

Introduction to ACPI Based Memory Hot-Plug

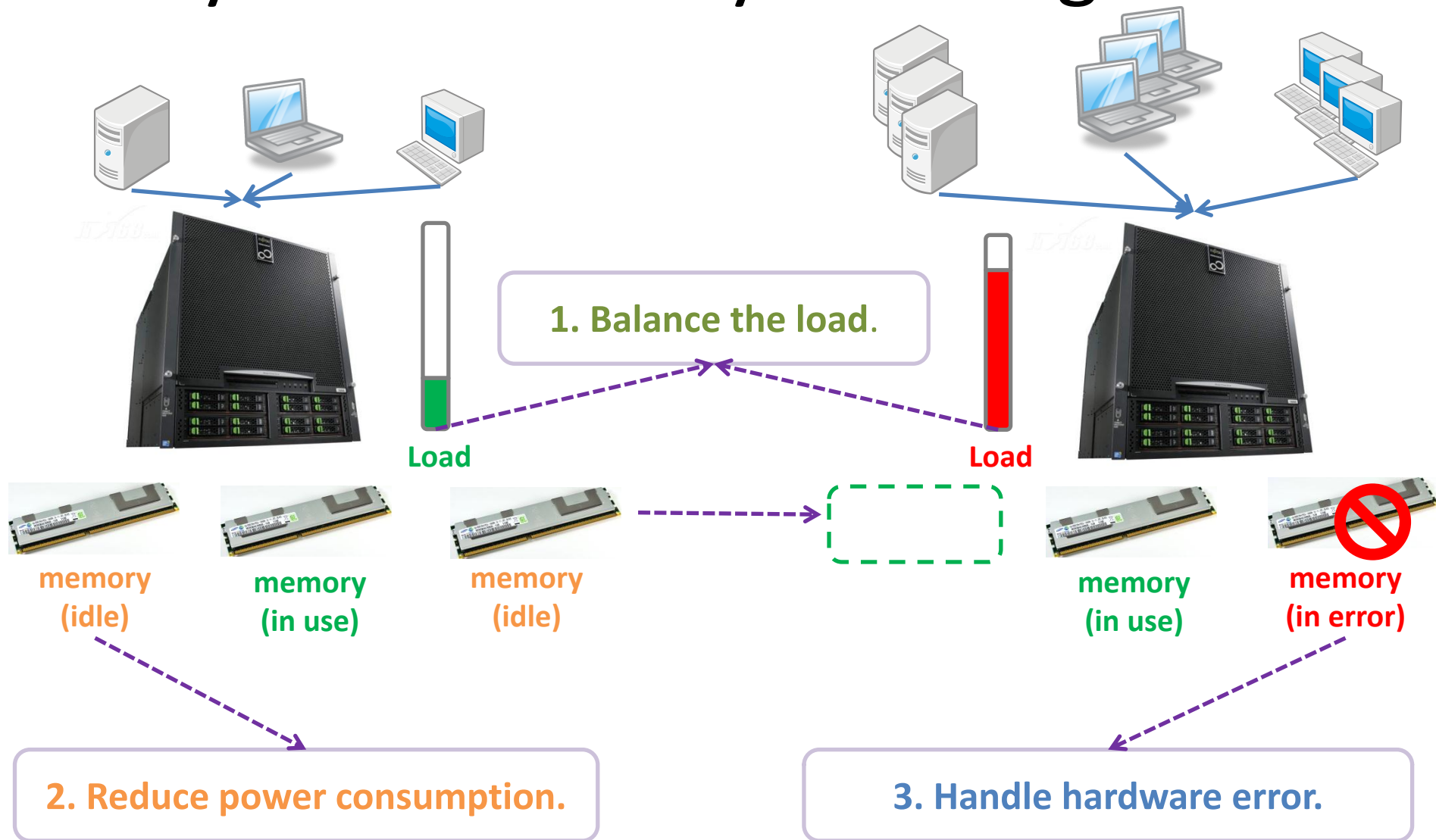
Tang Chen <tangchen@cn.fujitsu.com>

Zhang Yanfei <zhangyanfei@cn.fujitsu.com>

Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

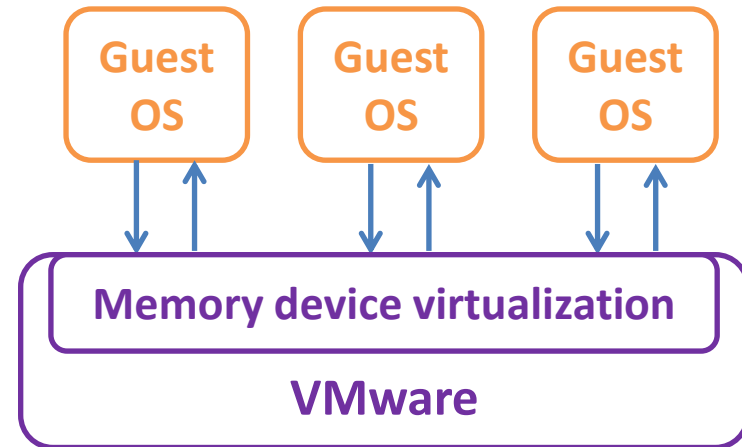
Why need memory Hot-Plug



Why need memory Hot-Plug

4. Guest OS should support memory hotplug.

- VMware supports virtual memory device hotplug for the virtual machine.
- The similar feature is being developed for KVM.



Firmware (ACPI BIOS)

+

Hardware (ACPI registers)

5. ACPI provides sufficient conditions

- With the help of ACPI, hardware and firmware are now able to support memory hotplug physically.

Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

ACPI & Memory Hot-Plug

- ACPI: **A**dvanced **C**onfiguration and **P**ower **I**nterface

ACPI is an interface specification of Operating System-directed motherboard device configuration and Power Management.

-- ACPI Specification 5.0

OS layer framework.

- ✓ Event handling
- ✓ API

Dynamic methods used at run time.

- ✓ `_EJ0`
- ✓ `_STA`
- ✓

SCI

(System Control Interrupt)

Kernel (Software)

ACPI Driver

Run Time

Boot Time

**Methods
(dynamic)**

**ACPI Tables
(static)**

ACPI BIOS (Firmware)

Static info used only at boot time.

- ✓ DSDT
- ✓ SRAT
- ✓

Event driven model.

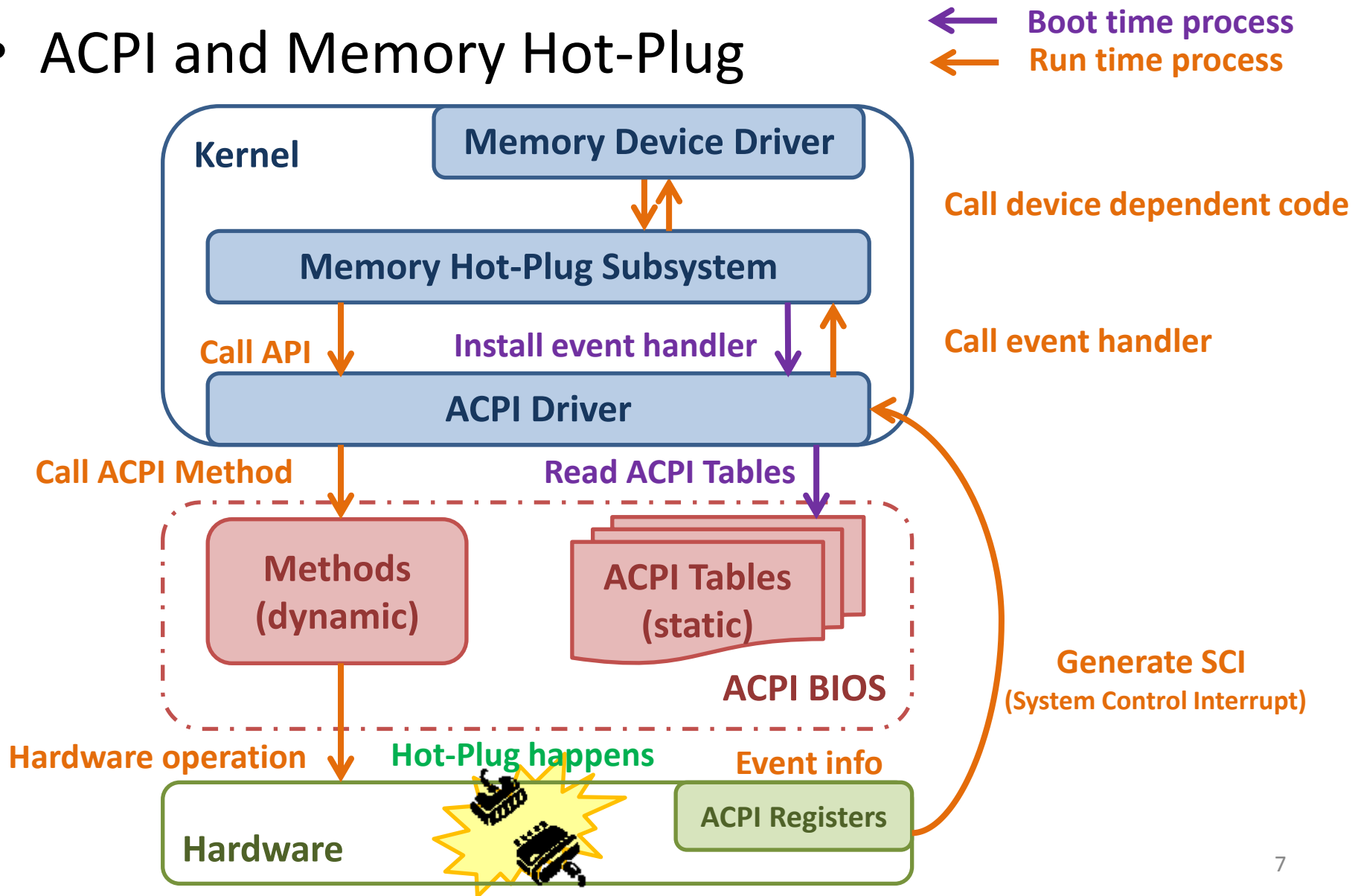
- ✓ Event registers
- ✓ Control registers
- ✓

