Copying Garbage Collection

Guido Tack
14. November 2001
tack@ps.uni-sb.de

1 The idea of *Copying GC*

1 The idea of *Copying GC*

- Two "semi-spaces" (*From*-space and *To*-space)

1 The idea of *Copying GC*

- Two "semi-spaces" (*From*-space and *To*-space)
- Only From-space "active"

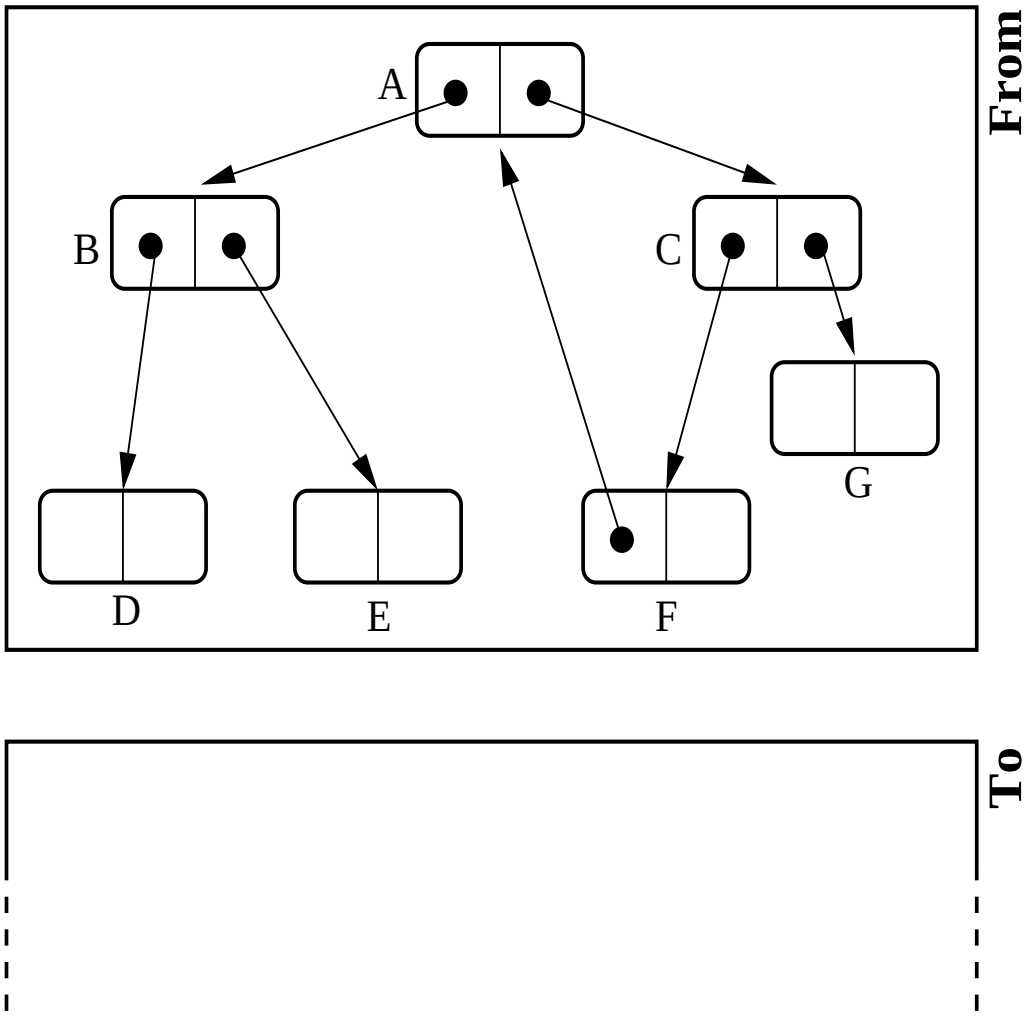
1 The idea of *Copying GC*

- Two "semi-spaces" (*From*-space and *To*-space)
- Only From-space "active"
- At GC time, copy the live nodes from From-Space to To-Space

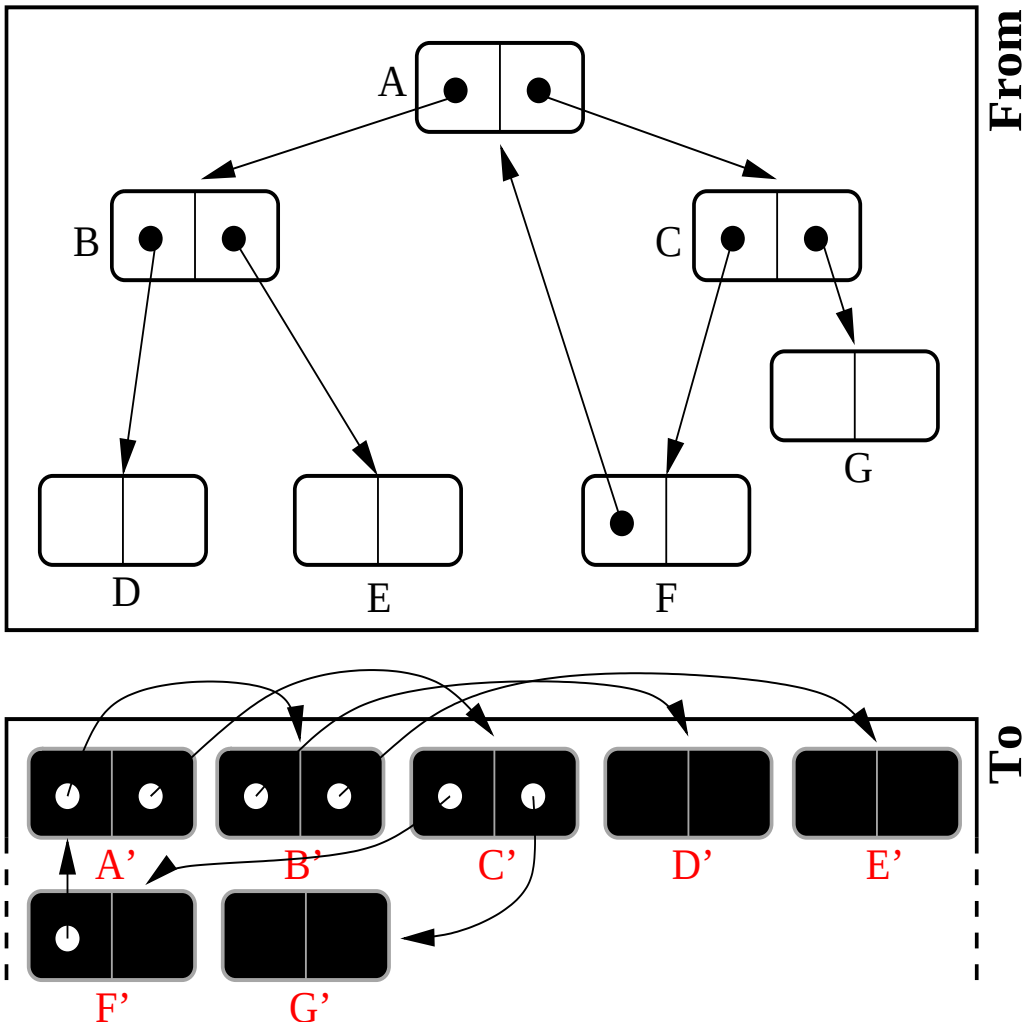
1 The idea of *Copying GC*

- Two "semi-spaces" (*From*-space and *To*-space)
- Only From-space "active"
- At GC time, copy the live nodes from From-Space to To-Space
- Then "flip" the spaces

2 Before GC



3 After GC



4 Cheney's algorithm(1)

4 Cheney's algorithm(1)

- Iterative algorithm

4 Cheney's algorithm(1)

- Iterative algorithm
- Interleaves copying and scanning

4 Cheney's algorithm(1)

- Iterative algorithm
- Interleaves copying and scanning
- Two pointers needed: scan / free

4 Cheney's algorithm(1)

- Iterative algorithm
- Interleaves copying and scanning
- Two pointers needed: scan / free
- Forwarding pointers used to preserve sharing

5 Cheney's algorithm(2)

5 Cheney's algorithm(2)

Tricolour abstraction:

5 Cheney's algorithm(2)

Tricolour abstraction:

- Black nodes: GC finished, not to be considered again

5 Cheney's algorithm(2)

Tricolour abstraction:

- Black nodes: GC finished, not to be considered again
- Grey nodes: Visited but not completed

5 Cheney's algorithm(2)

Tricolour abstraction:

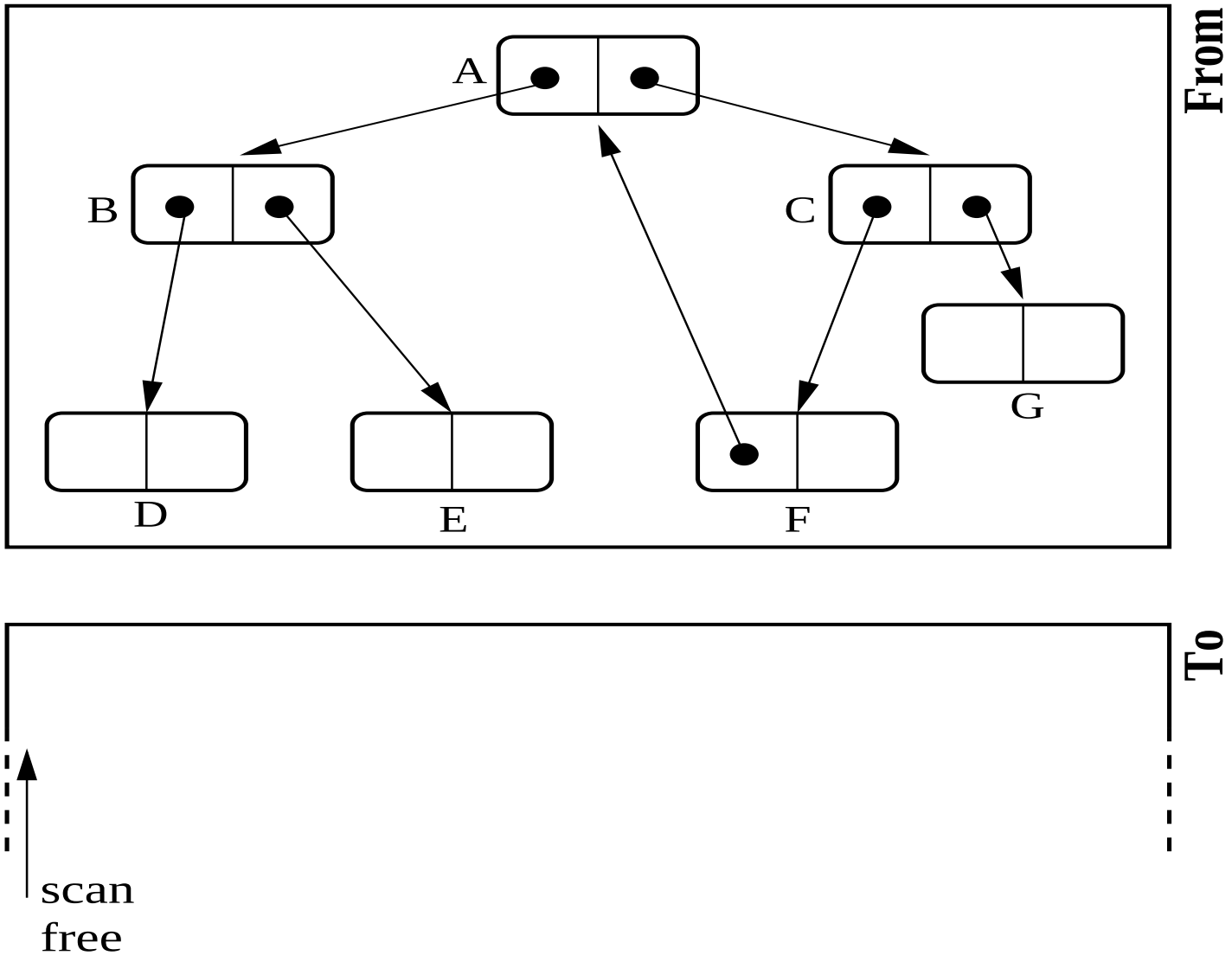
- Black nodes: GC finished, not to be considered again
- Grey nodes: Visited but not completed
- White nodes: Unvisited, considered garbage after tracing

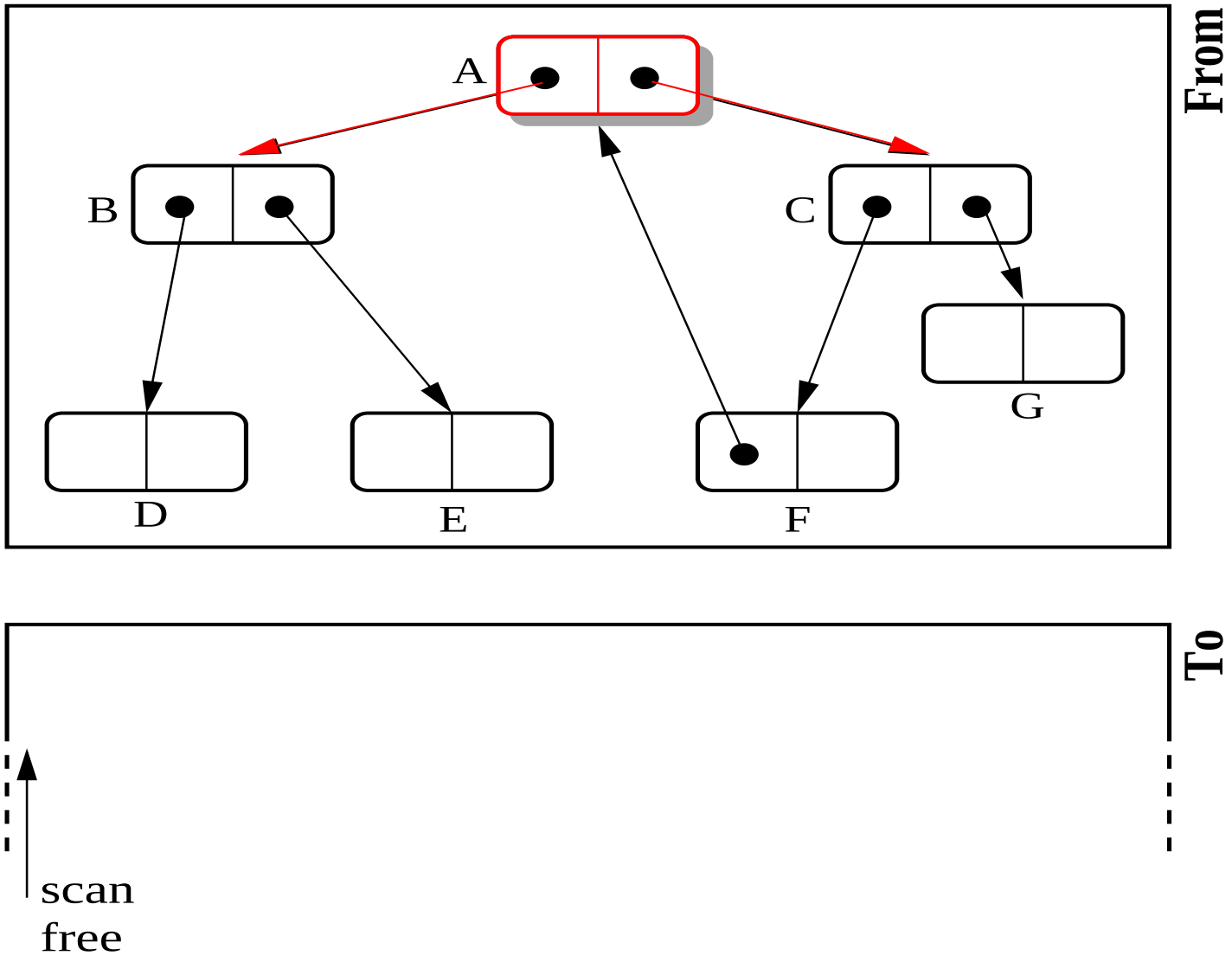
5 Cheney's algorithm(2)

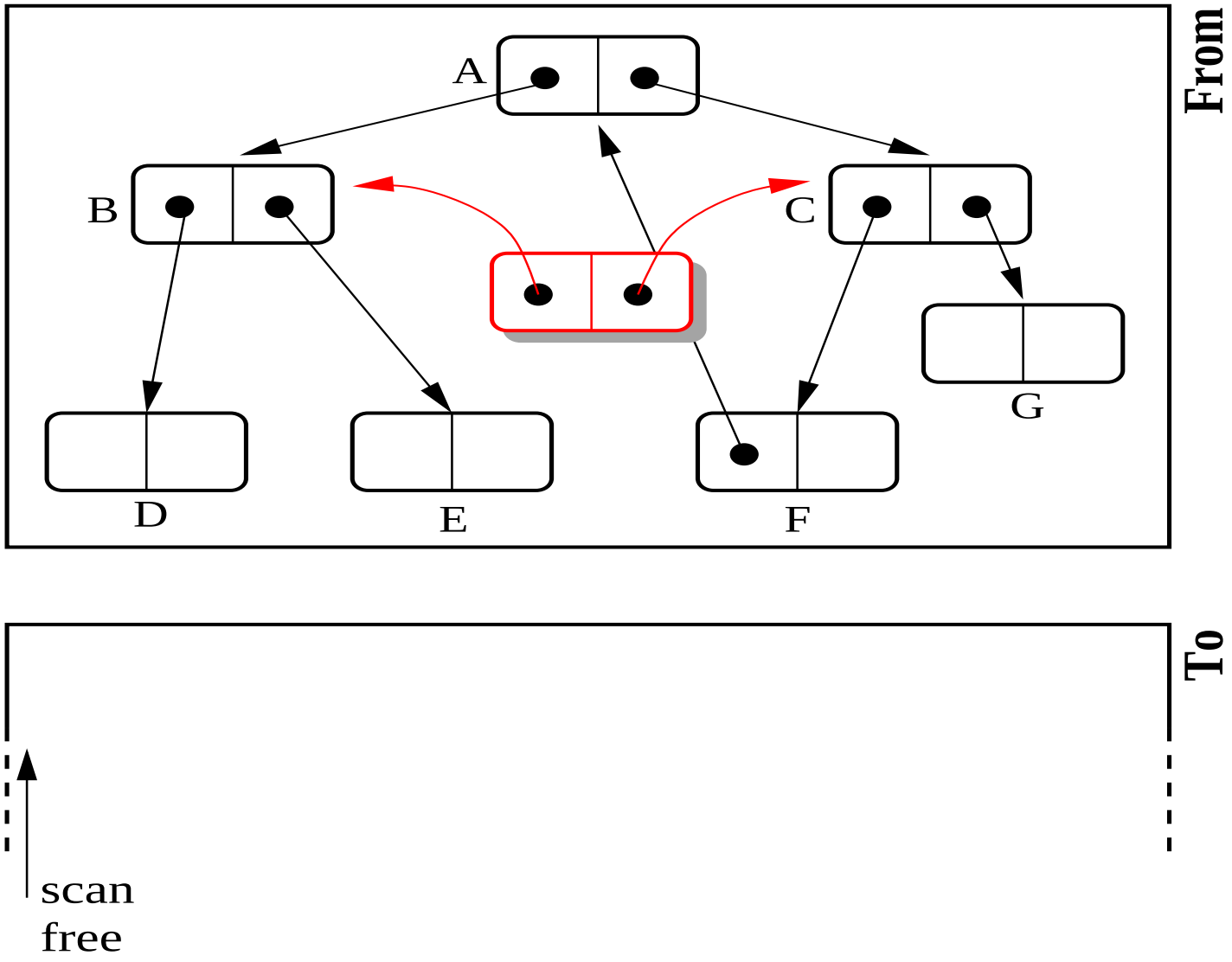
Tricolour abstraction:

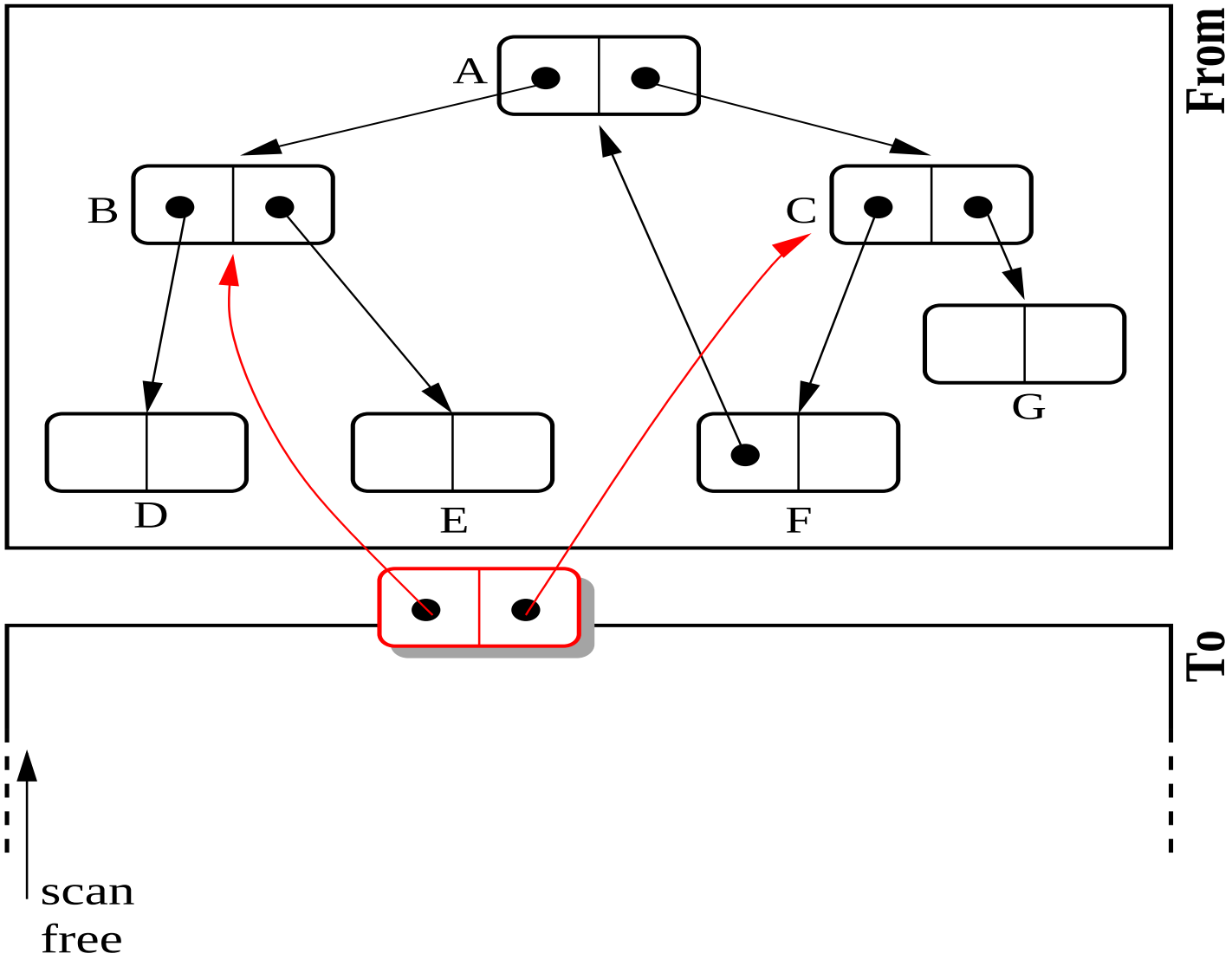
- Black nodes: GC finished, not to be considered again
- Grey nodes: Visited but not completed
- White nodes: Unvisited, considered garbage after tracing

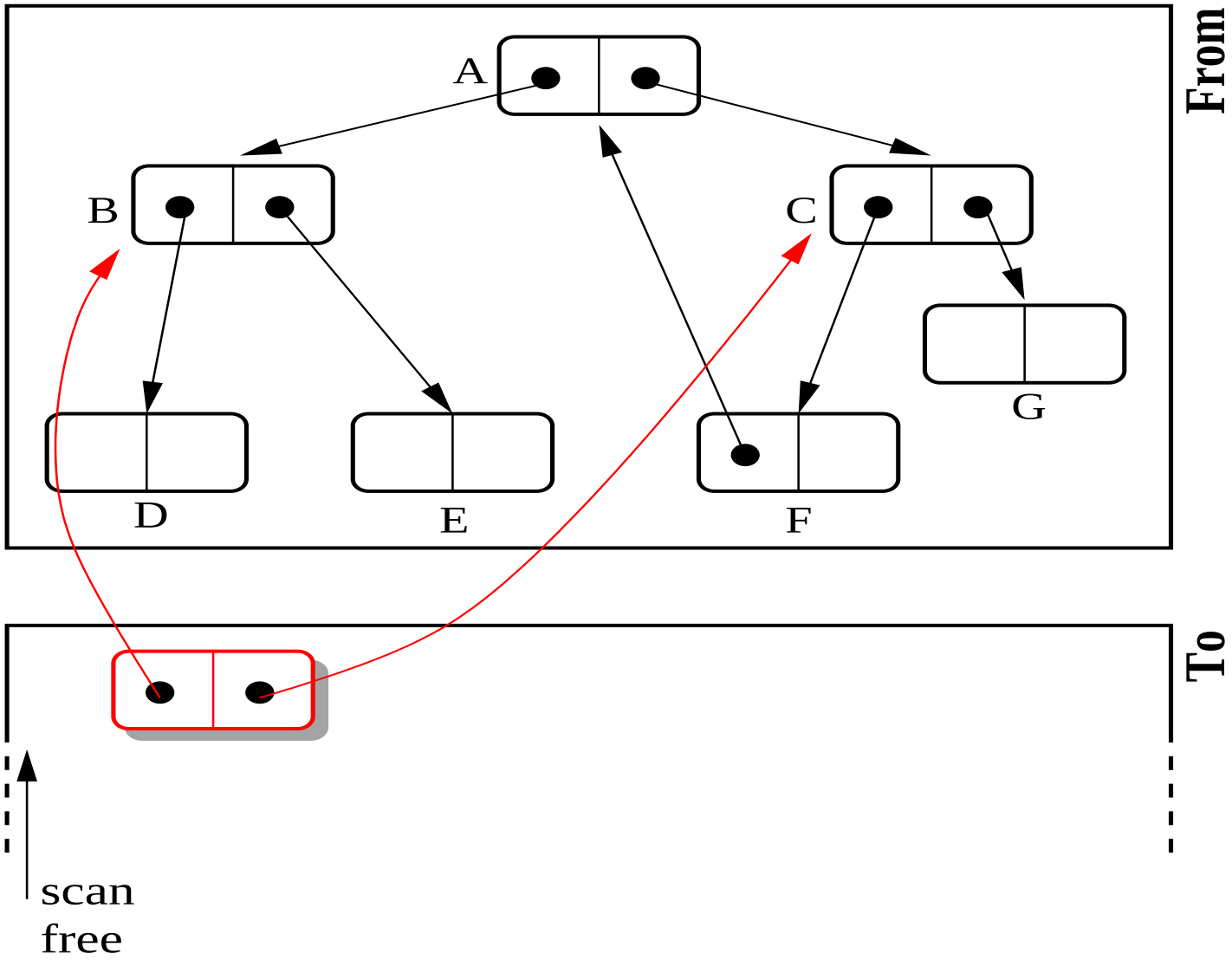
GC terminates when all reachable nodes are black

