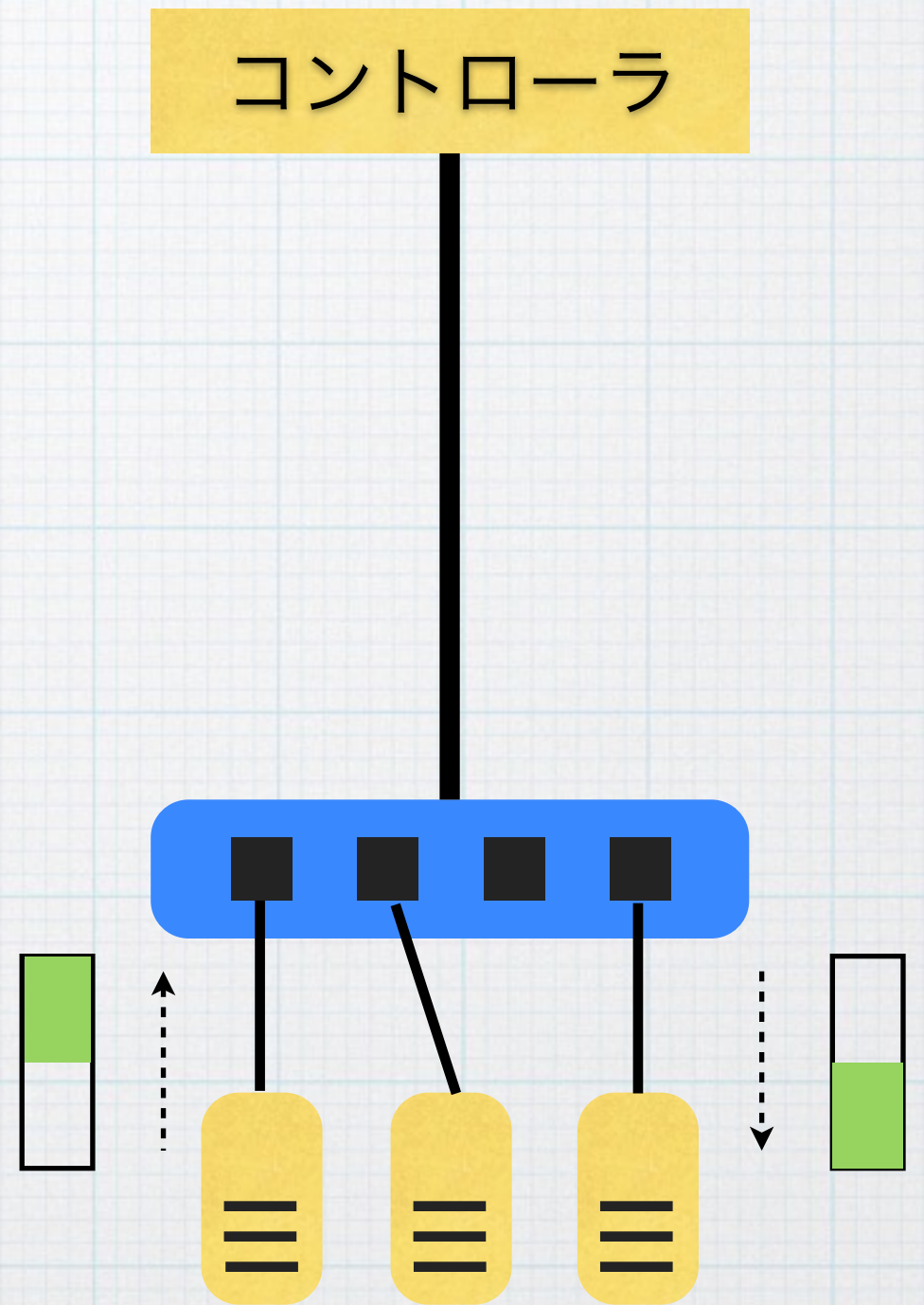
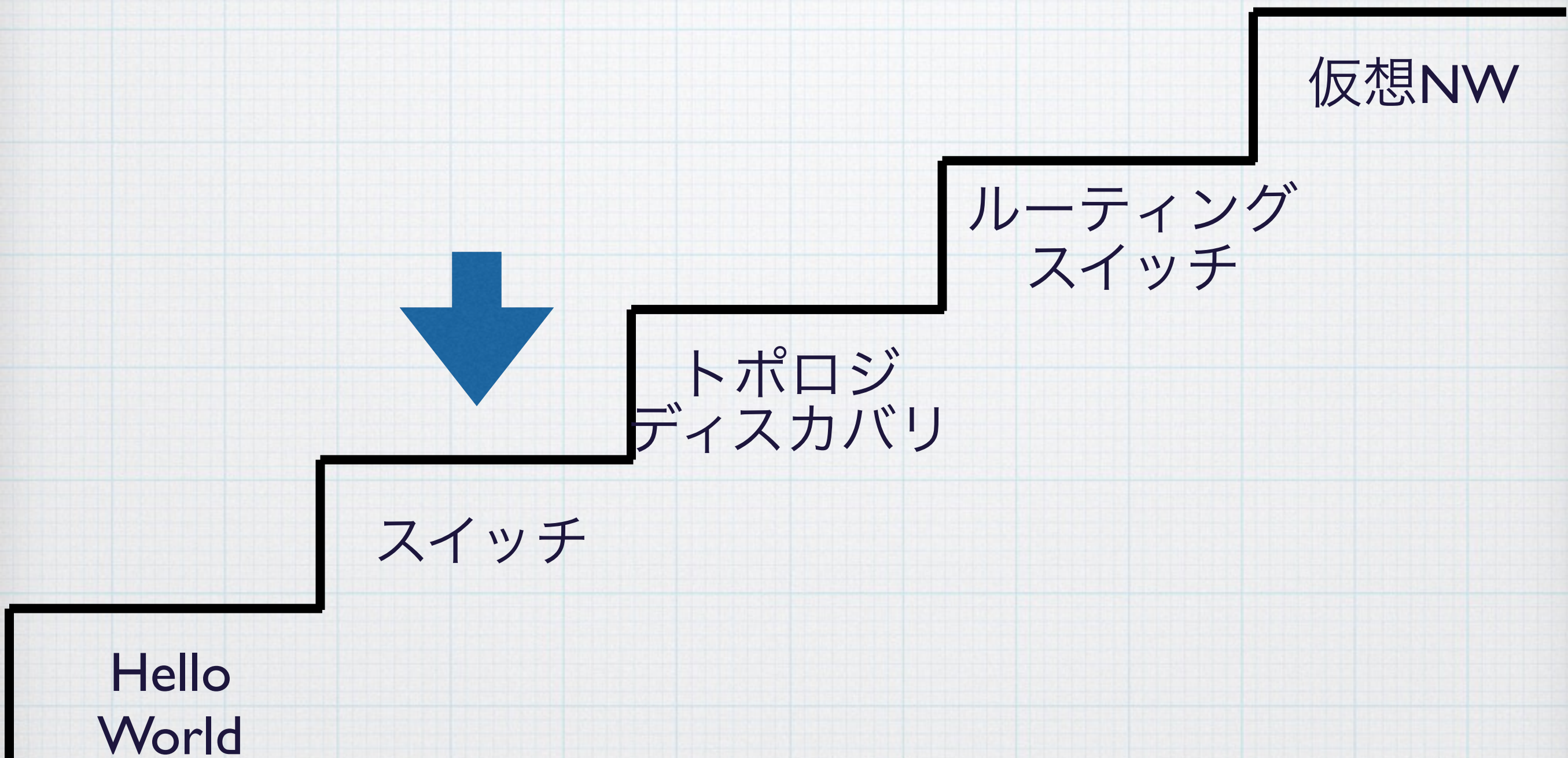


