HNUE JOURNAL OF SCIENCE DOI: 10.18173/2354-1059.2018-0011
Natural Sciences 2018, Volume 63, Issue 3, pp. 108-116
This paper is available online at http://stdb.hnue.edu.vn

THUẬT TOÁN NHÁNH CẬN GIẢI BÀI TOÁN LẬP LỊCH LUỒNG CÔNG VIỆC

Phan Thanh Toàn[1], Đặng Quốc Hữu[2] và Nguyễn Thế Lộc[3]
[1]*Khoa Sư phạm Kĩ thuật, Trường Đại học Sư Phạm Hà Nội*
[2]*Trung tâm Công nghệ thông tin, Trường Đại học Thương mại*
[3]*Khoa Công nghệ Thông tin, Trường Đại học Sư phạm Hà Nội*

Tóm tắt. Điện toán đám mây là một môi trường dịch vụ dựa trên nền tảng công nghệ thông tin và truyền thông, mọi tài nguyên trên hệ thống đều được cung cấp cho người sử dụng dưới dạng dịch vụ, và người sử dụng chỉ phải chi trả các tài nguyên thực dùng. Với sự ra đời của công nghệ điện toán đám mây rất nhiều các ứng dụng trong lĩnh vực công nghệ thông tin đã có những thay đổi căn bản, chuyển từ dạng cung cấp sản phẩm đóng gói sử dụng riêng rẽ sang dạng cung cấp dịch vụ và được duy trì, vận hành bởi nhà cung cấp dịch vụ qua đó giảm đáng kể chi phí cho người dùng. Trong thực tiễn và nghiên cứu khoa học có nhiều bài toán được biểu diễn dưới dạng mô hình luồng công việc như lập lịch cho dây chuyền sản xuất, lập lịch điều phối tài nguyên trong hệ điều hành, lập lịch thời khóa biểu. Lập lịch là hoạt động nhằm gán các tác vụ vào thực hiện trên các tài nguyên tính toán và thảo mãn các ràng buộc về thứ tự các tác vụ trong luồng công việc cũng như các giới hạn về tài nguyên. Đa số các bài toán thuộc họ lập lịch đã được chứng minh thuộc lớp NP-Khó [1], do vậy việc tìm ra các thuật toán lập lịch nhằm cực tiểu hóa chi phí hoàn thành luồng công việc là một lĩnh vực khó và đã thu hút được sự quan tâm của nhiều nhà khoa học. Bài báo này đề xuất một thuật toán lập lịch luồng công việc mới nhằm cực tiểu hóa chi hoàn thành luồng công việc trong môi trường thực thi điện toán đám mây dựa trên phương pháp nhánh cận.

Từ khóa: Lập lịch luồng công việc, ứng dụng luồng công việc, điện toán đám mây, phương pháp nhánh cận.

1. Mở đầu

Trong những năm gần đây điện toán đám mây đã được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau của cuộc sống và nghiên cứu khoa học. Trong môi trường điện toán đám mây mọi tài nguyên phần cứng, phần mềm đều được cung cấp cho khách hàng dưới dạng dịch vụ, khách hàng chỉ phải chi trả phí sử dụng theo tài nguyên thực dùng.

Bài toán lập lịch luồng công việc là một bài toán đã được nghiên cứu từ những năm 1950, và bài toán này đã được chứng minh thuộc lớp NP-Khó. Nhiều ứng dụng khoa học được mô hình hóa bởi dạng đồ thị luồng công việc như ứng dụng Montage [1], CyberShake [2], Epigenomics [3], LIGO [4]. Vấn đề lập lịch thực thi luồng công việc trong môi trường điện toán đám mây về bản chất là tìm phương án ánh xạ những tác vụ của luồng công việc tới các máy chủ của đám mây thỏa mãn ràng buộc về thứ tự của các tác vụ trong luồng công việc và chi phí hoàn thành luồng

