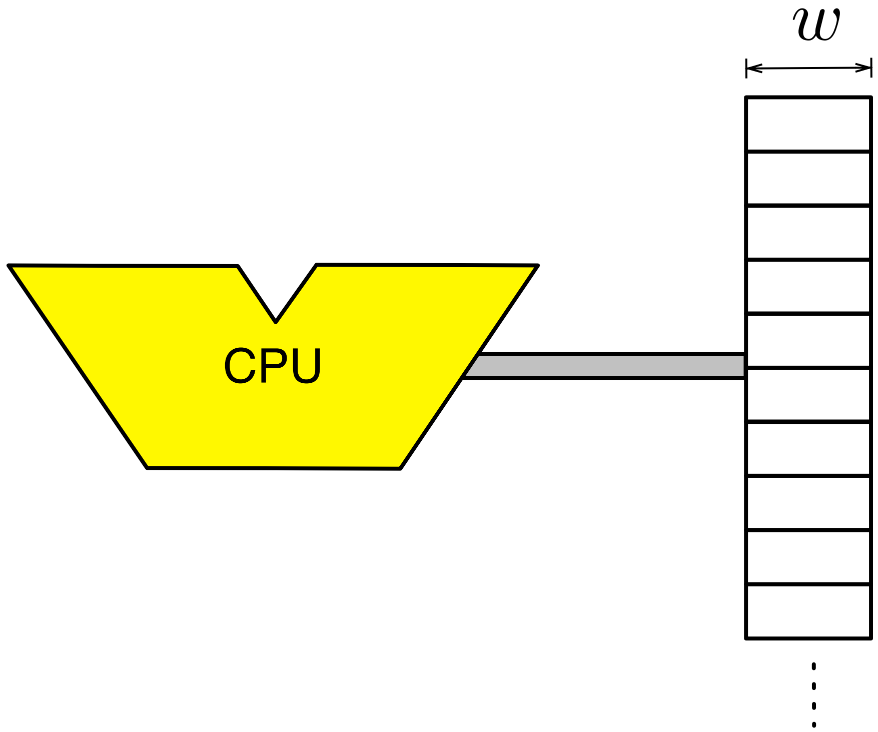
Partial Sums on the Ultra-Wide Word RAM

Philip Bille
Inge Li Gørtz
Frederik Rye Skjoldjensen

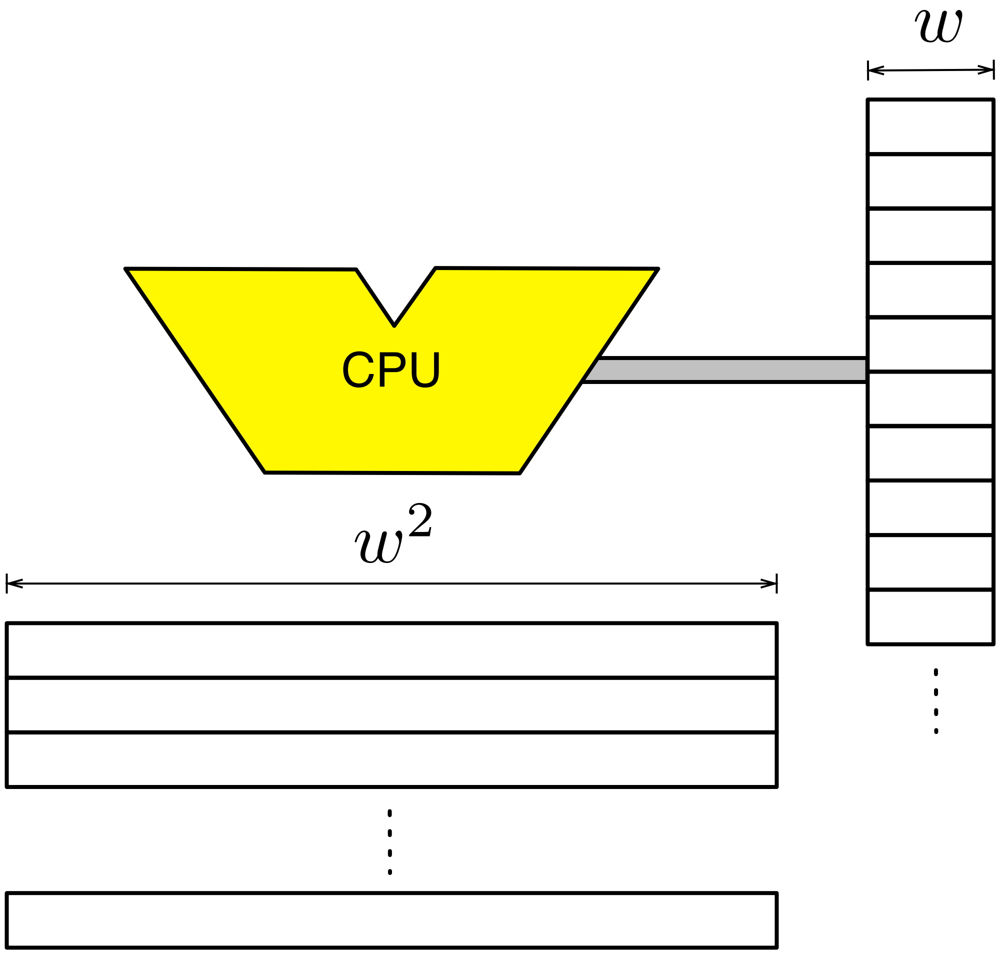
Outline

- Word RAM and ultra-wide word RAM
- Partial sums
- Fenwick tree
- UWRAM Fenwick tree

