ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

TÍNH TOÁN SONG SONG

Đề bài tập lớn số 3

Hiện thực Association Rules bằng Threading Building Blocks và Cilk Plus

GVHD: Nguyễn Quang Hùng
Nguyễn Mạnh Thìn
SV: Tạ Huỳnh Thùy Linh- 1511778
Nguyễn Thiện Quang - 1512648

TP. HỒ CHÍ MINH, THÁNG 4/2018

Mục lục

1 Giới thiệu bài toán

Trong lĩnh vực Data Mining, Association Rule là một kỹ thuật để tìm ra các mối quan hệ giữa các đối tượng trong tập dữ liệu lớn. Kỹ thuật này được ứng dụng nhiều trong phân tích cơ sở dữ liệu về các giao dịch thị trường, phân tích dữ liệu khoa học, Web mining, chuẩn đoán bệnh,... Nội dung cơ bản của Association Rule có thể được mô tả như sau. Cho $I = \{i_1, i_2, ..., i_d\}$ là tập tất cả các phần tử và $T = \{t_1, t_2, ..., t_n\}$ là tập tất cả các giao dịch trong cơ sở dữ liệu. Một Association Rule là một cách diễn đạt hàm ý dạng $X \Rightarrow Y$, với X và Y à hai tập phần tử rời nhau $X \cap Y = \varnothing$, và X là điều kiện cho trước, Y là kết quả theo sau. Bài toán Association Rule bao gồm việc tìm ra tất cả các luật nói lên mối quan hệ hay tương quan giữa các phần tử trong cơ sở dữ liệu giao dịch T. Độ hỗ trợ(Support) và độ tin cậy(Confidence) là hai tham số dùng để đo lường một Association Rule. Độ hỗ trợ của một luật là tần suất của giao dịch chứa tất cả các phần tử trong cả hai tập X và Y . Độ tin cậy của một luật là xác suất xảy ra Y khi đã biết X . Công thức của hai tham số này là:

$$\textbf{Support, } s(X \Rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$
$$\textbf{Confidence, } c(X \Rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Với $\sigma(X) = |\{t_i | X \subseteq t_i, t_i \in T\}|$ là số lần xuất hiện của tập phần tử t_i trong cơ sở dữ liệu giao dịch T.

Để giải quyết bài toán Association Rule, người ta chia thành 2 tác vụ nhỏ:
1. Tạo ra tập phần tử phổ biến: mục tiêu là tìm tất cả các tập phần tử có độ ủng hộ $\geq$ độ ủng hộ tối thiểu (minsup). Những tập phần tử này được gọi là các tập phần tử phổ biến.
2. Sinh luật: mục tiêu là lấy được tất cả các luật có độ tin cậy cao, là các luật có độ tin cậy $\geq$ độ tin cậy tối thiểu (minconf) từ tập phần tử phổ biến đã được xác định ở tác vụ trước. Những luật này được gọi là luật mạnh.

Mục đích của bài toán Association Rule là tìm tất cả các luật mạnh với độ tin cậy tối thiểu và độ phổ biến tối thiểu được xác định trước.

2 Lập trình trên Xeon Phi

- Xoen Phi: là bộ đồng xử lý (coprocessor) dựa trên kiến trúc Đa nhân Tích hợp (MIC) do Intel phát triển, được xây dựng trên những gì tinh túy nhất từ trước, với hơn 50 nhân x86, Xoen Phi tập trung vào tính toán thông thường dựa trên diện toán song song (parallel computing) có năng lực tính toán chính xác kép (FP64) đạt hơn 1 TFlops (1 triệu tỷ phép toán mỗi giây).
- Bộ đồng xử lý Xoen Phi cung cấp hiệu quả năng suất tính toán cao, hỗ trợ vector và băng thông bộ nhớ cục bộ, trong khi vẫn duy trì khả năng lập trình và hỗ trợ liên kết với bộ xử lý Intel Xoen Phi.

