

CS186 Discussion #13

(Advanced Concurrency)

Lock Granularity

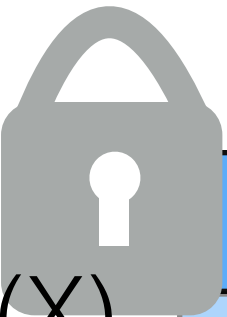
sid	points	grade
Bob	43	C
Joe	99	C
Alice	87	C
Suzy	50	C
Ted	73	C
Tim	12	C

T1: UPDATE students SET grade='A' WHERE points >= 70;

T2: UPDATE students SET grade='F' WHERE points < 70;

Lock Granularity

T1(X)




sid	points	grade
Bob	43	C
Joe	99	C
Alice	87	C
Suzy	50	C
Ted	73	C
Tim	12	C

T1: UPDATE students SET grade='A' WHERE points >= 70;

T2: UPDATE students SET grade='F' WHERE points < 70;

Lock Granularity

T1(X)




sid	points	grade
Bob	43	C
Joe	99	A
Alice	87	A
Suzy	50	C
Ted	73	A
Tim	12	C

T1: UPDATE students SET grade='A' WHERE points >= 70;

T2: UPDATE students SET grade='F' WHERE points < 70;

Lock Granularity

T2(X)



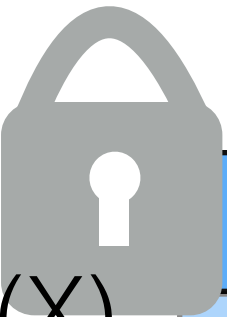
sid	points	grade
Bob	43	C
Joe	99	A
Alice	87	A
Suzy	50	C
Ted	73	A
Tim	12	C

T1: UPDATE students SET grade='A' WHERE points >= 70;

T2: UPDATE students SET grade='F' WHERE points < 70;

Lock Granularity

T2(X)



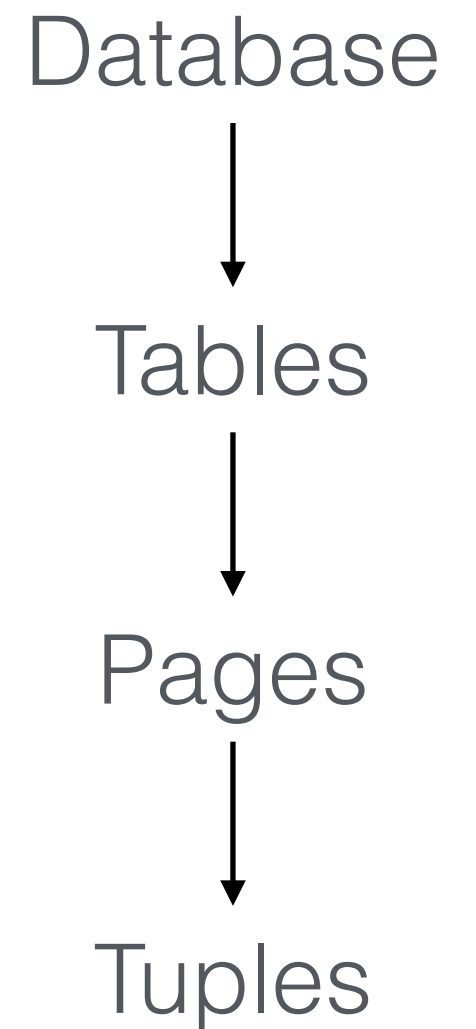
sid	points	grade
Bob	43	F
Joe	99	A
Alice	87	A
Suzy	50	F
Ted	73	A
Tim	12	F

T1: UPDATE students SET grade='A' WHERE points >= 70;

T2: UPDATE students SET grade='F' WHERE points < 70;

Lock Hierarchy

- Each Xact starts at root of hierarchy
- Gets locks in top-down order
- Releases locks in bottom-up order



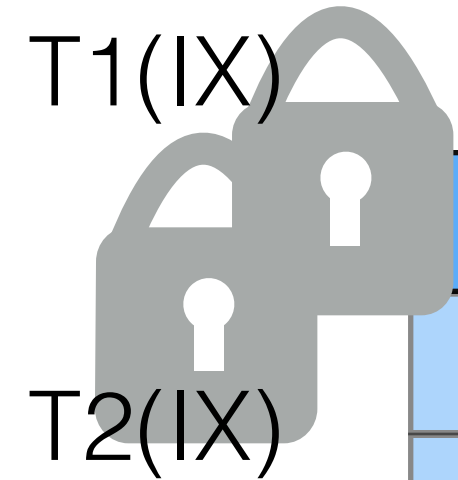
Locks

- IS: intent to get S lock(s) at finer granularity
- IX: intent to get X lock(s) at finer granularity
- SIX: shared lock, with intent to get X lock(s) at finer granularity

Lock Compatibility Matrix

	IS	IX	SIX	S	X
IS	✓	✓	✓	✓	—
IX	✓	✓	—	—	—
SIX	✓	—	—	—	—
S	✓	—	—	✓	—
X	—	—	—	—	—

Lock Granularity



sid	points	grade	
Bob	43	F	T2(X)
Joe	99	A	T1(X)
Alice	87	A	T1(X)
Suzy	50	F	T2(X)
Ted	73	A	T1(X)
Tim	12	F	T2(X)

Worksheet - Lock Granularity

Suppose a transaction, T1, wants to scan a table R and update a few of its tuples. What kind of locks should T1 have on R, its pages, and the tuples that are updated?

Suppose a transaction, T1, wants to scan a table R and update a few of its tuples. What kind of locks should T1 have on R, its pages, and the tuples that are updated?

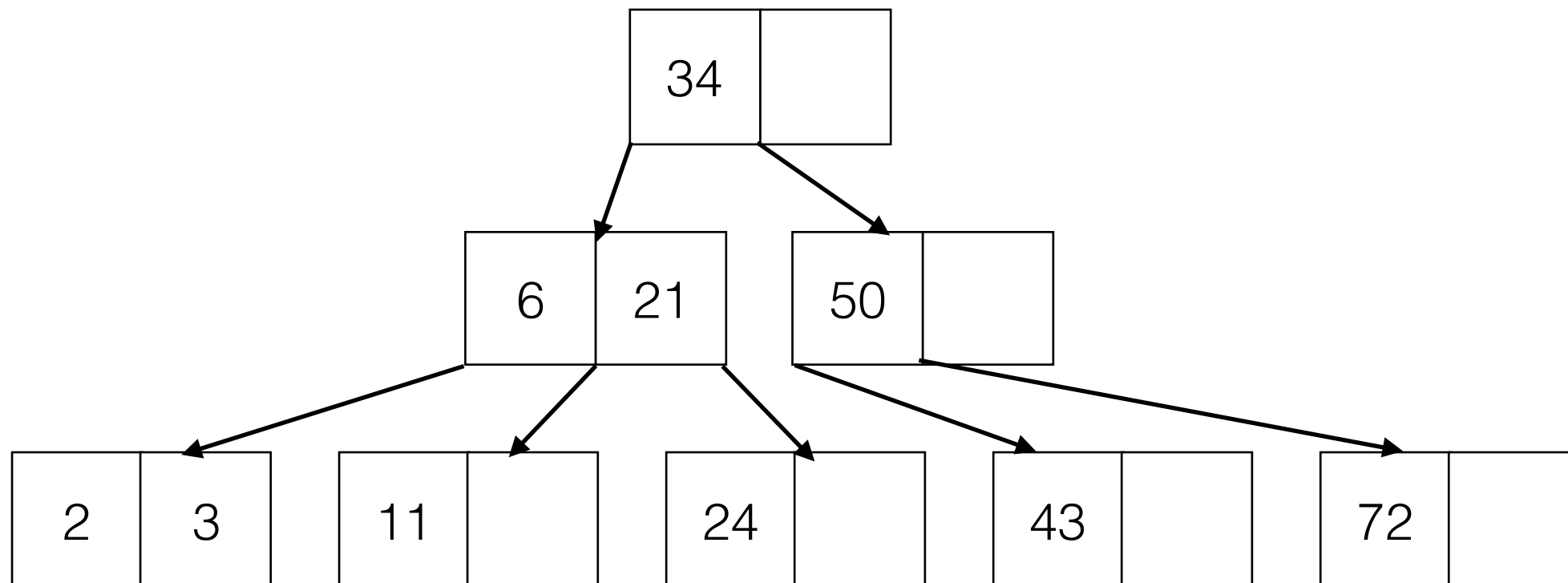
- SIX lock on R
- SIX lock on pages
- X lock on updated tuples

Is an S lock compatible with an IX lock?
Explain why or why not. Make your
description as simple as possible.

Is an S lock compatible with an IX lock?
Explain why or why not. Make your
description as simple as possible.

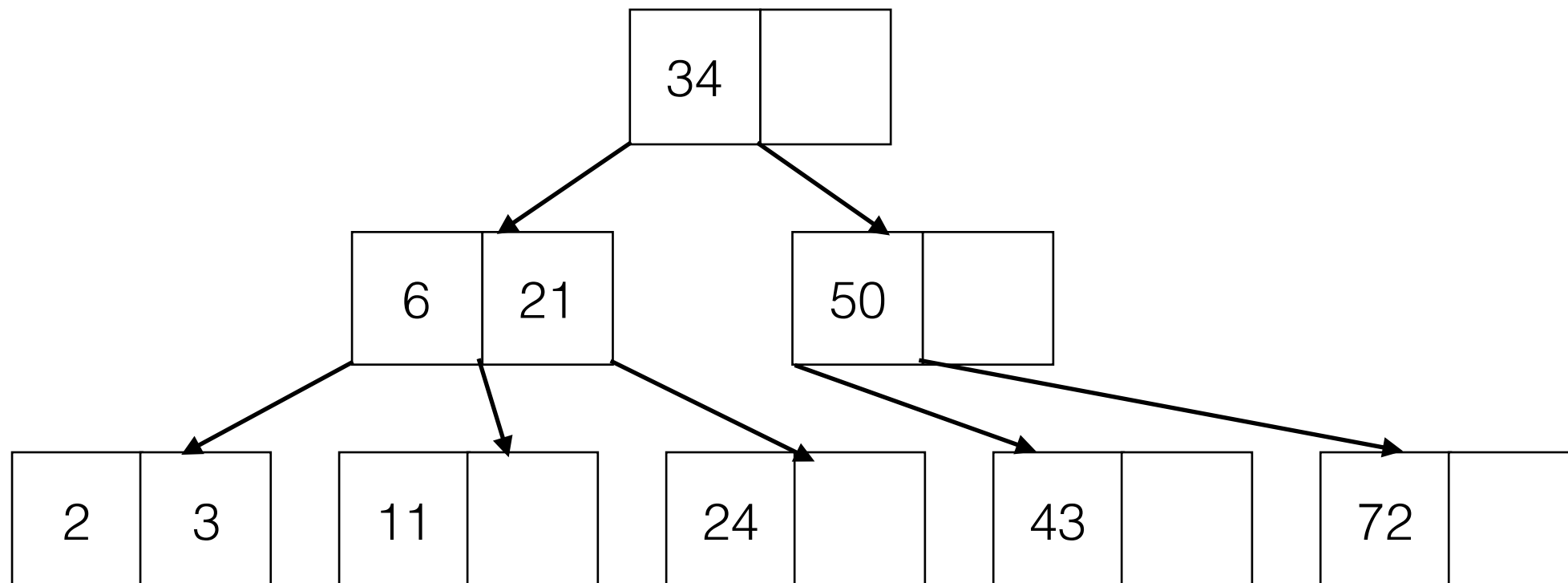
- Incompatible:
 - T1 has S lock on Students table to calculate average grade
 - T2 wants IX lock to change some grades

Index Concurrency

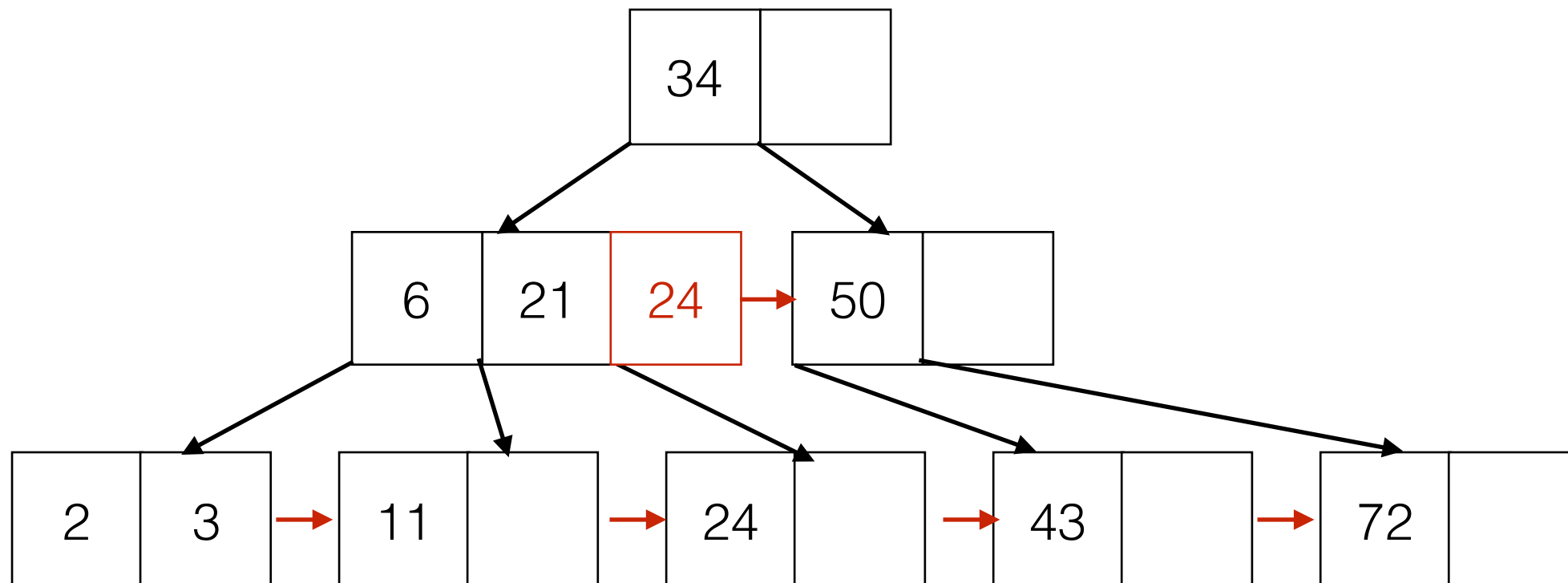


2PL on B+ tree is a bad idea!
Upper levels just to direct traffic.

