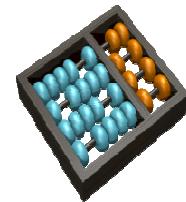




Escalonamento em Nuvens Híbridas

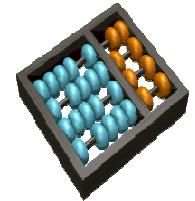


Programa de Pós-Graduação em Ciência da Computação
Universidade Estadual de Campinas - UNICAMP
Instituto de Computação – IC

Erick Aguiar Donato

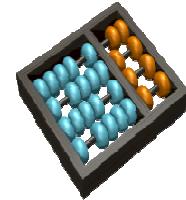
30 de Outubro de 2013

Tópicos



- Introdução
- Exemplo de Escalonador
 - Condor
- Escalonamento em Nuvens Híbridas
 - Artigos
- Conclusão

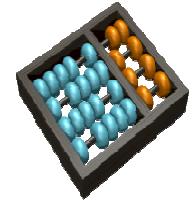
Computação nas Nuvens



O que é Computação nas Nuvens?



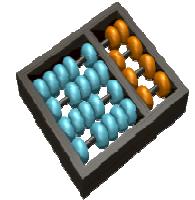
Definição



Definição (NIST)

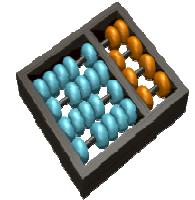
- *National Institute of Standards and Technology*
- *Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*

Características



- Provisionamento dinâmico de recursos
- Geo-distribuição e ubiquidade de acesso a rede
- Orientado ao serviço
- Preço baseado no uso
- (Boutaba, 2010)

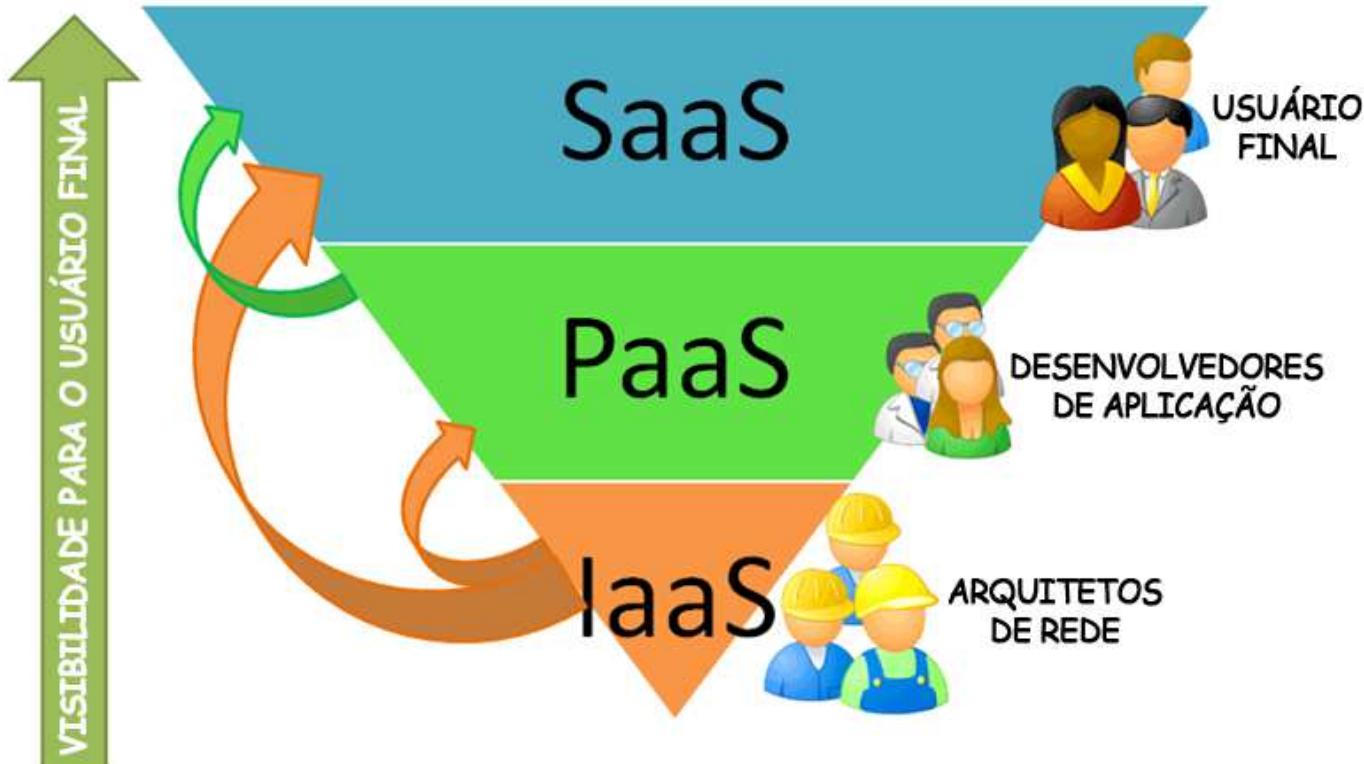
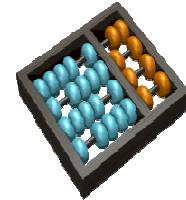
Benefícios para empresas