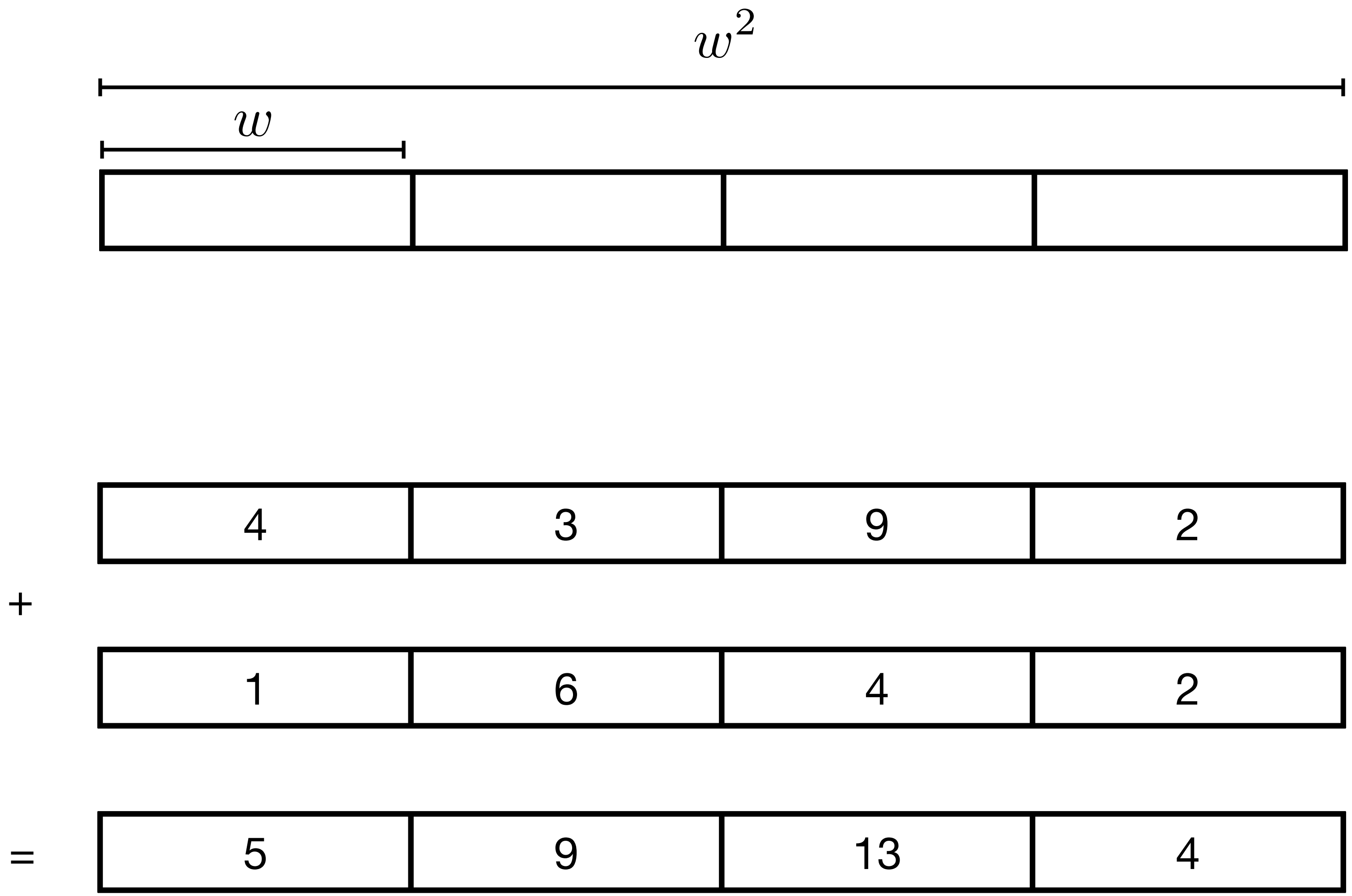
Word RAM



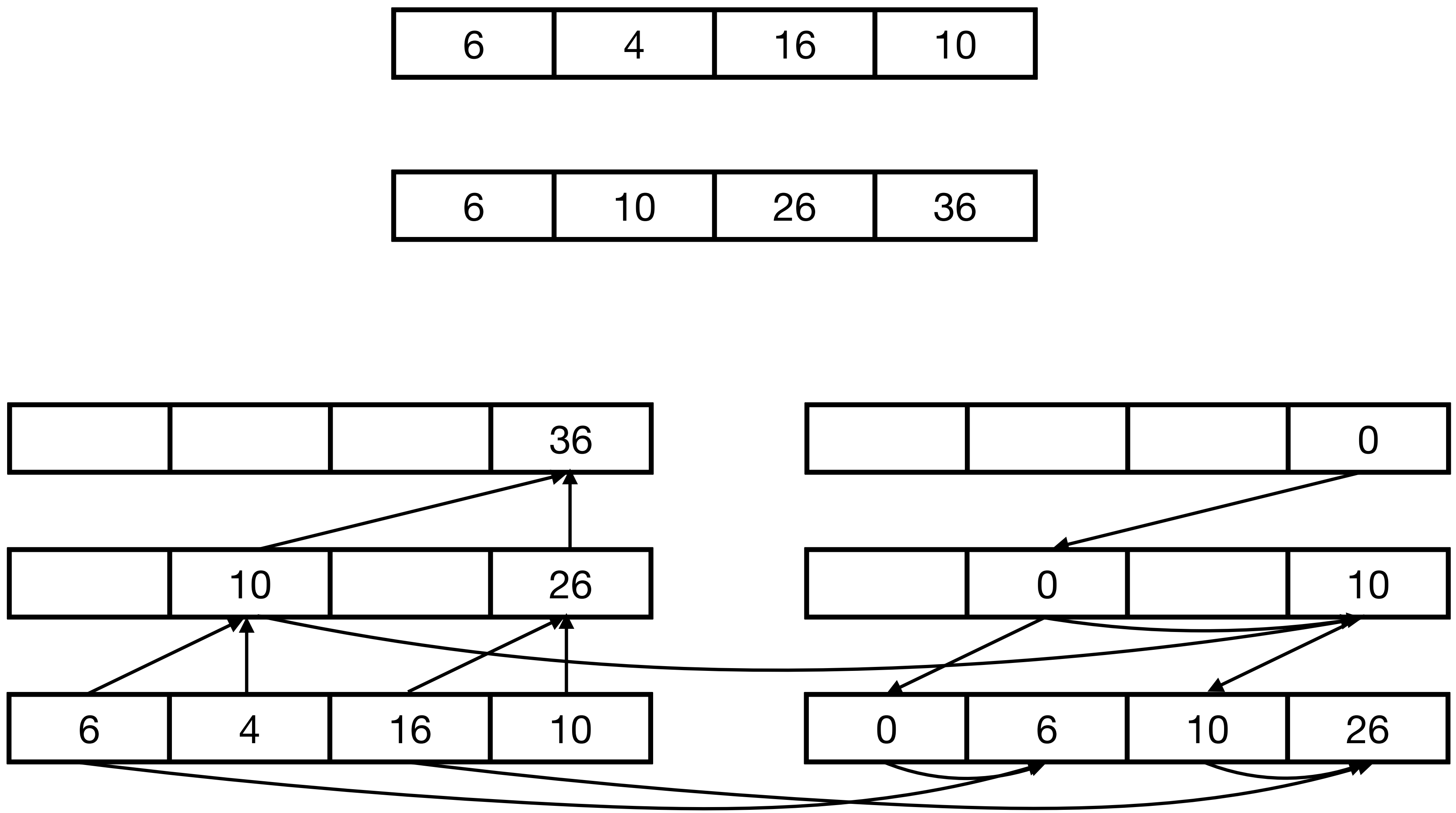
Ultra-Wide Word RAM



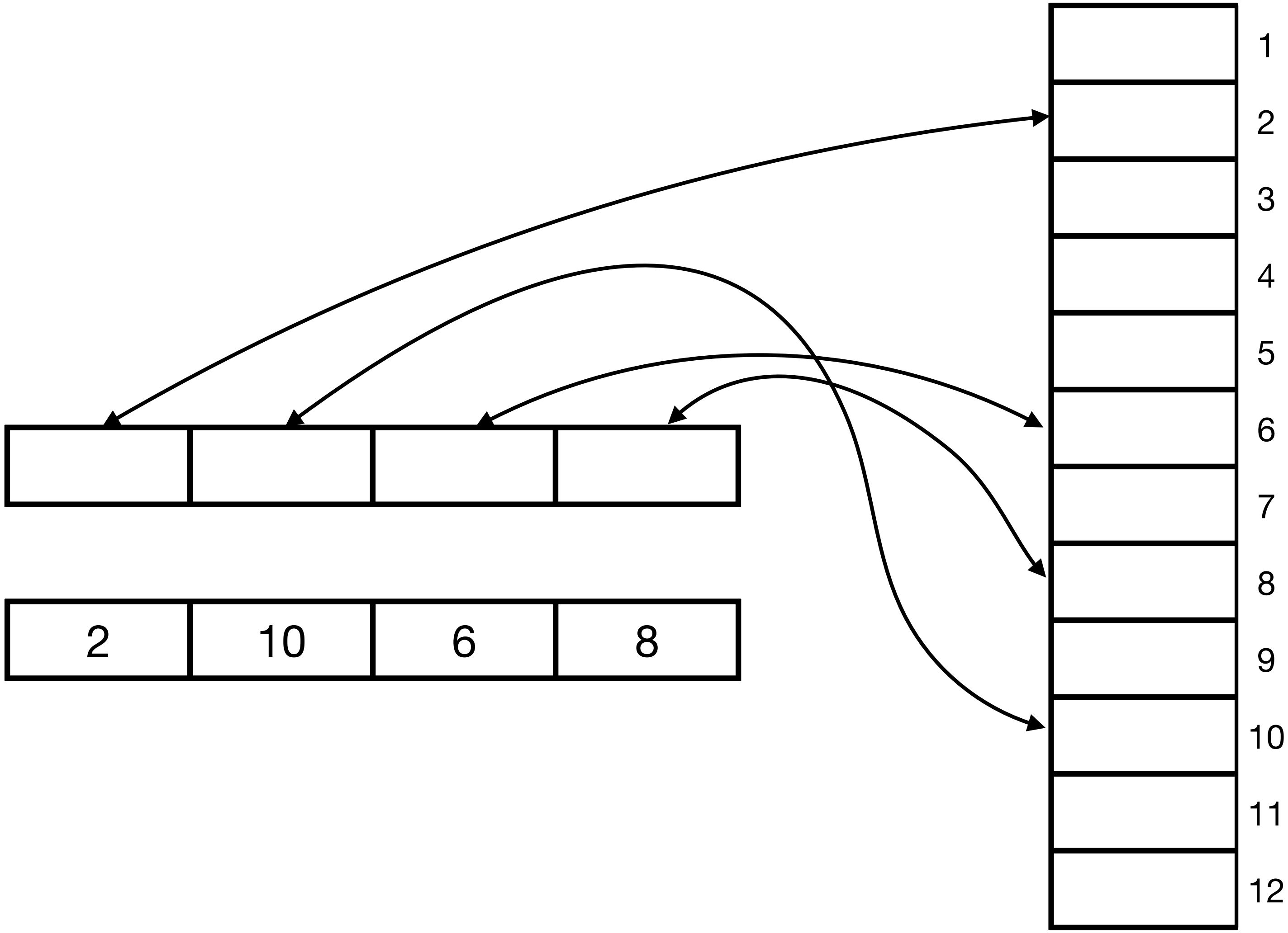
Word-Level Parallelism



Word-Level Parallelism



Scattered Memory Access



