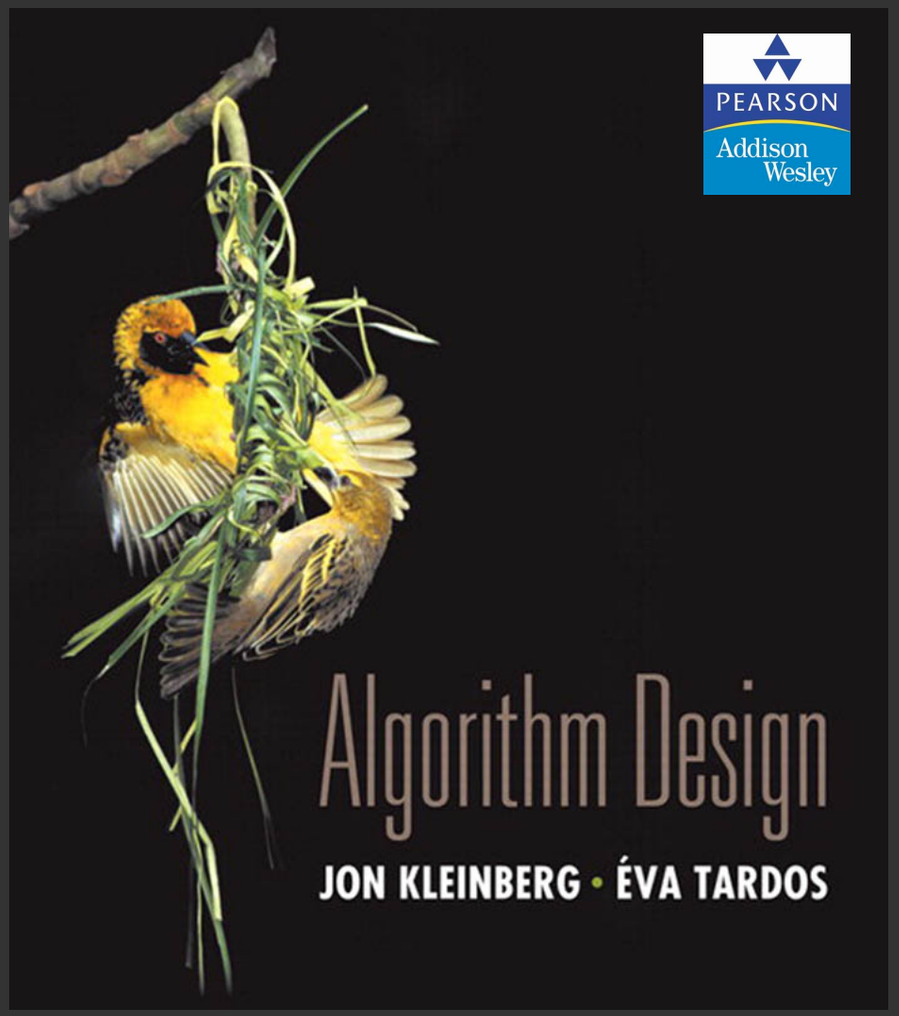


5. Divide And Conquer I

- *3-way partitioning demo*
- *randomized quickselect demo*

Lecture slides by Kevin Wayne

http://www.cs.princeton.edu/~wayne/kleinberg-tardos

Last updated on 3/5/18 9:47 AM

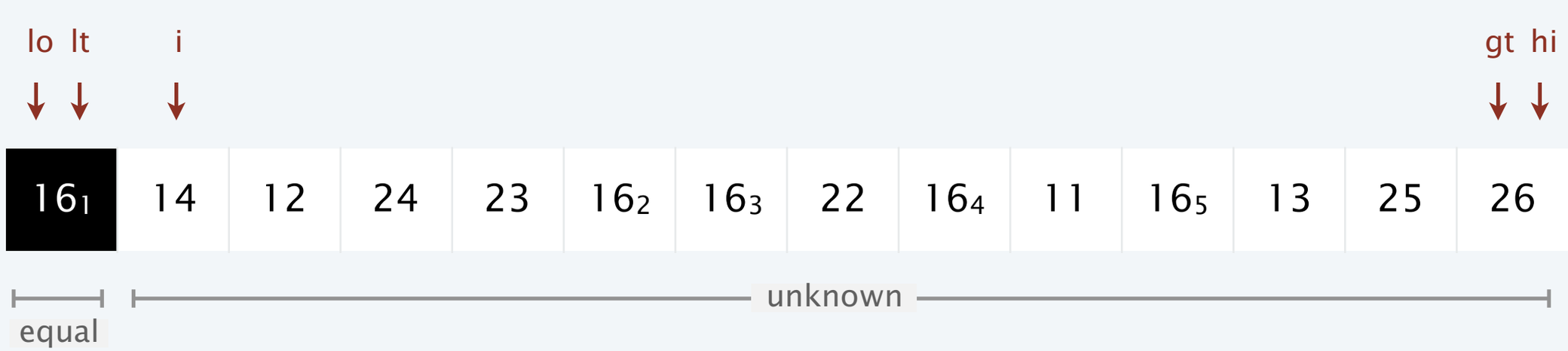