Ngày nhận bài: 18/9/2017. Ngày sửa bài: 26/3/2018. Ngày nhận đăng: 31/3/2018.
Tác giả liên hệ: Phan Thanh Toàn. Địa chỉ e-mail: pttoan@hnue.edu.vn

công việc là nhỏ nhất. Đã có nhiều công trình nghiên cứu xem xét vấn đề này nhằm tối thiểu hóa các hàm mục tiêu khác nhau như tổng chi phí, tổng thời gian thực thi tại các máy chủ,... nhưng chưa có công trình nào giải quyết bài toán với hàm mục tiêu là thời gian thực hiện (makespan). Bài báo này nhằm giải quyết vấn đề đó.

2. Nội dung nghiên cứu

2.1. Những công trình nghiên cứu liên quan

Bài toán lập lịch luồng công việc đã được chứng minh là thuộc lớp NP-Khó [5] nghĩa là thời gian để tìm ra lời giải tối ưu tăng rất nhanh theo kích cỡ dữ liệu đầu vào, vì vậy đã có nhiều công trình nghiên cứu nhằm tìm ra lời giải đúng hoặc gần đúng của bài toán này.

N.S.Grigoreva [6] đã đề xuất thuật toán lập lịch điều phối các tác vụ của luồng công việc vào thực hiện trên một hệ thống đa bộ vi xử lí nhằm cực tiểu hóa thời gian hoàn thành luồng công việc. Tác giả đã sử dụng kết hợp phương pháp nhánh cận và kĩ thuật tìm kiếm nhị phân để tìm ra phương án xếp lịch có thời gian hoàn thành luồng công việc là nhỏ nhất.

Sadhasivam đã đề xuất thuật toán lập lịch luồng công việc dựa trên sự cân bằng tải trong môi trường điện toán đám mây [7]. Thuật toán không chỉ đáp ứng các yêu cầu từ người sử dụng mà còn cung cấp khả năng sử dụng tài nguyên một cách hiệu quả. Đây là thuật toán theo hướng nâng cao hiệu quả dịch vụ dựa trên Meta-heuristic.

Các tác giả trong bài báo [8] đã đề xuất thuật toán EGA (Enhanced Genetic Algorithm) lập lịch bằng phương pháp di truyền. Trong công trình các tác giả sử dụng thuật toán Enhanced Max Min trong bước khởi tạo quần thể nhằm tìm ra các cá thể tốt cho quá trình tiến hóa.

Pandey [9] đã đề xuất thuật toán lập lịch luồng công việc PSO Heuristic (Particle Swarm Optimization Heuristic – PSO_H) trong môi trường điện toán đám mây dựa trên chiến lược tối ưu bày đàn. Rajkumar đã đề xuất một thuật toán lập lịch phân cấp [10] và đưa vào các tham số dịch vụ khác nhau, chẳng hạn như thời gian đáp ứng. Thuật toán sử dụng tham số này như một quyền ưu tiên để lựa chọn các tác vụ lập lịch. Cao và các đồng nghiệp đã trình bày thuật toán lập lịch dựa trên giải thuật ABC (Activity Based Costing) [11]. Thuật toán này gán mức ưu tiên cho mỗi tác vụ trong luồng công việc theo các tham số về thời gian, không gian, các tài nguyên và chi phí, quá trình lập lịch sẽ sử dụng mức ưu tiên này để quyết định các tác vụ được chọn trong quá trình lập lịch.