スイッチを作ろう

OpenFlow1.3編

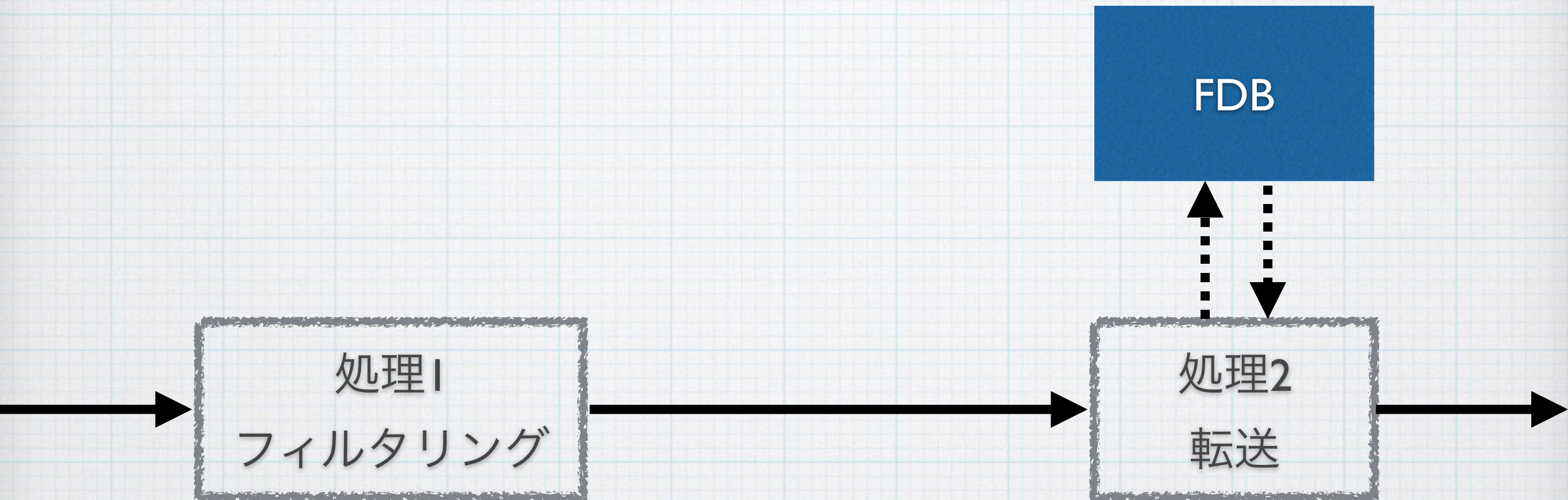




おさらい

OpenFlow1.0でのパケット処理

ラーニングスイッチ



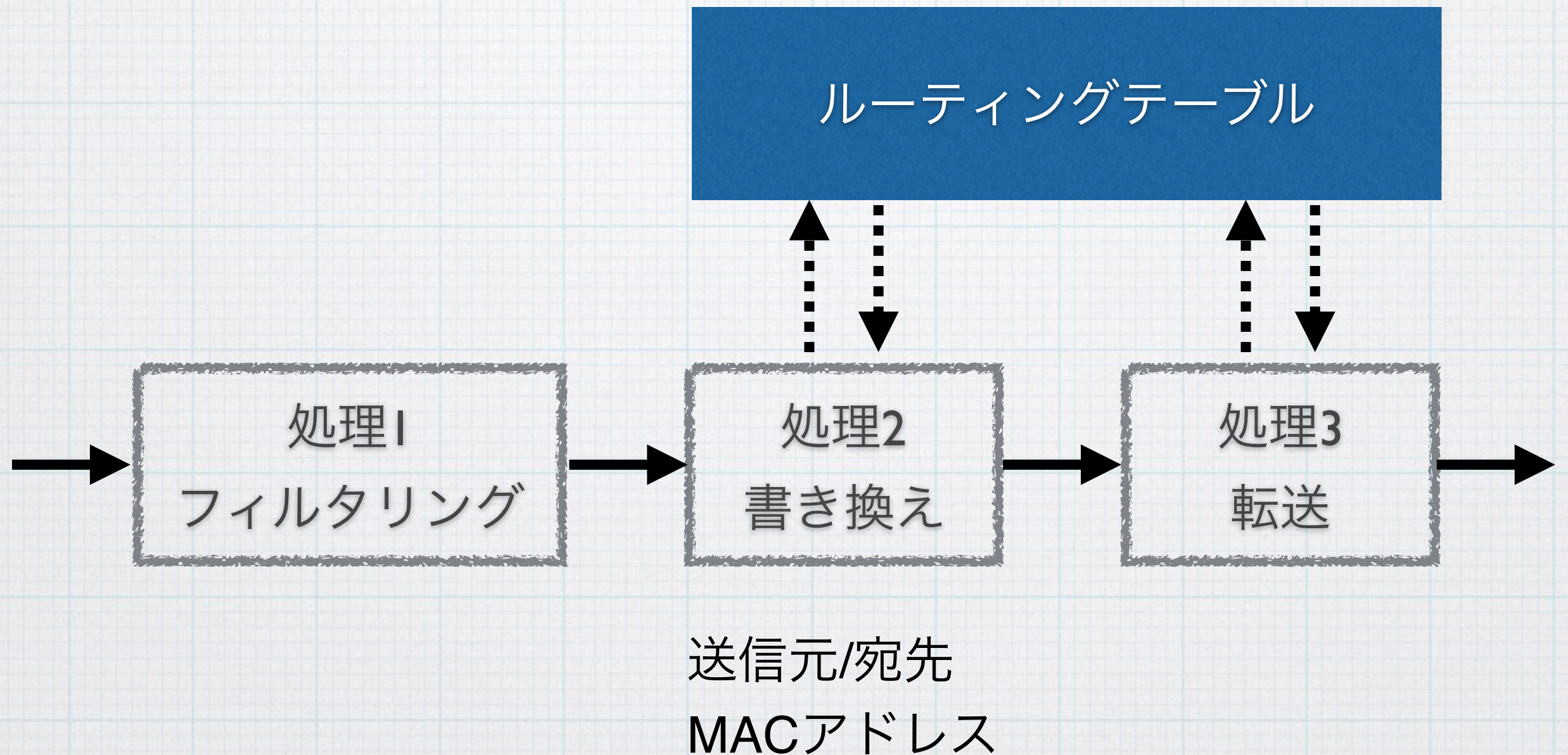
宛先 MAC =

- 802.1D/802.1Q reserved MAC
- マルチキャスト

をドロップ

ポートn番へ
or
FLOODING

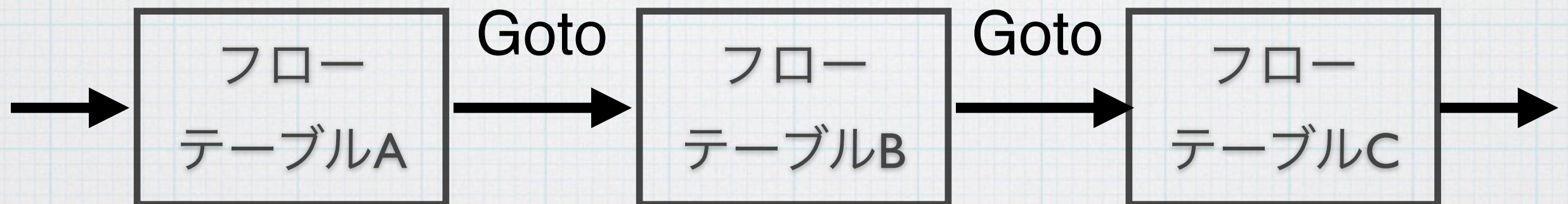
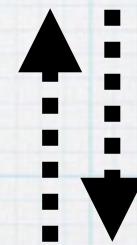
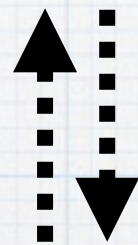
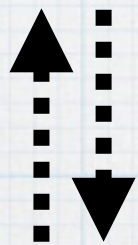
例: ルータ



