TRLMS: Two-Stage Resource Scheduling Algorithm for Cloud Based Live Media Streaming System

We propose an efficient Two-stage Resource scheduling algorithm for cloud based Live Media Streaming system (TRLMS). It transforms the cloud-based resource scheduling problem to a min-cost flow problem in a graph, and solves it by an improved successive Short Path (SSP) algorithm. Simulation results show that TRLMS can enhance user demand satisfaction by 17.1% than mean-based method, and its time complexity is much lower than original SSP algorithm.

**SUMMARY**
This letter proposes an efficient Two-stage Resource scheduling algorithm for cloud based Live Media Streaming system (TRLMS). It transforms the cloud-based resource scheduling problem to a min-cost flow problem in a graph, and solves it by an improved successive Short Path (SSP) algorithm. Simulation results show that TRLMS can enhance user demand satisfaction by 17.1% than mean-based method, and its time complexity is much lower than original SSP algorithm.

**key words**: cloud computing, live media streaming, min-cost flow

1. Introduction

Live Media Streaming (LMS) has become one of the most popular applications over the Internet, and today’s globalized LMS system is faced by highly dynamic user demands from all over the world [1], the fluctuation of user demands in the over-provision and under-provision problem, and poses a big challenge to schedule massive resources for globalized demands with time/region diversities in cloud.

A proper LMS system can leverage cloud computing to lower schedule cost and serve more user demands. There are several research efforts being made to address the problem described above. With the existing mean-based methodologies (e.g., that in [1]–[4]) and stochasticity-based ones [5], some challenging issues still remain open: (1) Considering stochastic demands in smaller time scale can help satisfy more demands from users than methods only consider mean value of demands, but it will lead to a high computational complexity of resources scheduling. (2) Content distribution is essential to guarantee QoS (Quality of Service) in a LMS system, but it is not properly considered in existing stochasticity-based methods.

Our letter mainly makes two contributions: (1) We formulate the problem setting for the general model of live media system, and transform it into a min-flow problem. (2) We inspect the inner structure of the SSP-based solution, and then propose an efficient Two-stage Resource scheduling algorithm for cloud based Live Media Streaming system (TRLMS), resulting in reducing the time complexity significantly. Compared with mean-based method, TRLMS increases the percentage of demands satisfied (PDS) by 17.1%, with low computation complexity as well, which will be analyzed later in Sect. 3.

2. System Model

The general model we consider can be viewed as the abstraction of typical cloud based LMS system, such as IPTV network of AT&T [6]. As shown in Fig. 1, the globalized LMS system utilizes server resources in C geographically dispersed data centers (represented by center$^c$, $1 \leq c \leq C$) from one or more cloud platforms, and users in A regions around the world connect to these data centers to watch I video channels. As in popular cloud platform, these C data centers are connected via high-bandwidth backbones. Similar with typical IPTV network [6], regions can be metropolitan areas, requests in one region are redirected to nearest 1-2 data centers, in other words, each data center may cover one or more regions. Complying with the coverage in request dispatching is helpful to guarantee QoS. The coverage is denoted as $U_{a,c}$ $(1 \leq a \leq A)$, with $U_{a,c} = 1$ presents that center$^c$ covers region$^a$, or $U_{a,c} = 0$ for not. In each center$^c$, $(1 \leq c \leq C)$, the cost of one VM is $e_i$ and one VM can satisfy $h_i$ resource demands concurrently. For next T time slots, in time slot t $(1 \leq t \leq T)$, the number of VM scheduled to channel$^i$, $(1 \leq i \leq I)$ in center$^c$ is $Z_{c,i,t}$ and the proportion of $Z_{c,i,t}$ used for region$^a$ is $U_{a,c}$. Same to the model in [5], assume that during time slot t, the demands of channel$^i$ from region$^a$ is represented by the same random variable $D_{a,i,t}$, and the demands from all regions is denoted as $D_{a,i}$. In the mean-based methods in previous work [1]–[4], to satisfy given demands, the target can be minimizing the resource cost within T time slots, as shown in Eq. (1), where $Z_{c} = \sum_{t=1}^{T} \sum_{i=1}^{I} Z_{c,i,t}$, $E(D_{a,i,t})$ is the expected value of stochastic variable $D_{a,i,t}$; the second constraint signifies that all allocated resources in one data center will be used to...
satisfy expected demands; the third constraint presents that coverage constraint is imposed on request dispatching.