Selvi và các cộng sự đã đề xuất thuật toán lập lịch luồng công việc trong môi trường điện toán lưới (Grid) [12], trong công trình tác giả đã vận dụng thuật toán tiến hóa vi phân (DE) vào giải bài toán lập lịch luồng công việc trên môi trường điện toán lưới nhằm cực tiểu thời gian hoàn thành luồng công việc (makespan), trong công trình tác giả đã chỉ ra giá trị Makespan tìm được bởi thuật toán đề xuất là nhỏ hơn so với thuật toán PSO. Xu và các cộng sự đã đề xuất thuật toán COODE [13] (Current Optimum Opposition-based Differential Evolution) nhằm tìm giá trị tối ưu cho các hàm số dựa theo phương pháp tiến hóa vi phân đối xứng, trong công trình tác giả đã đề xuất công thức tìm điểm đối xứng của một điểm dựa theo giá trị tối ưu hiện tại nhằm thay đổi toán tử đột biến trong phương pháp tiến hóa vi phân và tác giả đã so sánh thuật toán COODE với các thuật toán DE và ODE, kết quả đã chỉ ra thuật toán đề xuất COODE tốt hơn các thuật toán đối sánh.

2.2. Mô hình toán học bài toán lập lịch luồng công việc trong môi trường điện toán đám mây

Giả sử cần sắp xếp lịch biểu cho một luồng công việc trong môi trường đám mây với các giả thiết như sau:

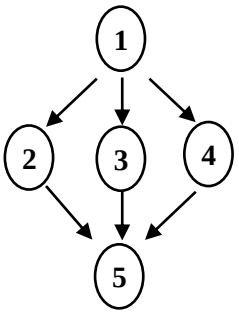
- Luồng công việc được biểu diễn bởi đồ thị G = (V, E), với V là tập đỉnh của đồ thị, mỗi đỉnh biểu thị cho một tác vụ.
- T ={T_1, T_2,…,T_M} là tập các tác vụ, M là số lượng tác vụ của luồng công việc đang xét.
- E là tập cạnh thể hiện mối quan hệ cha-con giữa các tác vụ. Cạnh (T_i, T_j) ∈ E cho biết tác vụ T_i là cha của tác vụ T_j, dữ liệu đầu ra của T_i sẽ là dữ liệu đầu vào cho tác vụ T_j (xem Hình 1).
- Tập máy chủ của đám mây kí hiệu là S = {S_1, S_2,….,S_N}, N là số lượng máy chủ của đám mây.
- Mỗi tác vụ có thể được thực thi trên một máy chủ bất kì, máy chủ đó phải thực hiện toàn bộ tác vụ từ đầu đến cuối.
- Khối lượng tính toán (Workload) của tác vụ T_i kí hiệu là W_i với đơn vị đo là flop (floating point operations: phép tính trên số thực dấu phảy động). W_i được cho trước ($\forall i$ = 1,2, …M)
- Tốc độ tính toán của máy chủ S_i, đơn vị là MI/s (million instructions/second), được kí hiệu bởi P(S_i), là giá trị được cho trước ($\forall i$ = 1,2, …M)
- Giữa hai máy chủ S_i, S_j bất kỳ ($1 \le i,j \le N$) có một đường truyền với băng thông, đơn vị là Megabit/s, được biểu thị bởi hàm hai biến B() được định nghĩa như sau:

$$B: S \times S \rightarrow R^+$$
$$(S_i, S_j) \rightarrow B(S_i, S_j)$$

- Giả thiết hàm băng thông B() thỏa mãn các điều kiện sau:
  - $B(S_i,S_i) = \infty$: thời gian truyền tại chỗ bằng không
  - $B(S_i,S_j) = B(S_j,S_i)$: tốc độ truyền hai chiều bằng nhau
  - Giá trị $B(S_i,S_j)$ được cho trước ($\forall i,j$).
- Khối lượng dữ liệu do tác vụ T_i chuyển tới tác vụ T_j, kí hiệu là D_{ij} với đơn vị là Megabit, là giá trị cho trước ($\forall i,j$).
- Mỗi phương án xếp lịch thực thi luồng công việc tương đương với một hàm f()

