Strict two-phase locking (2PL)

1. If a Transaction wants to read (modify/write) an “object”, it first obtains a *shared* (*exclusive*) lock on the “object”

2. All locks held by a transaction are released when the transaction is complete (upon Commit)

> A shared lock on an object can be obtained in the absence of an exclusive lock on the object *by another transaction*.
>
> An exclusive lock can be obtained in the absence of any lock by another transaction

Basically, locking is concerned with ensuring atomic and isolation properties of individual transactions, while exploiting parallelism/interleaving.

What “objects” can be locked?

Entire tables

Individual records within a table

A set of records that satisfy a condition (e.g., TransNumber = abc)

An entire indexing structure on an attribute for a table

Individual nodes (index pages) within the indexing structure

Individual data pages

In general, we want exclusive locks on the smallest “objects” possible?

Can individual attribute fields of an individual record be locked?
Check it out….

Which of these are legal schedules (that interleave transactions T1 and T2) under 2PL?

These schedules only show the locks (X: exclusive lock; S: shared lock), reads (R) and writes (W). Between a read and write of an object (resource), there would also be an operation on the object (e.g., update a tuple, an index, etc), which I don't show.

Remember, in 2PL, locks are only released upon commit, but can be changed
(Shared -> Exclusive) during transaction

S4

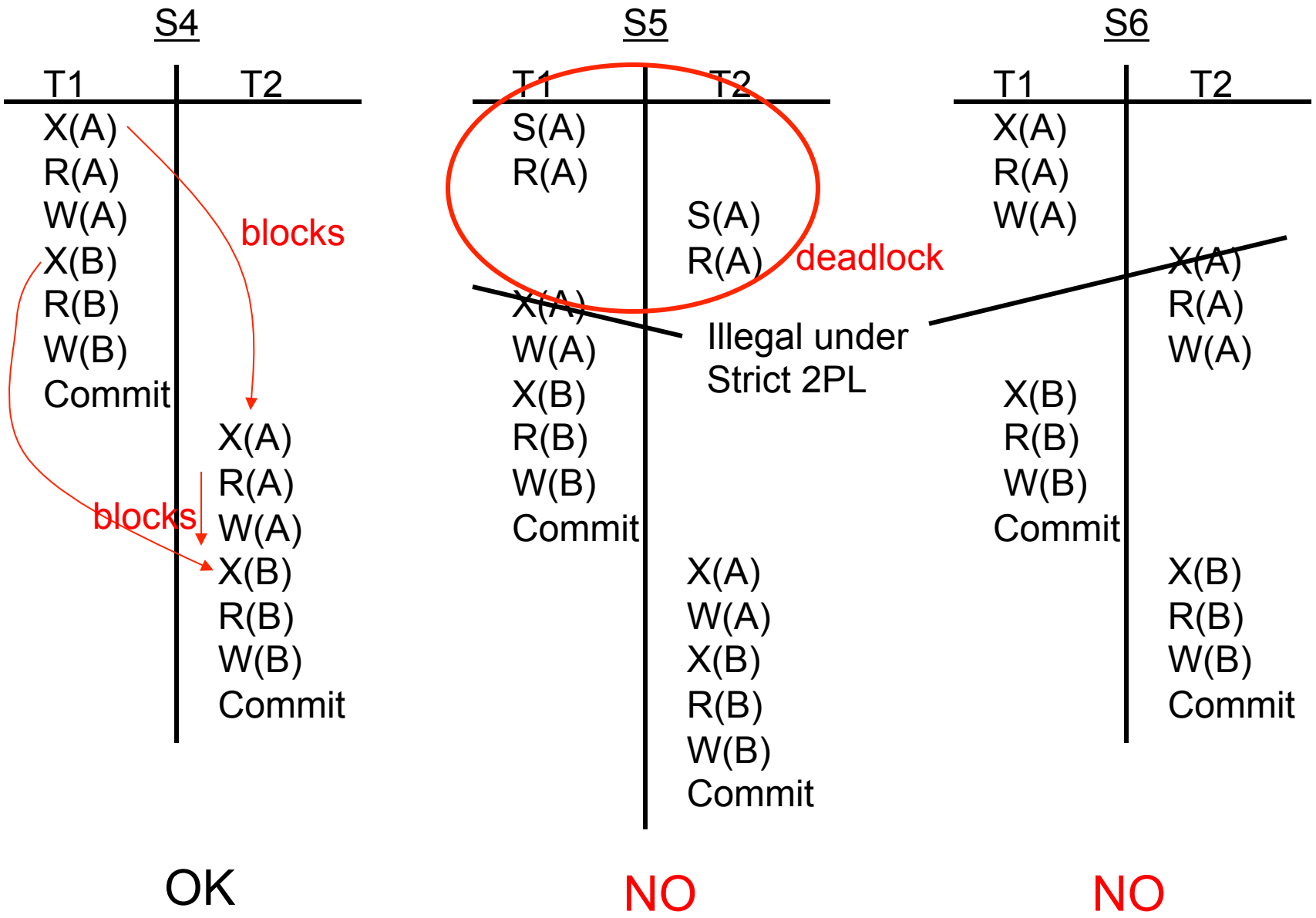
T1	T2
X(A)	
R(A)	
W(A)	
X(B)	
R(B)	
W(B)	
Commit	
	X(A)
	R(A)
	W(A)
	X(B)
	R(B)
	W(B)
	Commit