Section 2.3

5. Divide and Conquer

- *3-way partitioning demo*
- *randomized quickselect demo*

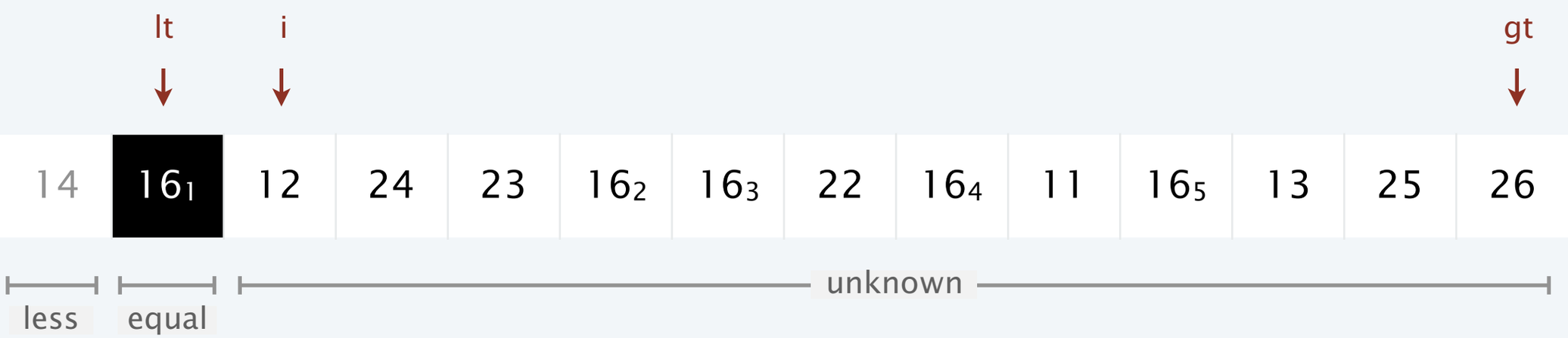
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



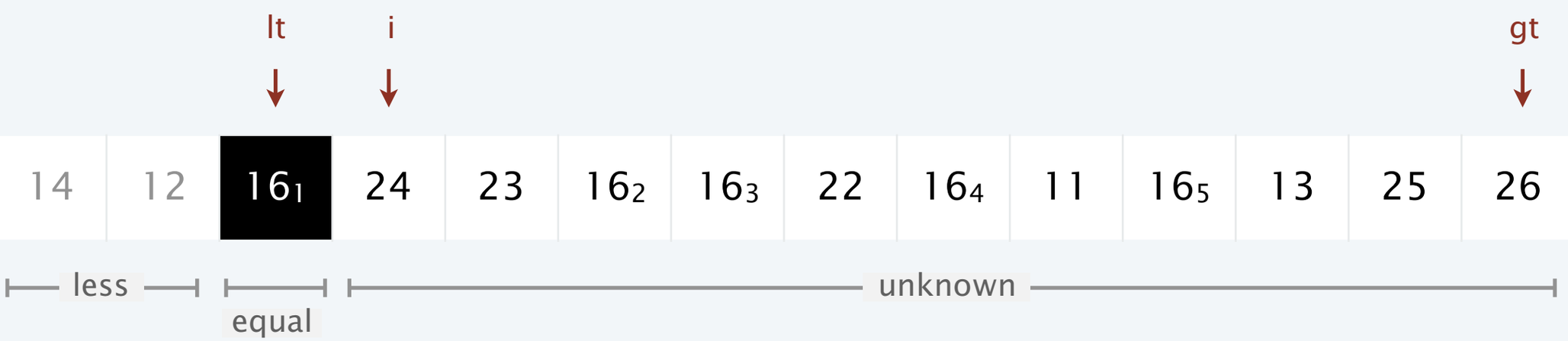
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