$$f : T \rightarrow S$$
$$T_i \rightarrow f(T_i)$$

Trong đó $f(T_i)$ là máy chủ chịu trách nhiệm thực thi tác vụ T_i



***Hình 1**: Đồ thị biểu diễn một luồng công việc với 5 tác vụ*

Biến quyết định và hàm mục tiêu

- Sử dụng biến nhị phân x_j^k ; với $x_j^k = 1$ nếu tác vụ T_k được thực hiện trên máy chủ S_j
- Biến $d_{i,j}^k$ biểu diễn khối lượng dữ liệu được truyền từ máy chủ S_i đến máy chủ S_j cho tác vụ T_k nếu $x_j^k = 1$
- $txcost_{i,j}$ là chi phí truyền một đơn vị dữ liệu từ máy chủ S_i đến S_j cho tác vụ T_k, nếu $d_{i,j}^k > 0$ và $x_j^k = 1$
- $excost_j$ biểu diễn chi phí sử dụng máy chủ tính toán S_j để thực hiện tác vụ T_k nếu $x_j^k = 1$
- $extime_j^k$ biểu diễn thời gian thực hiện tác vụ T_k trên máy chủ S_j nếu $x_j^k = 1$

Hàm mục tiêu

$$C = \sum_{i,j \in S, k \in T} d_{i,j}^k \times tx\cos t_{i,j} \times x_j^k + ex\cos t_j \times extime_j^k \times x_j^k$$

$$\boldsymbol{C \rightarrow min}$$

Các điều kiện ràng buộc :

(a) $\forall k \in T$, $j \in S$; $x_j^k \geq 0$; biến nhị phân nhận giá trị 0 hoặc 1

(b) $\forall k \in T$, $i,j \in S$; $d_{i,j}^k \geq 0$; khối lượng dữ liệu truyền giữa các tác vụ đảm bảo lớn hơn hoặc bằng 0

(c) $\forall k \in T$; $tdata^k \geq 0$; tổng khối lượng dữ liệu truyền tới tác vụ T_k phải lớn hơn hoặc bằng 0

(d) $\forall i,j \in S$; $txcost_{i,j} \geq 0$; chi phí truyền thông lớn hơn hoặc bằng 0

(e) $\forall k \in T$, $j \in S$; $excost_j \geq 0$; chi phí thực thi tác vụ lớn hơn hoặc bằng 0

(f) $\forall k \in T$, $j \in S$; $extime_j^k \geq 0$; thời gian thực hiện tác vụ đảm bảo lớn hơn hoặc bằng 0

(g) $\sum_{j \in S} x_j^k = 1$; đảm bảo mỗi tác vụ T_k chỉ được thực hiện trên một máy chủ xác định

(h) $\sum_{j \in S} x_j^k \times d_{i,j}^k = tdata^k$; tổng khối lượng dữ liệu được chuyển tới tác vụ T_k

(i) $\sum_{i,j \in S, k \in T} x_j^k \times d_{i,j}^k = \sum_{k \in T} tdata^k$; đảm bảo tính cân bằng giữa dữ liệu vào và ra của các tác vụ trong luồng công việc

2.3. Giải pháp đề xuất

Nhánh cận là một kĩ thuật duyệt có sử dụng hàm cận dưới nhằm cắt nhánh để giảm bớt không gian tìm kiếm trong quá trình duyệt. Thuật toán duyệt nhánh cận gồm hai thủ tục chính:

- Phân nhánh: phân hoạch tập các phương án ra thành các tập con với kích thước càng ngày càng nhỏ cho đến khi thu được phân hoạch tập các phương án ra thành các tập con một phần tử
- Tính cận: đưa ra cách tính cận cho giá trị hàm mục tiêu của bài toán trên mỗi tập con trong phân hoạch của tập các phương án.

