



一种混合云环境下 SLA 约束的云资源调度方法

徐晓峰¹, 徐劲松²

(1. 中国移动通信集团江苏有限公司, 江苏 南京 210003;

2. 南京邮电大学通达学院, 江苏 南京 210003)

摘要:混合云提供了一种本地资源不足以向公有云申请资源的灵活做法,但使用哪一个公有云服务或者如何选择公有云服务需要仔细斟酌。提出一种基于 SLA 约束的云资源调度方案。该方案将向外部云调度的关键 SLA 度量看作用户的满意度和最低代价,并将这些 SLA 分量进行量化比较,以动态调整内部云资源无法服务而需要向外调度资源的情况。实验结果表明,该方案能够实现成本和实时性的最大满意度。

关键词:云计算;资源管理;高效调度;服务等级协议

中图分类号:TP338.8

文献标识码:A

doi: 10.11959/j.issn.1000-0801.2016054

An SLA-based resource scheduling method in hybrid cloud environment

XU Xiaofeng¹, XU Jinsong²

1.China Mobile Group Jiangsu Co., Ltd., Nanjing 210003, China

2.Tongda College of Nanjing University of Posts & Telecommunications, Nanjing 210003, China

Abstract: Hybrid cloud provides a flexible approach to the application of local resources to the public cloud. A resource scheduling scheme with the minimum cost of external cloud resource constrained by SLA was presented. The key SLA measure of the outer part of the cloud was considered as the user's satisfaction and the lowest cost. Then, the SLA component was quantized to dynamically adjust the internal cloud resource to be unable to service and need to dispatch resources. Experimental results show that the scheme can achieve the maximum satisfaction of cost and real-time performance.

Key words: cloud computing, resource management, efficient scheduling, service level agreement

1 引言

互联网上爆炸式增长的行业数据处理需求的出现,要求企业在软硬件以及服务支持上做更大的投资。云计算提供了一种替代方案来取代企业自身购买软硬件和基础设施

施构建大型数据中心的传统做法。但迁移到云计算环境并不意味着企业会抛弃原有的基础设施投资,通常企业会将关键数据、关键应用任务部署到私有云环境中替代原有的架构。根据效益最大化的原则,当私有云不足以满足企业计算需求时,有必要购买外部公有云的服务以满足溢出的

收稿日期:2015-11-20;修回日期:2016-01-07

基金项目:江苏省自然科学基金资助项目(No.BK2009426);江苏省高校自然科学基金资助项目(No.14KJD520005)

Foundation Items: The Natural Science Foundation of Jiangsu Province(No. BK2009426), The Natural Science Fund of Higher Education of Jiangsu Province(No.14KJD520005)

计算需求。这种包含私有云和公有云的混合模式称为混合云(hybrid cloud)^[1,2]。在这种情况下,可以将企业自有的私有云看作企业的固定成本,而即买即用的外部公有云服务就是企业可能支付的可变成本。为了达到企业收益的最大化,在私有云资源有空余的情况下,尽量将计算限定在私有云完成,只有自身资源不足以满足现在计算需求时,才使用外部的公有云服务。

为了管理私有云与公有云组成的混合云,参考文献[3]针对私有云与混合云提出了虚拟基础设施管理的解决方案,其关键组件在混合云实现。参考文献[4]将混合云中的负载分为基本负载和进入负载,基本负载是处理私有数据所产生的负载,使用公有云处理进入负载,将一般的负载问题转化为一个超图分割问题^[9]进行求解。参考文献[6]在 Hadoop MapReduce 中使用遗传算法对任务进行调度。参考文献[7]使用 SOA 架构的调度方法,通过定义成本函数实现最小代价的调度。参考文献[8]着重于稳定性、灵活性和安全的负载均衡实现。参考文献[9]在网格中通过测量 QoS 来实现启发式的调度。以上方法都具有一定的优越性,但要么并不适合在混合云环境中应用,要么很难实现 SLA 约束。

本文提出一种称为“混合云环境中 SLA 约束的自适应调度”(adaptive scheduling with SLA satisfaction in hybrid cloud environment, ASS)算法,以解决混合云的调度问题。通过执行时间的预判和动态规划的方法,调度算法可以实现 SLA 约束下对资源的最佳配置;算法试图最大化私有云的资源利用率,仅在资源不足的情况下向公有云请求调度,实现该算法的经济性。实验结果表明,该算法能够有效降低子任务的等待、执行时间,较好地保障了云资源调度的经济性。

2 ASS 算法的设计和建模

2.1 混合云中 SLA 约束下的自适应调度

为保证混合云环境下请求的 SLA,首先处理私有云的负载问题。算法试图将最主要的工作交由私有云完成,也即在相同的条件和常规操作成本下,尽最大可能将工作部署给私有云,以避免额外费用的产生。为了最大化私有云的资源利用率,该问题可以看作一个 NP 完全的背包问题。使用执行时间估计和动态规划可以实现适当的抢占式调度,得到一个近似最优解。