S5

T1	T2
S(A)	
R(A)	
	S(A)
	R(A)
X(A)	
W(A)	
X(B)	
R(B)	
W(B)	
Commit	
	X(A)
	W(A)
	X(B)
	R(B)
	W(B)
	Commit

S6

T1	T2
X(A)	
R(A)	
W(A)	
	X(A)
	R(A)
	W(A)
X(B)	
R(B)	
W(B)	
Commit	
	X(B)
	R(B)
	W(B)
	Commit



OK NO NO

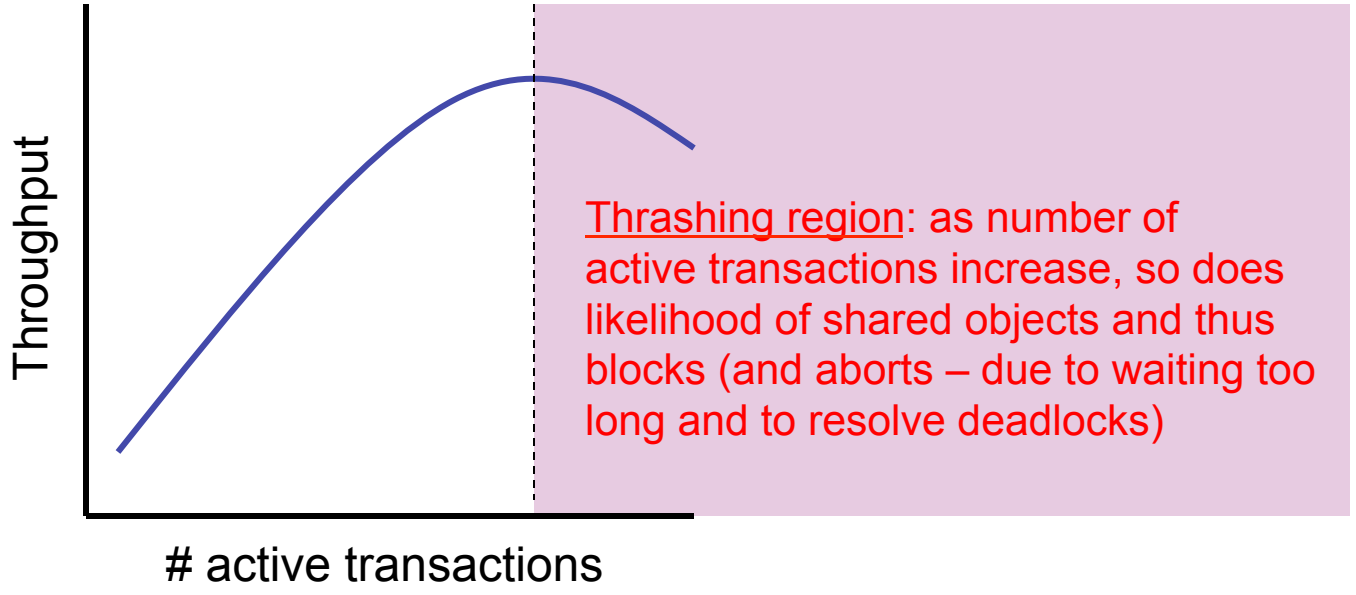
Can you do any interleaving of T1 and T2 under strict 2PL at all?