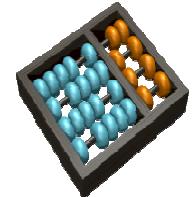
- Baixo investimento inicial
- Baixo custo operacional
- Altamente escalável
- Fácil acesso
- Reduz risco com tecnologia
- Mantém o foco no negócio



Tipos de Serviços

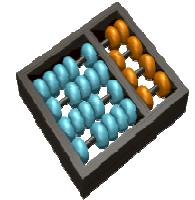


Tipos de Serviços



| Exemplos de modelos de computação nas nuvens | |
|--|--|
| Modelo | Exemplos |
| SAAS | CRM(Customer Relationship Management) da Sales Force |
| | Google Docs |
| | Mail Live Office |
| PAAS | Windows Azure |
| | Google App Engine |
| | Amazon AWS EC2 |
| | Aneka |
| IAAS | Eucalyptus |
| | Amazon AWS EC2 |
| | Windows Azure |

Tipos de Nuvens

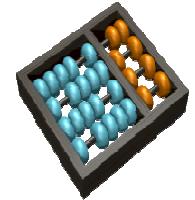


- Públicas
 - Amazon EC2
 - Microsoft Azure

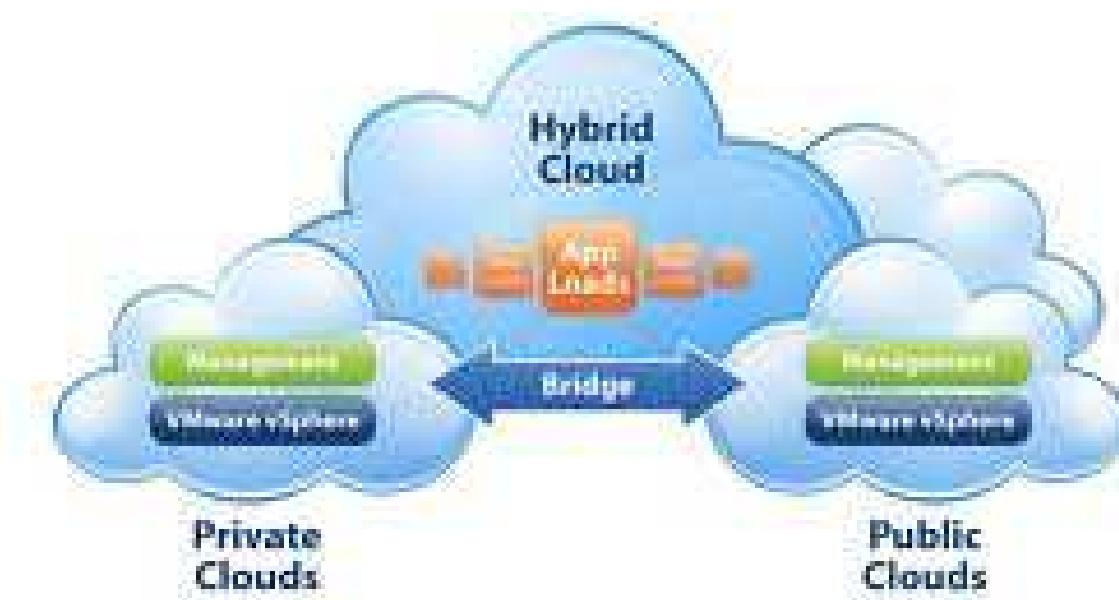
- Privadas
 - Eucalyptus
 - OpenNebula



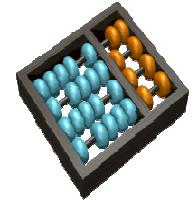
Tipos de Nuvens



- Híbridas
 - Exemplo: Eucalyptus + Amazon EC2



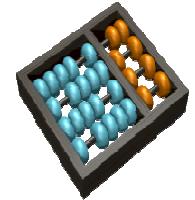
Escalonamento



. Definição geral

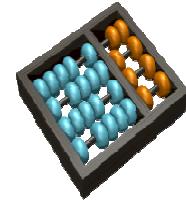
- Seja $A = \{a_1, a_2, \dots, a_n\}$ o conjunto de tarefas submetidas pelos usuários e que serão executadas no sistema distribuído heterogêneo R . O escalonador é responsável pela construção de um seqüenciamento de tarefas A nos recursos R seguindo um ou mais objetivos a serem otimizados