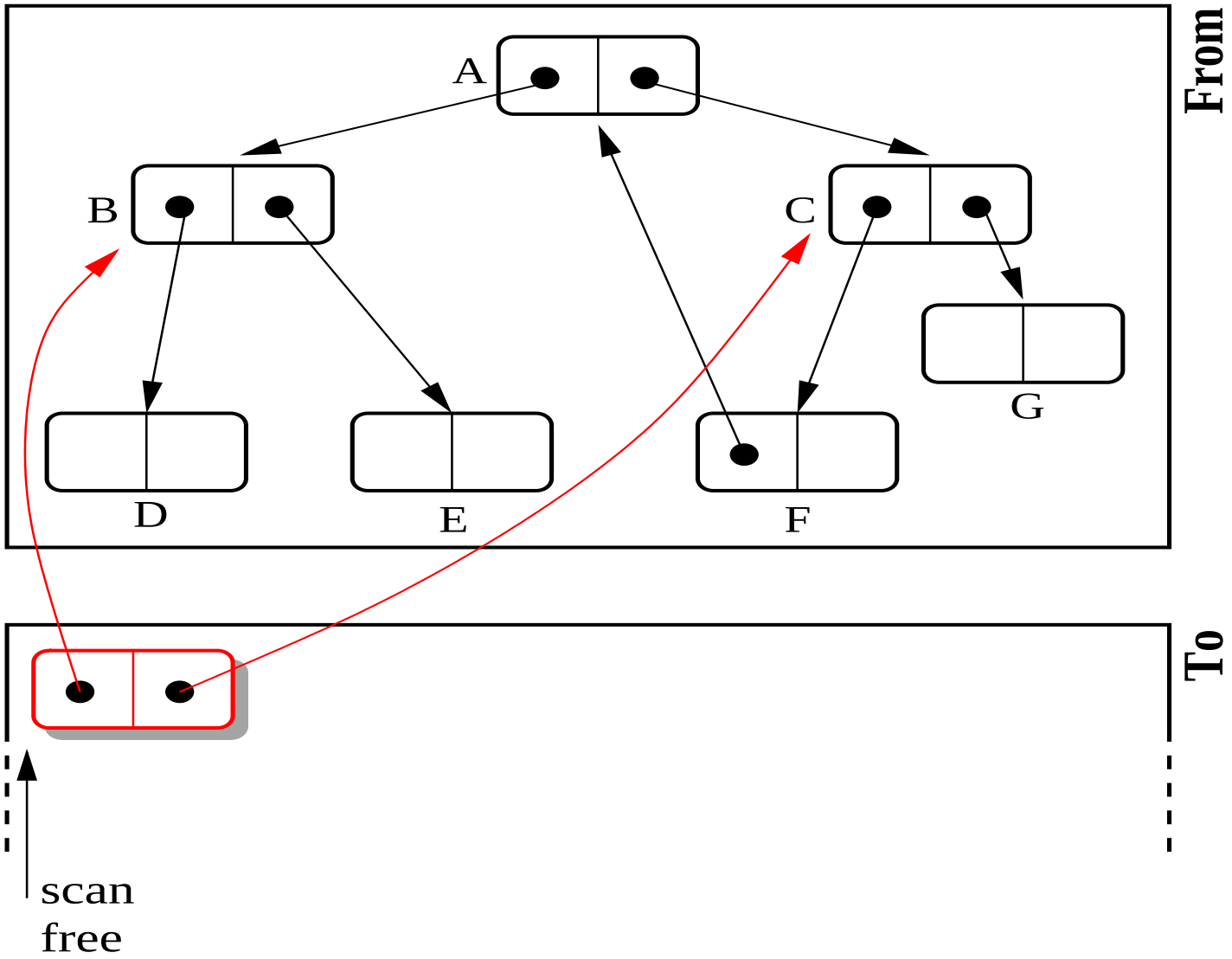

From
A
B
C
D
E
F
G
To
scan
free

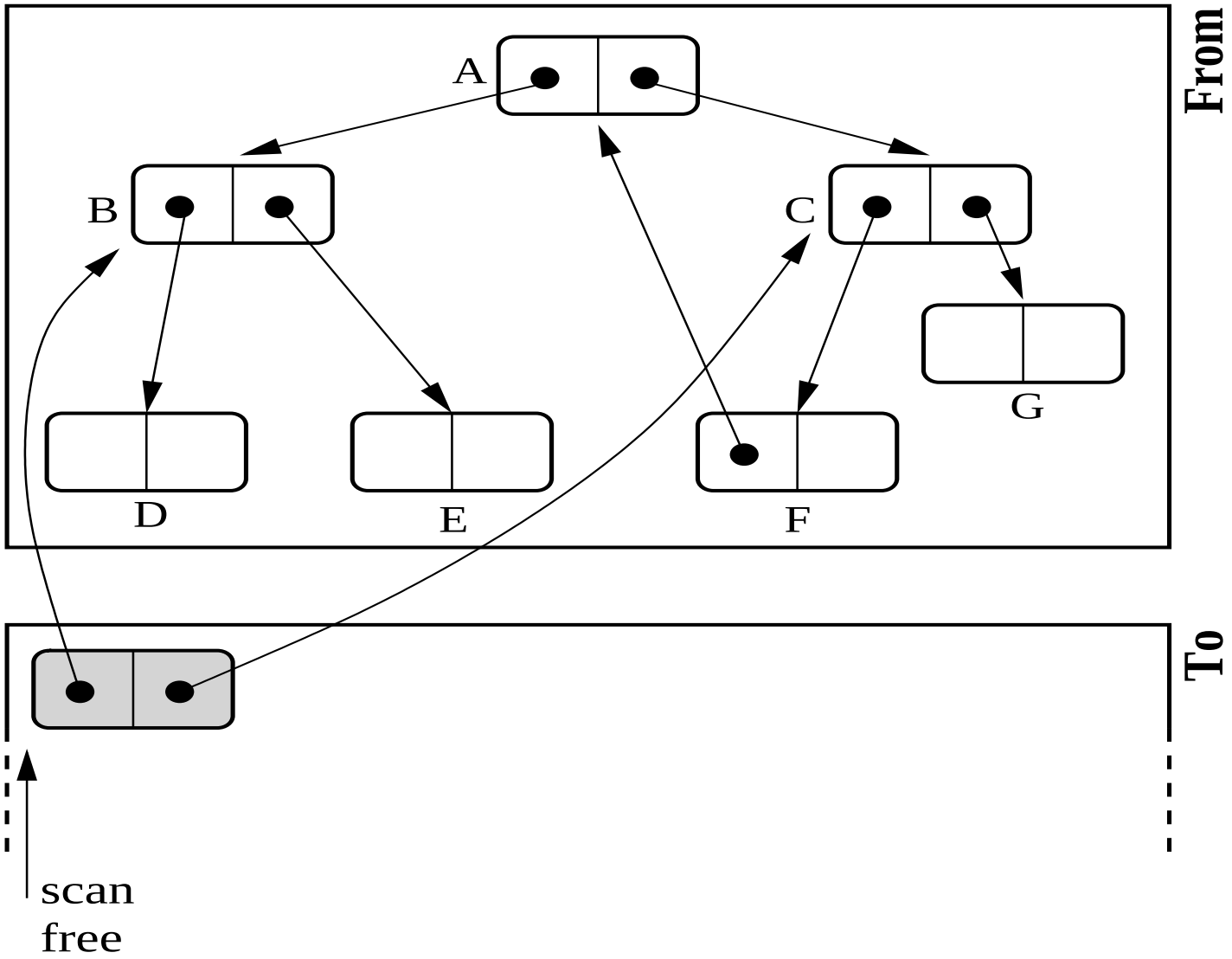

From
A
B
C
D
E
F
G
To
scan
free

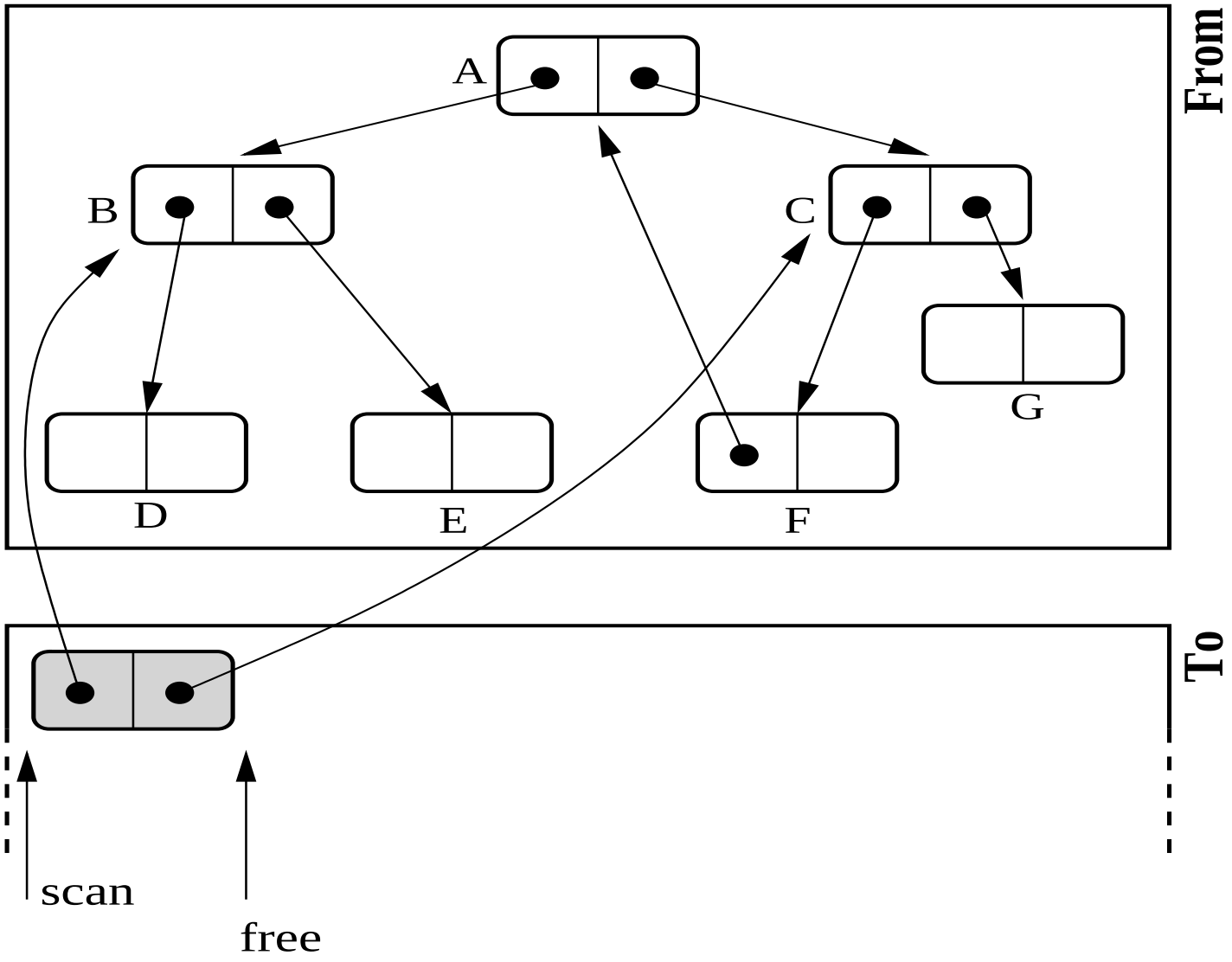

From
A
B
C
D
E
F
G
To
scan
free

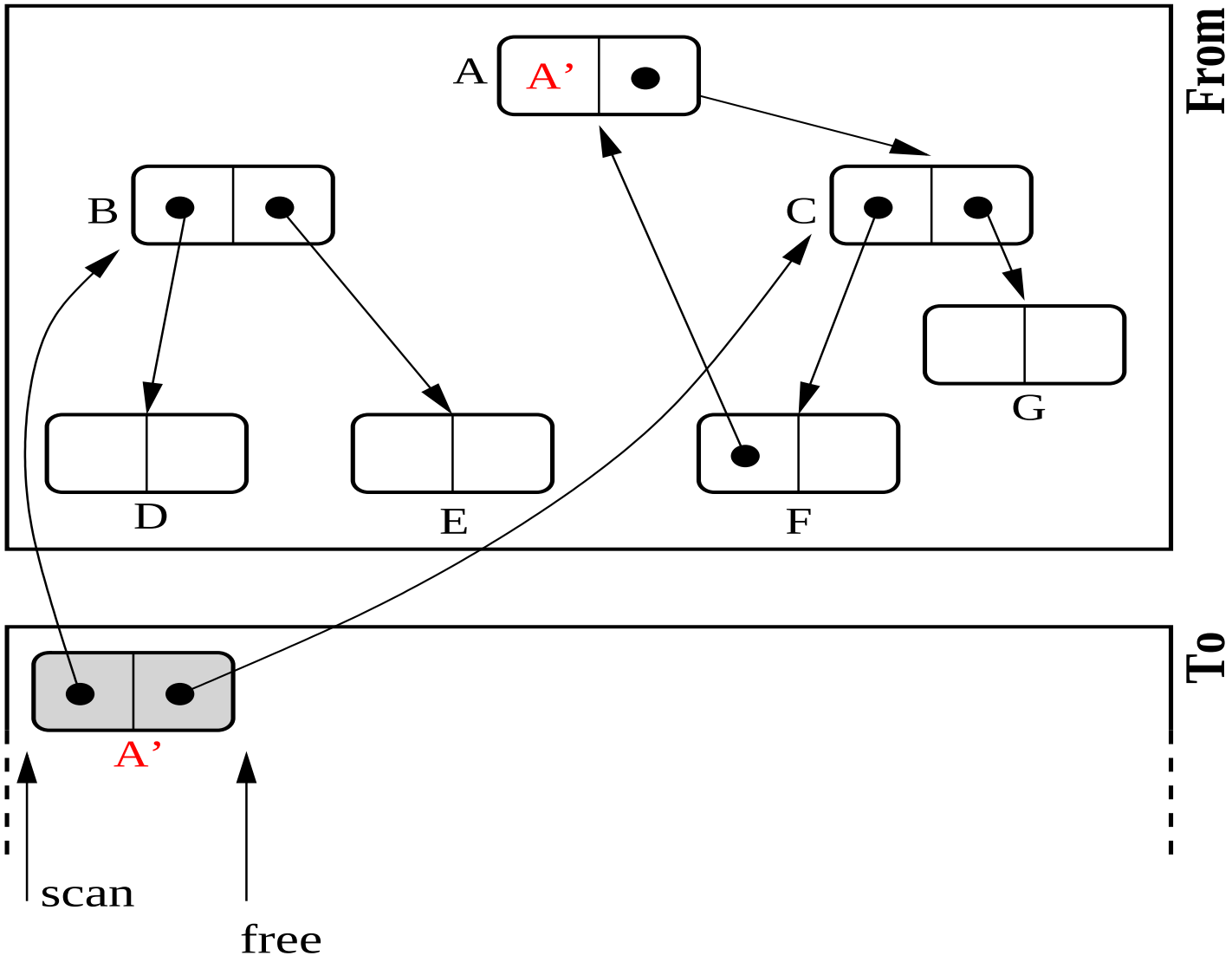

A
B
C
D
E
F
G
From
To
scan
free

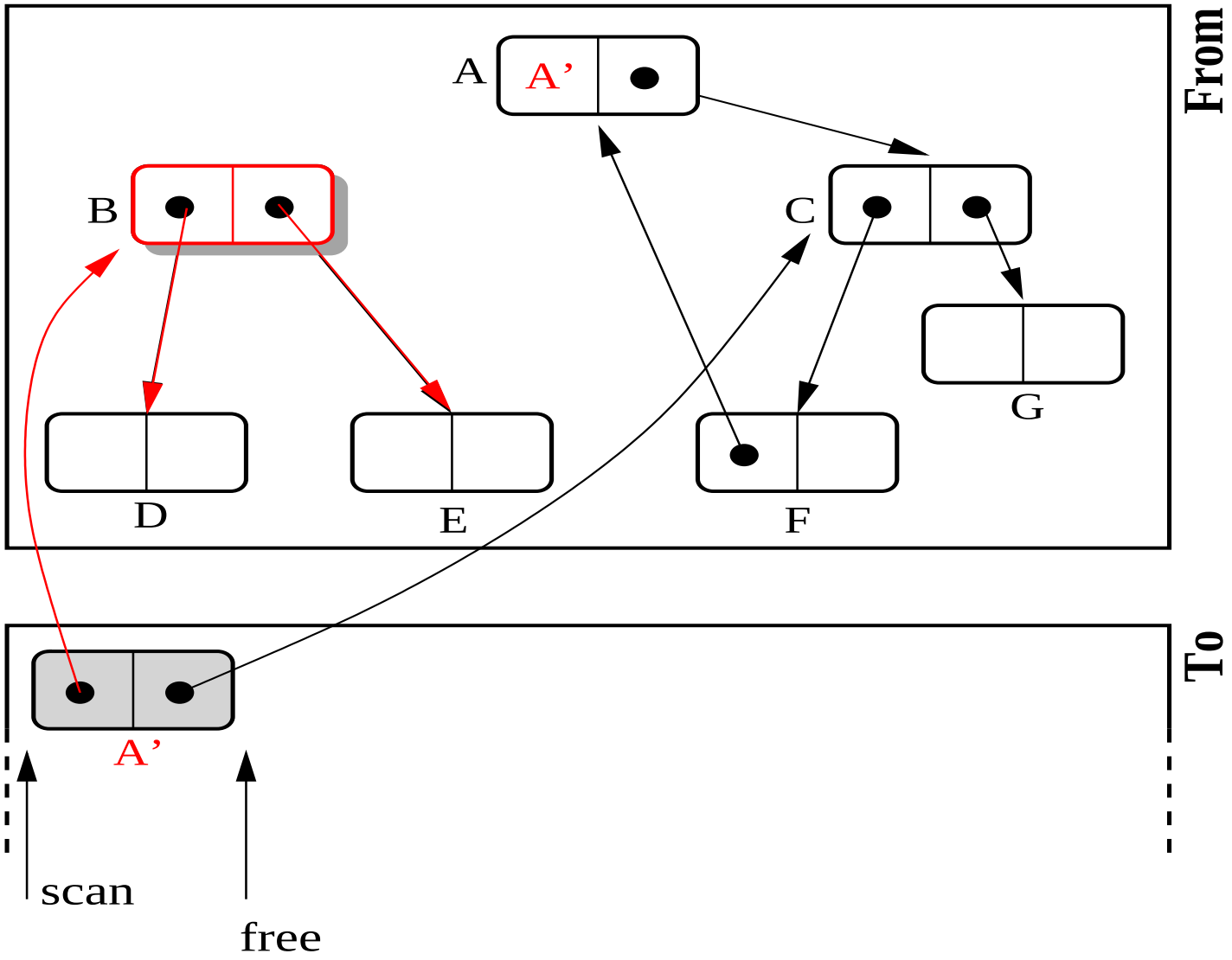

From
A
B
C
D
E
F
G
To
scan
free

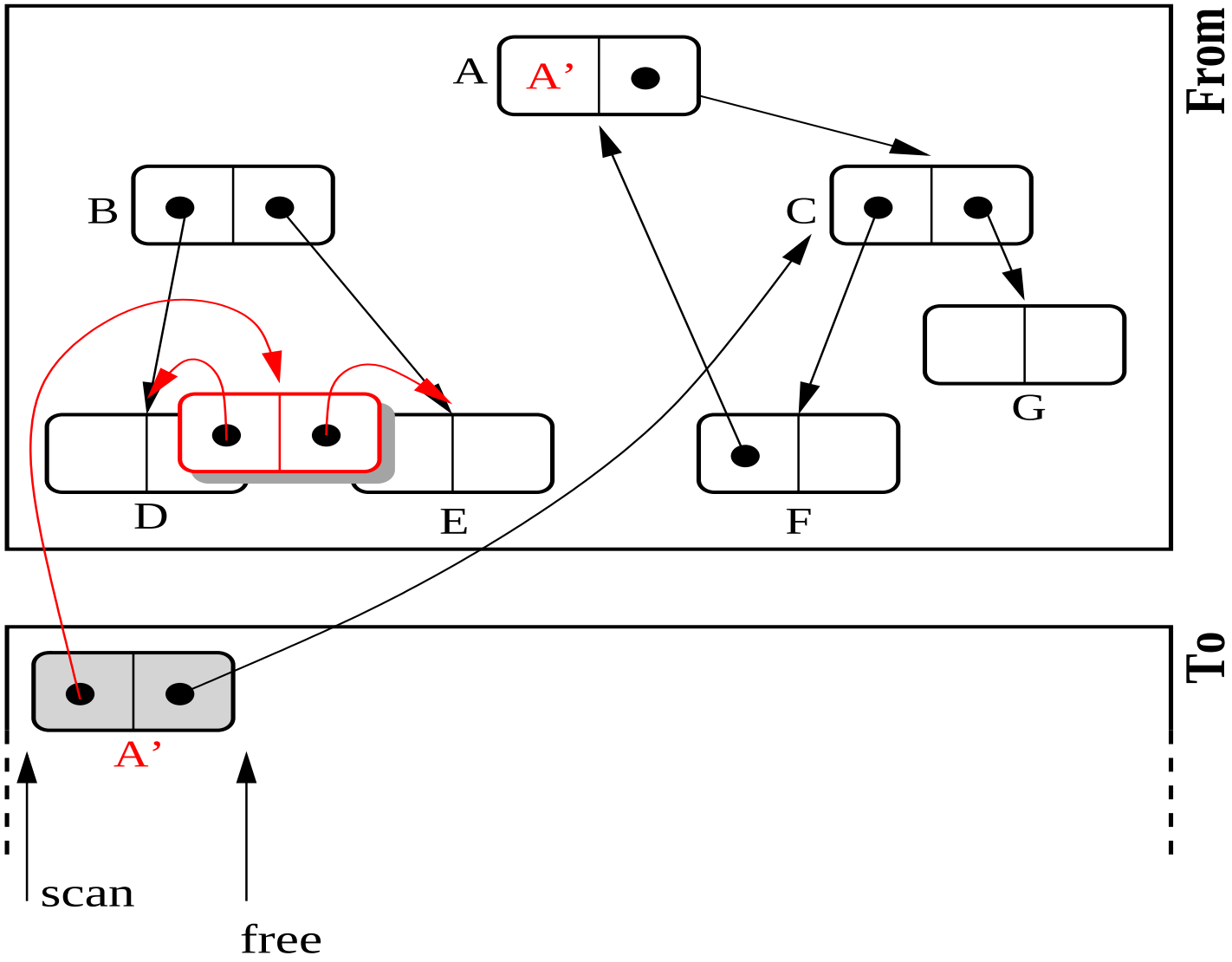

From
A
B
C
D
E
F
G
To
scan
free

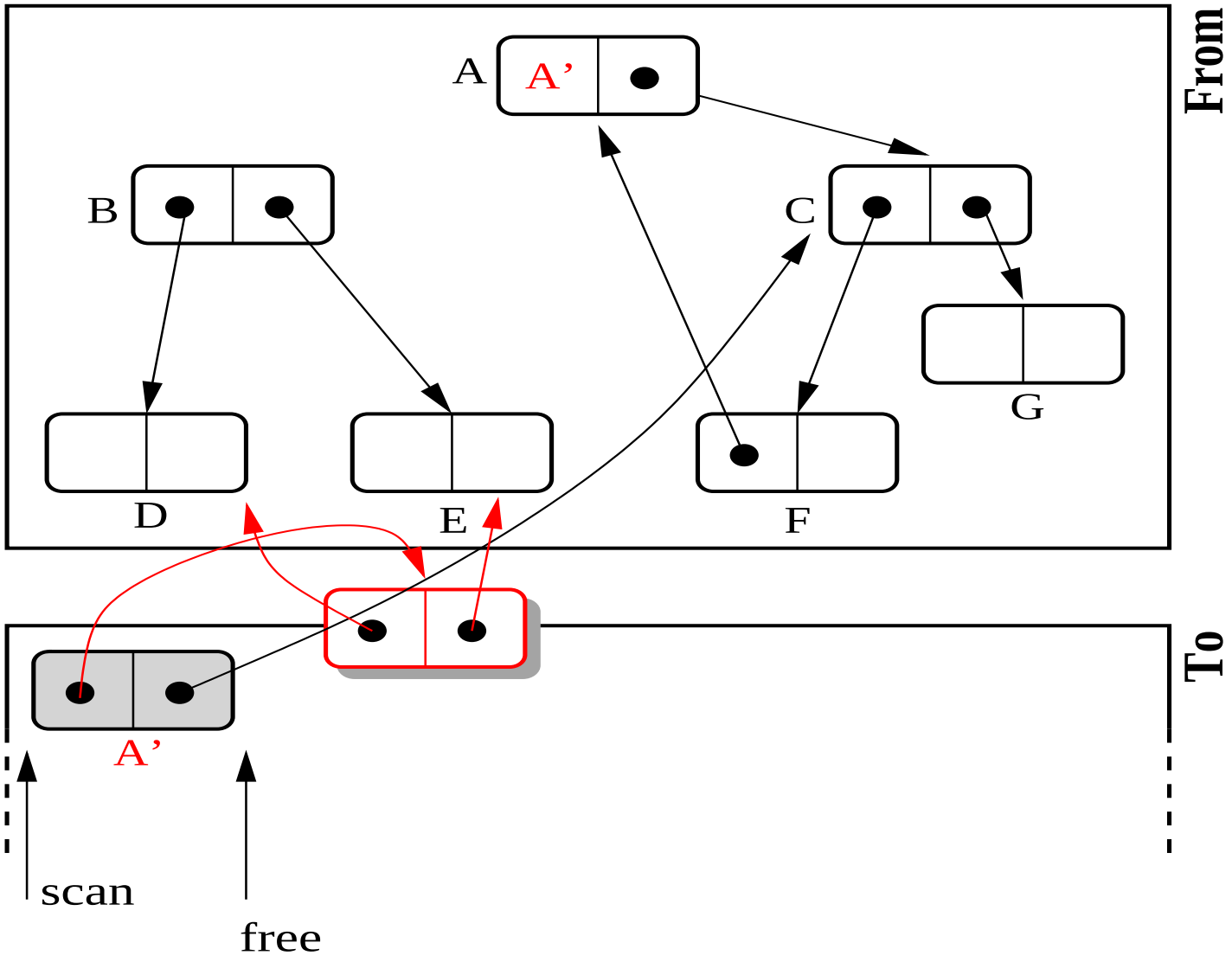

From
A
B
C
D
E
F
G
To
scan
free

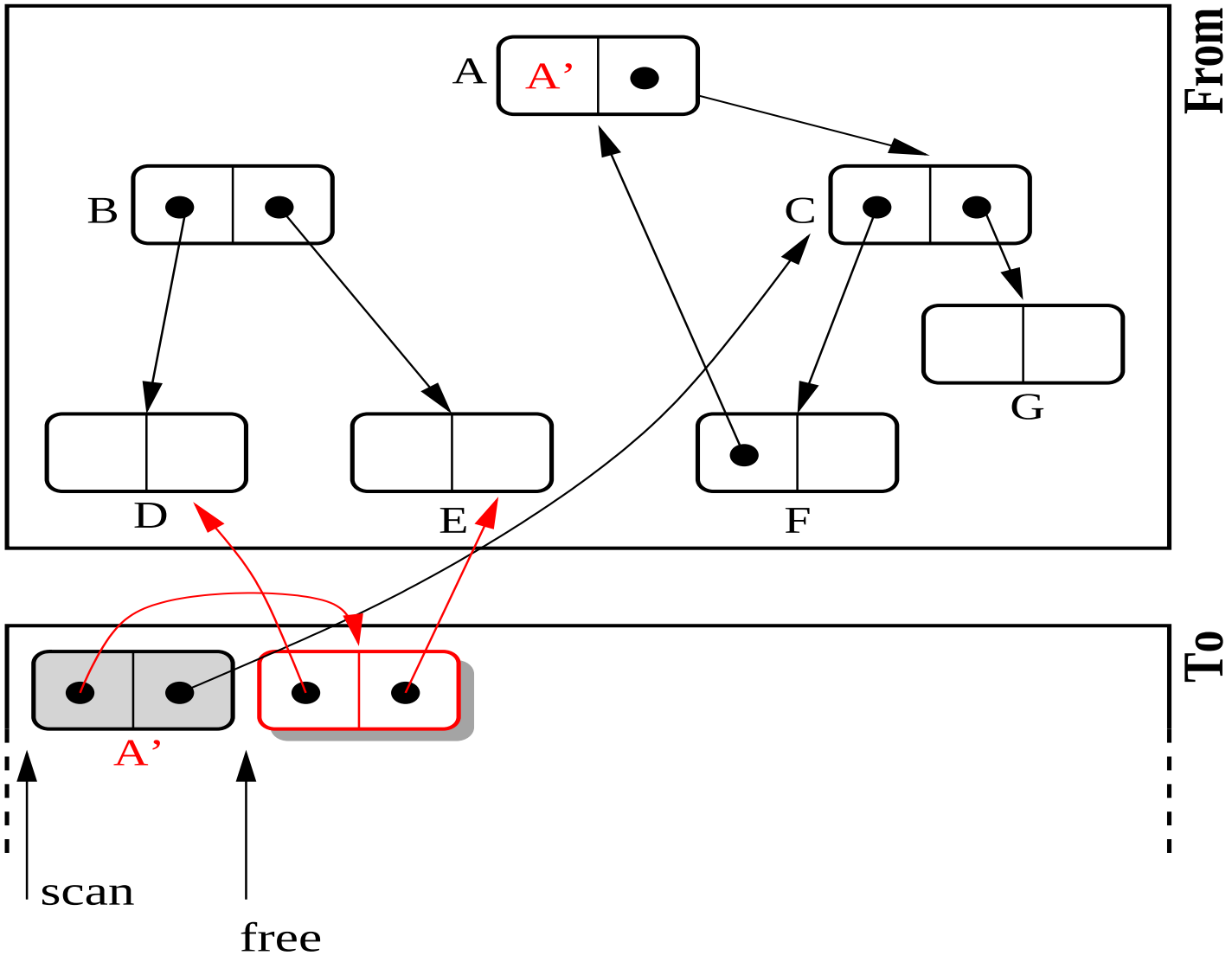

From
A
B
C
D
E
F
G
To
scan
free

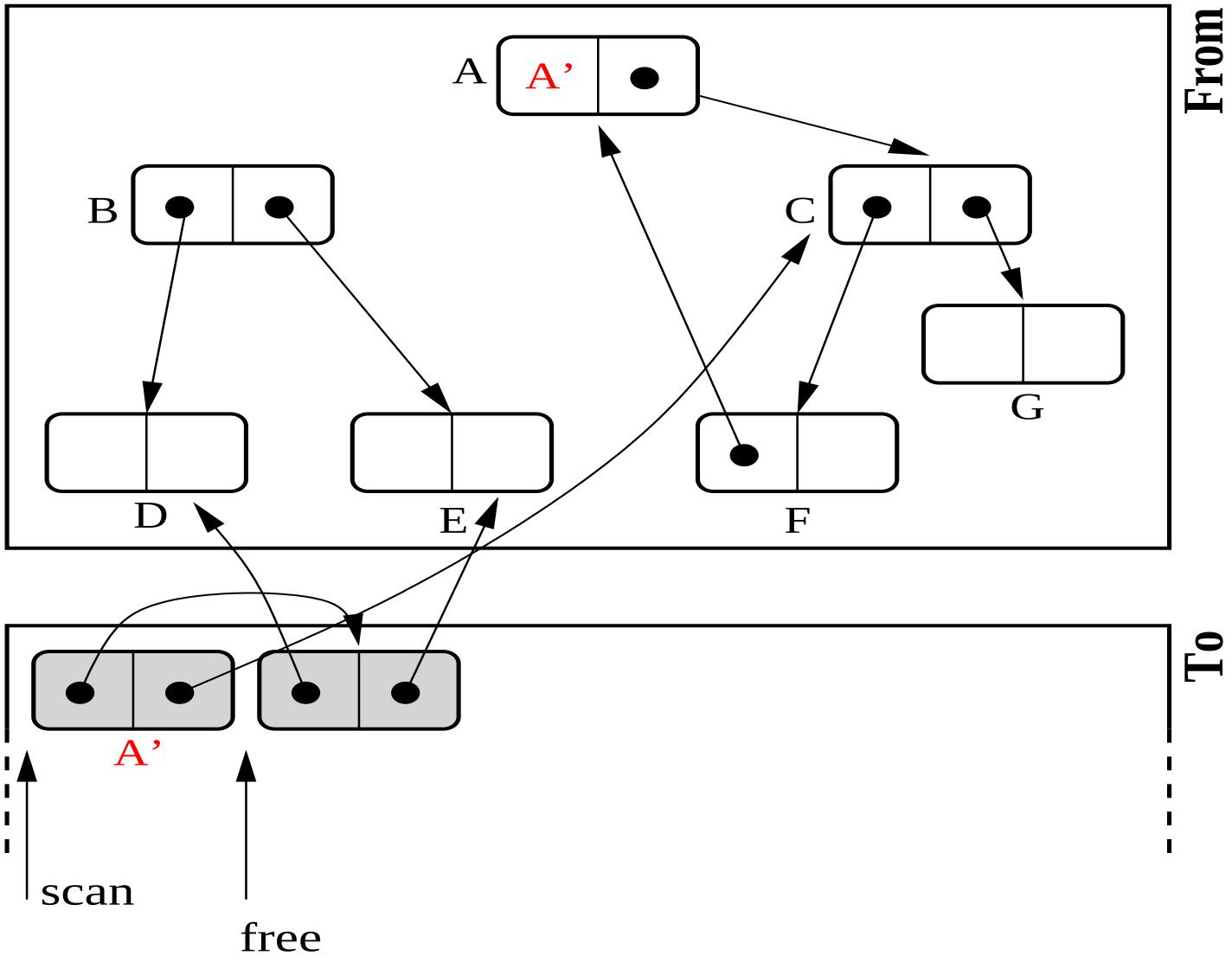

