Figure 21.1 Illustrating cascading rollback (a process that never occurs in strict or cascadeless schedules). (a) The read and write operations of three transactions. (b) System log at point of crash. (c) Operations before the crash.

(a)

T_1	T_2	T_3
read_item(A)	read_item(B)	read_item(C)
read_item(D)	write_item(B)	write_item(B)
write_item(D)	read_item(D)	read_item(A)
	write_item(D)	write_item(A)

(b)

		A	B	C	D
		30	15	40	20
(b)	[start-transaction,T_3]				
	[read_item,T_3,C]				
*	[write_item,T_3,B,15,12]		12		
	[start-transaction,T_2]				
	[read_item,T_2,B]				
**	[write_item,T_2,B,12,18]		18		
	[start-transaction,T_1]				
	[read_item,T_1,A]				
	[read_item,T_1,D]				
	[write_item,T_1,D,20,25]				25
	[read_item,T_2,D]				
**	[write_item,T_2,B,25,26]				26
	[read_item,T_3,A]				
	← system crash				

*T_3 is rolled back because it did not reach its commit point.
**T_2 is rolled back because it reads the value of item B written by T_3.

Figure 21.1 continued. (c) Operations before the crash.

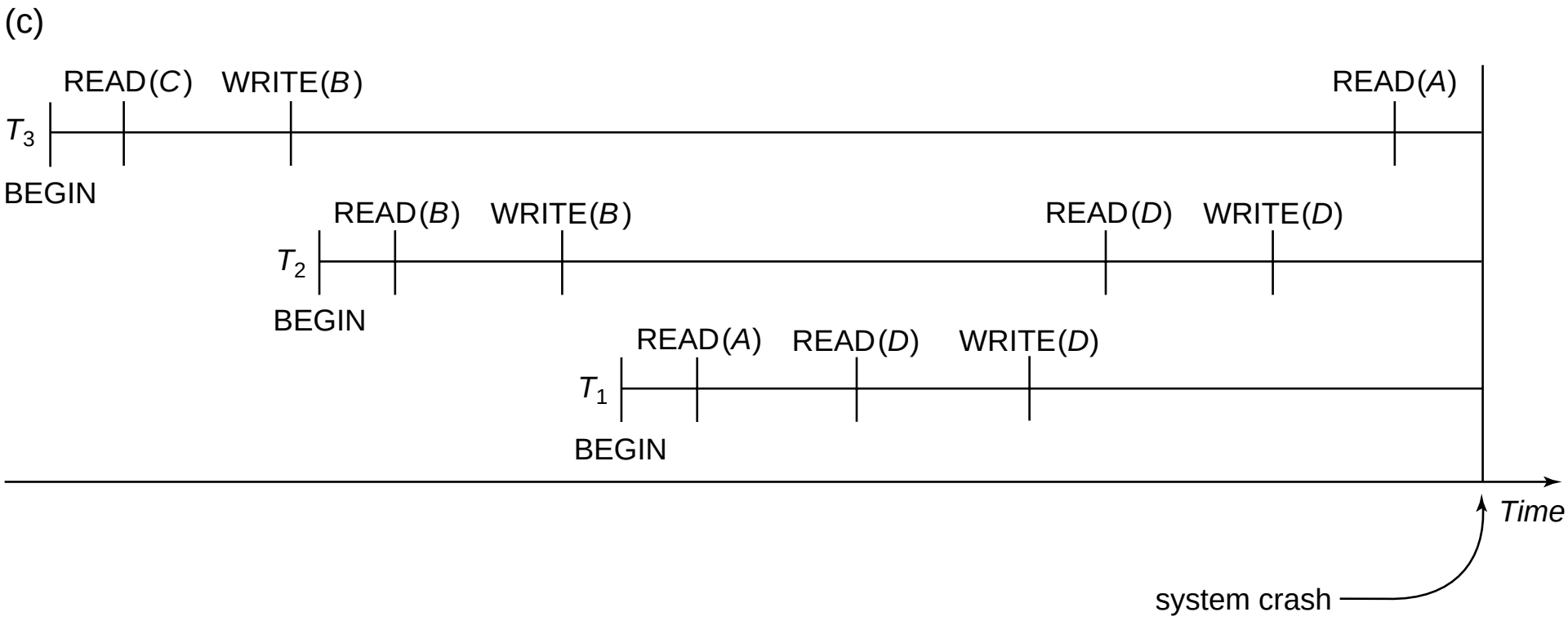

Figure 21.2 An example of recovery using deferred update in a single-user environment. (a) The read and write operations of two transactions. (b) The system log at the point of crash.

(a)

T_1	T_2
read_item(A)	read_item(B)
read_item(D)	write_item(B)
write_item(D)	read_item(D)
	write_item(D)

(b)

[start-transaction,T_1]
[write_item,T_1,D,20]
[commit,T_1]
[start-transaction,T_2]
[write_item,T_2,B,10]
[write_item,T_2,D,25] ← system crash

The [write_item,...] operations of T_1 are redone.
T_2 log entries are ignored by the recovery process.

Figure 21.3 An example of recovery in a multiuser environment.