- (Bittencourt e Madeira, 2012)

Escalonamento



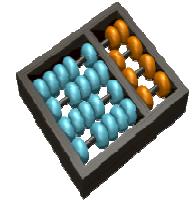
- . Escalonamento é essencialmente um processo de tomada de decisão
- . Permite a utilização/compartilhamento de recursos entre processos a serem executados
- . O escalonador é responsável por decidir qual tarefa computacional deve executar em qual recurso
- . Para isso pode levar em consideração
 - . Tempo e o custo da computação
 - . Tempo e o custo da transmissão dos dados
 - . Localização das fontes de dados

Escalonamento



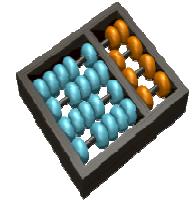
- Objetivo clássico (Exemplo)
 - Minimizar o tempo de execução (makespan)
- Tarefas submetidas pelos usuários podem ser divididas em duas formas
 - Tarefas independentes
 - Tarefas dependentes
 - Workflows
 - Normalmente representadas por um DAG (Directed Acyclic Graph)

Escalonamento

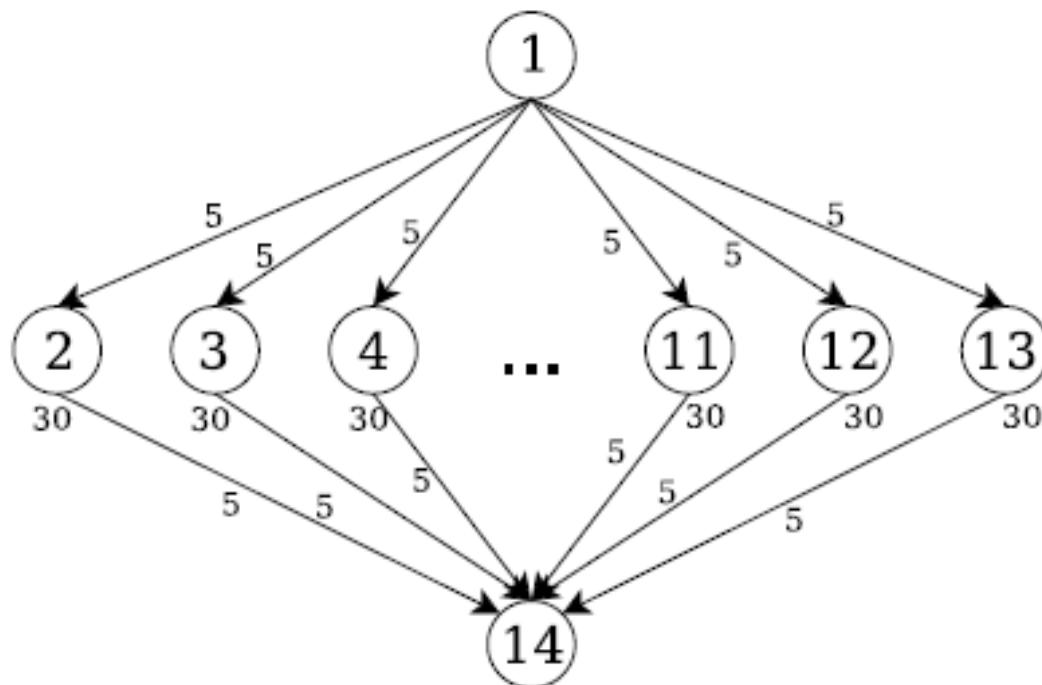


- Workflows
 - Conjunto de tarefas dependentes
- Podem ser representadas da seguinte forma
 - Seja o DAG $G = (V, E)$, com N nós (tarefas)
 - V é o conjunto de N nós, onde $N = |V|$ e t , pertencente a V , representa uma tarefa indivisível que deve ser executada em algum recurso
 - E é o conjunto de arestas direcionadas, onde cada aresta e , pertencente a E , representa a dependência de dados entre dois nós