当私有云已经达到最大负载时,新的计算请求将使用外部资源完成,这时混合云可以处理有限的 SLA 约束下

的运算调度,避免 SLA 违约。新的请求将动态部署到公共云中,同时算法需要保证用户支付的费用维持最小的代价。本文通过成本函数计算租用的公共资源代价,该问题也可以映射到一个 NP 完全问题,使用动态规划,可以解出一个最小成本的近似最优解。

该混合云环境下的调度系统架构如图 1 所示。用户通过代理或者其他方式向私有云请求资源,允许用户对于应用的性能、成本等偏好进行选择。调度系统部署在私有云中的主节点。

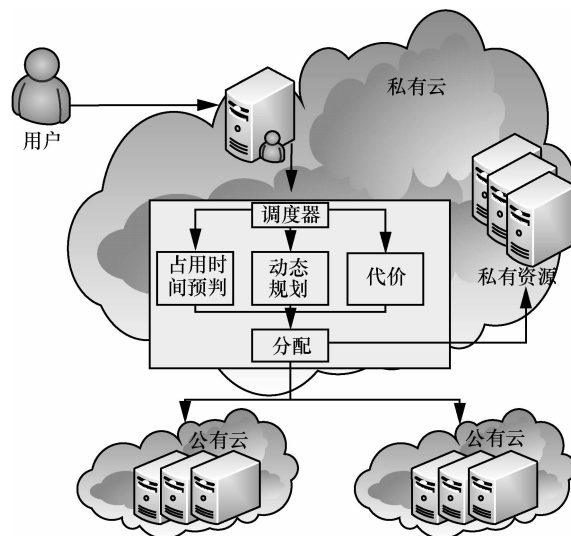


图 1 混合云环境调度示意

调度系统沿用 Hadoop 的做法,初始化可调度资源为固定大小的插槽(slot),这些插槽有一些共同的信息,如可用数据大小、可执行代码大小以及数据本地化等。当一个新的任务请求到达时,调度器就被触发。调度器首先将请求的作业分为可并行执行的任务,然后用插槽的大小来估算每个任务的执行时间和调用外部云资源的代价,最后使用动态规划的方法通过成本和调度时间的约束进行调度。调度算法试图在满足 SLA 约束的情况下实现成本最优。子任务可能分配到私有云,也可能分配到公有云。

2.2 形式化描述

为了更好地说明该自适应调度算法,将具体的 SLA 目标抽取出来进行定义。

定义 1 定义用户请求的任务 i 为 $Task_i = \{V_{i1}, V_{i2}, \dots, V_{in}, \dots, V_{ij}\}, 1 \leq n \leq j$, 其中 V_{in} 表示任务 i 的第 n 个子任务。

定义 2 定义用户请求的基本单位为子任务,第 i 个任务的第 j 个子任务表示为 $V_{ij} = \{T_i, VC_{ij}, VD_{ij}, C_i\}$ 。其中, T_i 表示子任务的时间约束,应用需要在此时间内交付给用户,



否则将产生违约; VC_{ij} 表示子任务的代码大小,单位为MIS(百万指令); VD_{ij} 表示子任务的数据大小,单位为MB; C_i 表示租用公有云的代价,用户可以在SLA中约束预算。

本文使用WFG^[7]来描述子任务间的关系,子任务之间的关系可以看作一个DAG,其中顶点表示子任务,子任务间的优先关系用有向线段表示。WFG使用4种状态跟踪调度的情况,分别为阻塞、准备、执行、完成。

为了简化子任务工作量的计算,将子任务分为同样的大小,如果出现子任务过大不能分割的情况,可以将子任务理解为串行的几个子任务。

定义3 定义 $PrR_k=\{Pr\mu_k, L_k, DS_k\}$ 为编号为 k 的私有资源插槽。其中, $Pr\mu_k$ 表示虚拟资源插槽的计算能力,其单位为MIPS(百万指令/s); $L_k=\{V_{ij}\}$ 表示子任务 V_{ij} 分配到私有插槽 k 中的信息副本,分配的过程是自动而随机的,仅表示该资源插槽 k 可以查询到 V_{ij} 的信息; DS_k 表示私有插槽 k 的磁盘速度,其单位为MB/s,用来指示数据读写的速度,在调度数据分配的时候不可用。

在本系统模型中,规定一个资源插槽在同一时刻只能运行一个子任务,插槽是虚拟机组成的最小单位。因此,实体机形成的虚拟机包含的插槽数目由实体机的CPU核心数来规定。如实体机的CPU是4核的,则在该实体机上生成的每一个虚拟机尽量包含4个插槽。如果一台机器生成的资源插槽超过了CPU的核心数目,该资源插槽的效率将迅速减弱。