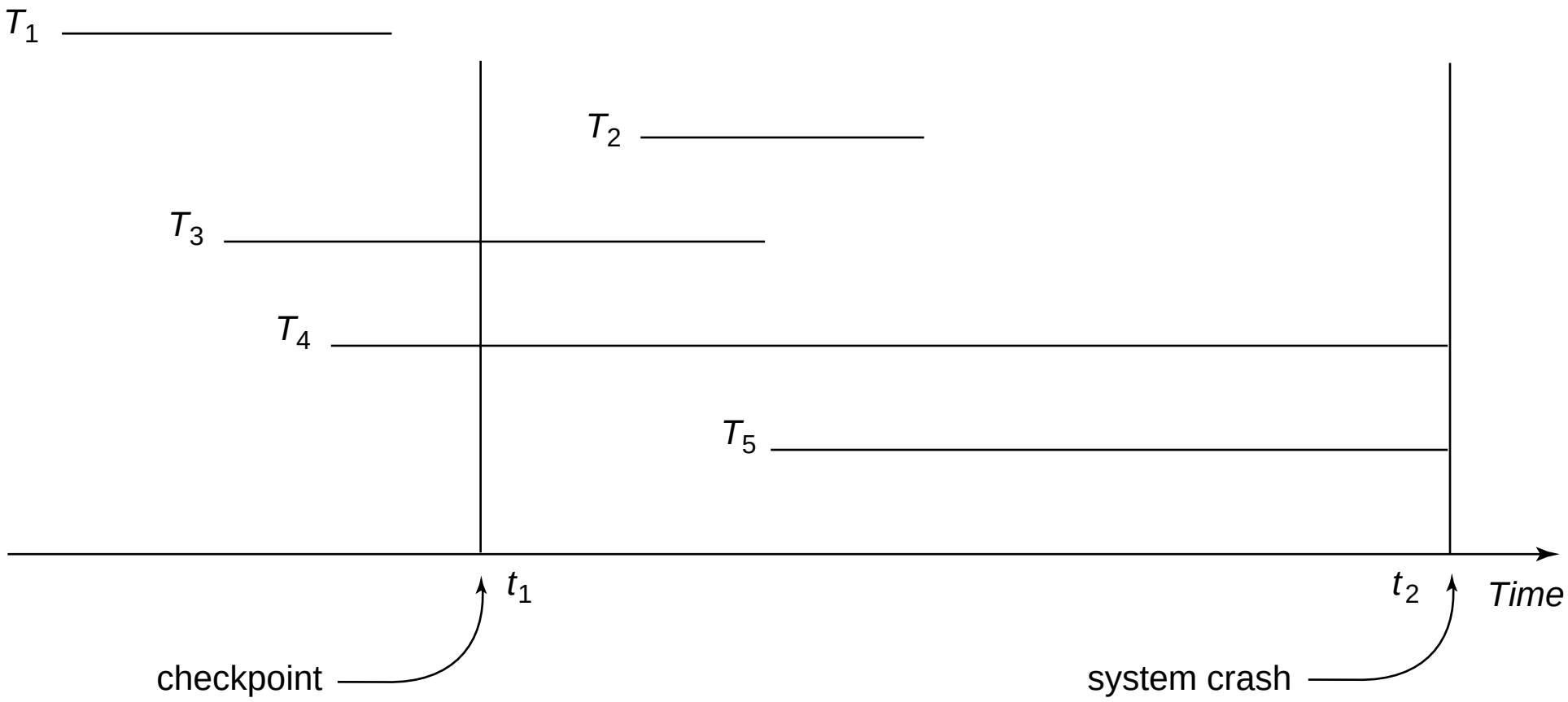

Figure 21.4 An example of recovery using deferred update with concurrent transactions. (a) The read and write operations of four transactions. (b) System log at the point of crash.

(a)

T_1	T_2	T_3	T_4
read_item(A)	read_item(B)	read_item(A)	read_item(B)
read_item(D)	write_item(B)	write_item(A)	write_item(B)
write_item(D)	read_item(D)	read_item(C)	read_item(A)
	write_item(D)	write_item(C)	write_item(A)

(b)

[start_transaction,T_1]
[write_item,T_1,D,20]
[commit,T_1]
[checkpoint]
[start_transaction,T_4]
[write_item,T_4,B,15]
[write_item,T_4,A,20]
[commit,T_4]
[start_transaction,T_2]
[write_item,T_2,B,12]
[start_transaction,T_3]
[write_item,T_3,A,30]
[write_item,T_2,D,25] ← system crash

T_2 and T_3 are ignored because they did not reach their commit points.
T_4 is redone because its commit point is after the last system checkpoint.

Figure 21.5 An example of shadow paging.