ACPI Registers

Hardware

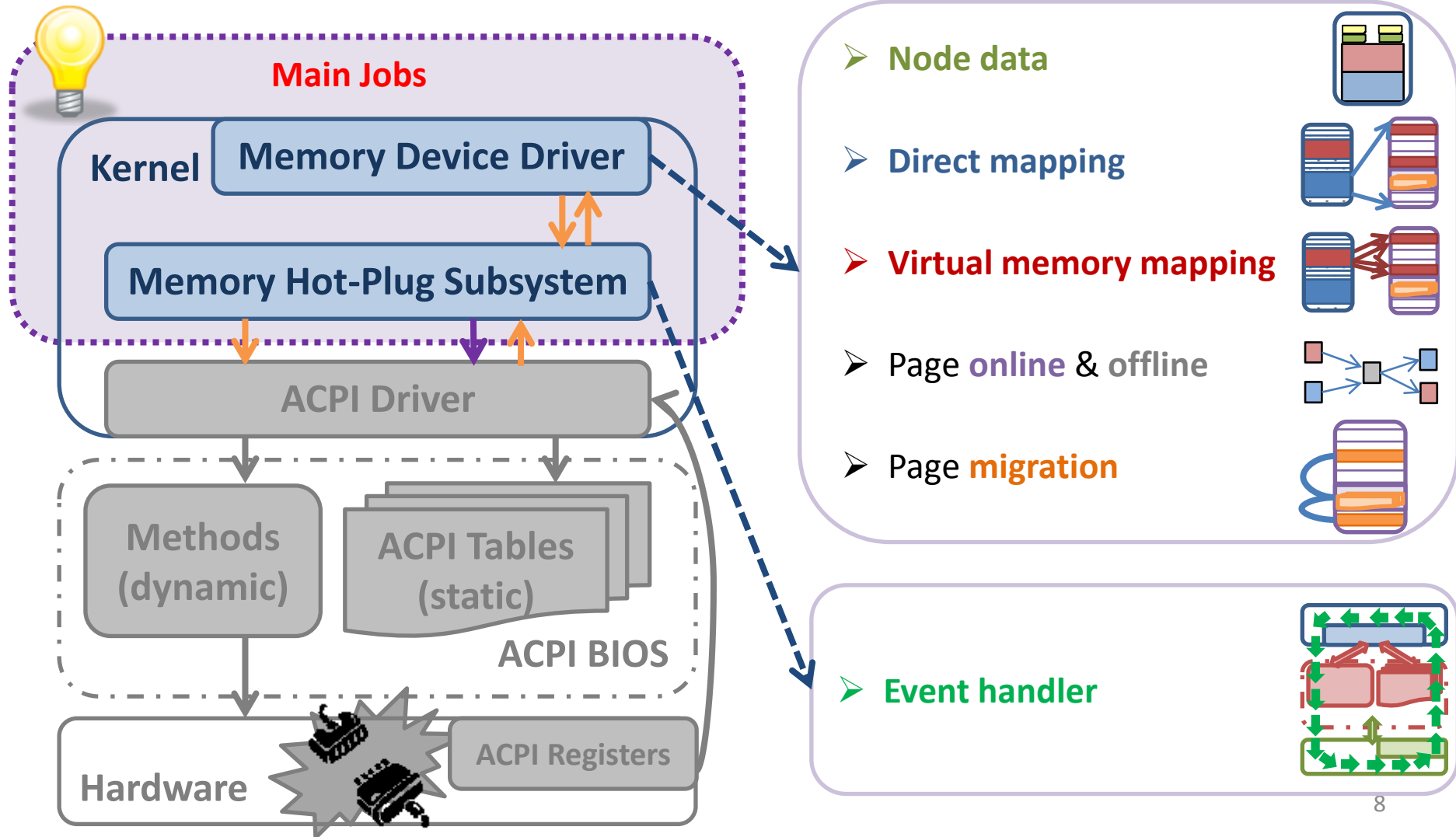
ACPI & Memory Hot-Plug

- ACPI and Memory Hot-Plug



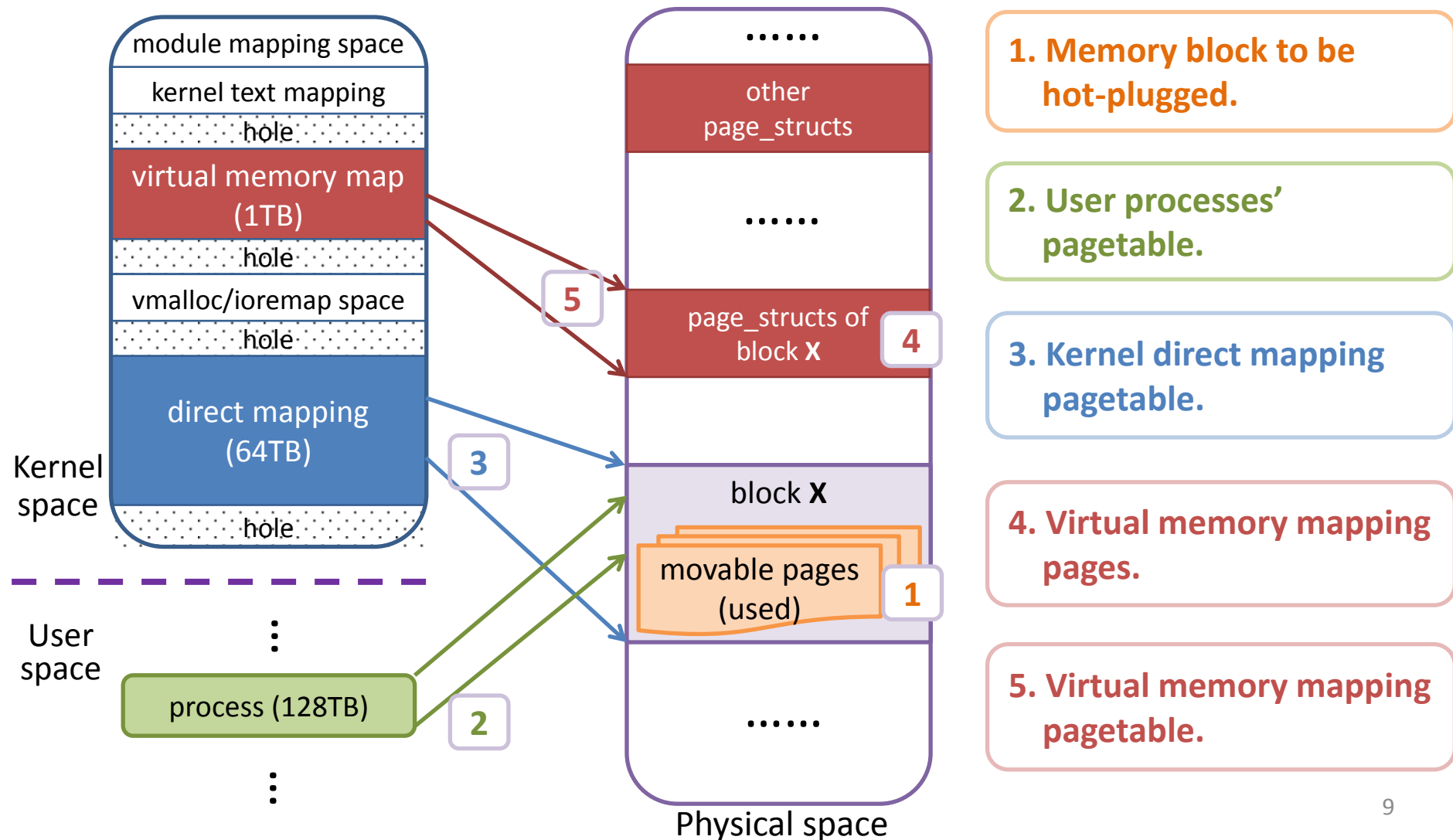
ACPI & Memory Hot-Plug

- Main jobs of Memory Hot-Plug



ACPI & Memory Hot-Plug

- Things associated with Memory Hot-Plug



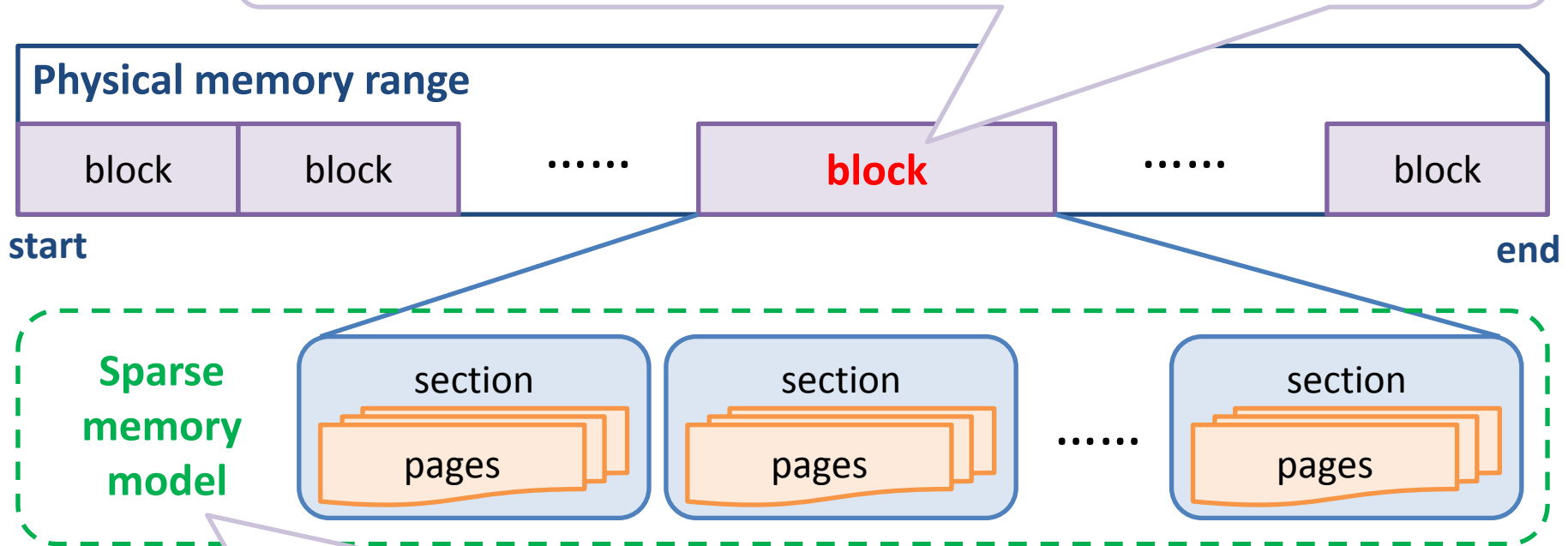
Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

Memory hot-add

- Generic memory definition & sparse memory model

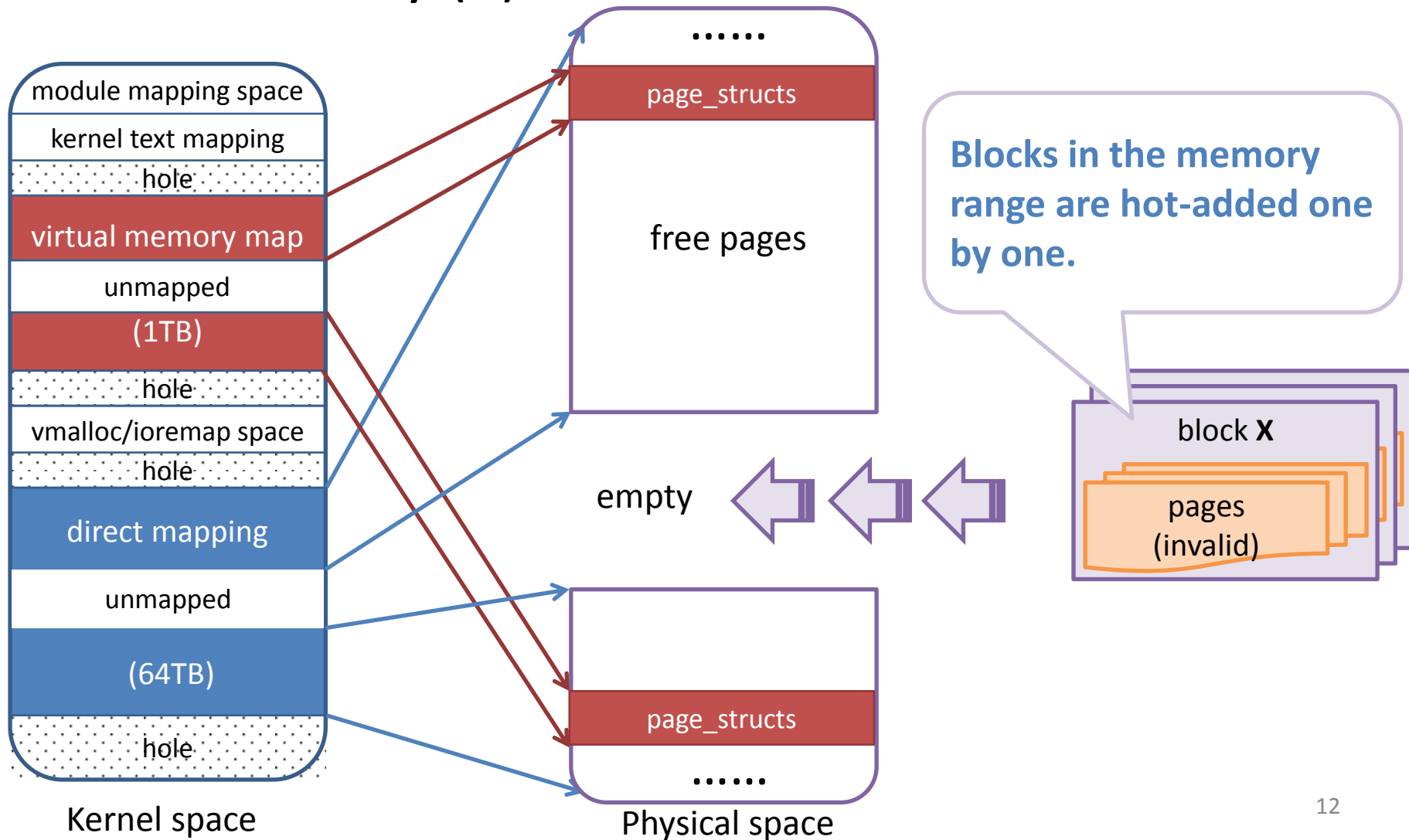
Generic memory definition divides a memory range into several blocks.
(128MB per block by default on x64)



Sparse memory model divides a memory range into several sections, in which the memory is contiguous.
(128MB per section, one section per block by default on x64)

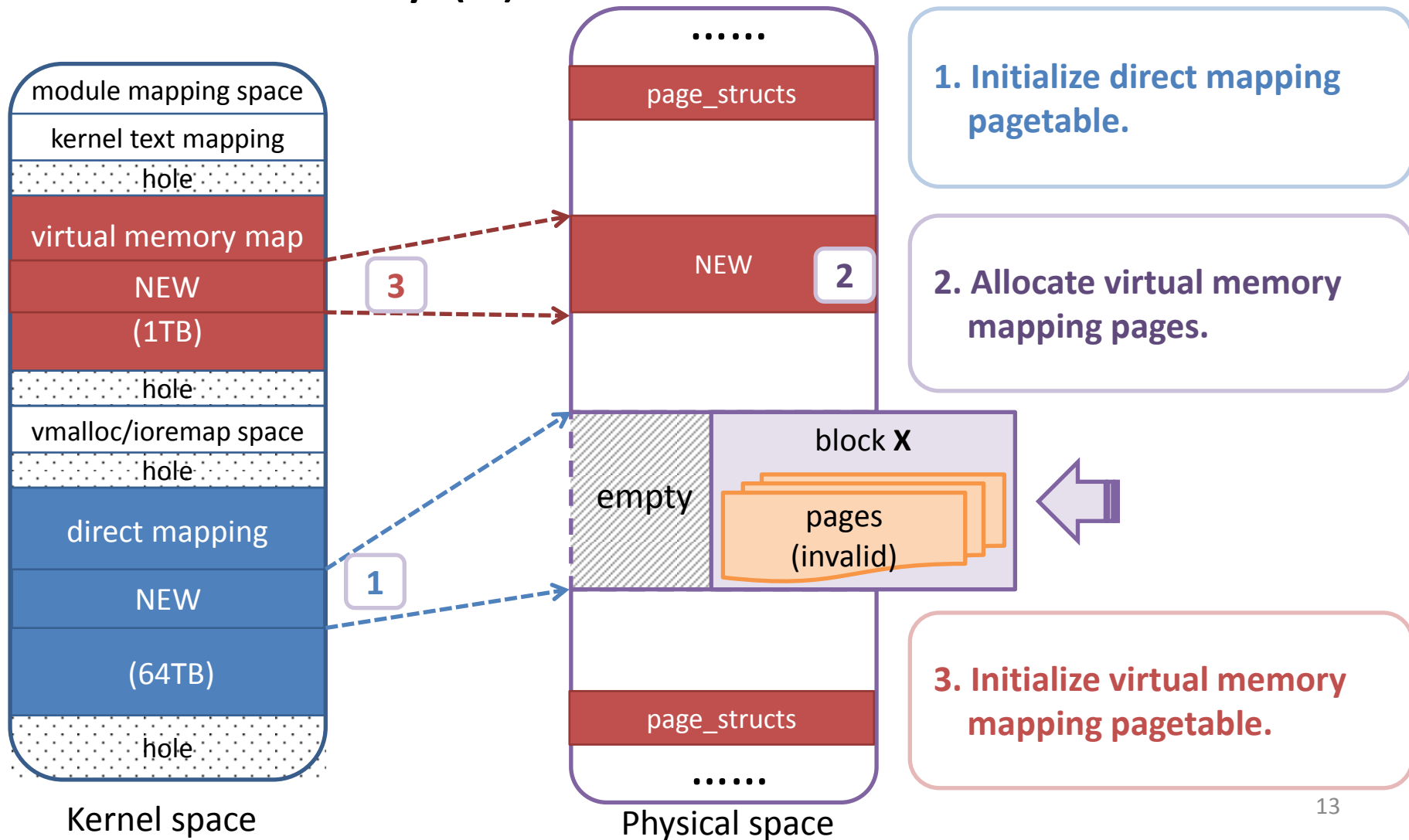
Memory hot-add

- Add memory (1)



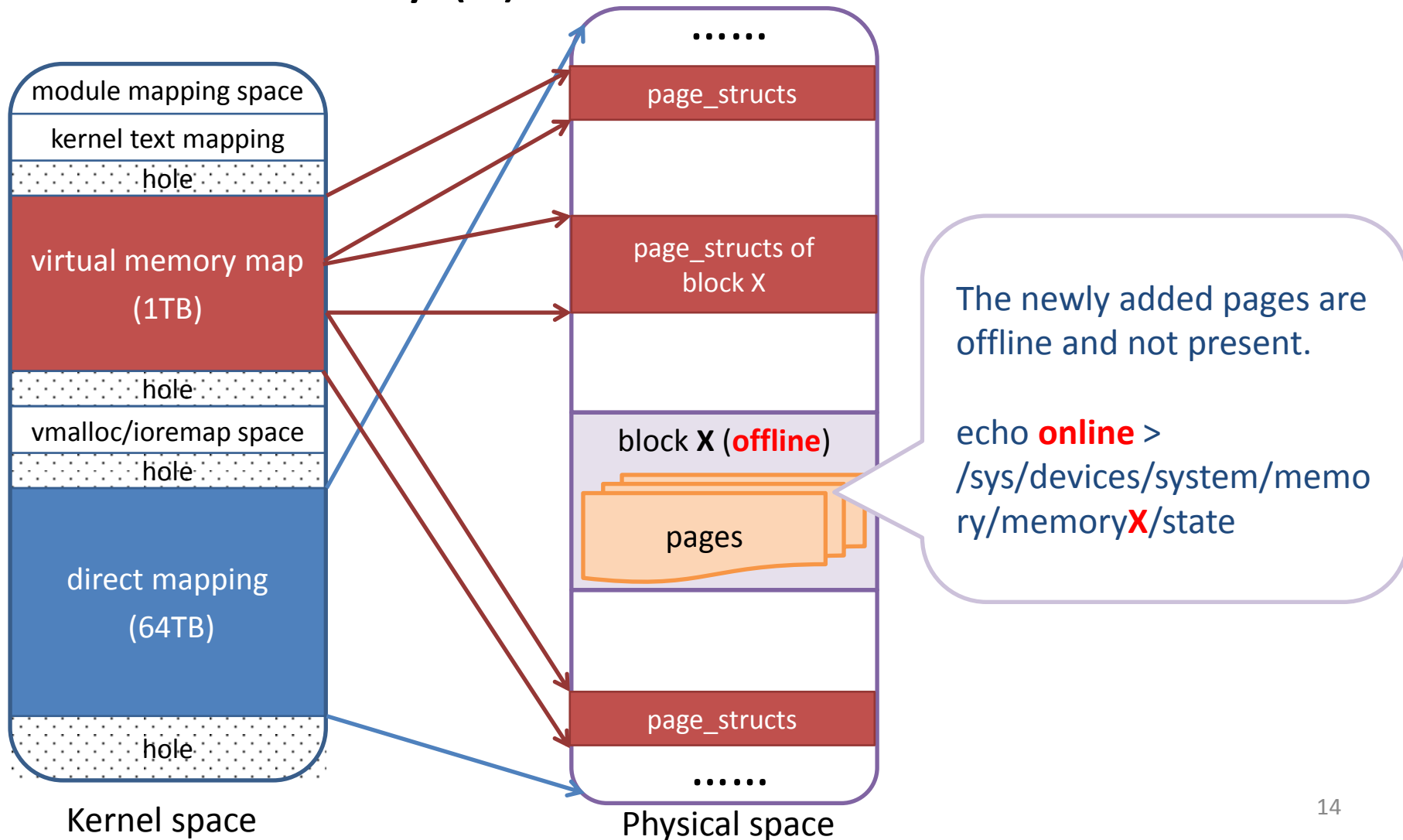
Memory hot-add

- Add memory (2)



