CFQ vs Containers

2008/2/26

NTT Open Source Software Center
Fernando Luis Vázquez Cao
Hiroaki Nakano

Agenda

1. Block I/O resources and cgroups
2. Cgroups
3. I/O group scheduling
4. ioband

1 · Block I/O resources and cgroups

1.1. I/O bandwidth control: state of things

- State of things
  - CFQ's IO priority is an attribute of a process so it affects all devices it sends I/O requests to
  - I/O priority can be set by PID, PGRP, or UID, but...
  - ...all the processes that fall within the same class/priority are scheduled together

1.2. I/O bandwidth control: goals

- Goals
  - Being able to define arbitrary groupings of processes and...
  - ...treat each group as a single scheduling entity
  - Provide (soft) data rate guarantees
  - Perform I/O bandwidth control independently on each device
  - Scheduler-independent I/O bandwidth control
  - Usable even when the generic make_request_fn function is not used

1.3. I/O bandwidth control

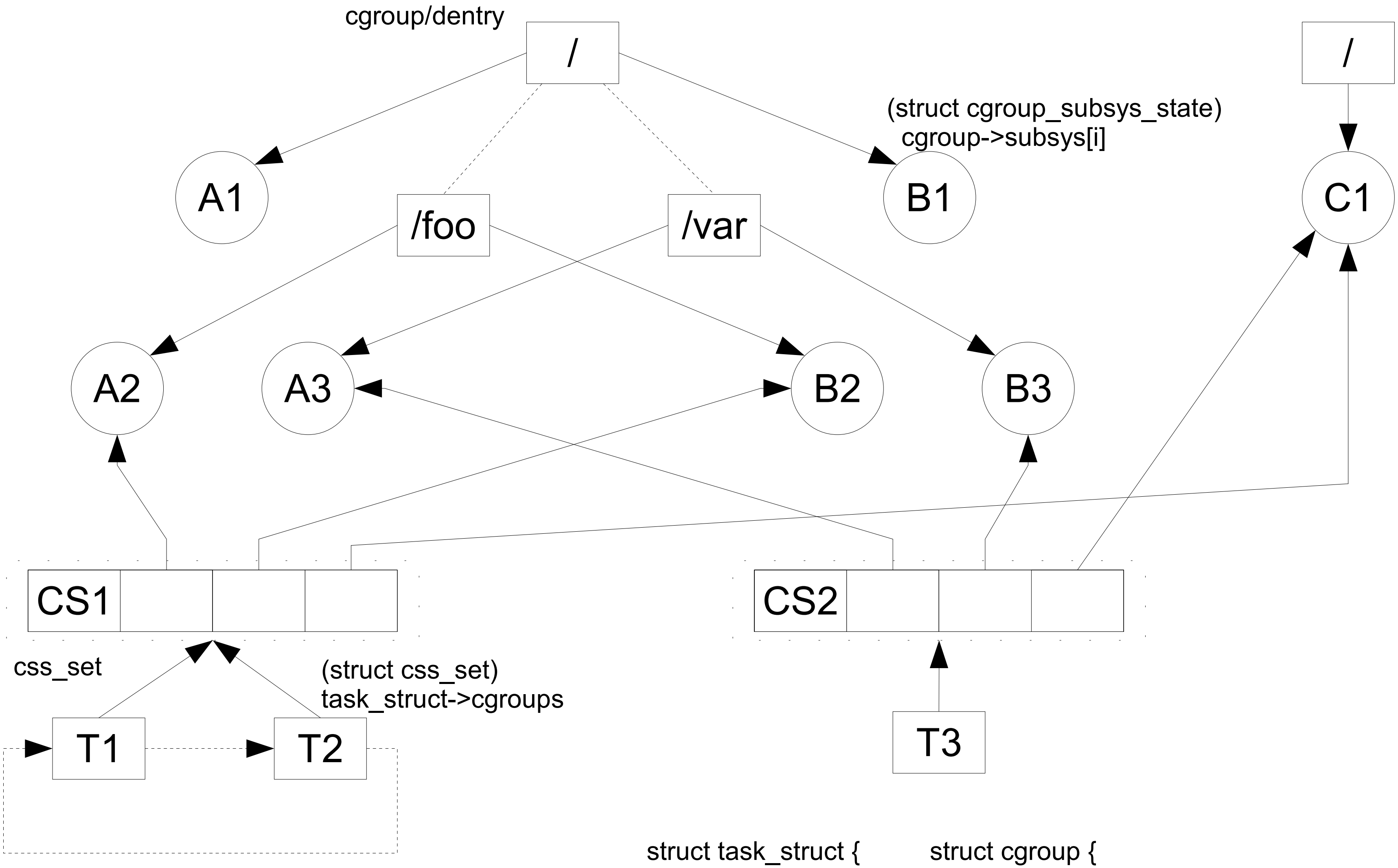
- What kind of things can be done?
  - I/O prioritization
    - ionice-like approach
  - Proportional bandwidth scheduling
    - Each process/group of processes has a weight that determines the share of bandwidth they receive
  - I/O limiting
    - Set an upper limit to the bandwidth a group of tasks can use