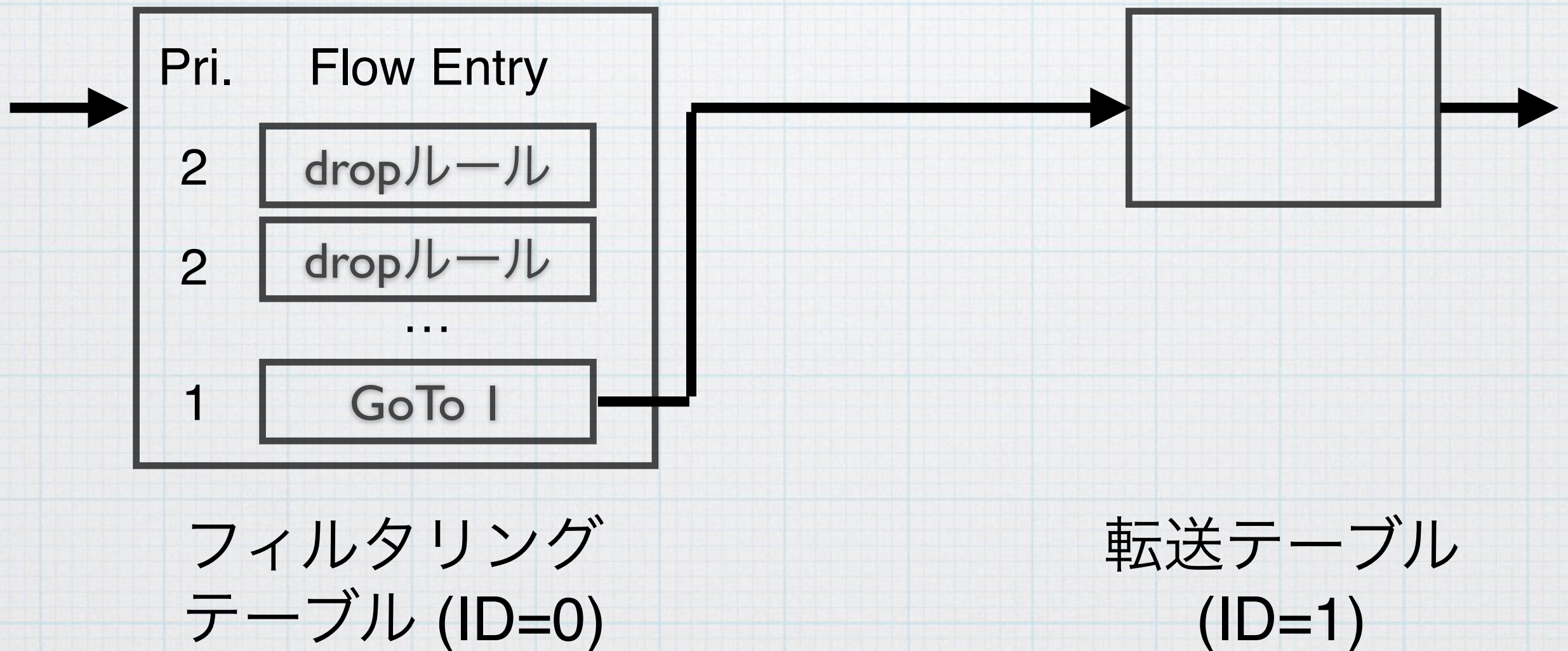
OpenFlow1.3でのパケット処理

マルチプルテーブル

コントローラ



ラーニングスイッチ



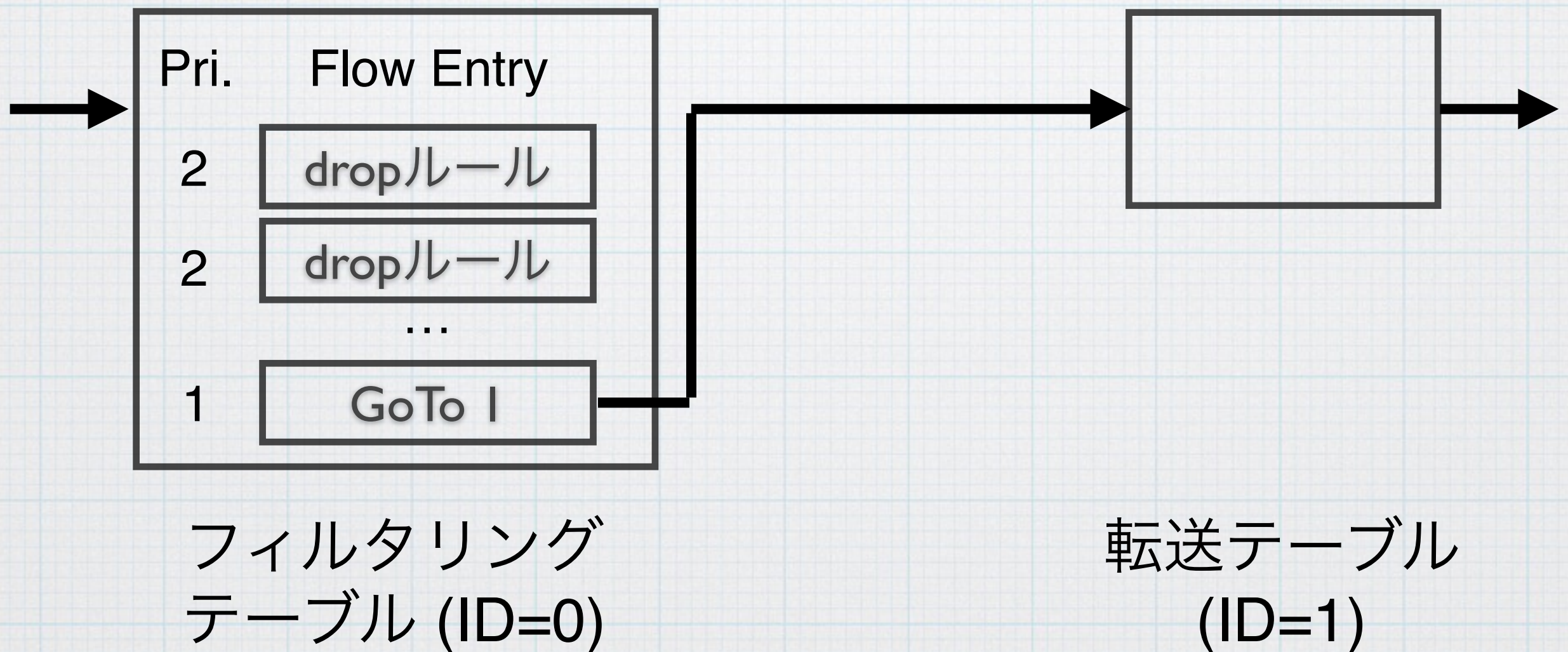

```
send_flow_mod_add(  
    datapath_id,  
    table_id: 0,  
    idle_timeout: 0,  
    priority: 2,  
    match: Match.new(ether_destination_address: '01:00:5e:00:00:00',  
                     ether_destination_address_mask: 'ff:ff:ff:00:00:00')  
)
```