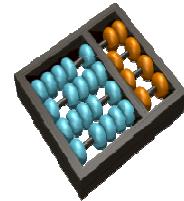
Escalonamento



- Workflows
 - Exemplo

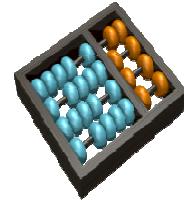


Escalonamento em Nuvens Híbridas



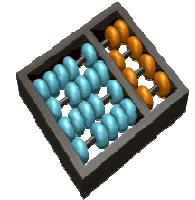
- . Quando a infra-estrutura da nuvem privada for insuficiente para suprir as requisições dos usuários, recursos de uma nuvem pública devem ser disponibilizados
- . Isso envolve duas questões fundamentais
 - . Quais recursos devem ser pedidos?
 - . Quais tarefas devem ser executadas nos recursos emprestados?

Escalonamento em Nuvens Híbridas

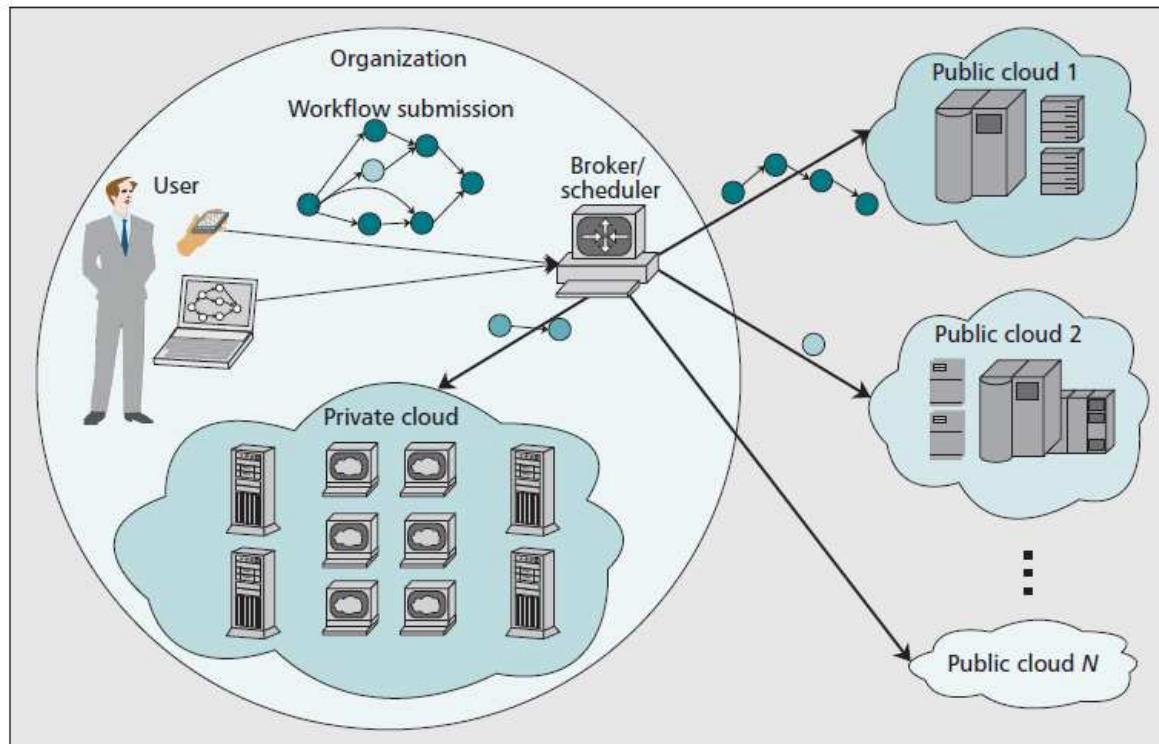


- . O escalonamento pode fornecer ao usuário um grande aproveitamento dos recursos da nuvem privada e a utilização dos recursos da nuvem pública.
- . Proporciona **elasticidade** quando a demanda exceder a capacidade dos recursos privados
- . No entanto, o orçamento para tais operações deve ser levado em consideração para não extrapolar os limites definidos