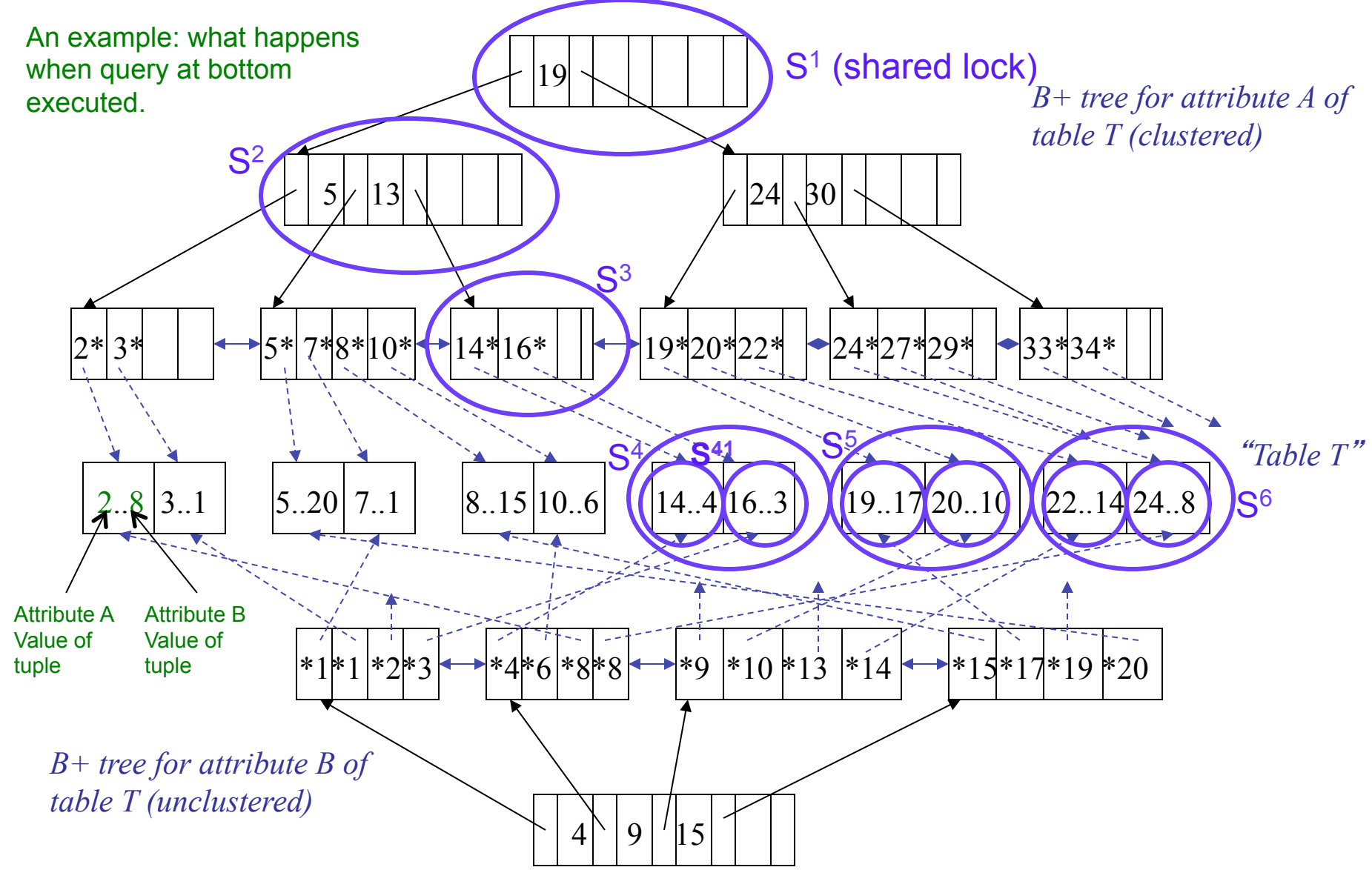
In cases where transactions involve the same objects, Strict 2PL can radically limit opportunities for parallelism/interleaving

…. But Strict 2PL makes interleaving safe, and the good news is that

in practice, there are many transactions that do not involve the same objects and that can be interleaved to improve throughput

and even transactions that share objects (through reads) can be interleaved with strict 2PL (and shared locks)