定义4 定义 $PuR_q=\{Pu\mu_q, cC_q, cD_q, cti_q, cto_q, NB\}$ 为公有云提供的编号为 q 的资源插槽。其中, $Pu\mu_q$ 表示计算能力,单位为MIPS; cC_q 表示计算价格,单位是元/MIS; cD_q 表示存储价格,单位是元/MB; cti_q 表示数据复制到插槽的价格,单位是元/MB; cto_q 表示数据从插槽复制出去的价格,单位是元/MB; NB 表示网络带宽,其影响需要调度的子任务数据在插槽间的传输时间,其单位为MB/s。

从定义4可以看出,公有云用计算能力报价的形式给出了对计算单元的SLA约束。租用的公共资源可以定义为具有不同定价策略而形成的额外费用。实际上由于私有云的私有属性,可以认为私有云的操作和维护成本一直固定。则混合云中的成本变化主要在租用公有云资源时产生。对于不同的公有云,虽然其SLA模板或者定价策略不同,但服务商不会隐藏其报价,可以较容易地构建成本函数。

定义3和定义4反映了私有云或公有云的不同计算

能力,其取决于不同的CPU速度和内存容量。

定义5 定义 $WR_i=\sum_{j=1}^N \frac{VC_{ij}}{T_i}$ 为资源需求权重,表示任务 i 在资源槽中的分布率。其中, N 为任务 i 的子任务数量。

调度器可以根据先算出的资源权重大小决定每个任务需要分配多少资源。

通过以上定义与描述,在IaaS层进行的混合云调度可以描述为规划问题:在现有私有云资源和外部云资源使用价格可知的情况下,如何分配需要调度的子任务才可以使得私有云上的资源得到最大化利用,并且使用外部资源的代价最小。问题的目标是在满足任务SLA的前提下,最大化任务完成的效用。

2.3 基于实时性和价格的调度

一般的SLA都会就执行时间(execution time)、可靠性(reliability)、可用性(availability)、声誉(reputation)、价格(price)等做出规定。在本文描述的混合云环境中,用户应用是向私有云提交。虽然用户也需要与该私有云进行SLA协商并签订协议,但服务选择的过程已经完成,用户最终关心的是支付的价格和服务的满意程度。而且由于该私有云在子任务路径上已经占有了关键节点——用户接口,因此可以假设该私有云在盈利的情况下接受用户请求,或者用户不得不选择该私有云完成应用。

在混合云环境下,普通用户在选择某一应用时其实已经选择了服务的关键节点,对可靠性、可用性、声誉等都做了经验判断并选择接受,而对于价格,可能选择包月等套餐形式支付,执行时间也仅以用户体验为主。而在私有云通过自身调度就能完成应用的情况下,可靠性、可用性、声誉等都是有所保证的,代价是私有云支付的固定成本,从投资角度看不需要考虑,用户仅需要考察是否能保证应用的实时性。当私有云需要调度外部的公有云共同完成应用时,需要选择支付成本最少的公有云进行调度。因此,混合云环境下的SLA约束主要体现在应用对私有云的实时性要求和私有云调度公有云资源的代价问题。

混合云调度问题简化为一个背包问题,该问题已被证明是一个NP完全问题。以下根据定义1~定义4列出调度的模型为:

$$\begin{aligned} & \min \sum_{k=1}^m \sum_{j=1}^n CostF_{ij}[k] + EEt[k, ij] + Est[k] \\ \text{s.t. } & \sum_{k=1}^m \sum_{j=1}^n \frac{VC_{ij}}{Pr\mu_k} \leq T_i \ \& \ \sum_{k=1}^m \sum_{j=1}^n (cC_q \times VC_{ij} + cD_q \times VD_{ij}) \leq C_i \quad (1) \end{aligned}$$

其中, $\sum_{k=1}^m \sum_{j=1}^n \frac{VC_{ij}}{Pr\mu_k} \leq T_i$ 为任务 i 在私有云中的时间约束, $\sum_{k=1}^m \sum_{j=1}^n (cC_q \times VC_{ij} + cD_q \times VD_{ij}) \leq C_i$ 为任务 i 在公有云的预算控制约束, $Est[k] = \frac{VC_{ij[k]remain}}{Pr\mu_k}$ 表示预计的完成时间, 即还有多少时间私有云资源插槽 k 可以调度下一个子任务; $EEt[k, ij] = \frac{VC_{ij}}{Pr\mu_k} + Dtt$ 表示预计执行时间, 包含代码的执行时间和数据在网络和磁盘上的传输时间; $Dtt = \frac{VD_{ij}}{NB} + \left(\frac{VD_{ij}}{DS_k} \right) \times 2$ 表示数据传输时间, $\frac{VD_{ij}}{NB}$ 表示数据在网络上的传输时间, $\left(\frac{VD_{ij}}{DS_k} \right) \times 2$ 表示数据在磁盘上的读写时间; $CostF = VC_{ij} \times cC_k + VD_{ij} \times cD_k + VD_{ij} \times \left(\frac{1}{cti_k} + \frac{1}{cto_q} \right)$ 为调用外部公有云的成本。