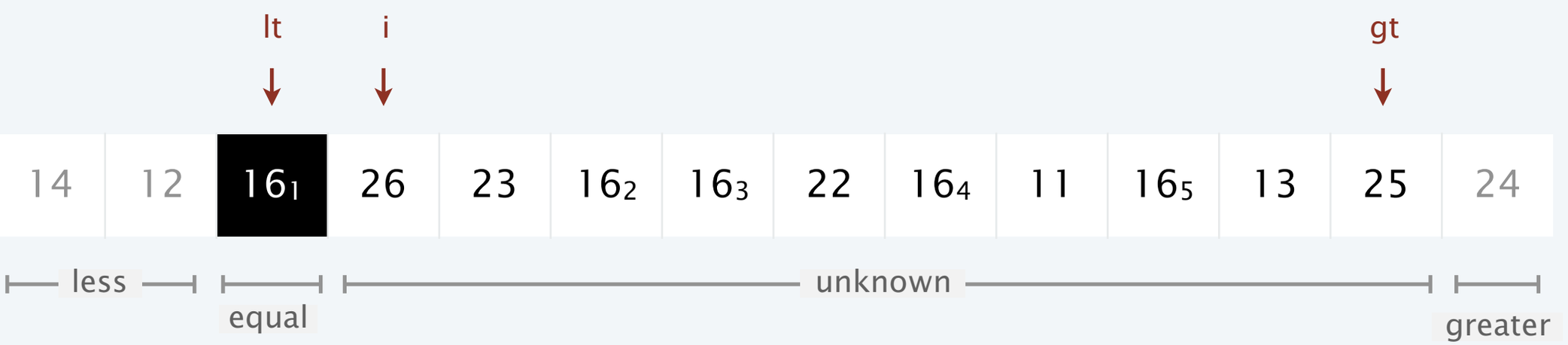
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



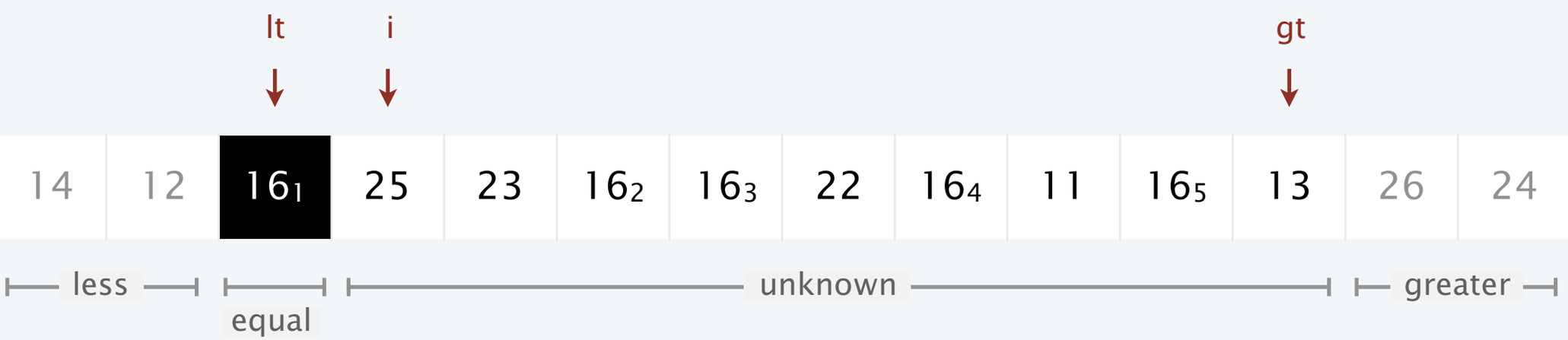
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