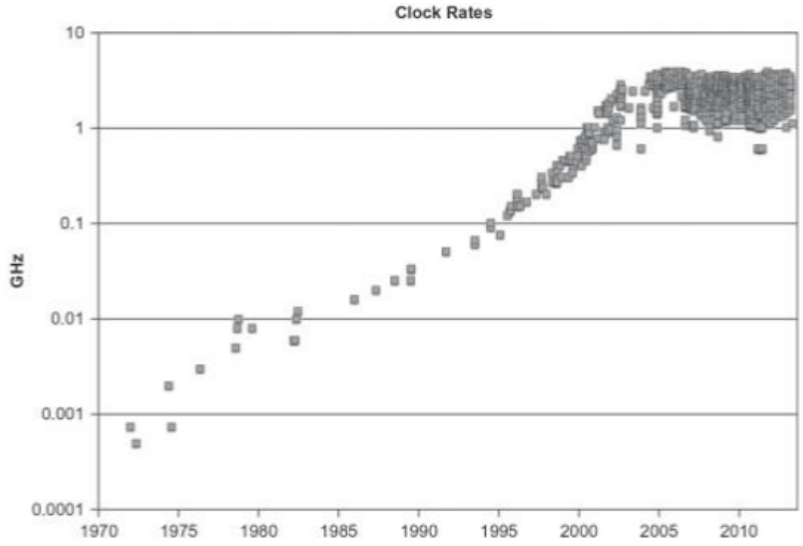


FIGURE 1.1

Processor/Coprocessor Speed Era [Log Scale].

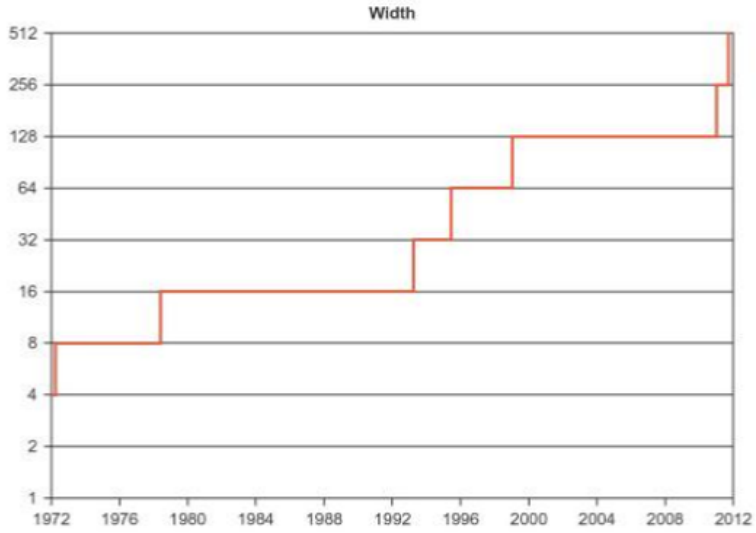


FIGURE 1.3

Processor/Coprocessor Vector Parallelism [Log Scale].

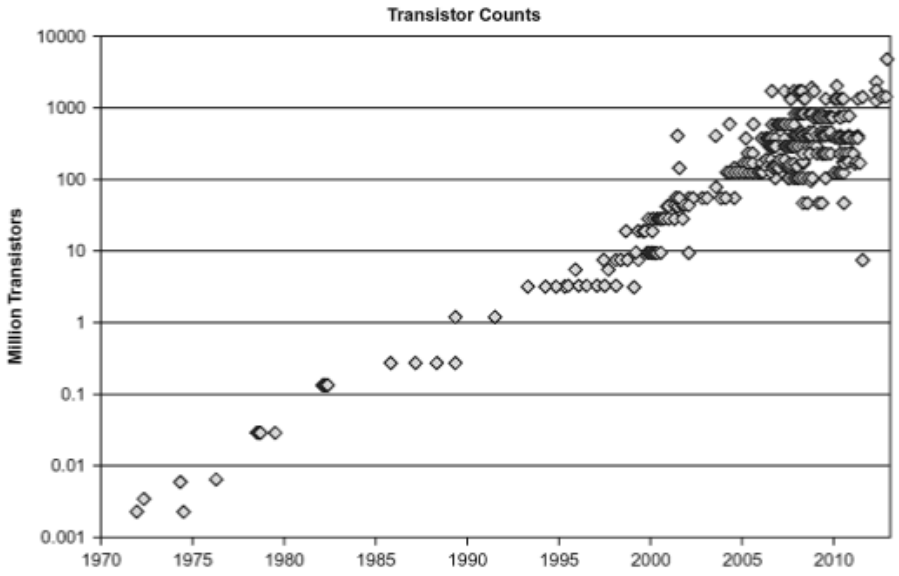


FIGURE 1.4
Moore's Law Continues, Processor/Coprocessor Transistor Count [Log Scale].