Thuật toán nhánh cận đề xuất

Biểu diễn lời giải

Mỗi phương án xếp lịch được biểu diễn bởi một vector có độ dài bằng số tác vụ trong luồng công việc. Giá trị tương ứng với mỗi vị trí i trong vector biểu diễn số hiệu máy chủ thực thi tác vụ i. Ví dụ: Xét luồng công việc với 5 tác vụ T = {T_1, T_2,..., T_5}, tập máy chủ gồm 3 máy S = {S_1, S_2, S_3}. Khi đó phương án xếp lịch (1,2,1,3,2) được biểu diễn như sau:

T_1	T_2	T_3	T_4	T_5
S_1	S_2	S_1	S_3	S_2

Hàm tính cận dưới cho phương án bộ phận

Mỗi lời giải của bài toán là một vector M chiều x = (x_1, x_2,...,x_M); với $x_i \in$ S. Gọi c_{min} = min{P(S_i)}; $i \in S$; giá trị nhỏ nhất về năng lực tính toán trong số các máy chủ của hệ thống đám mây.

Cận dưới cho phương án bộ phận (x_1, x_2,...,x_L) tương ứng với việc sắp xếp tập L tác vụ T_L ={T_1, T_2,...,T_L} thực hiện trên các máy chủ tương ứng S_L= (S_{x1}, S_{x2},..,S_{xL}). Khi đó chi phí thực hiện của phương án bộ phận là:

$$\delta = \sum_{i,j \in S, k \in T_L} d_{i,j}^k \times tx\cos t_{i,j} \times x_j^k + ex\cos t_j \times extime_j^k \times x_j^k$$

Hàm cận dưới của phương án bộ phận ($x_1, x_2, ..., x_L$) được tính theo công thức sau đây

$$g(k) = \delta + C_{\min} \times \sum_{j \in S, k \in T - T_L} excost_j \times extime_j^k \times x_j^k$$

Chứng minh:

Ta có $min\{\sum_{i,j \in S, k \in T} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k\}$

$$= min\left\{ \sum_{i,j \in S, k \in T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k + \sum_{i,j \in S, k \in T - T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k \right\}$$

$$= min\{\delta + \sum_{i,j \in S, k \in T - T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k\} =$$

$$= \delta + min\left\{ \sum_{i,j \in S, k \in T - T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times extime_j^k \times x_j^k \right\}$$

$$= \delta + min\left\{ \sum_{i,j \in S, k \in T - T_L} d_{i,j}^k \times tfcost_{i,j} \times x_j^k + excost_j \times \frac{W_k}{P(S_j)} \times x_j^k \right\}$$

$$\geq \delta + min\left\{ \sum_{i,j \in S, k \in T - T_L} excost_j \times \frac{W_k}{P(S_j)} \times x_j^k \right\}; because \sum_{i,j \in S, k \in T - T_L} d_{i,j}^k \times tfcost_{i,j} \geq 0$$

$$\geq \delta + \frac{1}{C_{max}} \left\{ \sum_{j \in S, k \in T - T_L} excost_j \times W_k \times x_j^k \right\} = g(k)$$

Do vậy, g(k) là hàm cận dưới của lời giải bộ phận cấp k.

Thuật toán đề xuất

```
Function cost(x1, x2,..,xM)
begin
return  sum_{i,j∈S,k∈T_L} d_{i,j}^k × txcost_{i,j} × x_j^k + excost_j × extime_j^k × x_j^k  ;
end;
Procedure SchedulingBranch(k)
begin
 for j:=1 to M do
  if UCV(j,k) then
  begin
   a[i]:=j;
   if i=M then Ghinhan;
   else if g(k)< fopt then
     SchedulingBranch(k+1);
  end;
end;
```

```
Procedure UCV(j,k)
begin
 var i:integer;
 for i=1 to k-1 do
   if j=a_i then
     return false;
   else return true;
end;
Procedure ghinhan
begin
 double c = cost(x_1, x_2,..,x_M);
 if c < fopt then
   fopt = c;
end
Algorithm Scheduling
begin
  1. tính ma trận chi phí thực thi các tác vụ trên các máy chủ
  2. Tính ma trận chi phí truyền dữ liệu giữa các máy chủ
  3. double fopt = +∞;
  4. SchedulingBranch(T_1);
  5. writeln(fopt);
end
```

2.4. Thực nghiệm

Để kiểm chứng thuật toán đề xuất chúng tôi đã sử dụng công Visual studio 2012, trên nền ngôn ngữ lập trình C#. Các chương trình chạy trên máy tính cá nhân với bộ vi xử lý Intel Core i5 2.2 GHz, RAM 4 GB, hệ điều hành Windows 7 Ultimate.