Partial Sums

- Partial sums. Maintain array A[1,..., n] of integers support the following operations.
  - SUM(i): return A[1] + A[2] + ... + A[i]
  - UPDATE(i, Δ): set A[i] = A[i] + Δ

-	1	2	1	1	0	2	3	1	0	1	3	4	1	1	1	2
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Partial Sums

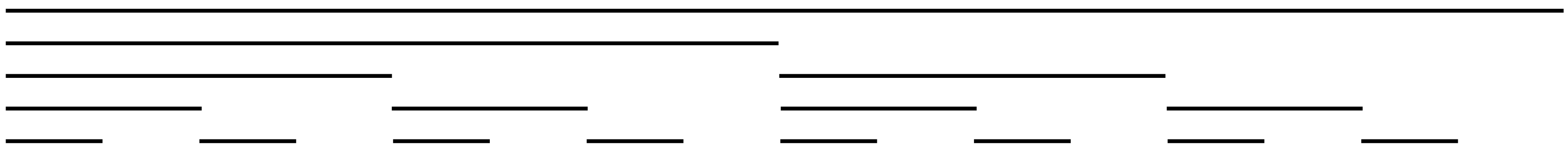
time	space	model	ref
$\Theta(\log n)$	in-place	WRAM	misc
$O(\log \log n)$	$O(n)$	RAMBO	[BKMN2006]
$O(1)$	$O(n^{\varepsilon w})$	RAMBO	[BKMN2006]
$O(\log \log n)$	$O(nw \log \log n)$	UWRAM	[FLNS2015]
$O(1)$	$O(n^{\varepsilon w} + nw \log \log n)$	UWRAM	[FLNS2015]
$O(\log \log n)$	in-place	UWRAM	**[new]**
$O(1)$	in-place	UWRAM w. mul	**[new]**