From
A
A'
B
C
G
D
E
F
To
A'
scan
free

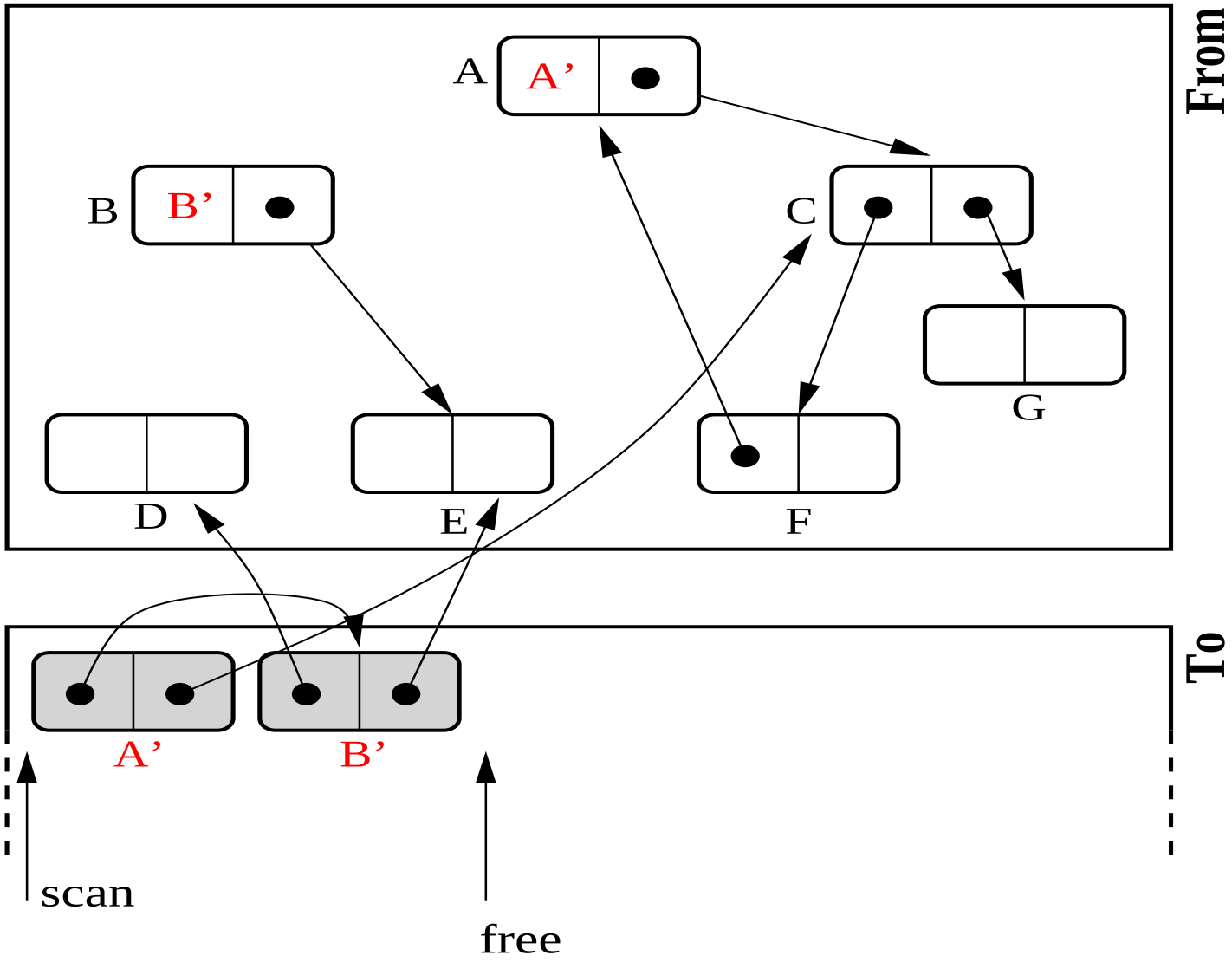

From
A
A’
B
C
G
D
E
F
To
A’
scan
free

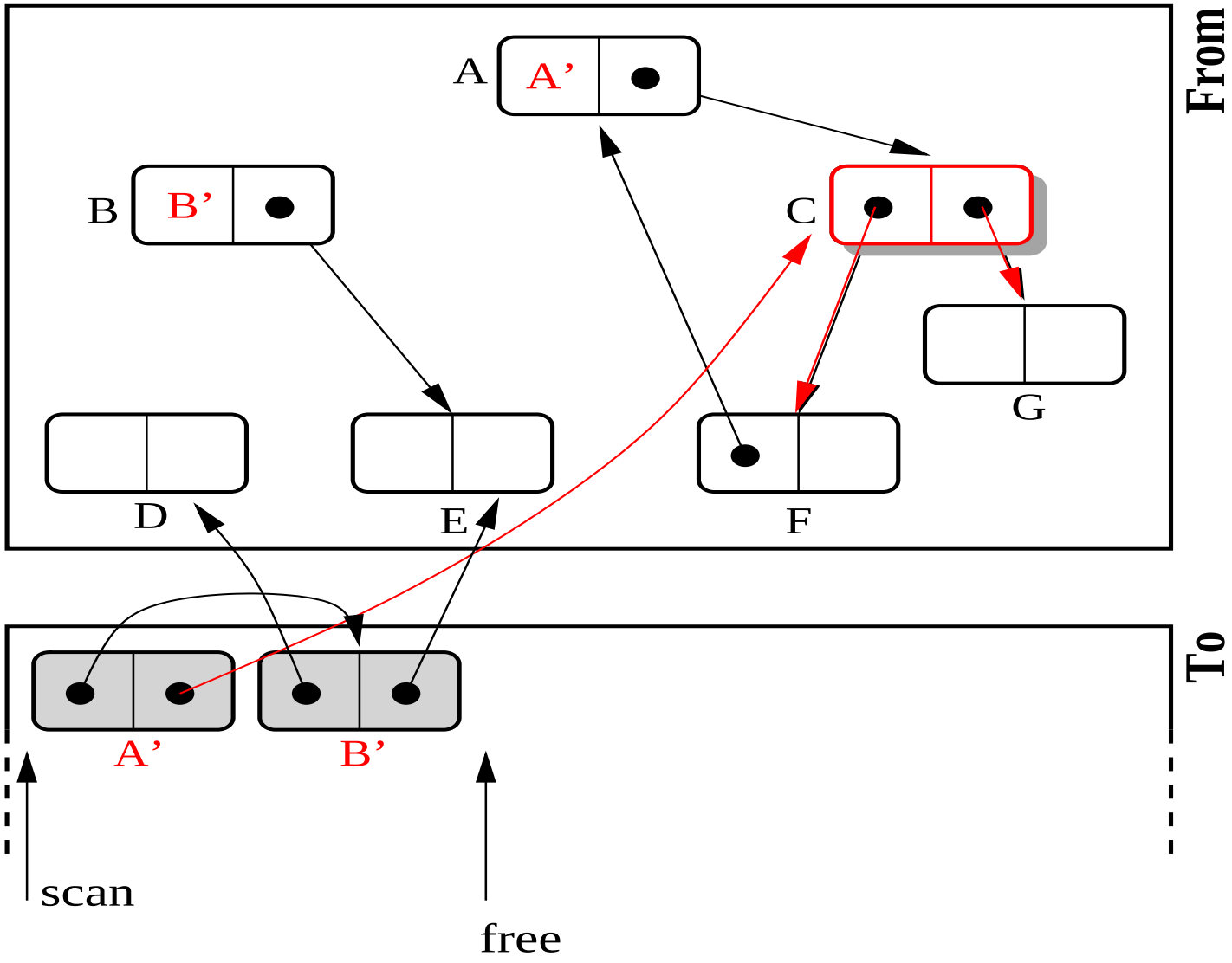

From
A
A'
B
C
G
D
E
F
To
A'
scan
free

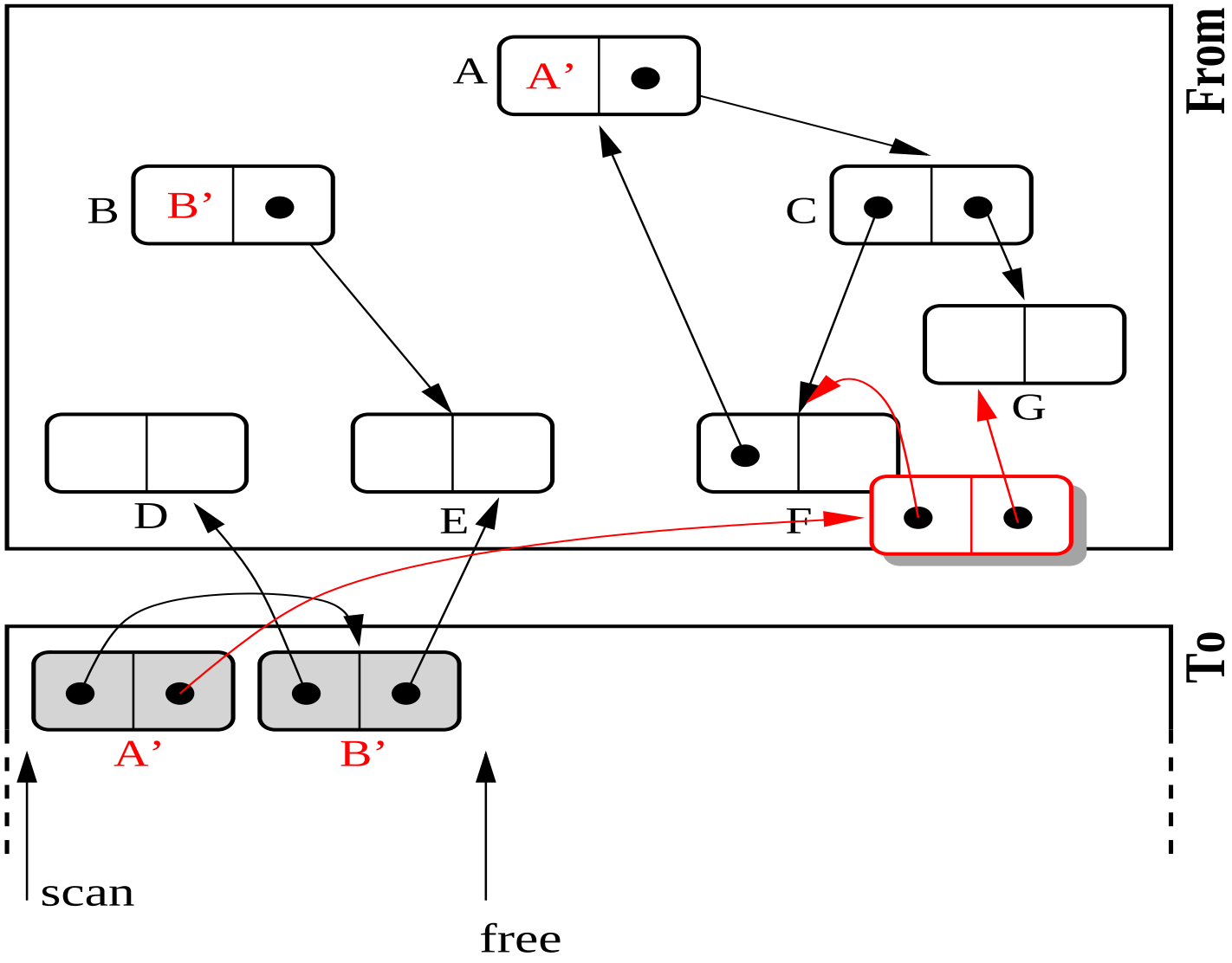

From
A
A’
B
C
G
D
E
F
To
A’
scan
free

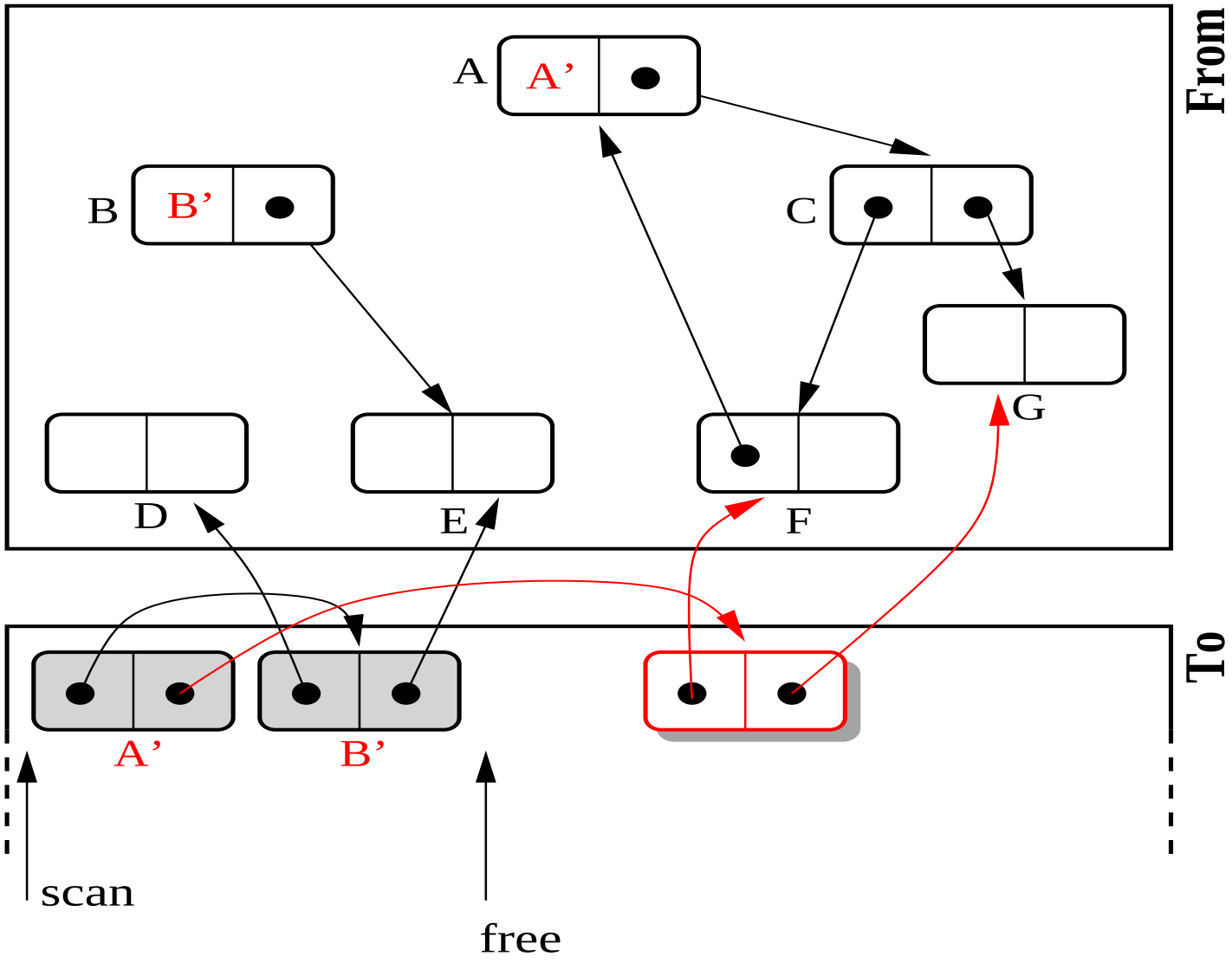

From
A
A’
B
C
G
D
E
F
To
A’
scan
free

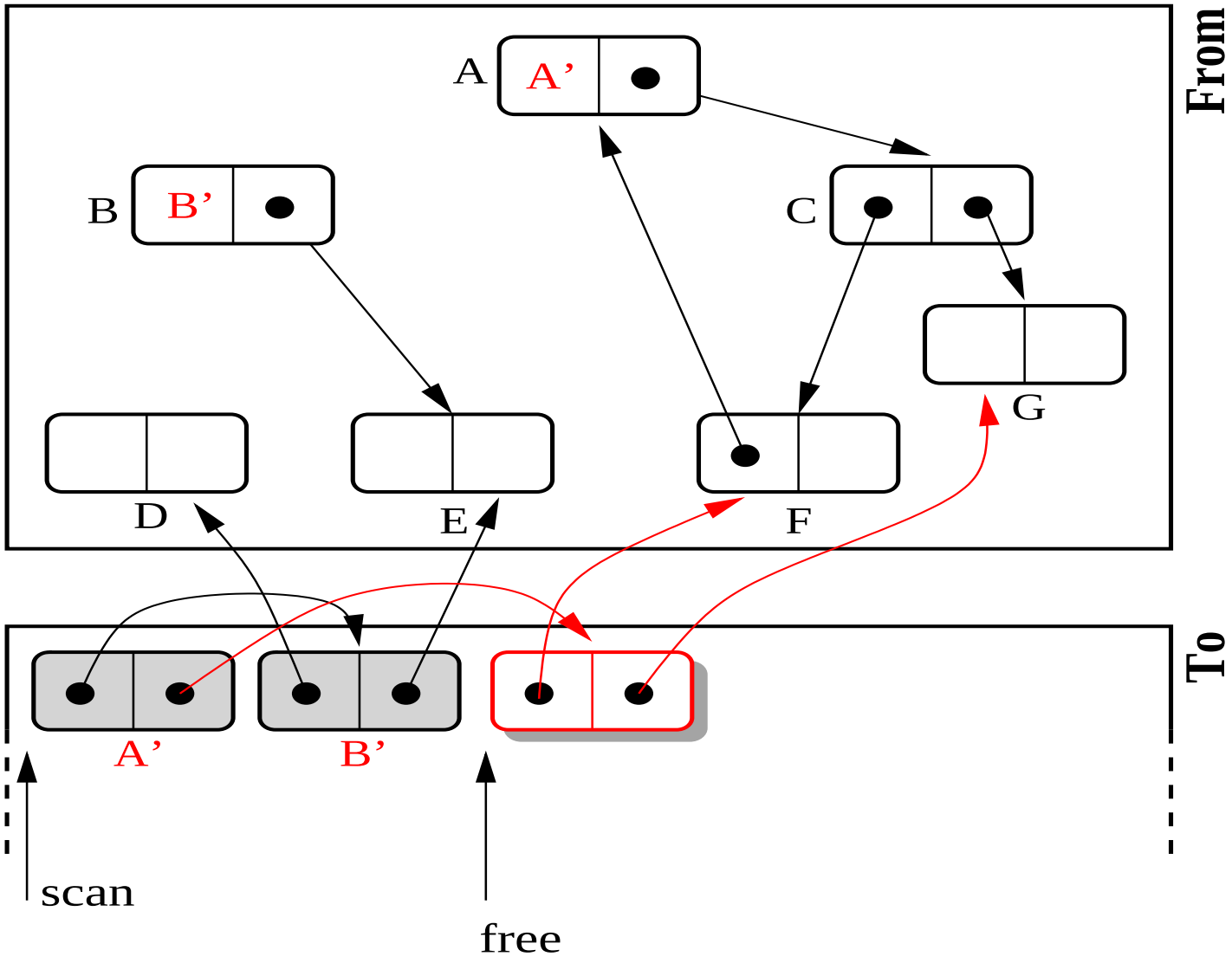

From
A
A'
B
C
G
D
E
F
To
A'
scan
free

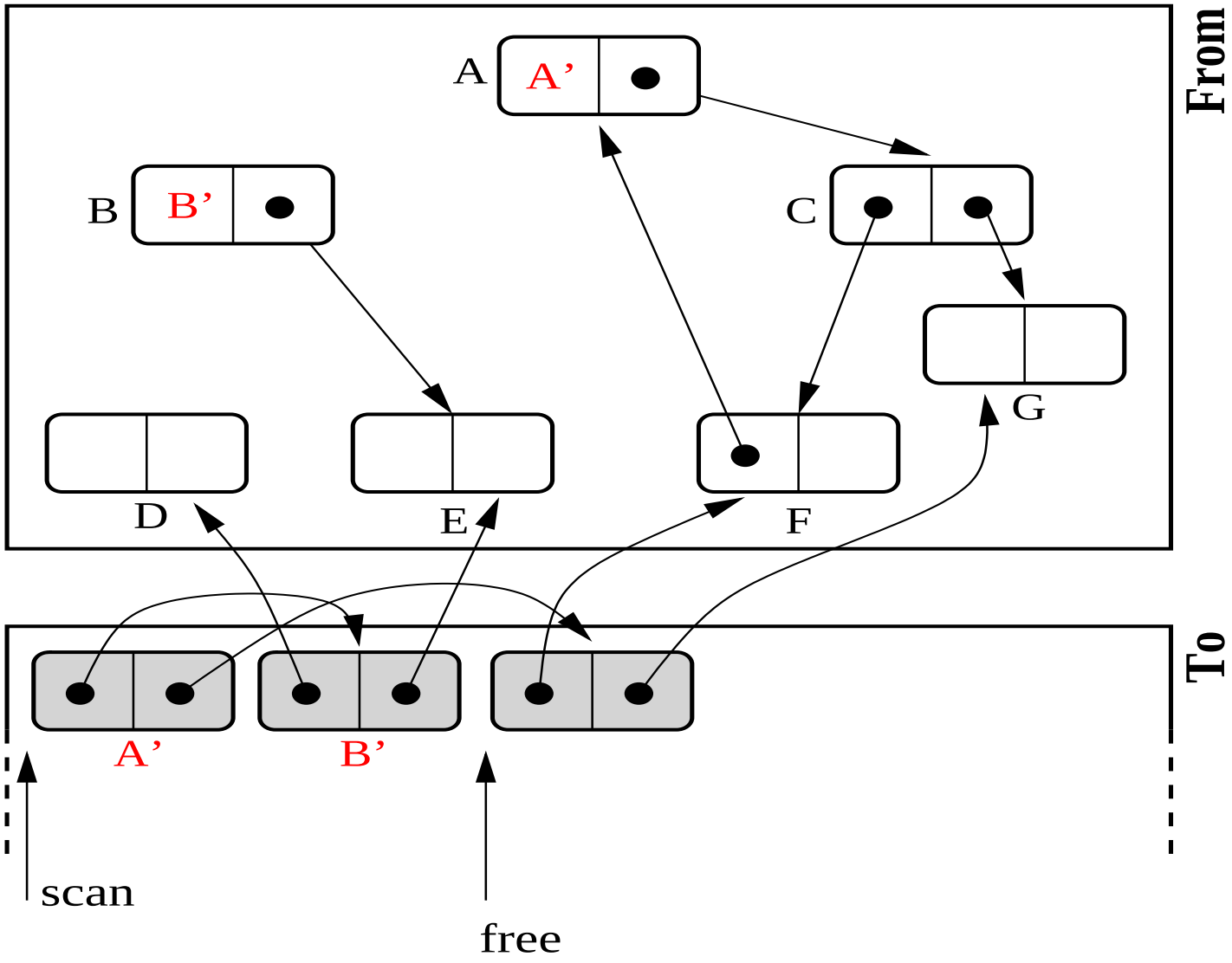

From
A
A'
B
B'
C
G
D
E
F
To
A'
B'
scan
free

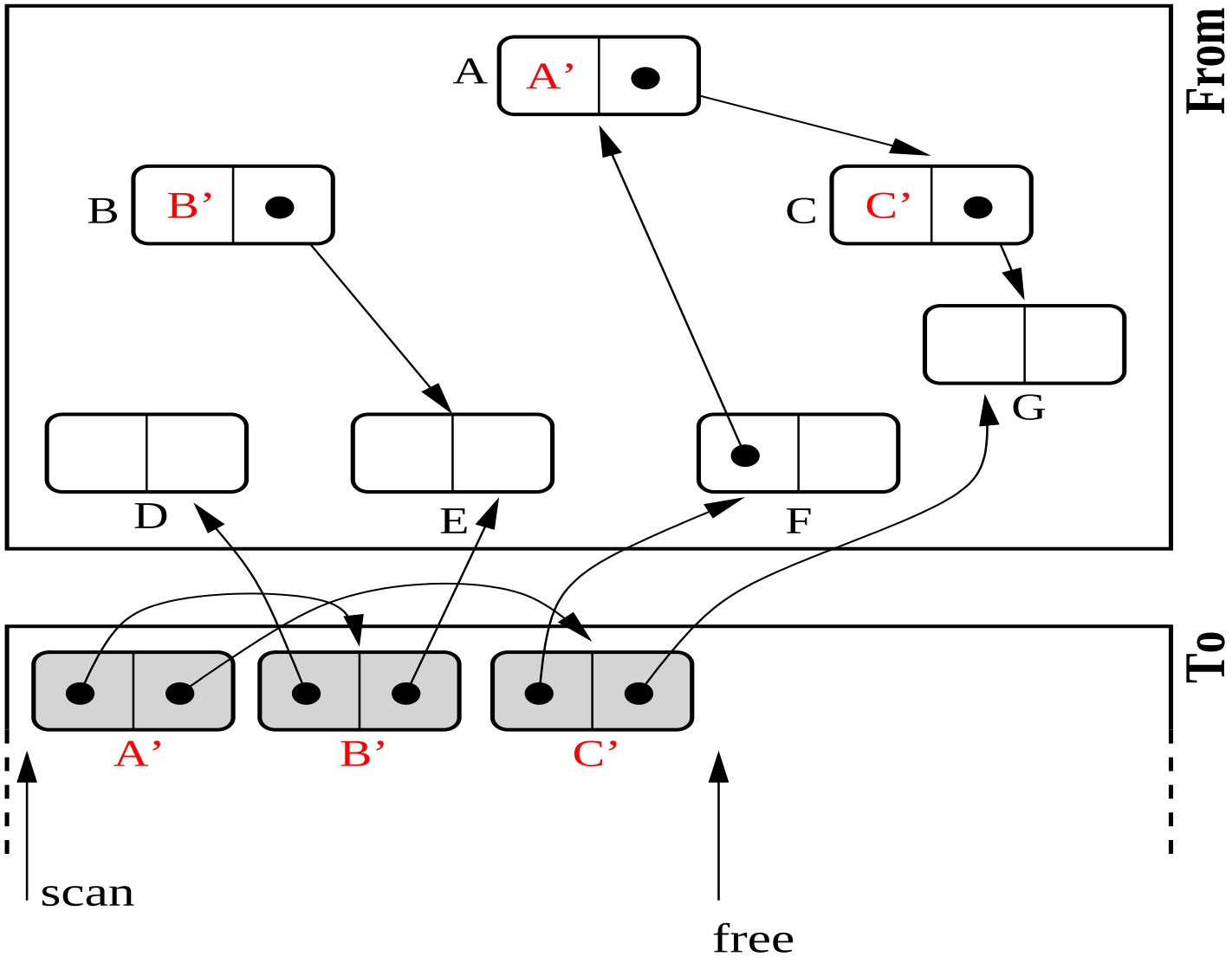

From
A
A’
B
B’
C
G
D
E
F
To
A’
B’
scan
free