2.4.1. Phân nhóm dữ liệu

Dữ liệu thực nghiệm bao gồm:

- Dữ liệu về tốc độ tính toán của các máy chủ và băng thông giữa các máy chủ được lấy từ các công ty cung cấp dịch vụ cloud như Amazon [19].

- Dữ liệu luồng công việc được lấy từ các bộ dữ liệu thử nghiệm được xây dựng theo độ trù mật khác nhau và các luồng công việc từ các ứng dụng thực tế như ứng dụng Montage [1].

- Dựa theo tính chất của môi trường điện toán đám mây, đây là một môi trường tính toán không đồng nhất, tốc độ tính toán các máy chủ và băng thông không đồng đều, đồng thời cũng dựa theo tính chất các luồng công việc, số lượng tác vụ, độ trù mật của đồ thị luồng công việc chúng tôi đã tiến hành phân loại dữ liệu theo các nhóm với quan hệ về tốc độ tính toán các máy chủ và băng thông khác nhau, độ trù mật của đồ thị luồng công việc cũng được thử nghiệm qua hệ số α khác nhau. Chi tiết các nhóm dữ liệu theo số lượng máy chủ N, số tác vụ M và hệ số α như sau:

- Nhóm 1: M = 10, N = 3; Nhóm 2: M = 10, N = 5, Nhóm 3: M = 20, N = 3, Nhóm 4: M = 20, N = 5.

- Mỗi nhóm lại bao gồm ba thực nghiệm khác nhau về tỷ lệ số cạnh trên số đỉnh của đồ thị luồng công việc, kí hiệu là α và tính bởi công thức:

$$\alpha = \frac{|E|}{M \times (M-1)/2}$$

2.4.2. Tham số cấu hình

Các tham số cấu hình của đám mây được thiết lập trong miền giá trị như sau:

Tốc độ tính toán P của các máy chủ: từ 1 đến 250 (million instructions/s); khối lượng dữ liệu D giữa các tác vụ: từ 1 đến 1000 (MB); băng thông giữa các máy chủ B:từ 10 đến 100 (Mega bit/s).

Bảng 1. Ma trận dữ liệu truyền thông, chi phí tính toán

TP[5x3]		**PC1**	**PC2**		**PC3**
	T_1	0.1*25	0.2*25		0.3*25
	T_2	0.1*25	0.2*25		0.3*25
	T_3	0.1*25	0.2*25		0.3*25
	T_4	0.1*25	0.2*25		0.3*25
	T_5	0.1*25	0.2*25		0.3*25
PP[3x3]		**PC1**	**PC2**		**PC3**
	PC_1	0	0.1		0.1
	PC_2	0.1	0		0.1
	PC_3	0.1	0.1		0
$DS_{T2,T3,T4}$ [2x2]		**Data Size (MB)**	**DS_{T5} [2x2]=**		**DataSize (MB)**
	Input	10		**Input**	30
	Output	10		**Output**	60

Giá trị ở bảng trên được lấy từ bảng giá sử dụng dịch vụ của Amazon EC2 [15] cho các tài nguyên trong phạm vi 1.1$ - 1.28$/giờ.