Fenwick Tree

-	1	3	1	5	0	2	3	11	0	1	3	8	1	2	1	24
-	1	3	1	5	0	2	3	11	0	1	3	8	1	2	1	13
-	1	3	1	5	0	2	3	6	0	1	3	8	1	2	1	5
-	1	3	1	2	0	2	3	4	0	1	3	7	1	2	1	3
-	1	2	1	1	0	2	3	1	0	1	3	4	1	1	1	2
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

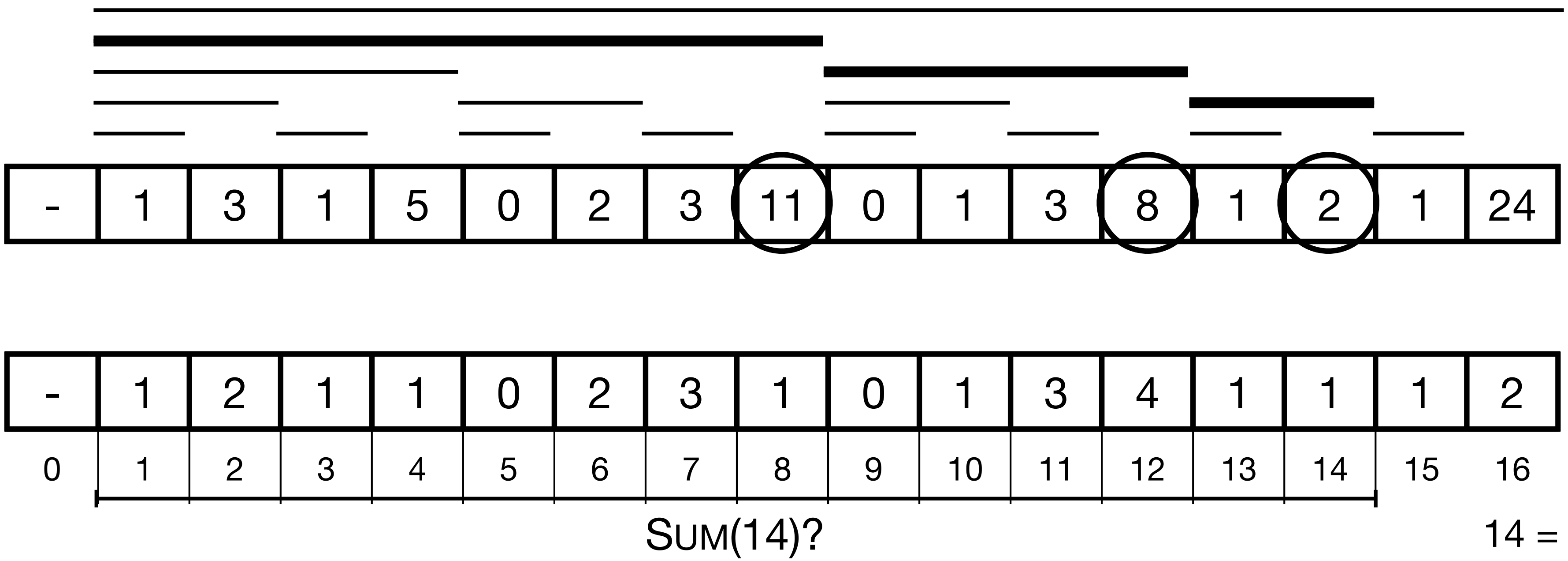
- Fenwick tree. Replace A by another array F.
  - Replace all even entries A[2i] by A[2i - 1] + A[2i].
  - Recurse on the entries A[2, 4, .., n] until we are left with a single element.