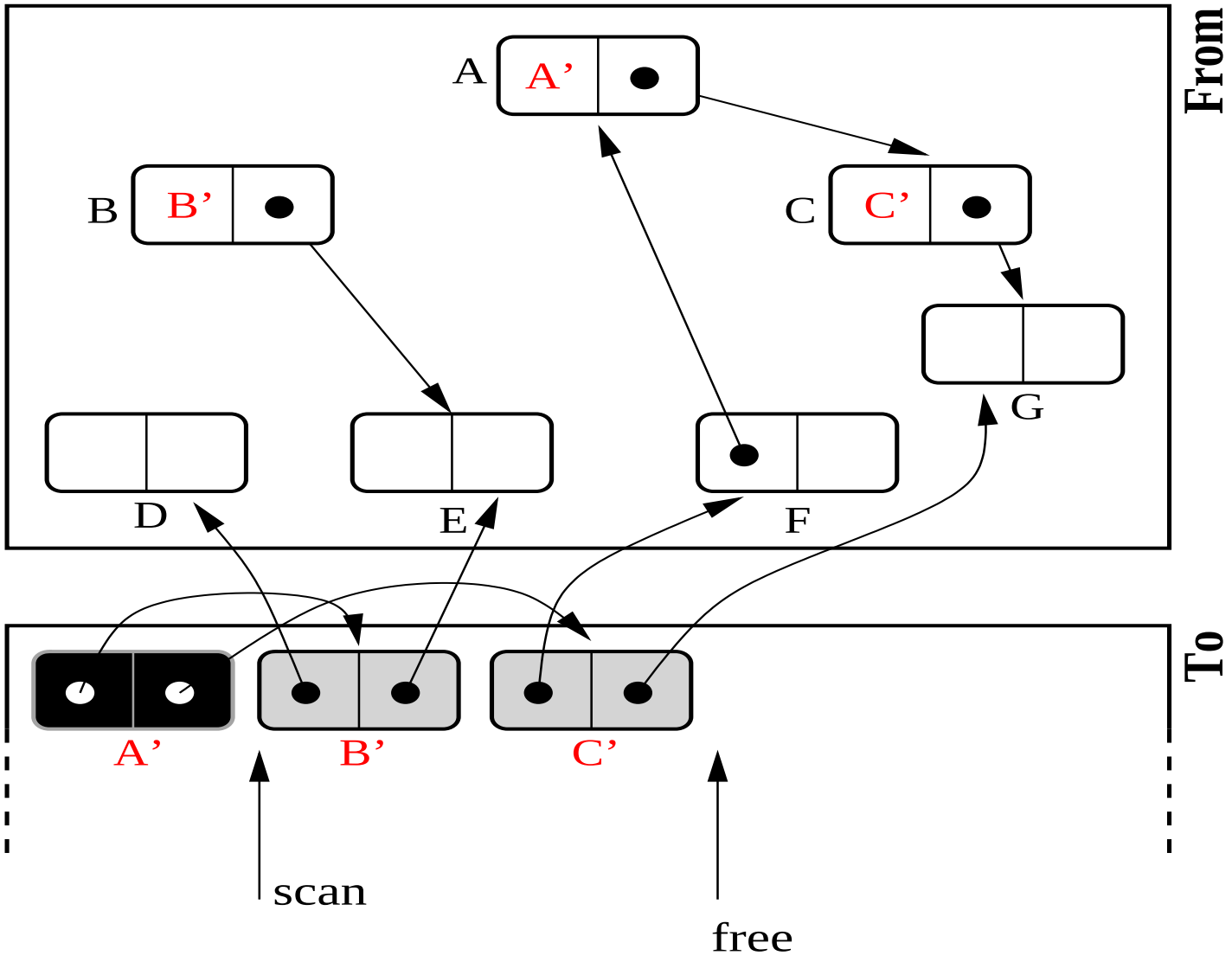

From
A
A’
B
B’
C
G
D
E
F
To
A’
B’
scan
free

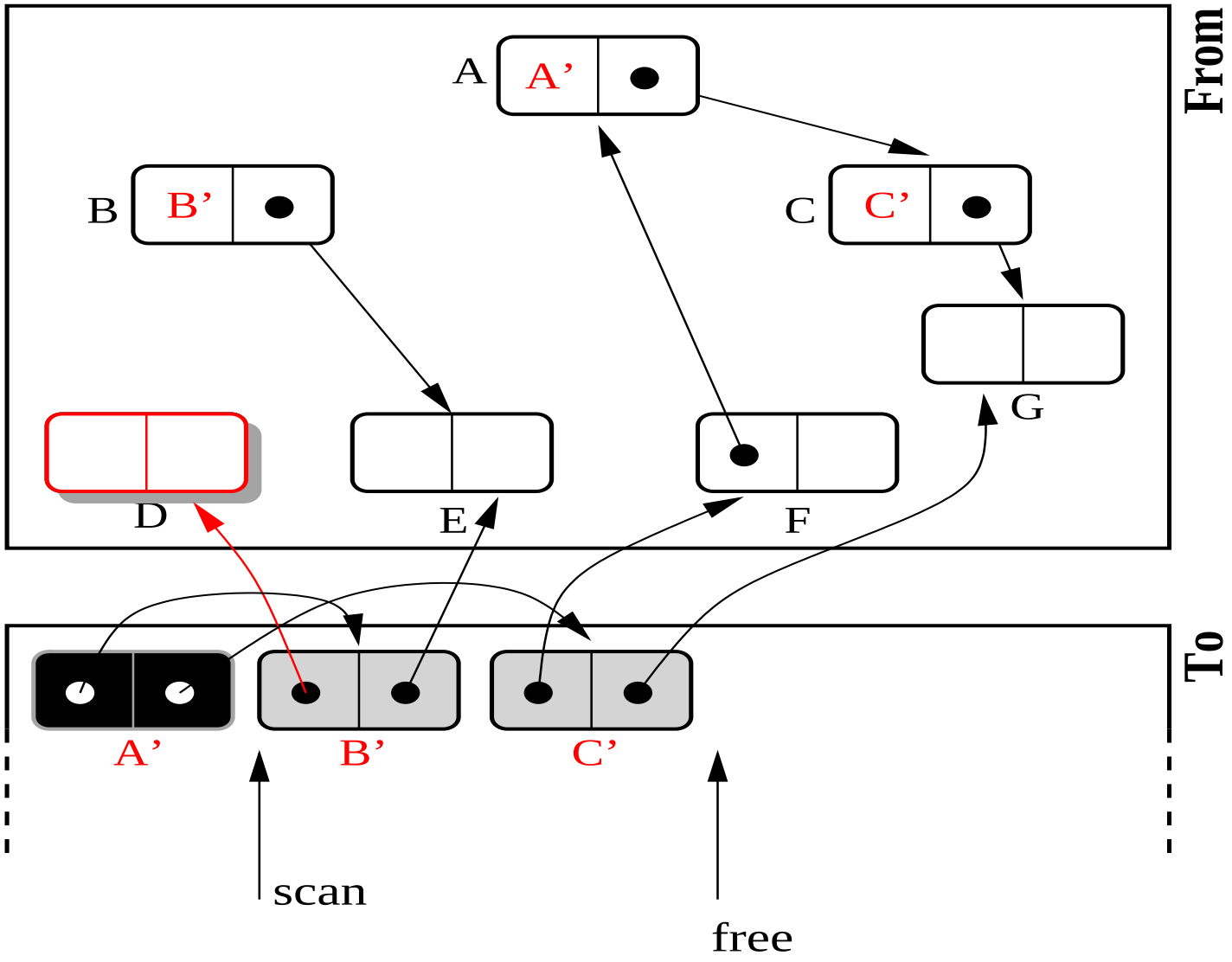

From
A
A’
B
B’
C
G
D
E
F
To
A’
B’
scan
free

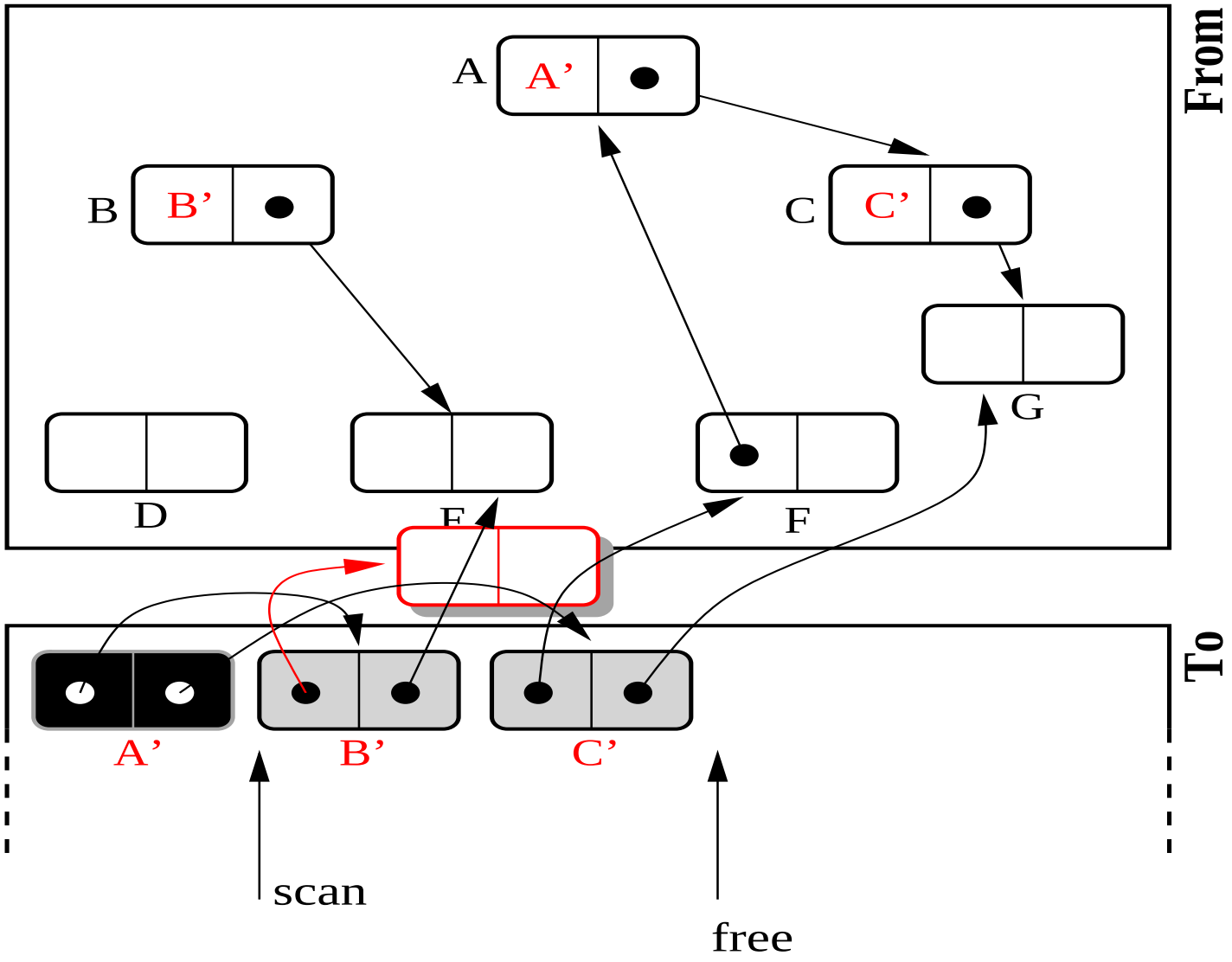

From
A
A'
B
B'
C
G
D
E
F
To
A'
B'
scan
free

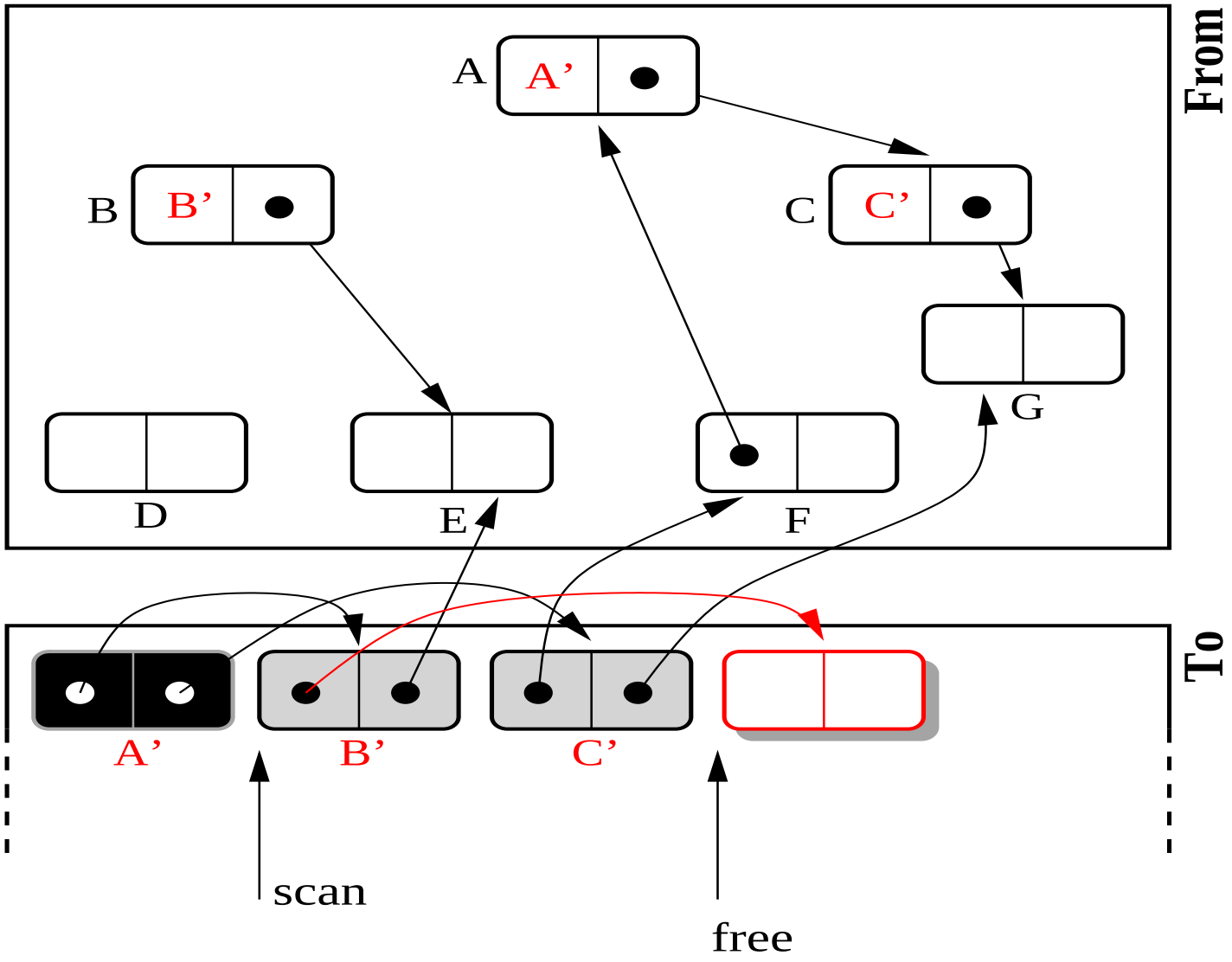

From
A
A'
B
B'
C
G
D
E
F
To
A'
B'
scan
free

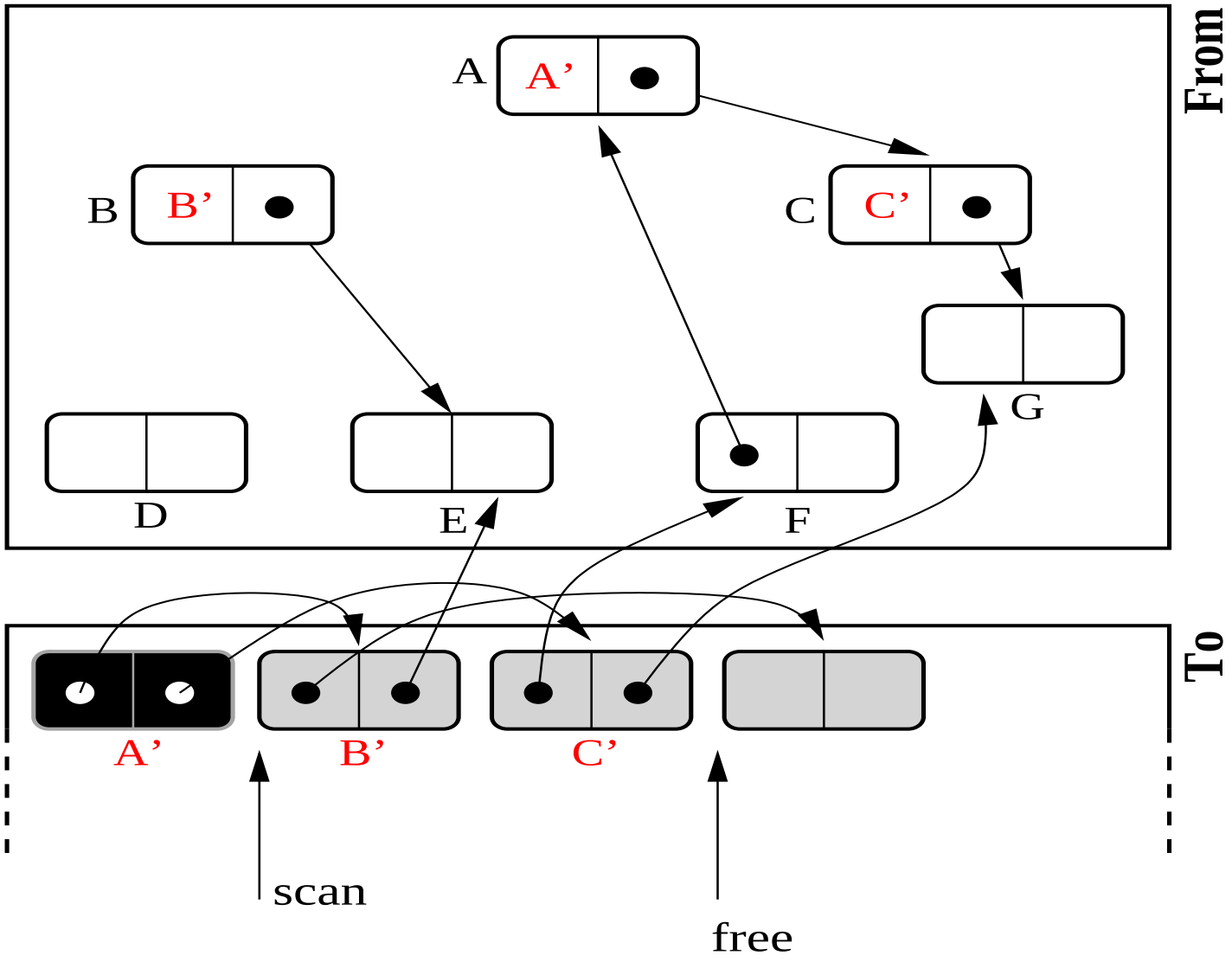

From
A
A’
B
B’
C
C’
G
D
E
F
To
A’
B’
C’
scan
free

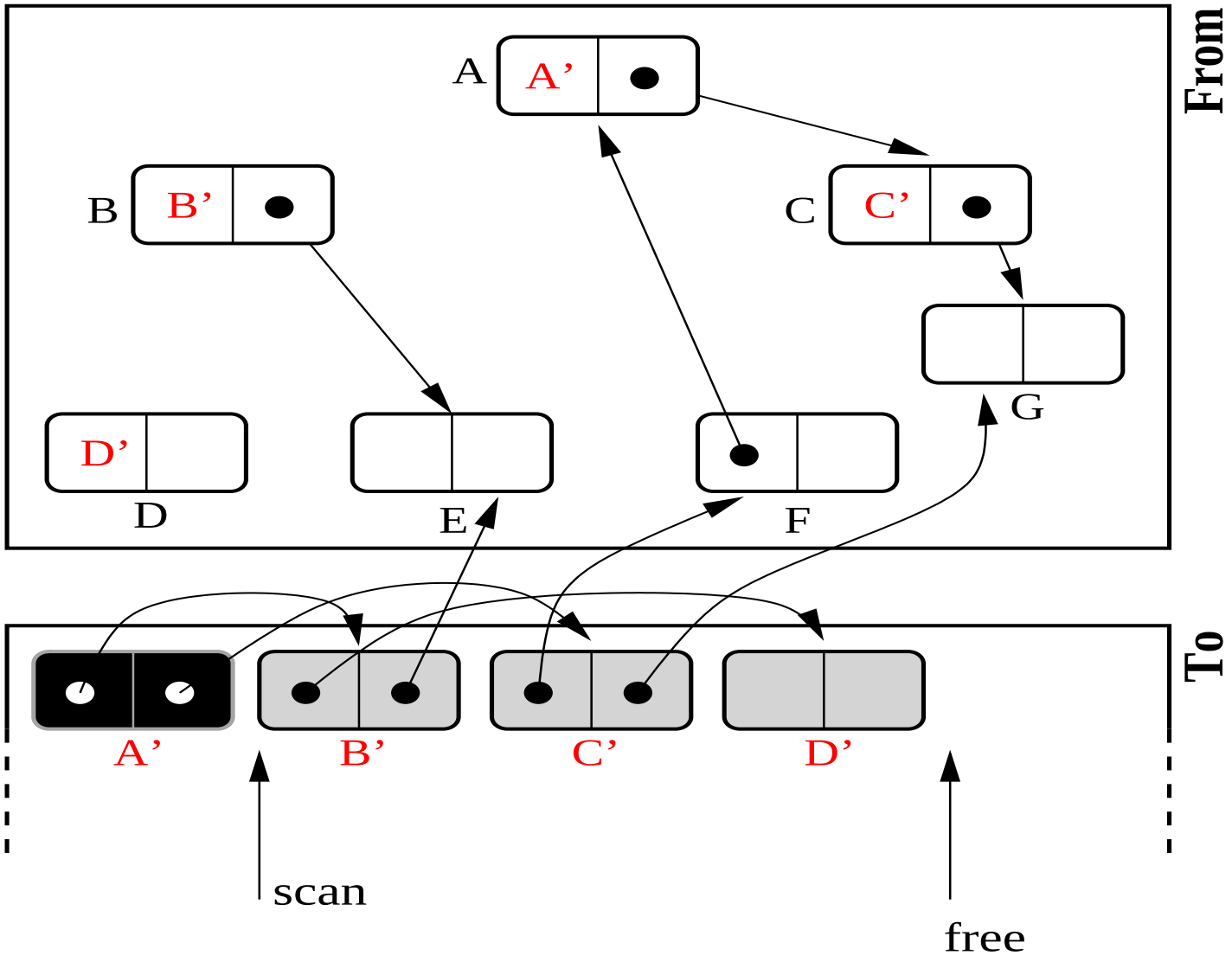

From
A
A’
B
B’
C
C’
G
D
E
F
To
A’
B’
C’
scan
free

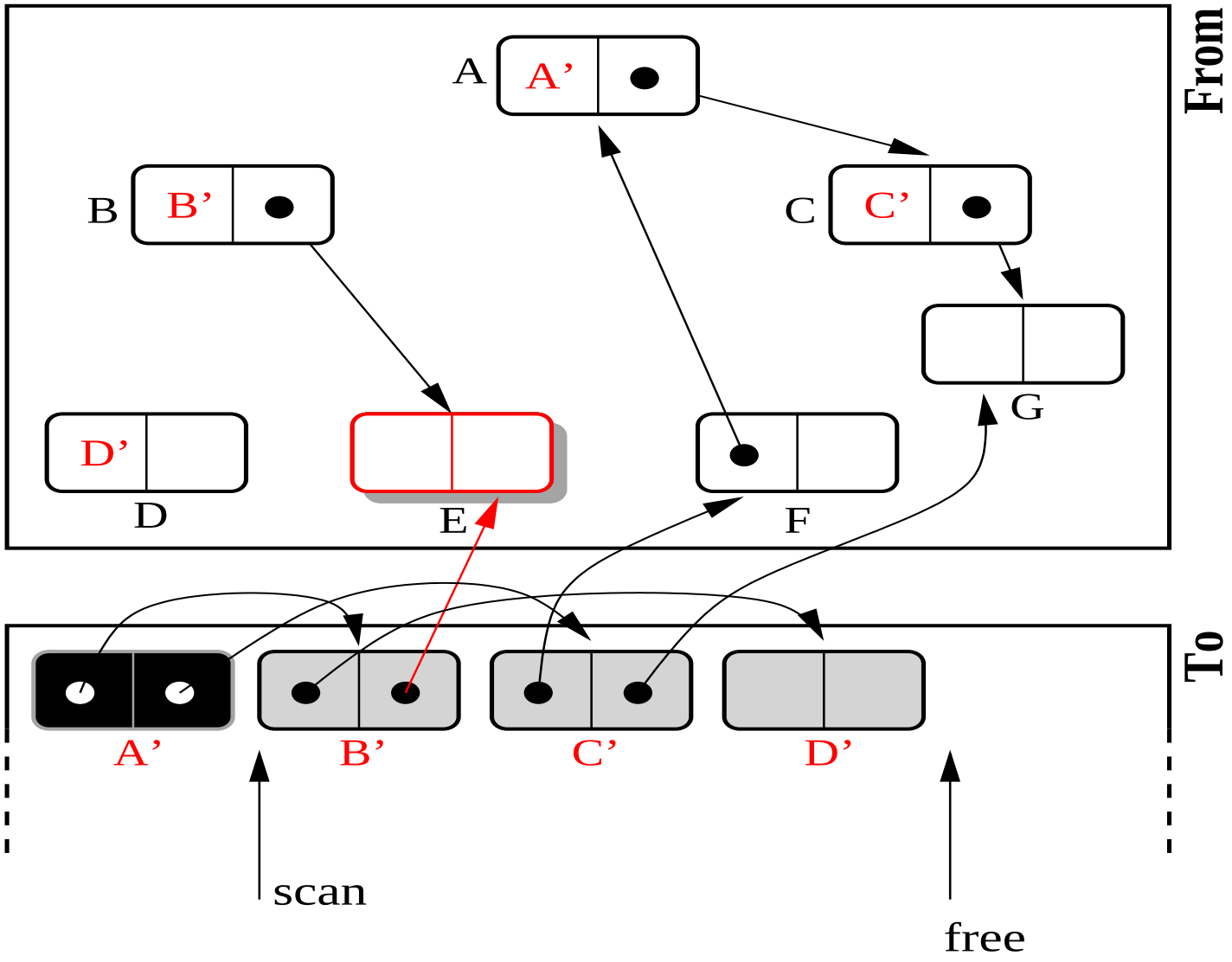

From
A
A'
B
B'
C
C'
G
D
E
F
To
A'
B'
C'
scan
free