以上的规划问题可以看作一个 0-1 背包问题, 可以使用启发式贪婪算法来求得一个近似的最优解。

2.4 调度步骤

以上问题中, 私有云和公有云的调度有不同的目标。对私有云而言, 如何最大化利用现有资源是最值得关心的问题; 而公有云的问题是如何最小化租用的成本。

当新的任务到达调度器时, 将所有未完成的任务放置到调度池中等待调度, 而执行完的任务需要移出调度池, 具体调度步骤如下。

(1) 基于权重的子任务分配

当新的任务到达调度器时, 使用 Facebook 的公平算法将调度请求分配到资源插槽, 其方法是将剩余的资源平均分配给同时到达的资源请求。但这种方法可能会导致需要资源多的请求不能获得更多资源, 因此需根据资源需求权重, 计算每一个任务需要分配多少资源到插槽上。分配步骤假设任务全部分布在私有资源插槽, 计算将会产生多少子任务副本数量及分布情况, 最后形成的分布记为集合 $NR_{(k)}$ 。

(2) 计算不同插槽的估计完成时间

对每一个插槽 k 计算 $EEt[k, ij] + Est[k]$ 。如果得到的估计完成时间已经高于 SLA 给出的时间约束 D_i , 则该插槽将不运行子任务 V_{ij} 。所有满足子任务 V_{ij} 时间约束的插槽形成一个候选集 U , 这个候选集成为以后确定是否需要调度公有云资源的依据。如果候选集的插槽数目少于调度池中需要调度的子任务数量, 为满足 SLA 中的时间约束, 需

要调度公有云的资源。

(3) 用动态规划方法进行候选插槽调度

在候选集确定的情况下, 根据最小执行时间的原则将子任务分配给资源插槽。受到分配的数据位置、网络带宽、插槽计算能力的影响, 即使是相同的子任务和插槽, 子任务分配位置的不同也会造成执行时间的不同。如子任务 V_{ij} 开始时分配到插槽 k 上, 但插槽 q 计算出的估计完成时间比插槽 k 短, 这时将子任务转移到插槽 q 上运算会带来传输时延。由于规划的目的是使用最小的执行时间完成子任务, 因此在需要进行子任务迁移时需要计算传输时延。具体算法如下。

算法 1 最小完成时间约束的私有插槽选择

输入: 分配 V_{ij} 的插槽分布 $NR_{(k)}$

输出: 满足最小完成时间的 V_{ij} 与插槽对应关系 Z

```
for each task  $V_{ij}$ 
  for each  $k \in NR(k)$ 
    { 计算  $Est[k] \leftarrow VC_{ij[k]remain}/Pr\mu_k$ ;
      if ( $V_{ij} \in L_k$ )  $Dtt \leftarrow 0$ ;
      else  $Dtt \leftarrow VD_{ij}/NB + 2 \cdot VD_{ij}/DS_k$ ;
      计算  $EEt[k, ij] = VC_{ij}/Pr\mu_k + Dtt$ ;
      if ( $Est[k] + EEt[k, ij] \leq D_i$ )  $U \leftarrow NR(k)$ ; }
  for each  $k \in U$ 
    {  $NR_{(k)} \leftarrow V_{ij}$ ;
       $Est[k] \leftarrow Est[k] - EEt[k, ij]$ ;
      Matrix  $Z[N \cdot M] \leftarrow Min\{Est[k]\}$ ;
```

(4) 在插槽的等待队列中选择可以转移的运算符任务

私有资源插槽中被选择但还没执行的子任务也可以在公有云中执行。每一个任务有其不同的特点, 有的任务需要处理大量数据, 但执行程序很短, 有的任务执行程序很长, 但只要处理很少的数据。当新的任务需要调度而私有云资源紧张时, 对步骤(3)已安排的子任务进行重新检查, 也许可以找出既满足 SLA 时间约束, 又有较少支出到公有云代价的子任务。判断该队列中的子任务是否能够迁移到公有云的方法是计算队列中每一个子任务从私有云到公有云的数据传输时间, 看其是否满足 SLA 的时间约束。具体算法如下。

算法 2 等待子任务队列的重选择算法

输入: 等待子任务队列 Q , 公有云插槽集合 PR_q

输出: 可以转移到公有云执行的子任务集合 P

```
for each task  $V_{ij} \in Q$  {
  for each  $q \in PR_q$ 
```



```

{ 计算  $Dtt \leftarrow VD_j / NB + 2 \cdot VD_j / DS_k$ ;
  计算  $EEt[k, ij] = VC_j / Pr\mu_k + Dtt$ ;
  if ( $EEt[q, ij] \leq T_i$ )  $P \leftarrow V_j$ ;
}

```

(5) 按最小代价的原则选择公有云插槽

该步骤的算法类似于步骤(3),最小化时间换成最小化代价作为动态规划的目标。首先用完成时间约束查找可选的公有云插槽集合 PR_q ,然后针对该集合找出最小代价组合。具体算法如下。