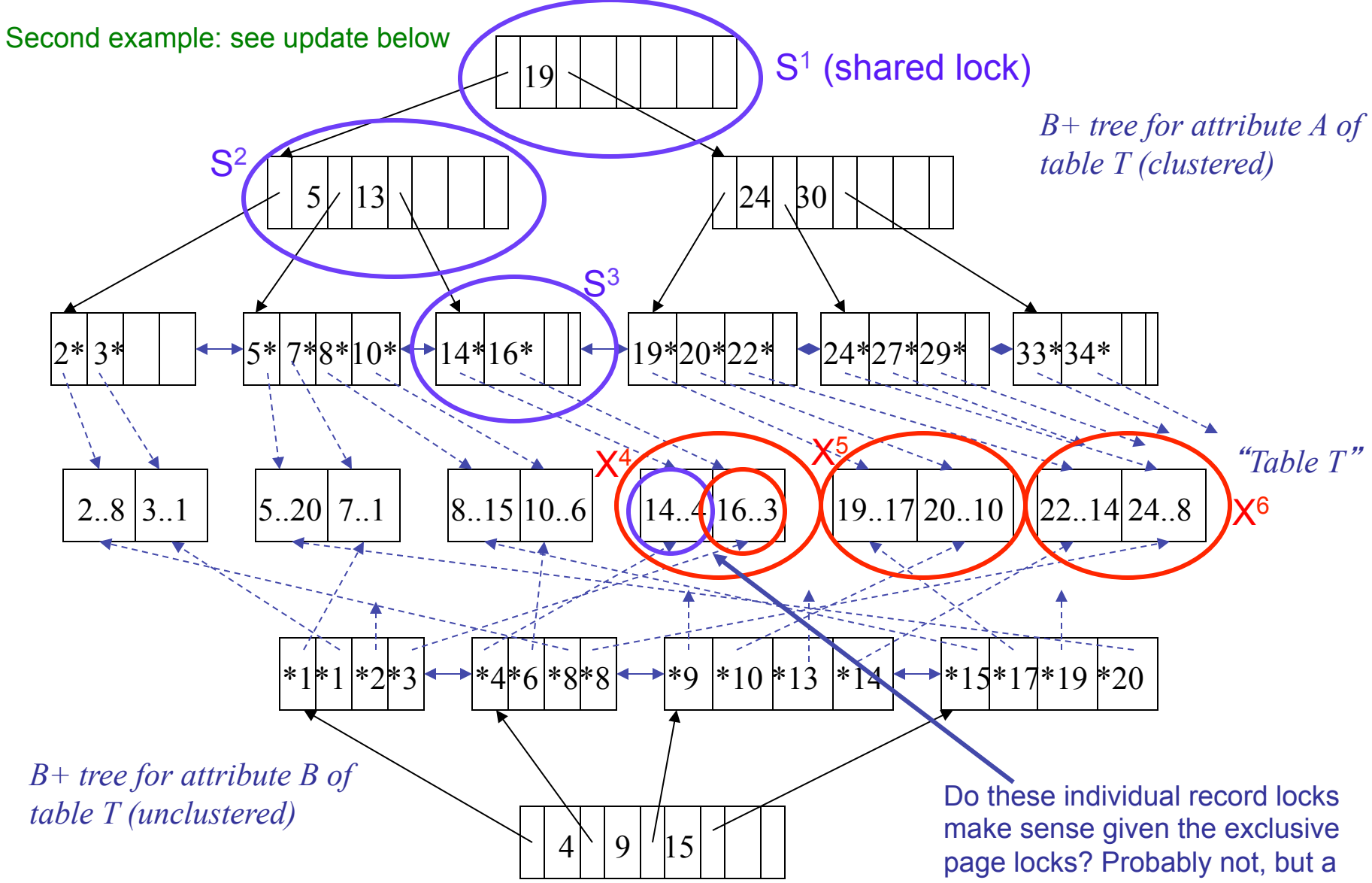




SELECT T.C FROM T WHERE **T.A > 14** AND T.B <= 10

What locks and in what order?

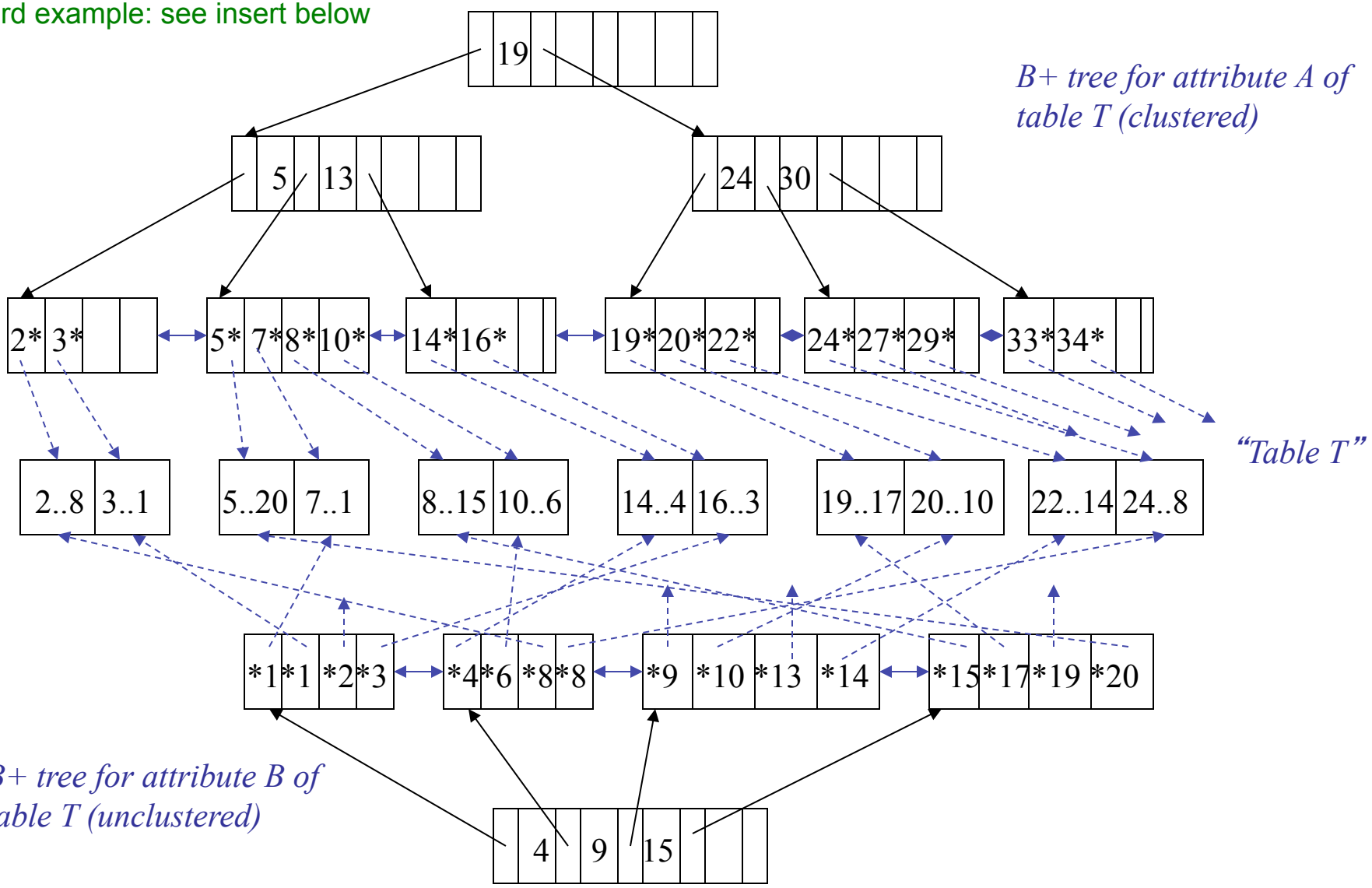
Second example: see update below



Do these individual record locks make sense given the exclusive page locks? Probably not, but a shared lock of the node, and an exclusive lock of a tuple would make sense. The granularity of objects that can be locked will vary with SQL platform