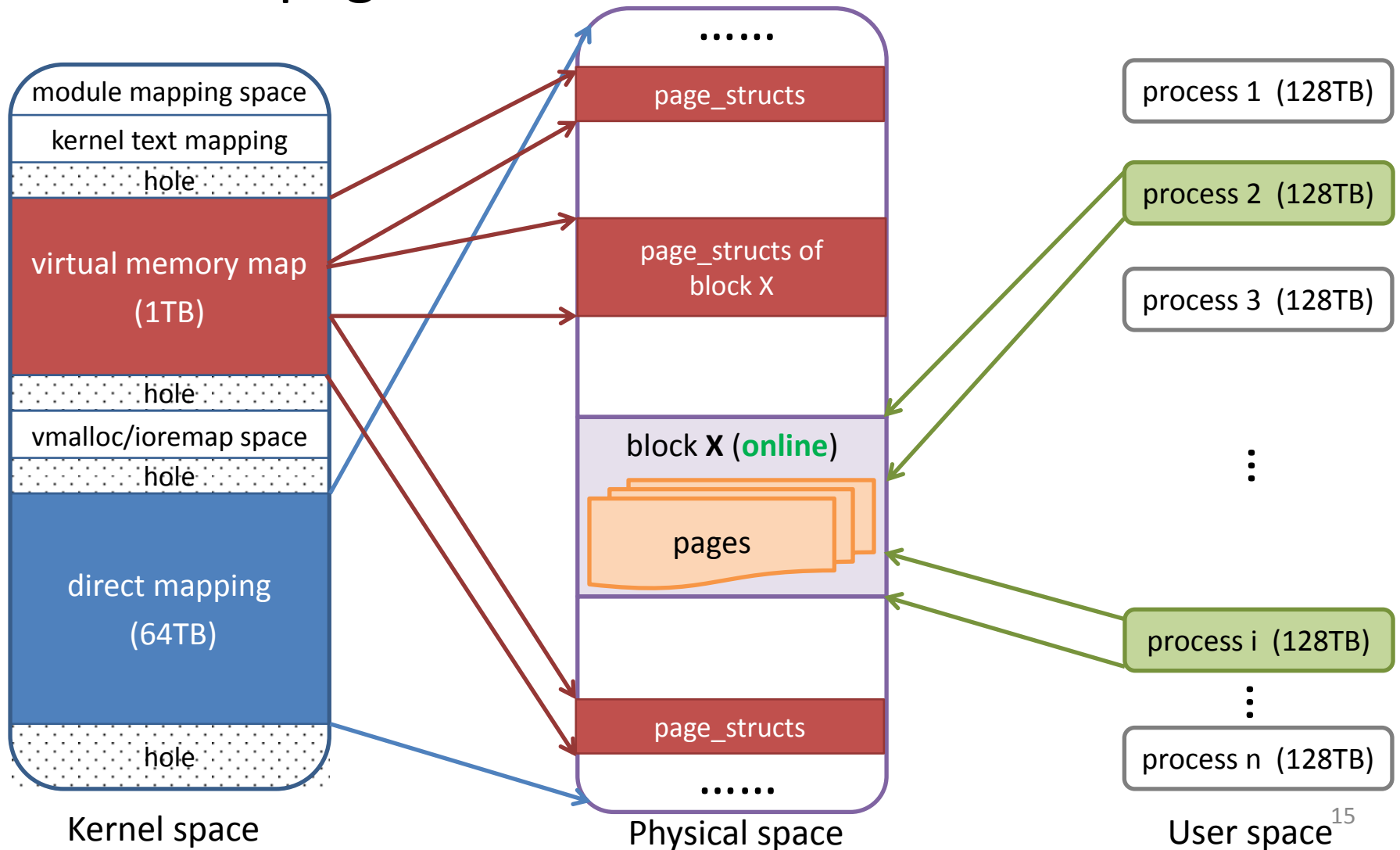
Memory hot-add

- Add memory (3)



Memory hot-add

- Online pages



Memory hot-add

- Configuration
 - mm/Kconfig

config **MEMORY_HOTPLUG**

bool "Allow for memory hot-add"

depends on SPARSEMEM || X86_64_ACPI_NUMA

depends on HOTPLUG && ARCH_ENABLE_MEMORY_HOTPLUG

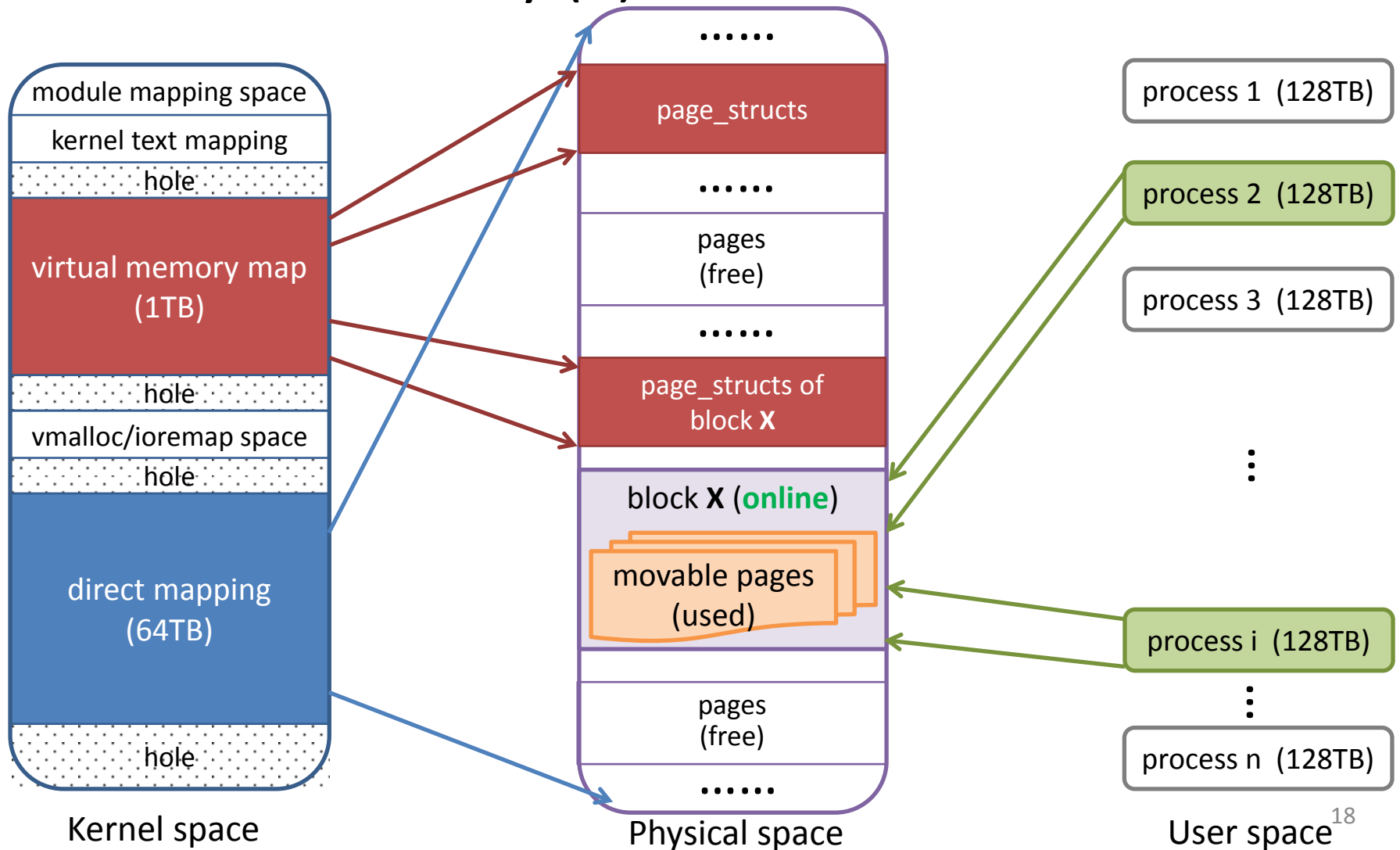
depends on (IA64 || X86 || PPC_BOOK3S_64 || SUPERH || S390)

Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

Memory hot-remove

- Remove memory (1)



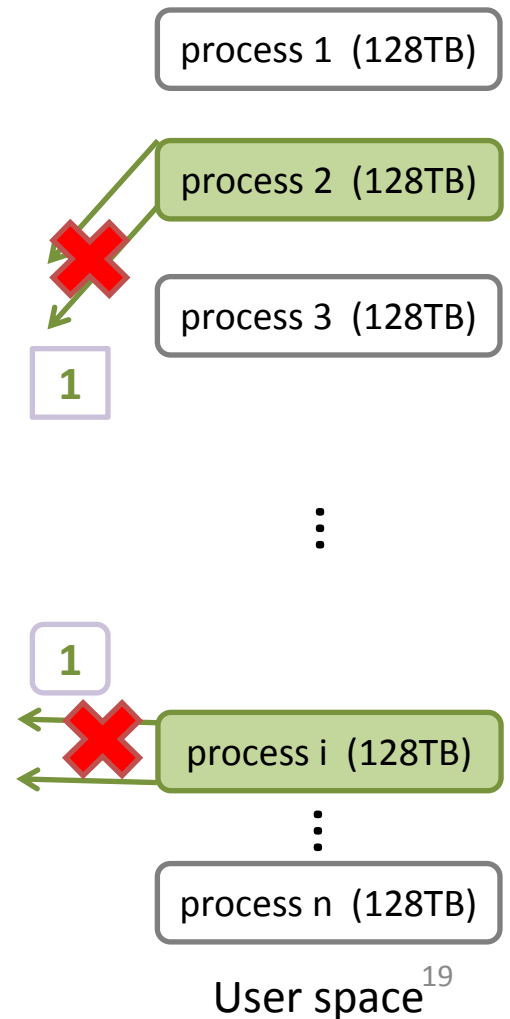
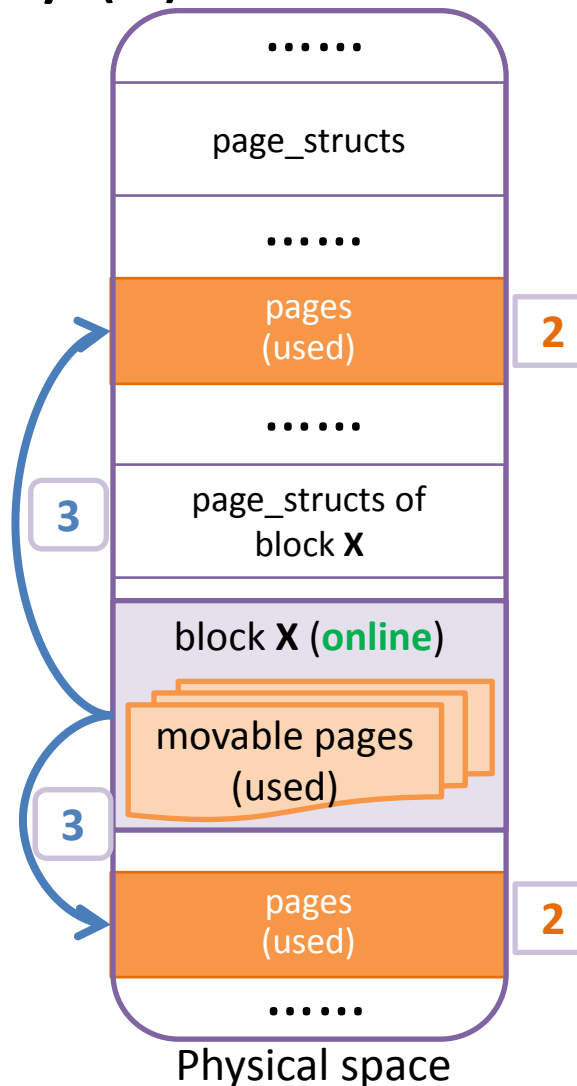
Memory hot-remove

- Remove memory (2)

1. **Unmap user pages.**
 - Kernel will generate a page fault for each process who access these pages, and the process will wait till the migration is over.

2. **Allocate new pages.**

3. **Copy data from old pages to new pages.**



Memory hot-remove

- Remove memory (3)

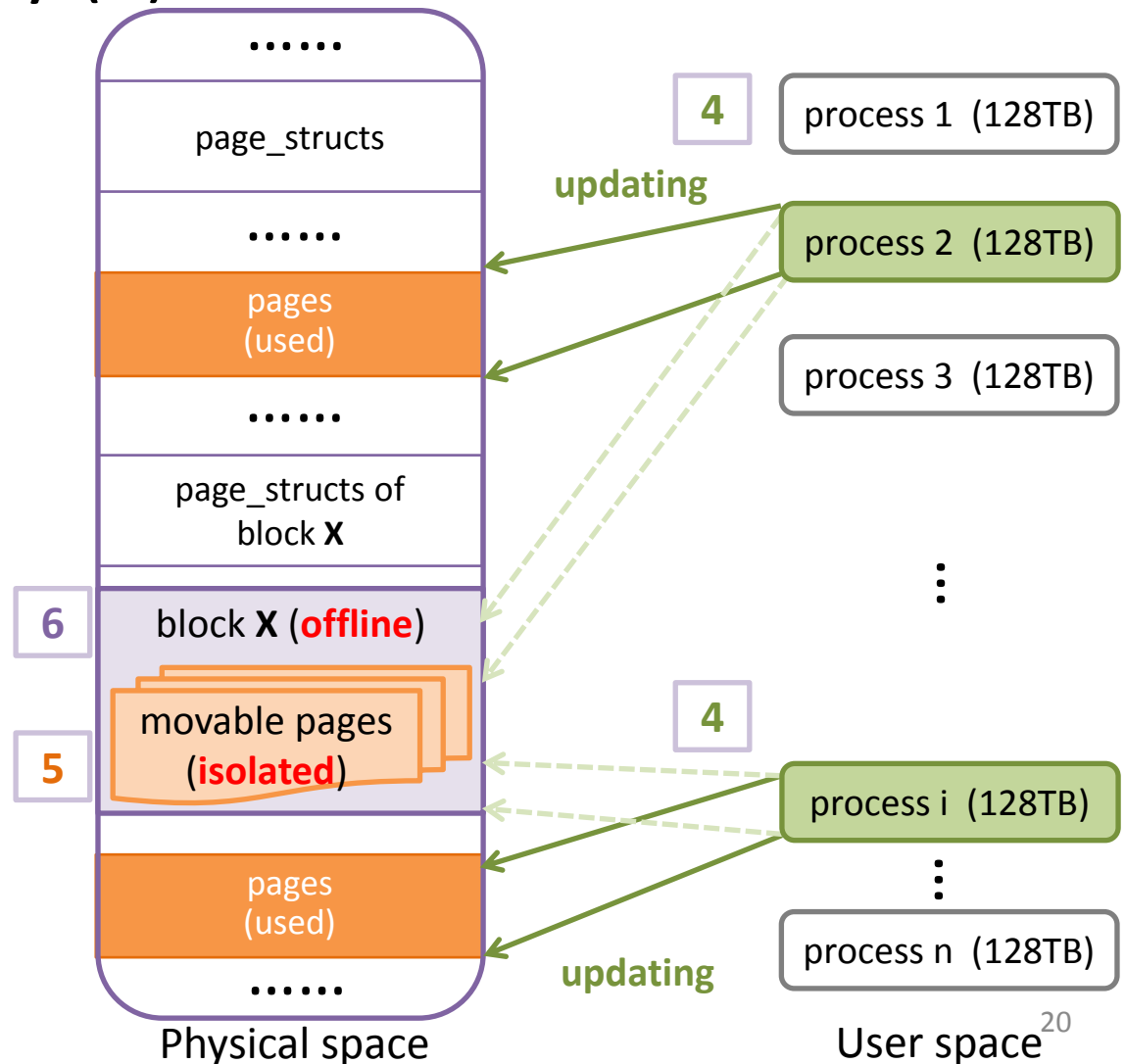
4. Update user processes' pagetable.

- Also wake up all the processes waiting for these pages.