2 · Cgroups

2.1. Cgroups

- Subsystem/controller
  - Is a part of the kernel, commonly a system resource, which might have an interest in what a group of processes are doing
- Cgroup
  - Is a group of processes that share a set of parameters used by one or more subsystems
  - Characteristics
    - Cgroups are hierarchical
    - Each cgroup hierarchy is controlled through a cgroup filesystem whose tree of directories follows the structure of the cgroup hierarchy

2.2. Cgroups internals



cg_list (anchored at css_set->tasks)

```
struct task_struct {
  ...
  css_set *cgroups;
  ...
}
```

```
struct cgroup {
  struct list_head sibling;
  struct list_head children;
  struct cgroup *parent;
  struct dentry *dentry;
  struct cgroup_subsys_state *subsys[];
  struct list_head css_sets;
}
```

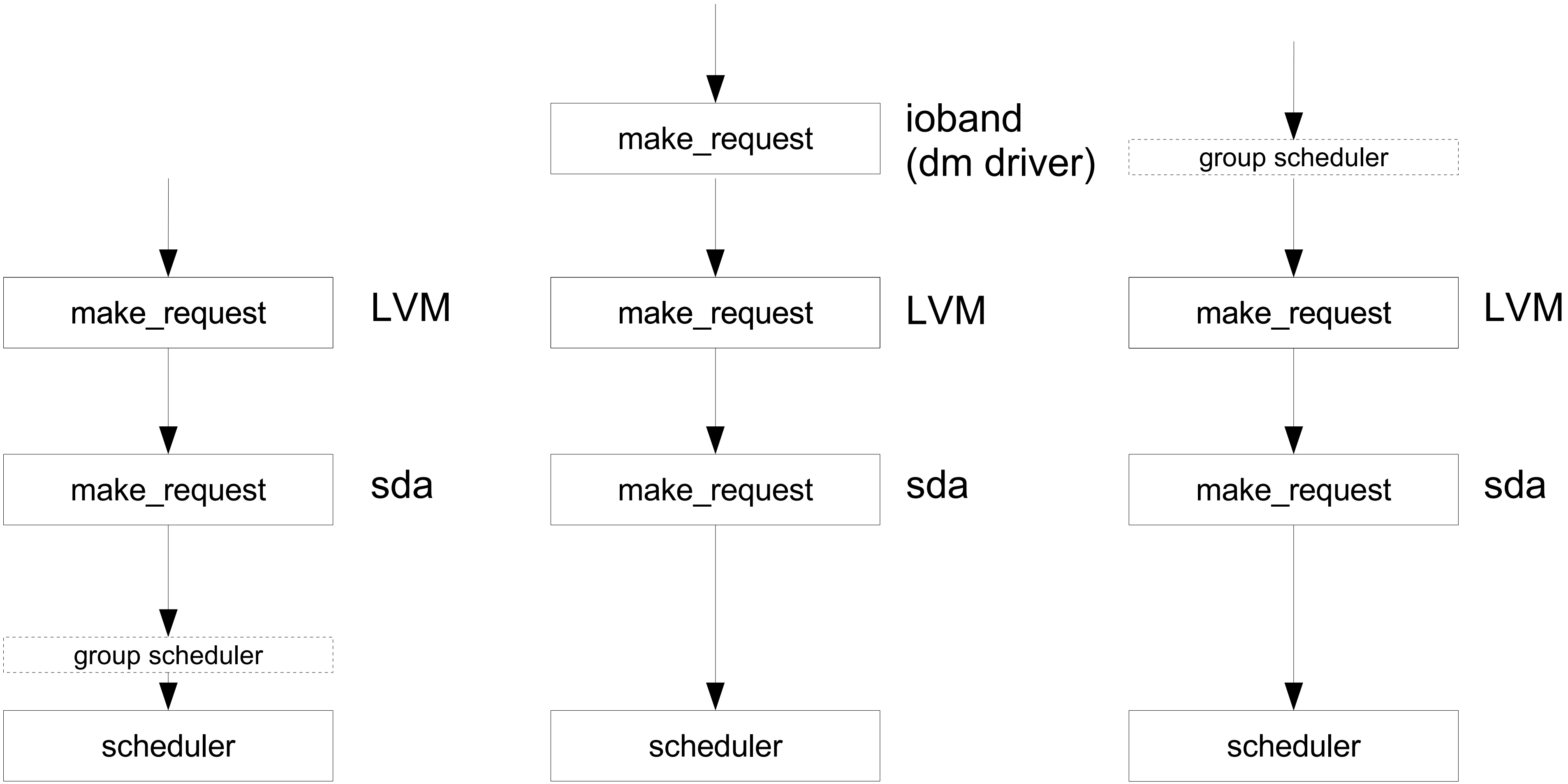
Original figure by Paul Menage

3 · I/O group scheduling

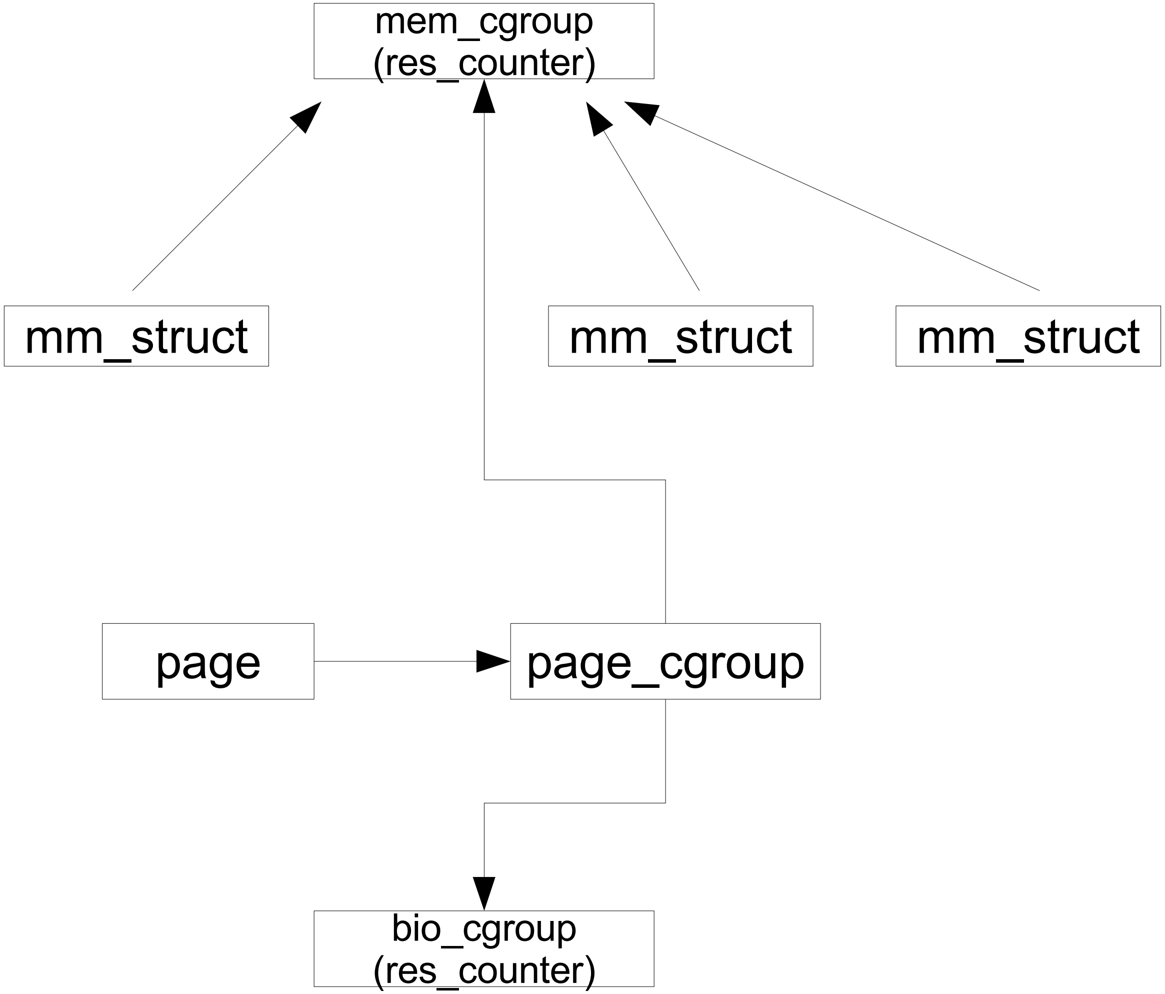
3.1. What do we need?