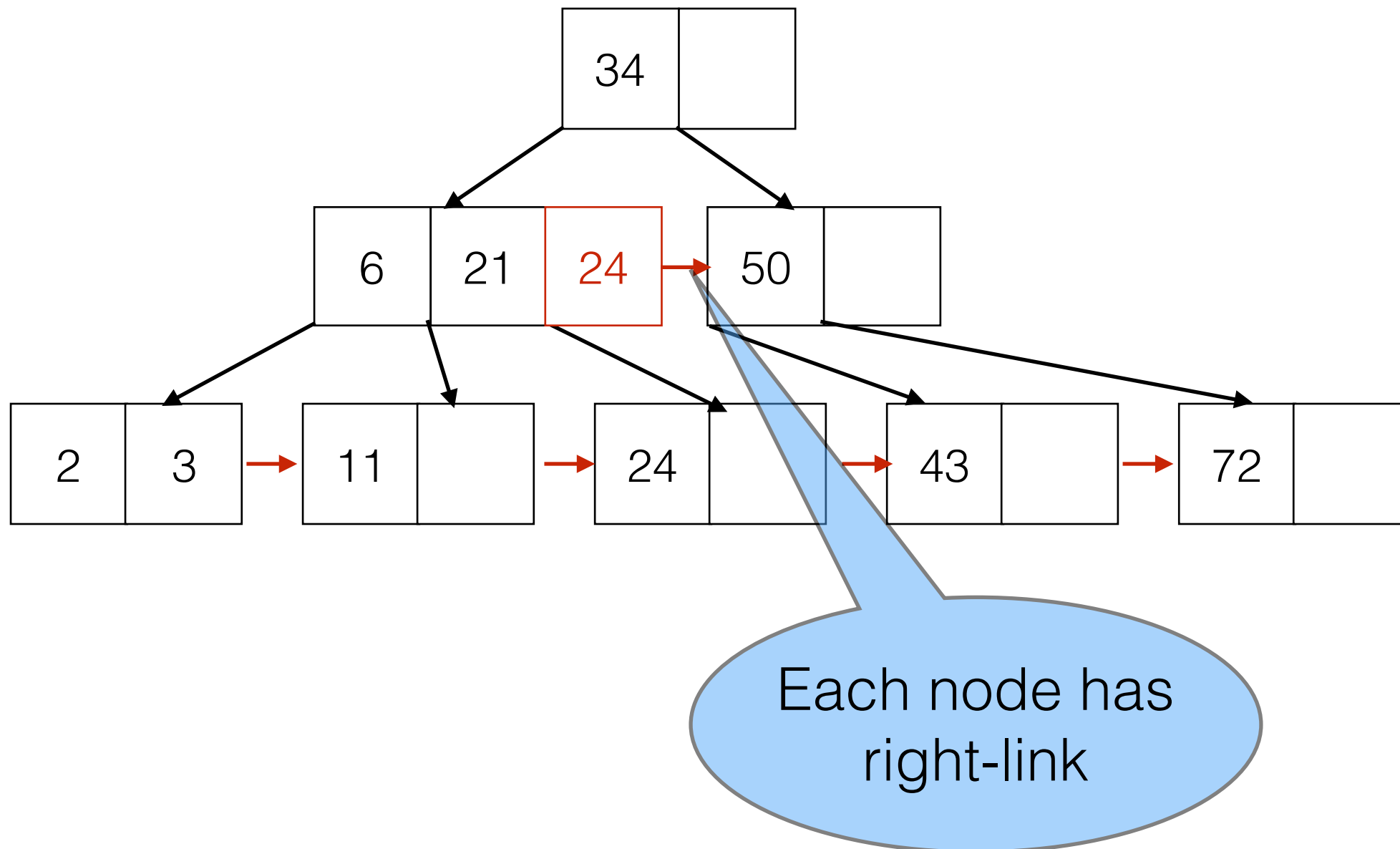
B+ Tree



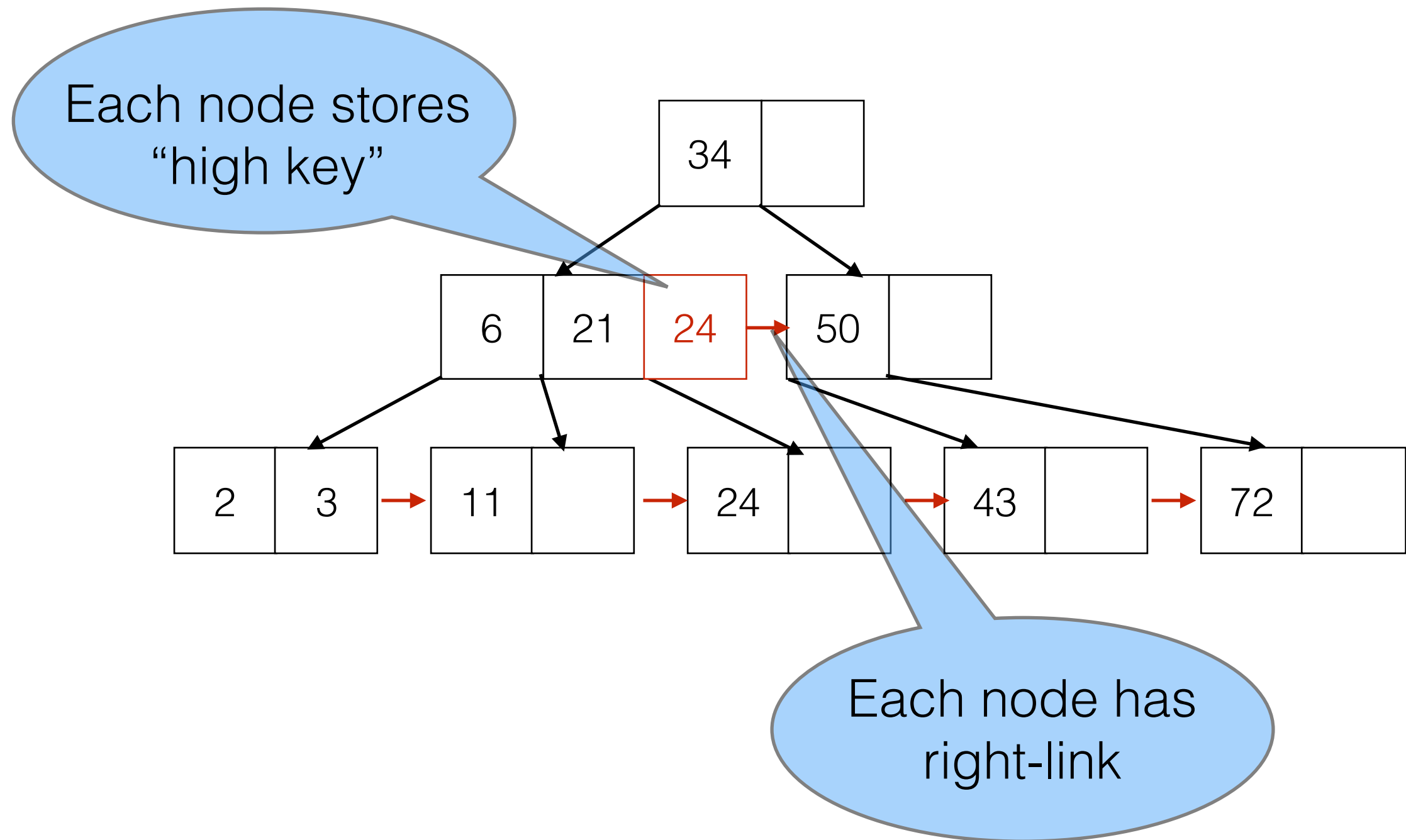
B-Link Tree



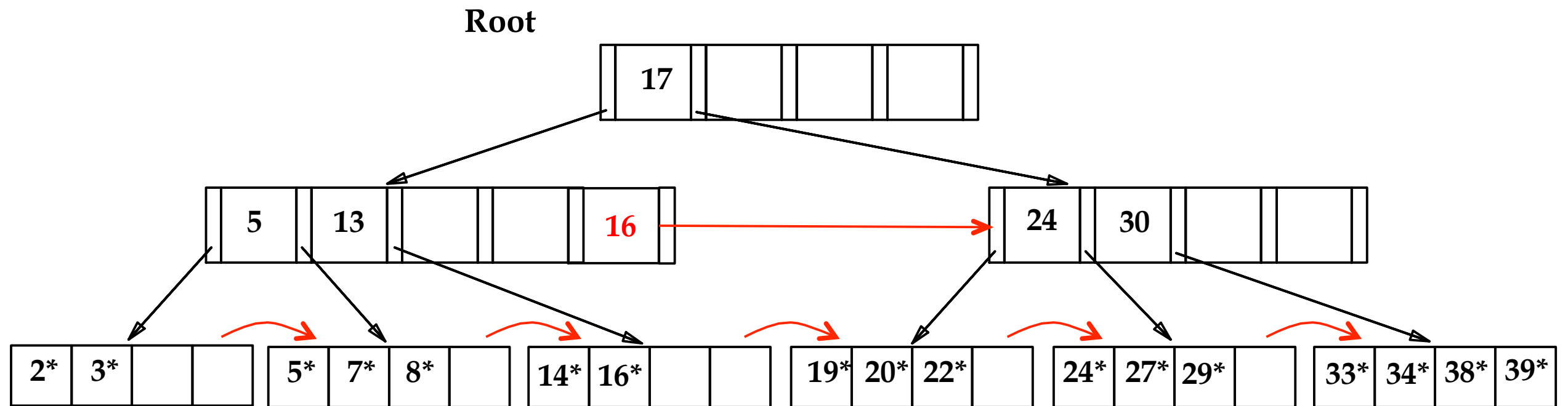
B-Link Tree



B-Link Tree

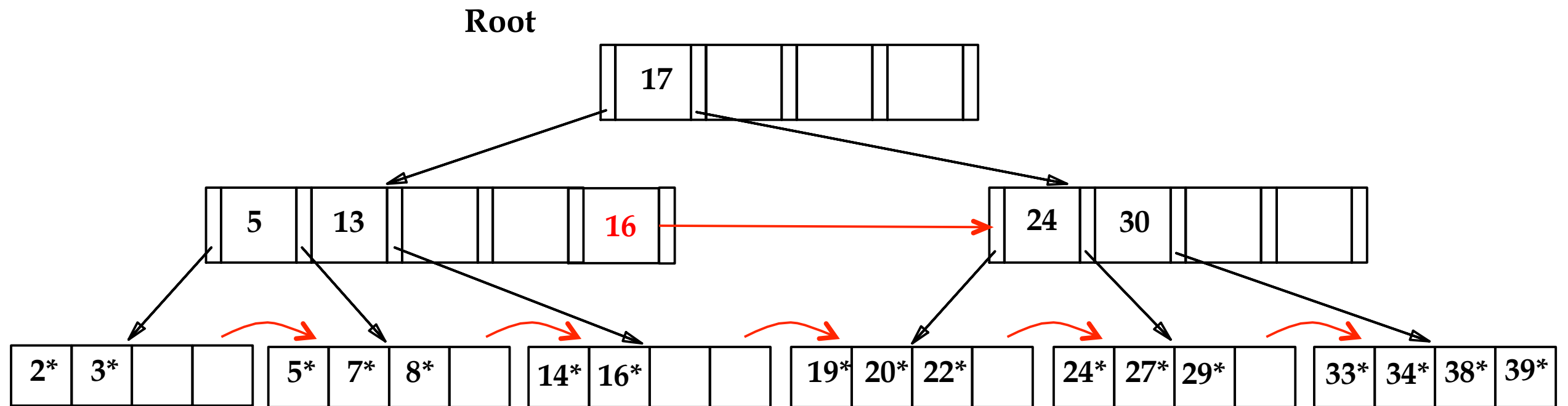


Search with Move Right



T1 searching for 39

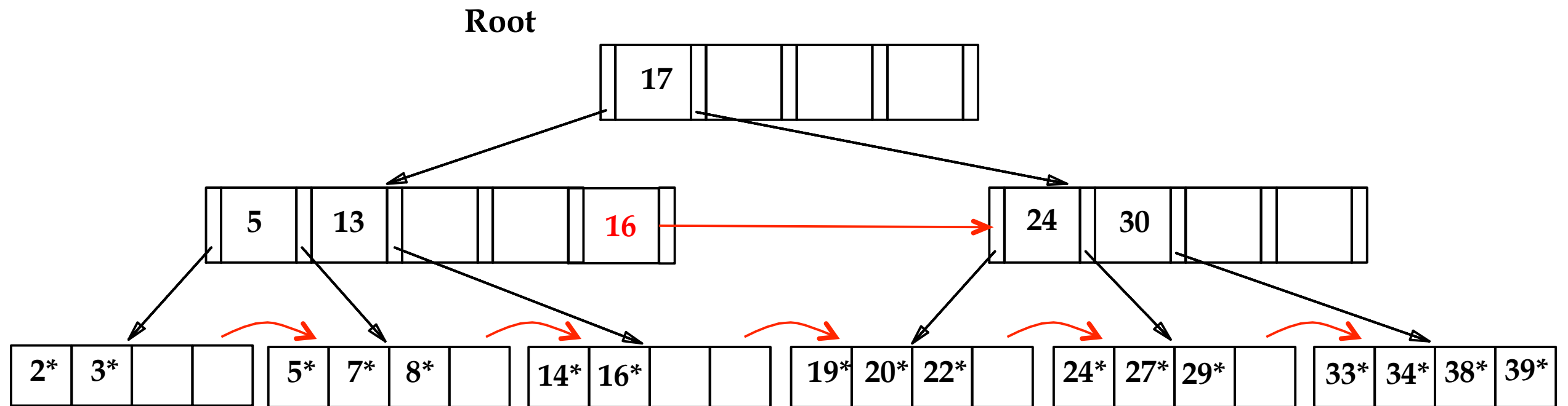
Search with Move Right



T1 searching for 39

Reads: [24 | 30 | |]

Search with Move Right

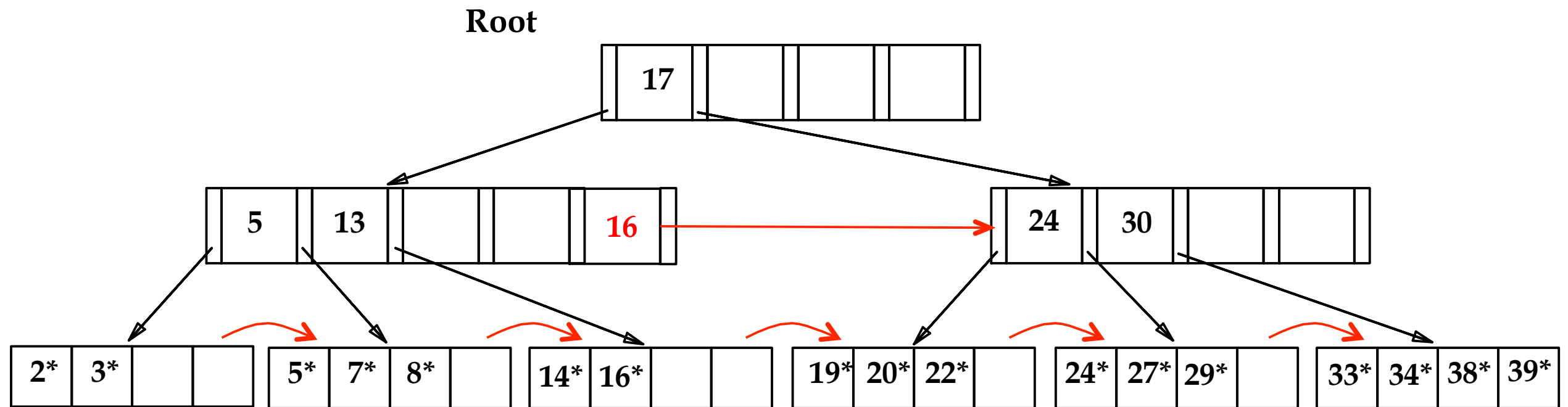


T1 searching for 39

Reads: [24 | 30 | |]

T2, inserting 38, splits rightmost leaf

Search with Move Right



T1 searching for 39

Reads:

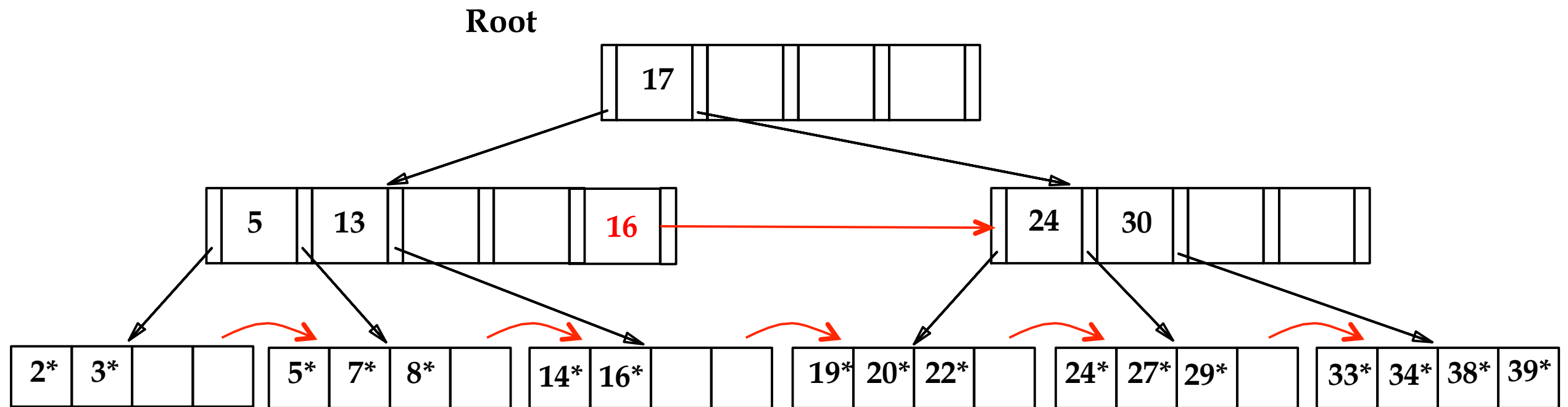
24	30		
----	----	--	--

T2, inserting 38, splits rightmost leaf

T1 arrives at leaf that looks like:

33*	34*	35*	
-----	-----	-----	--

Search with Move Right



T1 searching for 39

Reads: [24, 30, ,]

T2, inserting 38, splits rightmost leaf

T1 arrives at leaf that looks like: [33*, 34*, 35*,]

T1 moves right until it arrives at: [38*, 39*, ,]

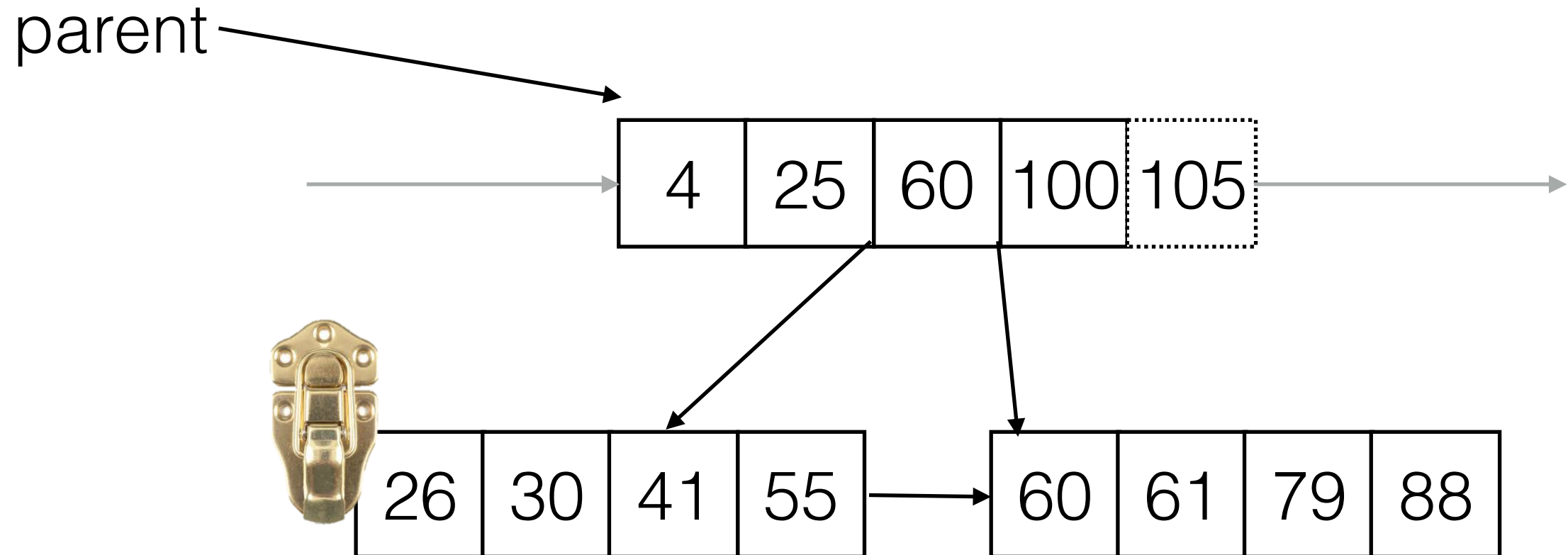
Latches

- Unlatches almost immediately
- Fast
 - In-memory next to data structure
- Only DBMS engineers have access to APIs