\[
\min \left\{ \sum_{c=1}^{C} e_{Z_c} \right\} \quad s.t. \quad \sum_{a=1}^{A} u_{i,a,c,t} = 1 \quad \forall i, a, c, t \\
\sum_{c=1}^{C} \sum_{a=1}^{A} z_{i,a,c,t} \leq U_{a,c} \quad \forall i, a, c, t
\]

(1)

The mean-based scheme like [1]–[4] may be inefficient, but one can maximize the satisfied demands while accounting for the full demand distribution [5]. Based on the coarse-grained scheduling by mean-based method in the initial stage, we can further adjust the scheduling to satisfy more demands by considering stochasticity of demands.

Set \( z_{i,a,c,t} \) and \( u_{i,a,c,t} \) as the variable containing the adjusted value of \( z_{i,a,c,t} \) and \( u_{i,a,c,t} \). \( m_{i,a,t} = \sum_{c=1}^{C} h_{i,a,c} \) and \( m_{i,t} = \sum_{a=1}^{A} m_{i,a,t} \). Then the realization of stochastic demand \( d_{a,t} \) and \( d_{i,t} \) for any channel \( i \), the satisfied demands meeting coverage constraint is \( \text{sat}_{i}^{cov} = \sum_{c=1}^{T} \sum_{a=1}^{A} \min (m_{i,a,t}, d_{i,t}). \) And in online services, extra requests for one channel can be redirected to data centers violating coverage constraints, the number of all satisfied demands (including both violating and meeting coverage constraint) is \( \text{sat}_{i}^{all} = \sum_{c=1}^{T} \min (m_{i,a,t}, d_{a,t}) \). The target in a globalized LMS system can be written as the weighted summation of \( \text{sat}_{i}^{all} \) and \( \text{sat}_{i}^{cov} \), then rewritten as the weighted summation of \( \text{sat}_{i}^{all} \) and \( \text{sat}_{i}^{cov} \) for all channel; \( R = \sum_{c=1}^{T} \left( w_1 \text{sat}_{i}^{all} + w_2 \text{sat}_{i}^{cov} \right) \). And according to [5], for random variable \( x \) and constant \( N \), if \( x \) is discrete non-negative, it has \( E(\min(x,N)) = \sum_{n=1}^{N} \Pr(x \geq n) \), then the expected revenue for stochastic demand can be further written as follows:

\[
E(R) = \sum_{i=1}^{T} \sum_{n=1}^{m_{i}} \left( \sum_{i=1}^{m_{i}} w_1 \Pr(d_{i,t} \geq n) + \sum_{a=1}^{A} m_{i,a,t} w_2 \Pr(d_{i,a,t} \geq n) \right)
\]

And the revenue can be transformed to expected cost:

\[
E(C) = \sum_{i=1}^{T} \sum_{n=1}^{m_{i}} \left( \sum_{i=1}^{m_{i}} w_1 \Pr(d_{i,t} \leq n) + \sum_{a=1}^{A} m_{i,a,t} w_2 \Pr(d_{i,a,t} \leq n) \right)
\]

(3)

Then the optimization problem can be stated as Eq. (4), where constant \( Z_c \) is the amount of server resources in center \( c \). Under the constraint in Eq. (4), \( E(C) + E(R) = (w_1 + w_2) \sum_{i=1}^{C} h_{i}Z_{i,c} \) and minimal \( E(C) \) is equal to the maximal \( E(R) \). To solve the problem, we need to find the optimal \( m_{i,a,t}, z_{i,a,t}, u_{i,a,c,t} \), which will be given in the next section.