5. Isolate old pages.

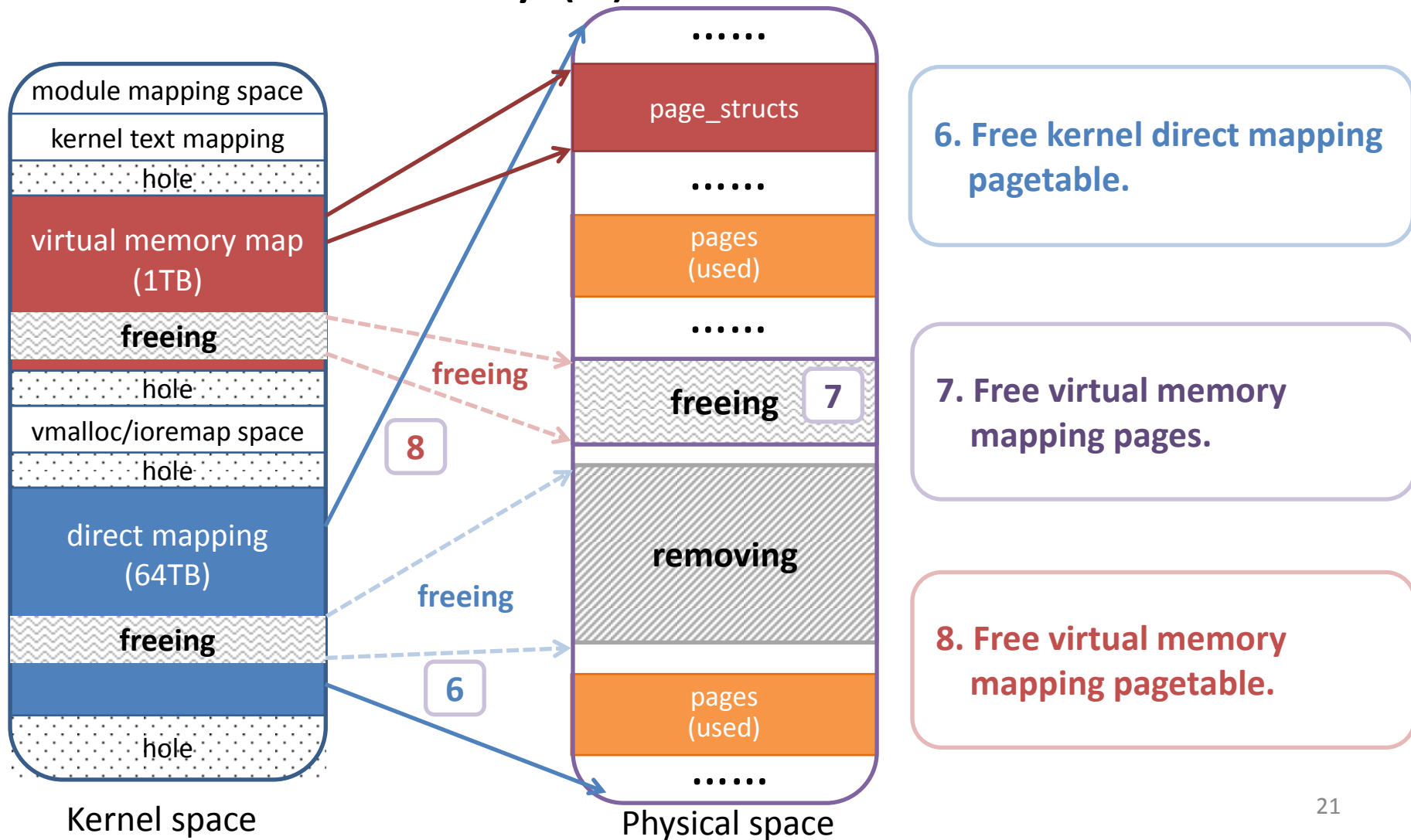
- Pages are not in the buddy system, and won't be allocated to anyone.

6. Set the block state to offline.



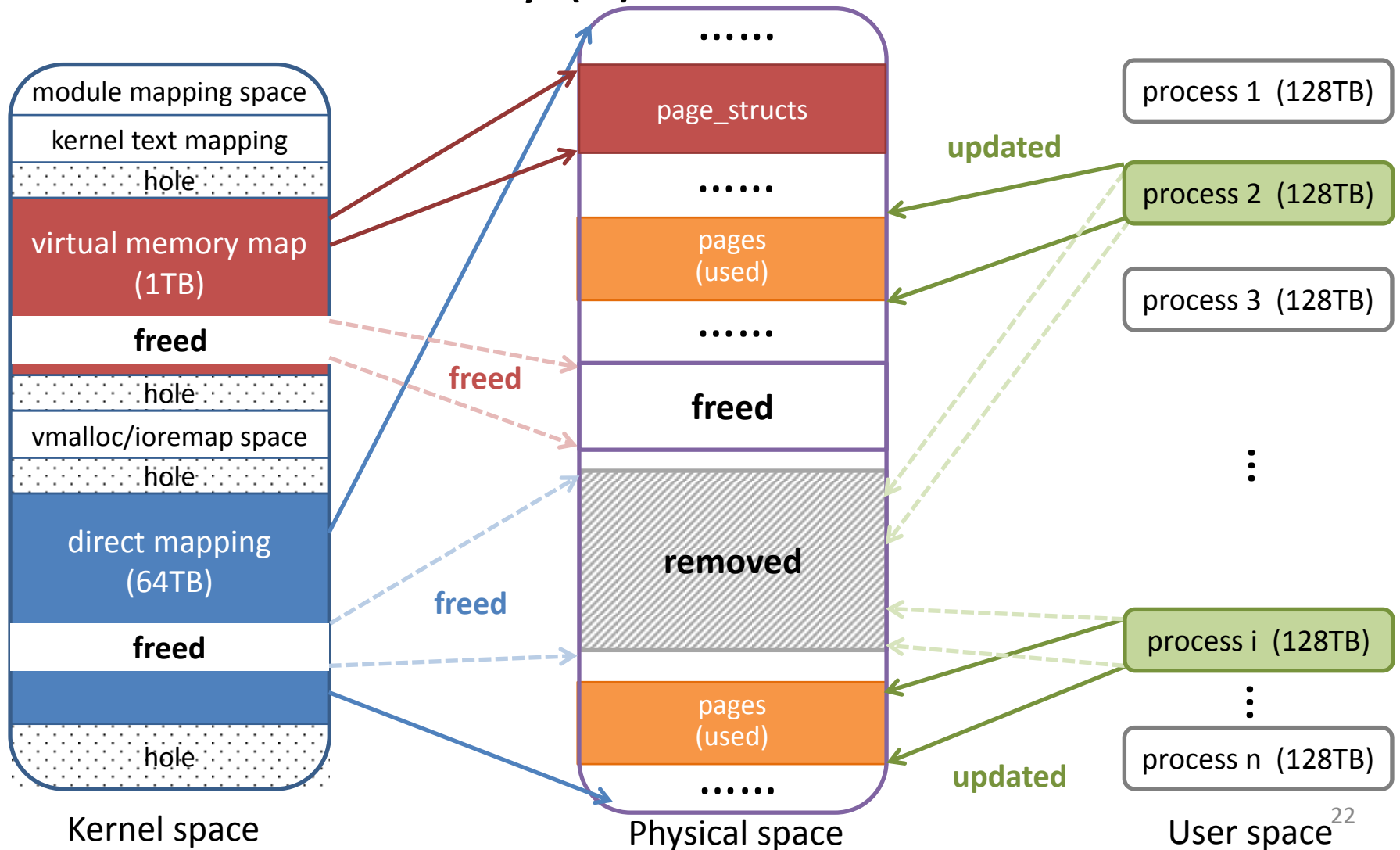
Memory hot-remove

- Remove memory (4)

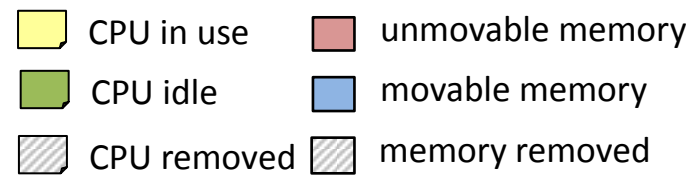


Memory hot-remove

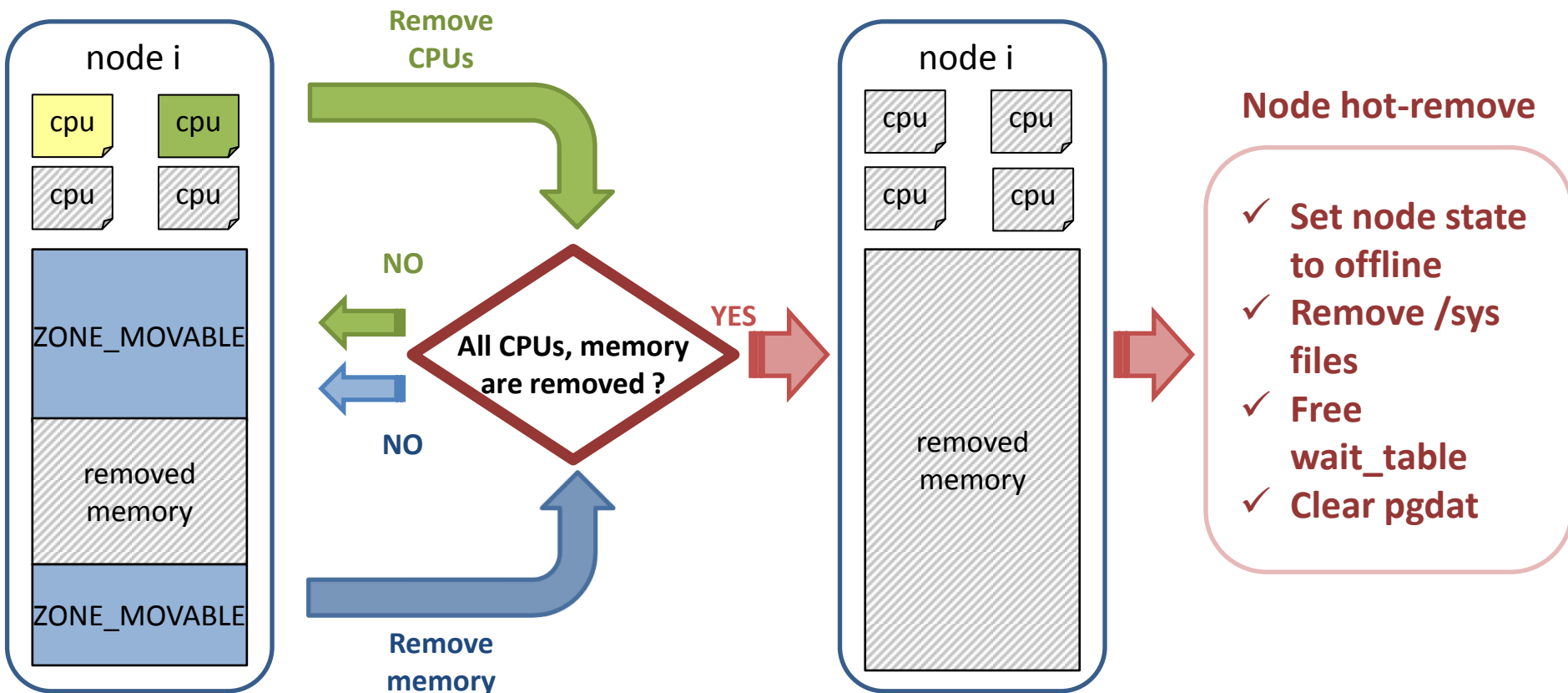
- Remove memory (5)



Memory hot-remove



- Post work: automatically remove the node



Memory hot-remove

- Merged into Linux 3.9
- Configuration
 - mm/Kconfig

config **MEMORY_HOTREMOVE**

bool "Allow for memory hot remove"

