Consolidated Review of

Choreo: Network-Aware Task Placement for Cloud Applications

1. Strengths

The problem the paper identifies of placing tasks for network intensive applications in the cloud is interesting and promising, and also appears to be new. The topic is novel and the approach is very reasonable. Evaluation on EC2 and Rackspace is nice.

The paper is well-written and easy to read. Skillful handling of most of the obvious issues. The technique appears to provide substantial benefit compared with reasonable benchmarks.

2. Weaknesses

The experimental results, while promising, did not thoroughly explore the space of many different users, many different types of applications, etc.

There is not much surprising novel results in the proposed methodology or the evaluations. The longevity of the work is unclear as cloud providers may also evolve in the way they manage the data center network resources. It is not clear how strong the results really are.

The networks measured in the paper are data center networks, not the Internet.

3. Comments

This is a nice paper, but it seems better suited to a cloud computing conference than IMC. The end of the introduction surprised me: normally batch / offline optimization gives better performance than online optimization where some decisions need to be made before all information is available. That made me expect that knowing all applications before having to place any would be better. It may be worth highlighting in the introduction that the reason that the real-time case is better is that the algorithm has better information about actual network performance in that case.

The paper takes a blackbox approach (which is justified due to a lack of visibility into how Cloud providers manage the data-center networks) to placing applications for ensuring good overall performance. However, if the Cloud providers also perform various optimizations or impose policies to rate-control traffic between VMs, such interaction may be suboptimal leading to unwanted interference. It's useful comment on why this would not be a problem for Choreo or how this can be prevented.

The paper also makes assumption that there is no internal optimization such as VM migration that would impact the original measured bandwidth from accurately representing the future expected throughput between VMs. Such potential limitations should be stated. It is not clear what the ground truth is in Figures 6 and 7 where the packet train based measurements are evaluated for its accuracy.

The paper reads well so I'm going to focus on a few areas where I felt the paper didn't quite hit the right issues.

- 2.1 "Choreo models the inter-task network needs of an application by profiling the number of bytes sent, rather than the rate observed. " the use of the number of bytes sent for profiling applications to describe the network demand can be incomplete, as it doesn't capture the time dimension of how often the communication needs to occur, e.g., a batch job vs. interactive job with the same amount of the data exchanged have very different network bandwidth requirements.
- Given network-awareness may not be the only metric, e.g., CPU, memory, storage resources are also factors that Cloud users want to optimize for the overall task completion time, the authors need to comment on how the proposed scheduling algorithm can be extended to consider these other resource factors.
- Do authors account for the cost of doing the measurements needed to run Choreo when evaluating gains? - What happens if there are many users using Choreo?
- How does Choreo handle more interactive applications (e.g., a web server)?
- Handling an application that changes over time: is there a risk of thrashing? What happens if CPU needs change but network needs don't change? What happens if there are many applications in use in the network, such that application network needs are always changing?
- It would have been interesting to see some results for real, network intensive applications, versus artificially created ones.
- Section 3.2, the assumption that c1/c2-1 other connections exist assumes fairness. But Jacobson showed in the early 1990s that TCP does not divide the pipe fairly, there tend to be winning and losing connections.... So why should I believe this formula?
- Figure 4(b), there is a pretty big gap at 1s. Any ideas what is causing that?
- The equation at the end of Section 4.1 seems to mix packets (the first option in the min) with bytes (the second option). Measurements based on packet pairs typically suffer from problems with interrupt coalescence. It would be interesting to hear why this is not a problem in this case. On page 6, the first two rules seem to have the same "if" condition, but phrased differently. Is there a difference between "at the link" and "on the link"? If so, please make it clearer. If not, please use the same preposition. In Fig 6, it would be useful to consider bursts between 500 and 2000 packets; it seems that a much better tradeoff may be possible in that range. When reading the performance figures at the end of Section 6.2, I initially compared the 8~13% slowdown with the median 7~15% improvement, and concluded that Choreo is not very useful. Perhaps make clearer that the fair comparison is between the 8~13% slowdown and the 13~28% speedup given that improvement occurs.
- Section 4.1 -- why use packet trains instead of packet pairs? Packet pairs have errors, but it requires many less packets and presumably could be done more swiftly? These two points also raise an issue -- implicit in much of the paper is that accurate measurements are important to placement -- how important? If you're wrong by 25%, do you change placements in damaging ways??? You've shown 10%-40%

improvement -- if doing one or two simpler measures gets 10-25% improvement, is that OK? Also, many of the measurements suggest that EC2 and Rackspace are sufficiently provisioned in network capacity -- you don't see bottlenecks and everyone's given about 300Mb/s in Rackspace.

What happens if demand shoots up such that EC2 or Rackspace is a bit short on capacity? I'm reminded of Eric Schmidt's comment that running data centers is much like running the old mainframe centers -- and what killed mainframe centers wasn't their usual performance -- it was when corporations deferred upgrades and the computer got overloaded and suddenly everything ran too slowly. It seems to me the work in this paper is really targeted at the loaded environment, and we don't have a loaded environment. (Side note -- I still gave the paper a high rating, as achieving better performance in an unloaded environment is certainly of interest).

4. Summary from PC Discussion

Strengths:

- Network-aware task placement for cloud apps is an important problem and this is the first in-depth investigation of this problem.
- The study is systematically carried out, The paper is well written. The benefit from the evaluation is clear for EC2 and Rackspace.

Weaknesses:

- The focus of bandwidth-intensive apps is somewhat narrow: without considering different types of apps and user requirements.
- The measurement methodology is fairly simple without introducing any new techniques.
- It's questionable whether the approach will work in the long run (as it makes assumptions on the stability of the network bandwidth availability)

Overall the reviews are mostly positive, as the work is carried out in real optional cloud services and did demonstrate some benefit.

There is some issue with how useful the approach can be as it heavily depends on the variability of network resources. I also believe that ultimately there should be some kind of network QoS or SLA that we can obtain from cloud service providers.

5. Authors' Response

Choreo is a first step at tackling the problem of improving application performance in cloud networks, and is intended to be used by clients in the event that cloud providers do not offer SLAs (though it could also be used in conjunction with SLAs). While our main focus is on bandwidth-intensive applications, we discuss how Choreo's optimization process could be extended to take into account other user requirements (Section 5).

Choreo is not appropriate for every application, nor for every cloud environment. For instance, short-lived applications and interactive applications are likely not well served by Choreo. We have added a section (Section 7.1) detailing some of the fundamental limitations of Choreo.

It is also hard to determine how Choreo will perform in future cloud networks. We believe that Choreo's general approach---of measuring the network and adapting to it---will still be useful as cloud networks change, since the process begins with measurement (and so doesn't rely on particular assumptions about the amount of bandwidth available). We have added a discussion (Sections 7.2 and 7.3) of how Choreo might work in different types of network environments, including environments that impose rate controls and VM migration, and with different types of applications.

In response to the individual reviews, we have made numerous edits to the text, including: clarifying the reason behind Choreo's improvements on real-time applications vs. batch jobs (Section 6.3), and its performance improvements in general (Section 6.2); condensing the bottleneck-location algorithm (Section 3.3.2); explaining our choice of packet trains over packet pairs (Section 3.2); clarifying how figures 6 and 7 were created (Section 4.1); detailing our dataset (Section 6.1); and explicitly accounting for the cost of measurement in Choreo (Sections 6.2 and 6.3).