



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY



Viện Công nghệ Thông tin và Truyền thông

# Sử dụng mạng Bayesian ước lượng tài nguyên khả dụng cho bài toán lập lịch trong môi trường Cloud Computing

Sinh viên thực hiện: Trương Quang Khánh  
Giảng viên hướng dẫn: TS. Nguyễn Bình Minh

Ngày 5 tháng 7 năm 2021



Giới thiệu chung

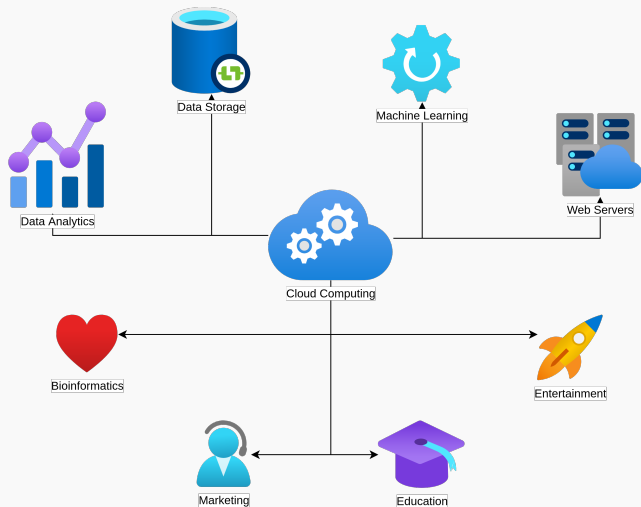
Các vấn đề của bài toán lập lịch thời gian thực

Giải pháp

Đánh giá hiệu năng

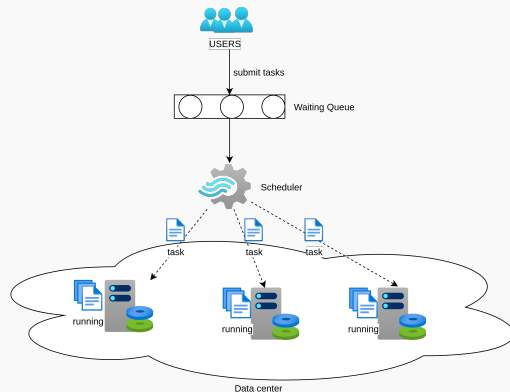
Kết luận

# Sự xuất hiện của Cloud Computing



## Luồng hoạt động

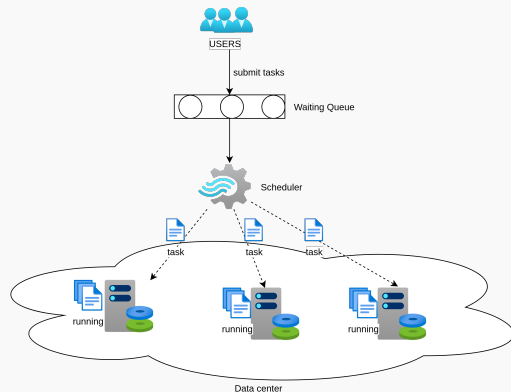
1. Người dùng gửi các tasks đến hệ thống
2. Các tasks được đưa đến hàng đợi cho đến khi được lập lịch
3. Bộ lập lịch tìm các máy tính phù hợp cho các tasks
4. Chuyển các tasks đến các máy tính và thực thi



Người dùng gửi tasks đến hệ thống

## Tại sao cần lập lịch?

1. Tiết kiệm chi phí nhờ sử dụng hiệu quả tài nguyên
2. Tăng chất lượng dịch vụ cung cấp cho người dùng



Người dùng gửi tasks đến hệ thống



## Hai kiểu lời giải phổ biến cho bài toán

- ▶ Có N tasks, M máy ảo, lời giải là cách ghép cặp task -> máy ảo cho N tasks.
- ▶ Có K nhóm của tasks, M máy ảo, lời giải là cách phân chia

$$\Theta = \{\theta_1, \dots, \theta_K\}$$

với

- ▶  $\theta_i$  là số lượng máy tính ảo được phân cho nhóm i
- ▶  $\sum_{i=1}^K \theta_i = M$



