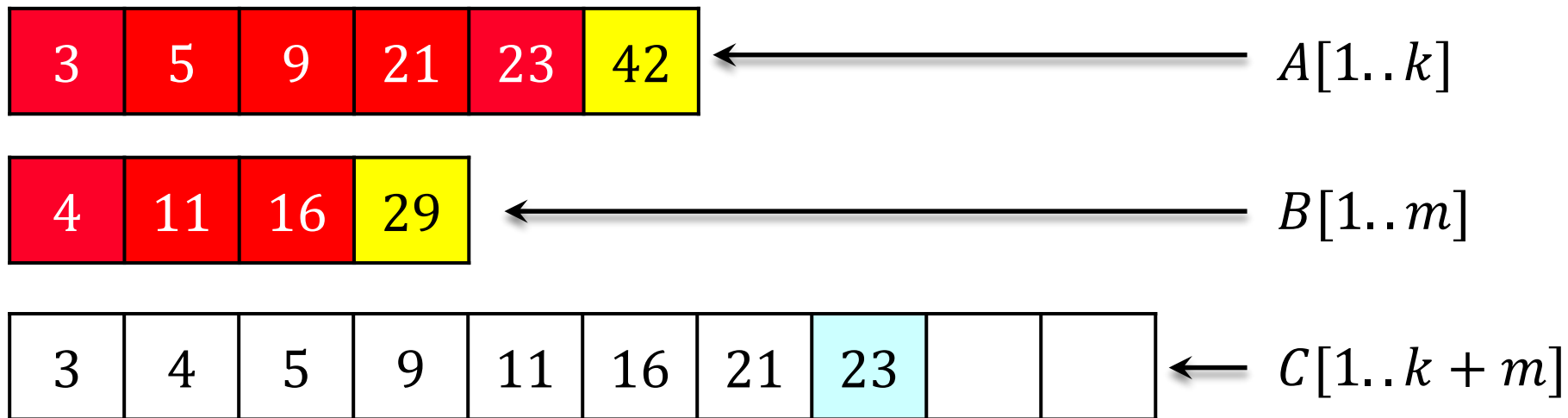
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



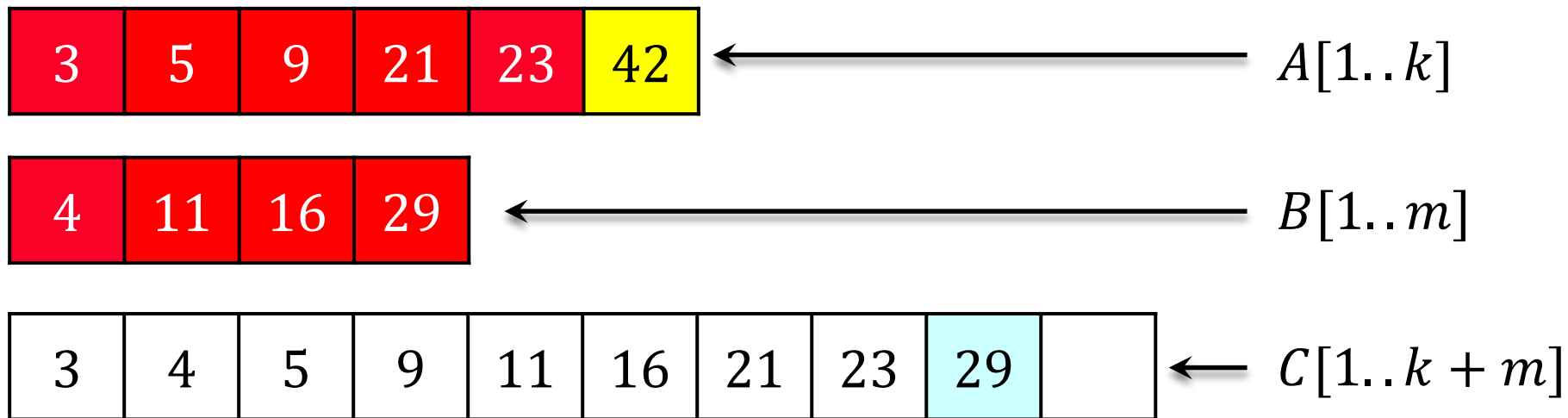
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