\[
\min(E(C)) \quad s.t. \quad \sum_{i=1}^{T} \sum_{n=1}^{m_{i}} z_{i,a,c,t} = Z_c \quad \forall c \\
\sum_{i=1}^{T} \sum_{n=1}^{m_{i}} u_{i,a,c,t} = 1 \quad \forall i, c, t \\
\sum_{i=1}^{T} \sum_{n=1}^{m_{i}} z_{i,a,c,t} \leq U_{a,c} \quad \forall i, a, c, t
\]

(4)

\[ \text{3. Algorithm} \]

To solve the problem, we construct a layered graph \( G_o = (V,E) \): (1) \( G_o \) consists of 8 layers (represented as \( L_1 \) to \( L_8 \)) with the node in \( L_n \) marked as \( v^t \), and there is no directed edge between nodes in the same layer. (2) \( G_o \) has one source node \( v^1 \) and one sink node \( v^8 \). (3) Each edge \( e \) has capacity \( |c(e)| \) and weight \( \omega(e) \), the flow through edge \( e \) is \( f(e) \) with flow value \( |f(e)| < |c(e)| \). (4) \( G_o \)’s structure is as follows: \( L_1 \) and \( L_8 \) contains the source node \( v^1 \) and sink node \( v^8 \), respectively; \( L_2 \) contains \( C \) nodes \( (v^2) \); \( L_3 \) contains \( C \times T \times I \) nodes \( (v^3_{i,t,c}) \) and the flow value into \( v^3_{i,t,c} \) is represented using \( h_{i,c} z_{i,t,c} \); \( L_4 \) contains \( A \times T \times I \) nodes \( (v^4_{i,t,c}) \) and the flow value into \( v^4_{i,t,c} \) is represented using \( m_{i,a,t} \); \( L_5 \) contains \( A \times T \times K \) nodes \( (v^5_{i,t,a}) \); \( L_6 \) contains \( T \times I \) nodes \( (v^6_{i,c}) \) and flow value into \( v^6_{i,c} \) is represented using \( m_{i,a,c} \); \( L_7 \) contains \( T \times I \times K \) nodes \( (v^7_{i,t,a,c}) \).

Then for a flow from \( v^1 \) to \( v^8 \), its value is \( f = \sum_{i=1}^{T} \sum_{c=1}^{C} \sum_{t=1}^{T} h_{i,c} z_{i,t,c} = \sum_{i=1}^{T} \sum_{a=1}^{A} \sum_{c=1}^{C} m_{i,a,t} \), and through setting appropriate \( w(e) \), the flow cost can be \( E(C) \). In all feasible flows from \( v^1 \) to \( v^8 \) with same flow value \( \sum_{c=1}^{C} h_{i,c} c \), the minimal-cost flow contains the solution to Eq. (4). We find the flow by Successive Short Path (SSP) algorithm, which is composed of many iterations. In each iteration, a residual graph \( G^* \) is constructed from \( G_o \), and a shortest path (with the lowest weight) is calculated from \( G^* \), then the shortest path is updated to \( G_o \) as a flow with value of \( 1 \). The iteration ends when targeted flow value in \( G_o \) is achieved. The structure of graph implies that not all weighted edges need to be considered in every iteration of SSP algorithm. As a result, we can obtain lemma 3.1.

Lemma 3.1. For each iteration of SSP algorithm in \( G_o \) in the residual graph, the optimal path between \( v^1 \) and \( v^8 \) is composed of these intermediate shortest path:

\[
\begin{align*}
&v^1 \Rightarrow v^3_{1,t,c} \Rightarrow v^4_{1,t,a} \Rightarrow v^5_{1,t,a} \Rightarrow v^6_{1,c} \Rightarrow v^7_{1,t,a,c} \Rightarrow v^8 \\
\end{align*}
\]

Herein \( v_{1,t,a,c} \) means the shortest path in multiple simple paths between nodes \( v_1 \) and \( v_2 \) in residual graph.