UPDATE T SET T.C = T.C+1 WHERE **T.A > 14** AND T.B <= 10

Third example: see insert below



B+ tree for attribute A of table T (clustered)

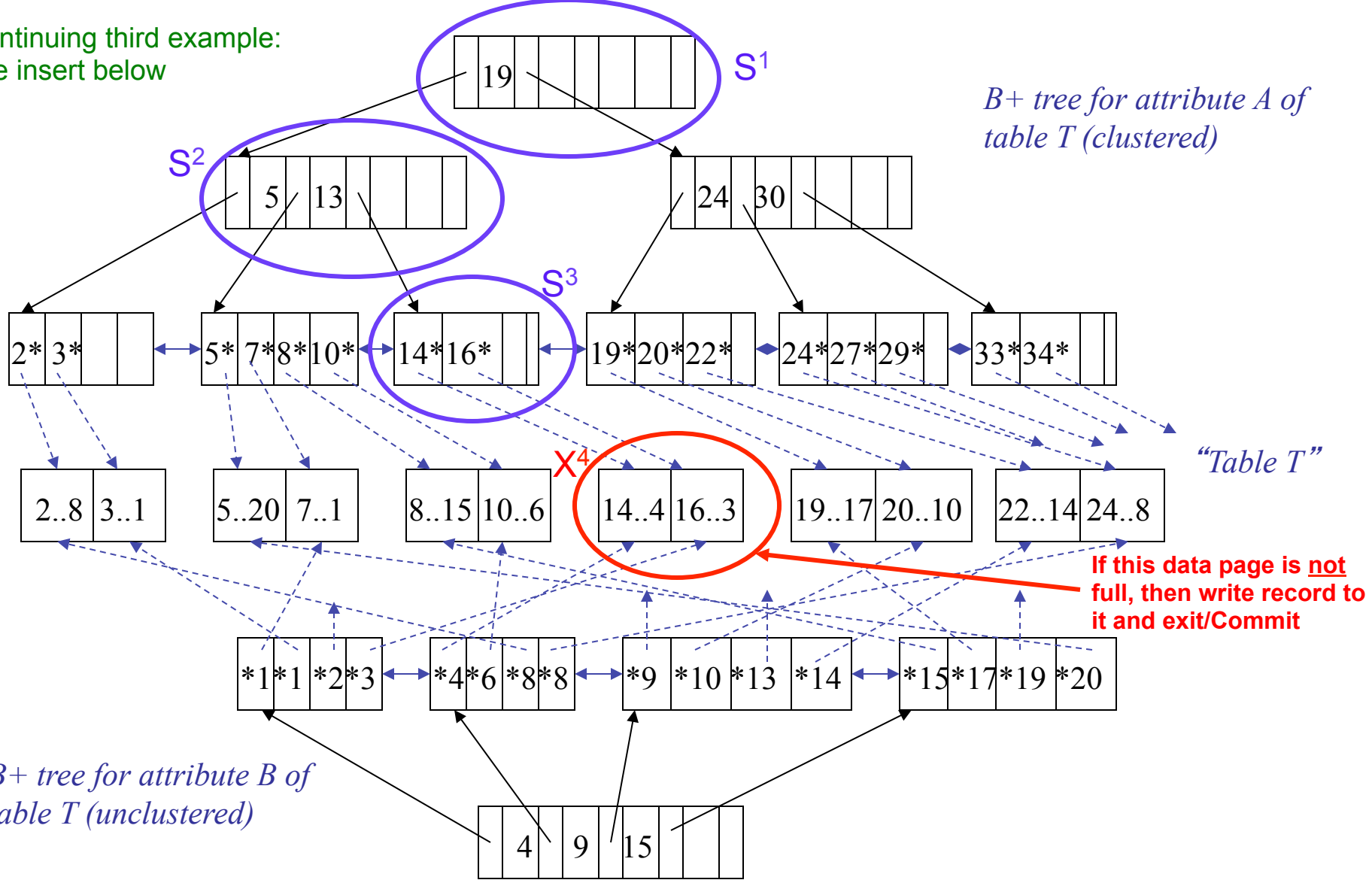
"Table T"

B+ tree for attribute B of table T (unclustered)

INSERT INTO T (A, B, C) VALUES (**18**, 12, …)

What locks and in what order?