算法3 最小代价的公有插槽选择

输入:公有云执行的子任务集合 P ,可选公有云插槽集合 PR_q

输出:满足最小代价的 V_j 与插槽对应关系 R

```

for each task  $V_j \in P$  {
  for each  $q \in PR_q$  {
     $PR_q \leftarrow V_j$ ;
    Matrix  $R[N \cdot M] \leftarrow \text{Min}\{CostF\}$ ;
  }
}

```

3 实验结果与对比分析

为了验证 ASS 算法的有效性,本文在 CloudSim 下设计实验。使用的硬件环境为 Intel 酷睿 2 Quad 3.2 GB CPU,二级缓存 6 MB×2,4 GB 内存。软件环境为

Windows7,使用 Eclipse 4.2 下配置的 ClousSim3.0.1 完成实验。为了验证算法的有效性,将 ASS 算法调度与 FIFO、公平调度算法进行比较。

首先在 CloudSim 环境下建立数据中心代表私有云,该数据中心设置为 1 000 MIPS 的计算能力,5 GB 内存,1 TB 的存储空间和 10 Gbit/s 的带宽。以下实验中,数据中心分布到 100 个插槽(VM)中,每个插槽有 5 GB 的存储,512 MB 内存和 1 Gbit/s 的带宽以及运算能力为 10~50 MIPS 的 1 个 CPU 核心。同时设置用户提交的子任务有 200~400 MB 的文件、20~40 MB 的输出速度以及 400~1 000 MIS 的计算量。

3.1 时延评价

本实验对使用 ASS、FIFO 调度和公平调度的执行时间(指子任务从开始执行到执行完成的时间)、等待时间(指子任务从到达队列等待到任务开始执行的时间)、完成时间(指子任务从请求直到最后完成的时间)进行评价。设置需要调度的子任务分别为 50 个、100 个、200 个和 500 个,并不考虑公有云资源的参与情况。仿真结果如图2所示。

从图2中可以看出,申请子任务很多的情况下,ASS 可以比 FIFO 和公平调度的等待时间、执行时间和完成时间都更少。实际上,在计算资源需求和数据量相同的情况下,分配更多的计算资源能表现出更短的计算时间,而 ASS 方法使用估计最小完成时间的方法来分配更强大的资源给予任务,因此执行时间较短。而对多个子任务请求的情况,

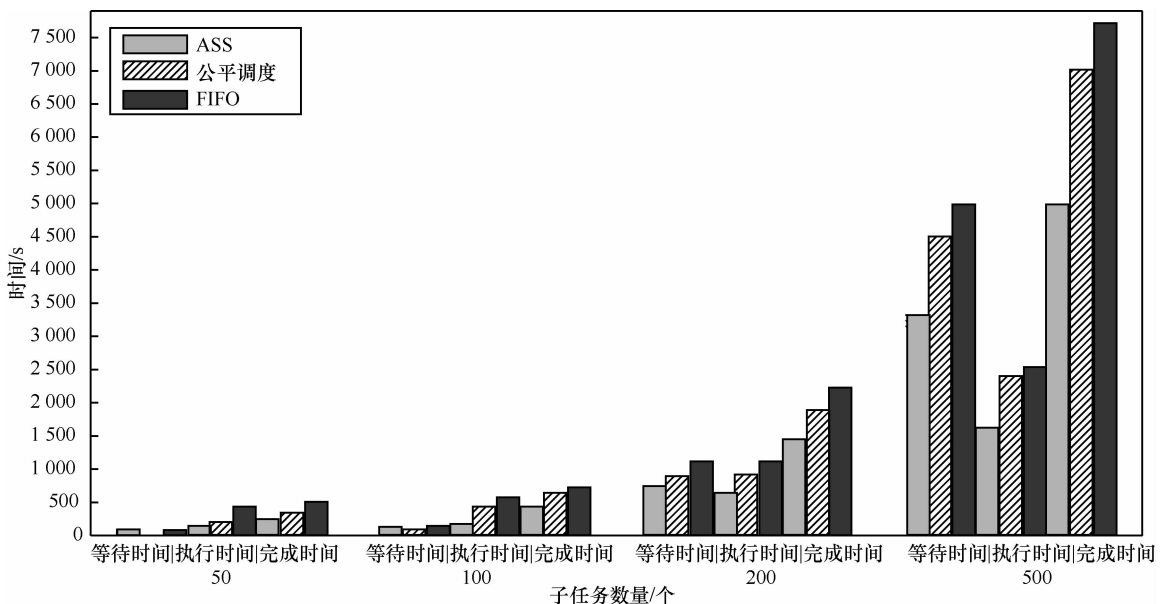


图2 子任务等待时间、运行时间和完成时间

ASS算法使用资源需求概率来克服等待时间的浪费,相对来说,资源插槽和子任务相同的情况下,执行时间更短。ASS算法的完成时间更短,云资源供应商可以具备更多的盈利能力,对用户来说,能得到更实时的反馈和响应。