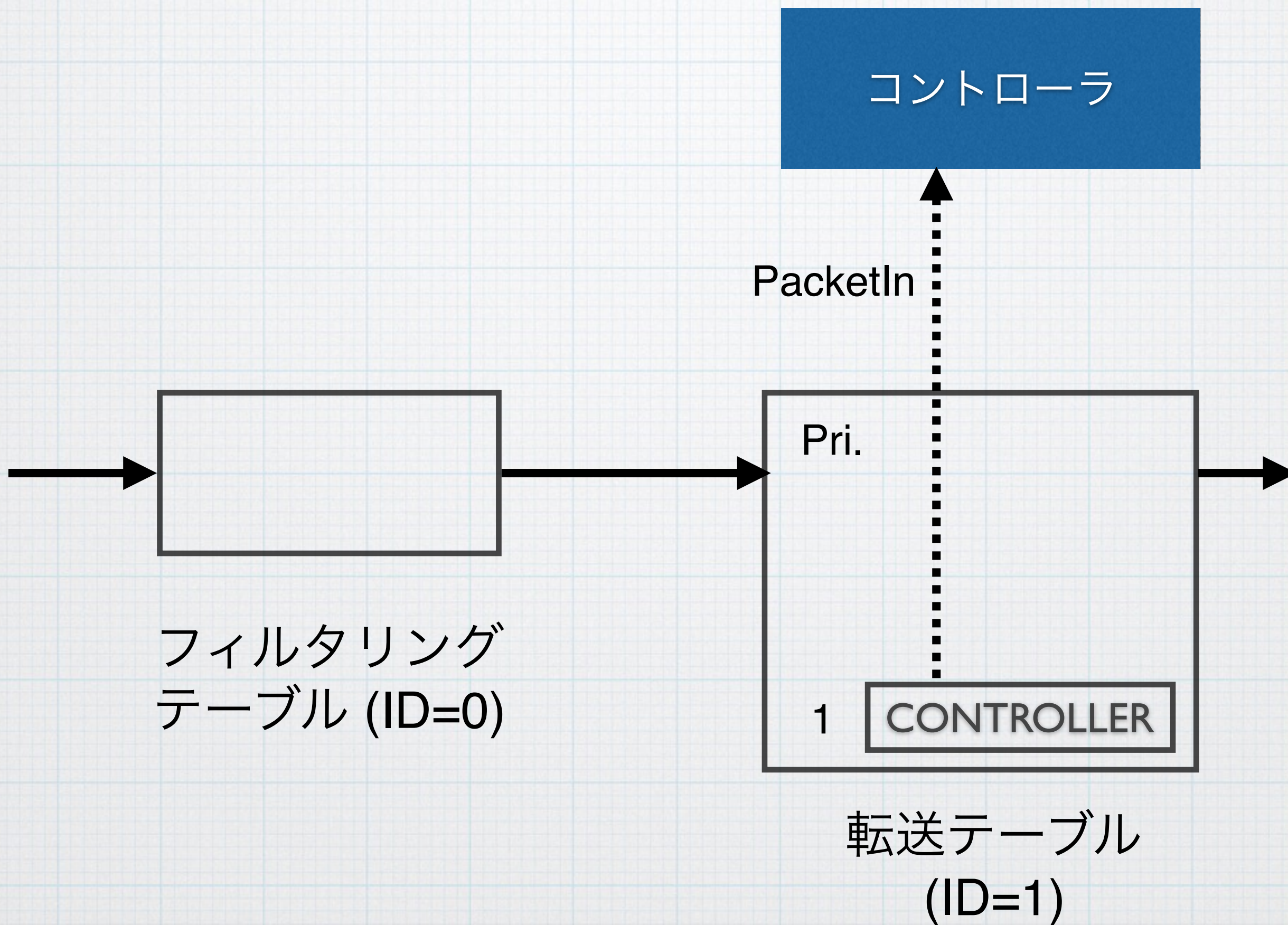
- ・マルチキャストを落とす
- ・最初のテーブル (ID=0) で処


```
send_flow_mod_add(  
    datapath_id,  
    table_id: 0,  
    idle_timeout: 0,  
    priority: 1,  
    match: Match.new,  
    instructions: GotoTable.new(1)  
)
```

- ・ 処理を先にかかっていたら、
- ・ フィルタリング (pri=2) の後