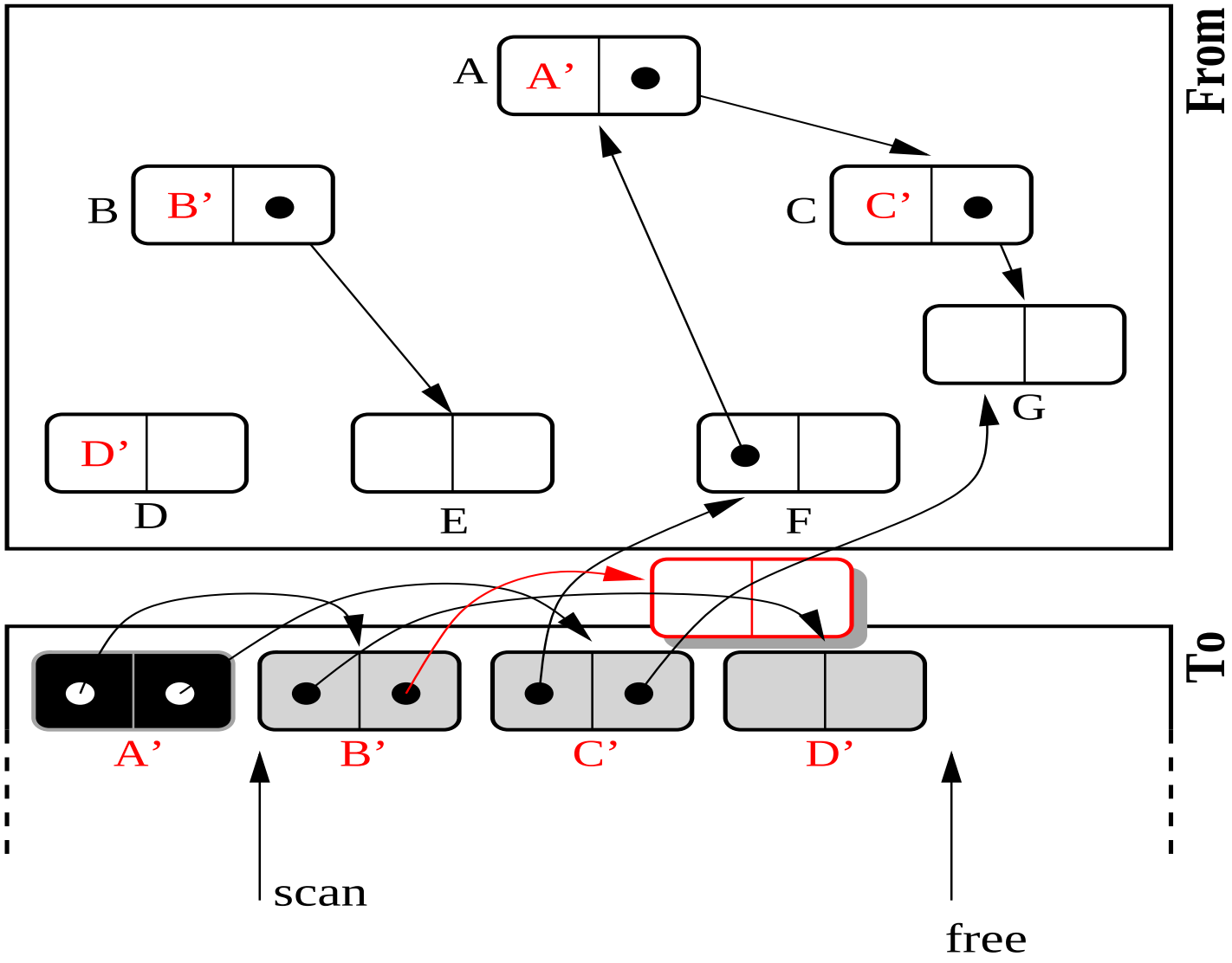

From
A
A'
B
B'
C
C'
G
D
E
F
To
A'
B'
C'
scan
free

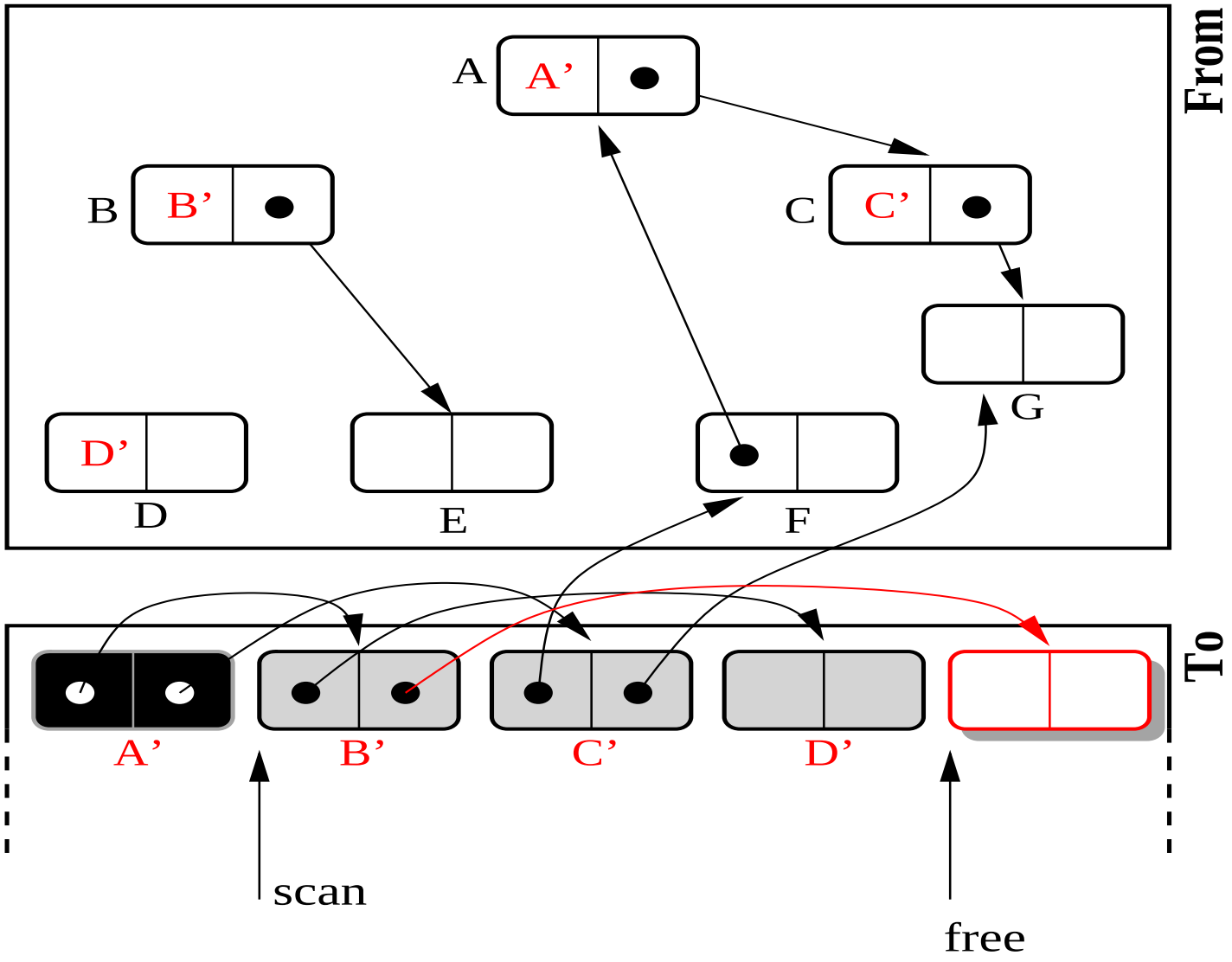

From
A
A'
B
B'
C
C'
G
D
E
F
To
A'
B'
C'
scan
free

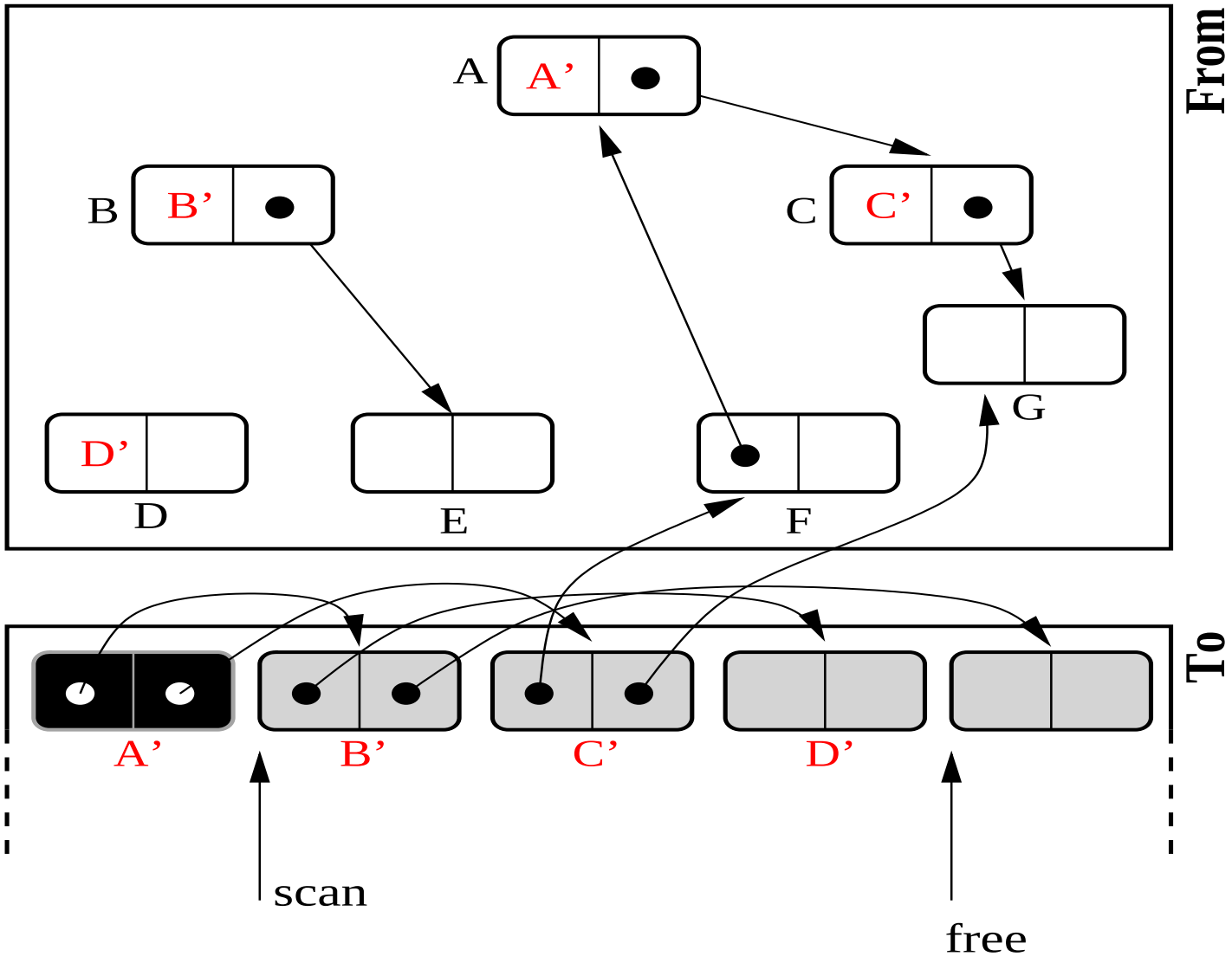

From
A
A'
B
B'
C
C'
G
D
E
F
To
A'
B'
C'
scan
free

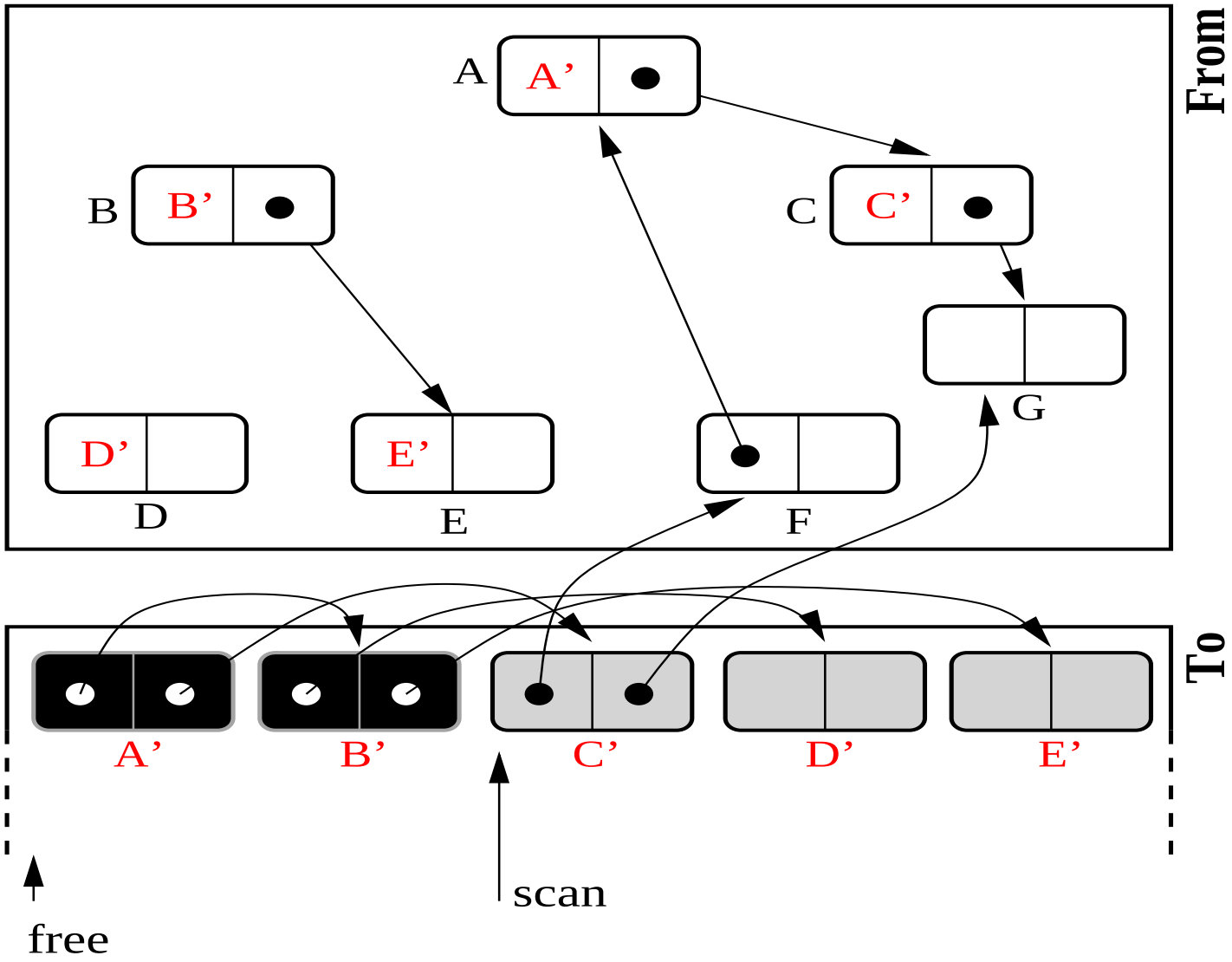

From
A
A'
B
B'
C
C'
G
D'
D
E
F
To
A'
B'
C'
D'
scan
free

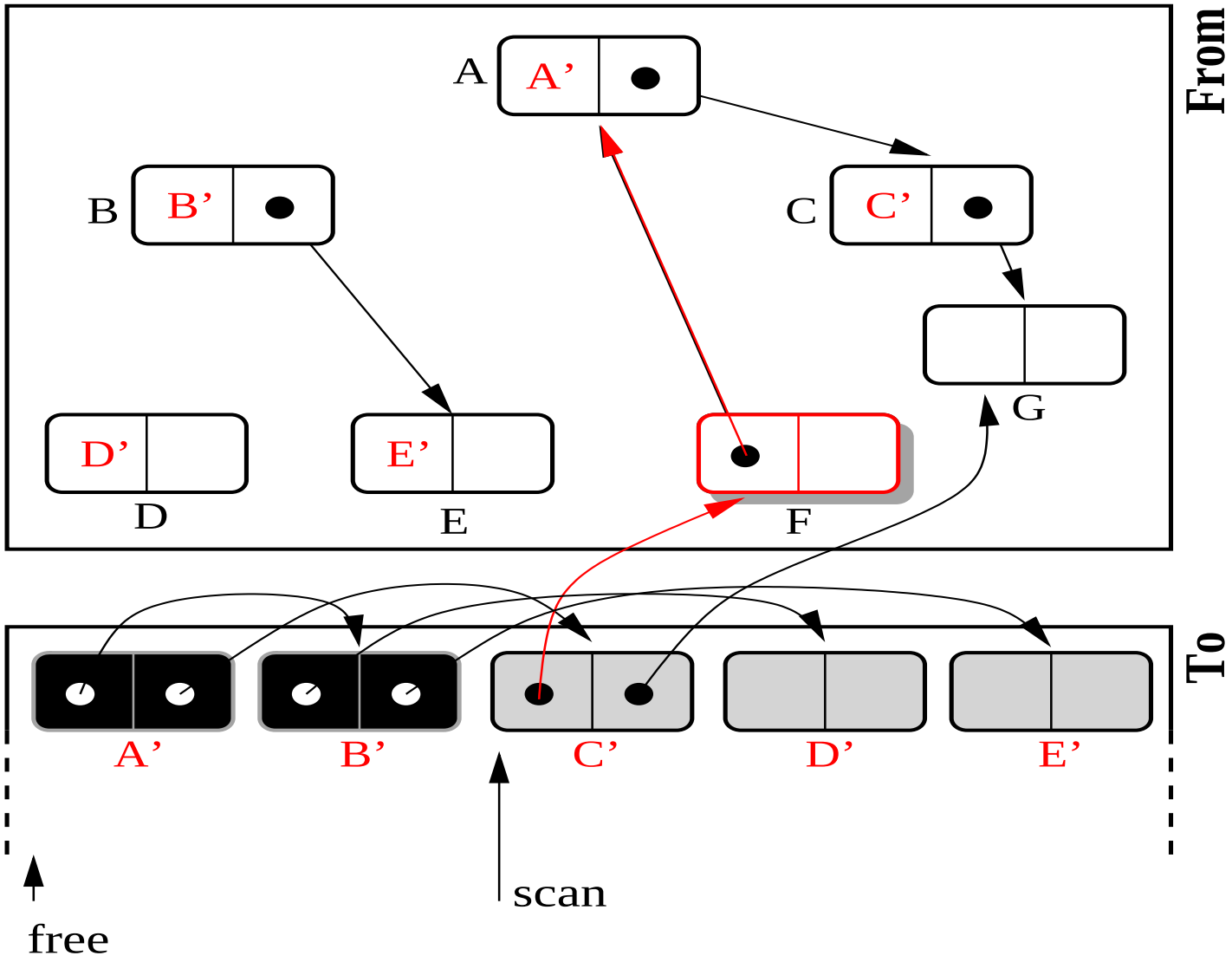

From
A
A'
B
B'
C
C'
G
D'
D
E
F
To
A'
B'
C'
D'
scan
free

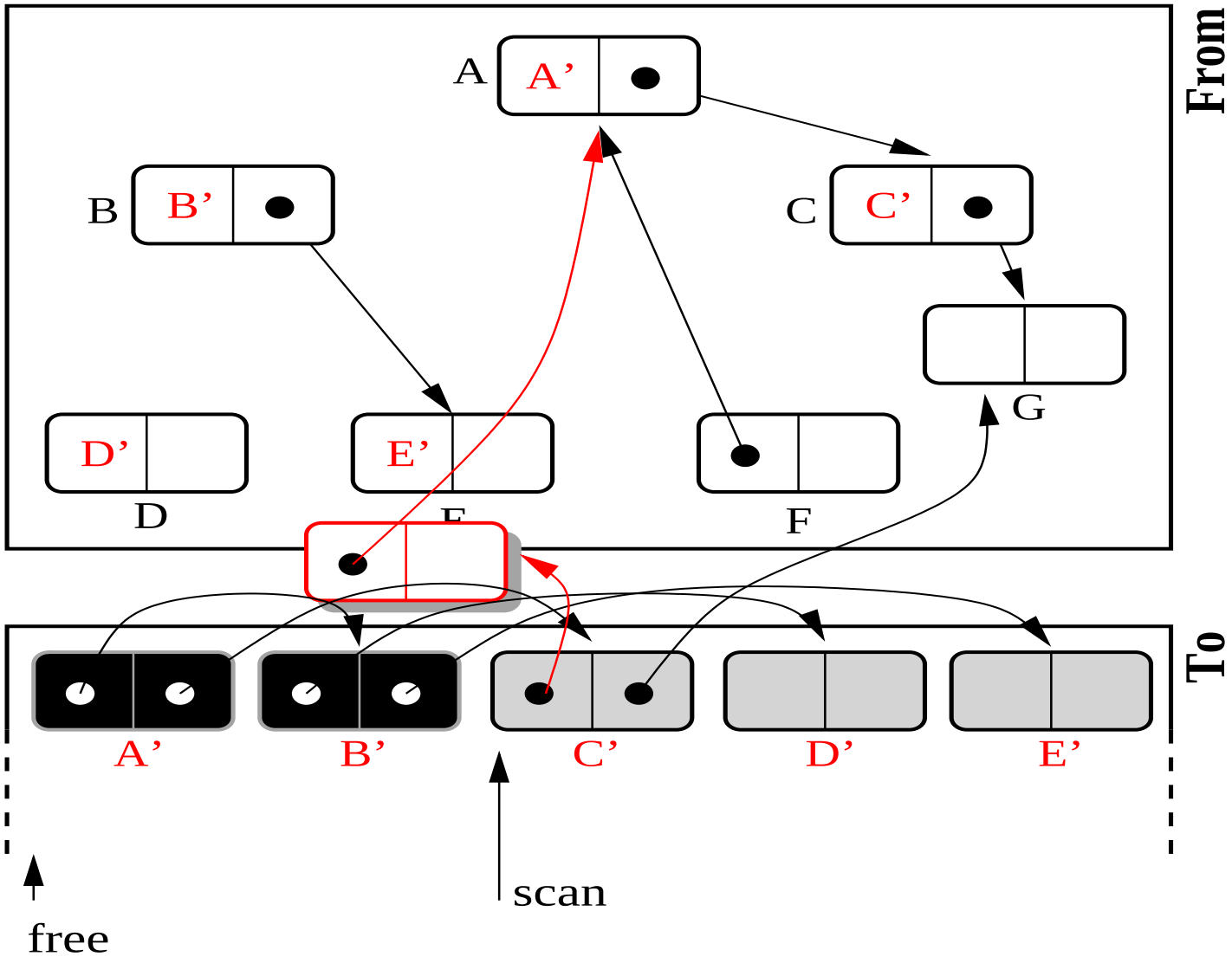

From
A
A'
B
B'
C
C'
G
D'
D
E
F
To
A'
B'
C'
D'
scan
free

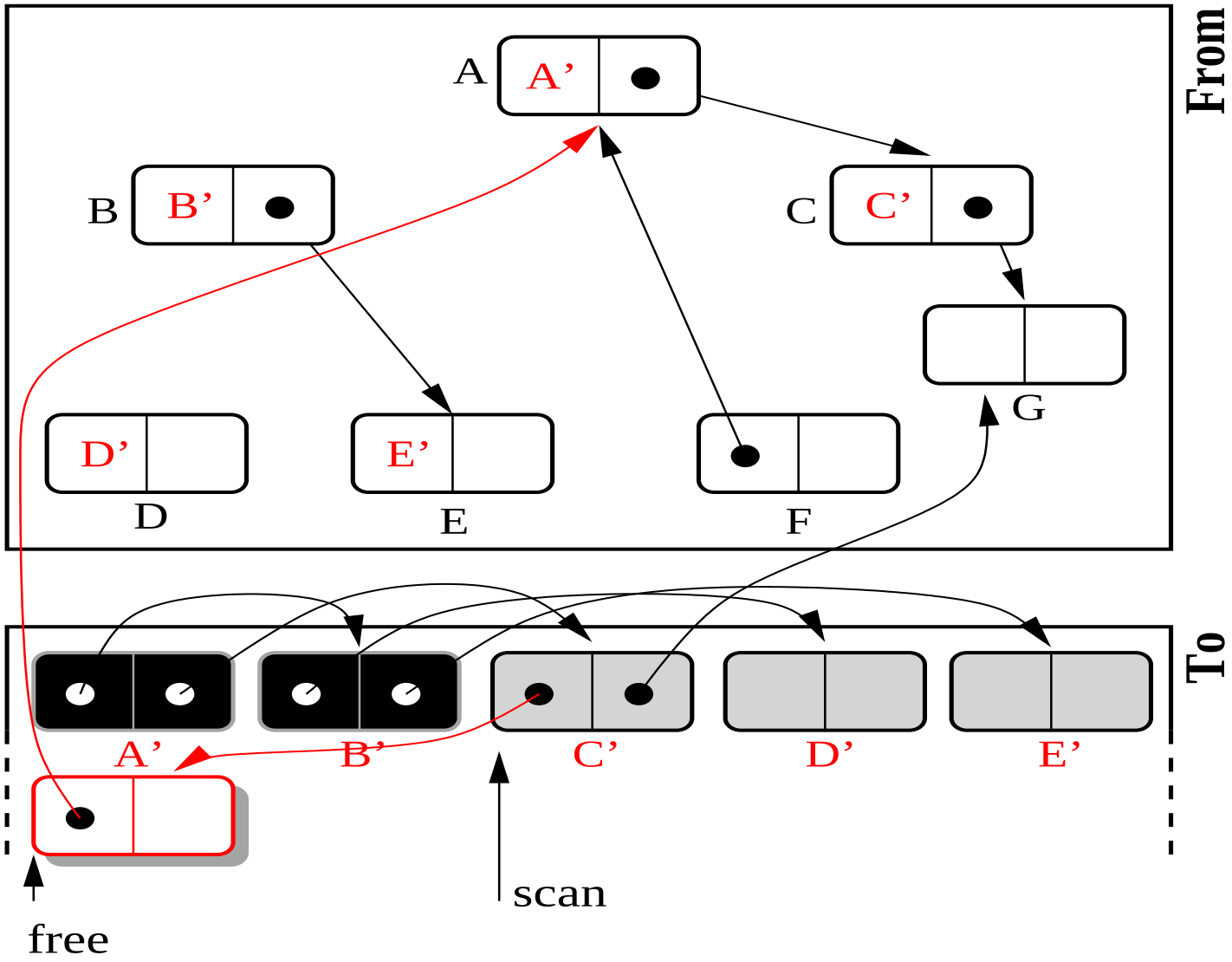

From
A
A'