3.2 SLA 满意度测量

定义SLA满意度为 $SR=k/n$,其中 SR 为调度的满意度, k 为在约定时间内完成的子任务的数量, n 为提交到资源的子任务数量。对几个算法的调度效率和可靠性进行比较,假设在未完成全部子任务的情况下,仍供应资源直到所有子任务结束。图3显示了请求200个子任务,并人为设置30s为调度的SLA约定时间,并在资源相同的情况下3种算法的比较情况。结果显示,ASS的算法具有更好的调度满意度。图4设置500个子任务,SLA约定时间为170s。图5设置700个子任务,同时SLA约定的时间为250s。可以看出,ASS算法的满意度明显较其他算法更高。

实际上,由于ASS算法设计的目标是最大化利用私有云资源,具有最小任务执行时间的子任务将被优先选择执行,这里试图最小化任务等待时间、执行时间和数据传输间隔等不同类型的任务,以降低总的任务执行时间,以达到更好的SLA满意度。

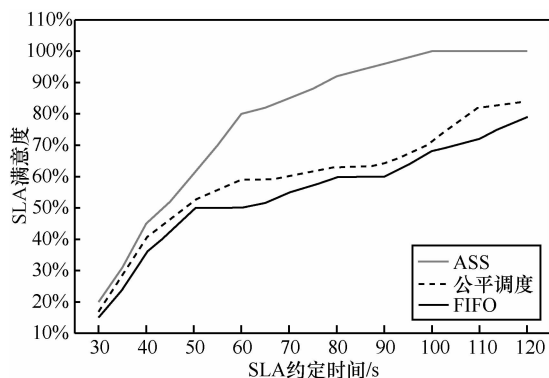


图3 200个子任务下的SLA满意度曲线

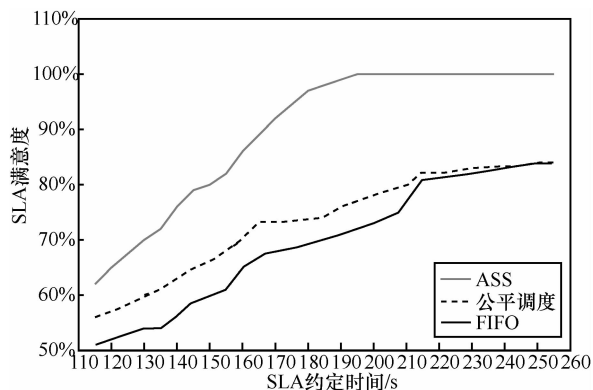


图4 500个子任务下的SLA满意度曲线

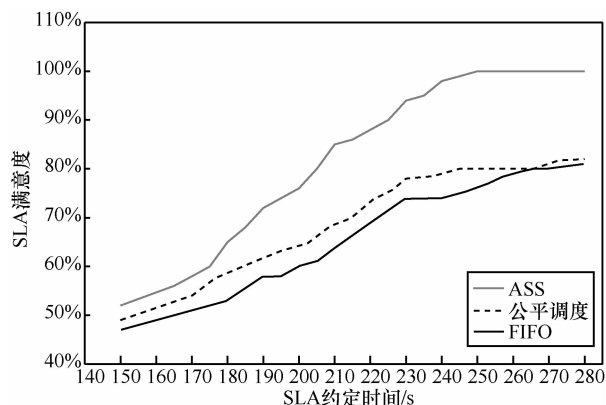


图5 700个子任务下的SLA满意度曲线

3.3 成本分析

此实验加入公有资源插槽的调度,以保证任务能在SLA约束下执行。对公有资源插槽的代价是此实验分析的目标。实验设计了200个、500个、700个子任务,并通过改变SLA限制的方式进行测试。同样将SLA限制表现为任务完成时间,并测试从宽松到严格的变化数据。实验结果如图6~图8所示。

