**Lab 05: Xây dựng bảng định tuyến cho Fat Tree và ngẫu nhiên cặp nguồn đích**

*Notes:*

*+ Có deadline*

*+ SV phải chụp lại các bức ảnh để minh chứng việc mình hoàn thành các bài tập trong từng Lab*

*+ Ghi câu trả lời (và cả ảnh chụp ra file pdf hoặc doc) rồi gửi đến địa chỉ email: nguyenthanh@soict.hust.edu.vn, với tiêu đề email và tên file như sau (chẳng hạn bạn sinh viên có tên là Nguyễn Văn An, mssv: 20160001:*

***IT3920-STUDENT\_ID-NAME-Lab05***

***Vd: IT3920-2016001-AnNV—Lab05***

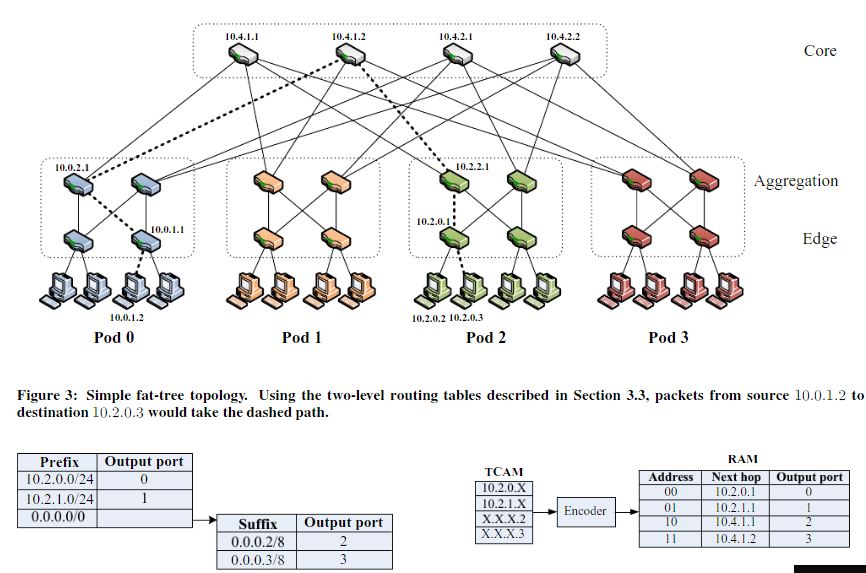
Ở bài Lab này, SV cần hoàn thành một ví dụ cho phép xây dựng bảng định tuyến của Fat Tree, sau đó sinh ngẫu nhiên cặp nguồn đích rồi dựa vào cặp nguồn đích đó, in ra màn hình đường đi của gói tin giữa từng cặp nguồn đích. Chú ý ta vẫn chưa thực hiện việc gửi gói tin. Giá trị k của Fat Tree cũng được nhập vào từ bàn phím để rồi sinh ra file NED.

**Lab05 sẽ có hạn cuối trước 12h đêm thứ 3 tuần sau (21/04). Chấp thuận cho SV có thể không kịp hoàn thành đủ 3 câu kia, nhưng tối thiểu là làm được 2 câu a, b.**

a. Xây dựng bảng định tuyến:

Bảng định tuyến là bảng ghi thông tin về đường đi đến một nút đích hoặc nhóm nút đích nào đó. Ở từng switch khác nhau sẽ có một bảng định tuyến. Với Fat Tree thì các switch cùng loại (cùng Edge Switch. Aggressive switch, và core switch) sẽ có bảng định tuyến giống nhau.

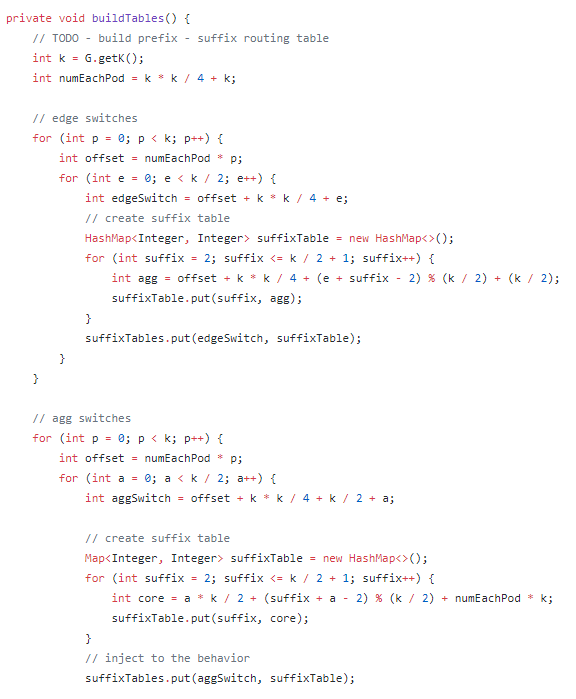
Xem trong bài báo này: <https://dl.acm.org/doi/pdf/10.1145/1402946.1402967>



Trong mã nguồn java này đã có cài đặt về cách xây dựng bảng định tuyến:

<https://github.com/bkhnk48/SSiNET/blob/master/java/custom/fattree/FatTreeRoutingAlgorithm.java>

chú ý ở hàm buildTable( )



Ở mã nguồn Java, người ta sử dụng HashMap và các cấu trúc như là Triple<Integer, Integer, Integer> và Pair<Integer, Integer>. Hiện tại C/C++ có thể chưa có những cấu trúc kia (Triple và Pair), thì (gợi ý) SV có thể dùng mẹo một triple (a, b, c) => d = (a << 16) | (b << 8) | c. Vì trong mã nguồn Java kia, các giá trị 0 <= a, b, c <= 255.

Viết mã này trong omnet++

b) Sinh ngẫu nhiên cặp nguồn đích:

với một giá trị k cho trước thì ta có k\*k\*k/4 host. SV viết chương trình chia đôi số host kia ra làm 2 tập bằng nhau. Rồi ghép cặp ngẫu nhiên giữa hai host trong từng cặp đó. Tức là host X lấy từ tập A sẽ ghép cặp với host Y lấy từ tập B. |A| = |B|.

in ra màn hình danh sách các cặp nguồn đích đó. Viết mã này trong omnet++

c) với từng cặp nguồn nguồn đích, in ra đường đi giữa chúng, căn cứ vào bảng định tuyến

SV có thể xem mã nguồn ở hàm public RoutingPath path(int source, int destination) {

Và hàm public int next(int source, int current, int destination) {

Các hàm này vẫn ở trong file kia

<https://github.com/bkhnk48/SSiNET/blob/master/java/custom/fattree/FatTreeRoutingAlgorithm.java>

Để tạo ra được địa chỉ IPv4 của các switch và host, SV cần tham khảo hàm: private void buildAddress() {

Tại file này:

<https://github.com/bkhnk48/SSiNET/blob/master/java/custom/fattree/FatTreeGraph.java>