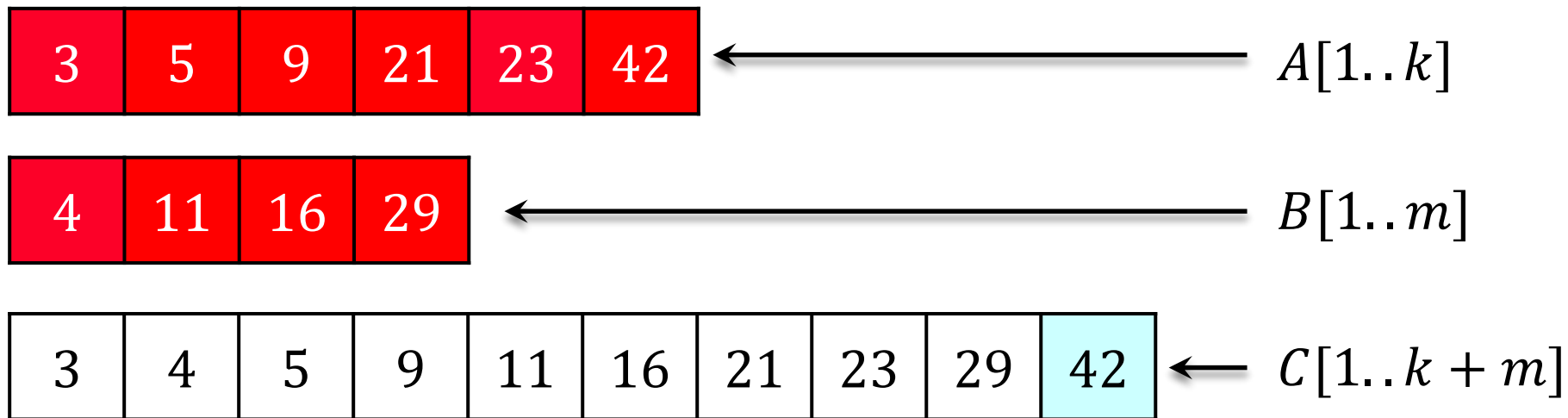
Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



Összefuttatásos rendezés

- Alap: két rendezett sorozat összefuttatása
- Legyen $A[1..k]$ és $B[1..m]$ két rendezett vektor, ekkor:



Összefuttatás algoritmus

- $\text{Összefuttat}(A, B, C)$

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

Összefuttatás algoritmus

- Összefuttat(A, B, C)

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

Összefuttatás algoritmus

- Összefuttat(A, B, C)

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

Összefuttatás algoritmus

- Összefuttat(A, B, C)

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

Összefuttatás algoritmus

- Összefuttat(A, B, C)

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

Összefuttatás algoritmus

- Összefuttat(A, B, C)

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

Összefuttatás algoritmus

- $\text{Összefuttat}(A, B, C)$

$ai \leftarrow 1, bi \leftarrow 1, ci \leftarrow 1$		
$ai \leq k \wedge bi \leq m$		
$A[ai] < B[bi]$	$A[ai] = B[bi]$	$A[ai] > B[bi]$
$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$	$C[ci] \leftarrow A[Ai]$ $ai \leftarrow ai + 1$ $ci \leftarrow ci + 1$ $C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$	$C[ci] \leftarrow B[Bi]$ $bi \leftarrow bi + 1$
$ci \leftarrow ci + 1$		
C tömbbe tegyük a maradékot		

- Ez egy külön ciklus

Összefuttatásos rendezés

- Összefuttatásnál
 - összehasonlítások száma: $k + m - 1$
- RENDEZÉS:
 - Az üres és az egyelemű sorozat biztosan rendezett.
 - $S[1..n]$ ($n > 1$) tömböt ha rendezni szeretnénk:
 - szétvágjuk két részre (S_1, S_2) – elfelezzük
 - ezeket külön-külön rendezzük
 - összefuttatjuk
 - MergeSort-nak is hívják

Összefuttatásos rendezés

- MergeSort(S)

Length(S) > 1	
Szétvág(S, S_1, S_2)	SKIP
MergeSort(S_1) MergeSort(S_2)	
Összefuttat(S_1, S_2, S)	

Összefuttatásos rendezés – tömbre

- MergeSort(A, k, v)

$k < v$	
$h \leftarrow \frac{k + v}{2}$ Szétvág(S, S_1, S_2, h)	SKIP
MergeSort(A, k, h) MergeSort($A, h + 1, v$)	
Összefuttat($A[k, h], A[h + 1, v], A[k, v]$)	

- A külső hívás
 - MergeSort($A, 1, n$)
- Az összefuttatásnál szükséges egy segédtömb használata

Összefuttatásos rendezés

- Tömbökre szokás egy másik, iteratív stratégiát is követni:
 - Minden lépésben végigmegyünk rajta egyre növekvő $h = 1, 2, 2^2, 2^3, \dots, 2^{\lceil \log_2 n \rceil - 1}$ lépésközzel, és az ilyen hosszú szomszédos tömbrészeket fésüljük össze, egy segédtömböt használva
- Példa ...