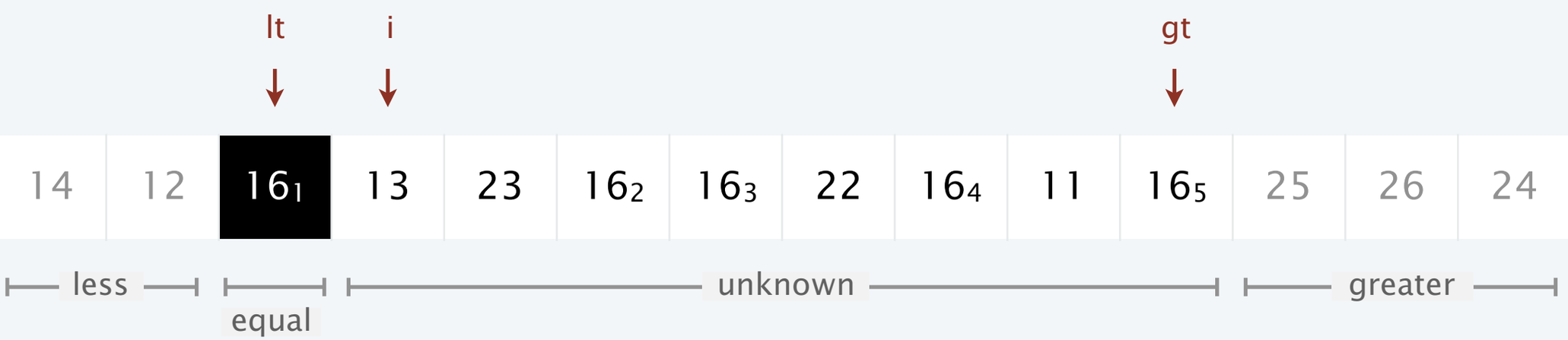
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



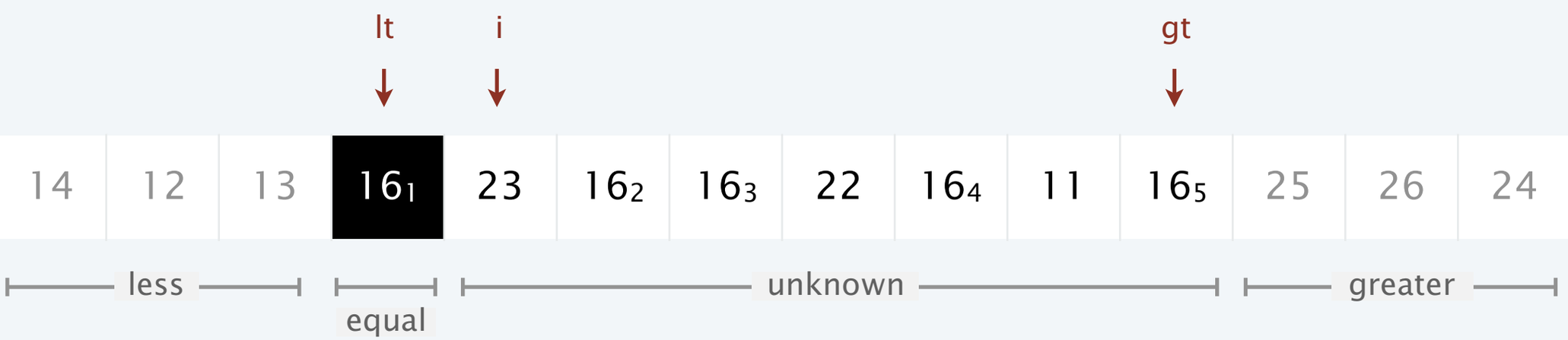
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