このルール (pri=1) にヒット

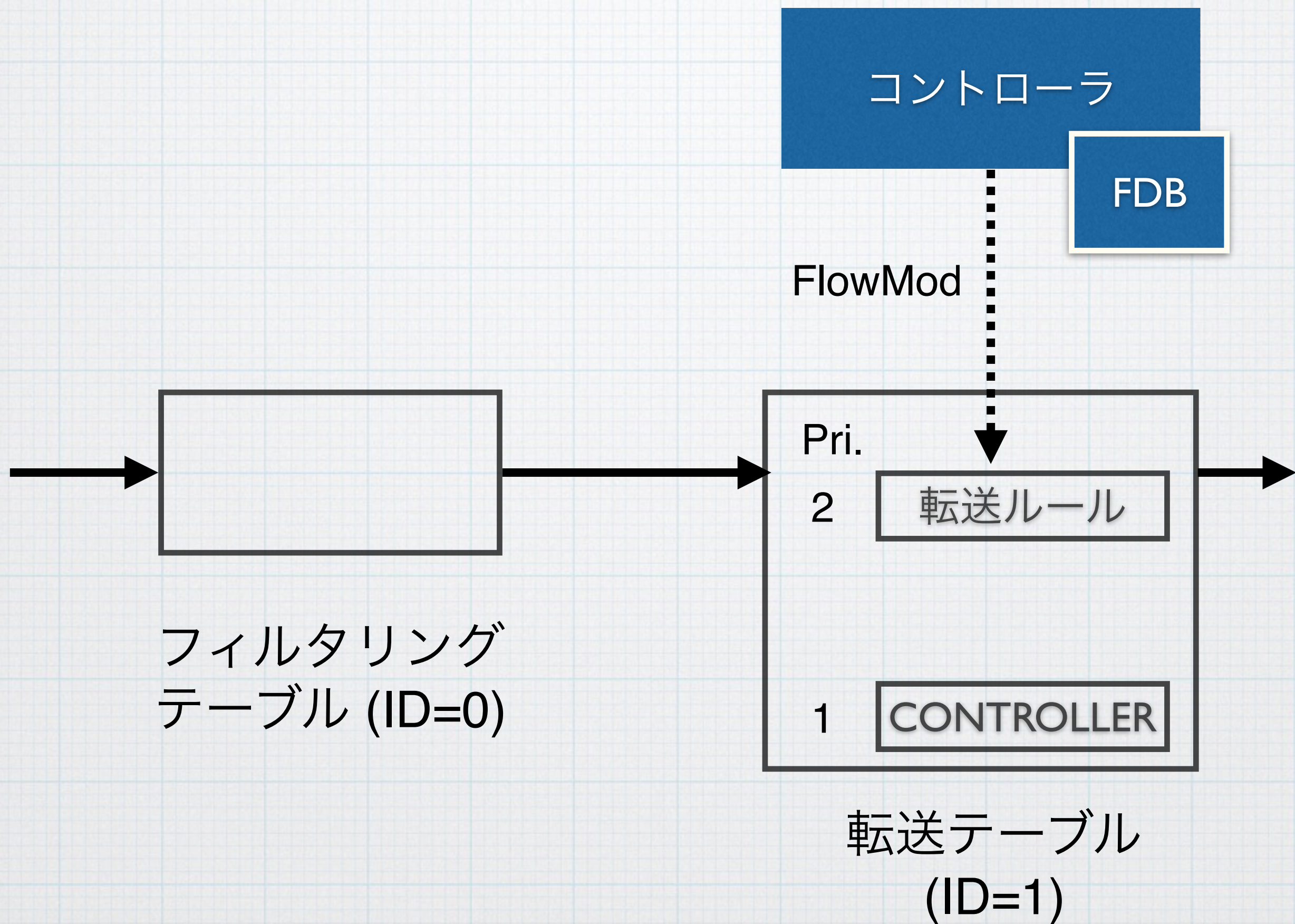
ラーニングスイッチ(再掲)






```
send_flow_mod_add(  
    datapath_id,  
    table_id: 1,  
    idle_timeout: 0,  
    priority: 1,  
    match: Match.new,  
    instructions: Apply.new(SendOutPort.new(:controller))  
)
```