Összefuttatásos rendezés

8	2	3	11	4	5	9	7	1	6	10
---	---	---	----	---	---	---	---	---	---	----

2	8	3	11	4	5	7	9	1	6	10
---	---	---	----	---	---	---	---	---	---	----

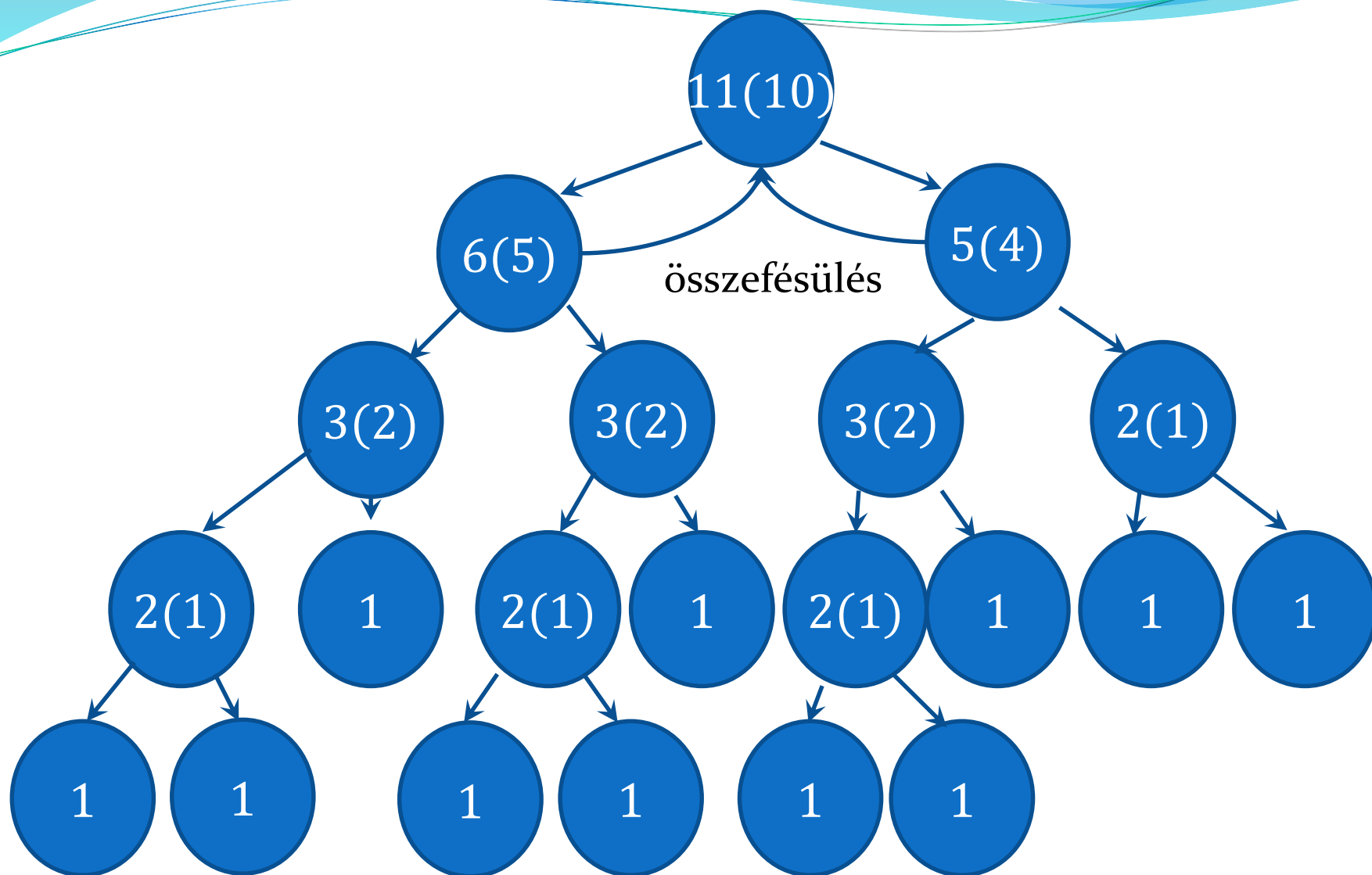
2	3	8	11	4	5	7	9	1	6	10
---	---	---	----	---	---	---	---	---	---	----