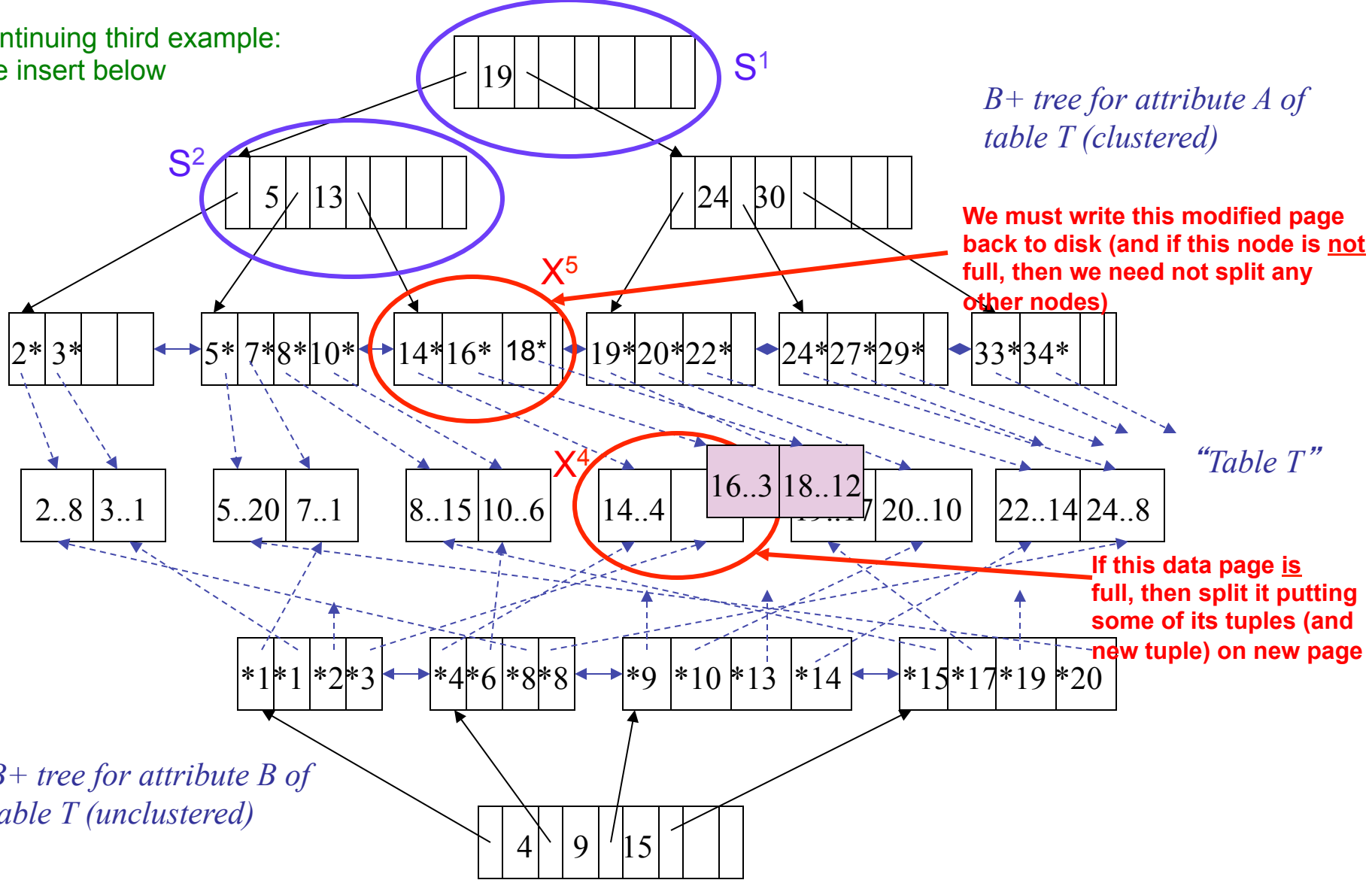
Continuing third example:
see insert below



INSERT INTO T (A, B, C) VALUES (**18**, 12, …)

Continued on next page

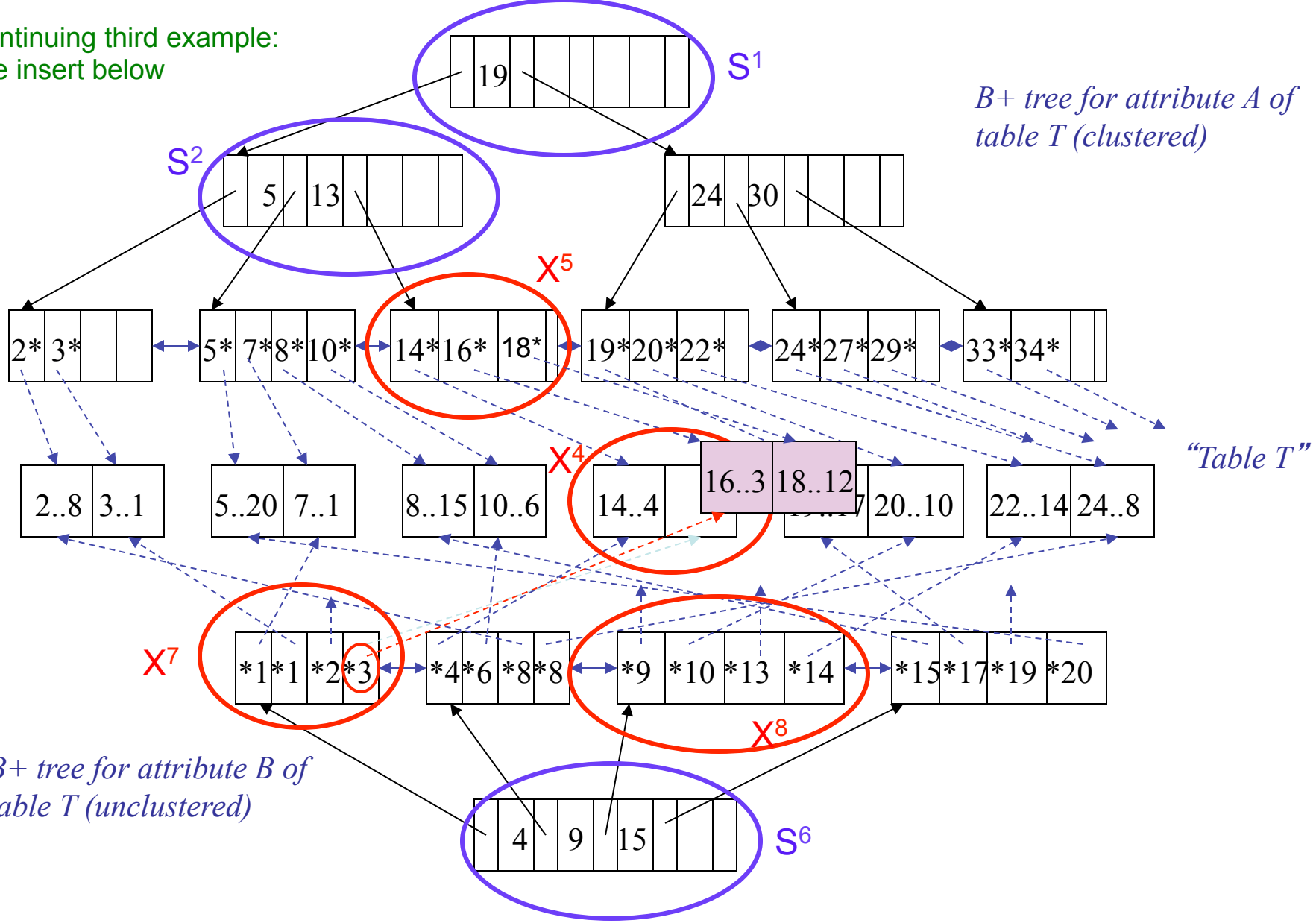
Continuing third example:
see insert below