select MEMORY_ISOLATION

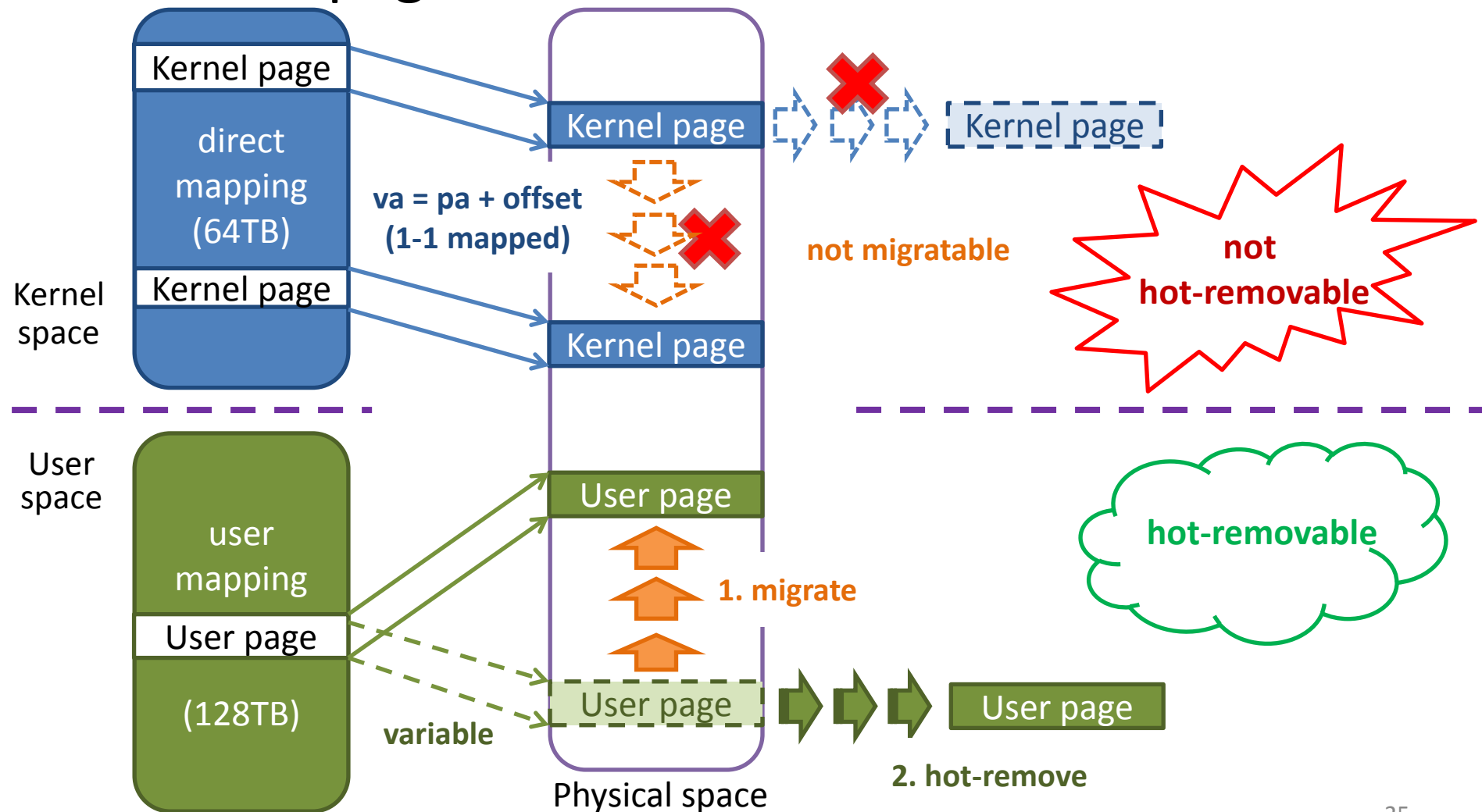
select HAVE_BOOTMEM_INFO_NODE if X86_64

depends on MEMORY_HOTPLUG && ARCH_ENABLE_MEMORY_HOTREMOVE

depends on MIGRATION

Memory hot-remove

- Kernel pages cannot be hot-removed

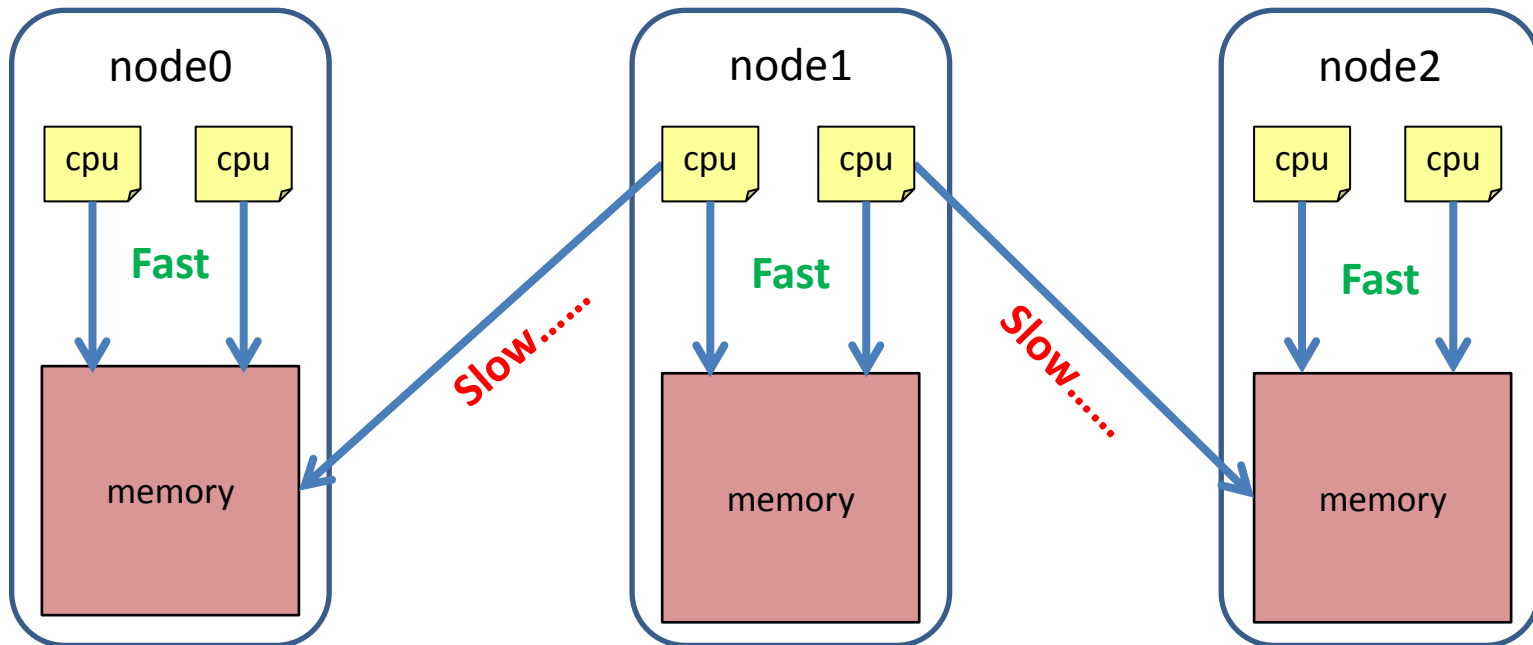


Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

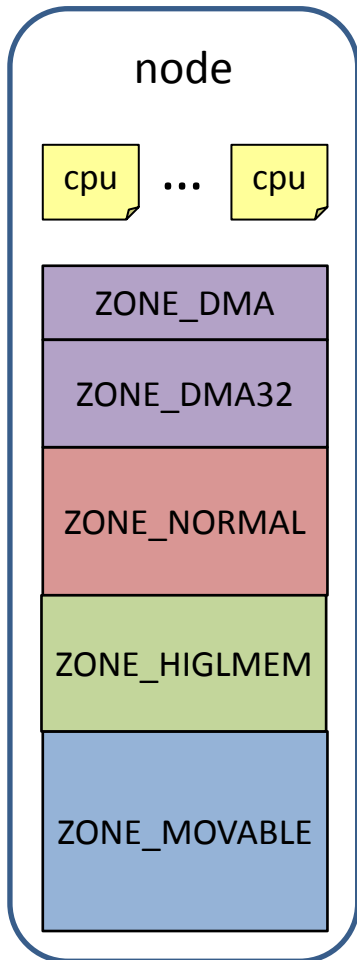
Movable node

- NUMA: **N**on-**U**niform **M**emory **A**ccess
 - A node consists of a set of CPUs and memory.
 - CPUs access memory in the same node faster.
 - Meaningful in 64bits platform only. 32bits platform has only one node.



Movable node

- Zone: Different types of memory in a node



ZONE_DMA / ZONE_DMA_32 (64bits only): used for DMA.

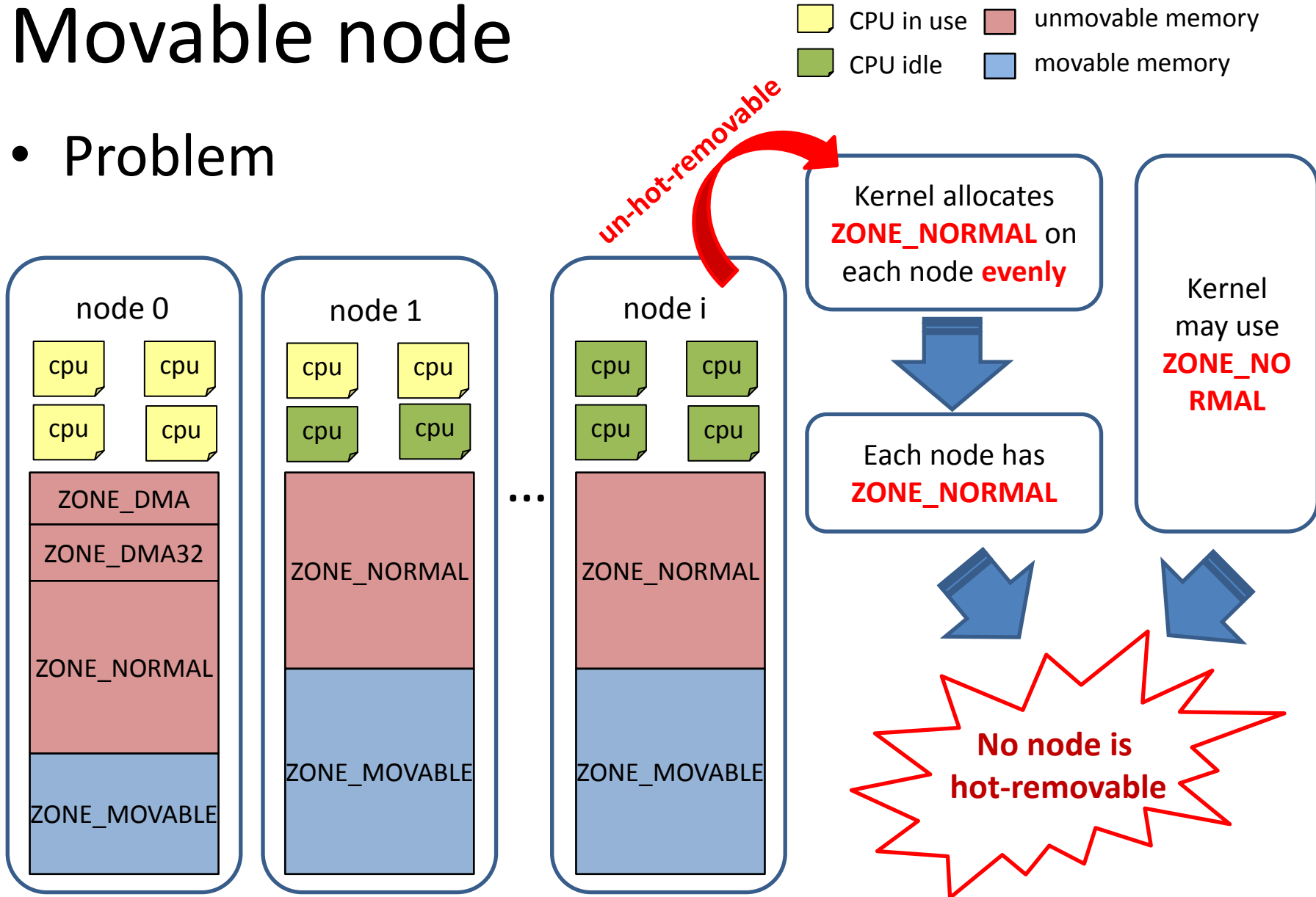
ZONE_NORMAL: Memory directly mapped, used by kernel and user space.

ZONE_HIGHMEM: For 32bits kernel to access memory not directly mapped.

ZONE_MOVABLE: Memory can be migrated. (user space only)

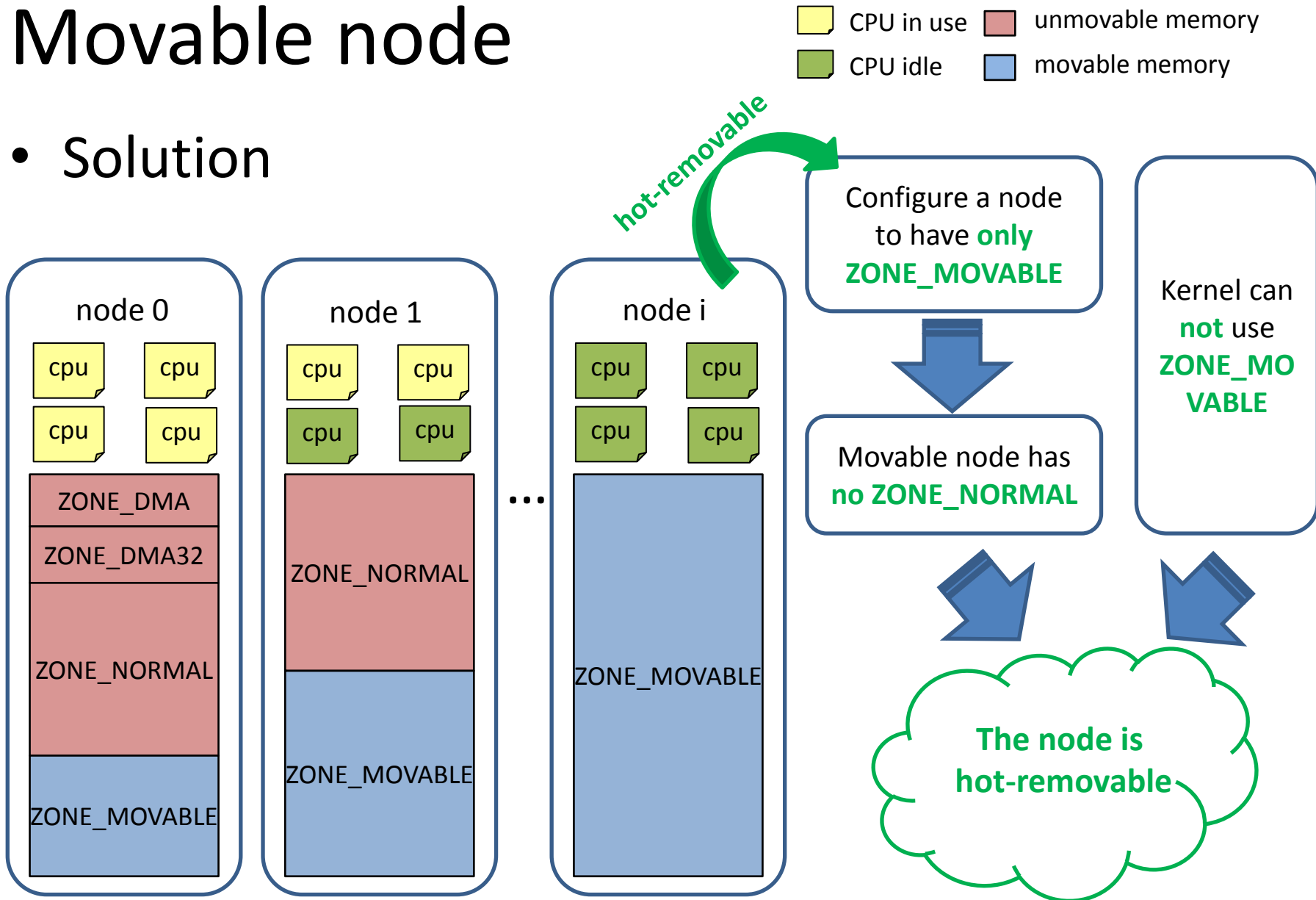
Movable node

- Problem



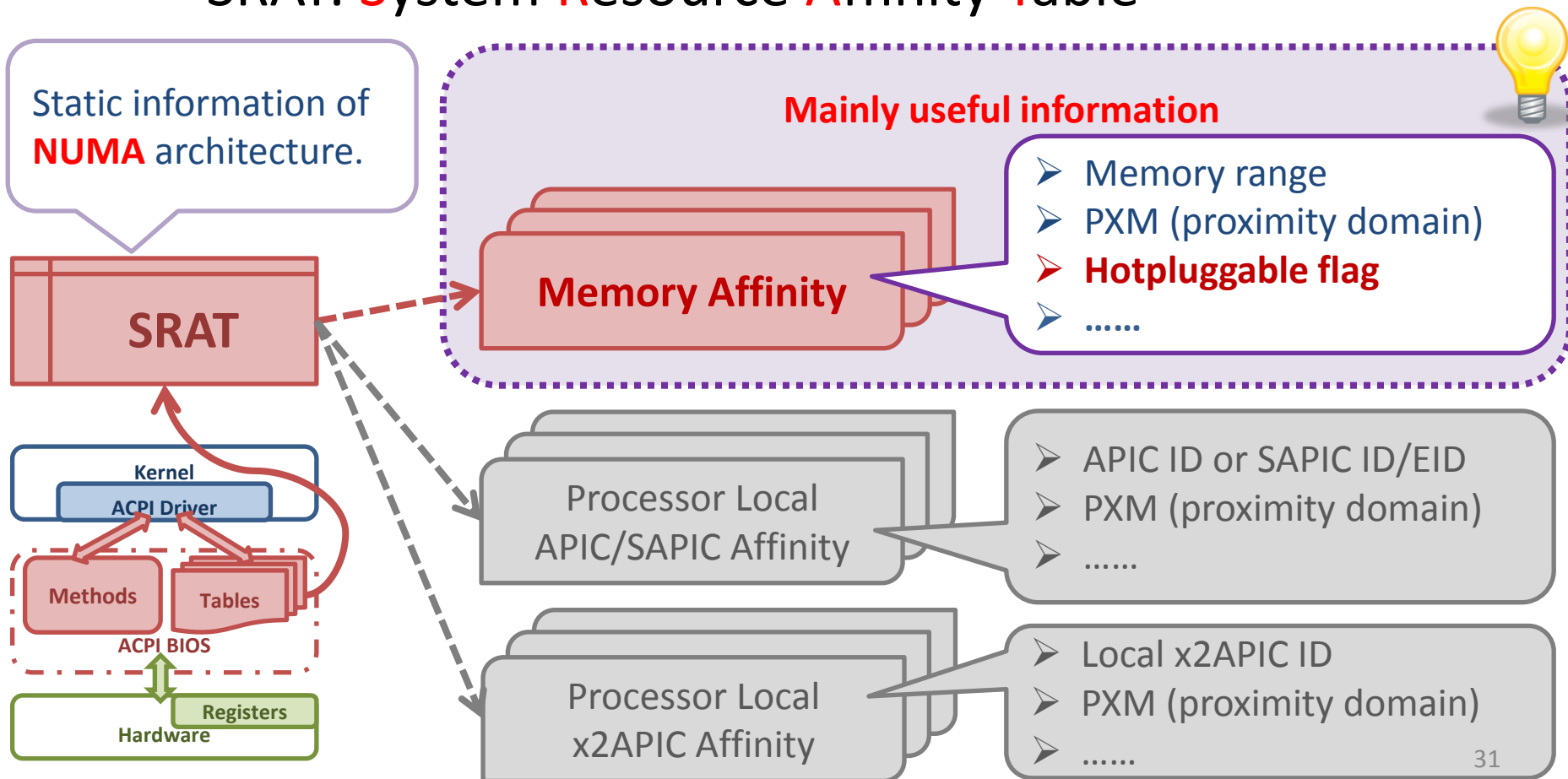
Movable node

- Solution



Movable node

- Static configuration
 - SRAT: **S**ystem **R**esource **A**ffinity **T**able

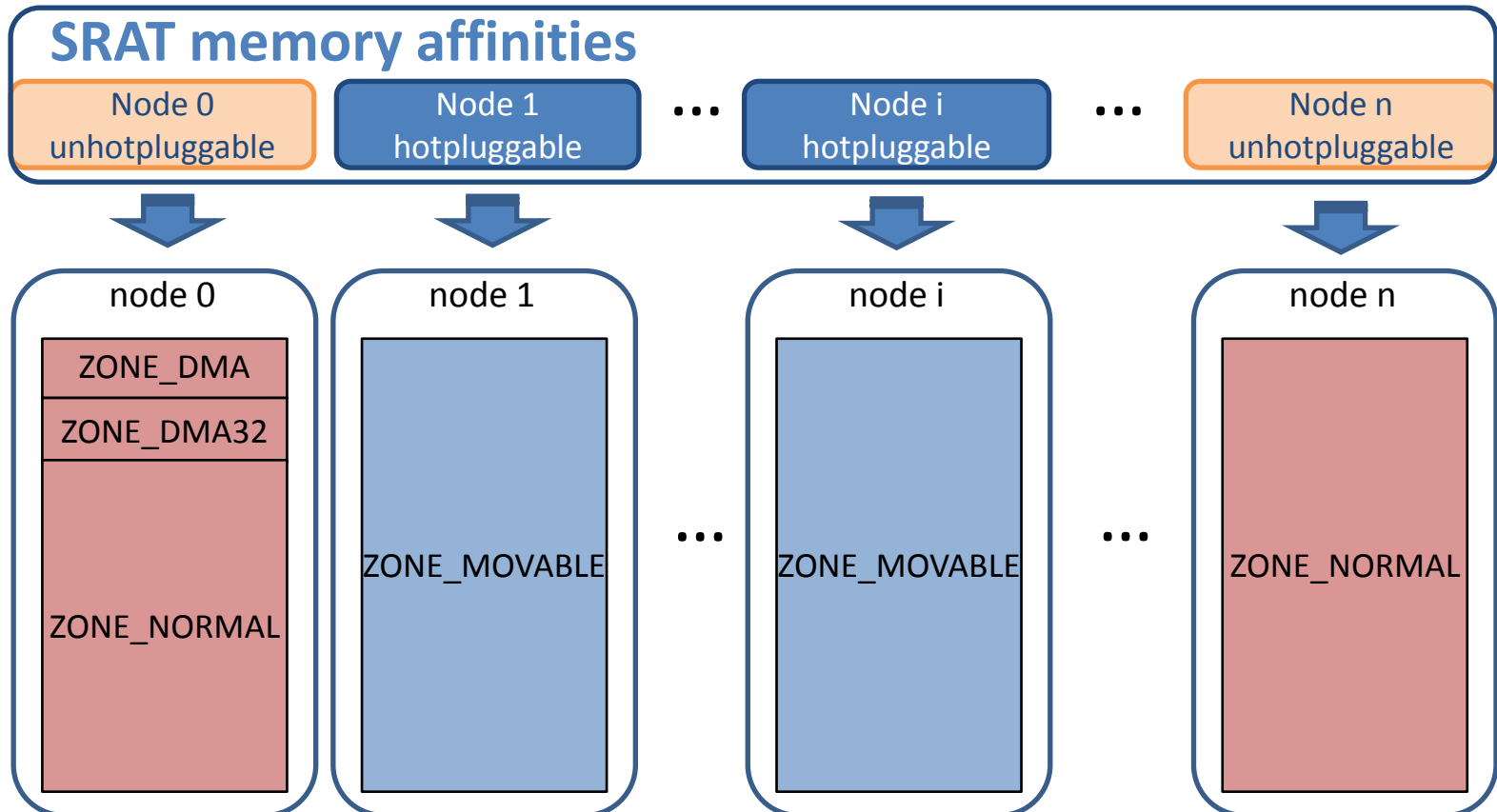


Movable node

■ unmovable memory
■ movable memory

- movable_node
(New)

- Use SRAT to arrange ZONE_MOVABLE.
- Only for memory hotplug users.
- Still being pushing.