Fenwick Tree



-	1	3	1	5	0	2	3	11	0	1	3	8	1	2	1	24
-	1	2	1	1	0	2	3	1	0	1	3	4	1	1	1	2
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

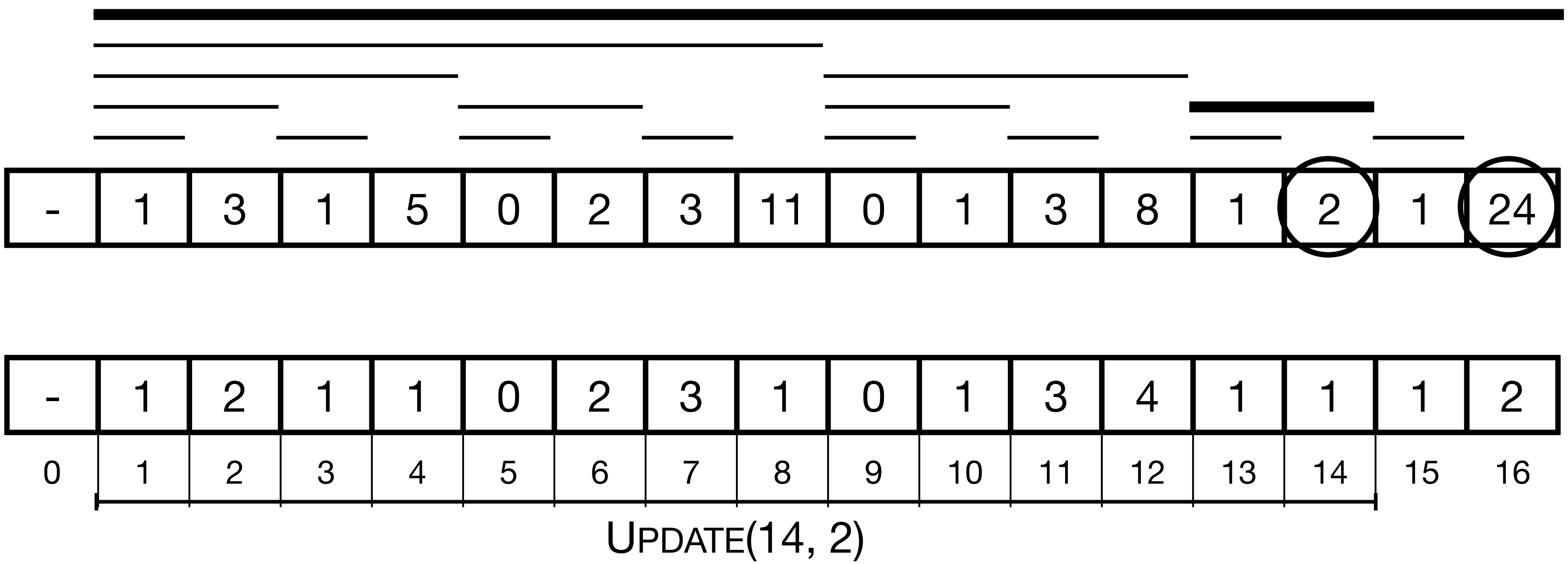
Fenwick Tree



$14 = 1110_2$
$12 = 1100_2$
$8 = 1000_2$
$0 = 0000_2$

- SUM.
  - SUM(i): add largest partial sums covering [1,..,i].
  - Indexes i_0, i_1, .. in F given by $i_0 = i$ and $i_{j+1} = i_j$ - rmb(i_j), where rmb(i_j) is the integer corresponding to the rightmost 1-bit in i. Stop when we get 0.
- Time. O(log n)