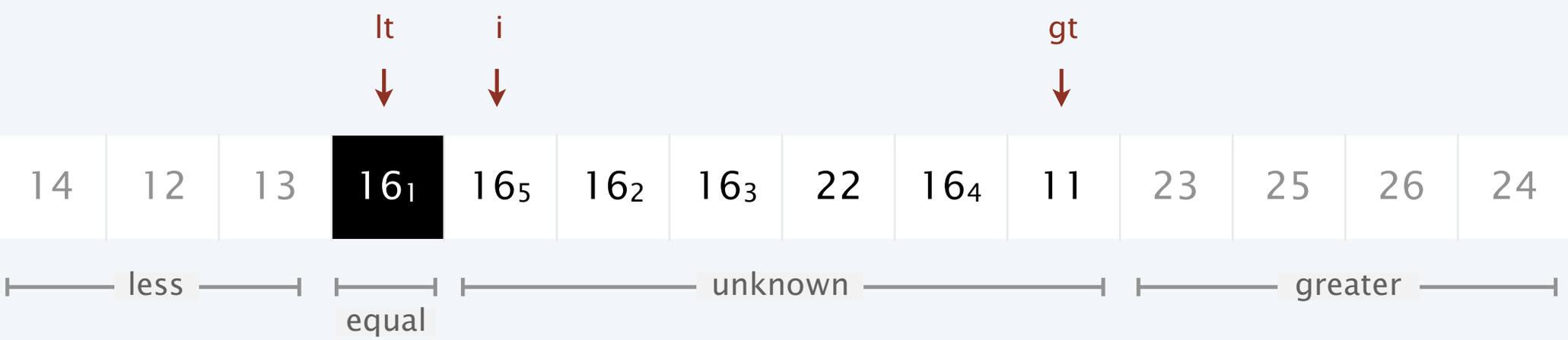
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i

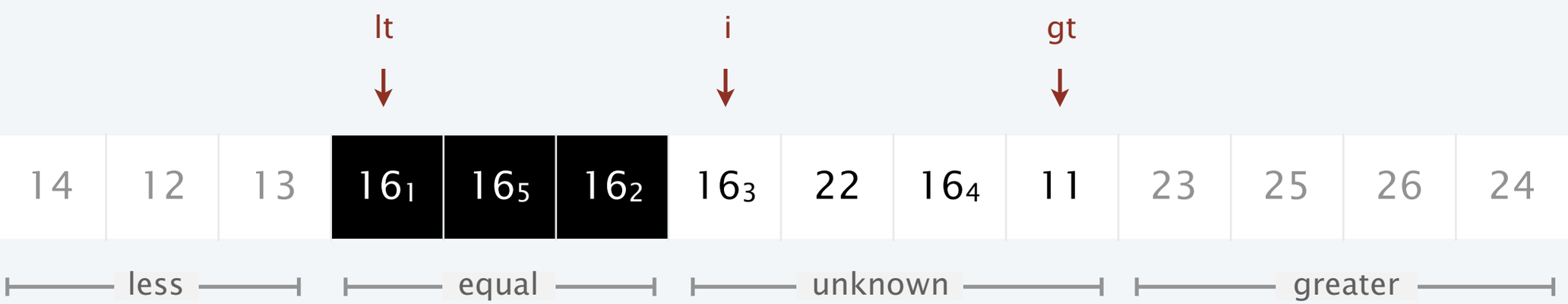


Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i

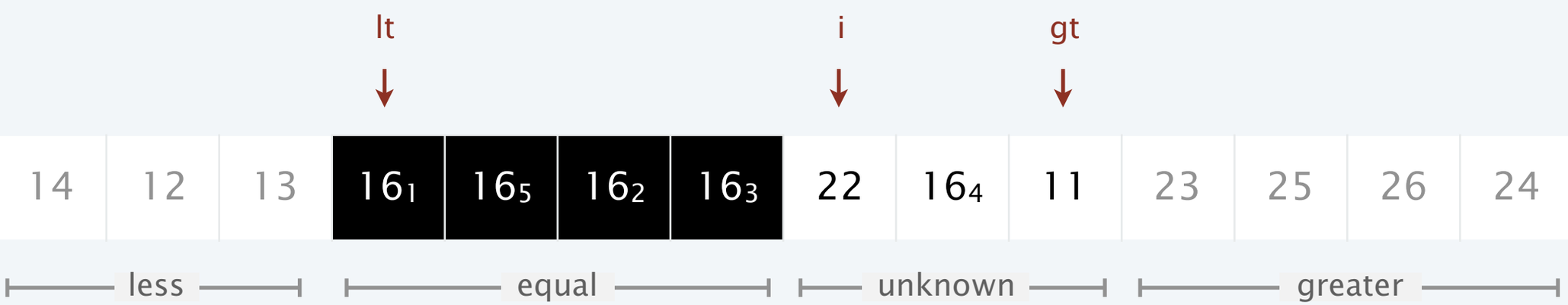
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