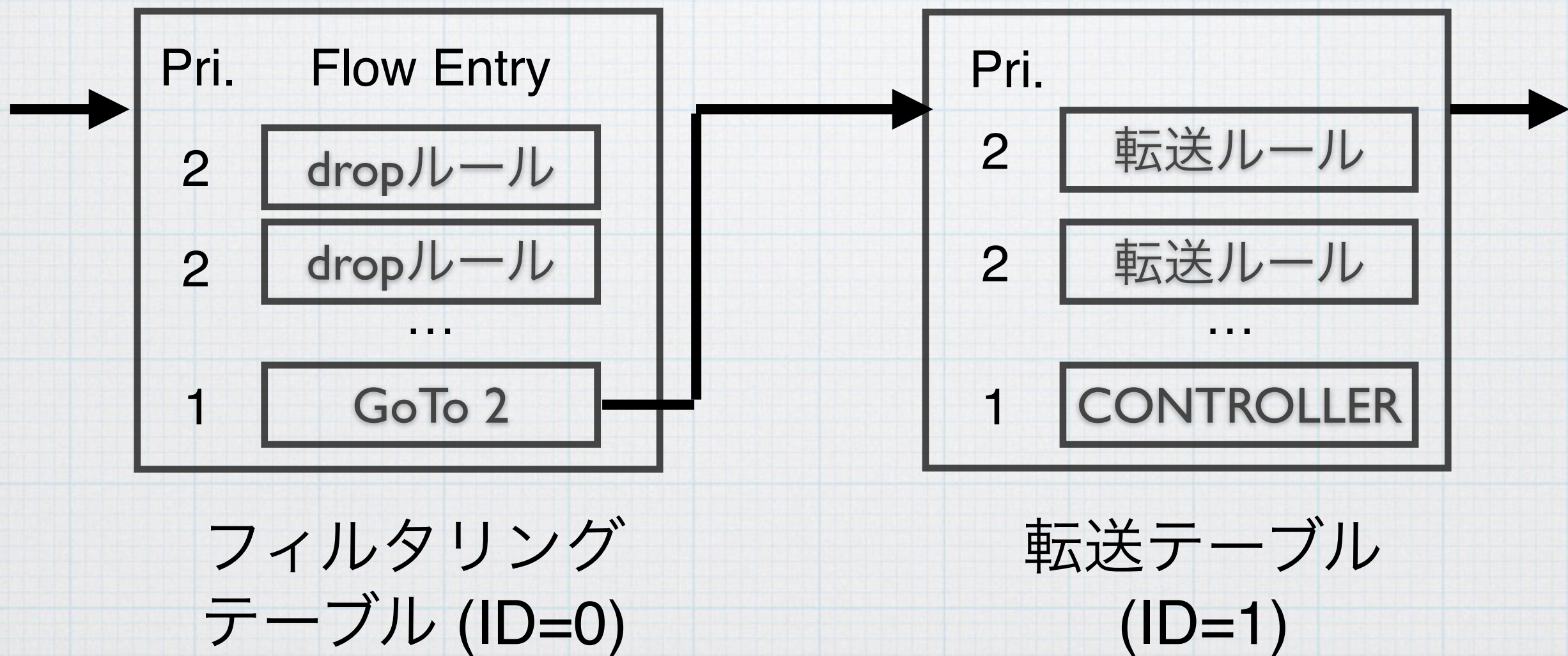
- ・明示的にPacketInを起こす




```
send_flow_mod_add(  
    datapath_id,  
    table_id: FORWARDING_TABLE_ID,  
    idle_timeout: AGING_TIME,  
    priority: 2,  
    match: Match.new(in_port: packet_in.in_port,  
                     ether_destination_address: packet_in.destination_mac,  
                     ether_source_address: packet_in.source_mac),  
    instructions: Apply.new(SendOutPort.new(port_no))  
)
```

- ・ packet_inの送信元+ポートをフローエントリとして追加
- ・ 優先度はPacketInより高い

コントローラ



OpenFlow1.3の利点

OpenFlow 1.0の問題点 I

Flow Table

dropルール1

dropルール2

dropルール3

..

書き換えルール1

書き換えルール2

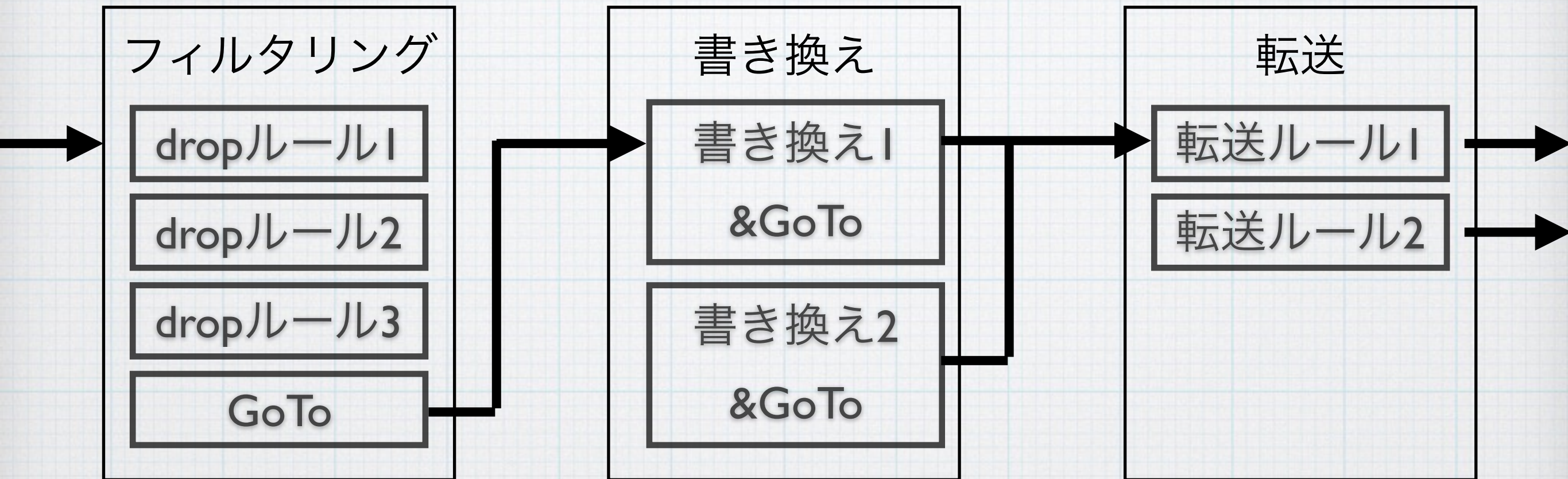
..