Insertion

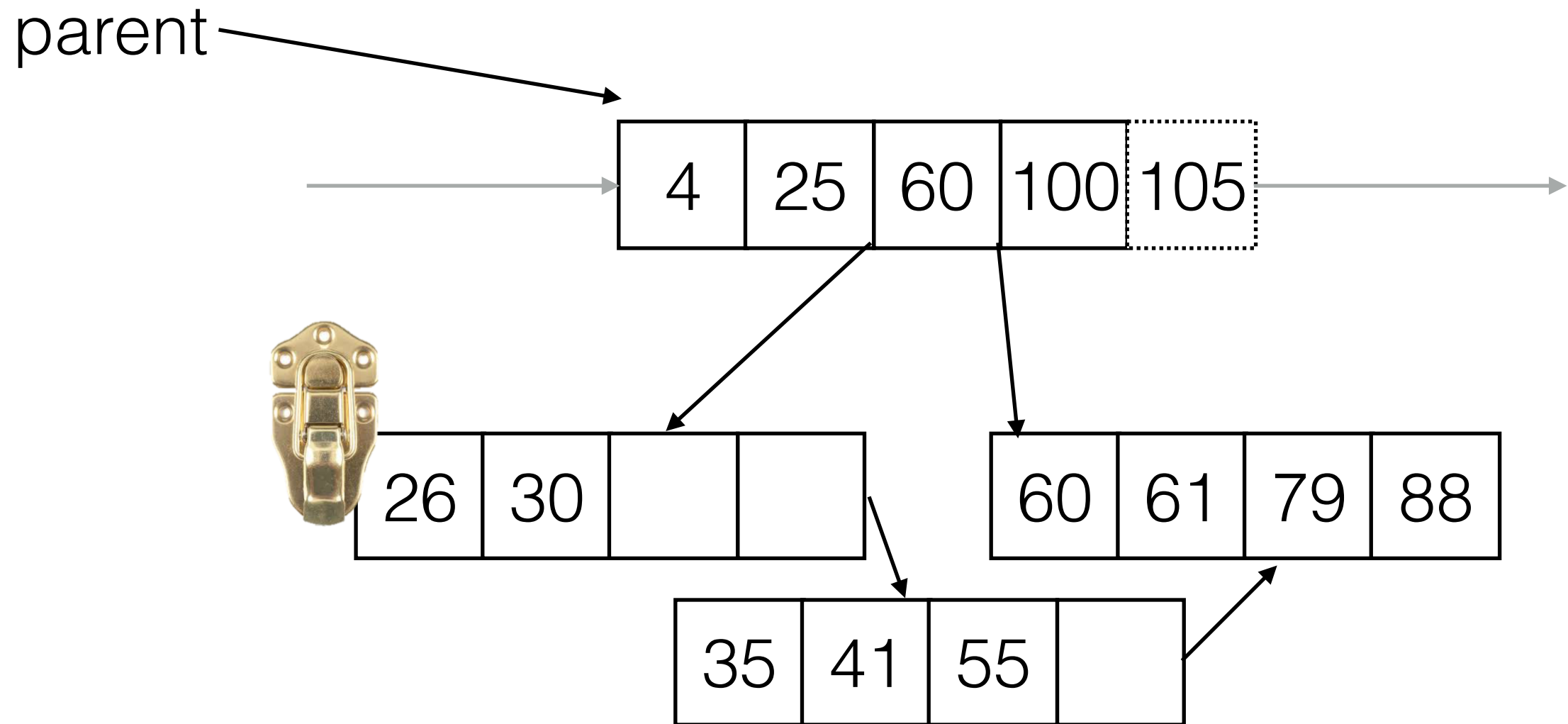
- While traversing to leaf, keep stack of parents
- If splits leaf, latch on parent
 - If high key on parent $<$ key to insert, move right while latching and unlatching
 - Else insert key