B
B'
C
C'
G
D'
D
E
F
To
A'
B'
C'
D'
scan
free

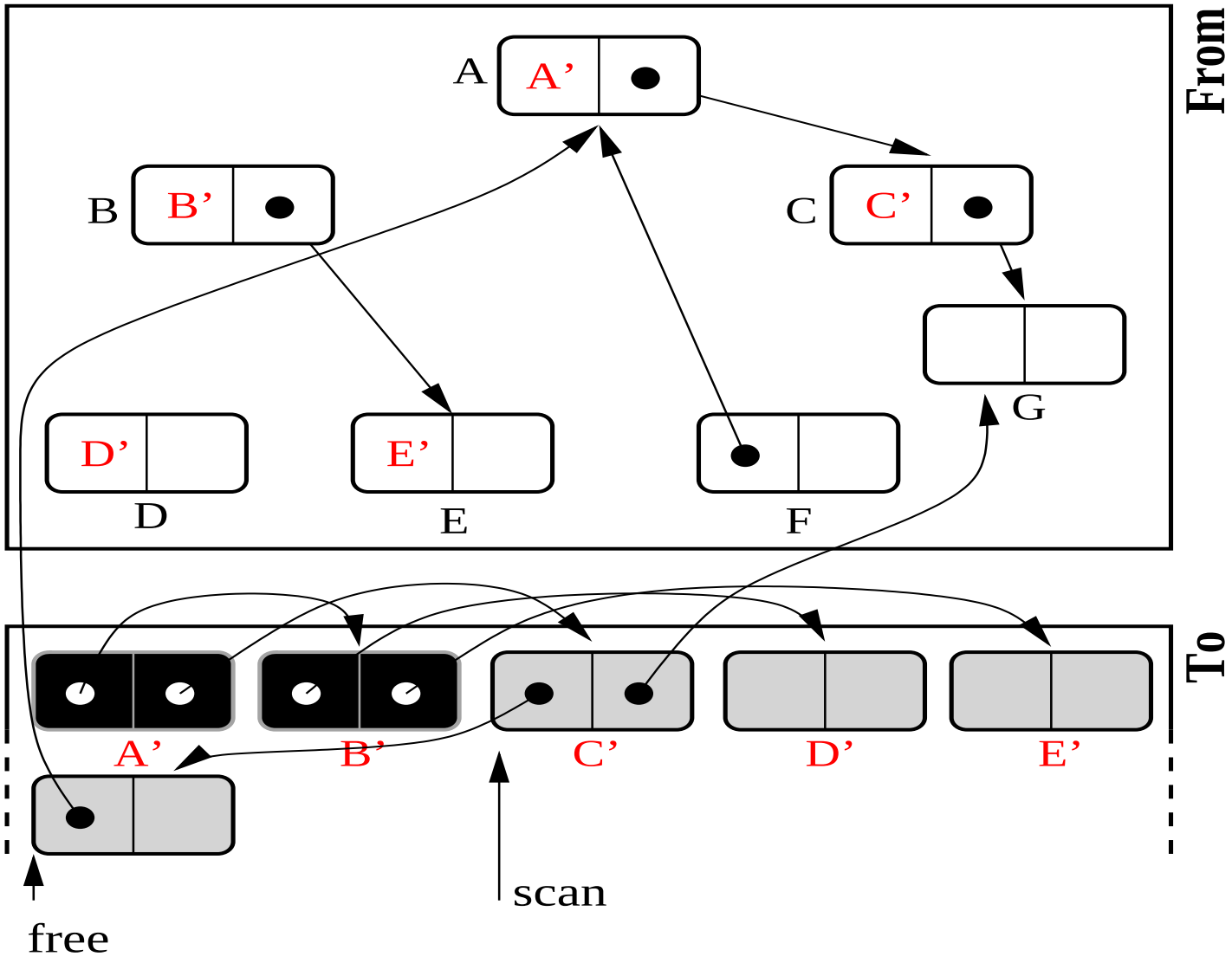

From
A
A'
B
B'
C
C'
G
D'
D
E
F
To
A'
B'
C'
D'
scan
free

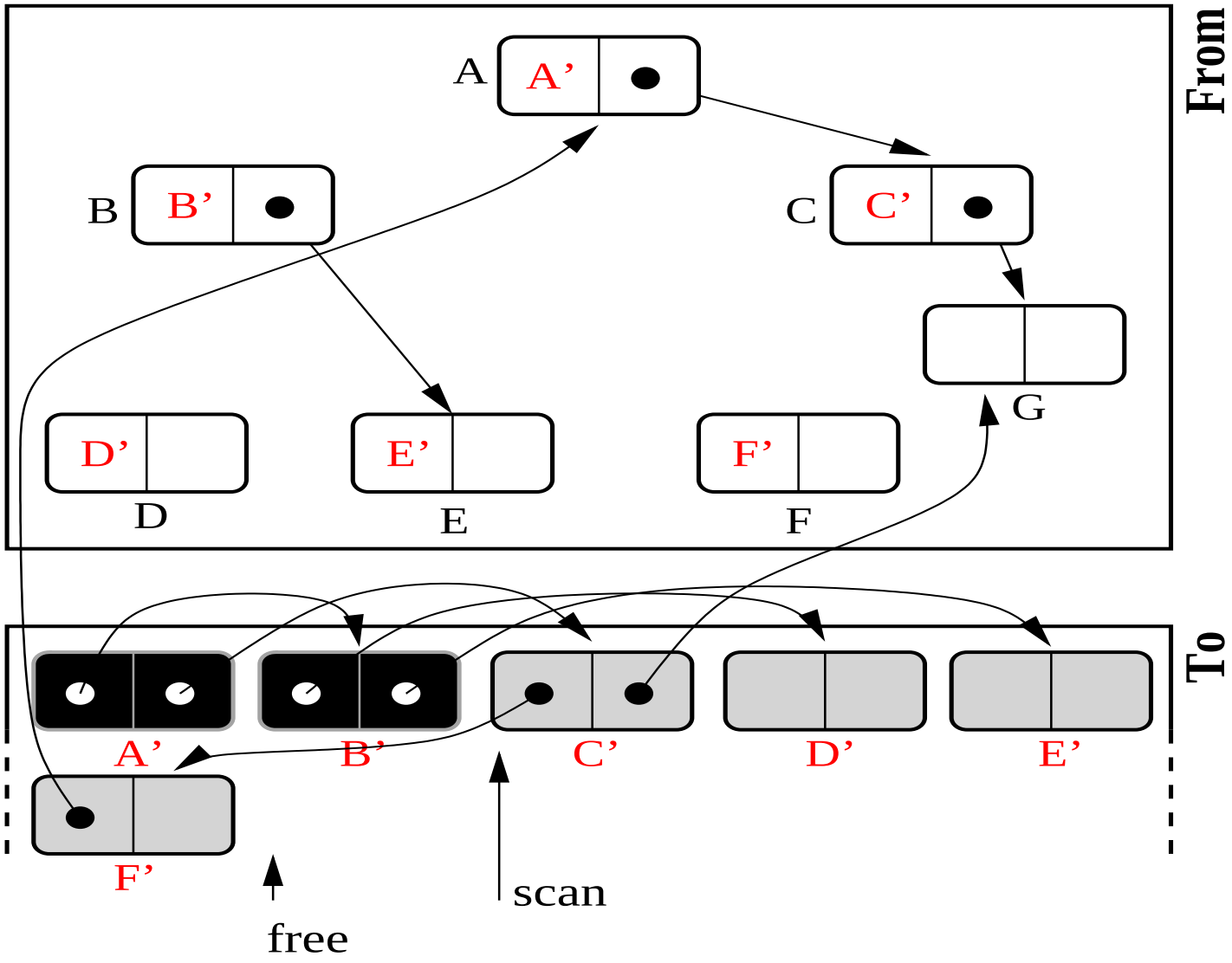

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F
To
A'
B'
C'
D'
E'
scan
free

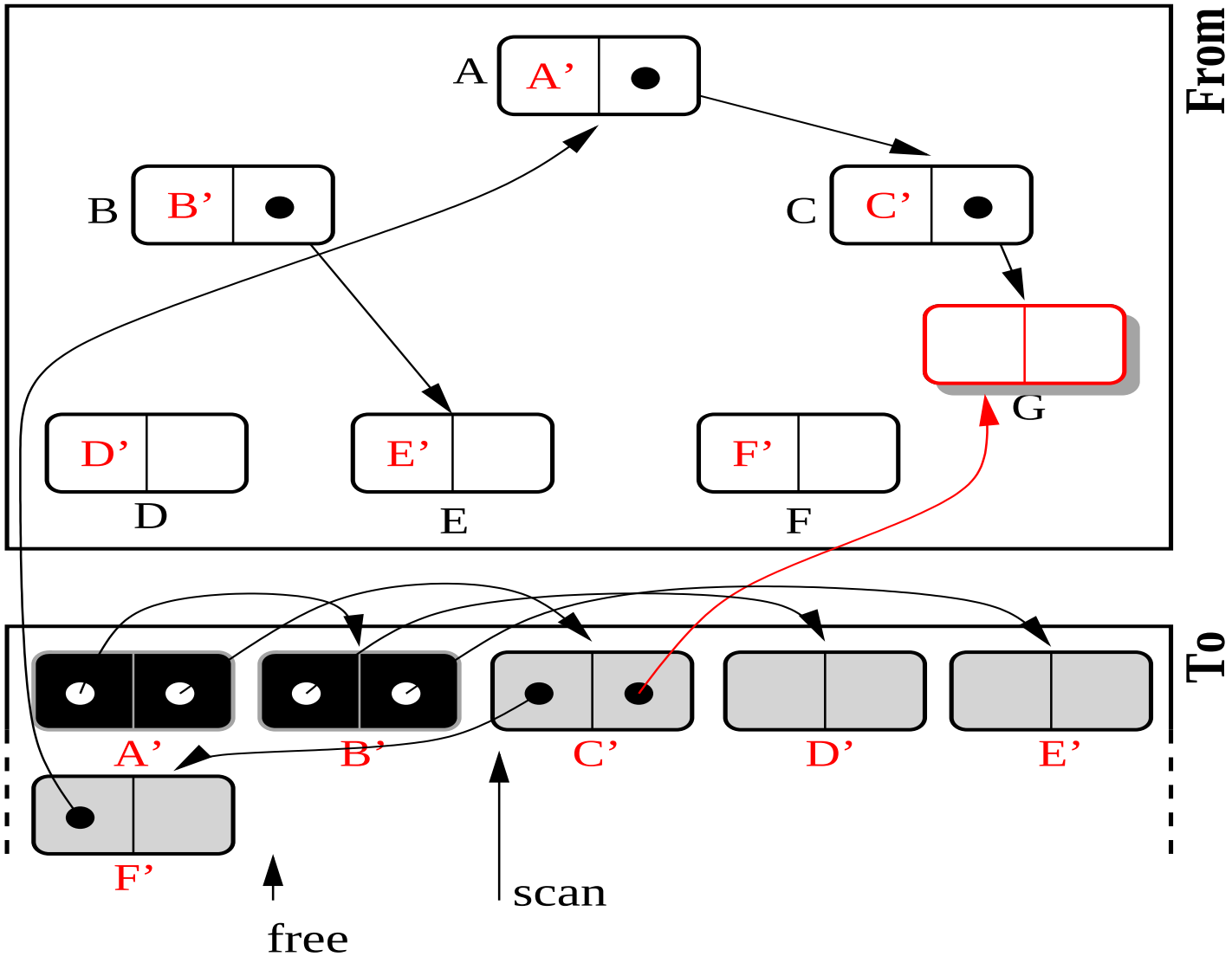

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F
To
A'
B'
C'
D'
E'
scan
free

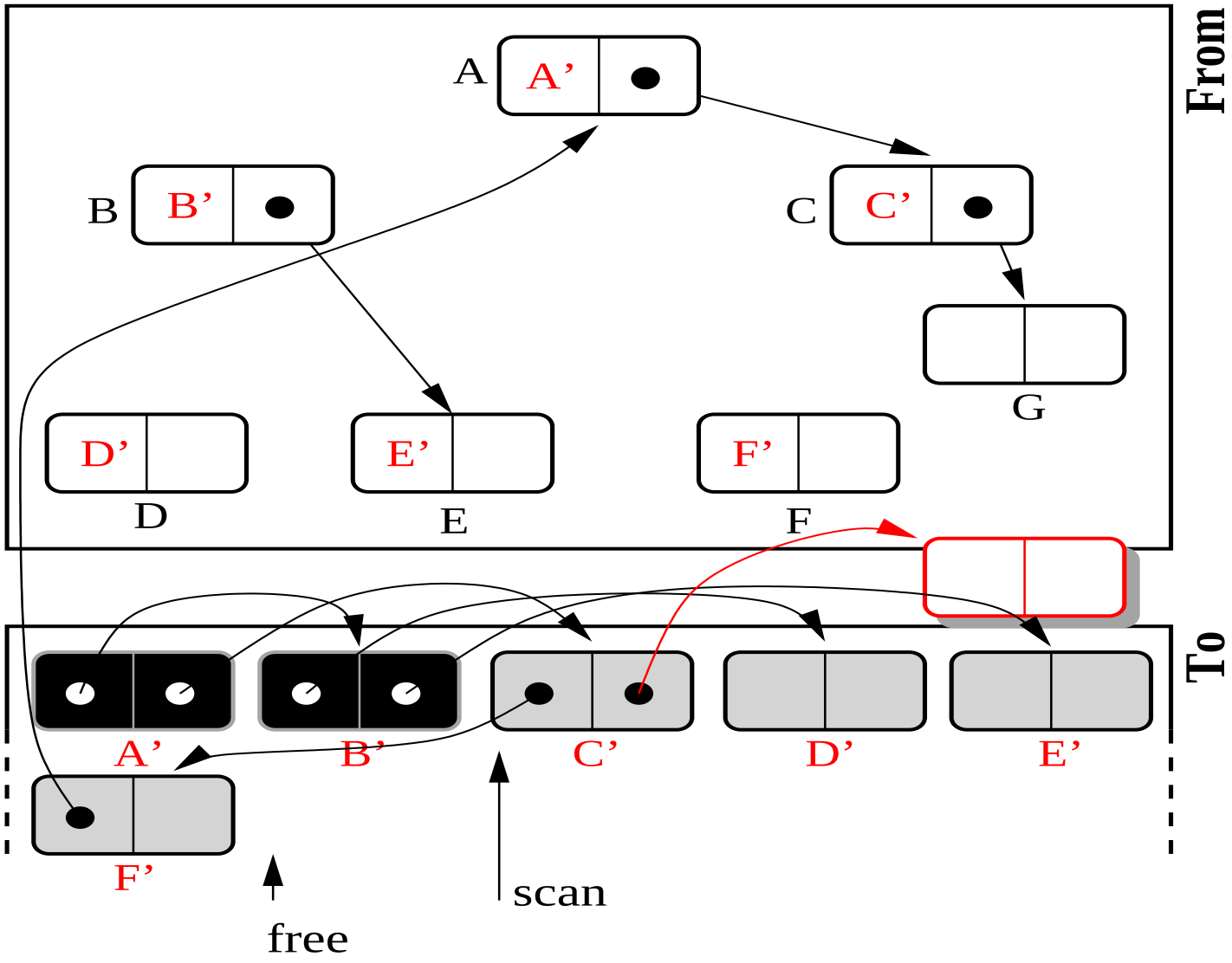

From
A
A'
B
B'
C
C'
G
D'
E'
D
E
F
To
A'
B'
C'
D'
E'
scan
free

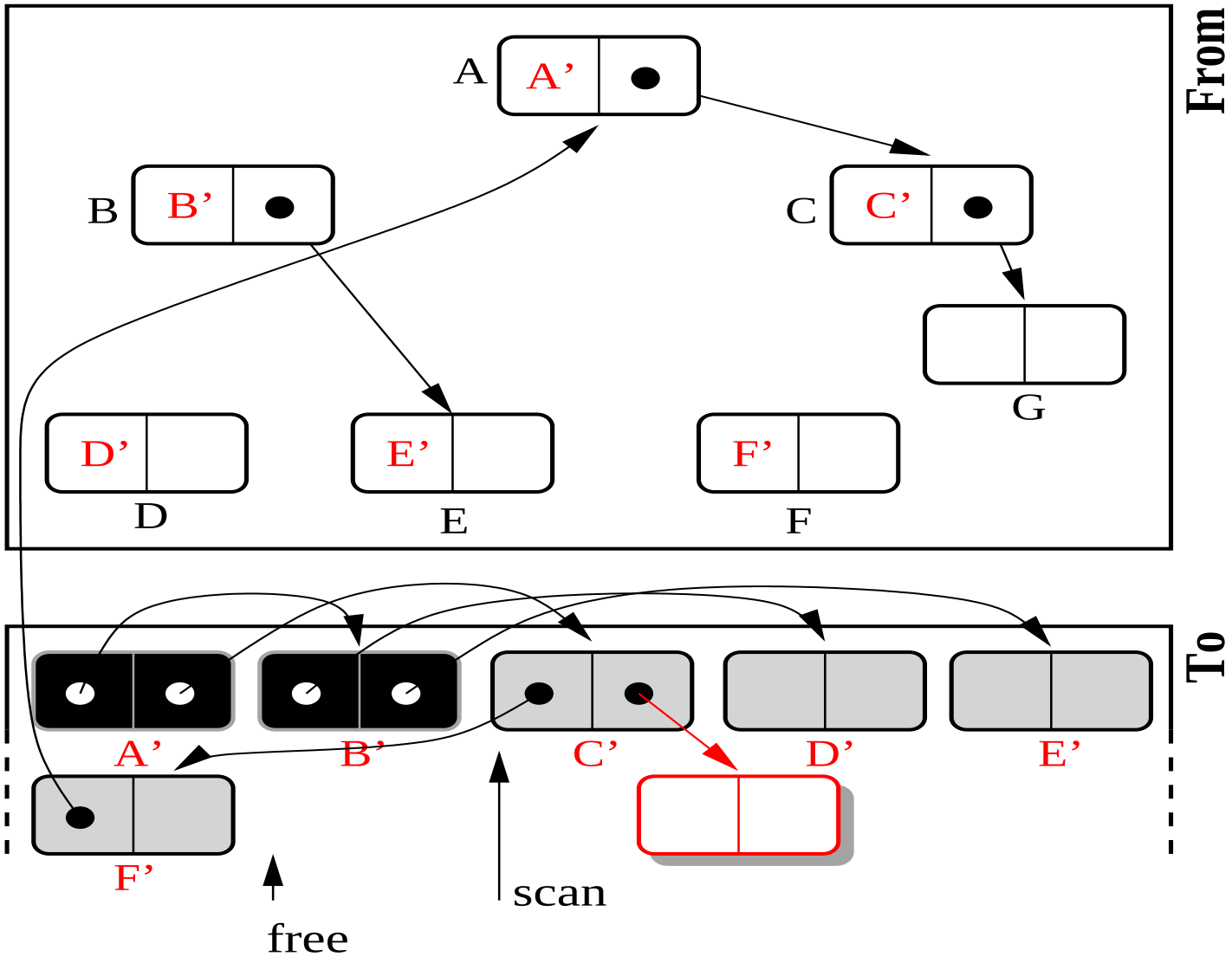

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F
To
A'
B'
C'
D'
E'
free
scan