## Heuristic

Là những thuật toán đưa ra lời giải chấp nhận được, thỏa mãn mục tiêu và ràng buộc của bài toán

- ▶ Nhóm các thuật toán dựa trên quần thể như GA, PSO, ...
- ▶ Nhóm các thuật toán tìm kiếm lời giải tối ưu như Tabu Search, Beam Search

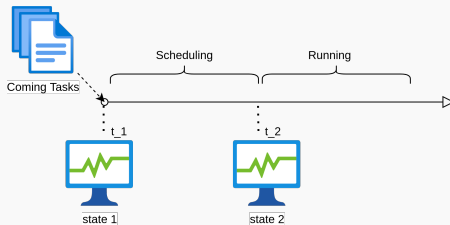
## Đặc điểm chung

- ▶ Tìm nghiệm tối ưu hàm  $\mathcal{F}$  nào đó trong miền không gian có ràng buộc
- ▶ Hàm  $\mathcal{F}$  sẽ được tính dựa trên các thông tin về hệ thống tại thời điểm lập lịch

# Sai số do độ trễ trong quá trình lập lịch



## Trạng thái tài nguyên



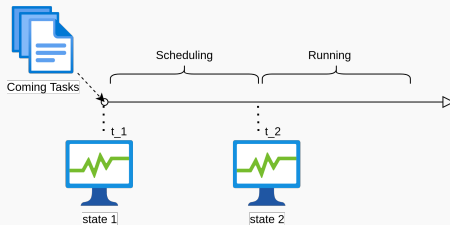
Trạng thái máy tính





## Trạng thái tài nguyên

- ▶ Tại thời điểm  $t_1$ 
  - ▶ available\_cpu =  $c_1$
  - ▶ available\_memory =  $m_1$
- ▶ Tại thời điểm  $t_2$ 
  - ▶ available\_cpu =  $c_2$
  - ▶ available\_memory =  $m_2$

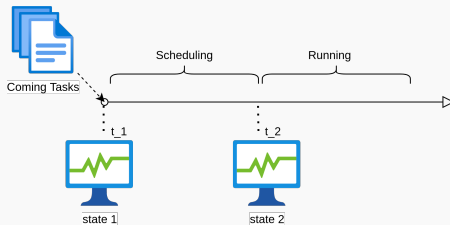


Trạng thái máy tính



## Trạng thái tài nguyên

- ▶ Tại thời điểm  $t_1$ 
  - ▶ available\_cpu =  $c_1$
  - ▶ available\_memory =  $m_1$
- ▶ Tại thời điểm  $t_2$ 
  - ▶ available\_cpu =  $c_2$
  - ▶ available\_memory =  $m_2$



Trạng thái máy tính

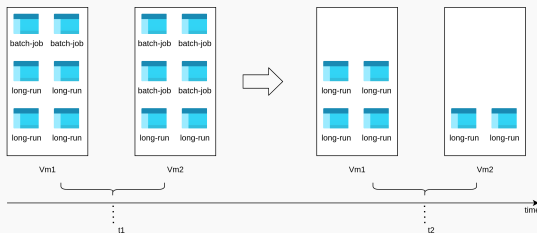
## Vấn đề 1

$$c_1 \neq c_2 \text{ hoặc } m_1 \neq m_2$$

# Mất cân bằng khối lượng công việc khi hoạt động



- ▶ long-run: các tasks dạng service luôn luôn chạy
- ▶ batch-job: các tasks thuộc dạng batch, thời gian thực thi từ vài trăm mili giây tới vài giây

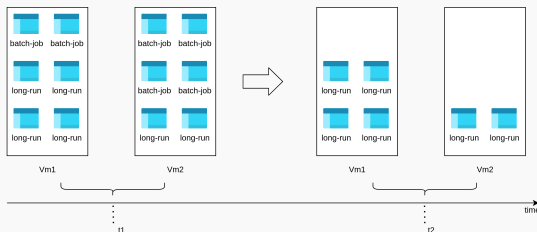


Mất cân bằng khối lượng tasks

# Mất cân bằng khối lượng công việc khi hoạt động



- ▶ long-run: các tasks dạng service luôn luôn chạy
- ▶ batch-job: các tasks thuộc dạng batch, thời gian thực thi từ vài trăm mili giây tới vài giây