- Lập trình trên Xoen Phi là lập trình ứng dụng tận dụng tối đa lợi ích từ tính toán song song từ các process, có khả năng tạo hệ thống hung mạnh, cung cấp hiệu suất vượt trội và hiệu quả năng lực.

3 Các giải thuật

Nhiều giải thuật đã được tạo để giải bài toán Association Rules như AIS, SETM, APRIORI, FP-growth, Eclat,Partition...

Trong đó giải thuật phổ biến và đơn giản nhất là giải thuật Apriori. Giải thuật này sử dụng một quy tắc thực tế : bất kì tập con khác rỗng nào của một frequent itemset cũng là một frequent itemset.

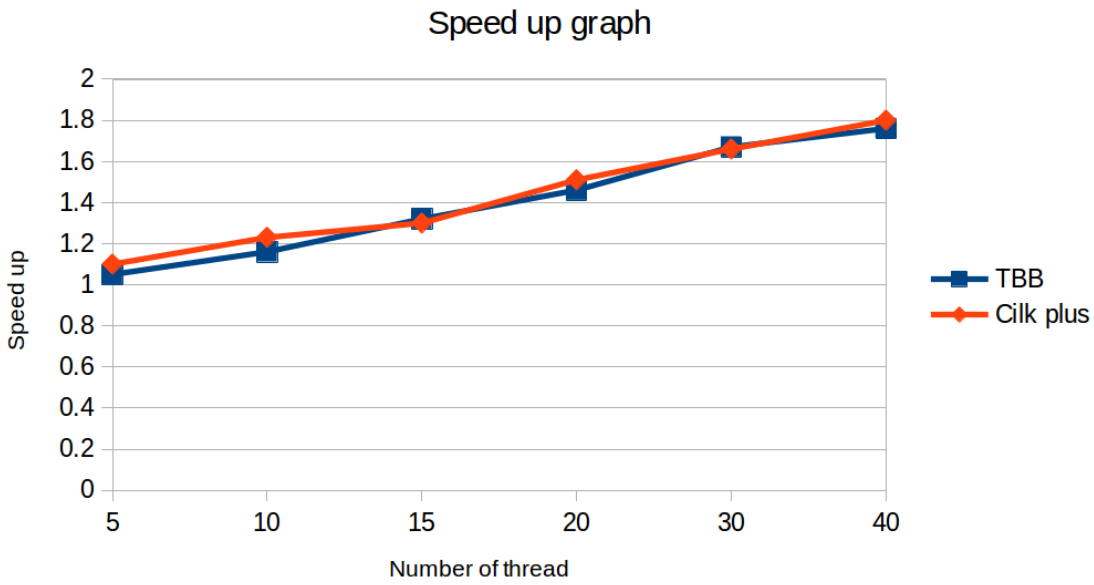
Tư tưởng chính của giải thuật Apriori :
- Tìm frequent itemsets (itemset có tần số xuất hiện $\geq$ min sup): itemset có k items (k − itemset) sẽ được dùng để tìm itemset có k + 1 ((k + 1) - itemset) items. Đầu tiên tìm (1 - itemset(L_1)). Sau đó dùng L_1 để tìm L_2 , dùng L_2 để tìm L_3,... đến khi không tìm được itemset nào nữa.
- Từ frequent itemsets, tìm ra các luật kết hợp mạnh.