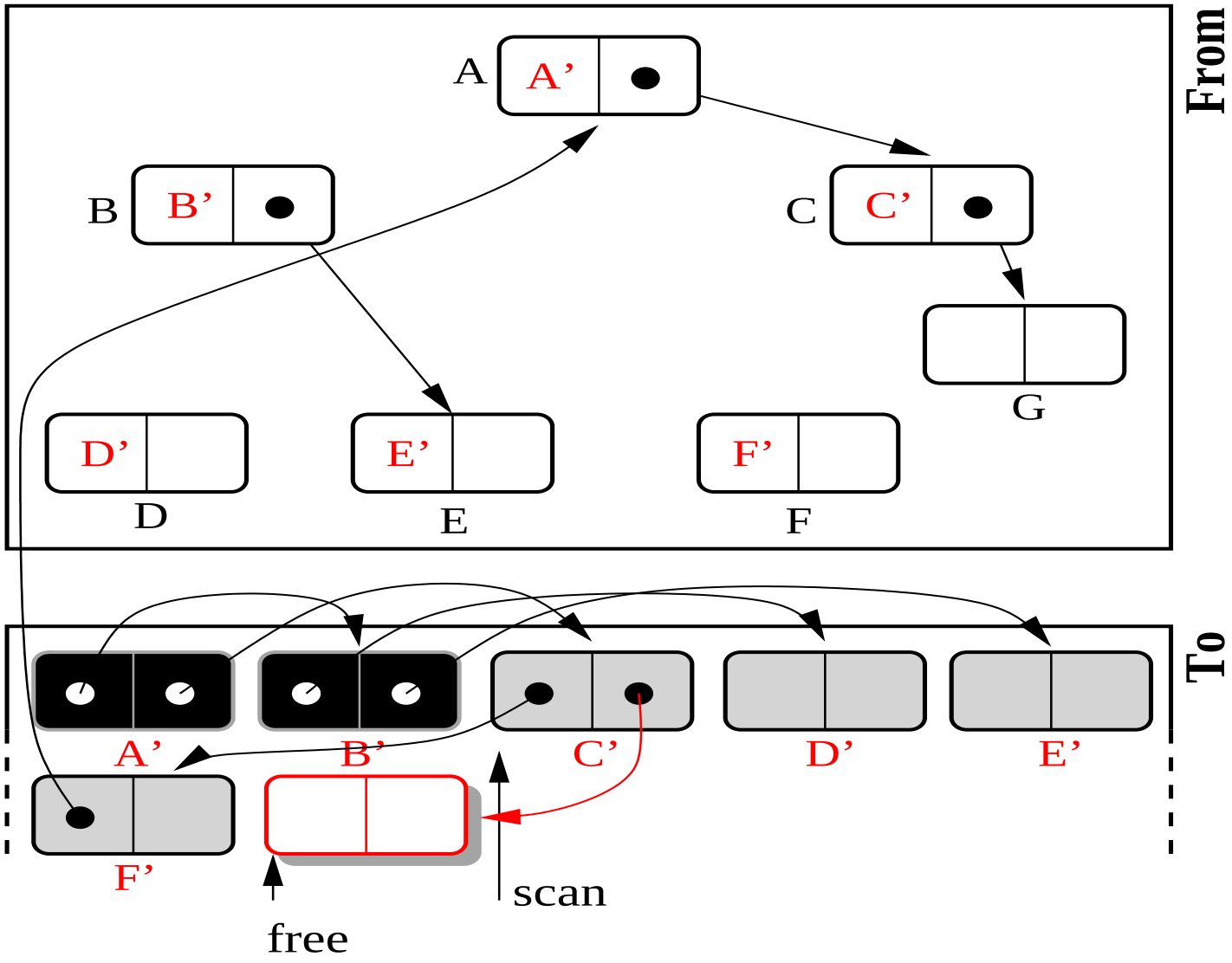

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F
To
A'
B'
C'
D'
E'
free
scan

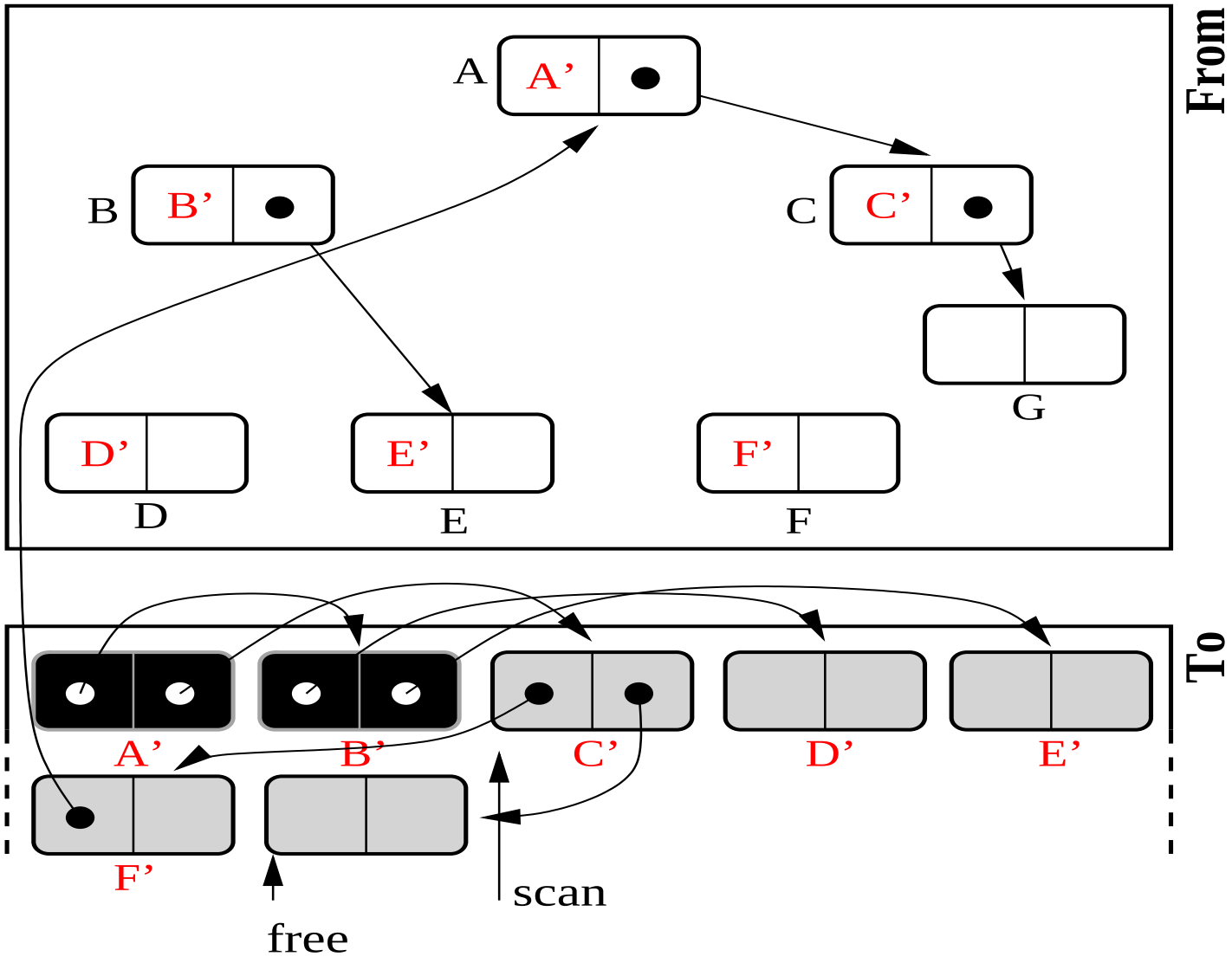

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
free
scan

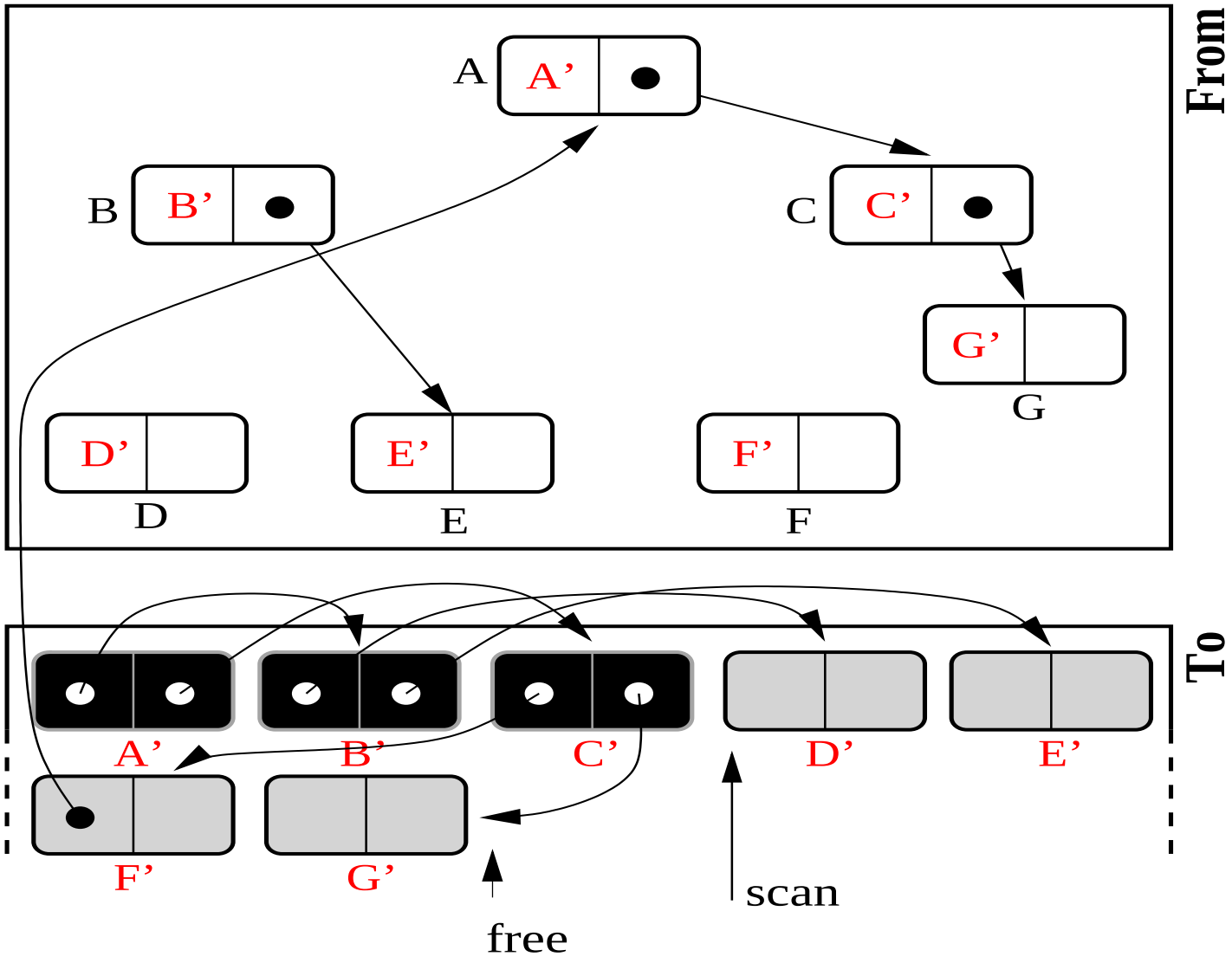

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
free
scan

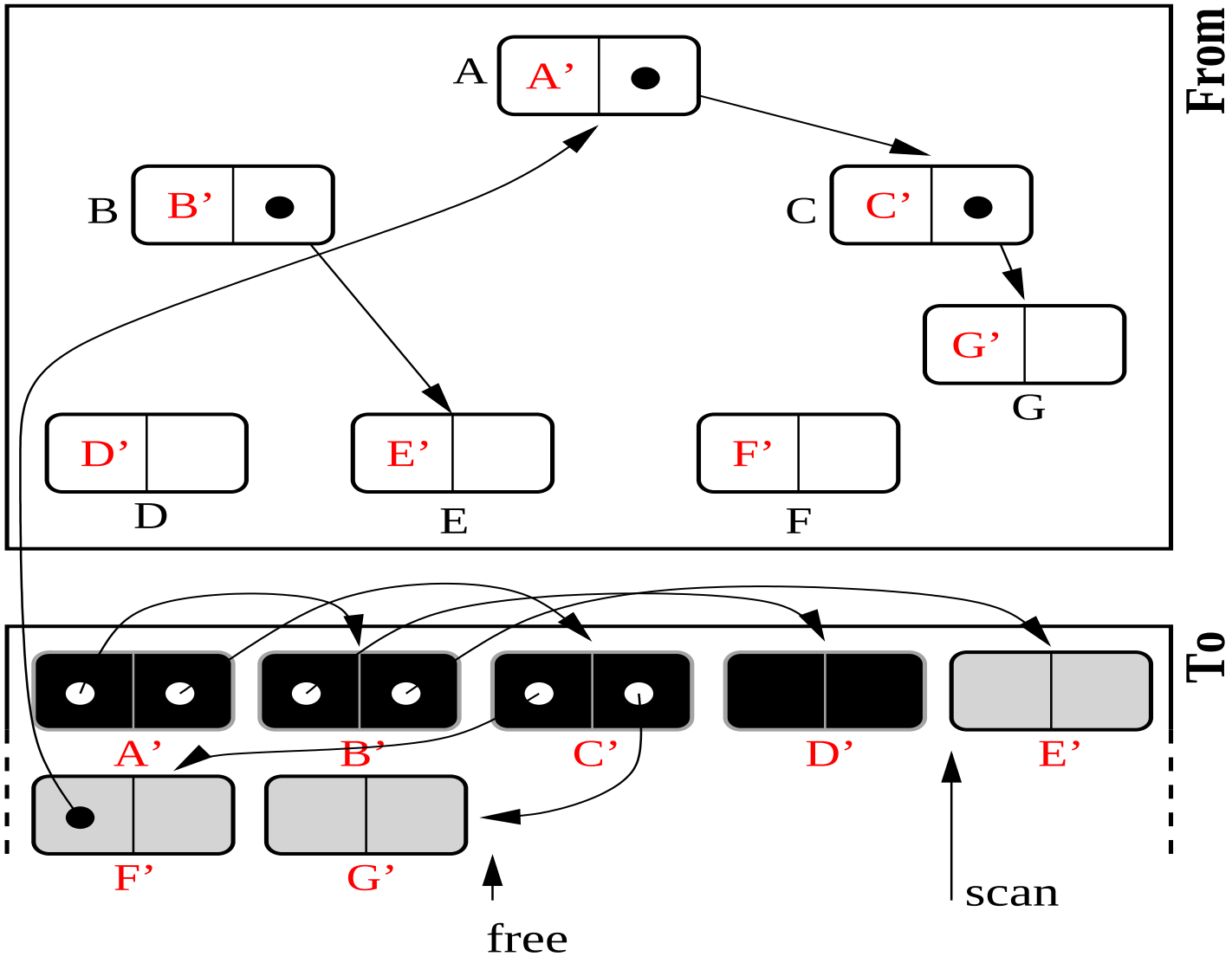

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
free
scan

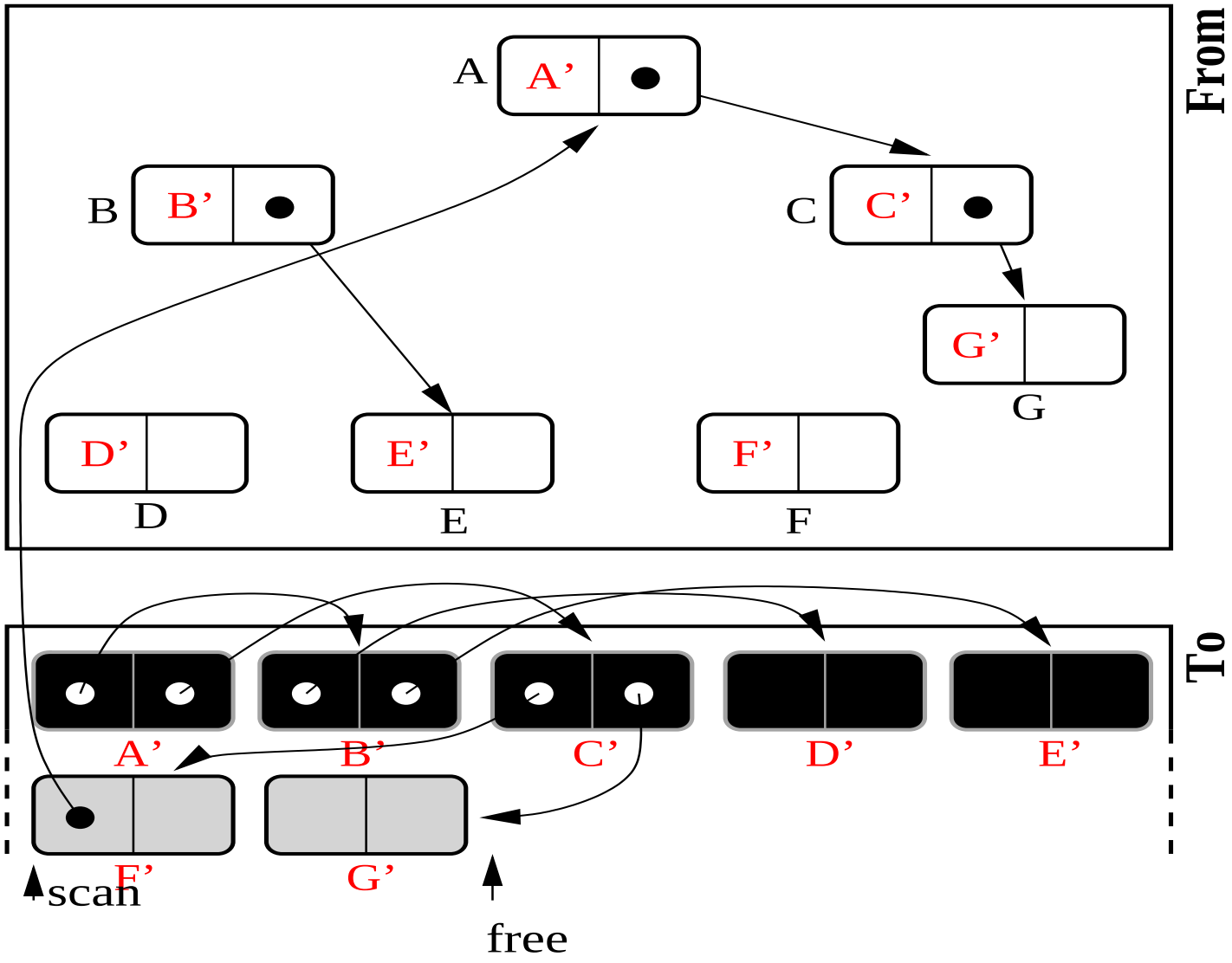

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
free
scan

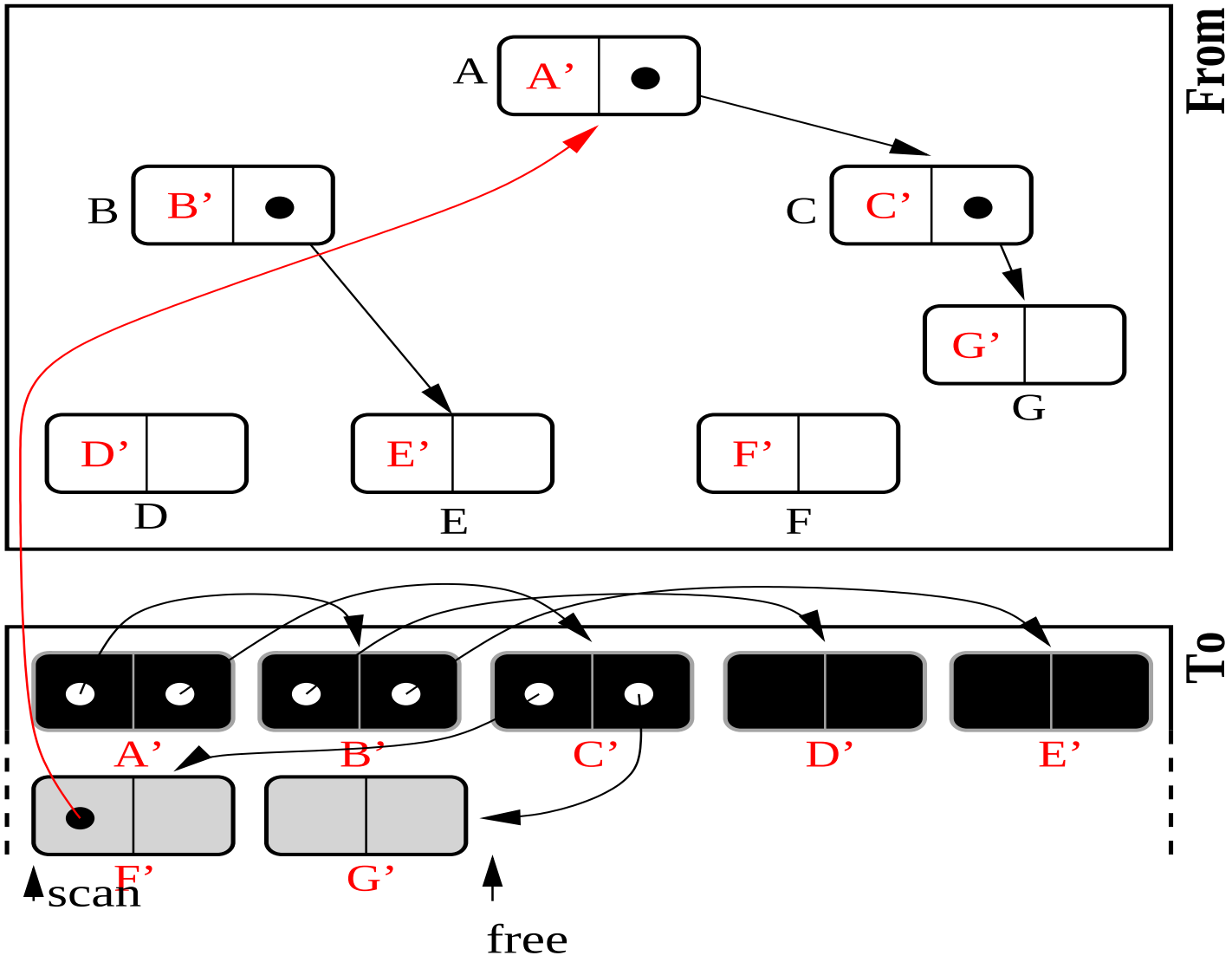

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
free
scan

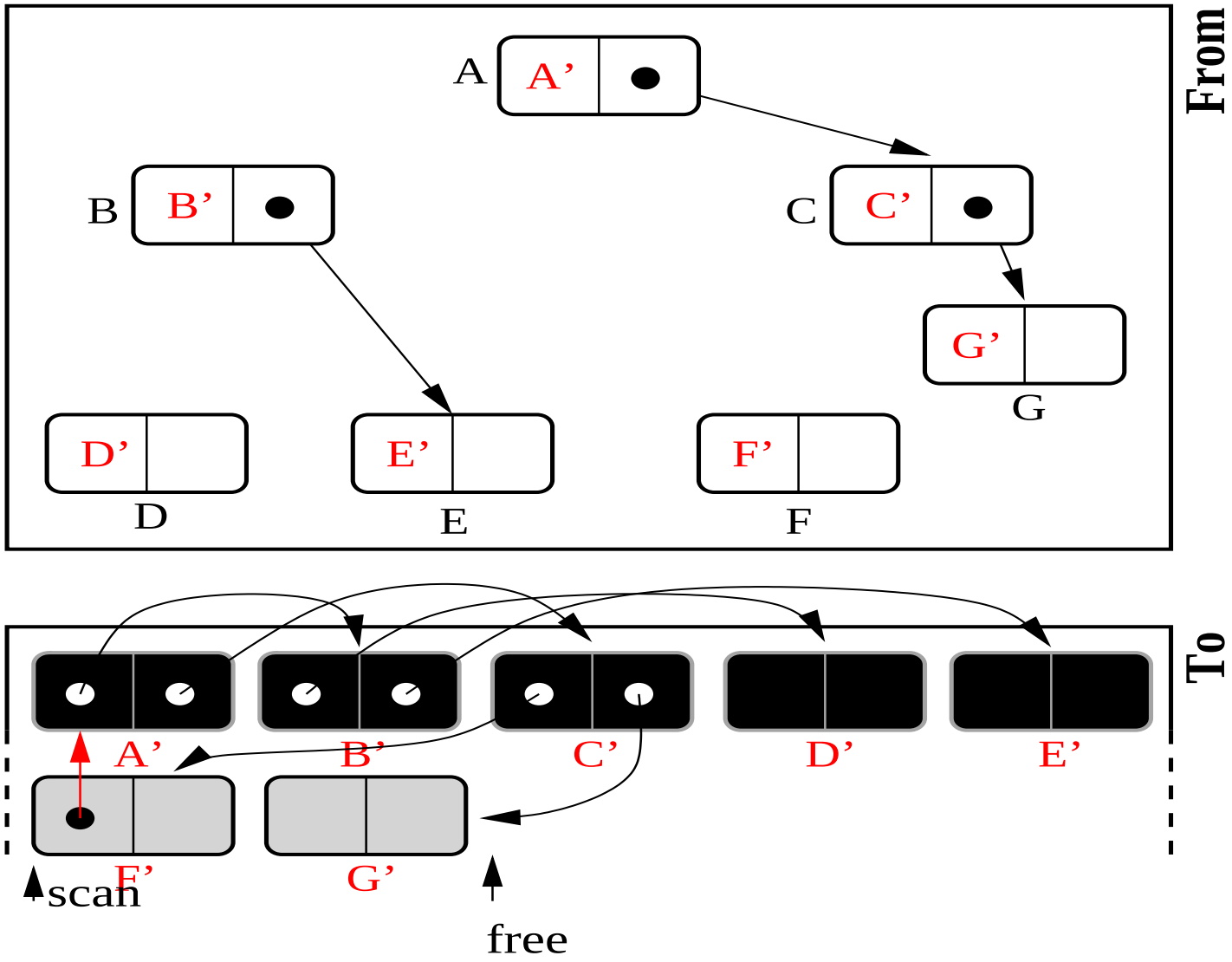

From
A
A'
B
B'
C
C'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
free
scan