- Cgroups-aware I/O scheduling
- I/O tracking
- Group scheduling algorithm

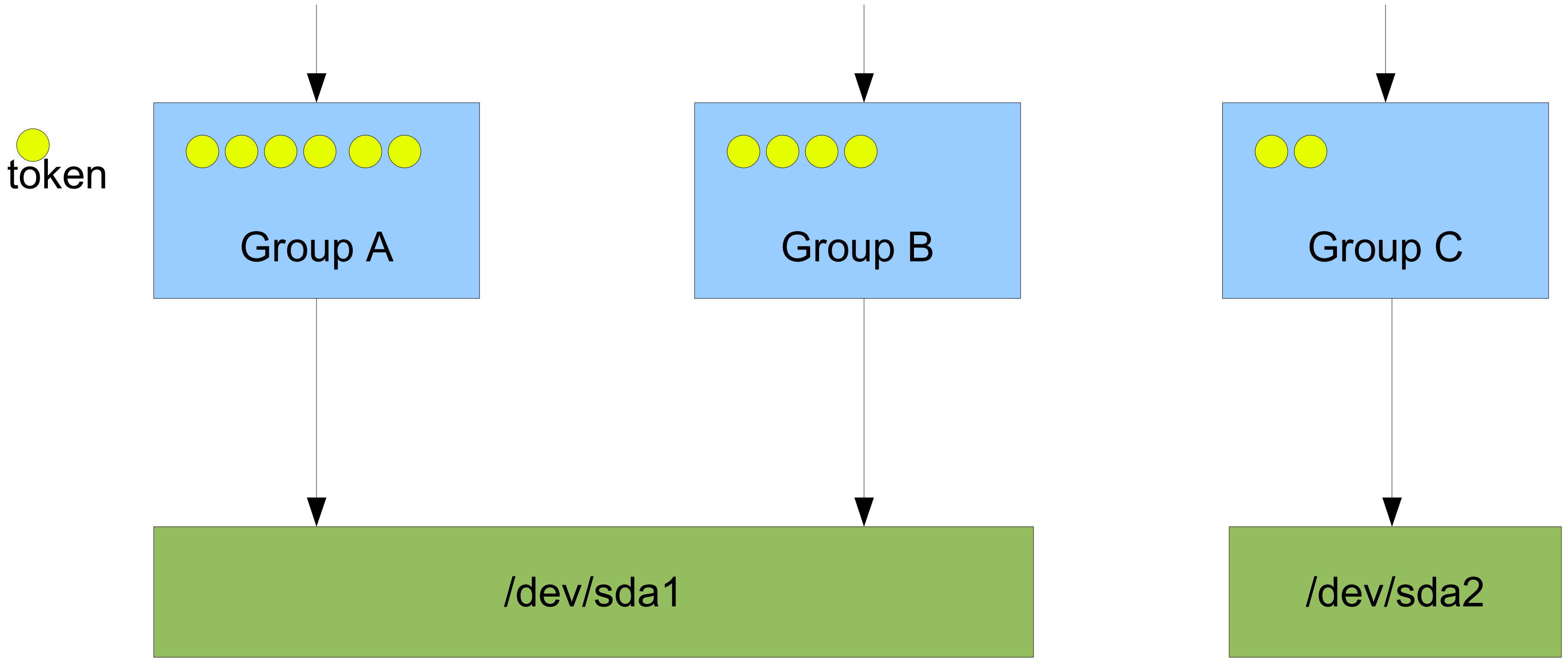
3.2. Cgroups-aware I/O scheduling



3.3. Tracking I/O



3.4. I/O group scheduler: algorithm

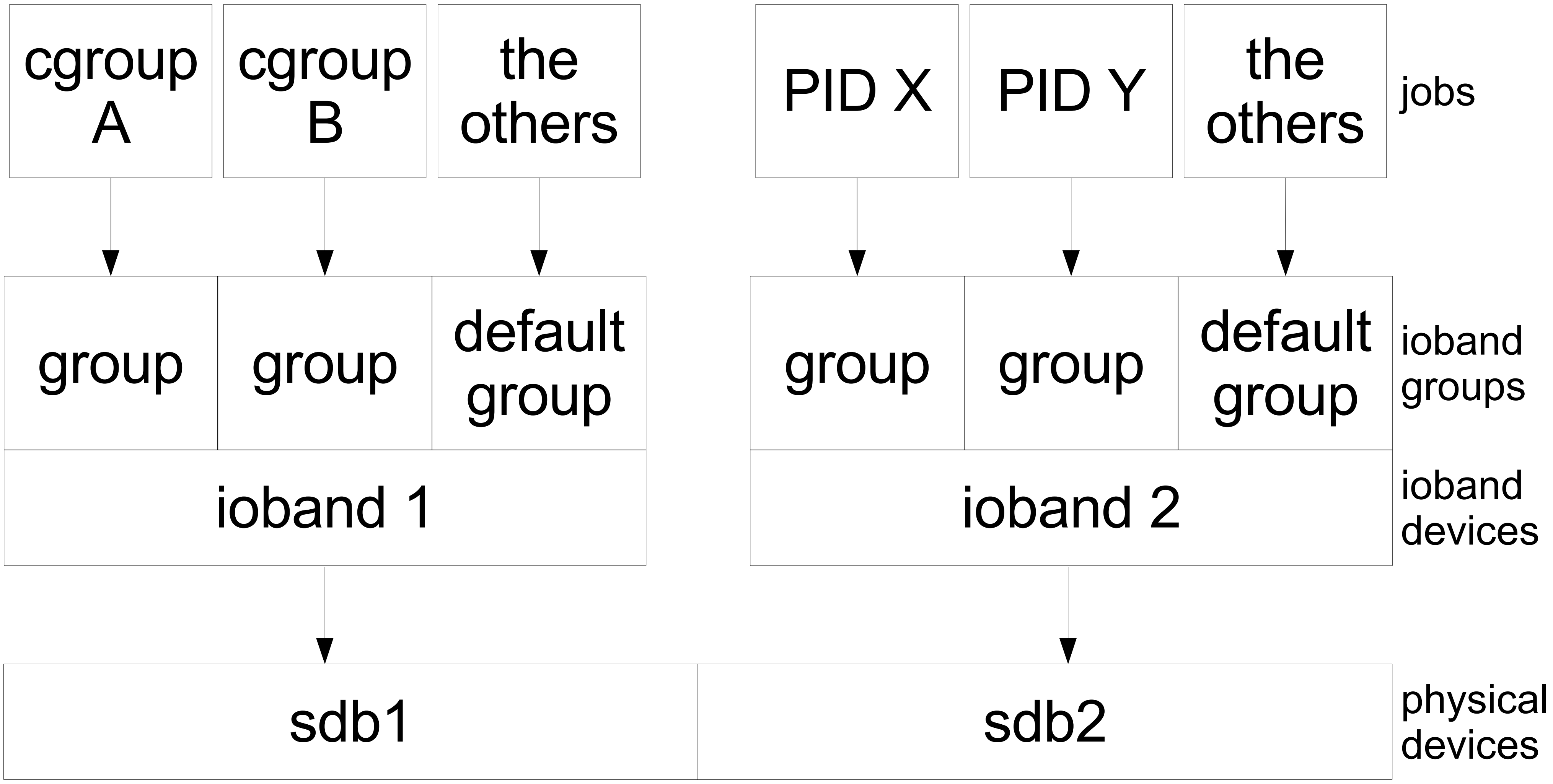


3 : 2 : 1

4．ioband

4.1. dm-ioband

- I/O bandwidth controller implemented as a device-mapper driver
- Bandwidth assigned according to the relative weight of each job



4.2. dm-io band – pros and cons

- Pros
  - Works with any I/O scheduler
    - This is a direct consequence of using a dm driver for the implementation
  - Each device can be configured independently
    - As opposed to CFQ's I/O priority which affects all I/O generated by a process
- Cons
  - Overkill?

la fin

Thanks for your attention

Contact: fernando@oss.ntt.co.jp

nakano.hiroaki@oss.ntt.co.jp