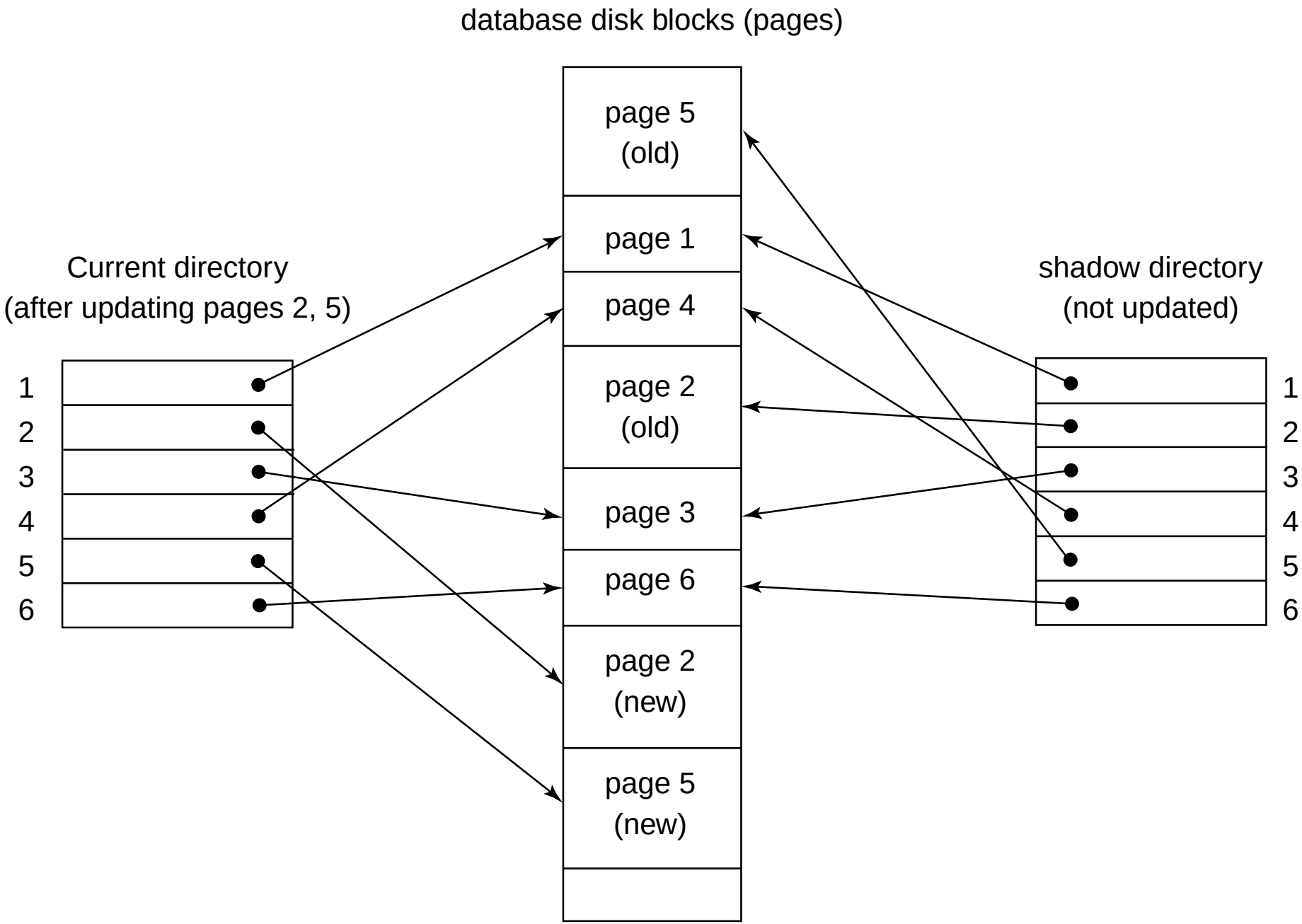

Figure 21.6 An example of recovery in ARIES. (a) The log at point of crash. (b) The Transaction and Dirty Page Tables at time of checkpoint. (c) The Transaction and Dirty Page Tables after the analysis phase.

(a)

LSN	LAST_LSN	TRAN_ID	TYPE	PAGE_ID	OTHER INFORMATION
1	0	T1	update	C	...
2	0	T2	update	B	...
3	1	T1	commit		...
4	begin checkpoint				
5	end checkpoint				
6	0	T3	update	A	...
7	2	T2	update	C	...
8	7	T2	commit		...

(b)

TRANSACTION TABLE

TRANSACTION ID	LAST LSN	STATUS
T1	3	commit
T2	2	in progress

DIRTY PAGE TABLE

PAGE ID	LSN
C	1
B	2

(c)

TRANSACTION TABLE

TRANSACTION ID	LAST LSN	STATUS
T1	3	commit
T2	7	commit
T3	6	in progress

DIRTY PAGE TABLE

PAGE ID	LSN
C	1
B	2
A	6

Figure 21.7 An example schedule and its corresponding log.

[start_transaction,T_1]
[read_item,T_1,A]
[read_item,T_1,D]
[write_item,T_1,D,20]
[commit,T_1]
[checkpoint]
[start_transaction,T_2]
[read_item,T_2,B]
[write_item,T_2,B,12]
[start_transaction,T_4]
[read_item,T_4,D]
[write_item,T_4,D,15]
[start_transaction,T_3]
[write_item,T_3,C,30]
[read_item,T_4,A]
[write_item,T_4,A,20]
[commit,T_4]
[read_item,T_2,D]
[write_item,T_2,D,25] ← system crash