1 問題

複数スイッチ対応版（multi_learning_switch.rb）の動作解説

2 解答

2.1 コードの解説

multi_learning_switch.rb のメソッド毎にコメントとしてコード（1）の解説を追記する．また，コード中に追記された logger.info メソッドは動作確認のために挿入されたものである

```
1  $LOAD_PATH.unshift __dir__
2
3  require 'fdb'
4
5  class MultiLearningSwitch < Trema::Controller
6    timer_event :age_fdbs, interval: 5.sec
7
8  # Trema の開始時に全ての FDB (Forwarding DB)を管理する連想配列を作成
9  # 開始メッセージを表示
10   def start(_argv)
11     @fdbs = {}
12     logger.info 'MultiLearningSwitch␣started.'
13   end
14
15 # 各スイッチ（lsw1-4）が接続されたらそれぞれについて FDB を作成
16 # 各 FDB の識別はスイッチの ID で行う
17   def switch_ready(datapath_id)
18     @fdbs[datapath_id] = FDB.new
19   end
20
21 # Packet In が発生したら呼ばれるメソッド
22 # 宛先が 802.1D 予約済みMACアドレスである場合は何もしない
23 # Packet In が発生したスイッチの FDB を取得し，
24 # (宛先アドレス，転送先ポート) =
25 # (Packet In の送信元アドレス，Packet In の受信ポート)を記録・更新する
26 # 最後に flow_mod_and_packet_out メソッドを実行
27   def packet_in(datapath_id, message)
28     return if message.destination_mac.reserved?
29     logger.info "スイッチ␣#{datapath_id}␣にて␣Packet\_In␣が発生しました"
30     @fdbs.fetch(datapath_id).learn(message.source_mac, message.in_port)
31     flow_mod_and_packet_out message
32   end
```

```
33
34  # 5 秒毎に実行されるメソッド
35  # 有効期限が切れた FDB のエントリがあれば削除する
36    def age_fdbs
37      @fdbs.each_value(&:age)
38    end
39
40    private
41
42  # Packet In の宛先 Mac アドレスから FDB により転送先ポート番号を得る
43  # 転送先ポートが見つかれば flow_mod を実行
44  # さらに転送先ポートが見つかればそのポートからパケットを転送し，
45  # 転送先ポートが見つからなければフラッディングする（packet_out で処理）
46    def flow_mod_and_packet_out(message)
47      port_no = @fdbs.fetch(message.dpid).lookup(message.destination_mac)
48      flow_mod(message, port_no) if port_no
49      packet_out(message, port_no || :flood)
50    end
51
52  # Packet In の転送先ポートが見つかった場合は
53  # 今後同じパケットをスイッチのみで同様に処理できるよう
54  # send_flow_mod_add を実行する
55    def flow_mod(message, port_no)
56      send_flow_mod_add(
57        message.datapath_id,
58        match: ExactMatch.new(message),
59        actions: SendOutPort.new(port_no)
60      )
61      logger.info "スイッチ␣#{message.datapath_id}␣の␣FlowTable␣にルールを追加しました"
62    end
63
64  # Packet\_In を指定のポート（port_no）から送信するメソッド
65  # port_no に :flood が指定された場合はフラッディングする
66    def packet_out(message, port_no)
67      send_packet_out(
68        message.datapath_id,
69        packet_in: message,
70        actions: SendOutPort.new(port_no)
71      )
72      logger.info "スイッチ␣#{message.datapath_id}␣のポート␣#{port_no}␣からパケットを送信しました"
73    end
74  end
```

ソースコード 1　multi_learning_switch.rb

2.2　動作確認

以下の順にパケットを転送し，その際のログメッセージと動作概要を示す．trema.multi.conf に記述されているトポロジは図 1 の通りである．FDBn，FTn は スイッチ n における FDB，FlowTable を意味する（FDB についてはコントローラで実装されている）．