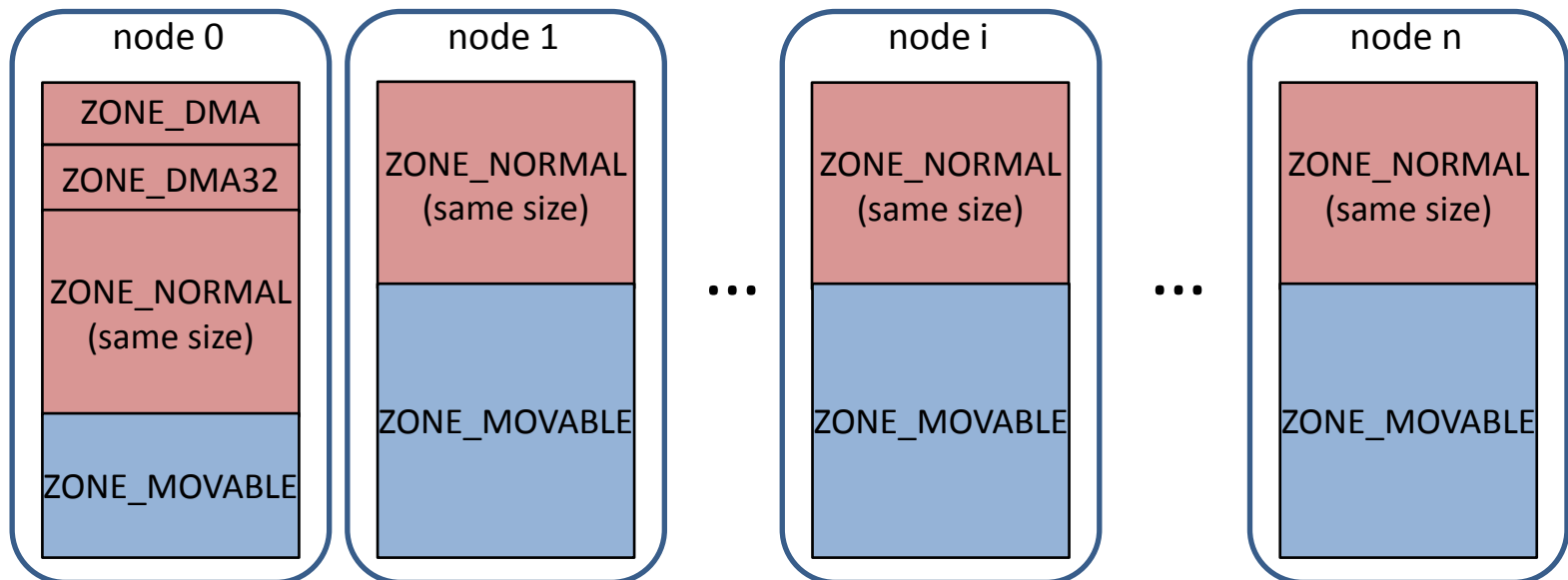


Movable node

■ unmovable memory
■ movable memory

- The old way (no performance lost)
 - kernelcore / movablecore = nn {G|M|K} (**Old**)

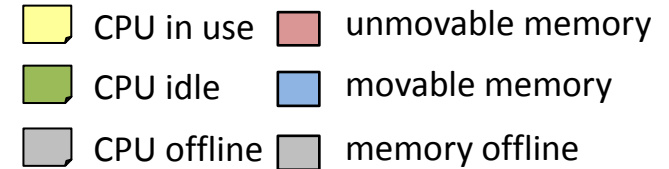
- Allocate ZONE_MORMAL in each node evenly.
- For regular users.



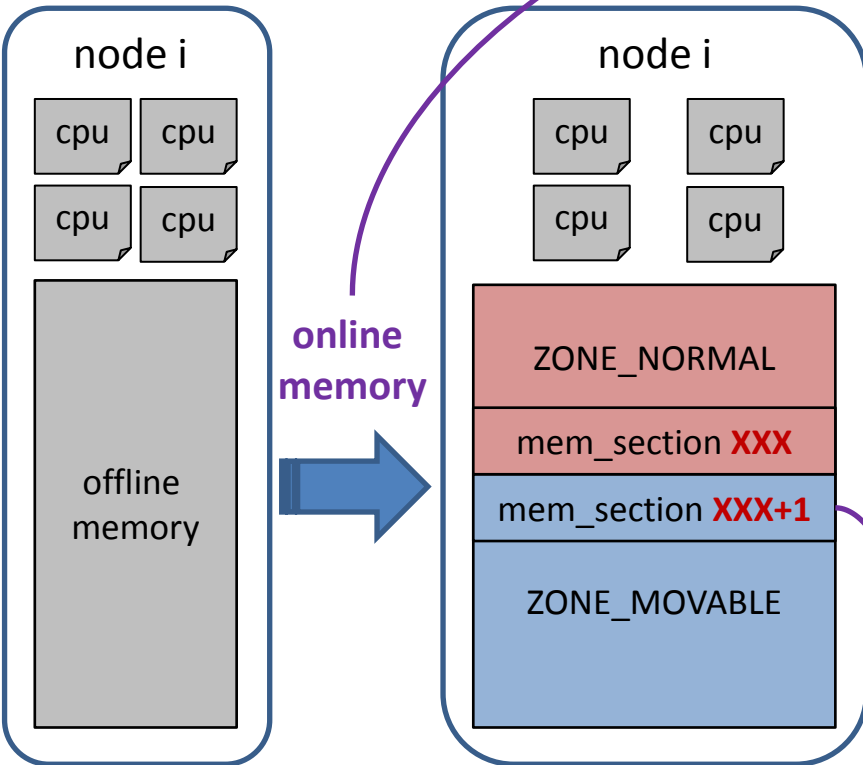
Movable node

- Dynamic configuration

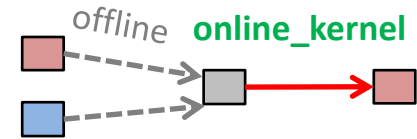
offline and online again



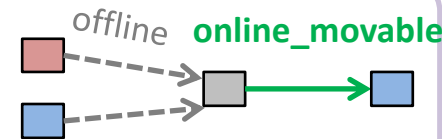
echo **COMMAND** >
/sys/devices/system/node/nodei/memoryXXX/state



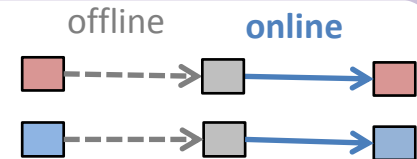
1. **online_kernel (NEW)**
Set to ZONE_NORMAL.



2. **online_movable (NEW)**
Set to ZONE_MOVABLE.



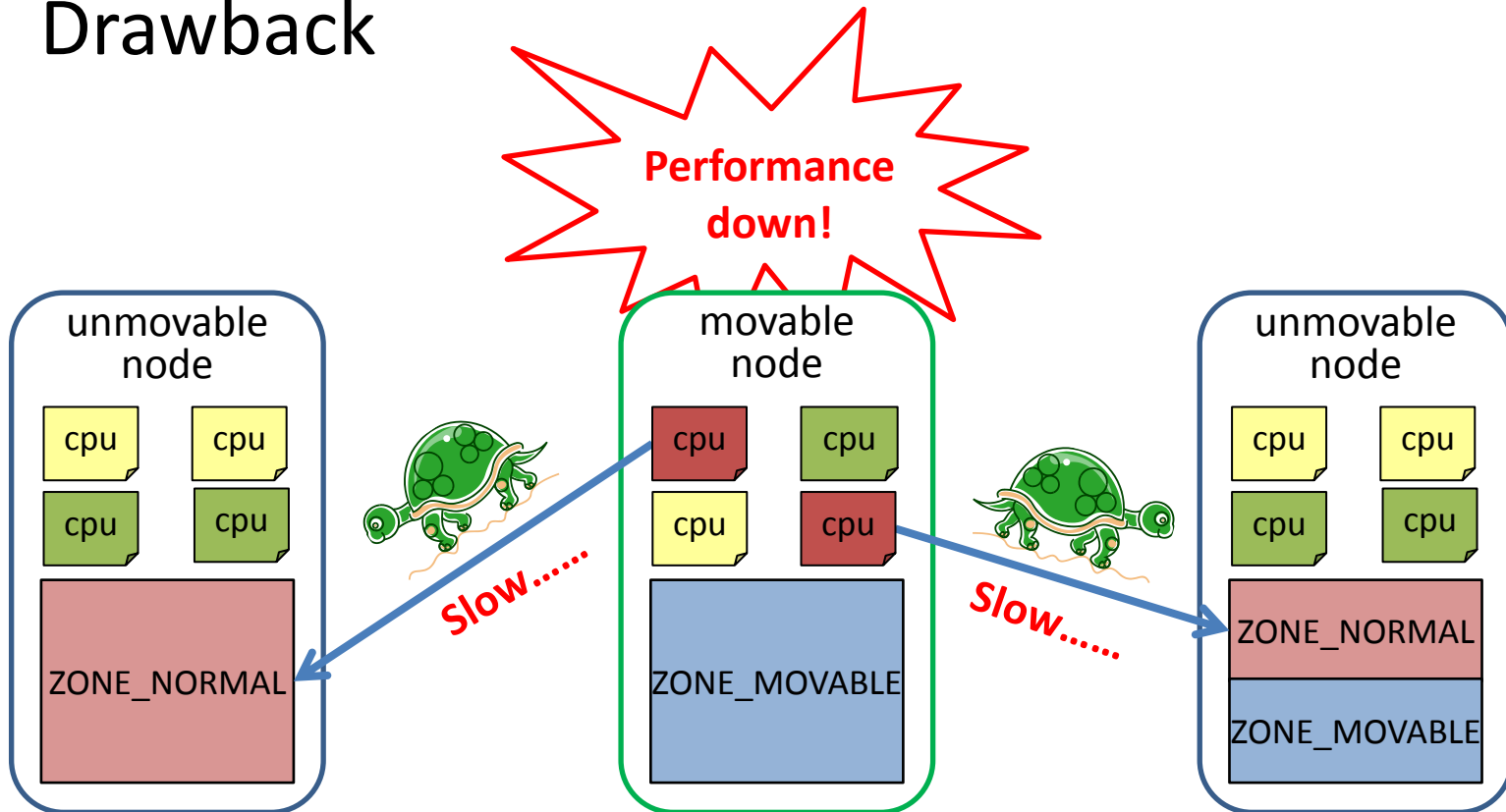
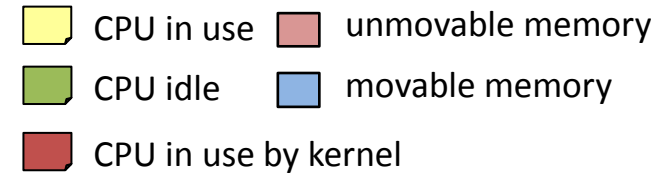
3. **online (Improved)**
Keep the previous state.
(ZONE_NORMAL for the first time)



Rule: **ZONE_MOVABLE** should always be after **ZONE_NORMAL**, never overlaps.

Movable node

- Drawback



No good enough way to solve this problem now.

Movable node

- Merged into Linux 3.8
- Configuration
 - mm/Kconfig

config **MOVABLE_NODE**

boolean "Enable to assign a node which has only movable memory"

depends on HAVE_MEMBLOCK

depends on NO_BOOTMEM

depends on X86_64

depends on NUMA

default n

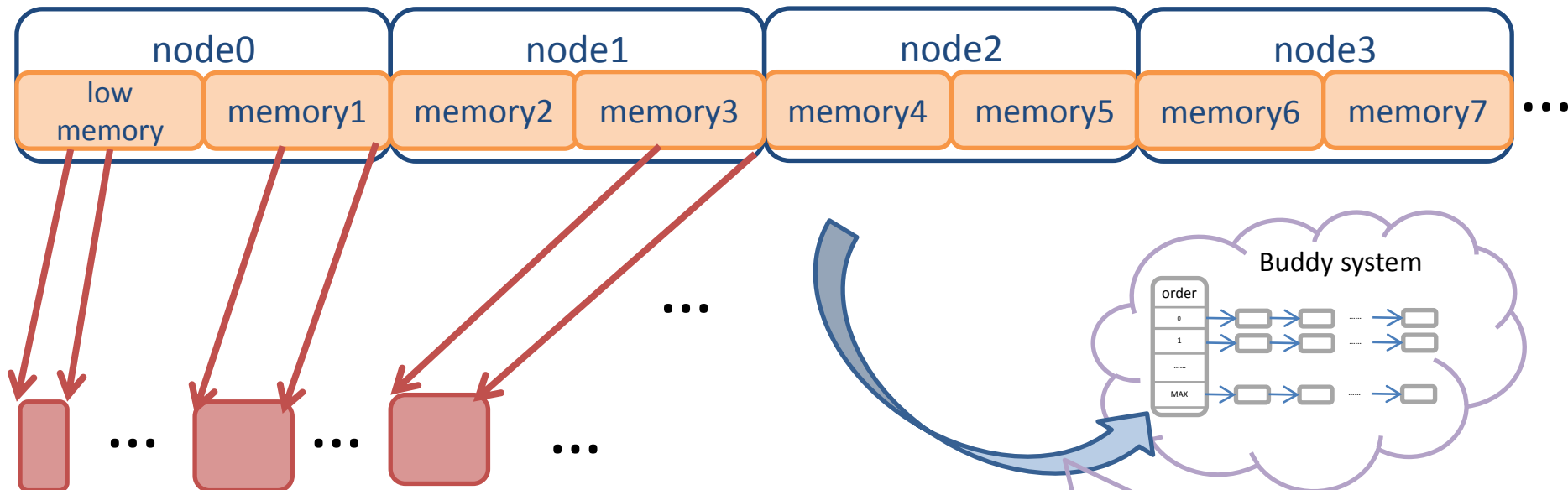
Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

Bootmem handling

- Memblock: A bootmem allocator
 - Consists of two arrays

`memblock.memory[]`: All the present memory in the system.