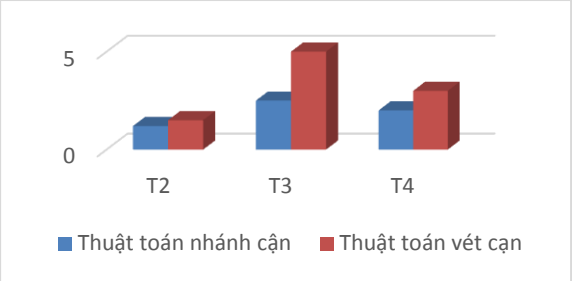
2.4.3. Kết quả thực nghiệm

Bảng 2. Kết quả thực nghiệm thuật toán nhánh cận đề xuất trên các bộ dữ liệu thực nghiệm

Dữ liệu	M	N	α	Thuật toán nhánh cận		Thuật toán duyệt toàn bộ	
				Giá trị tối ưu	Thời gian	Giá trị tối ưu	Thời gian
T1	5	3	0,4	3498,3	1(giây)	3498,3	1(giây)
T2	5	3	0,6	2764,7	1,2(giây)	2764,7	1,5 (giây)
T3	10	3	0,26	5892	2,5 (giây)	5892	4 (giây)
T4	10	3	0,3	7470,7	2 (giây)	7470,7	3 (giây)
T5	10	5	0,2	4866,2	5 (giây)	4866,2	28 (phút)
T6	10	5	0,53	5583,9	7 (giây)	5583,9	30 (phút)
T7	10	8	0,2	2733,6	14 (giây)	2733,6	45 (phút)
T8	10	8	0,5	2736,8	15 (giây)	2736,8	48 (phút)
T9	20	3	0,15	8679	6 (phút)	8679	121 (giờ)
T10	20	5	0,3	8685,4	6 (phút)	8685,4	132 (giờ)

Nhận xét: Bài toán lập lịch luồng công việc là bài toán thuộc lớp NP-khó, thời gian tính toán tăng theo hàm mũ của kích thước dữ liệu đầu vào, bài toán này có độ phức tạp tính toán là $O(M^N)$, với

M là số tác vụ trong luồng công việc và N là số máy chủ. Thuật toán nhánh cận đề xuất cho phép giải bài toán với kích thước dữ liệu đầu vào trung bình và nhỏ, thời gian thực hiện thuật toán nhánh cận nhỏ hơn đáng kể so với thuật toán duyệt toàn bộ. Hình 2 chỉ ra kết quả so sánh thời gian thực hiện thuật toán nhánh cận và thuật toán duyệt toàn bộ với 3 bộ dữ liệu thực nghiệm T1, T2, T3. Các bộ dữ liệu có kích thước đầu vào khác nhau và hệ số trù mật α của đồ thị luồng công việc là khác nhau.



Hình 2. So sánh thời gian thực hiện thuật toán nhánh cận và thuật toán duyệt toàn bộ

3. Kết luận

Lập lịch luồng công việc là một bài toán khó và được ứng dụng trong nhiều lĩnh vực của cuộc sống và khoa học như lập thời khóa biểu, điều phối tài nguyên trong hệ điều hành, lập lịch thực thi luồng công việc trong điện toán đám mây. Trong môi trường điện toán đám mây mọi tài nguyên phần cứng, phần mềm đều được cung cấp cho khách hàng dưới dạng dịch vụ và khách hàng sẽ phải trả chi phí cho các tài nguyên sử dụng. Việc lập lịch điều phối các tác vụ của khách hàng vào thực thi trên các máy chủ sao cho chi phí sử dụng tài nguyên nhỏ nhất là vấn đề quan trọng của điện toán đám mây. Bài báo này đã trình bày các nội dung chính sau:

- Đề xuất mô hình lý thuyết cho bài toán cực tiểu hóa chi phí thực thi luồng công việc trong môi trường điện toán đám mây

- Xây dựng hàm cận dưới và qua đó đề xuất thuật toán nhánh cận giải bài toán lập lịch luồng công việc.

Tiếp theo chúng tôi sẽ tiến hành nghiên cứu để giải bài toán này theo hướng tiếp cận metaheuristic nhằm tìm ra lời giải gần đúng cho bài toán với kích thước dữ liệu đầu vào lớn.