doInsertion: split



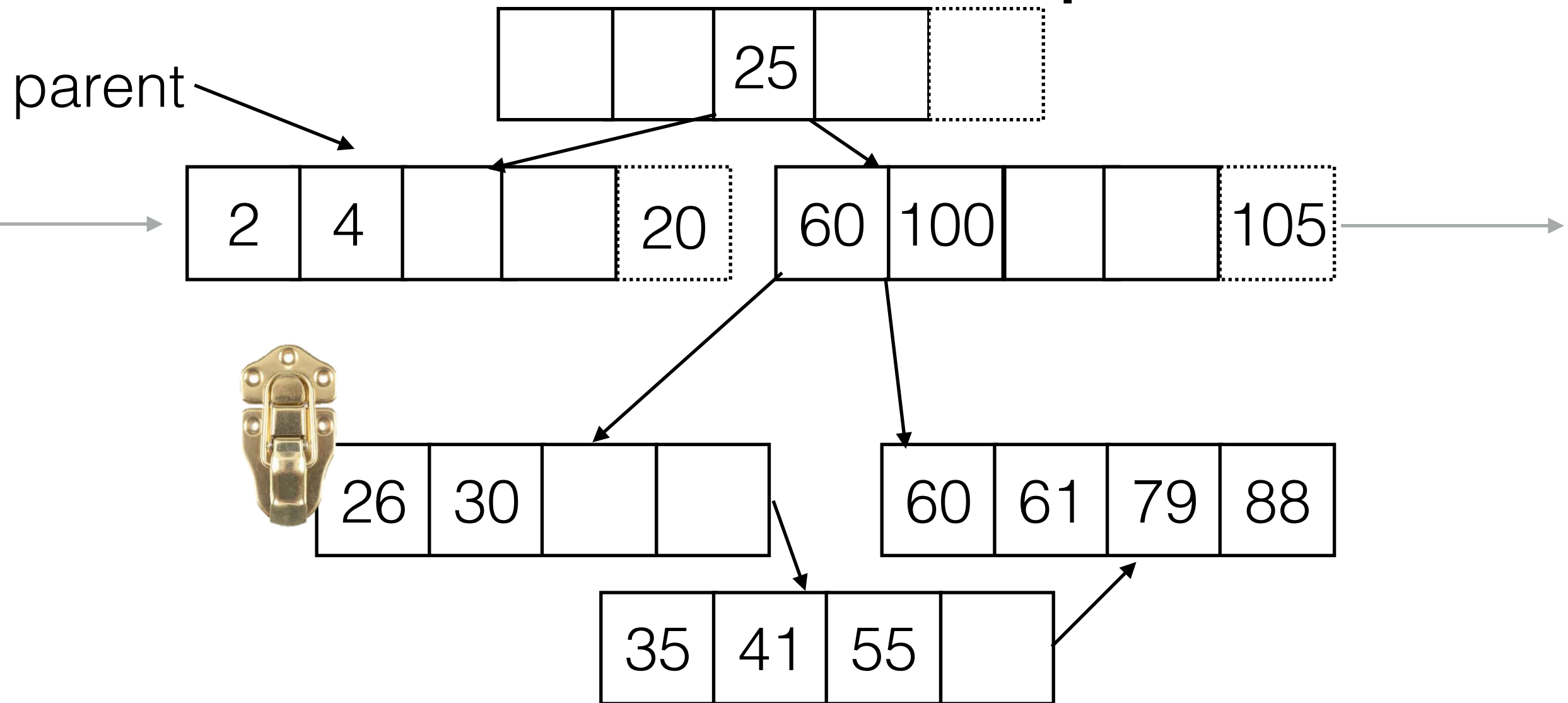
Insert 35

doInsertion: split



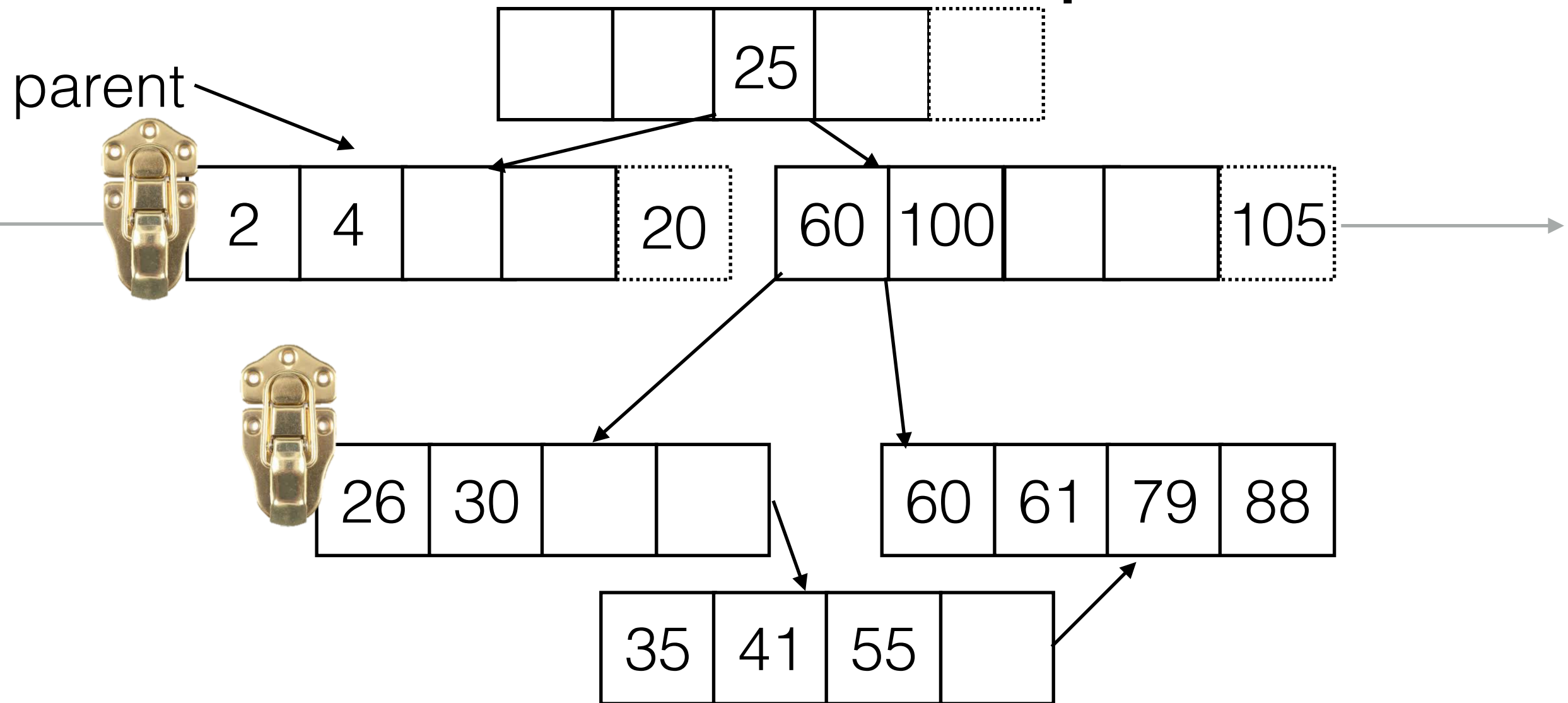
Insert 35

doInsertion: split



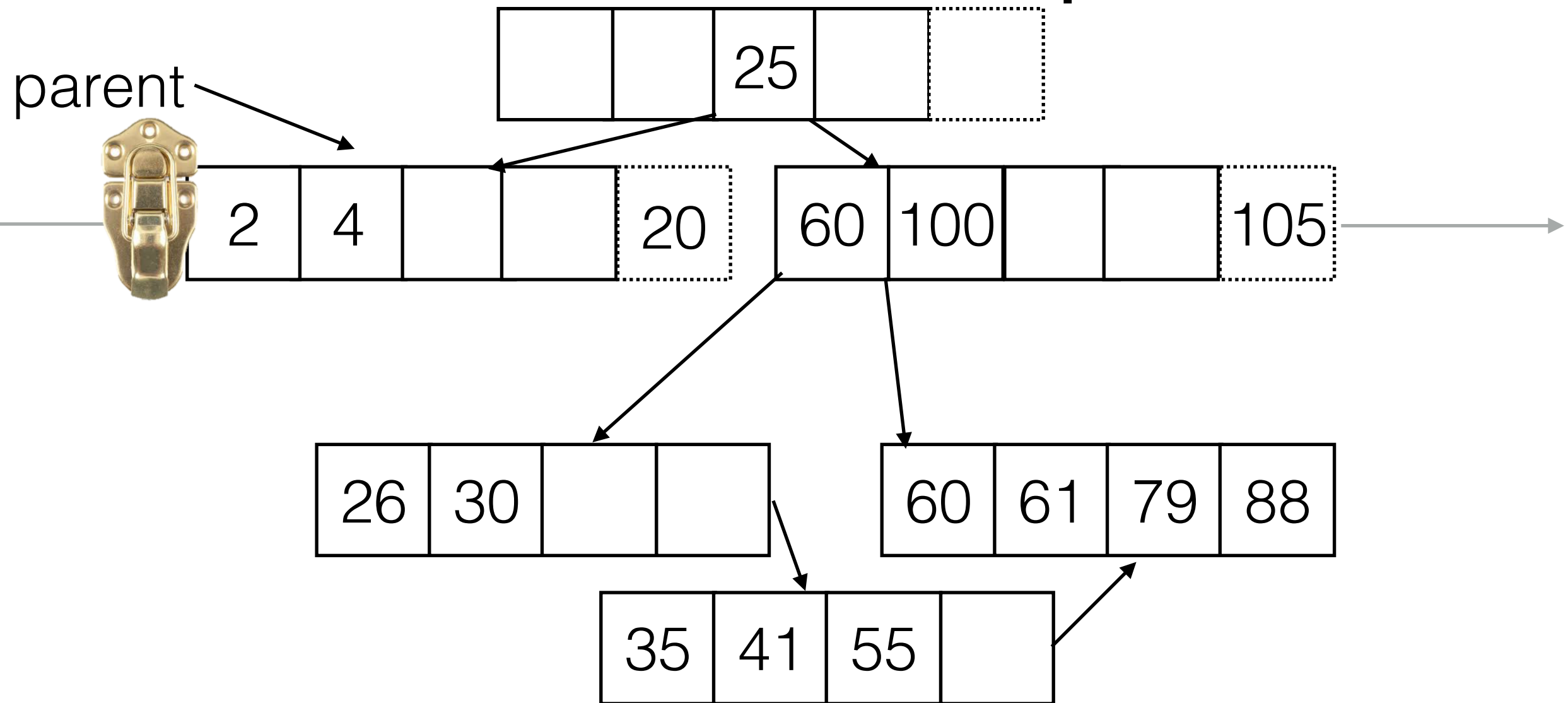
Insert 35

doInsertion: split



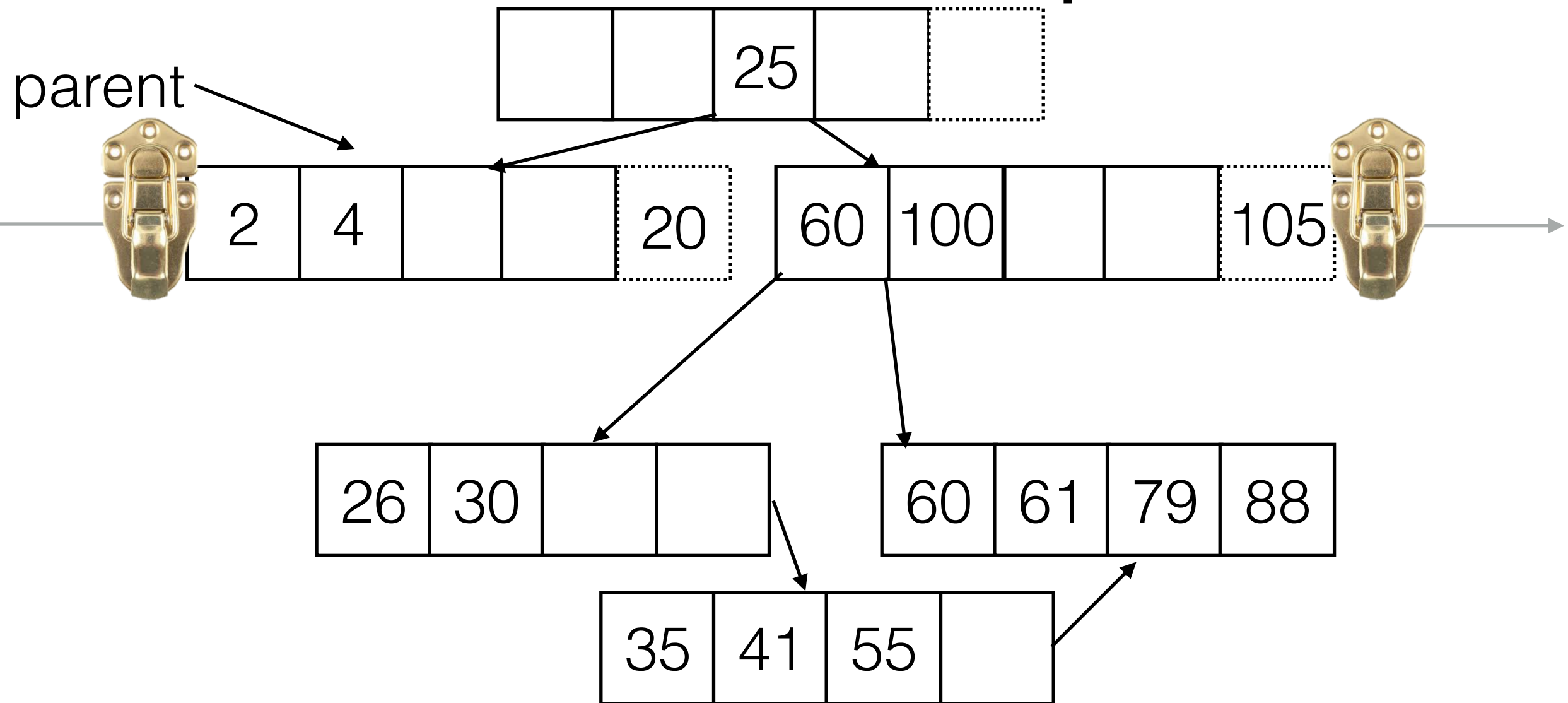
Insert 35

doInsertion: split



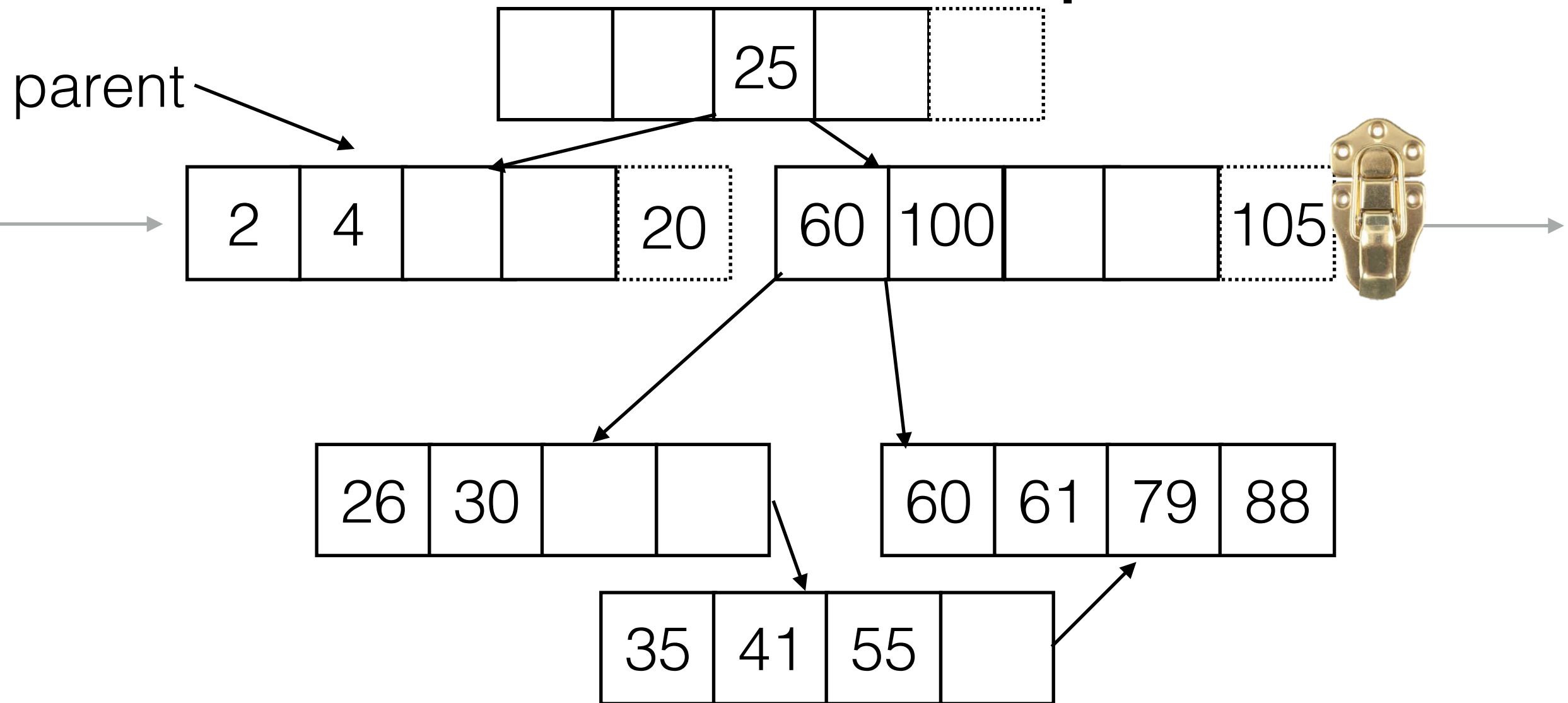
Insert 35

doInsertion: split



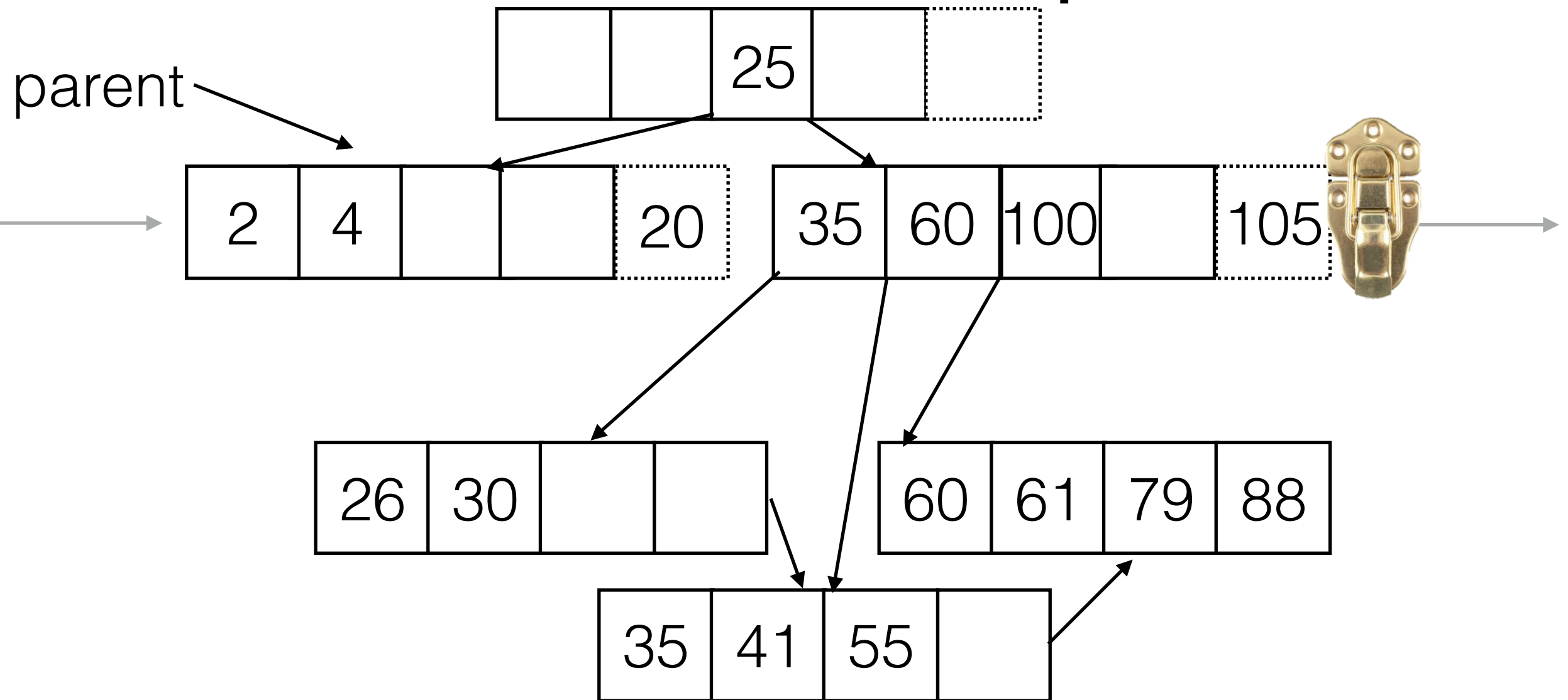
Insert 35

doInsertion: split



Insert 35

doInsertion: split



Insert 35

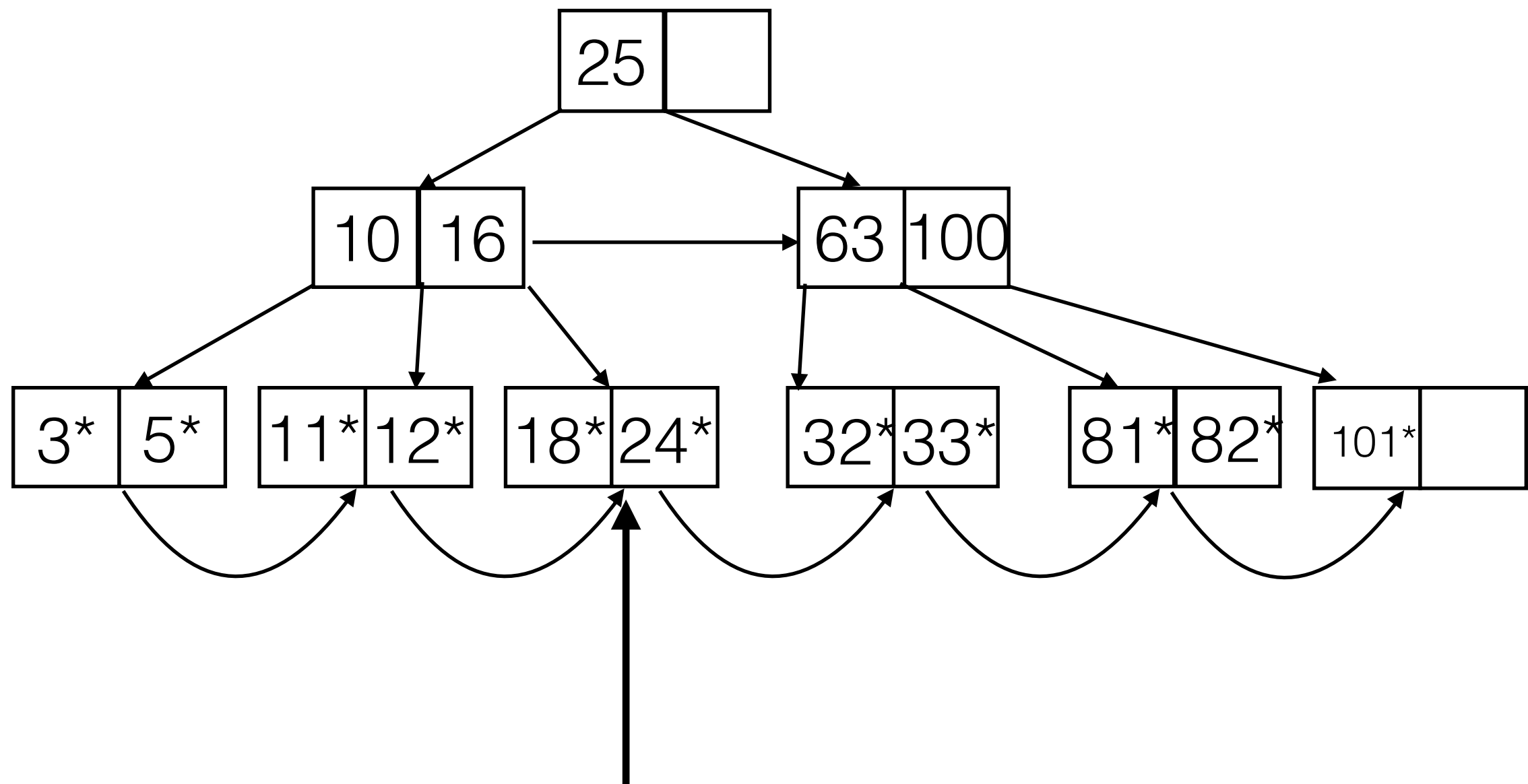
Worksheet - B-Link Trees

What is the difference
between a lock and a latch?

What is the difference between a lock and a latch?

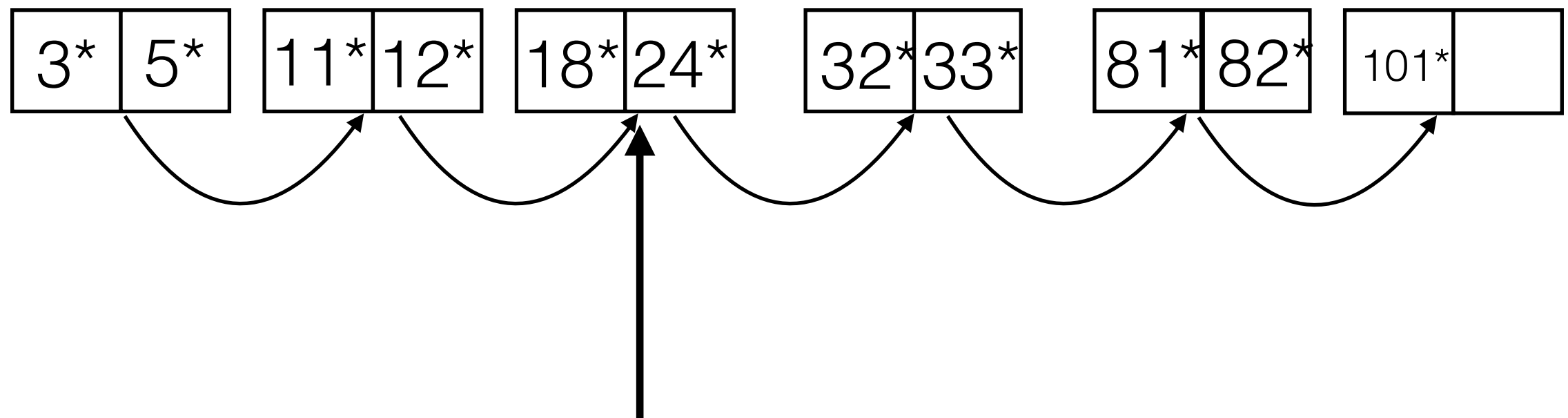
- Latches:
 - Internal to DBMS
 - Provide memory consistency
 - Unlatched almost immediately
- Locks:
 - Often held for entire transaction duration
 - Provide logical transactional consistency

How many pages will be read to find 24?
How many latches did T1 have to make?



T1 has pointer to leaf page to read

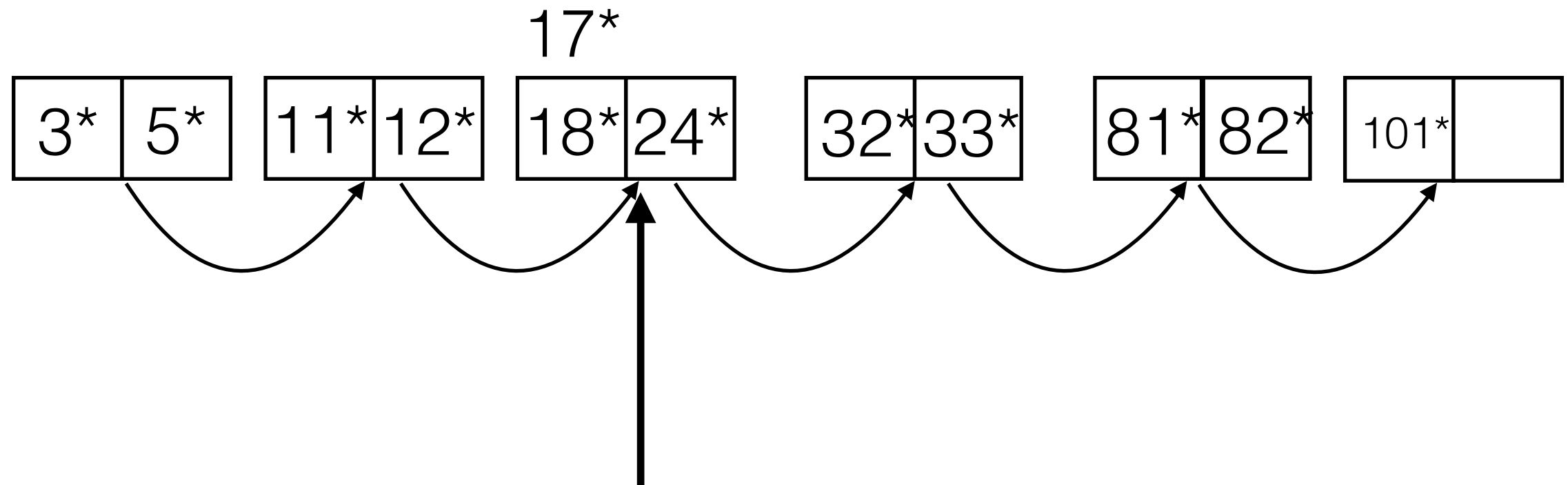
How many pages will be read to find 24?
How many latches did T1 have to make?



T1 has pointer to leaf page to read

How many pages will be read to find 24?
How many latches did T1 have to make?

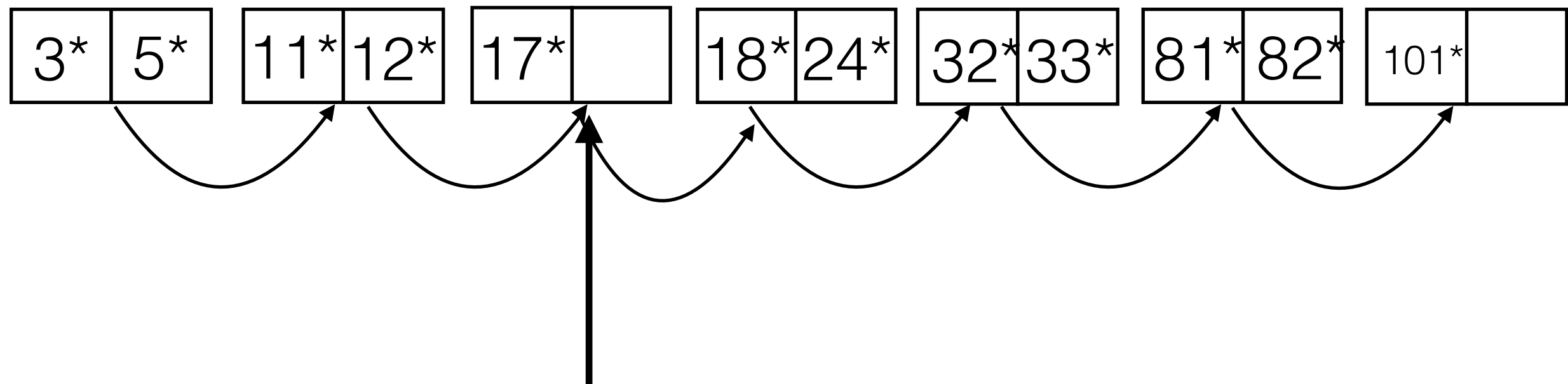
Insert 17



T1 has pointer to leaf page to read

How many pages will be read to find 24?
How many latches did T1 have to make?

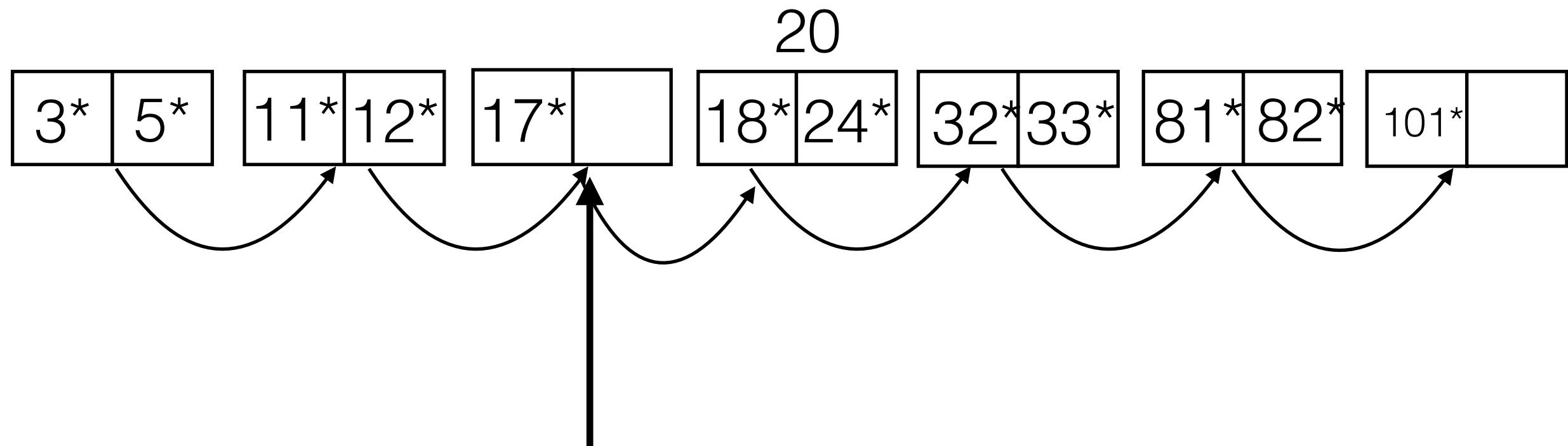
Insert 17



T1 has pointer to leaf page to read

How many pages will be read to find 24?
How many latches did T1 have to make?