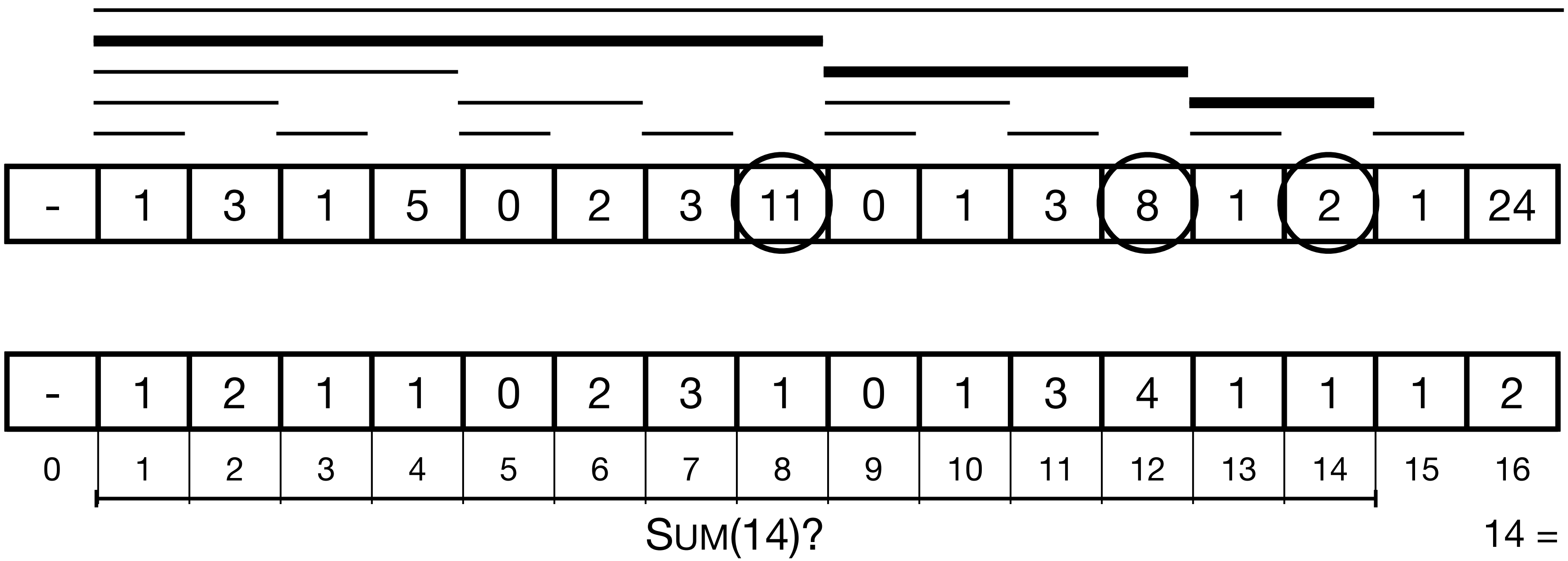
Fenwick Tree



$14 = 1110_2$
$16 = 10000_2$

- UPDATE.
  - UPDATE(i, Δ): add Δ to partial sums covering i.
  - Indexes i_0, i_1, .. in F given by $i_0 = i$ and $i_{j+1} = i_j + \text{rmb}(i_j)$. Stop when we get n.
- Time. O(log n)

UWRAM Fenwick Tree



$14 = 1110_2$
$12 = 1100_2$
$8 = 1000_2$
$0 = 0000_2$

- SUM.
  - SUM(i): add largest partial sums covering [1,..,i].
  - Indexes i_0, i_1, .. in F given by $i_0 = i$ and $i_{j+1} = i_j$ - rmb(i_j), where rmb(i_j) is the integer corresponding to the rightmost 1-bit in i. Stop when we get 0.

UWRAM Fenwick Tree

$14 = 1110_2$
$12 = 1100_2$
$8 = 1000_2$
$0 = 0000_2$

14	14	14	14
8	4	2	1
8	4	2	0
14	6	2	0
6	2	0	0
8	12	14	0

- Time. O(log log n)

UWRAM Fenwick Tree

time	space	model
O(log log n)	in-place	UWRAM
O(1)	in-place	UWRAM w. mul

Open Problems

- Can we implement SELECT(j): return the smallest i such that SUM(i) ≥ j?
- What is the precise relation between UWRAM and RAMBO?