TÀI LIỆU THAM KHẢO

[1] G. B. Berriman, *et al.*, 2004. *Montage: A Grid Enabled Engine for Delivering Custom Science-Grade Mosaics On Demand*. Proc. of the SPIE Conference.

[2] P. Maechling *et al.*, 2006. *SCEC CyberShake Workflows, Automating Probabilistic Seismic Hazard Analysis Calculations*, Springer.

[3] "USC Epigenome Center". http://epigenome.usc.edu. [Online]. http://epigenome.usc.edu

[4] LIGO Project. LIGO - Laser Interferometer Gravitational Wave Observatory. [Online]. http://www.ligo.caltech.edu.

[5] J.D. Ullman, 1975. *NP-complete scheduling problems*. Journal of Computer and System Sciences, pages 384-393, volume 10, issue 3.

[6] N.S.Grigoreva, 2014. *Branch and Bound Method for Scheduling Precedence Constrained Tasks on Parallel Identical Processors*, Proc. of the World Congress on Engineering, Vol II,

[7] R. Rajkumar, T. Mala, 2012. *Achieving Service Level Agreement in Cloud Environment using Job Prioritization in Hierarchical Scheduling*, Proceeding of International Conference on Information System Design and Intelligent Application, vol. 132, pp. 547-554.

[8] S. Singh, M.Kalra, 2014. *Task scheduling optimization of independent tasks in cloud computing using enhanced genetic algorithm*, International Journal of Application or Innovation in Engineering & Management, vol.3, Issue 7.

[9] S. Pandey, L. Wu1, S. M. Guru, R. Buyya1, 2010. *A Particle Swarm Optimization (PSO)-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments*, Proc. of 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), pp. 400-407.

[10] R. Rajkumar, T. Mala, 2012. *Achieving Service Level Agreement in Cloud Environment using Job Prioritization in Hierarchical Scheduling*, Proceeding of International Conference on Information System Design and Intelligent Application, vol. 132, pp 547-554.

[11] Q. Cao, W. Gong and Z. Wei, 2009. *An Optimized Algorithm for Task Scheduling Based On Activity Based Costing in Cloud Computing*, In Proceedings of Third International Conference on Bioinformatics and Biomedical Engineering, pp.1-3

[12] S.Selvi, Dr. D.Manimegalai, Dr.A.Suruliandi, 2011. *Efficient Job Scheduling on Computational Grid with Differential Evolution Algorithm*, International Journal of Computer Theory and Engineering, vol. 3, No. 2, April.

[13] Q. XU, L.WANG, HE. Baomin, N.WANG, 2011. *Modified Opposition-Based Differential Evolution for Function Optimization*, Journal of Computational Information Systems, pp. 1582-1591.

[14] Phan Thanh Toàn, Nguyễn Thế Lộc, Nguyễn Doãn Cường, Đỗ Như Long, 2015. *Giải thuật tối thiểu hóa chi phí thực thi luồng công việc trong môi trường điện toán đám mây*, Tạp chí khoa học trường đại học Sư Phạm Hà Nội, pp. 47-55.

ABSTRACT

Branch and Bound Algorithm for Workflow Scheduling Problem

Phan Thanh Toan[1], Dang Quoc Huu[2] and Nguyen The Loc[3]

[1]*Faculty of Technology Education, Hanoi National University of Education*

[2]*Center for Information Technology, Vietnam University of Commerce*

[3]*Faculty of Information Technology, Hanoi National University of Education*

Cloud computing is a new trend of information and communication technology that enables resource distribution and sharing at a large scale. The Cloud consists of a collection of virtual machine that promises to provision on-demand computational and storage resources when needed. End-users can access these resources via the internet and have to pay only for their usage. Scheduling of scientific workflow applications on the Cloud is a challenging problem that has been the focus of many researchers for many years. In this work, we propose a novel algorithm for workflow scheduling that is derived from the Branch and Bound Algorithm.

Keywords: Workflow scheduling, branch and bound algorithm, Cloud computing.