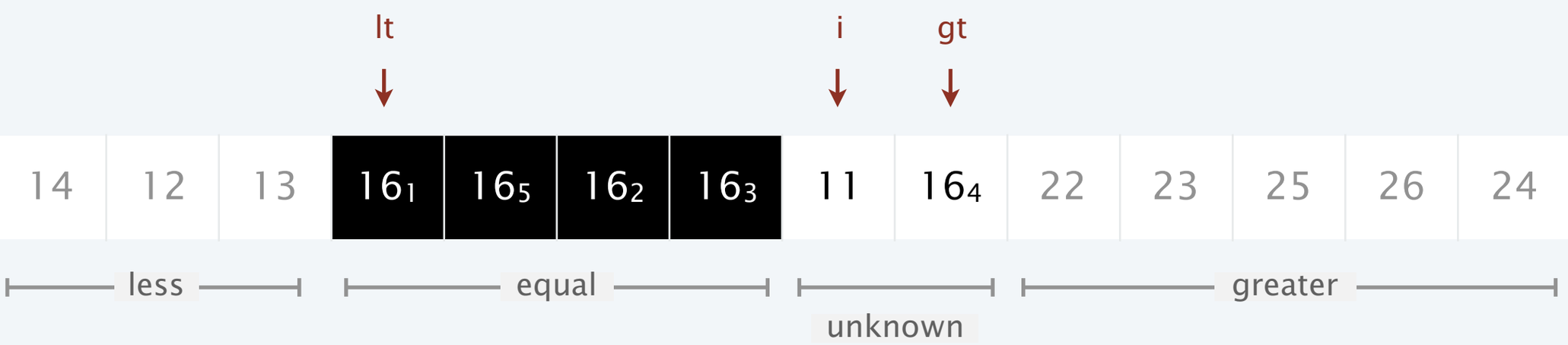
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



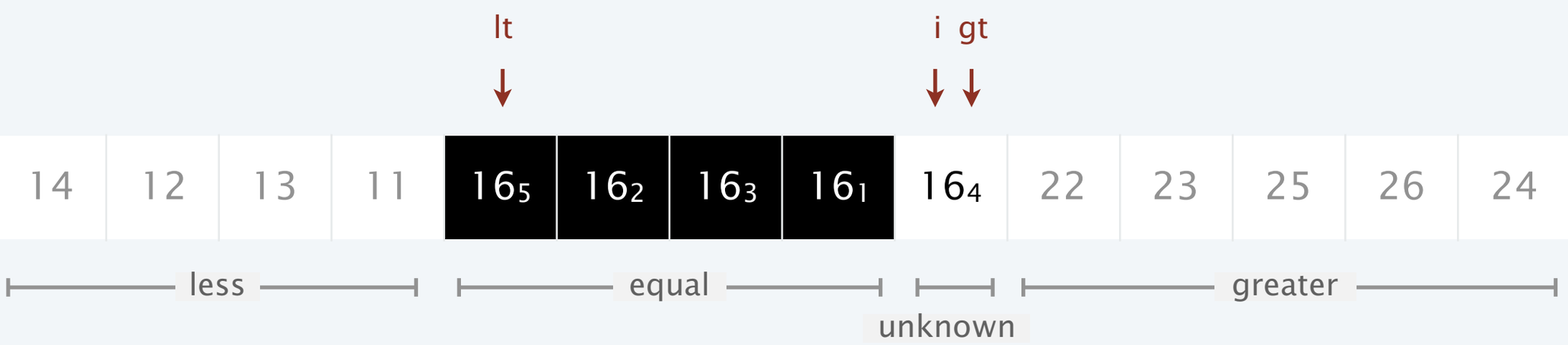
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