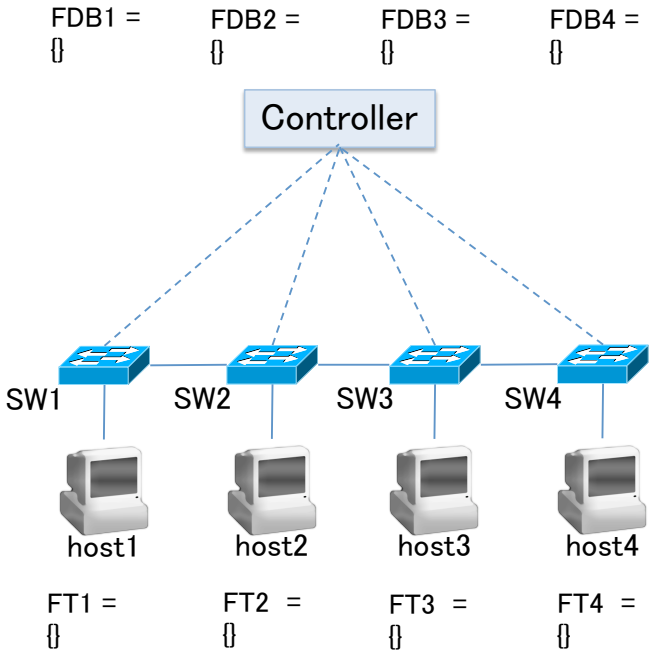

図 1　Example のトポロジ

2.2.1　(1) host1 から host2 へパケットを送る

ログメッセージ

```
スイッチ 1 にて Packet\_In が発生しました
スイッチ 1 のポート flood からパケットを送信しました
スイッチ 2 にて Packet\_In が発生しました
スイッチ 2 のポート flood からパケットを送信しました
スイッチ 3 にて Packet\_In が発生しました
スイッチ 3 のポート flood からパケットを送信しました
```

```
スイッチ 4 にて Packet\_In が発生しました
スイッチ 4 のポート flood からパケットを送信しました
```

動作概要

host1 から host2 宛てのパケットが送られると，SW1 にて Packet In が生じる．コントローラはパケットの送信元情報から SW1 の FDB に host1 宛てのエントリを生成し，そして host2（の MAC アドレス）宛てのエントリは SW1 の FDB に登録されていないため，SW1 に当該パケットをフラッディングさせる．その結果，SW2，SW3，SW4 についても同様の処理が生じる．host2 以外のホストは当該パケットの宛先が自分宛てでないため当該パケットを破棄する（図 2）．

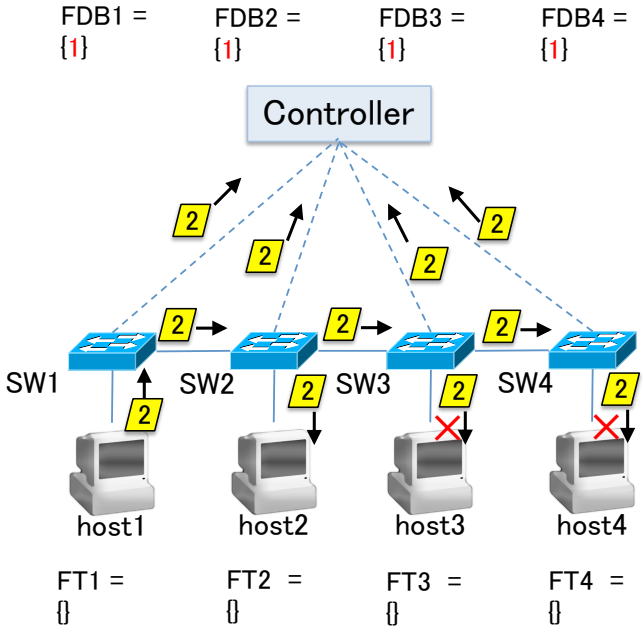

図 2　host1 から host2 へパケットを送る

2.2.2　(2) host2 から host1 へパケットを送る

ログメッセージ

```
スイッチ 2 にて Packet\_In が発生しました
スイッチ 2 の FlowTable にルールを追加しました
スイッチ 2 のポート 2 からパケットを送信しました
```

```
スイッチ 1 にて Packet\_In が発生しました
スイッチ 1 の FlowTable にルールを追加しました
スイッチ 1 のポート 1 からパケットを送信しました
```

動作概要

host2 から host1 宛てのパケットが送られると，SW2 にて Packet In が生じる．コントローラはパケットの送信元情報から SW2 の FDB に host2 宛てのエントリを生成し，そして host2 宛てのエントリは SW1 の FDB に登録されているため，SW2 に当該パケットを適切な転送先（SW1）に転送させる．このとき，同じパケットを今後スイッチで転送処理するため FlowTable に転送ルールを書き込む．SW1 についても同様であり，結果的にパケットは適切な宛先（host1）に到達する．（図 3）．

