Banker’s Algorithm

William ZHENG

Example

- **Considering a system with five processes T_0 through T_4 and three resources of type A, B, C.**
- **Resource type A has 10 instances, B has 5 instances and type C has 7 instances.**
- **Suppose at time t_0 following snapshot of the system has been taken**

Example of Banker's Algorithm

- 5 threads T_0 through T_4;

  3 resource types:

  A (10 instances), *B* (5 instances), and *C* (7 instances)

- Snapshot at time T_0:

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Need Matrix

- The content of the matrix ***Need*** is defined to be ***Max – Allocation***

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

Safety Algo - 2

- For i=1, $Need_{\mathbf{1}}$ = (1,2,2)
- Finish[1] = false && $Need_{\mathbf{1}}$ (1,2,2) ≤Work (3,3,2)
- So $T_{\mathbf{1}}$ is kept in safe sequence

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

Safety Algo - 3

- Work = Work + Allocation$_1$
- Work = (3,3,2) + (2,0,0)=(5,3,2)
- Finish[5] = {false, true, false, false, false}

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 4

- For i=2, $Need_2$ = (6,0,0)
- Finish[2] = false && $Need_2$ (6,0,0) > Work (5,3,2)
- So T_2 must wait

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 5

- For i=3, $Need_{\mathbf{3}}$ = (0,1,1)
- Finish[3] = false && $Need_{\mathbf{3}}$ (0,1,1) ≤ Work (5,3,2)
- So $T_{\mathbf{3}}$ is kept in safe sequence

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 6

- Work = Work + Allocation_3
- Work = (5,3,2) + (2,1,1)=(7,4,3)
- Finish[5] = {false, true, false, true, false}

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 7

- For i=4, $\mathbf{Need_3}$ = (4,3,1)
- Finish[4] = false && $\mathbf{Need_4}$ (4,3,1) ≤ Work (7,4,3)
- So T_4 is kept in safe sequence

	Need		
	A	*B*	*C*
T_0	7	4	3
T_1	1	2	2
T_2	6	0	0
T_3	0	1	1
T_4	4	3	1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 8

- Work = Work + Allocation$_4$
- Work = (7,4,3) + (0,0,2)=(7,4,5)
- Finish[5] = {false, true, false, true, true}

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 9

- For i=0, $Need_0$ = (7,4,3)
- Finish[0] = false && $Need_0$ (7,4,3) ≤ Work (7,4,5)
- So T_0 is kept in safe sequence

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 10

- Work = Work + Allocation_0
- Work = (7,4,5) + (0,1,0)=(7,5,5)
- Finish[5] = {true, true, false, true, true}

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 11

- For i=2, $Need_2$ = (6,0,0)
- Finish[2] = false && $Need_2$ (6,0,0) ≤ Work (7,5,5)
- So T_2 is kept in safe sequence

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safety Algo - 12

- Work = Work + Allocation_2
- Work = (7,5,5) + (3,0,2)=(10,5,7)
- Finish[5] = {true, true, true, true, true}

	Need
	A B C
T_0	7 4 3
T_1	1 2 2
T_2	6 0 0
T_3	0 1 1
T_4	4 3 1

	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 5 3	3 3 2
T_1	2 0 0	3 2 2	
T_2	3 0 2	9 0 2	
T_3	2 1 1	2 2 2	
T_4	0 0 2	4 3 3	

Safe Sequence

- The system is in a safe state since the sequence < T_1, T_3, T_4, T_2, T_0> satisfies safety criteria

T_1 Request (1,0,2)

T_1 checks
Step1: $Request_1 \leq Need_1$ i.e.,$(1,0,2) \leq (1,2,2)$ and
Step2: $Request_1 \leq Available$ i.e.,$(1,0,2) \leq (3,3,2)$

According to Step3 in Resource-Request Algorithm for Process T_i
$Available = Available - Request_i$;
$Allocation_i = Allocation_i + Request_i$;
$Need_i = Need_i - Request_i$;

We have
$(3,3,2) - (1,0,2) = (2,3,0)$ Available
$(2,0,0) + (1,0,2) = (3,0,2)$ Allocation
$(1,2,2) - (1,0,2) = (0,2,0)$ Need

T_1 Request (1,0,2)

- Determine whether this new state is safe. Execute Safety Algo

Safety Algo - 1

- m=3, n=5
- Work = Available = (2,3,0)
- Finish[5] = {false, false, false, false, false}

- For i=0, $Need_0$ = (7,4,3)
- Finish[0] = false && $Need_0$ (7,4,3) > Work (2,3,0)
- So T_0 has to wait

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 2

- For i=1, $Need_1$ = (0,2,0)
- Finish[1] = false && $Need_1$ (0,2,0) ≤ Work (2,3,0)
- So T_1 is kept in safe sequence

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 3

- Work = Work + Allocation_1
- Work = (2,3,0) + (3,0,2)=(5,3,2)
- Finish[5] = {false, true, false, false, false}

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 4

- For i=2, $Need_2$ = (6,0,0)
- Finish[2] = false && $Need_2$ (6,0,0) > Work (5,3,2)
- So T_2 must wait

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 5

- For i=3, $Need_{\mathbf{3}}$ = (0,1,1)
- Finish[3] = false && $Need_{\mathbf{3}}$ (0,1,1) ≤ Work (5,3,2)
- So $T_{\mathbf{3}}$ is kept in safe sequence

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 6

- Work = Work + Allocation$_3$
- Work = (5,3,2) + (2,1,1)=(7,4,3)
- Finish[5] = {false, true, false, true, false}

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 7

- For i=4, $Need_{\mathbf{4}}$ = (4,3,1)
- Finish[4] = false && $Need_{\mathbf{4}}$ (4,3,1) ≤ Work (7,4,3)
- So $T_{\mathbf{4}}$ is kept in safe sequence

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 8

- Work = Work + Allocation$_4$
- Work = (7,4,3) + (0,0,2)=(7,4,5)
- Finish[5] = {false, true, false, true, true}

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 9

- For i=0, $\mathbf{Need_0}$ = (7,4,3)
- Finish[0] = false && $\mathbf{Need_0}$ (7,4,3) ≤ Work (7,4,5)
- So $\mathbf{T_0}$ is kept in safe sequence

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 10

- Work = Work + Allocation_0
- Work = (7,4,5) + (0,1,0)=(7,5,5)
- Finish[5] = {true, true, false, true, true}

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 11

- For i=2, $Need_2$ = (6,0,0)
- Finish[2] = false && $Need_2$ (6,0,0) ≤ Work (7,5,5)
- So T_2 is kept in safe sequence

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safety Algo - 12

- Work = Work + $\text{Allocation}_{\mathbf{2}}$
- Work = (7,5,5) + (3,0,2)=(10,5,7)
- Finish[5] = {true, true, true, true, true}

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
T_0	0 1 0	7 4 3	2 3 0
T_1	3 0 2	0 2 0	
T_2	3 0 2	6 0 0	
T_3	2 1 1	0 1 1	
T_4	0 0 2	4 3 1	

Safe Sequence

- The system is in a safe state since the sequence $<T_1, T_3, T_4, T_0, T_2>$ satisfies safety criteria