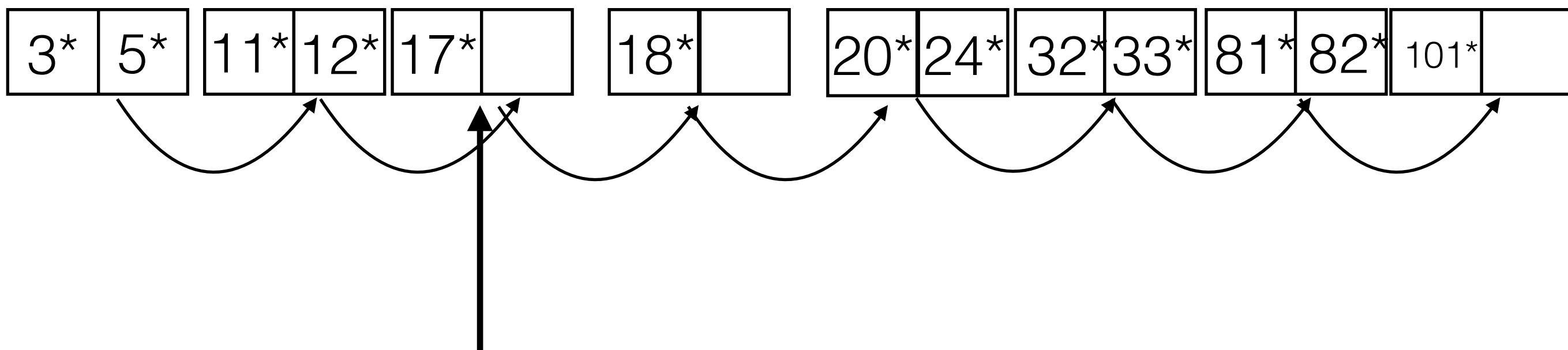
Insert 20



T1 has pointer to leaf page to read

How many pages will be read to find 24?
How many latches did T1 have to make?

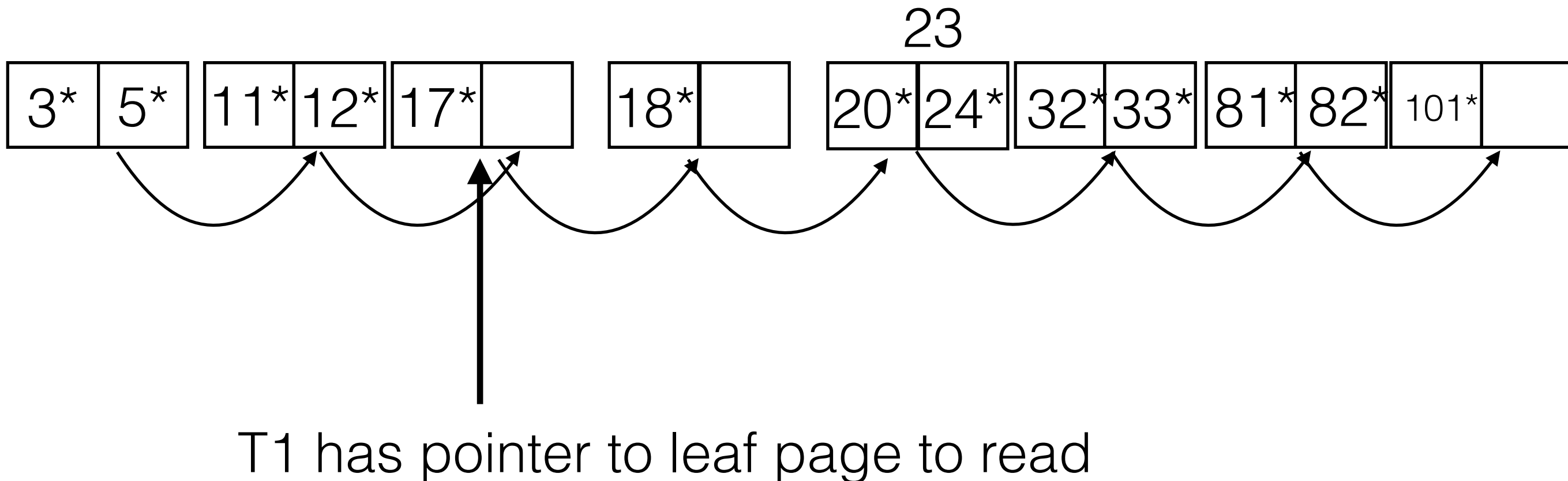
Insert 20



T1 has pointer to leaf page to read

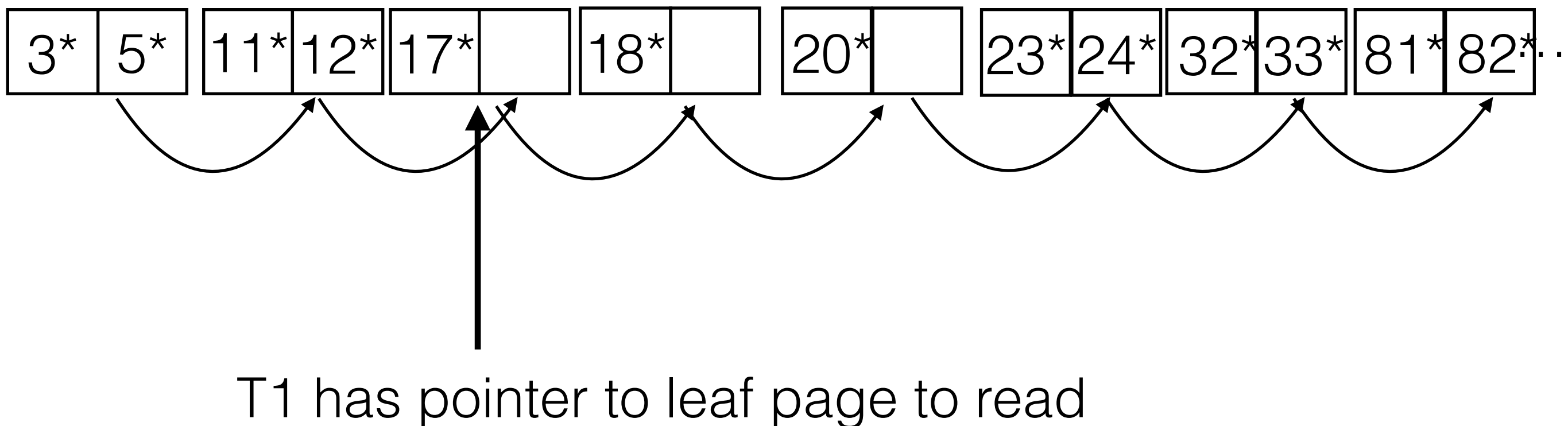
How many pages will be read to find 24?
How many latches did T1 have to make?

Insert 23

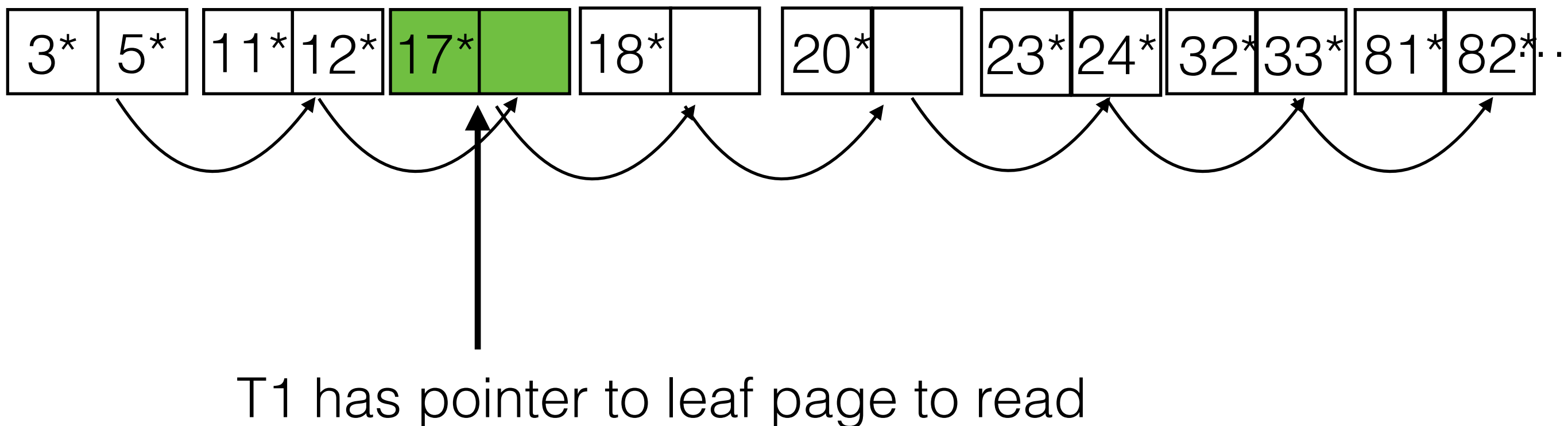


How many pages will be read to find 24?
How many latches did T1 have to make?

Insert 23

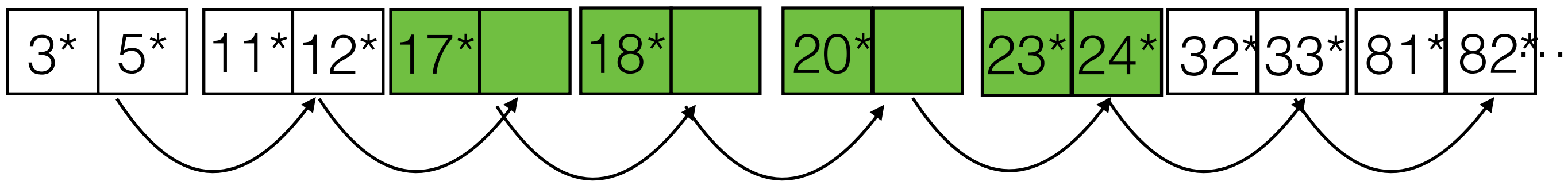


How many pages will be read to find 24?
How many latches did T1 have to make?

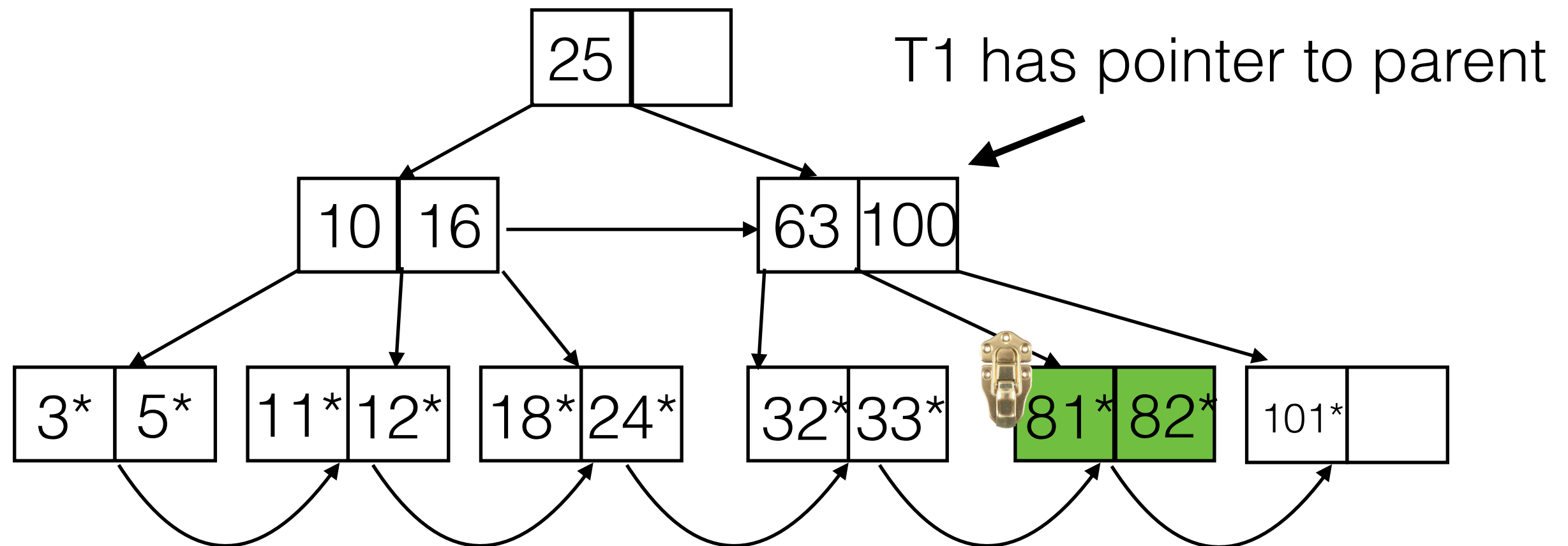


How many pages will be read to find 24?
How many latches did T1 have to make?

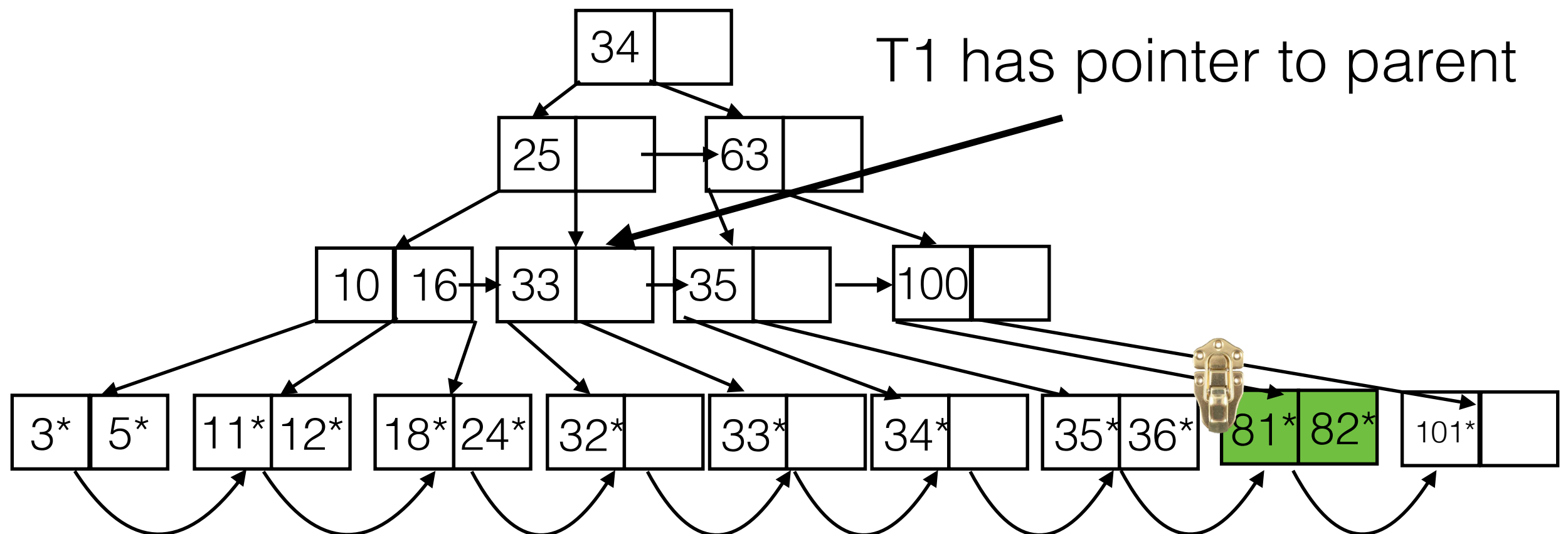
4 pages read
0 latches



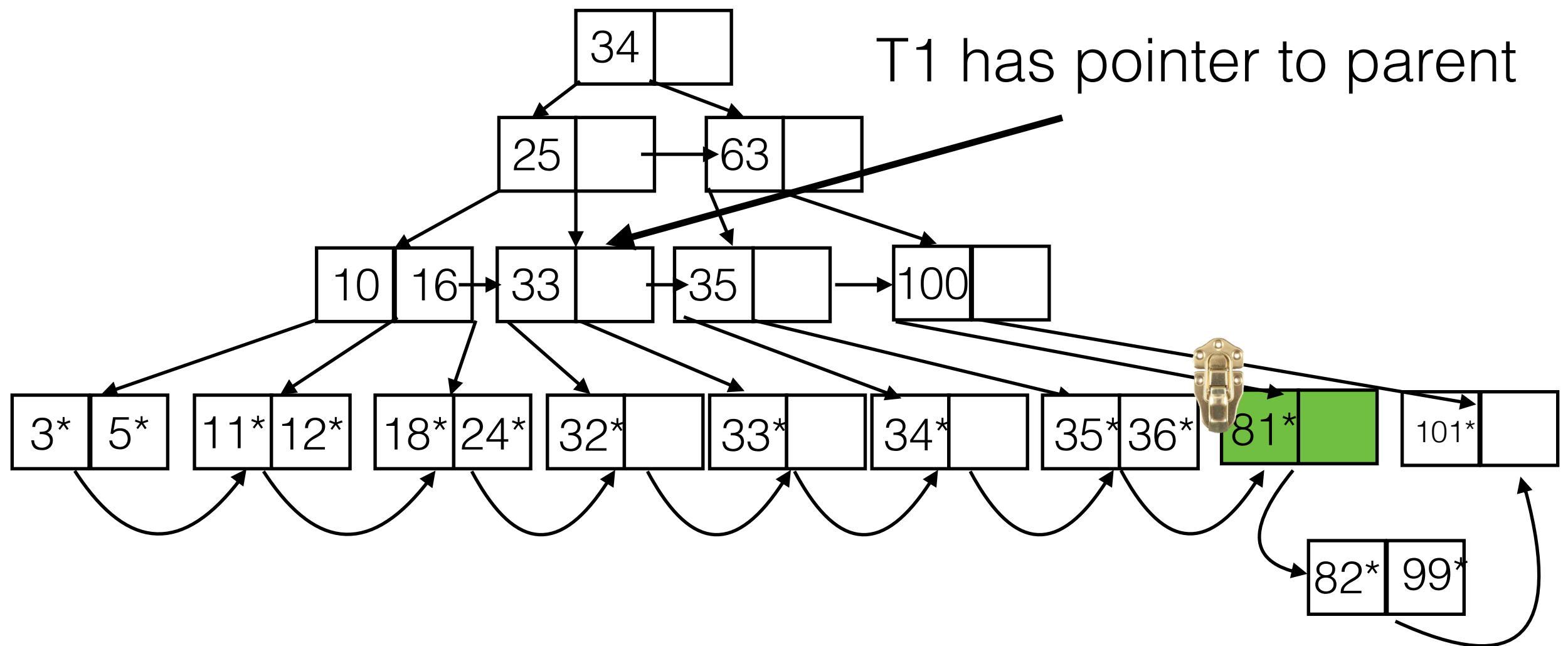
How many latches made when
inserting 99?