图6、图7和图8反映出在严格限制SLA的情况下,3种算法的成本基本相同,这是由于在严格限制的情况下,私有云能完成的子任务数量相对有限,只能大量使用公有云

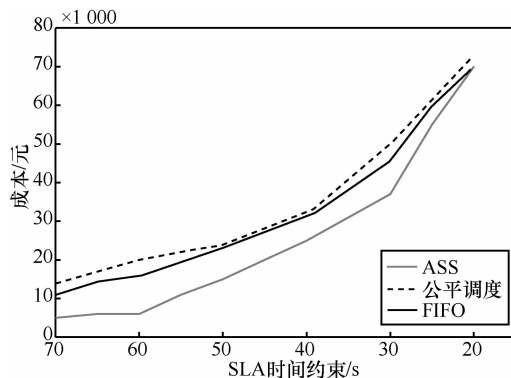


图6 200个子任务下的成本比较

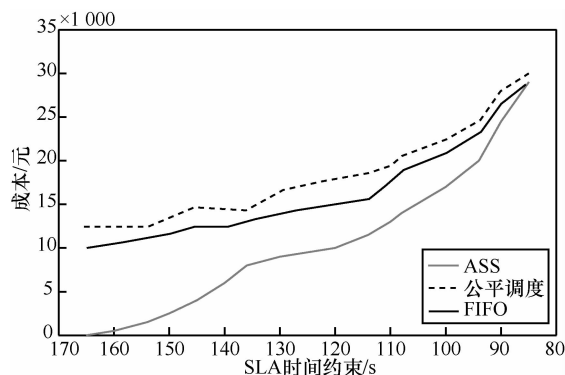


图7 500个子任务下的成本比较

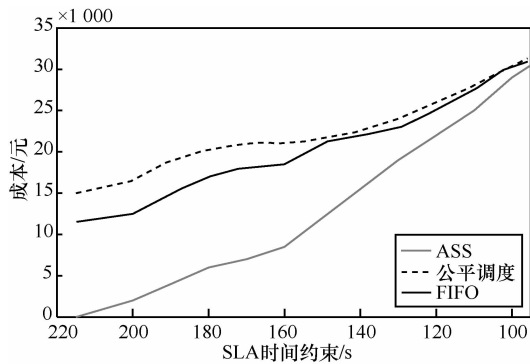


图 8 700 个子任务下的成本比较

资源进行计算,以便完成所有的计算需求,3 种算法最终趋向于在没有私有云资源的情况下,应用完全由公有资源完成。实际上 ASS 算法不仅会在资源过载的情况下选择公有资源,在子任务被分配的情况下也会主动检查是否可以调度一些低成本的任务到公有资源插槽上运行,以最大化利用私有云资源。

3.4 与其他调度方法的比较验证

将 ASS 算法与其他调度方法的功能和特点进行比较,比较结果见表 1。

选择参考文献[11]中的算法对 3 个实验再进行对比, COS 方法是使用二进制整数规划的方式进行外部云资源的调度。实验设置与以上 3 个实验相同。实验结果如图 9~图 11 所示。

从图 9 可看出,ASS 算法较 COS 方法需要更多的代价来完成应用,这是因为 ASS 算法更侧重响应时间的优先,而 COS 方法考虑了整个 SLA 的其他因素。从图 10 可以看出,ASS 算法在运算和完成时间上较 COS 方法更优。

实际上,图 9 的比较本身隐含了 ASS 算法向外部资源请求的代价,由于 ASS 算法尽量保证内部资源的分配优先,在代价方面要比 COS 方法更多。从图 11 可以看出,要达到同等满意度, COS 方法需要的执行时间比 ASS 算法更长,而 SLA 满意度是指任务在内部资源被调度的比例,假设内部资源和云代理处于同一私有云,而内部资源的成本

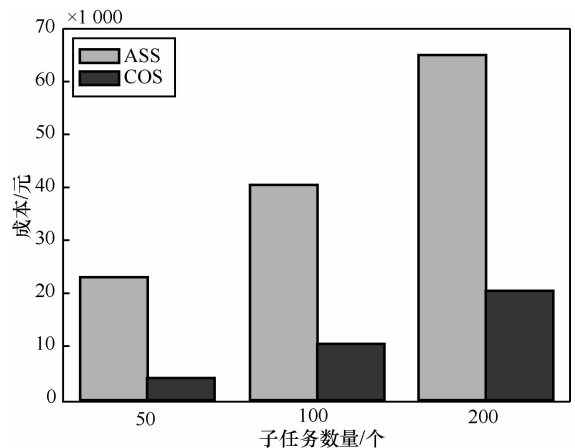


图 9 ASS 与 COS 执行代价的比较

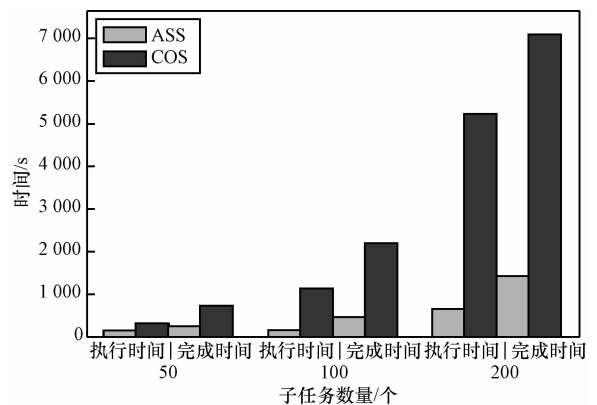


图 10 不同子任务下执行时间和完成时间的比较

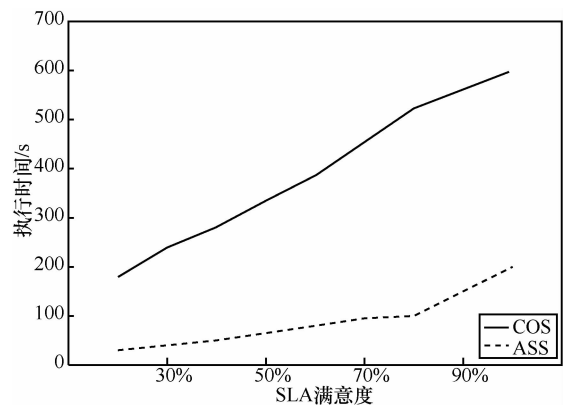


图 11 ASS 与 COS 的 SLA 满意度比较

表 1 ASS 与其他算法的对照比较

类别	ASS	FIFO	公平调度	COS ^[10]	IWF ^[4]	GAB ^[6]	CMS4M ^[7]
SLA	✓	×	×	△	×	×	×
代价	✓	×	×	✓	×	×	△
混合云	✓	×	×	✓	✓	×	×
任务估计	✓	×	×	×	✓	×	×
多租户	✓	×	✓	✓	△	×	△
算法	动态规划	FIFO	FIFO	整数规划	频繁项集	遗传算法	CSMA