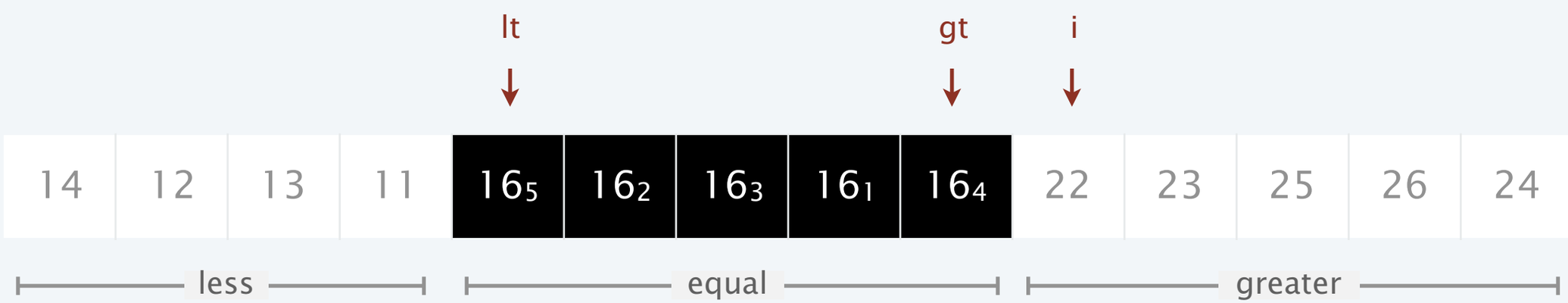
Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



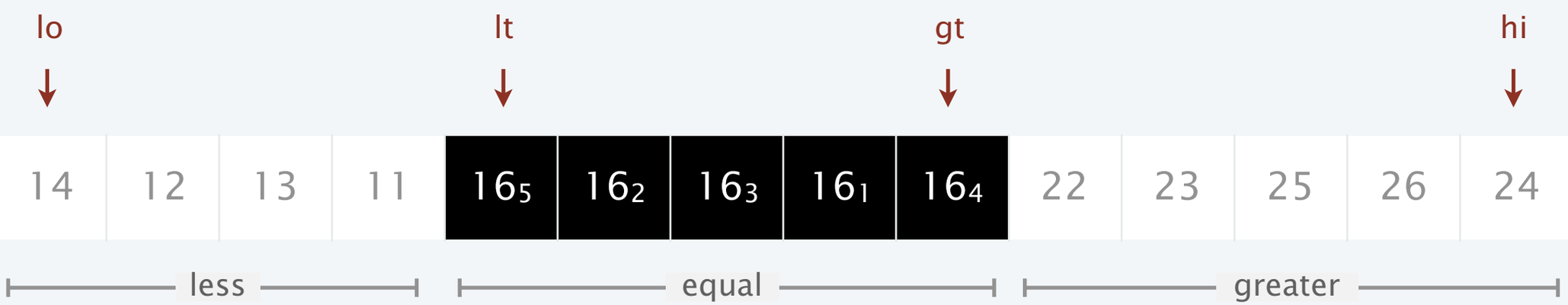
Dijkstra 3-way partitioning demo

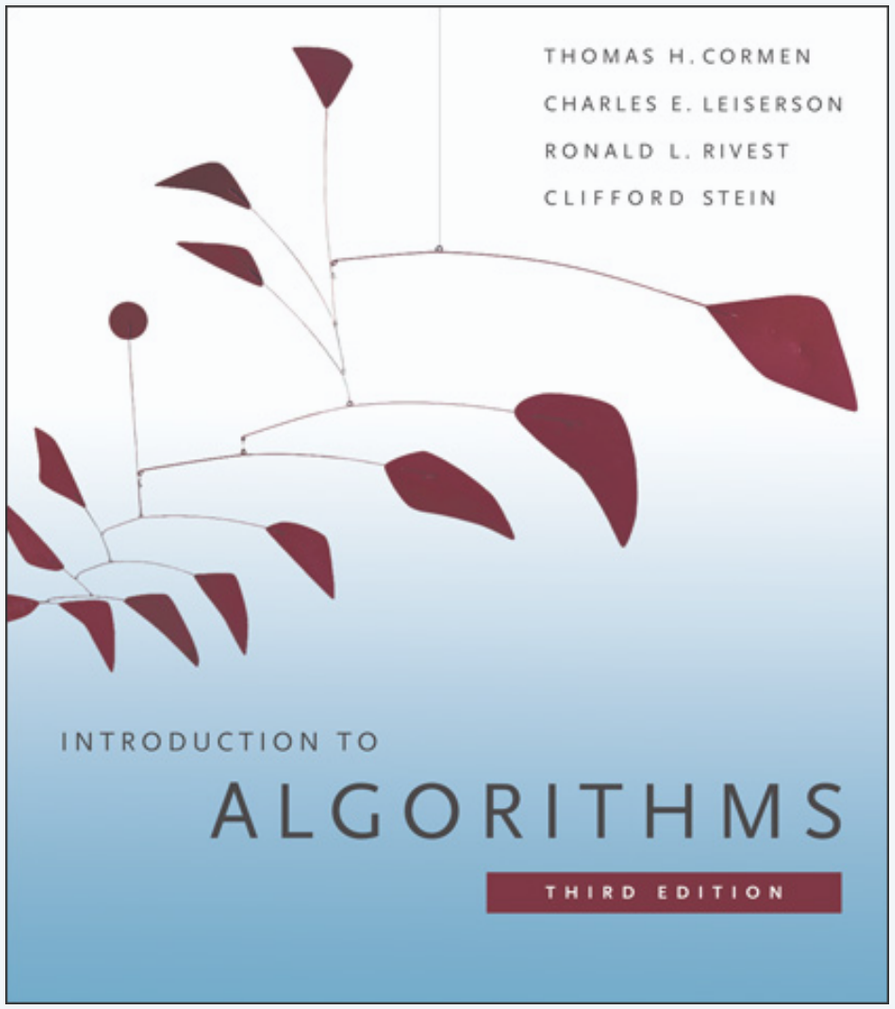
- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i