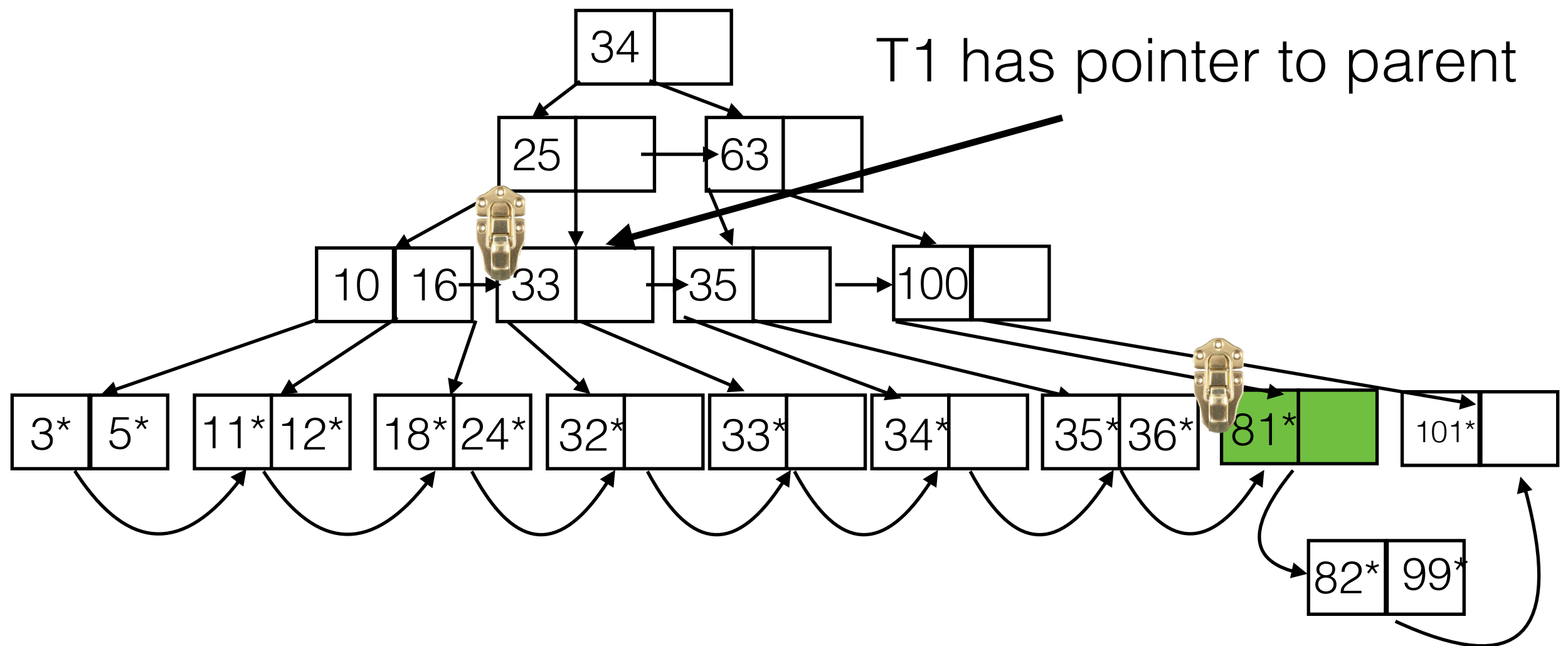
How many latches made when
inserting 99?



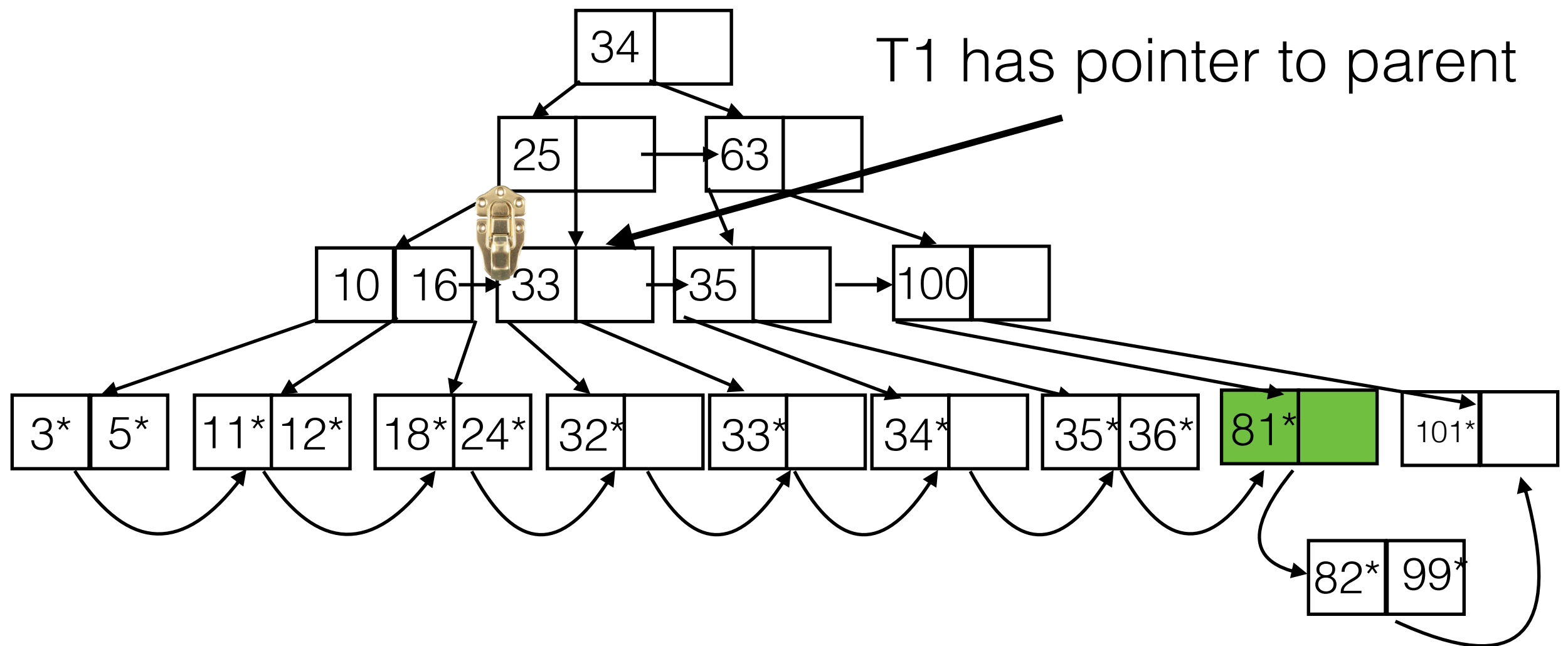
How many latches made when
inserting 99?



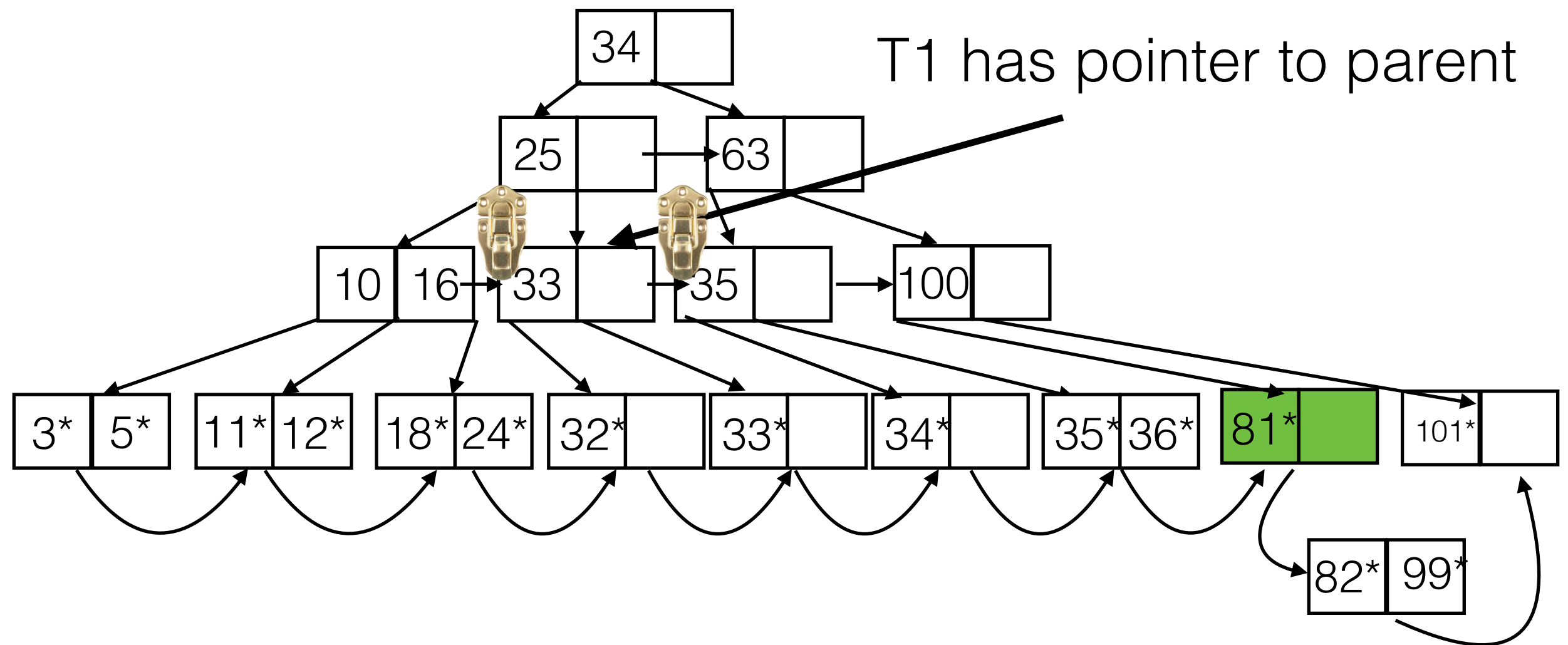
How many latches made when
inserting 99?



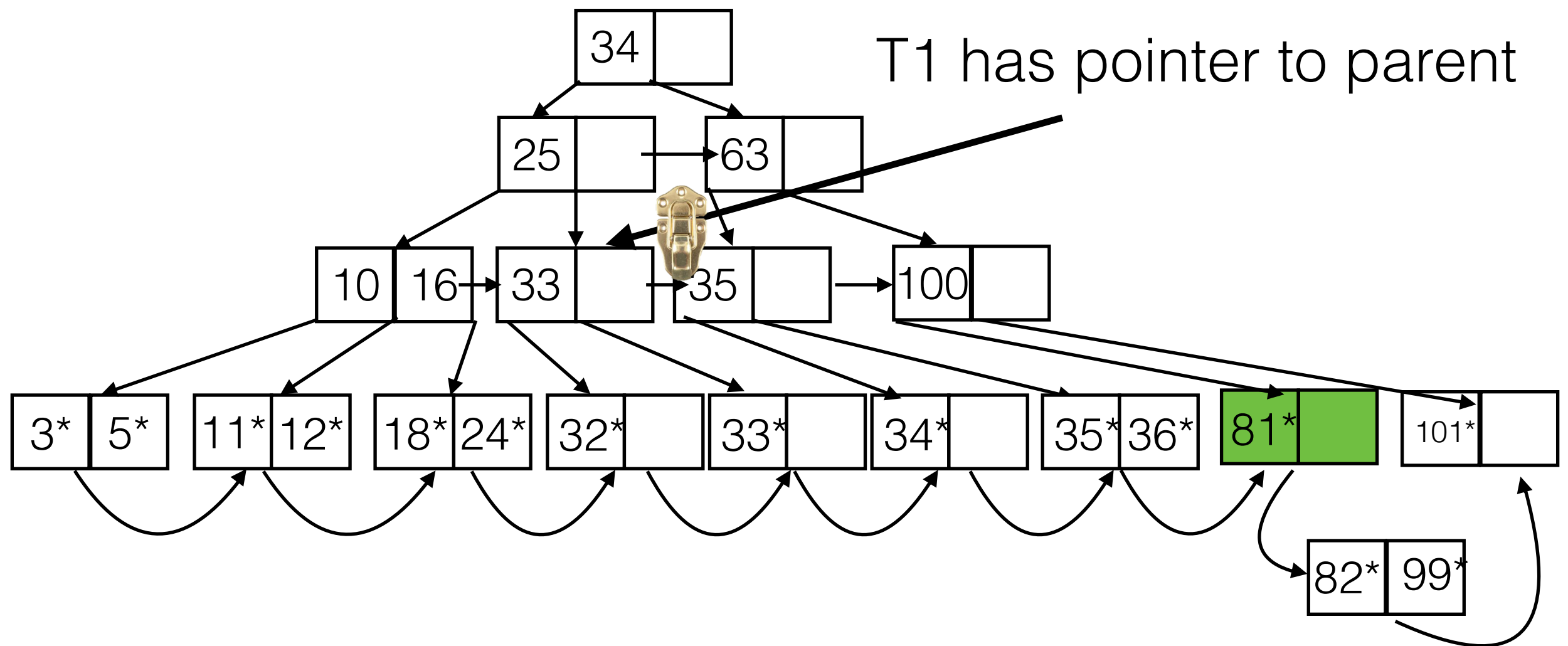
How many latches made when
inserting 99?



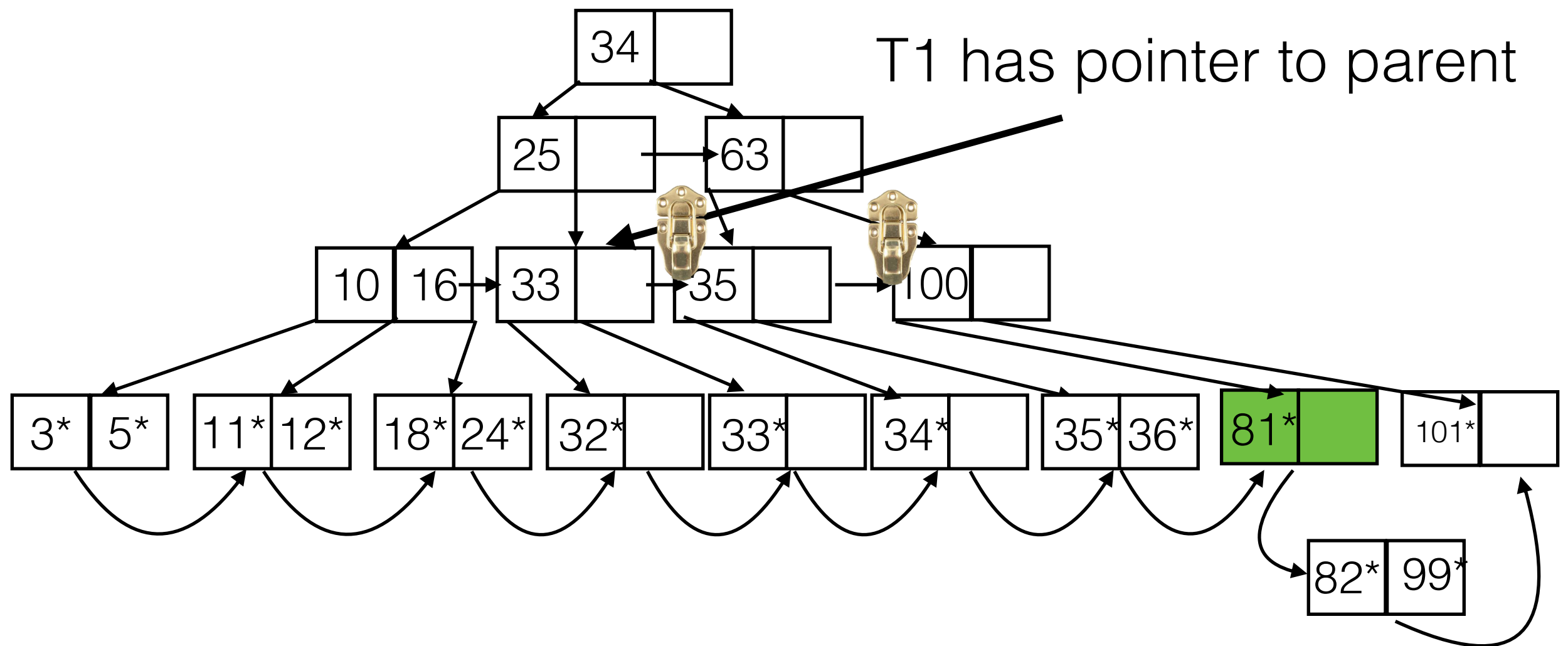
How many latches made when
inserting 99?



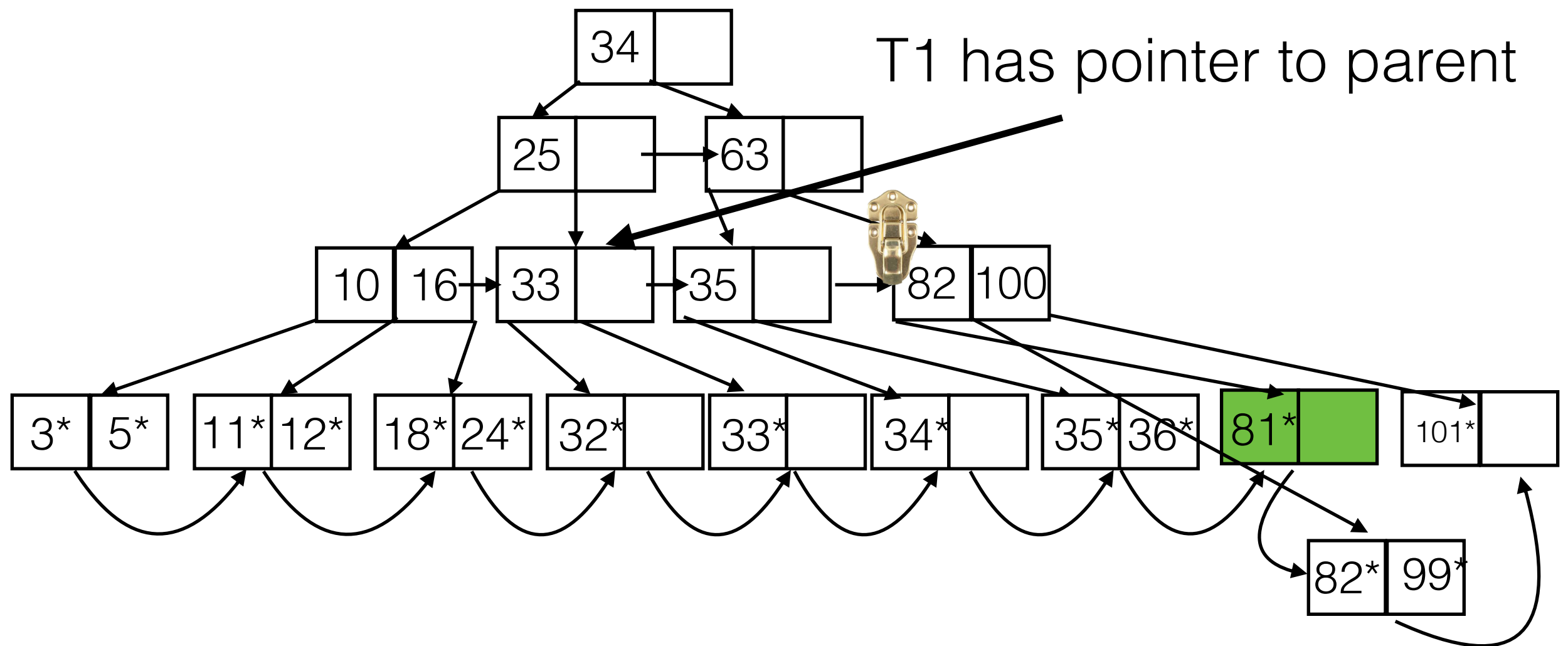
How many latches made when
inserting 99?