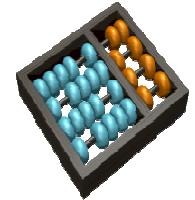
Escalonamento em Nuvens Híbridas



- Infra-estrutura de uma nuvem híbrida

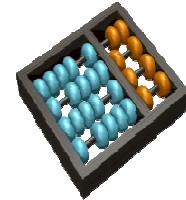


Escalonamento em Nuvens Híbridas

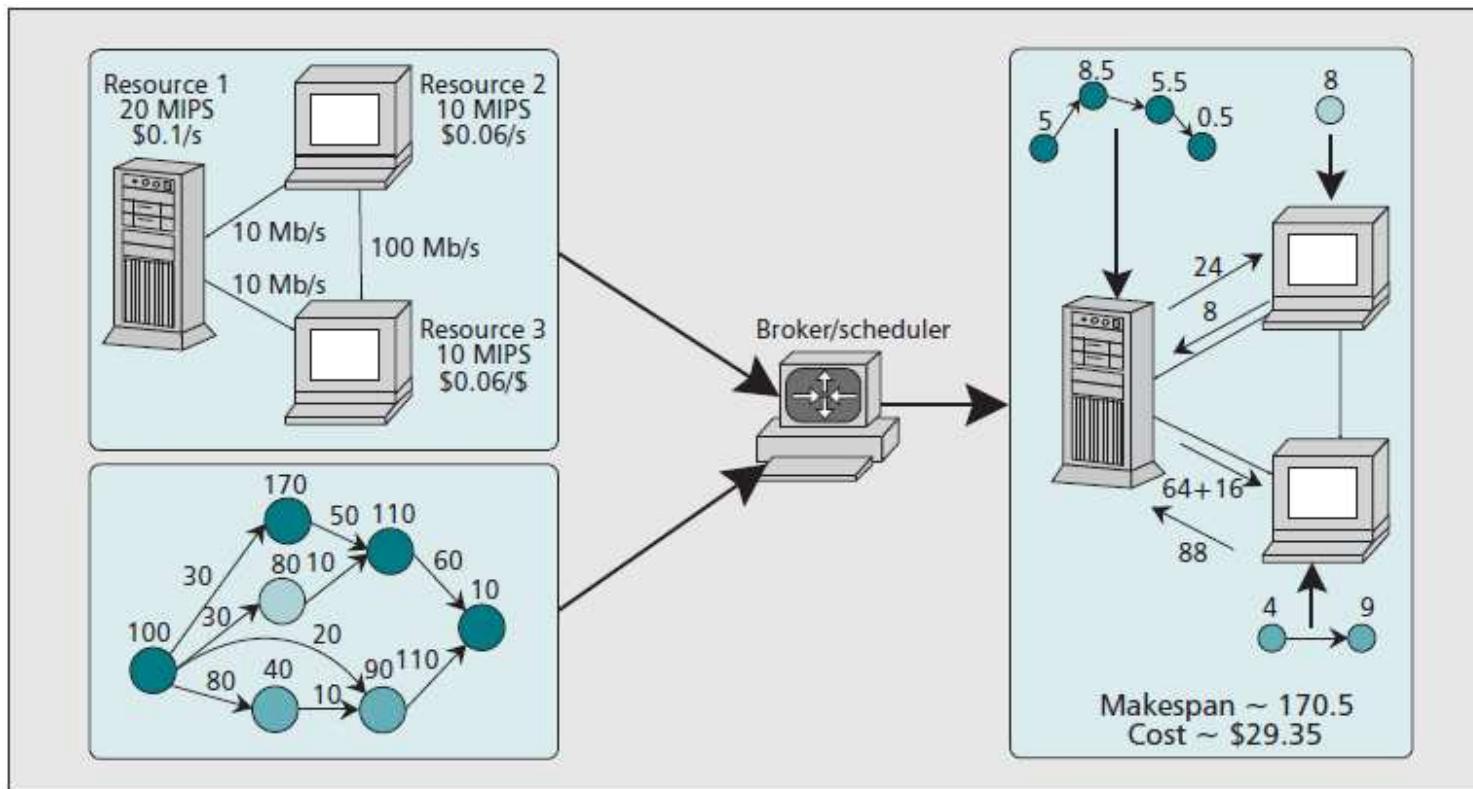


- . Desafios
 - . Como interagir com diferentes nuvens públicas
 - . Segurança
- . O problema de escalonamento é um problema **NP-completo**