Mất cân bằng khối lượng tasks

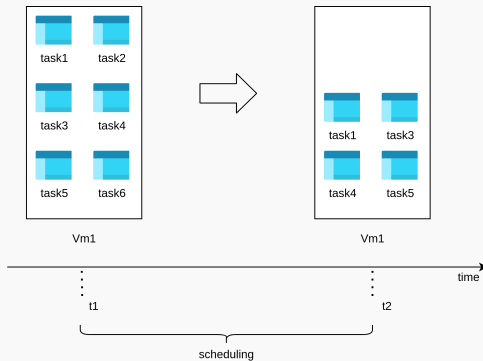
## Vấn đề 2

Các batch-job tasks kết thúc trong quá trình chạy phá vỡ trạng thái cân bằng khối lượng công việc giữa các máy ảo

# Ước lượng trạng thái các tasks

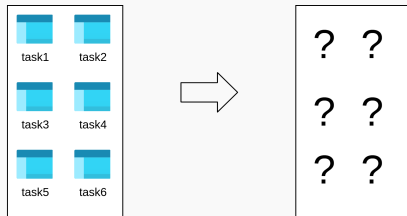


# Ước lượng trạng thái các tasks

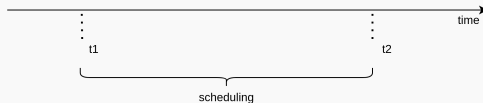


Trạng thái tài nguyên tại thời điểm lập lịch và thực thi

# Ước lượng trạng thái các tasks



$$P(\text{task hoạt động tại } t2 \mid \text{task hoạt động tại } t1) = ?$$



Trạng thái tài nguyên tại thời điểm lập lịch và thực thi

# Ước lượng tài nguyên khả dụng







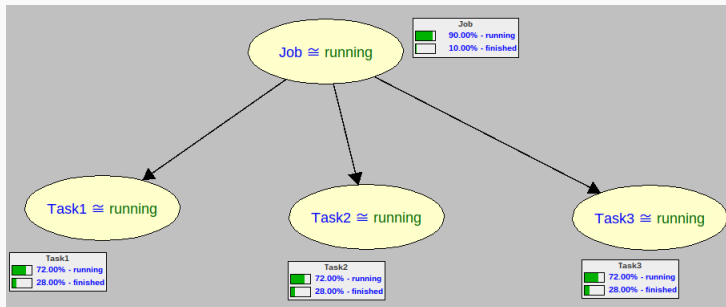
Tập các tasks đang chạy trên máy tính ảo tại thời điểm lập lịch

$$\mathcal{T} = \{task_1, task_2, \dots, task_K\}$$

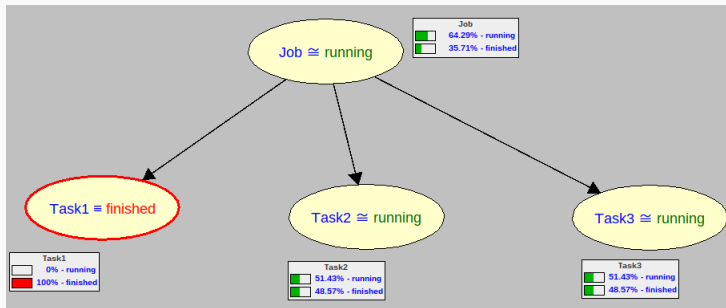
Ước lượng kì vọng tài nguyên khả dụng tại thời điểm thực thi

$$available\_resources = resources\_capacity - \sum_{i=1}^K resources\_usage_i \times p_i$$