Apriori Algorithm
1. Duyệt (Scan) toàn bộ transaction database để có được support S của 1 - itemset, so sánh S với min sup , để có được 1 - itemset(L_1).
2. Sử dụng L_{k-1} nối (join) L_{k-1} để sinh ra candidate k - itemset. Loại bỏ các itemsets không phải là frequent itemsets thu được k - itemset .
3. Scan transaction database để có được support của mỗi candidate k - itemset, so sánh S với

minsup để thu được frequent k - itemset (L_k).
4. Lặp lại từ bước 2 cho đến khi Candidate set (C) trống (không tìm thấy frequent itemsets). 5. Với mỗi frequent itemset I, sinh tất cả các tập con s không rỗng của I.
6. Với mỗi tập con s không rỗng của I, sinh ra các luật s ⇒ (I - s) nếu độ tin cậy (Confidence) của nó ≥ minconf

4 Kết luận

4.1 Kết quả



4.2 Threading Building Blocks

4.2.1 Kiến thức

1. $tbb :: task_scheduler_init$
Khởi tạo thư viện cho phép tạo ra các thread và định thời tác vụ cho thread trong bài toán song song.
2. $tbb :: blocked_range$
Không gian một chiều, sử dụng trong chỉ số lặp cho các hàm trong thư viện Thread Building Block.
3. $tbb :: parallel_reduce$
Template: $parallel_reduce < Range, Body >$ thực hiện song song vòng lặp và tạo ra một kết quả chung cho Body.
Template trên được dùng trong hàm tạo tập L, tính độ ủng hộ, tạo luật kết hợp và tạo chuỗi luật để ghi lên file.
4. $tbb :: auto_partitioner$
Tự động xác định số vòng lặp cho mỗi thread từ số vòng lặp đã cho, kích thước lặp tùy theo

bài toán.
5. $tbb :: parallel_sort$
Sắp xếp chuỗi số với độ phức tạp(Onlogn) trên một processor, dần đến O(N) khi các processor được thêm vào.
Template: $parallel_sort(i, j, comp)$: sắp xếp chuỗi [i,j) dùng tham số comp để xác định thứ tự.
Template trên được dùng trong hàm sắp xếp mỗi giao dịch theo thứ tự từ nhỏ đến lớn.

4.2.2 Khó khăn và kiến thức nhận được

Khó khăn: Trước khi làm bài tập lớn, chưa biết về TBB và thuật toán Association Rules, không có hướng dẫn dùng TBB với Xeon Phi trên máy tính của trường.
Kiến thức nhận được:
- Làm quen với TBB và lập trình song song với TBB
- Cách trao đổi, làm việc nhóm
- Cách dùng vim để code và debug
- Học được thuật toán Association Rule

4.3 Cilk Plus

4.3.1 Kiến thức

1. $cilk_for$: Cho phép hiện thực vòng lặp song song.
2. $cilk_spawn$: Đặc tả cho các function có thể thực hiện đồng thời.
3. $cilk_sync$: Đồng bộ các function thực hiện đồng thời rồi mới tiếp tục thực hiện các lệnh sau.
4. $Reducers$: Loại bỏ tranh chấp cho các biến cùng thực hiện trên 1 vùng nhớ bằng cách tự động tạo chế độ truy cập của chúng khi cần.
5. $SIMD-EnabledFunctions$: Xác định các hàm có thể được vector hóa khi được gọi bên trong biểu thức mảng hoặc vòng lặp $\#pragmasimdloop$.
6. $\#pragmasimd$: Chỉ định 1 vòng lặp sẽ được vector hóa.

4.3.2 Khó khăn và kiến thức nhận được

Khó khăn: Trước khi làm bài tập lớn, chưa biết về TBB và thuật toán Association Rules, không có hướng dẫn dùng Intel Cilk Plus với Xeon Phi trên máy tính của trường.
Kiến thức nhận được:
- Làm quen với Cilk Plus và lập trình song song với Cilk Plus
- Cách trao đổi, làm việc nhóm
- Cách dùng vim để code và debug
- Học được thuật toán Association Rule

Tài liệu

[1] $http://www.saedsayad.com/association_rules.htm/$ *Ngày xem:* 25/02-04/03/2018

[2] $http://www.kdd.org/exploration_files/hipp.pdf/$ *Ngày xem:* 25/02-04/03/2018

[3] $http://bis.net.vn/forums/t/389.aspx/$ *Ngày xem:* 25/02-04/03/2018

[4] *https://www.cilkplus.org/* *Ngày xem:* 25/02-30/04/2018

[5] *https://www.threadingbuildingblocks.org* *Ngày xem:* 25/02-30/04/2018