How many latches made when
inserting 99?



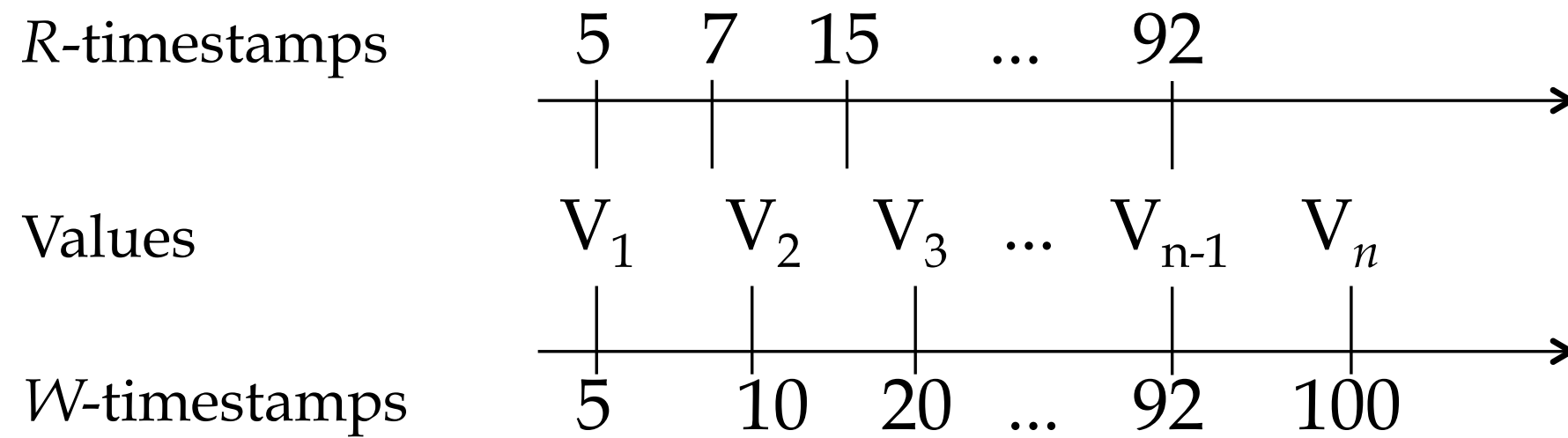
How many latches made when
inserting 99?



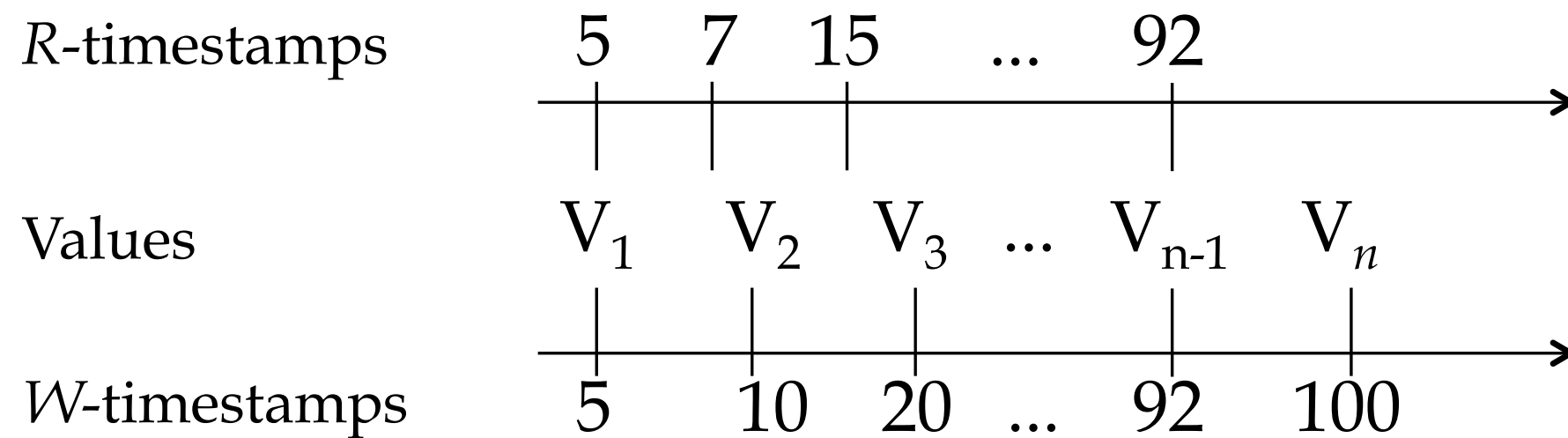
Multiversion Concurrency Control

- Alternative to 2PL
 - Less waiting, but more aborts
- Timestamp Ordered MVCC:
 - Each transaction gets timestamp upon entry
 - Keep timeline of read timestamps and versions