为固定成本的情况下,将内部资源的代价相对于外部资源的代价取极小值,应该可以得到 ASS 算法的代价比 COS 方法代价更小的结论。

以上实验反映了 ASS 算法能够以极小的外部资源代价更好地保障用户应用的 SLA 约束,以保证应用完成的可靠性。

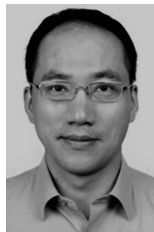
4 结束语

混合云结合了私有云安全、稳定以及公有云灵活、安全、经济的优点,是未来云计算的研究热点。根据混合云的特殊性,使用 SLA 约束的自适应调度,实现对成本和响应时间的最大满意度。算法使用估计执行时间和动态规划方法来实现私有云资源的最优化分配,以最大化利用私有云资源并满足 SLA 约束。实验表明,相较于其他调度算法,该算法能够较好地降低资源端的子任务等待、执行和完成时间,最大程度满足用户的 SLA 约束,且降低应用完成的成本支出,可以有效保障应用的可用性和可靠性。

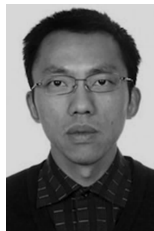
参考文献:

- [1] PRAHALAD H A, TALUKDER A, PARDESHI S, et al. Phoenix: system for implementing private and hybrid cloud for OMIC sciences applications[C]//The Conference on 2010 Seventh International of Wireless And Optical Communications Networks (WOCN), September 6-8, 2010, Colombo, Sri Lanka. Colombo: IEEE Press, 2010: 1-5.
- [2] GOIRI I, GUITART J, TORRES J. Characterizing cloud federation for enhancing providers' profit [C] //The IEEE 3rd International Conference on Cloud Computing (CLOUD), May 4-7, 2010, Miami, FL, USA. New Jersey: IEEE Press, 2010: 123-130.
- [3] SOTOMAYOR B, MONTERO R S, LORENTE I M, et al. Virtual infrastructure management in private and hybrid clouds[J]. IEEE Internet Computing, 2009, 13(5): 14-22.
- [4] ZHANG H, JIANG G, YOSHIHARA K, et al. Intelligent workload factoring for a hybrid cloud computing model[C] //The 2009 World Conference on Services-I, July 6-10, 2009, Los Angeles, CA, USA. New Jersey: IEEE Press, 2009: 701-708.
- [5] KARYPIS G, KUMAR V. Multilevel k-way hypergraph partitioning[J]. VLSI Design, 2000, 11(3): 285-300.
- [6] GE Y, WEI G. Ga-based task scheduler for the cloud computing systems [C]//The 2010 International Conference on Web Information Systems and Mining (WISM), October 23-24, 2010, Sanya, China. New Jersey: IEEE Press, 2010: 181-186.
- [7] GROUNDS N G, ANTONIO J K, MUEHRING J. Cost-minimizing scheduling of workflows on a cloud of memory managed multicore machines [C]//The 1st International Conference on Cloud Computing, December 1-4, 2009, Beijing, China. New York: Springer-Verlag, 2009: 435-450.
- [8] WU Z, LIU X, NI Z, et al. A market-oriented hierarchical scheduling strategy in cloud workflow systems[J]. The Journal of Supercomputing, 2013, 63(1): 256-293.
- [9] WU Z, LUO J. The measurement model of grid QoS [C]//The 10th International Conference on Computer Supported Cooperative Work in Design, 2006, Nanjing, China. New Jersey: IEEE Press, 2006: 1-6.
- [10] BOSSCHE V, VANMECHELEN K, BROECHHOVE J. Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds [J]. Future Generation Computer Systems, 2013, 29(4): 973-985.
- [11] FOX A, GRIFFITH R, JOSEPH A, et al. Above the clouds: A Berkeley view of cloud computing [J]. Dept Electrical Eng and Comput Sciences, University of California, Berkeley, Rep. UCB/EECS, 2009(1): 28.

【作者简介】



徐晓峰(1970-),男,中国移动通信集团江苏有限公司工程师、项目主管,主要研究方向为云计算应用系统、IDC 规划建设、CDN 及移动互联网应用。



徐劲松(1975-),男,博士,南京邮电大学通达学院副教授、教研室主任,主要研究方向为信息安全、云计算及大数据应用。