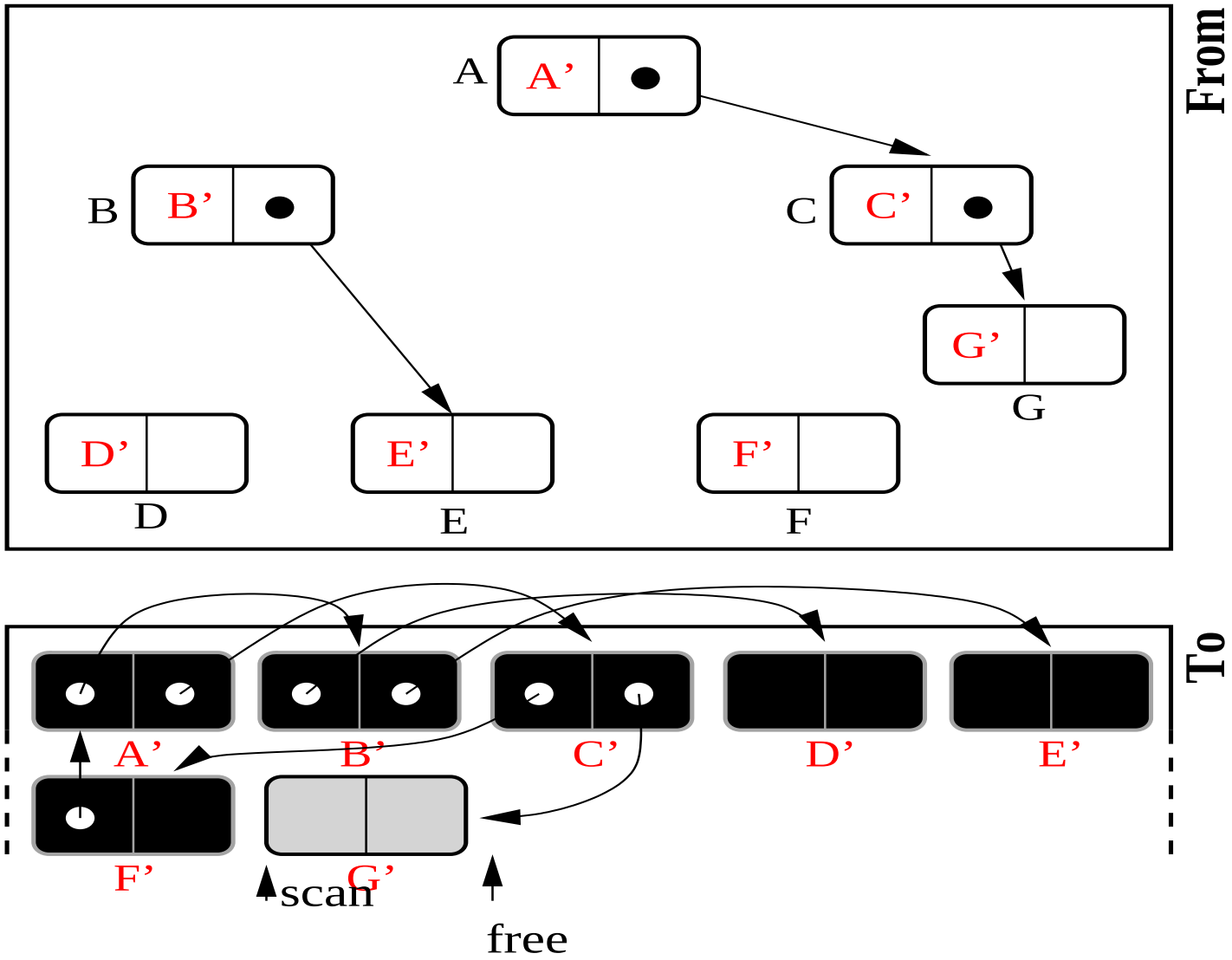

From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
free
scan

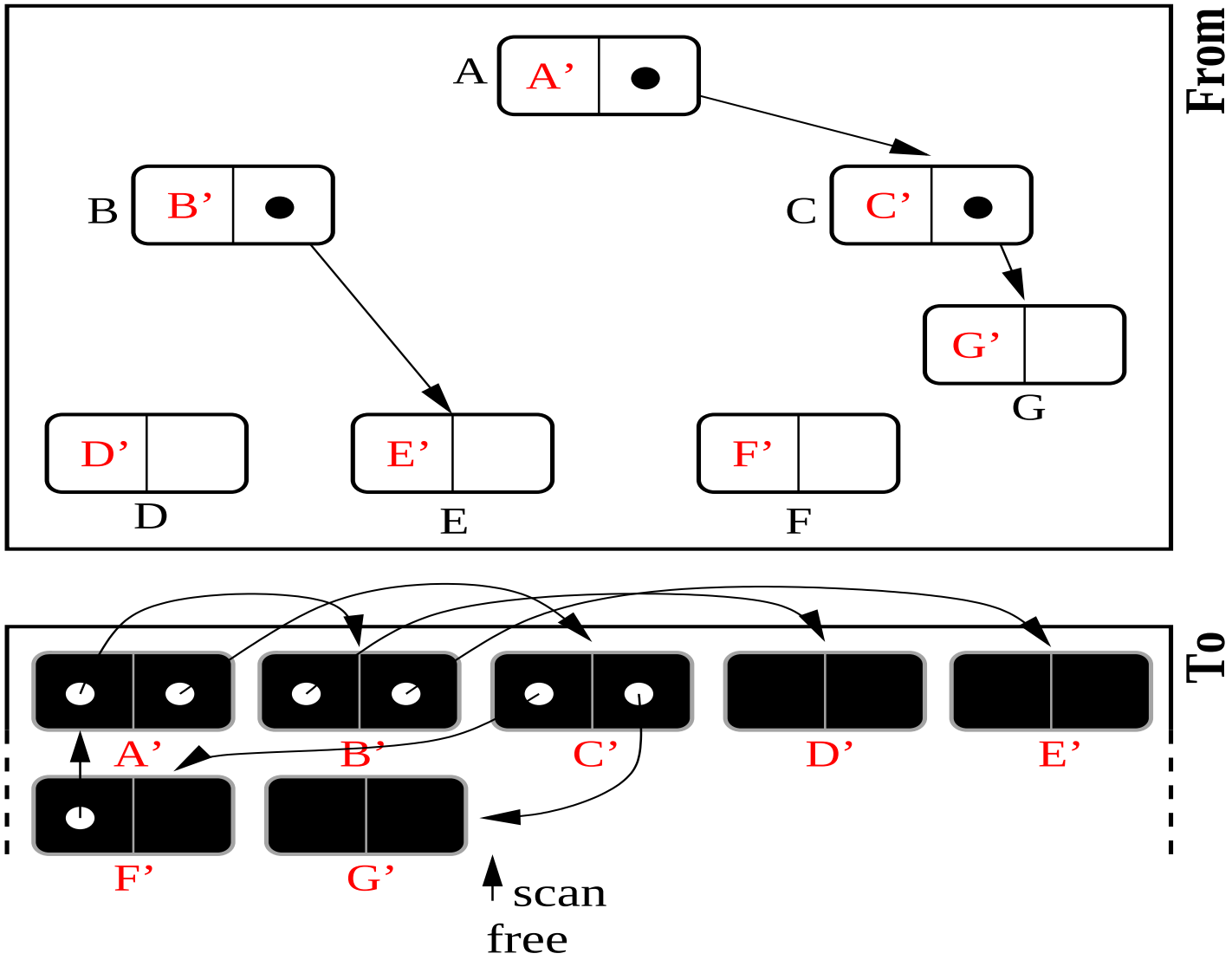

From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
free
scan


From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
scan
free


From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
scan
free


From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
scan
free


From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
scan
free


From
A
A'
B
B'
C
C'
G'
G
D'
D
E'
E
F'
F
To
A'
B'
C'
D'
E'
F'
G'
scan
free

6 Assumptions

6 Assumptions

- Size of nodes known

6 Assumptions

- Size of nodes known
- Child fields known

6 Assumptions

- Size of nodes known
- Child fields known
- Two logically contiguous areas for the heap

6 Assumptions

- Size of nodes known
- Child fields known
- Two logically contiguous areas for the heap
- Ability to mark nodes

7 Properties

7 Properties

- Constant stack required

7 Properties

- Constant stack required
- Time depends only on no. of *live* nodes

7 Properties

- Constant stack required
- Time depends only on no. of *live* nodes
- Covers cycles / sharing

7 Properties

- Constant stack required
- Time depends only on no. of *live* nodes
- Covers cycles / sharing
- Heap is compacted

7 Properties

- Constant stack required
- Time depends only on no. of *live* nodes
- Covers cycles / sharing
- Heap is compacted
- Stop/start collector

7 Properties

- Constant stack required
- Time depends only on no. of *live* nodes
- Covers cycles / sharing
- Heap is compacted
- Stop/start collector
- Nothing to do between GCs

8 Disadvantages

High Level

Low Level

8 Disadvantages

High Level

- Copying large objects is expensive

Low Level

8 Disadvantages

High Level

- Copying large objects is expensive
- Twice the logical memory needed

Low Level

8 Disadvantages

High Level

- Copying large objects is expensive
- Twice the logical memory needed

Low Level

- Breadth first $\Rightarrow$ decreased locality

8 Disadvantages

High Level

- Copying large objects is expensive
- Twice the logical memory needed

Low Level

- Breadth first $\Rightarrow$ decreased locality
- Paging issues

9 Variations(1)

9 Variations(1)

- Large Object Areas

9 Variations(1)

- Large Object Areas
  (possibly handled by different collector)

9 Variations(1)

- Large Object Areas
  (possibly handled by different collector)
- Areas for long living objects

9 Variations(1)

- Large Object Areas
  (possibly handled by different collector)
- Areas for long living objects
  (only scanned, not copied)

9 Variations(1)

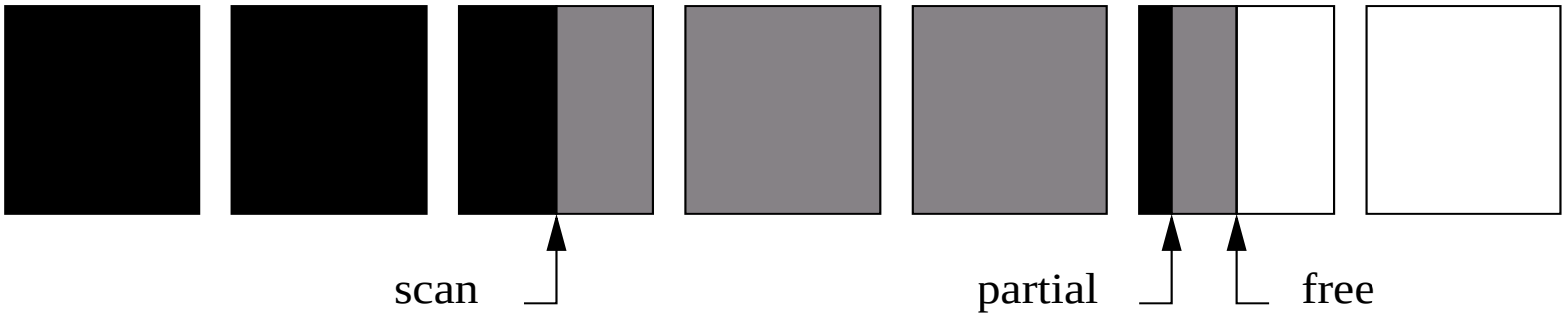
- Large Object Areas
  (possibly handled by different collector)
- Areas for long living objects
  (only scanned, not copied)
- Maintain locality by using other exploration strategies

9 Variations(1)

- Large Object Areas
  (possibly handled by different collector)
- Areas for long living objects
  (only scanned, not copied)
- Maintain locality by using other exploration strategies
  (but then stack becomes an issue)

10 Variations(2)

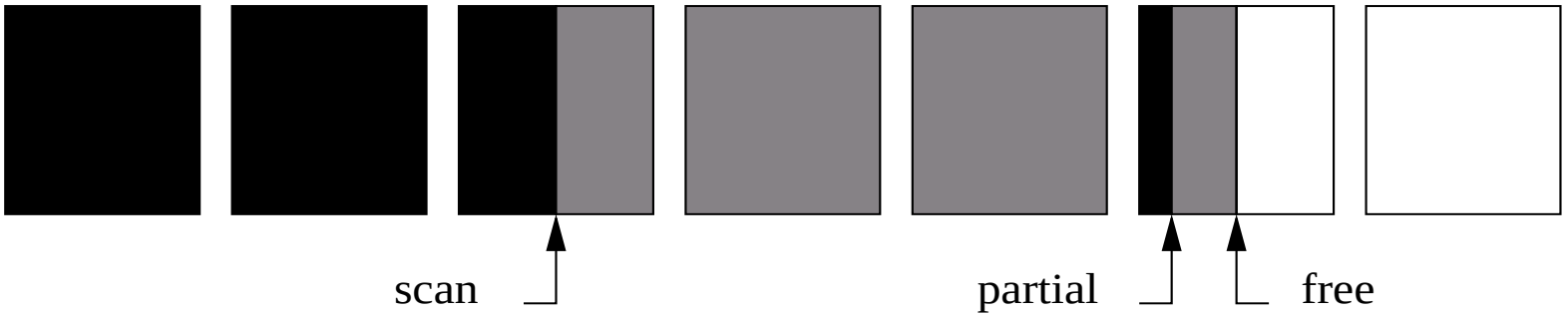
Approximately depth-first copying(1)



10 Variations(2)

Approximately depth-first copying(1)

Modification of Cheney's agorithm by Moon (1984)

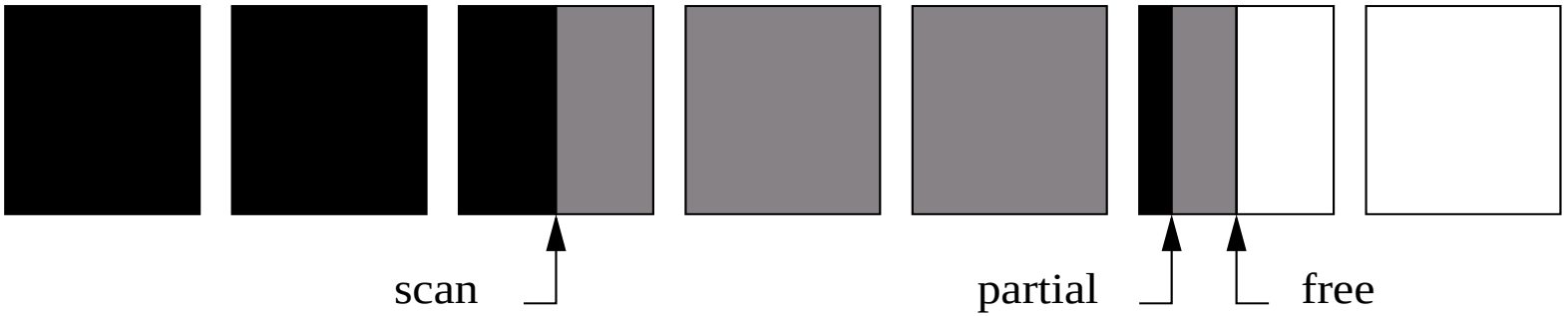


10 Variations(2)

Approximately depth-first copying(1)

Modification of Cheney's agorithm by Moon (1984)

- Always start scanning on the last partially filled page in To-space

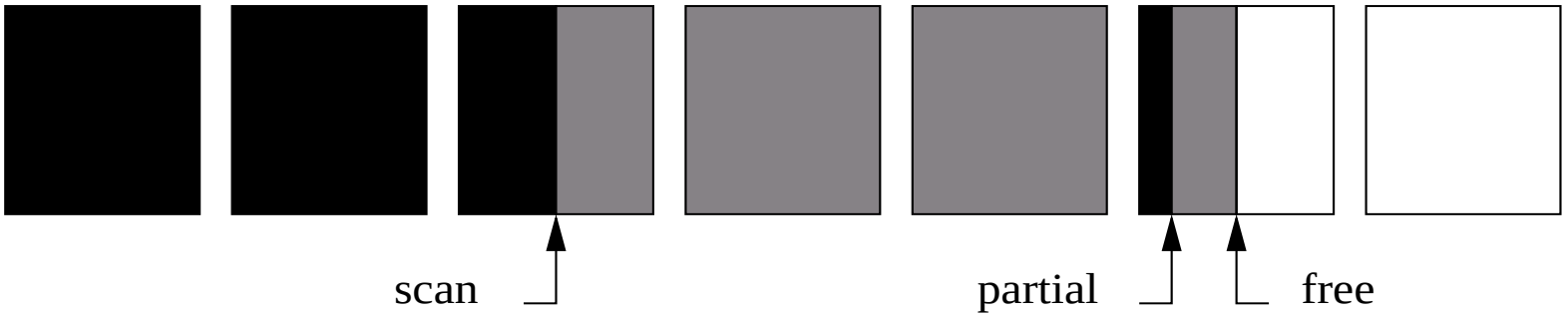


10 Variations(2)

Approximately depth-first copying(1)

Modification of Cheney's agorithm by Moon (1984)

- Always start scanning on the last partially filled page in To-space
- When that page is completed, continue with ordinary scan

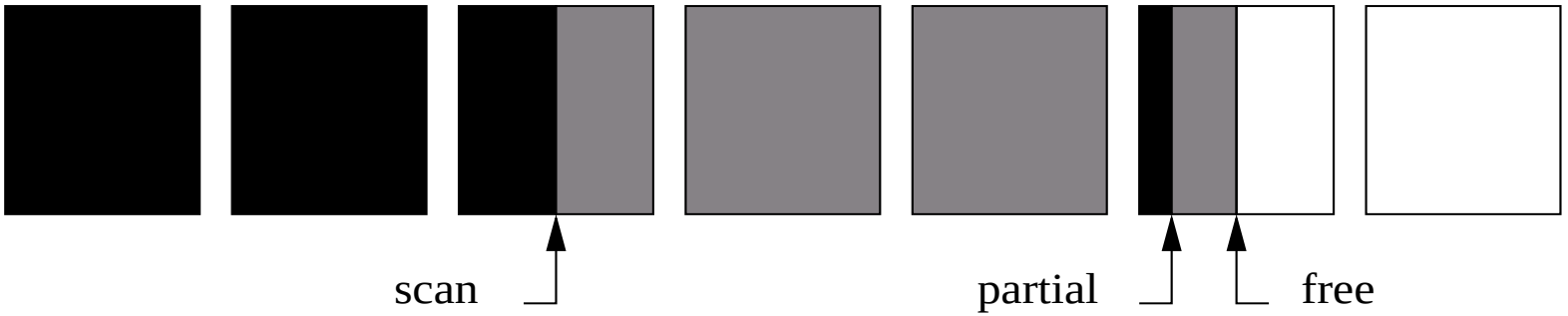


10 Variations(2)

Approximately depth-first copying(1)

Modification of Cheney's agorithm by Moon (1984)

- Always start scanning on the last partially filled page in To-space
- When that page is completed, continue with ordinary scan
- As soon as an object is copied, start partial scan again



11 Variations(3)

Approximately depth-first copying(2)

11 Variations(3)

Approximately depth-first copying(2)

- Drawback: Some nodes are scanned twice

11 Variations(3)

Approximately depth-first copying(2)

- Drawback: Some nodes are scanned twice
- Tests indicate a 15 percent improvement of locality

11 Variations(3)

Approximately depth-first copying(2)

- Drawback: Some nodes are scanned twice
- Tests indicate a 15 percent improvement of locality
- . . . and a 6 percent increased GC time

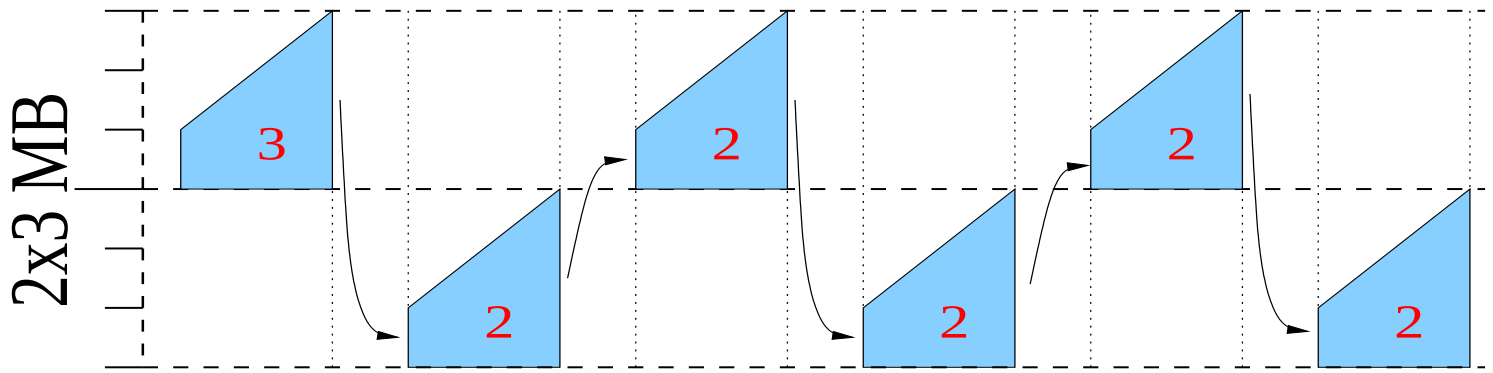
12 Increasing efficiency

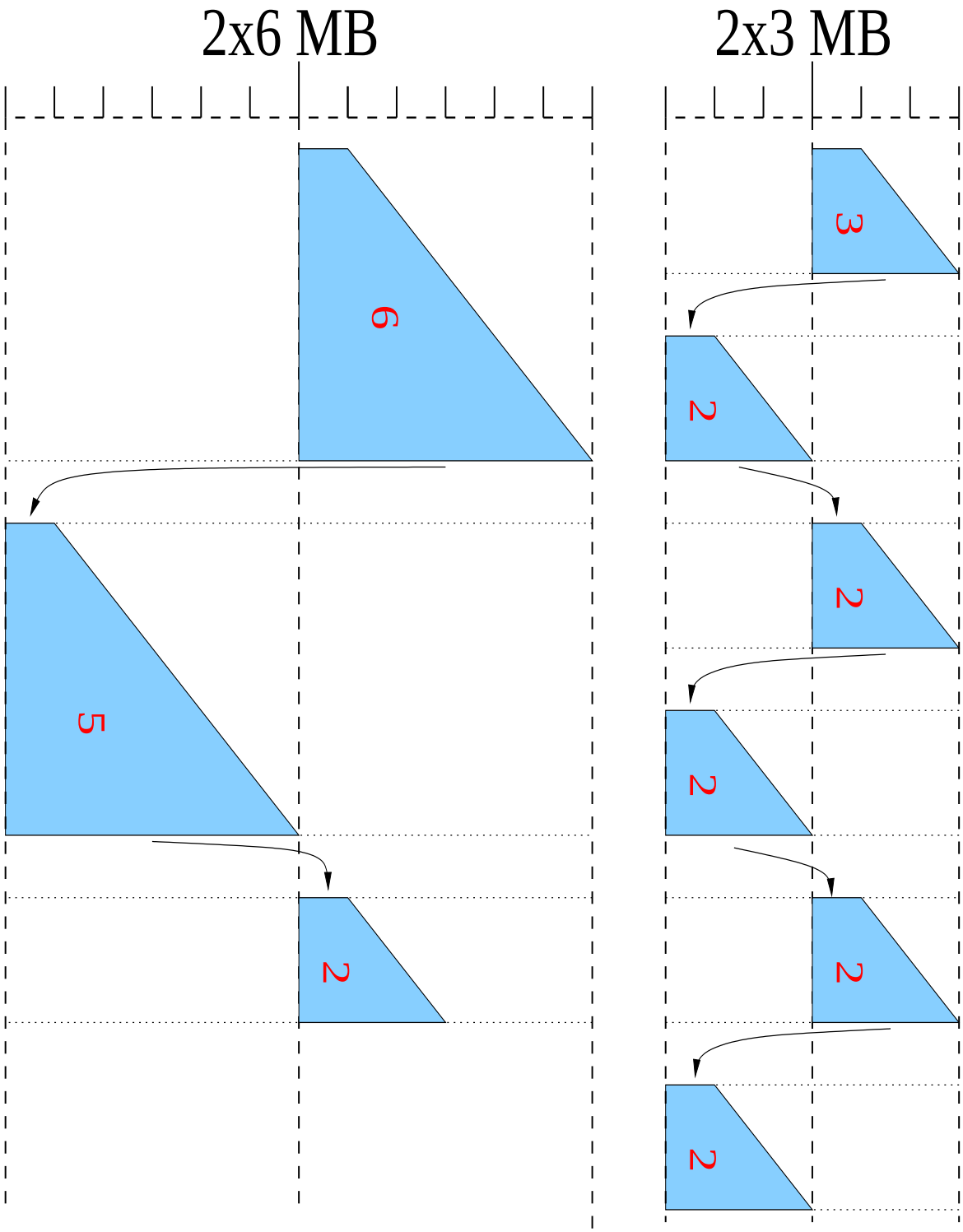
12 Increasing efficiency

- Making the semi-spaces bigger decreases frequency of GC

12 Increasing efficiency

- Making the semi-spaces bigger decreases frequency of GC
- Less frequent GC means older objects
  $\Rightarrow$ More garbage


2x3 MB
3
2
2
2
2
2


2x6 MB
6
5
2
2x3 MB
3
2
2
2
2
2

13 When to use copying

13 When to use copying

- Storage management dominated by allocation (alloc. is cheap)

13 When to use copying

- Storage management dominated by allocation
  (alloc. is cheap)
- Many small, short-lived objects
  (copying small objects not much more expensive than marking)

13 When to use copying

- Storage management dominated by allocation
  (alloc. is cheap)
- Many small, short-lived objects
  (copying small objects not much more expensive than marking)
- GC delay doesn't matter (no real-time system)

14 Outlook

14 Outlook

- Hybrid systems can be used

14 Outlook

- Hybrid systems can be used
  (e.g. use copying only for small objects, mark-sweep for large objects)

14 Outlook

- Hybrid systems can be used
  (e.g. use copying only for small objects, mark-sweep for large objects)
- Copying collection can be used as a foundation for

  GC algorithms.

14 Outlook

- Hybrid systems can be used
  (e.g. use copying only for small objects, mark-sweep for large objects)
- Copying collection can be used as a foundation for
  - incremental

  GC algorithms.

14 Outlook

- Hybrid systems can be used
  (e.g. use copying only for small objects, mark-sweep for large objects)
- Copying collection can be used as a foundation for
  - incremental
  - generational

  GC algorithms.