TO-MVCC

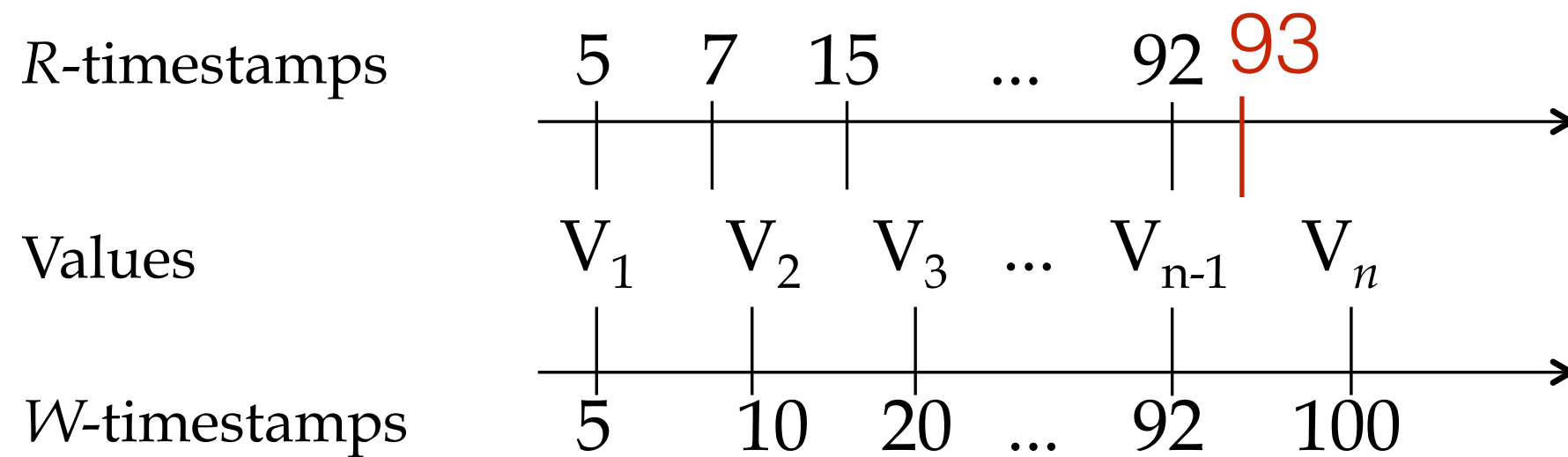


TO-MVCC



Reads: Read version with biggest timestamp smaller than given timestamp

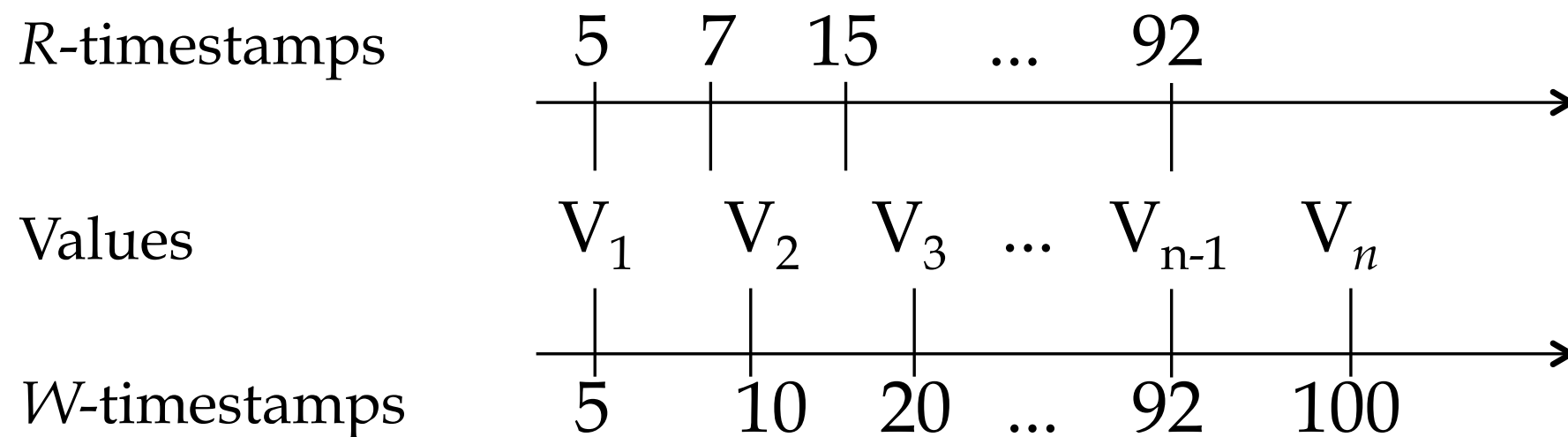
TO-MVCC



Reads: Read version with biggest timestamp smaller than current timestamp

$R(X) @ 93$ will read V_{n-1}

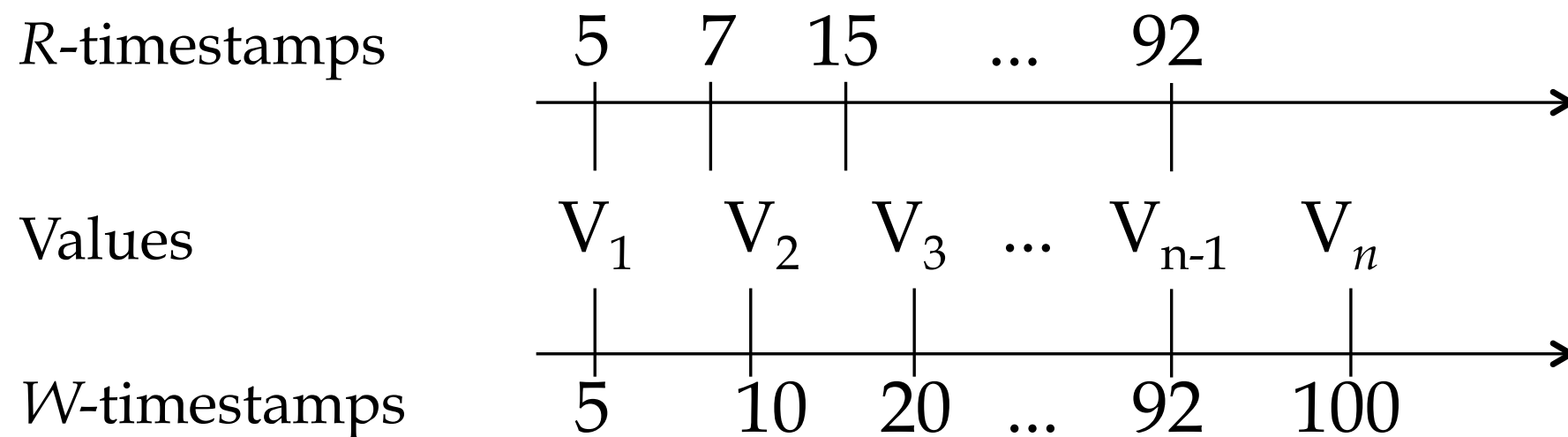
TO-MVCC



Writes: Find interval from given timestamp X to smallest timestamp bigger than X

If there a read is in the interval, **abort!**

TO-MVCC

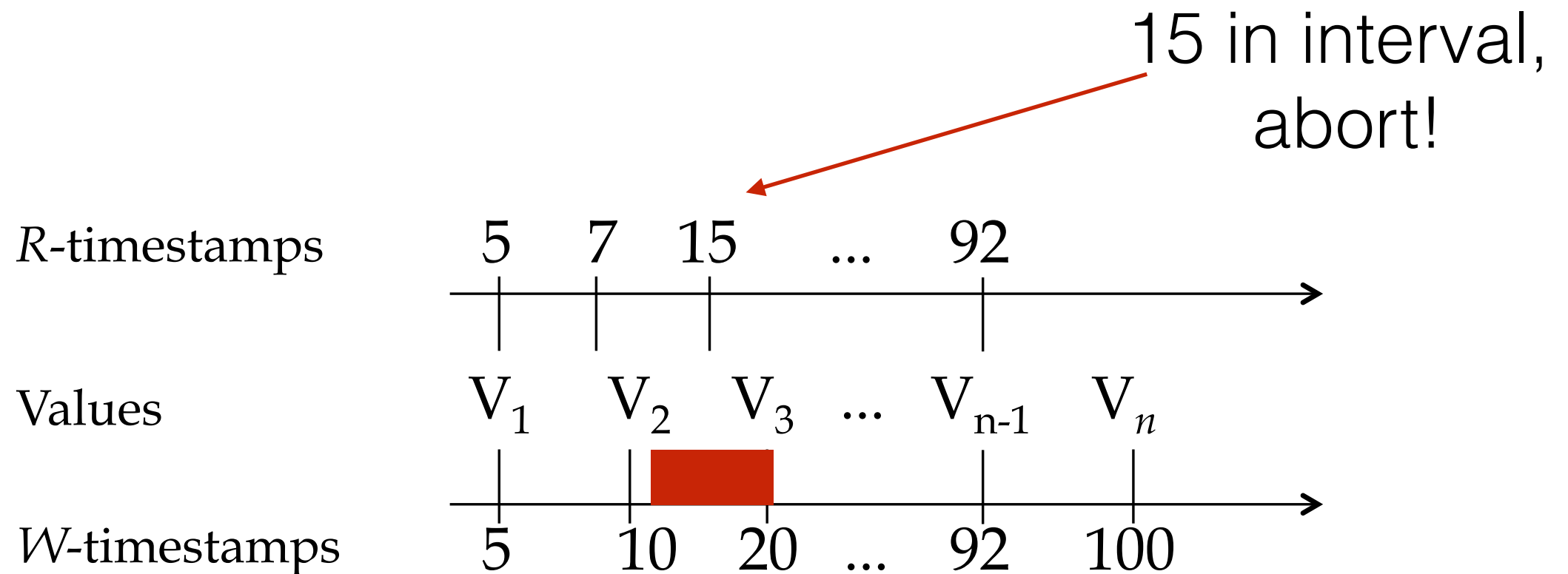


Writes: Find interval from given timestamp X to smallest timestamp bigger than X

If there a read is in the interval, **abort!**

$W(X)$ @ 11

TO-MVCC

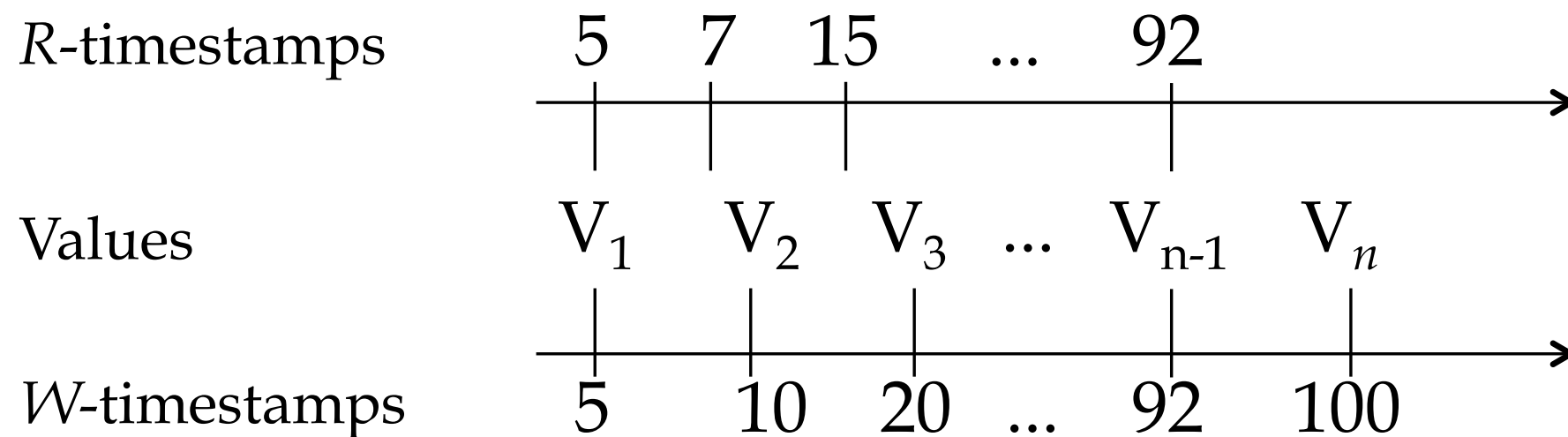


Writes: Find interval from given timestamp X to smallest timestamp bigger than X

If there a read is in the interval, **abort!**

$W(X)$ @ 11

TO-MVCC

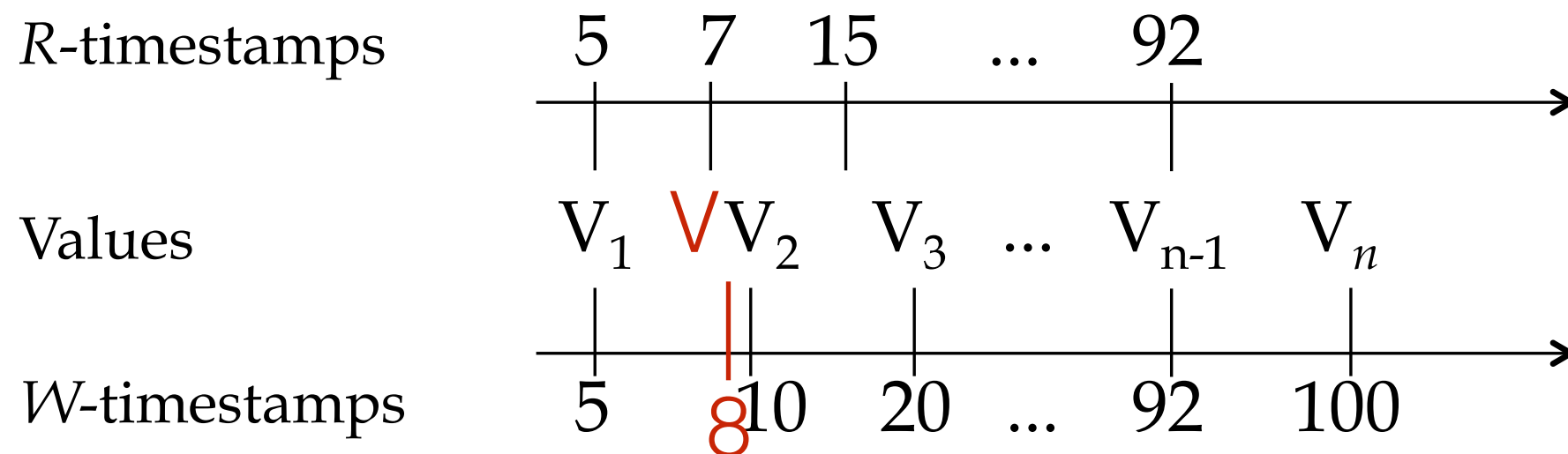


Writes: Find interval from given timestamp X to smallest timestamp bigger than X

If there a read is in the interval, **abort!**

$W(X)$ @ 8

TO-MVCC



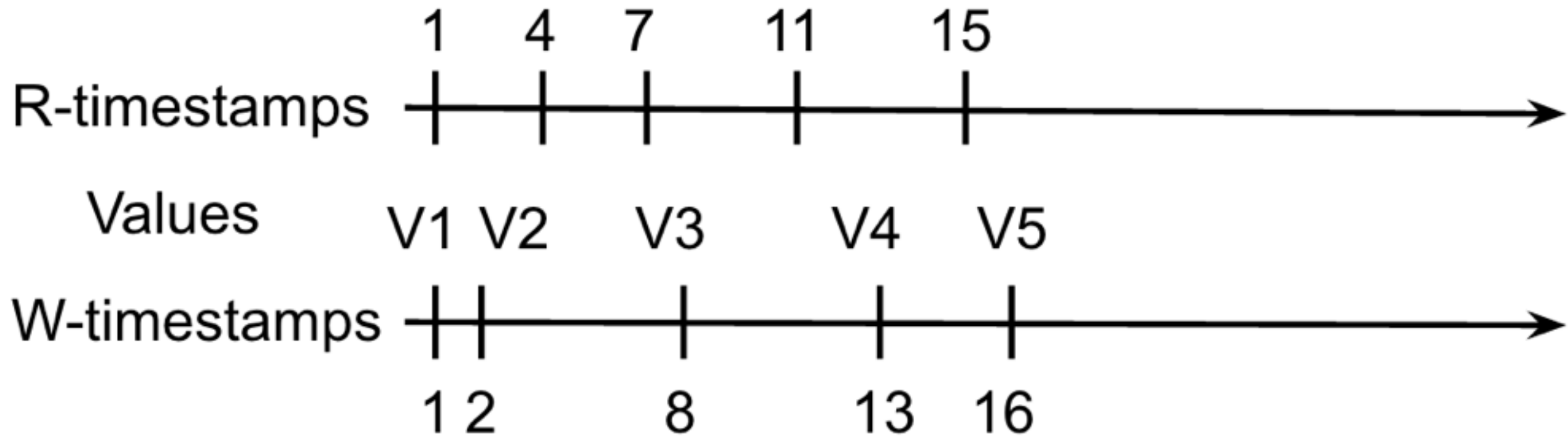
Writes: Find interval from given timestamp X to smallest timestamp bigger than X

If there a read is in the interval, **abort!**

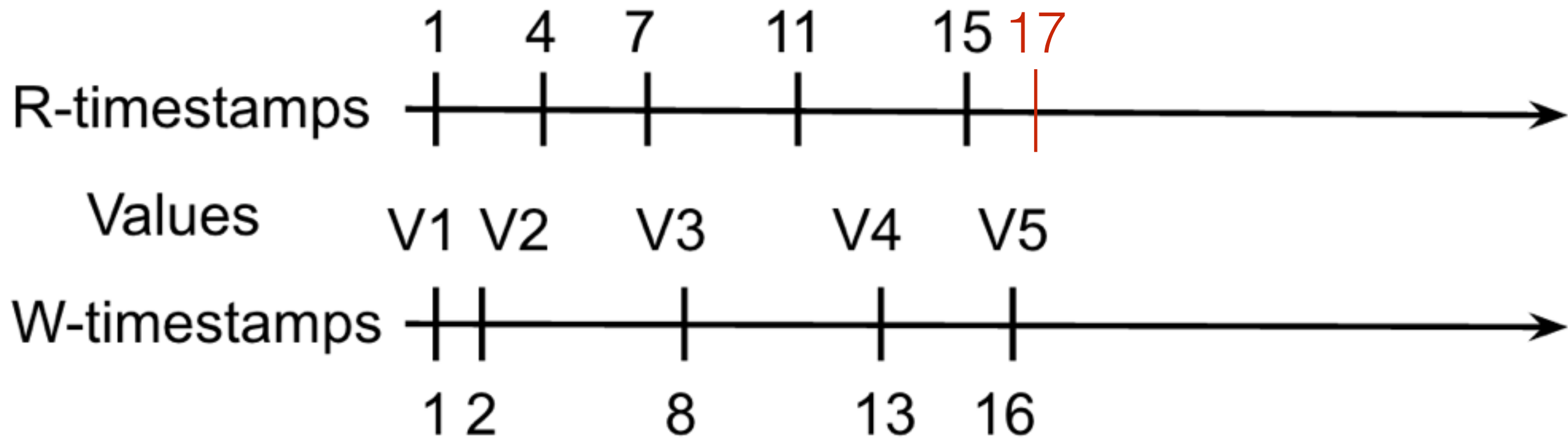
$W(X) @ 8$

Worksheet - MVCC

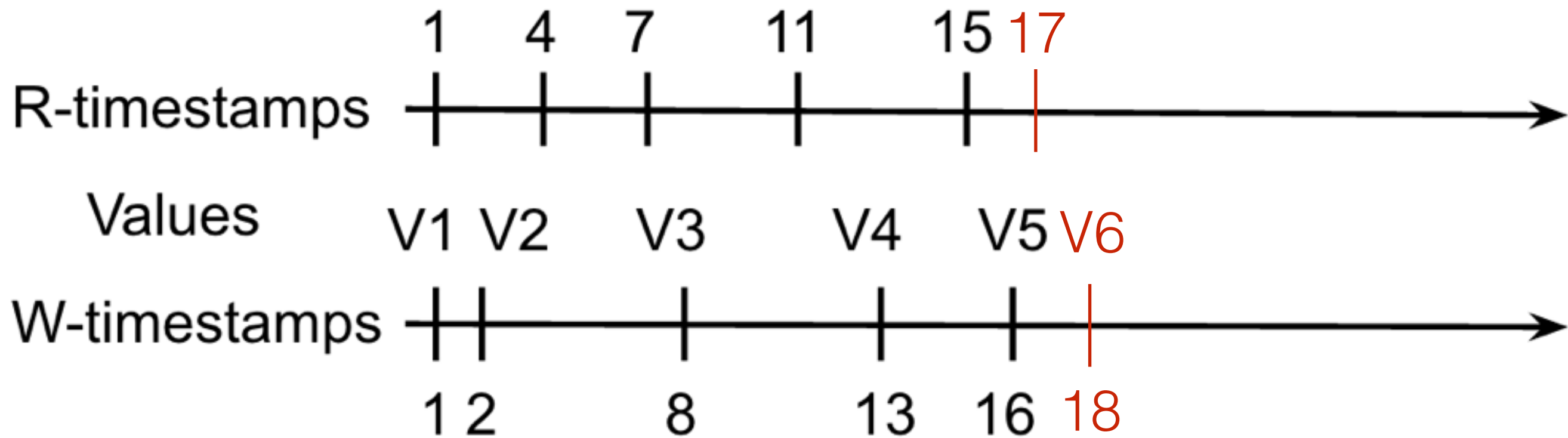
R(X)@17, W(X)@18, W(X)@14, W(X)@12, R(X)@20, R(X)@19,
R(X)@23, W(X)@26, W(X)@24



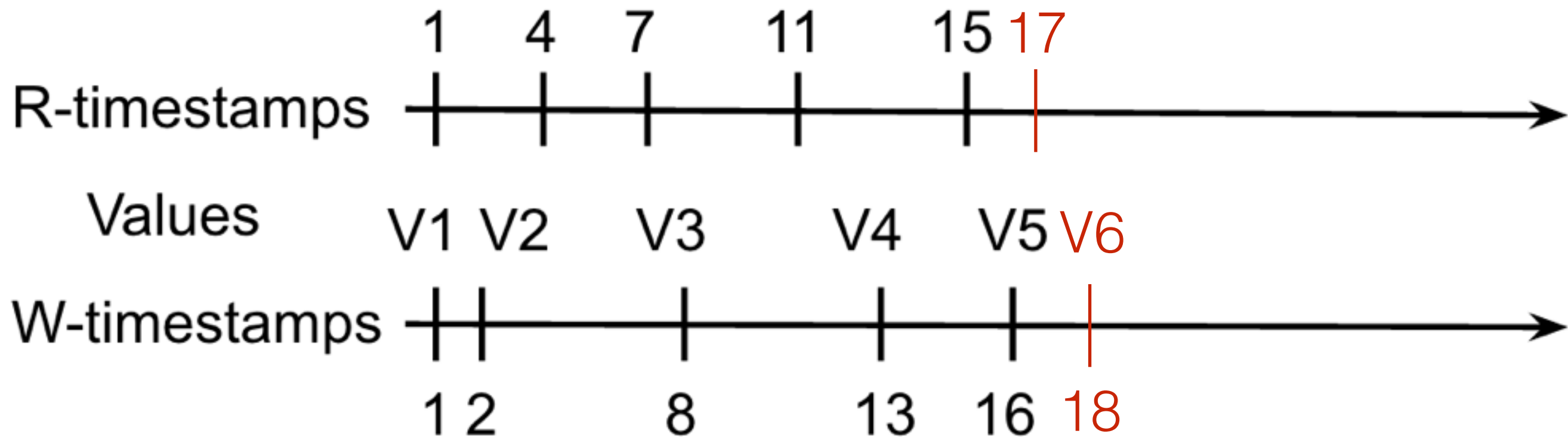
$R(X)@17$, $W(X)@18$, $W(X)@14$, $W(X)@12$, $R(X)@20$, $R(X)@19$,
 $R(X)@23$, $W(X)@26$, $W(X)@24$



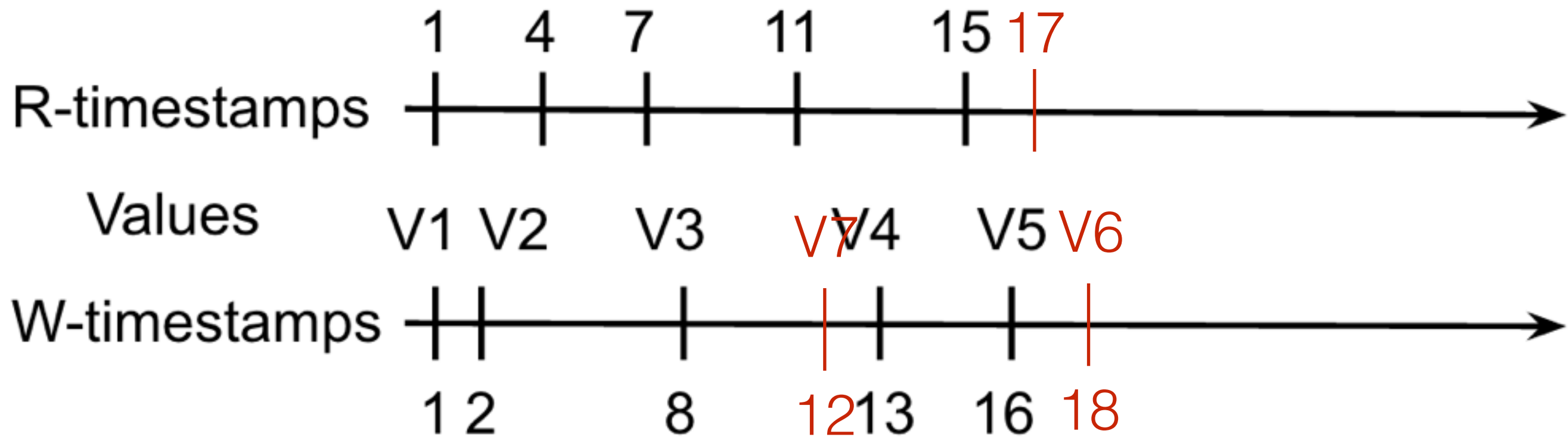
R(X)@17, W(X)@18, W(X)@14, W(X)@12, R(X)@20, R(X)@19,
R(X)@23, W(X)@26, W(X)@24



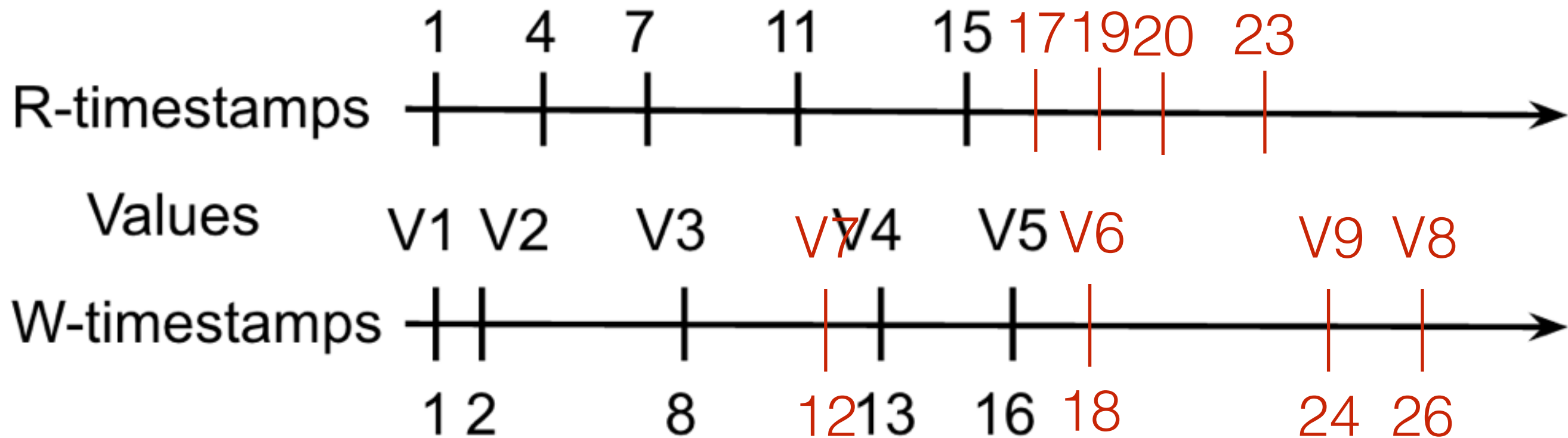
R(X)@17, W(X)@18, ~~W(X)@14~~, W(X)@12, R(X)@20, R(X)@19,
R(X)@23, W(X)@26, W(X)@24



$R(X)@17$, $W(X)@18$, ~~$W(X)@14$~~ , $W(X)@12$, $R(X)@20$, $R(X)@19$,
 $R(X)@23$, $W(X)@26$, $W(X)@24$



$R(X)@17$, $W(X)@18$, ~~$W(X)@14$~~ , $W(X)@12$, $R(X)@20$, $R(X)@19$,
 $R(X)@23$, $W(X)@26$, $W(X)@24$



Weak Isolation

- Transactions too restrictive
 - Allow changes to table while reading table?
- Transactions too expensive
 - Lots of waiting

Isolation Levels

- Read Uncommitted:
 - Can read dirty data
 - No locks on read
- Read Committed:
 - Only read committed data
 - Can unlock immediately after read

Isolation Levels

- Repeatable Read:
 - If value read twice in Xact, should see same committed version
 - Hold read locks until end of Xact
- Serializable

Snapshot Isolation

- Get snapshot of database when Xact starts
- When Xact commits items, those items must not have changed since snapshot
 - If T1 and T2 start with the same snapshot and both modify same tuples, only one can commit.

Worksheet - Weak Isolation

task	assignee
B	Bob
A	Alice
D	Bob

Outcome possible with serializable isolation?

task	assignee
B	Bob
A	Alice
D	Bob

Outcome possible with serializable isolation?

Yes. Serializable schedule is equivalent to serial execution.
Run T2 first, then abort T1 and T3.

task	assignee
B	Bob
A	Alice
D	Bob

Outcome possible with snapshot isolation?

task	assignee
B	Bob
A	Alice
D	Bob

Outcome possible with snapshot isolation?

Yes. Same reason as serializable. T2 can run first, commit, then T1 and T3 abort.

task	assignee
B	Bob
A	Alice
D	Bob
E	Alice

Outcome possible with serializable isolation?

task	assignee
B	Bob
A	Alice
D	Bob
E	Alice

Outcome possible with serializable isolation?

No. No way to run Xacts serially and have two Xacts commit.

task	assignee
B	Bob
A	Alice
D	Bob
E	Alice

Outcome possible with snapshot isolation?

Yes! T2 and T3 get same snapshot of initial db, and insert their tasks. Their modified snapshots appear consistent, so both commit. T1 aborts.

task	assignee
B	Bob
A	Alice

Outcome possible with serializable isolation?

task	assignee
B	Bob
A	Alice

Outcome possible with serializable isolation?

No. No way to run Xacts serially and no Xacts commit.

task	assignee
B	Bob
A	Alice

Outcome possible with read uncommitted isolation?

Yes. All three requests insert tasks at same time, and count number of rows. Sees 5 rows, so aborts.