Dijkstra 3-way partitioning demo

- Let p be pivot item.
- Swap p to index lo.
- Scan i from left to right.
  - $(A[i] < p)$: exchange $A[lt]$ with $A[i]$; increment both lt and i
  - $(A[i] > p)$: exchange $A[gt]$ with $A[i]$; decrement gt
  - $(A[i] = p)$: increment i





Section 7.1–7.3

5. Divide and Conquer

- *3-way partitioning demo*
- *randomized quickselect demo*

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

select the k = 8th smallest

65	28	59	33	21	56	22	95	50	12	90	53	28	77	39

k = 8th smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

choose a pivot element at random and partition

k = 8th smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

partitioned array

k = 8th smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

recursively select 8th smallest element in left subarray

28	33	21	56	22	50	12	53	28	39	59	65	95	90	77

k = 8th smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

choose a pivot element at random and partition

k = 8th smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

partitioned array

k = 8th smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

recursively select the 3rd smallest element in right subarray

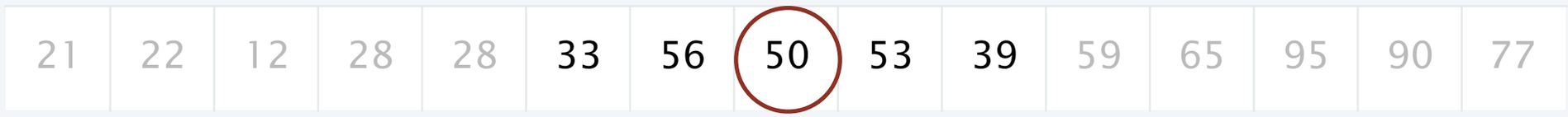
21	22	12	28	28	33	56	50	53	39	59	65	95	90	77

k = 3rd smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

choose a pivot element at random and partition

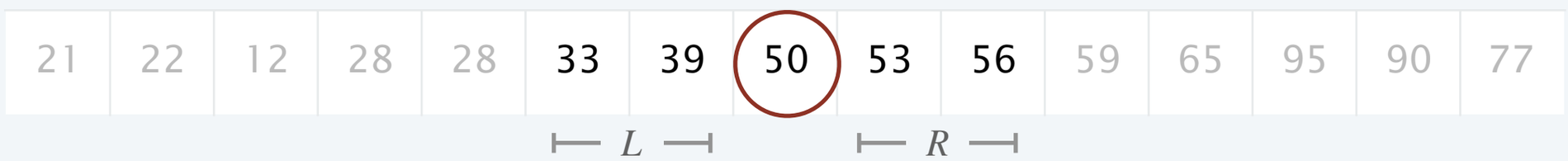


k = 3rd smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

partitioned array



k = 3rd smallest

Quickselect demo

- Pick a random pivot element $p \in A$.
- 3-way partition the array into L, M, and R.
- Recur in one subarray—the one containing the k^{th} smallest element.

stop: desired element is in middle subarray