2	3	4	5	7	8	9	11	1	6	10
---	---	---	---	---	---	---	----	---	---	----

1	2	3	4	5	6	7	8	9	10	11
---	---	---	---	---	---	---	---	---	----	----

Összefuttatásos rendezés

- Hatékonyságelemzés:
 - Az összehasonlítások számát becsüljük a legrosszabb esetben
 - Az összefuttatás k és m hosszú sorozatok összefésülésekor $k + m - 1$ összehasonlítást végez
 - $M\ddot{O}ö_{\text{sszefuttat}}(k, m) = k + m - 1$
 - ha n a két sorozat együttes hossza
 - $M\ddot{O}ö_{\text{sszefuttat}}(n) = n - 1$
 - A MergeSort eljárásnál a közel egyenlő szétvágást alapul véve: milyen hosszú részekre vágja az eljárás a (rész)sorozatokot a rekurzív hívások szintjein.



$n = 11$ -re

()-ben:öh-k száma $MÖ = 10 + (5 + 4) + (2 + 2 + 2 + 1) + (1 + 1 + 1) = 29$

Összefuttatásos rendezés

- Az egyenlő szétvágások bináris fája majdnem teljes.
- A fa magassága: $4 = \lceil \log_2 11 \rceil$
 - Általában is: $h = \lceil \log_2 n \rceil$
- A fa leveleihez nem tartozik összehasonlítás, így éppen h szinten kell összegezni az összehasonlítások számát
 - Ezek összege szintenként egyre csökken
 - mindenütt $\leq n - 1$
- Így $M\ddot{O}_{MS}(n) \leq (n - 1) * \lceil \log_2 n \rceil = \Theta(n * \log_2 n)$

Rendezések

- Összehasonlításos rendezőknél mit tudunk mondani a rendezés időigényére? Tudunk-e alsó becslést adni?
 - Ugyanaz, mintha barkóbázva kellene kitalálni, hogy az elemek melyik sorrendje (permutációja) az igazi sorrend
 - Kezdetben $n!$ lehetséges sorrend jön szóba.
 - Két elemet összehasonlítva, a válasz két részre osztja a sorrendeket
 - Ha például azt kapjuk, hogy $x < y$, akkor az olyan sorrendek, amikben x hátrébb van y -nál, már nem jönnek szóba
 - Ha a válasz mindig olyan, hogy minél több sorrend maradjon meg, akkor k kérdés után még szóba jövő sorrendek száma