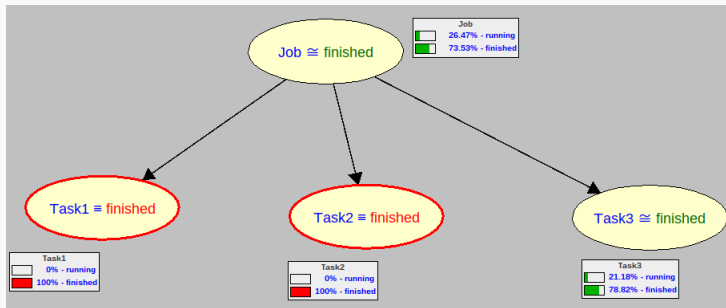
- ▶  $resources\_usage_i$  là tài nguyên  $task_i$  sử dụng
- ▶  $p_i$  là xác suất  $task_i$  còn hoạt động tại thời điểm thực thi



Ví dụ về mạng Bayesian cho 3 tasks



Ví dụ về mạng Bayesian cho 3 tasks



Ví dụ về mạng Bayesian cho 3 tasks

# Cân bằng khối lượng công việc giữa các máy ảo



Khối lượng công việc của M máy ảo

$$\mathcal{L} = \{l_1, l_2, \dots, l_M\}$$

Độ mất cân bằng

Để thể hiện độ mất cân bằng khối lượng công việc giữa các máy ảo, ta sử dụng phương sai của  $\mathcal{L}$

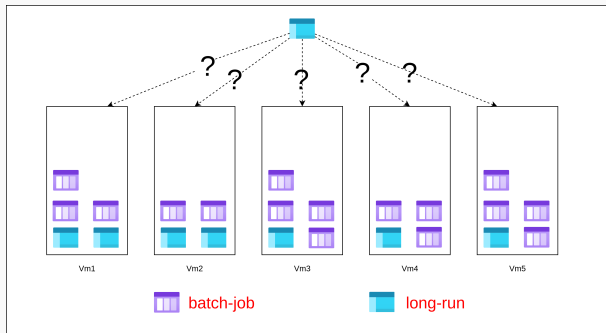
$$\mathcal{V}(\mathcal{L}) = \frac{1}{M} \times \sum_{i=1}^M (l_i - \bar{l})^2$$

# Lựa chọn máy tính ảo cho tasks



Tìm máy ảo cho long-run task:

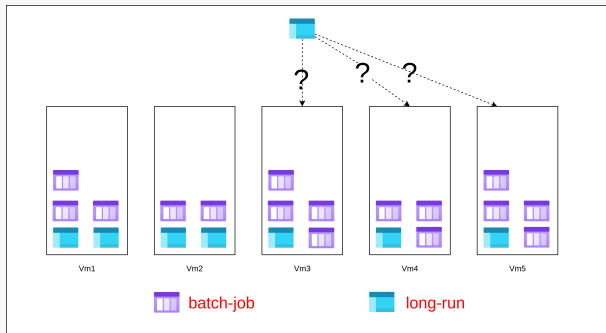
- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho long-run tasks nhỏ nhất
- ▶ Bước 2: Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ Bước 3: Cập nhật thông tin task được ghép cặp với máy ảo





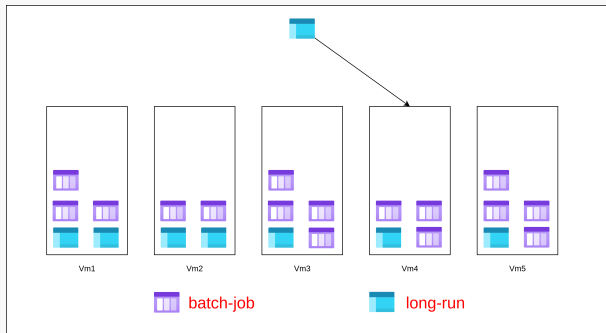
Tìm máy ảo cho long-run task:

- ▶ **Bước 1:** Chọn K máy ảo có lượng tài nguyên dành cho long-run tasks nhỏ nhất
- ▶ **Bước 2:** Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ **Bước 3:** Cập nhật thông tin task được ghép cặp với máy ảo



Tìm máy ảo cho long-run task:

- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho long-run tasks nhỏ nhất
- ▶ **Bước 2:** Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ Bước 3: Cập nhật thông tin task được ghép cặp với máy ảo



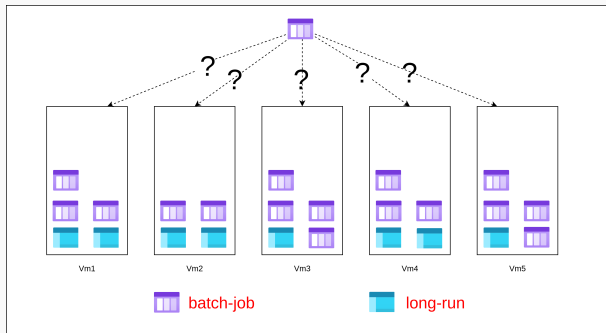
Tìm máy ảo cho long-run task:

- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho long-run tasks nhỏ nhất
- ▶ Bước 2: Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ **Bước 3:** Cập nhật thông tin task được ghép cặp với máy ảo



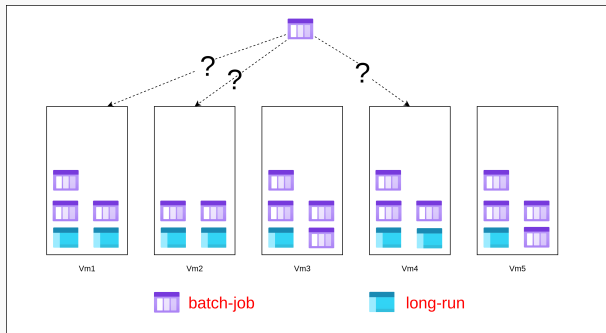
Tìm máy ảo cho batch-job task:

- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho batch-job tasks nhỏ nhất
- ▶ Bước 2: Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ Bước 3: Cập nhật thông tin task được ghép cặp với máy ảo



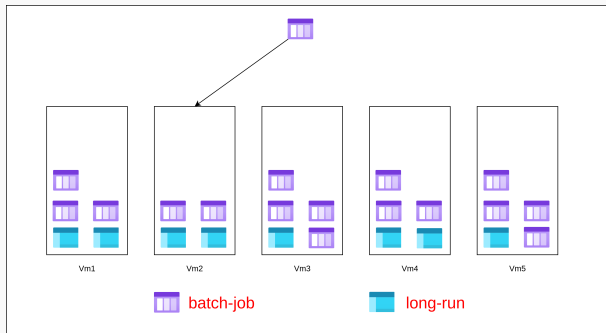
Tìm máy ảo cho batch-job task:

- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho batch-job tasks nhỏ nhất
- ▶ Bước 2: Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ Bước 3: Cập nhật thông tin task được ghép cặp với máy ảo



Tìm máy ảo cho batch-job task:

- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho batch-job tasks nhỏ nhất
- ▶ Bước 2: Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ Bước 3: Cập nhật thông tin task được ghép cặp với máy ảo

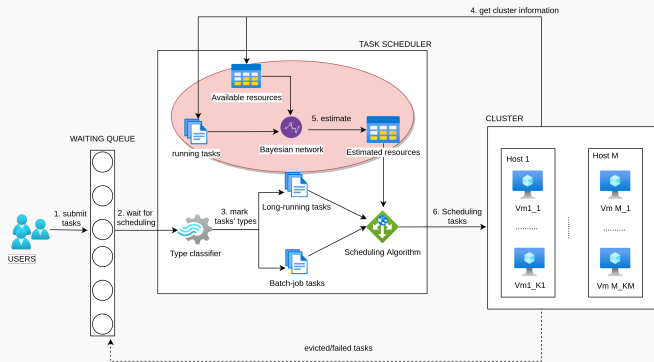


Tìm máy ảo cho batch-job task:

- ▶ Bước 1: Chọn K máy ảo có lượng tài nguyên dành cho batch-job tasks nhỏ nhất
- ▶ Bước 2: Trong K máy ảo tìm được, chọn máy có lượng tài nguyên khả dụng cao nhất cho task
- ▶ Bước 3: Cập nhật thông tin task được ghép cặp với máy ảo



