- No circular wait (ordered)
- Do not hold a wait
- Always preempt
- No mutex

Preemption - 4 methods

Wait

Deadlocks
All sizes stay here.

Unsafe steps.

Descend.

Avoidance (described)
Singh et al. — money

Singh version

Singh version

Banksers Achievement (Singhred)
If yes, determine safety

- if yes, move next

Role if yes, < money left

L. Customer C. amount $x

Is safe

Assume 2. Customers have borrowed a stake
Safety →
- pretend to allocate $x$ to $C_i$

→ check if money left is enough to satisfy 1 person
- pretend to give & take back all

- repeat

(all satisfied) → safe

 unsafe
Need

Alloc

\[ \text{vector of current balance} \]
\[ \text{vector of credit limits} \]
\[ \text{in bank} \]

Available = \$ in bank @ the moment

Work \[ [n] \text{ fin}[n] \]

4 temporary
Algorithm 4.31

1. \( \text{Assign } \text{F} \text{inish} = \text{T} \rightarrow \text{yes } \equiv \text{Safe} \)
2. \( \forall x \in \text{work} \quad \text{ alloc} [x] = \text{true} \)
3. \( \text{work} = \text{work} + \text{alloc} [x] \)
4. \( \text{find } \neg \text{ succ } + \text{pre } (f) \quad \text{ finishes } = \text{T} \)
5. \( \text{work } = \text{work} + f_{\text{in}} \quad \text{ alloc } + \text{ Free} \)
6. \( \forall x \in \text{work} \quad \text{ alloc [x] = false } \)
7. \( \text{ procedure } + \text{not } x \}
8. \text{ alloc } = \text{ alloc } - x \)
9. \( \text{ alloc } [x] = \text{true } \)
10. \( \text{ alloc } = \\text{ alloc } + x \)
pending & sho can be satisfy & as for

a check of those is above in

when someone receives money

---

pending in & (age + 12.5 = x)

x $ \times \text{ unsaves } = \text{ more interest for } x$

saves \text{ the customer }$
Prev, Avoidance, Detection

Allow deadlock to happen.

When it happens, detect & recover.
For each energy level, check the state of each wave function. Check for overlap. Check for decoupling. Check for resonance. Check if precise definition is needed.
Fed decision

Check to see how

Fig. - Check new measurement
Do more practice, no deadlock

If a firm/M
does not finish

REW

They're dead

If you finish

In address to updating process

Address of P

Doe's an incoming

Outgoing edge is not finished

Fine, I proceed with

Copy of the bag
P, P2, P3 are deadlocked.

How to recover?

- Typically one or more processes may terminate and the remaining processes release all resources. Rollback changes. Restart the process. Command 'pkill'.
We take \( P_1 \rightarrow P_2 \rightarrow P_3 \) and do some random etc
Also attention -

whether we can or cannot -

or never -

or almost and -

or of -

or P

or P

last one to work P

which before?