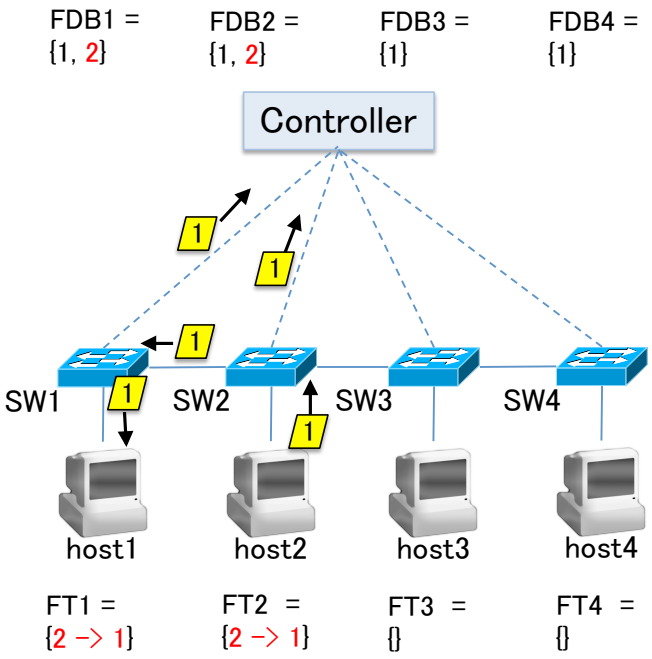

図 3　host2 から host1 へパケットを送る

2.2.3　(3) host1 から host2 へパケットを送る

ログメッセージ

```
スイッチ 1 にて Packet\_In が発生しました
```

スイッチ 1 の FlowTable にルールを追加しました
スイッチ 1 のポート 2 からパケットを送信しました
スイッチ 2 にて Packet_In が発生しました
スイッチ 2 の FlowTable にルールを追加しました
スイッチ 2 のポート 1 からパケットを送信しました

動作概要

(2) と同様である（図 4）.

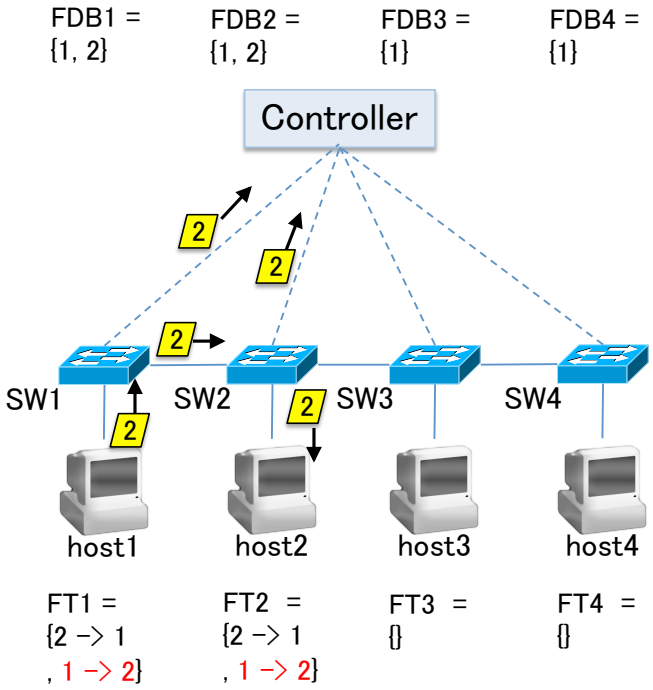

図 4　host1 から host2 へパケットを送る

2.2.4　(4) host1 から host2 へパケットを送る

ログメッセージ

（なし）

動作概要

SW1，SW2 の FlowTable に転送ルールが存在するため Packet In は生じない．したがってログメッセージは表示されず，スイッチによって当該パケットは適切に転送される（図 5）．

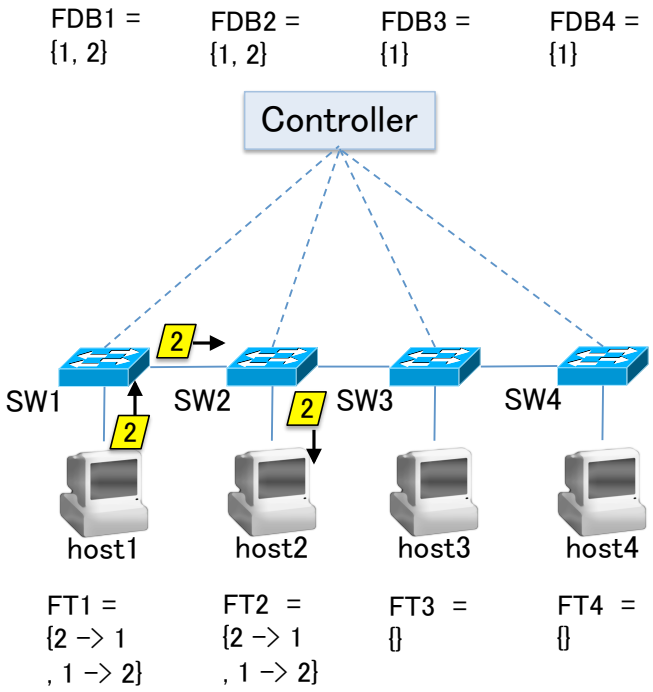

図 5　host1 から host2 へパケットを送る

2.2.5　(5) host2 から host3 へパケットを送る

ログメッセージ

```
スイッチ 2 にて Packet\_In が発生しました
スイッチ 2 のポート flood からパケットを送信しました
スイッチ 3 にて Packet\_In が発生しました
スイッチ 3 のポート flood からパケットを送信しました
スイッチ 1 にて Packet\_In が発生しました
スイッチ 1 のポート flood からパケットを送信しました
スイッチ 4 にて Packet\_In が発生しました
スイッチ 4 のポート flood からパケットを送信しました
```