- ▶ Dùng mạng Bayesian ước lượng tài nguyên khả dụng
- ▶ Dùng tài nguyên ước lượng để lập lịch cho tasks







## Mục đích

- ▶ Đánh giá hiệu năng so với Worstfit và FCFS
- ▶ So sánh hiệu quả của mạng Bayesian
- ▶ So sánh độ mất cân bằng trong thời gian chạy với Worstfit

## Kịch bản

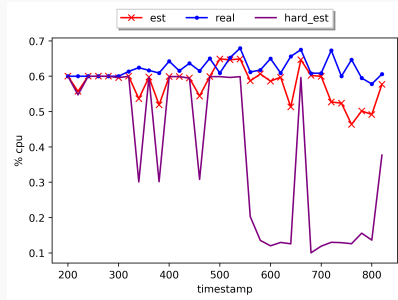
- ▶  $M = 10$  máy tính ảo
- ▶  $N = 15000$  tasks
- ▶  $K = 3$
- ▶  $T \sim \mathcal{P}(\lambda = 15)$
- ▶  $delay\_time \sim \mathcal{N}(\mu = 5, \sigma = 0.5)$

Kết quả về thời gian chạy của các tasks

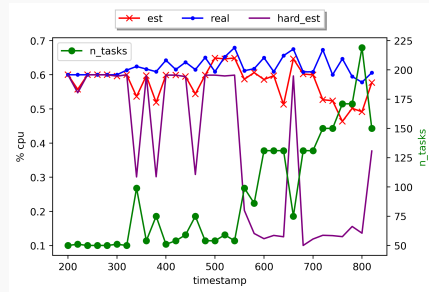
- ▶ Thuật toán đề xuất hoàn thành nhiều tasks hơn so với Worstfit và FCFS
- ▶ Thời gian trung bình hoàn thành một task tốt hơn FCFS là 50%, Worstfit là 20%

Running statistics over 1000s			
stats	FCFS	Worstfit	Resources balancing
count	13214	13925	14235
mean	10.62	6.34	5.42
std	54.24	33.37	27.31
min	0.31	0.12	0.21
50%	5.78	3.52	3.41
max	829.92	616.35	640.39

# So sánh độ chính xác của mạng Bayesian



(a) Tài nguyên khả dụng tại thời điểm thực thi



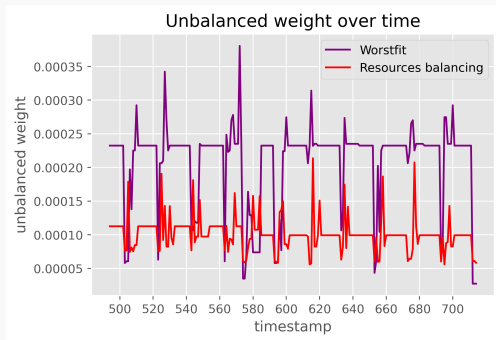
(b) Sai số với số lượng tasks đang chạy

Mạng Bayesian cho kết quả ước lượng (màu đỏ) có sai số với thực tế (màu xanh dương) nhỏ hơn so với việc dùng thông tin sai lệch (màu tím).

# So sánh mức độ mất cân bằng giữa các máy tính



- ▶ Độ mất cân bằng giảm sâu được thời điểm lập lịch (500, 520, ...) nhưng không duy trì được lâu
- ▶ Thuật toán đề xuất cho độ mất cân bằng ổn định và nhỏ hơn so với Worstfit



Độ mất cân bằng trong quá trình hoạt động





## Kết luận

- ▶ Thuật toán đã cải thiện được sai số giữa thời điểm lập lịch và thực thi
- ▶ Có thể cân bằng được khối lượng công việc trong quá trình chạy

## Định hướng phát triển

- ▶ Phát triển mô hình đồ thị Bayesian phù hợp với học liên tục
- ▶ Mở rộng bài sang bài toán resource scaling



Thank You For Listening!