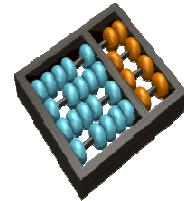
Escalonamento em Nuvens Híbridas



- Exemplo de escalonamento

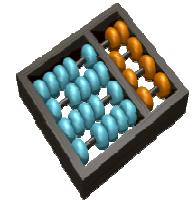


Escalonamento em Nuvens Híbridas



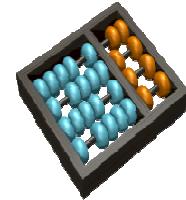
- . O escalonamento é feito quando os dados são enviados pelo usuário
- . O escalonador utiliza as informações disponíveis para computar o tempo de execução das tarefas nos recursos, o tempo de transmissão dos dados e o custo da execução das tarefas

Tópicos



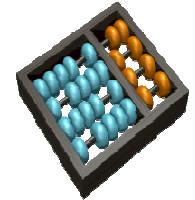
- Introdução
- Exemplo de Escalonador
 - Condor
- Escalonamento em Nuvens Híbridas
 - Artigos
- Conclusão

Condor



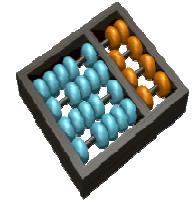
- Condor: A distributed Job Scheduler
 - Todd Tannenbaum, Derek Wright, Karen Miller e Miron Livny
 - Condor Research Project iniciou em 1984
- É um sofisticado escalonador de jobs distribuídos
- University of Wisconsin-Madison
 - Department of Computer Sciences
- Web site: www.cs.wisc.edu/condor
- Trabalha com ***compute-intensive jobs***

Condor



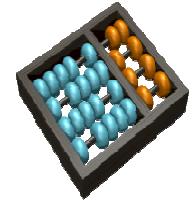
- Condor fornece:
 - Mecanismo de enfileiramento de jobs, políticas de escalonamento, esquemas de prioridade, monitoramento e gerenciamento de recursos
- O usuário submete seus jobs para o Condor que os coloca na fila e escolhe quando, onde executarão baseado em políticas e informações monitoradas

Condor



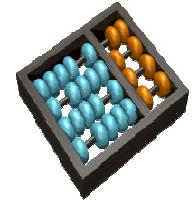
- Características
 - Submissão distribuída
 - Prioridade de Jobs
 - Prioridade de Usuários
 - Jobs dependentes
 - Workflows
 - Suporte a múltiplos modelos de jobs
 - Serial jobs, parallel jobs, etc

Condor



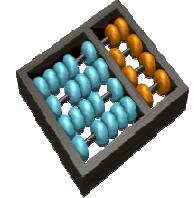
- Características
 - ClassAds
 - Mecanismo flexível para “juntar” as requisições de recursos com as ofertas dos recursos
 - Job checkpoint and migration
 - Checkpoint periódico
 - Job suspend and resume
 - Autenticação e Autorização
 - Plataformas heterogêneas

Condor



- Matching ClassAds
 - ClassAds pode ser satisfeito com outro
 - Job ads quer ser satisfeito com um Machine ads e vice versa
 - Requirements e Rank
 - Exemplo a seguir

Condor



Job ClassAd

```

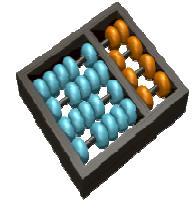
MyType = "Job"
TargetType = "Machine"
Requirements = ((Arch=="INTEL" && Op-
Sys=="LINUX") && Disk > DiskUsage)
Rank = (Memory * 10000) + KFlops
Args = "ini ./les.ini"
ClusterId = 680
Cmd = "/home/tannenba/bin/sim-exe"
Department = "CompSci"
DiskUsage = 465
StdErr = "sim.err"
ExitStatus = 0
FileReadBytes = 0.000000
FileWriteBytes = 0.000000
ImageSize = 465
StdIn = "/dev/null"
Iwd = "/home/tannenba/sim-m/run_55"
JobPrio = 0
JobStartDate = 971403010
JobStatus = 2
StdOut = "sim.out"
Owner = "tannenba"
ProcId = 64
QDate = 971377131
RemoteSysCpu = 0.000000
RemoteUserCpu = 0.000000
RemoteWallClockTime = 2401399.000000
TransferFiles = "NEVER"
WantCheckpoint = FALSE
WantRemoteSyscalls = FALSE
:
:
```