動作概要

SW2，SW3 において FDB のエントリに宛先アドレスが存在しないため (1) と同様に動作する（図 6）.

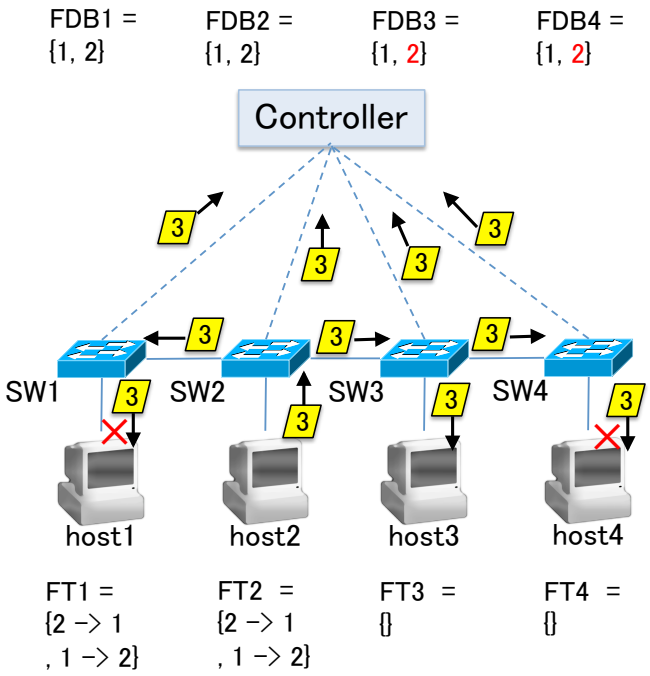

図 6　host2 から host3 へパケットを送る

2.2.6　(6) host4 から host2 へパケットを送る

ログメッセージ

```
スイッチ 4 にて Packet\_In が発生しました
スイッチ 4 の FlowTable にルールを追加しました
スイッチ 4 のポート 2 からパケットを送信しました
スイッチ 3 にて Packet\_In が発生しました
スイッチ 3 の FlowTable にルールを追加しました
スイッチ 3 のポート 2 からパケットを送信しました
スイッチ 2 にて Packet\_In が発生しました
スイッチ 2 の FlowTable にルールを追加しました
スイッチ 2 のポート 1 からパケットを送信しました
```

動作概要

SW4，SW3，SW2 において FDB のエントリに宛先アドレスが存在するため (2) と同様に動作する（図 7）.

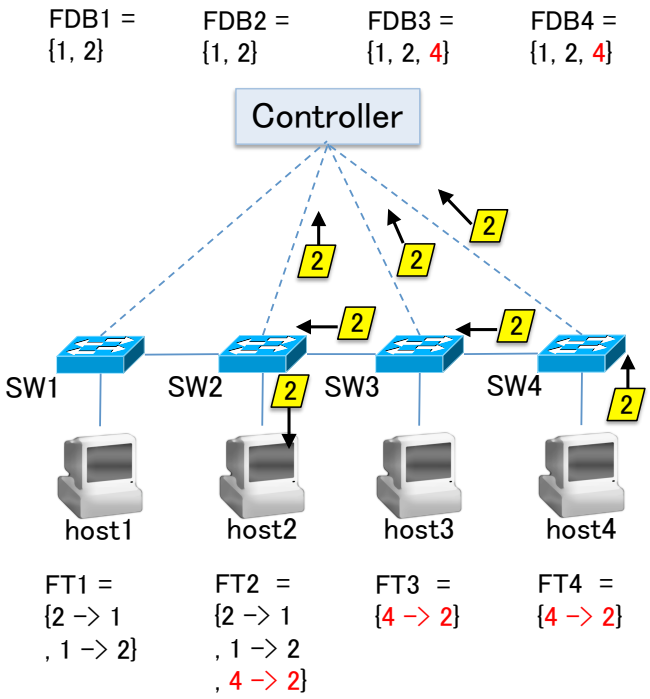

図 7　host4 から host2 へパケットを送る