転送ルール1

転送ルール2

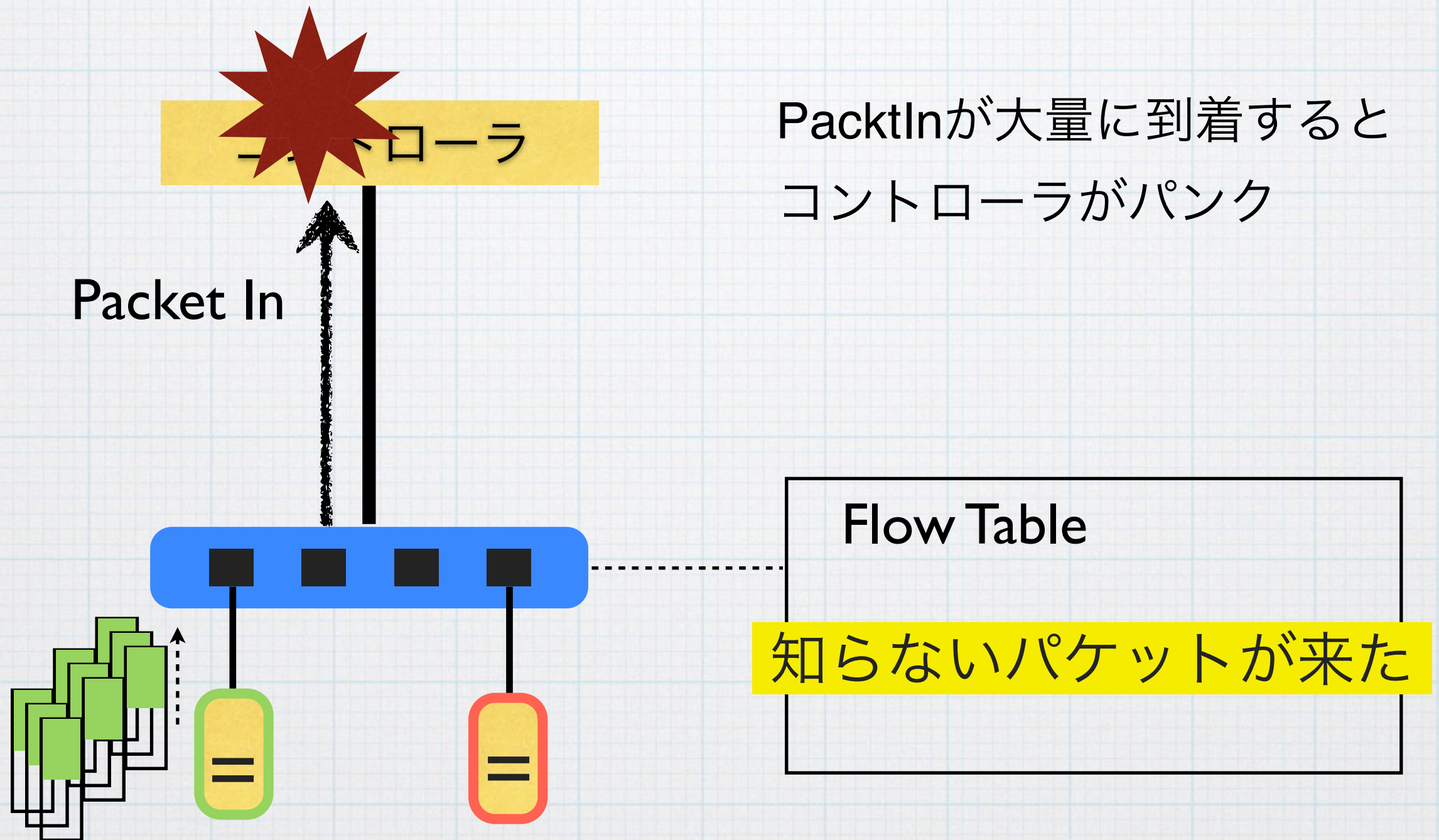
- OpenFlow 1.0 では、フローテーブルは一つだけ
- 役割の違うエントリが混在 & 混乱しがち