Machine ClassAd

```

MyType = "Machine"
TargetType = "Job"
Requirements = Start
Rank = TARGET.Department==MY.Department
Activity = "Idle"
Arch = "INTEL"
ClockDay = 0
ClockMin = 614
CondorLoadAvg = 0.000000
Cpus = 1
CurrentRank = 0.000000
Department = "CompSci"
Disk = 3076076
EnteredCurrentActivity = 990371564
EnteredCurrentState = 990330615
FileSystemDomain = "cs.wisc.edu"
IsInstructional = FALSE
KeyboardIdle = 15
KFlops = 145811
LoadAvg = 0.220000
Machine = "nostos.cs.wisc.edu"
Memory = 511
Mips = 732
OpSys = "LINUX"
Start = (LoadAvg <= 0.300000) &&
(KeyboardIdle > (15 * 60))
State = "Unclaimed"
Subnet = "128.105.165"
TotalVirtualMemory = 787144
:
:
```

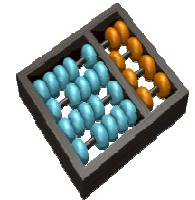
Condor



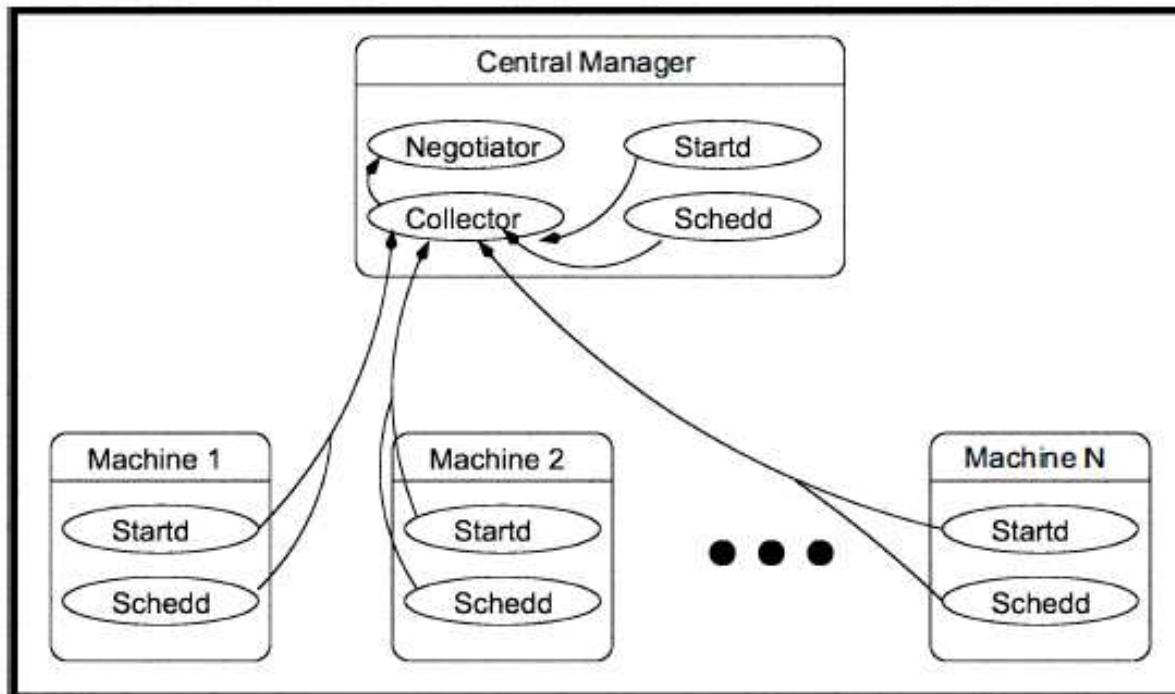
- Arquitetura

- Formada pelo Central manager e outras máquinas que formam o pool
- A regra é satisfazer as requisições que esperam com os recursos disponíveis
- Periodicamente, cada máquina envia o estado do pool para a Central Manager
- Periodicamente, a CM avalia o pool e tenta ligar as requisições com os recursos disponíveis

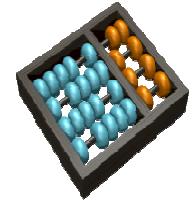
Condor



- Arquitetura



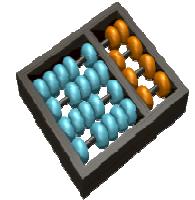
Condor



- Arquitetura

- Condor_startd
 - Representa a máquina para o pool
 - Define políticas e capacidades
- Condor_starter
 - Configura o ambiente de execução e monitora o job em execução
 - Retorna quando o job finaliza
- Condor_schedd
 - Representa o job para o pool
 - Armazena o job do usuário