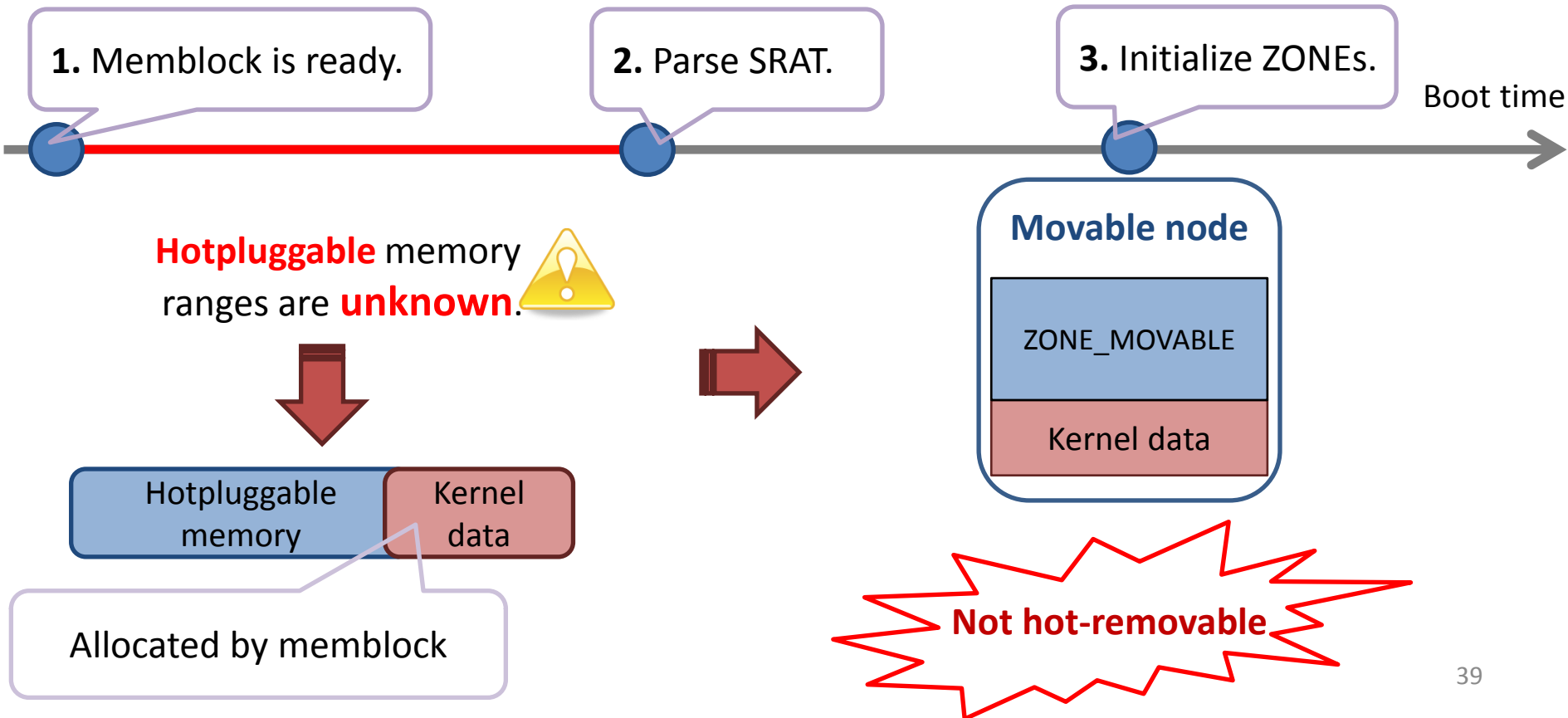


`memblock.reserve[]`: Allocated memory.

Free unused memory to buddy system at last.

Bootmem handling

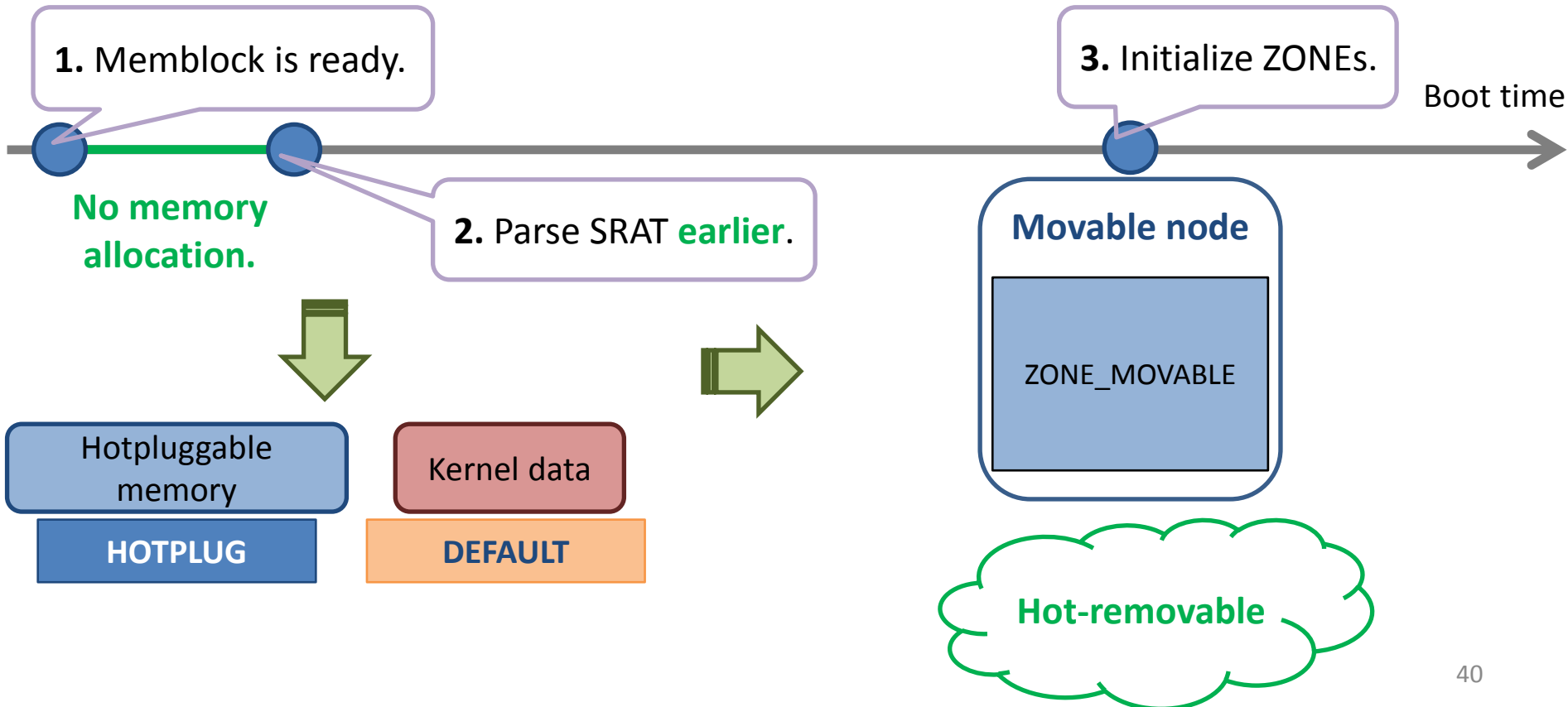
- Problem at boot time
 - Bootmem allocator memblock may allocate **hotpluggable** memory for kernel at boot time.



Bootmem handling

- Solution 1

1. Parse SRAT **earlier**, before memblock starts to work.
2. Introduce flags into memblock, and mark hotpluggable memory with a special **flag** in memblock.



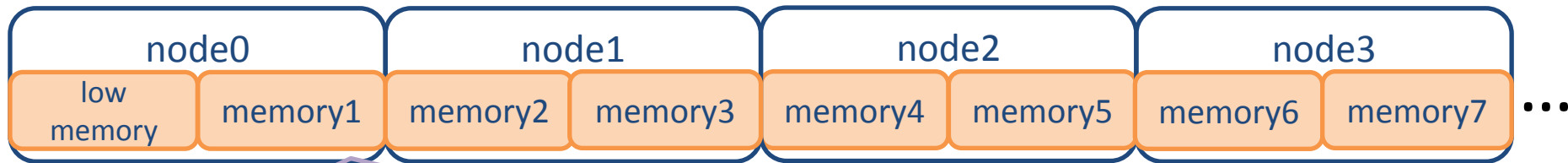
Bootmem handling

- unhotpluggable memory
- hotpluggable memory

1. Memblock is ready.

Boot time

memblock.memory[]



All the memory ranges in the system are put into memory[].

Allocated memory ranges will be put into reserve[].

empty

memblock.reserve[]

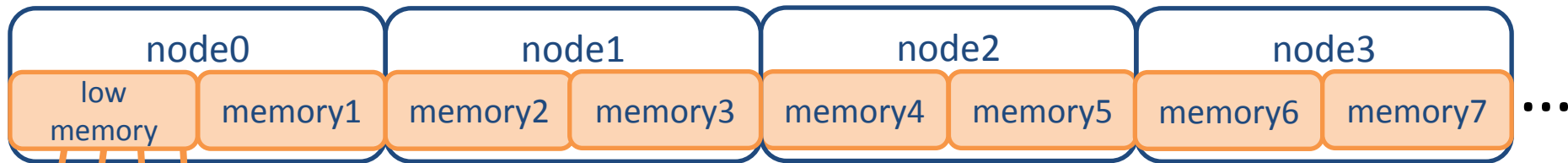
Bootmem handling

orange square unhotpluggable memory
blue square hotpluggable memory

2. Before parsing **SRAT**.

Boot time

memblock.memory[]

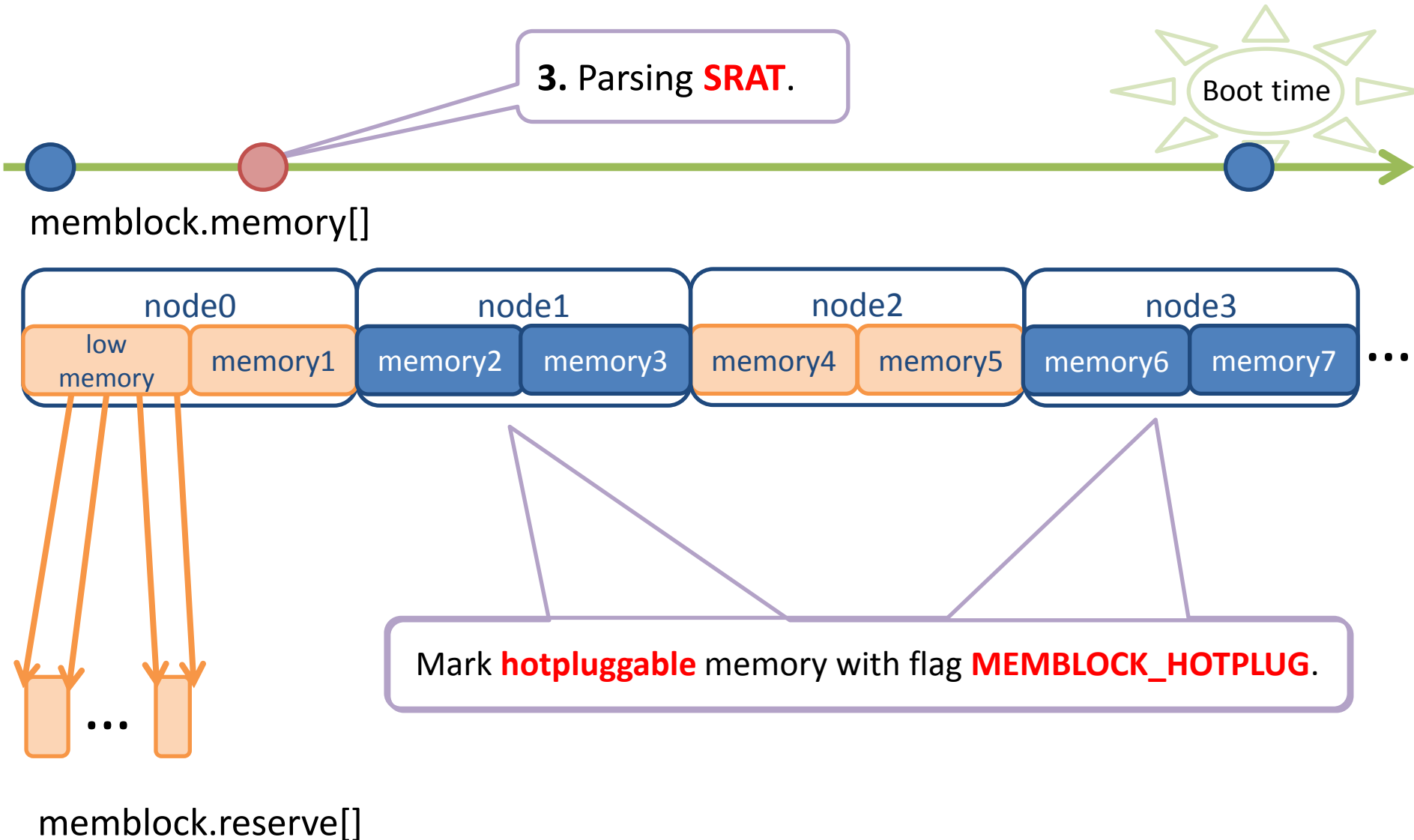


- Reserve kernel `_data`, `_text`, setup data, ...
- Any node the kernel resides in is **unhotpluggable**. (Not necessary to be node 0)
- **No new memory allocation**, so **no hotpluggable memory could be used by the kernel**.
- **Remaining memory are in memory array with no flag (currently still don't know which memory is hotpluggable)**

memblock.reserve[]

Bootmem handling

- unhotpluggable memory
- hotpluggable memory



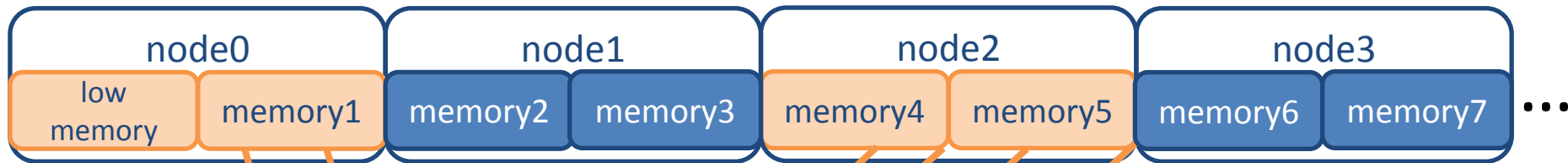
Bootmem handling

□ unhotpluggable memory
■ hotpluggable memory

4. After parsing SRAT, **hotpluggable** memory can not be allocated.

Boot time

memblock.memory[]



memblock.reserve[]

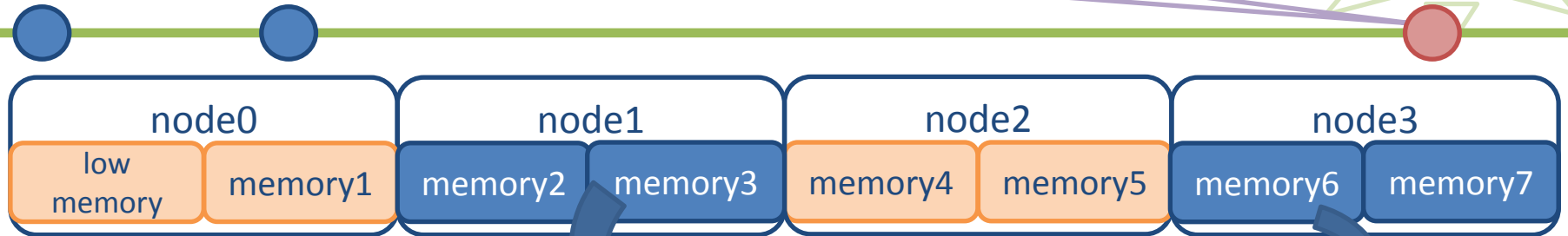
No hotpluggable memory used by kernel.

Bootmem handling

orange unhotpluggable memory
blue hotpluggable memory

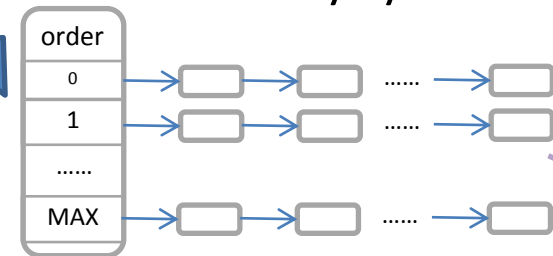
5. **Memory initialization** has been **finished**.