OpenFlow 1.3では

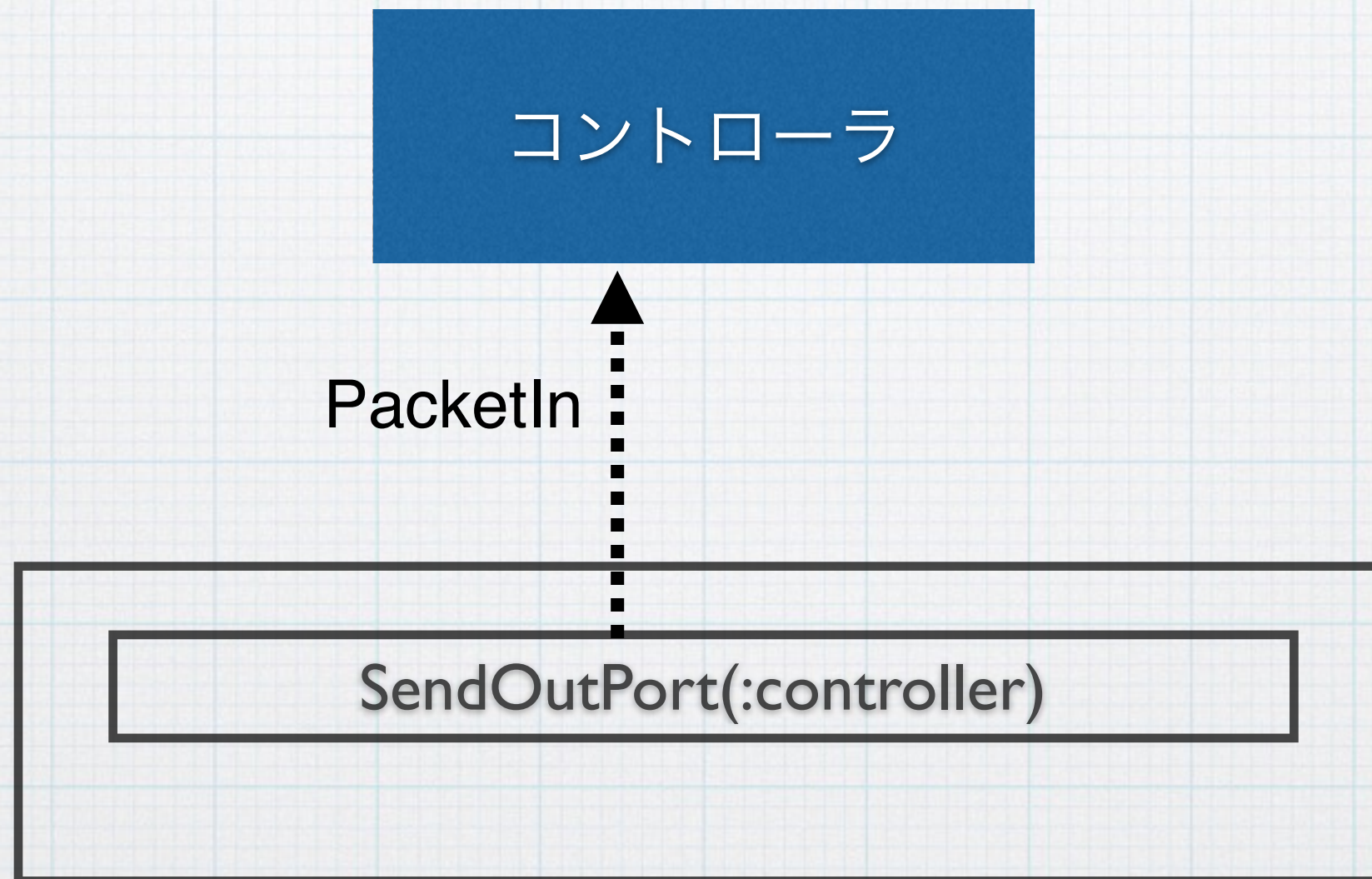


- 分けてスッキリ!
- 処理をパイプライン化できるので性能が上がる

OpenFlow 1.0の問題点 2



OpenFlow 1.3では



- 明示的に指定しない限り、PacketInは起こらない

アクションと インストラクション

- アクション
 - パケットの書き換えと転送 (OFI.0と同じ)
 - OFI.3で種類が増えました
- インストラクション
 - テーブルの移動 (GoTo)
 - アクションの実行方法の指定

```
send_flow_mod_add(  
    datapath_id,  
    table_id: 1,  
    idle_timeout: 0,  
    priority: 1,  
    match: Match.new,  
    instructions: GotoTable.new(2)  
)
```

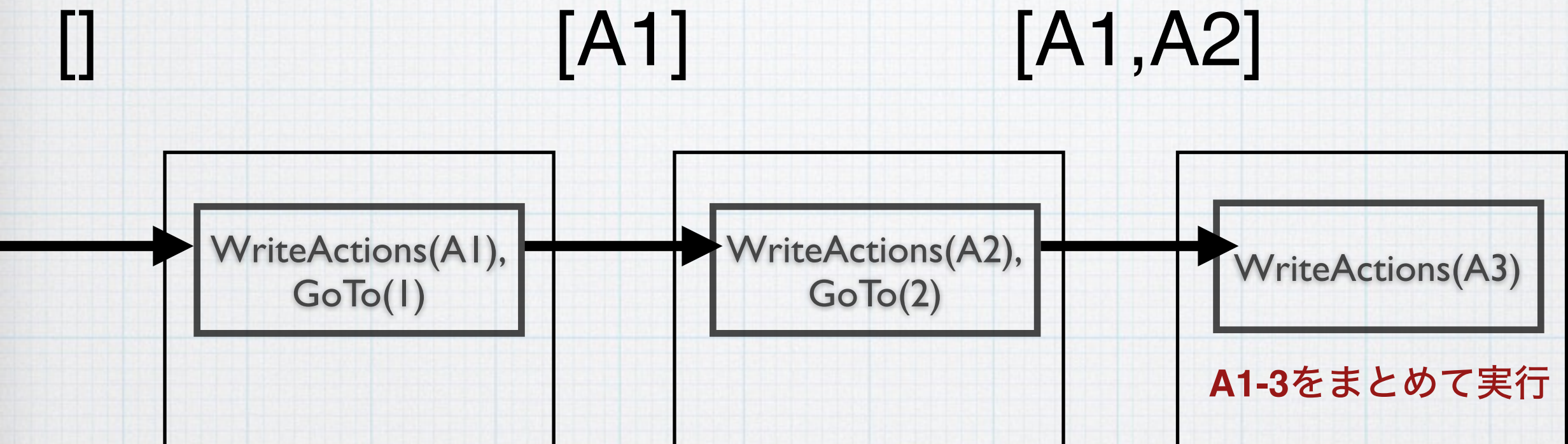
- ・ 処理をテーブル2へ移行


```
send_flow_mod_add(  
    datapath_id,  
    ...  
    actions: SendOutPort.new(1)  
)
```



```
send_flow_mod_add(  
    datapath_id,  
    ...  
    instructions: Apply.new(SendOutPort.new(1))  
)
```

アクションセット



- パケットに関連付いたアクション集合
- WriteActionsでアクションを追加
- GoToを含まないエントリにマッチしたら実行

まとめ

- OpenFlow 1.3 版スイッチの仕組み
- マルチプルテーブルの使い方
- アクションとインストラクション

レポート課題

- OpenFlow1.3 版スイッチの動作について、
`trema dump_flows` の出力を混じえながら
各ステップごとに説明しよう
- 実行のしかた

```
$ trema run lib/learning_switch13.rb  
  --openflow13 -c trema.conf
```