It is shown in lemma 3.1 that any shortest path is composed of intermediate shortest path between nodes in layer 1/3/4/6/8, only these layers are required to apply SSP algorithm. We can compress original graph \( G \) to a simplified 5-layer graph \( G \) (represented as \( L^1 \) to \( L^5 \)) with the node in \( L^8 \) marked as \( v^6 \): \( L_1 \) and \( L_5 \) contains the source node \( v^1 \) and sink node \( v^8 \), respectively; \( L_2 \) contains \( C \times T \times I \) nodes \( (v^2_{i,t,c}) \),
and the flow value on the edge from ˆv1 to ˆv3 is represented by h. Zc, L3 contains A × T × I nodes (ˆv3i,t,j), and there is an edge from ˆv3i,t,j to ˆv3i,t,j when Ua,c = 1; L4 contains T × I nodes (ˆv3i,t), and the flow value on the edge from ˆv3i,t to ˆv3i,t is mia,t, the value of the flow from ˆv3i,t to ˆv3i,t is mi,t.

Then in TRLMS, the amount of resources in each data center Zc is first estimated using mean-based algorithm, and then in TRLMS, an improved SSP algorithm is applied to get the optimized solution for stochastic demand. The details of TRLMS are given in Algorithm 1. Stage one (Line 1) is to estimate the amount of resources in each data center. In each iteration of stage two, a residual graph G′ is constructed from G (Line 4-8), and the flow through shortest path of G′ is updated to G (Line 10-15). The iteration ends with optimal solution (Line 18) when K is achieved, or with no solution (Line 16).

In all three scenarios, the demands for one channel follows a Zipf distribution and skewness exists in demand distribution, which is consistent with prior works [5]. By Zipf distribution, given a number e > 0, the probability for a single request to demand type-i resource is pi = i−e/(∑j=1,e j−e). Assuming that the total demands in regiona are proportional to the region’s population, i.e., the probability for a request to originate from regiona is qa. Therefore, the expected number of demands for all channels from all regions is denoted as λ, then Dia can be simulated by a Poisson distribution with a rate of piqaλ. According to common periodical fluctuation pattern, to approximate demand fluctuation in period of one day, we use sine function to compute the demands of one hour (Dia,t), i.e., for hour t, E(Dia,t) = (1 + sin(2πt/24 + Δ))piqaλ, where Δ is added to make ∑Tt=1 E(Dia,t)/T = piaqaλ (∀i,a). To be consistent with the work in [1], [2] where part of demands are satisfied by cloud and the remaining demands are satisfied by content distribution network, the number of resources in cloud is set to satisfy approximately half of all demands, i.e., for any c, Zc = Zc/2, where Zc is the solution to Eq. (1).

In all three scenarios, we set wSat = 1 and wCoe = 2, with \( h_c = 1 \) for all c, and we compare PDS of TRLMS with that of mean-based optimization algorithm, whose result can be accurately the same with the result of stage one in TRLMS.

4. Experiments

To verify the effectiveness of TRLMS in terms of percentage of demands satisfied (PDS) with given resource, as that in [5], we compare TRLMS with mean-based algorithm for our problem (stage one of TRLMS). The source code of experiment is available for free academic use at [7]. Since revenue can be viewed as weighted summation of the expectation of satisfied demands, and the upper bound of revenue is achieved when all demands are satisfied, then given a scheduling result, PDS is defined as the ratio of current revenue to the upper bound revenue. We choose 150 main cities all around the world with population and position information, regions and data center can be chosen from these 150 cities. The number of data centers C = 30, number of regions A = 150 and default number of channels I = 500. Since the value of C and A approximate that in real-world LMS system, and during experiment we found the result is not mainly affected significantly by C and A. Due to space limitation, we just present the effect of these two parameters with constant C and A.

