Safe and Unsafe States

At any instant of time, there is a current state consisting of *E*, *A*, *C*, and *R*. A state is said to be safe if it is not deadlocked and there is some scheduling order in which every process can run to completion even if all of them suddenly request their maximum number of resources immediately.

In Fig. 3-9(a) we have a state in which *A* has 3 instances of the resource but may need as many as 9 eventually. *B* currently has 2 and may need 4 altogether, later. Similarly, *C* also has 2 but may need an additional 5. A total of 10 instances of the resource exist, so with 7 resources already allocated, there are 3 still free.

	Has	Max
A	3	9
B	2	4
C	2	7

Free: 3
(a)

	Has	Max
A	3	9
B	4	4
C	2	7

Free: 1
(b)

	Has	Max
A	3	9
B	0	—
C	2	7

Free: 5
(c)

	Has	Max
A	3	9
B	0	—
C	7	7

Free: 0
(d)

	Has	Max
A	3	9
B	0	—
C	0	—

Free: 7
(e)

Here the state shown in the figure is safe.

	Has	Max
A	3	9
B	2	4
C	2	7

Free: 3
(a)

	Has	Max
A	4	9
B	2	4
C	2	7

Free: 2
(b)

	Has	Max
A	4	9
B	4	4
C	2	7

Free: 0
(c)

	Has	Max
A	4	9
B	—	—
C	2	7

Free: 4
(d)

Here this state is unsafe.

The Banker's Algorithm for a Single Resource

What the algorithm does is check to see if granting the request leads to an unsafe state. If it does, the request is denied. If granting the request leads to a safe state, it is carried out.

	Has	Max
A	0	6
B	0	5
C	0	4
D	0	7

Free: 10

(a)

	Has	Max
A	1	6
B	1	5
C	2	4
D	4	7

Free: 2

(b)

	Has	Max
A	1	6
B	2	5
C	2	4
D	4	7

Free: 1

(c)