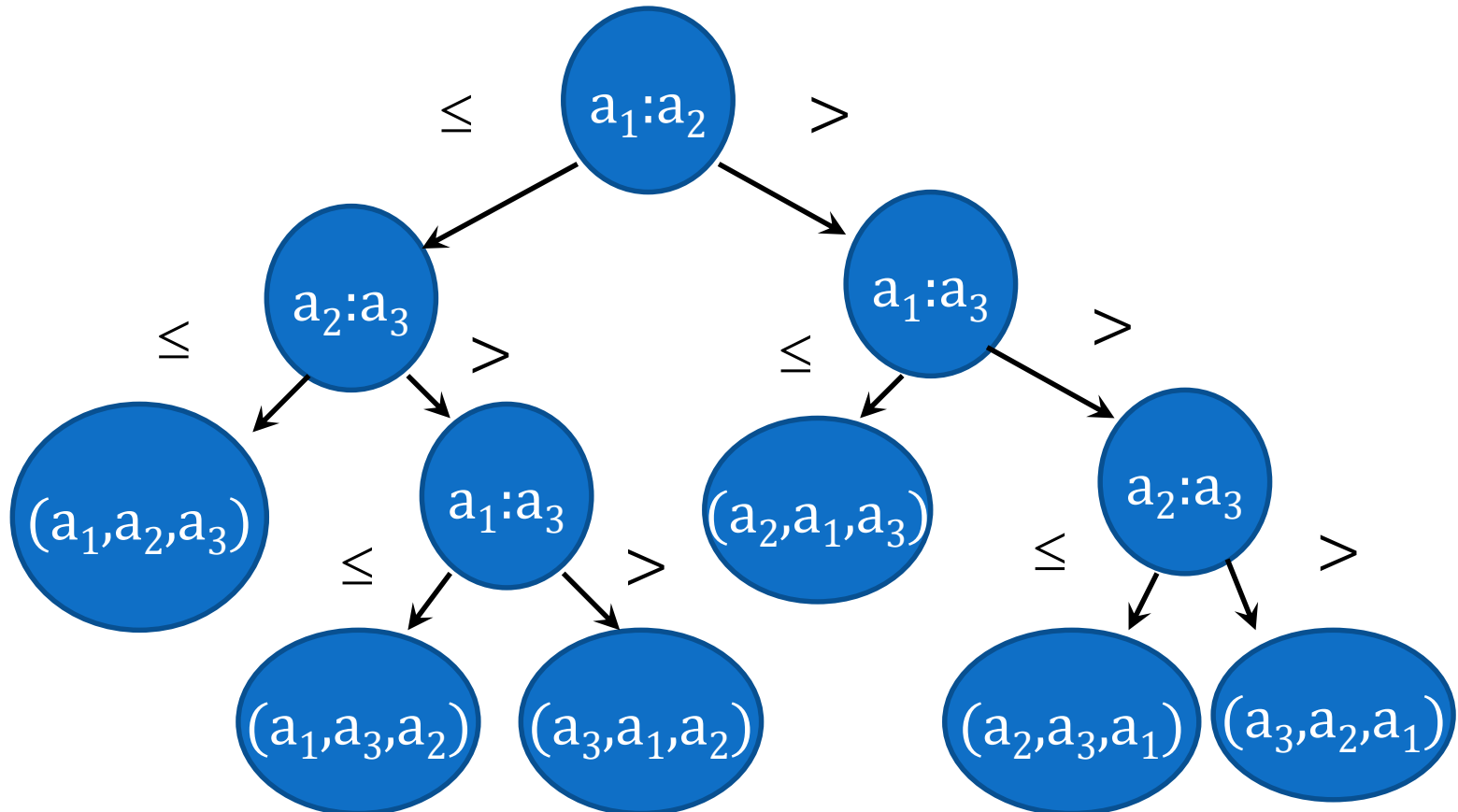
$$\frac{n!}{2^k}$$

Rendezések

- Döntési fa modell
 - Az összehasonlítások tekinthetők döntési fának, amik a rendezés során történt összehasonlításokat ábrázolják
 - Tegyük fel, hogy minden elem különböző
 - Egy belső csúcsot egy $a_i : a_j$ párral jelölhetünk, ahol
$$1 \leq i, j \leq n$$
 - A levelek egy-egy permutációnak feleltethetők meg

Rendezések

- Baloldali részfa: az $a_i \leq a_j$ után szükséges összehasonlítások
- Jobboldali részfa: az $a_i > a_j$ után szükséges összehasonlítások
- Levél: a rendezett sorrend



Alsó korlát a legrosszabb esetben

- Tétel: Bármely n elemet rendező döntési fa magassága
 $\Omega(n * \log_2 n)$
 - Bizonyítás: Vegyük az n elemet rendező döntési fát, jelöljük ennek magasságát h -val.
 - A fának $n!$ levele van – ezek a permutációk.
 - A h mélységű bináris fa leveleinek száma $\leq 2^h$, ezért
$$n! \leq 2^h$$
 - $h \geq \log_2(n!)$
 - felhasználva, hogy a log függvény monoton növvő

Alsó korlát a legrosszabb esetben

- Ismert a Stirling formula ($\sim 1730!$):

$$n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$$

- ahol e a természetes logaritmus alapja
- Ennek alapján

$$n! > \left(\frac{n}{e}\right)^n$$

- Behelyettesítve:
- $h \geq \log_2 \left(\left(\frac{n}{e}\right)^n \right) = n * \log_2 n - n * \log_2 e = \Omega(n * \log_2 n)$

Alsó korlát a legrosszabb esetben

- Következmény 1.
 - A kupacrendezés és az összefésüléses rendezés aszimptotikusan optimális összehasonlító rendezések.
- Bizonyítás
 - A kupacrendezés és az összefésüléses rendezés futási idejének felső korlátja $\Theta(n * \log_2 n)$ a legrosszabb esetre megadott $\Omega(n * \log_2 n)$ felső korláttal.
- Következmény 2.
 - Nincs lineáris idejű összehasonlításos rendezés
 - ☹️