Algorithm 1: The steps of TRLMS

1. Get Zc using mean-based algorithm.
2. Construct 5-layer graph Go initialize the flow between ˆv1 and ˆv5 with flow value \(|f|\) = 0.
3. While \(|f| < K = \sum_C h_cZc\) do:
   4. Get empty graph G′ and add all nodes of G to G′.
   5. For each edge e in G do:
      6. Add edge e to G′.
      7. If \(|f(e)| > 0\) then add reversed edge e′ to G′.
   8. End
   9. If the shortest path exists from ˆv1 to ˆv5 in G′ then:
      10. For each edge e in shortest path do:
          11. If e ∈ G then \(|f(e)| ← |f(e)| + 1\) in G else if the reversed edge e′ ∈ G then \(|f(e′)| ← |f(e′)| − 1\) in G.
      12. End
      13. \(|f| ← |f| + 1\).
   14. End
   15. Else exits with no solution
   16. End
   17. Extract the optimal solution from G.
In scenario 1, we plot all PDSs with $T = 1/12/24$, $Zipf = 1.0$ with skewness $S$ ranges from 1.0 to 5.0. In scenario 2 and 3, the mean-based algorithm (with $S = 5.0$) is compared with TRLMS (with $S = 1.0, 3.0, 5.0$, respectively); while in scenario 2, $Zipf = 1.0$ and the number of channels $I$ ranges from 100 to 1000; In scenario 3, $I = 500$ and $Zipf$ is in a broad range from 0.6 to 1.4.

The results of the three scenarios are shown in Figs. 2, 3 and 4, respectively. From these figures, We can see that:

1. compared to mean-based methods, TRLMS can effectively increase satisfied demands by 17.1% at most (the line marked by solid diamond and circle, respectively, in Fig. 2).
2. the effect of TRLMS is stable and not affected by number of channels and Zipf parameter, since as shown in Figs. 2 and 3, PDS does not increase as number of channels and Zipf parameter increase. That means TRLMS can be used with different number of channels and popularity distribution (i.e., Zipf parameter).
3. the skewness in demand distribution is utilized by TRLMS to satisfy more demands. Figures 3 and 4 confirm the conclusion as skewness increases.

5. Conclusion

In the letter, we propose a two-stage resource scheduling algorithm (TRLMS) for cloud based live media streaming system, we verify that compared with traditional mean-based algorithm, TRLMS can increase percentage of satisfied demands up to 17.1%, with lower enough computation complexity. As a result, TRLMS can be used as a meaningful tool for globalized live media streaming system in cloud.

Acknowledgments

This paper is supported by the National Natural and Science Foundation of China (No.61003052, 61103007), Natural Science Research of the Education Department of Henan Province (No.2010A520008, 13A413001, 14A520018), Henan Provincial Key Scientific and Technological Plan (No.102102210025), Program for New Century Excellent Talents of Ministry of Education of China (No.NCET-12-0692), Doctor Foundation of Henan University of Technology (No.2012BS011, 2013BS003).

References

[1] F. Wang, J. Liu, and M. Chen, “Calms: Cloud-assisted live media streaming for globalized demands with time/region diversities,” Proc. IEEE INFOCOM, pp.199–207, 2012.
[2] Y. Wu, C. Wu, B. Li, X. Qiu, and F.C. Lau, “Cloudmedia: When cloud on demand meets video on demand,” 31st International Conference on Distributed Computing Systems (ICDCS), pp.268–277, 2011.
[3] X. Qiu, H. Li, C. Wu, Z. Li, and F.C. Lau, “Cost-minimizing dynamic migration of content distribution services into hybrid clouds,” Proc. IEEE INFOCOM, pp.2571–2575, 2012.
[4] X. Qiu, H. Li, C. Wu, Z. Li, and F.C. Lau, “Dynamic scaling of vod services into hybrid clouds with cost minimization and qos guarantee,” 19th International Packet Video Workshop (PV), pp.137–142, 2012.
[5] Y. Rochman, H. Levy, and E. Brosh, “Resource placement and assignment in distributed network topologies,” Proc. IEEE INFOCOM, pp.1914–1922, 2013.
[6] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. Ramakrishnan, “Optimal content placement for a large-scale vod system,” Proc. 6th International Conference, p.4, 2010.
[7] “Source code of experiment,” http://users.eecs.northwestern.edu/~yuhong/trlms.html, 2014.