INSERT INTO T (A, B, C) VALUES (**18**, 12, …)

Continued on next page: must update B index too

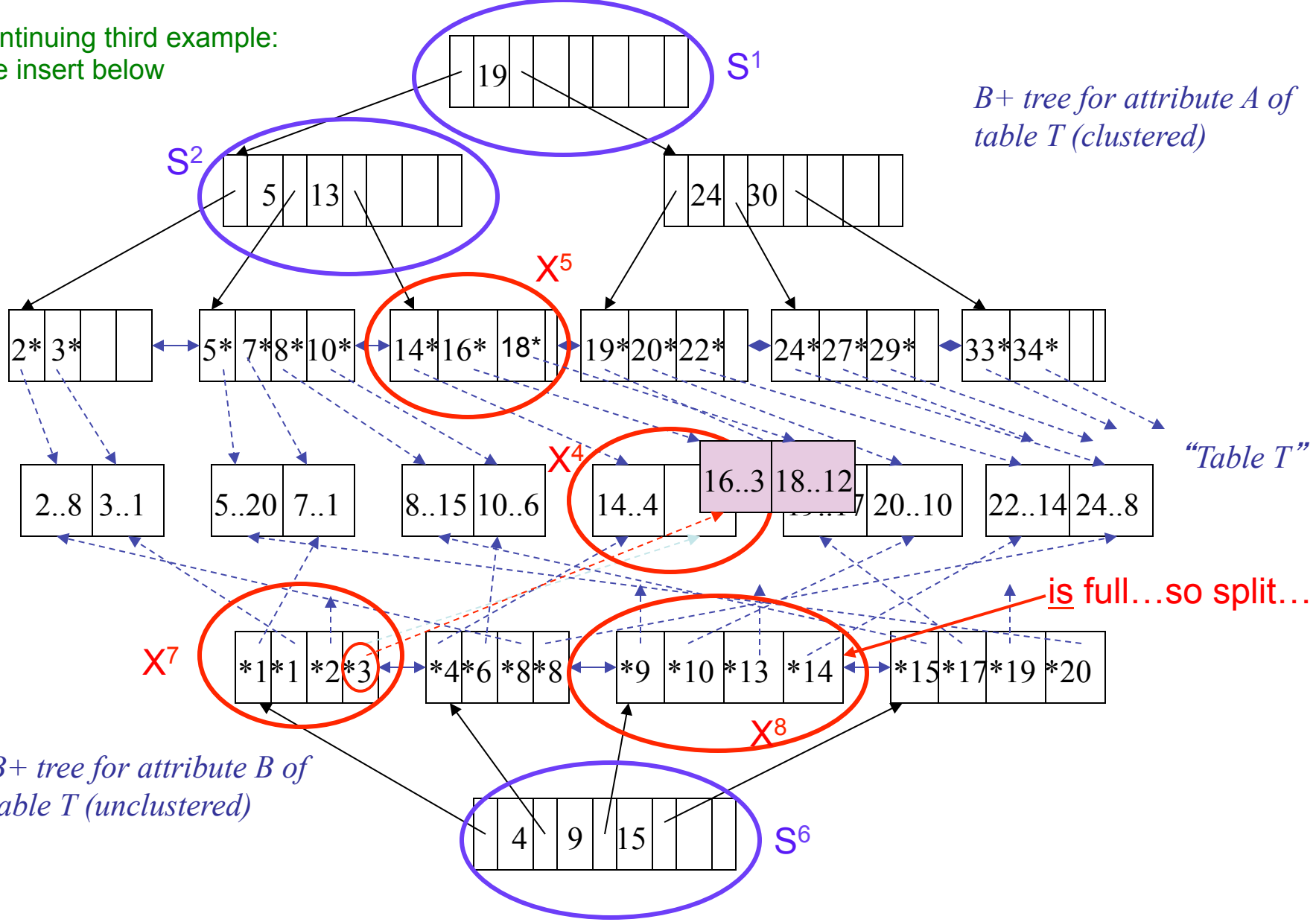
Continuing third example:
see insert below