Boot time



Free hotpluggable memory to buddy system.

Buddy system

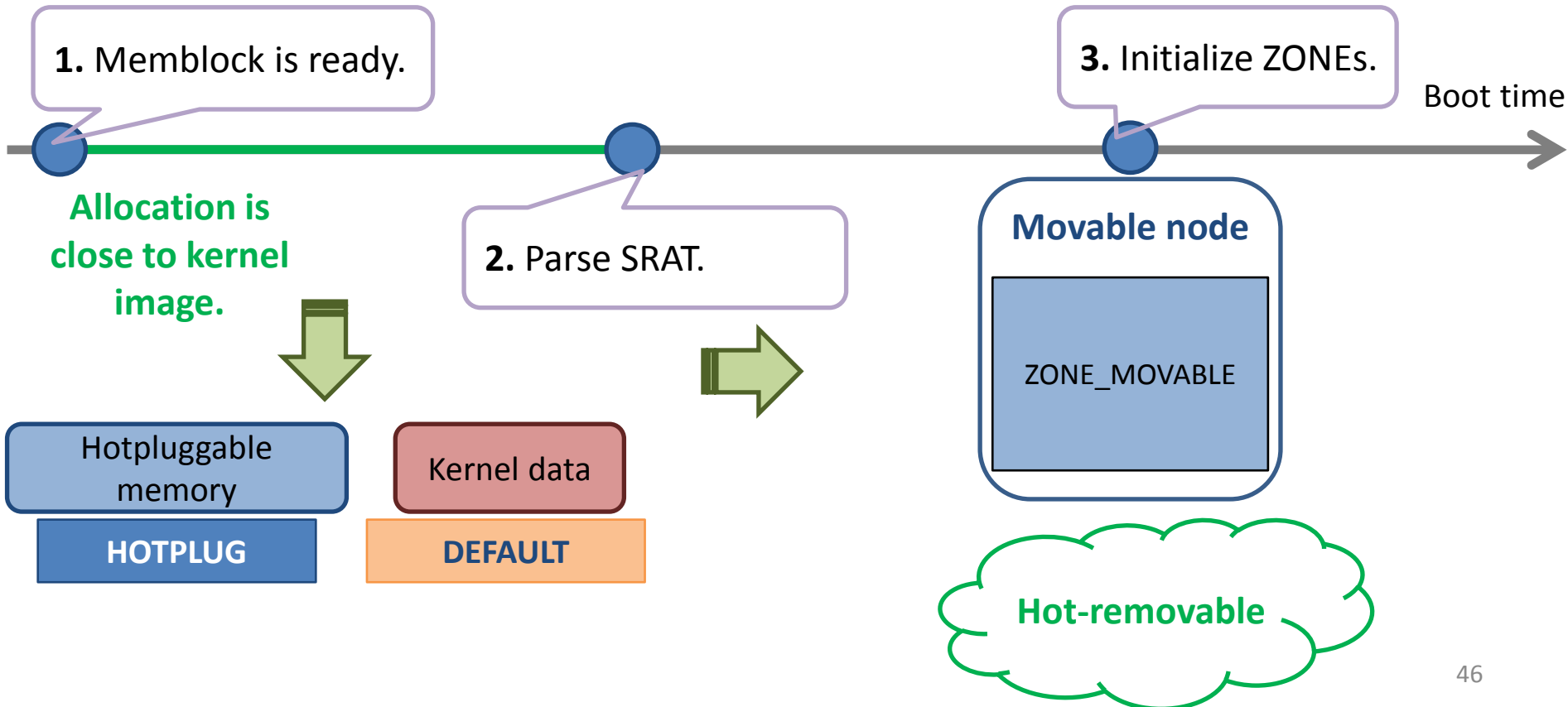


memblock.reserve[]

Bootmem handling

- Solution 2

1. After memblock starts working and before SRAT is parsed, allocate memory close to the kernel image.
2. Introduce flags into memblock, and mark hotpluggable memory with a special **flag** in memblock.



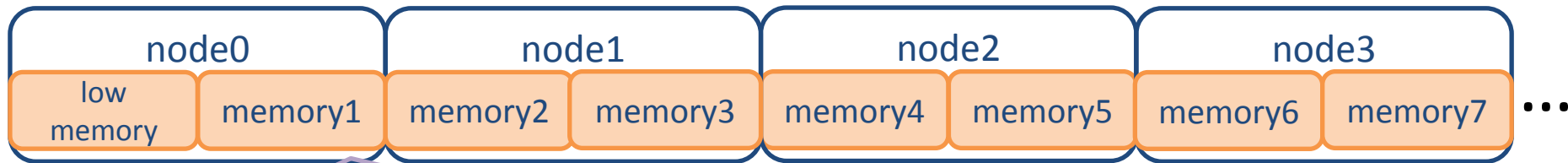
Bootmem handling

- unhotpluggable memory
- hotpluggable memory

1. Memblock is ready.

Boot time

memblock.memory[]



All the memory ranges in the system are put into memory[].

Allocated memory ranges will be put into reserve[].

empty

memblock.reserve[]

Bootmem handling

orange square unhotpluggable memory
blue square hotpluggable memory

2. Before parsing **SRAT**.

Boot time

memblock.memory[]

node0

node1

node2

node3

low
memory

memory1

memory2

memory3

memory4

memory5

memory6

memory7 ...

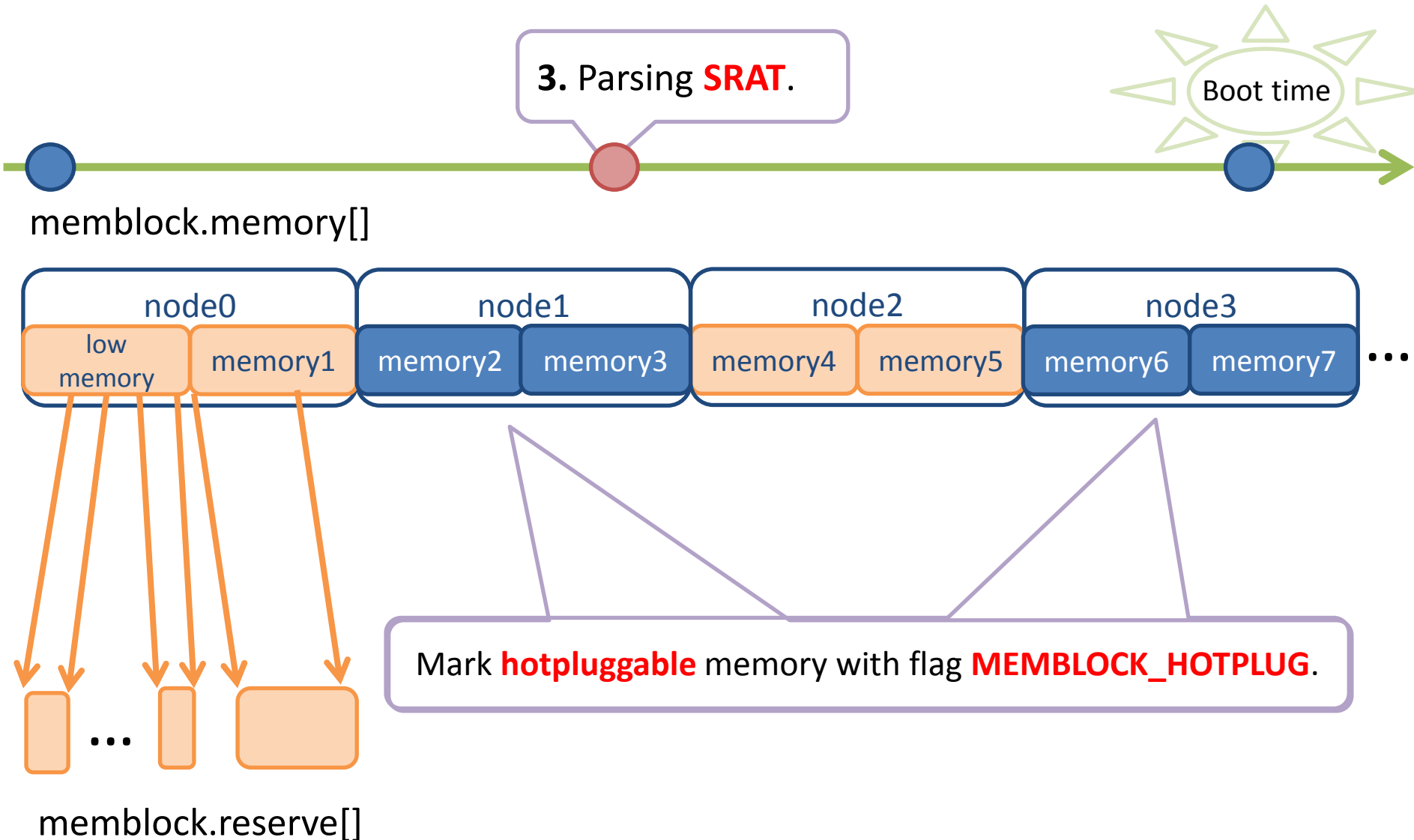
- Remaining memory are in memory array with **no flag** (currently still don't know which memory is hotpluggable)

- Reserve kernel _data, _text, setup data,
- Any node the kernel resides in is **unhotpluggable**. (Not necessary to be node 0)
- **It is highly likely that allocation near kernel is contained in the same NUMA node**

memblock.reserve[]

Bootmem handling

- unhotpluggable memory
- hotpluggable memory



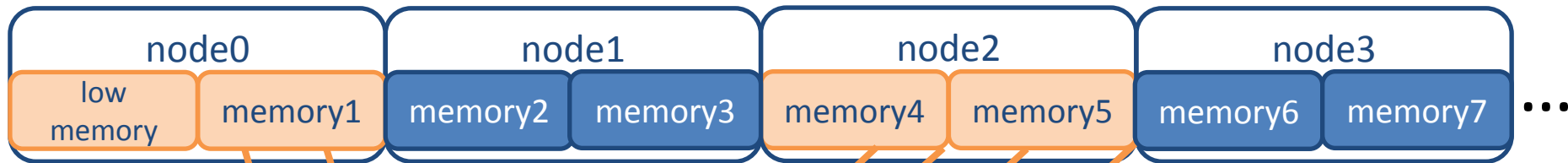
Bootmem handling

- unhotpluggable memory
- hotpluggable memory

4. After parsing SRAT, **hotpluggable** memory can not be allocated.

Boot time

memblock.memory[]



memblock.reserve[]

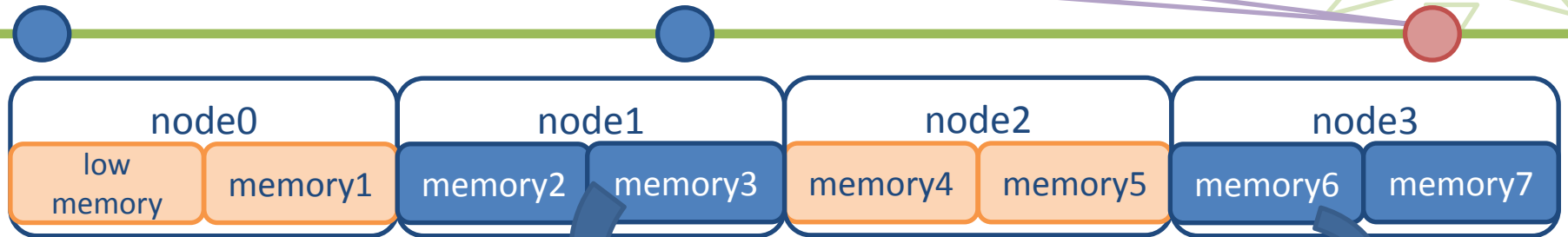
- No** hotpluggable memory used by kernel.
- *No*** need to allocate close the kernel image.

Bootmem handling

orange unhotpluggable memory
blue hotpluggable memory

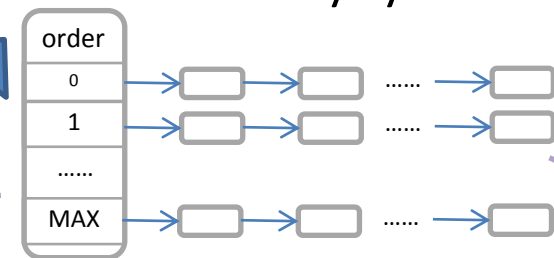
5. **Memory initialization** has been **finished**.

Boot time



Free hotpluggable memory to buddy system.

Buddy system



memblock.reserve[]

Agenda

1. Why need Memory Hot-Plug
2. ACPI & Memory Hot-Plug
3. Memory hot-add
4. Memory hot-remove
5. Movable node
6. Bootmem handling
7. Future work

Future work

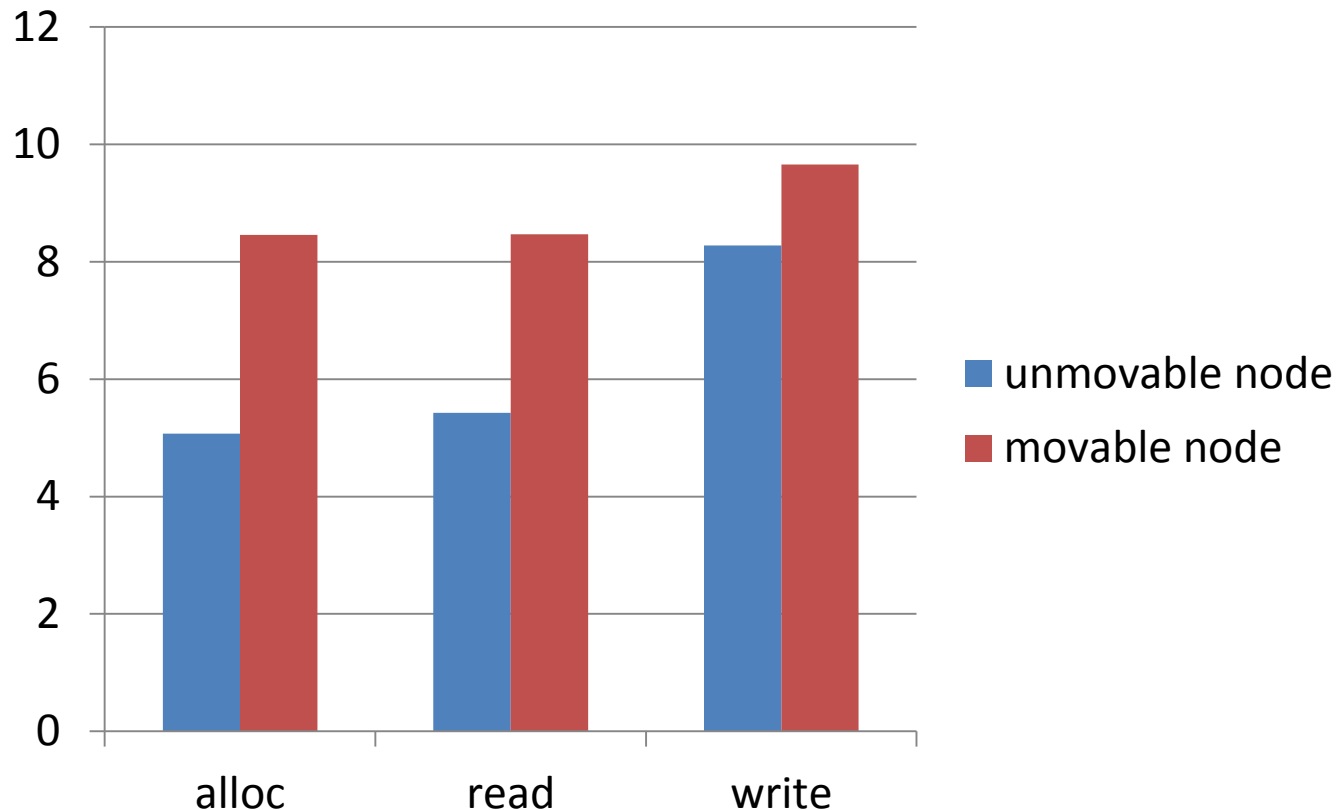
- Node local pagetable and vmemmap.
 - Improve performance.
- User space tools, like libnuma and numactl.
 - A library of functions.
 - Commands.

Thank you!
Q&A

Movable node

- Performance tests

Time of accessing 20GB memory (s)



Alloc: 40% down

Read: 33% down

Write: 12% down