INSERT INTO T (A, B, C) VALUES (**18**, 12, …)

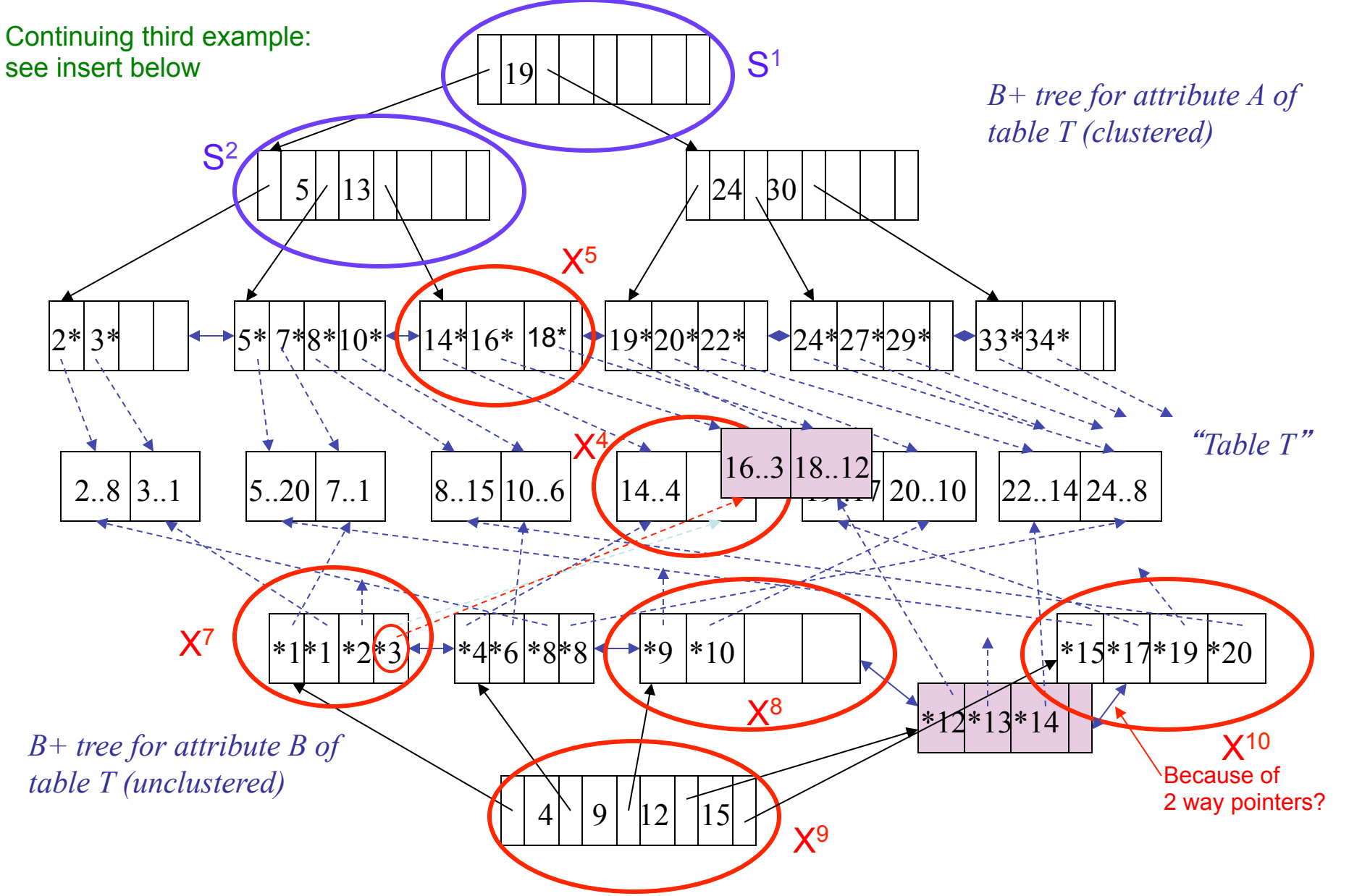
Continued on next page:

Continuing third example:
see insert below



INSERT INTO T (A, B, C) VALUES (**18**, 12, …)

Continued on next page


Continuing third example:
see insert below
19
S1
S2
5
13
24
30
B+ tree for attribute A of
table T (clustered)
X5
2* 3*
5* 7*8*10*
14*16*
18*
19*20*22*
24*27*29*
33*34*
"Table T"
X4
2..8
3..1
5..20
7..1
8..15
10..6
14..4
16..3
18..12
20..10
22..14
24..8
X7
*1*1
*2*3
*4*6
*8*8
*9
*10
X8
*15*17*19
*20
*12*13*14
X10
Because of
2 way pointers?
B+ tree for attribute B of
table T (unclustered)
4
9
12
15
X9
INSERT INTO T (A, B, C) VALUES (18, 12, …)