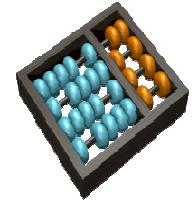
Condor



- Arquitetura

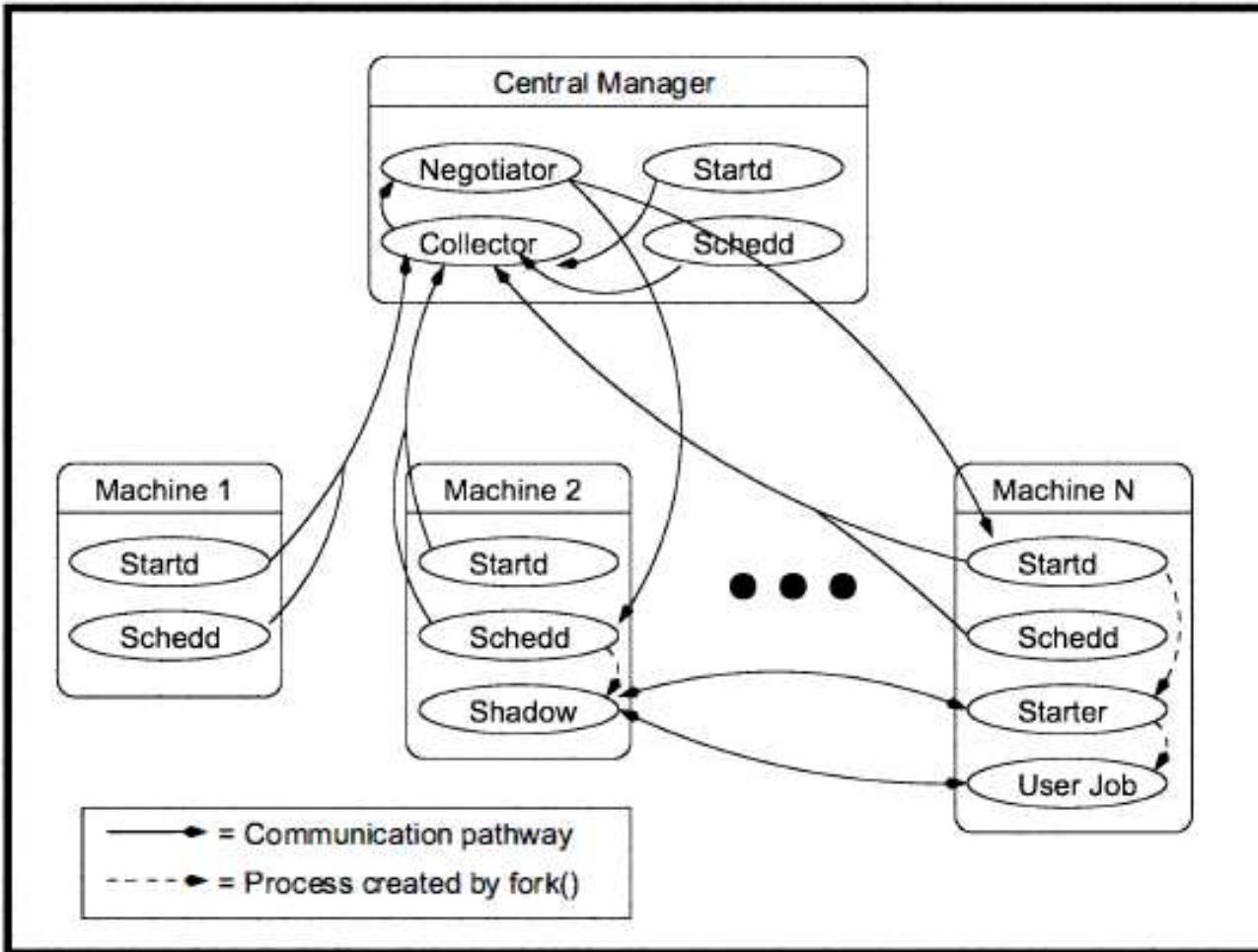
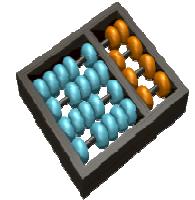
- Condor_shadow
 - Serve as requisições de acesso a arquivos para transferência, logs e reporta estatísticas
- Condor_collector
 - Coleta as informações sobre o estado do pool
 - Todos enviam atualizações para o collector
 - Pode ser um banco de dados dinâmico
- Condor_negotiator
 - Responsável pelo **matching**
 - Garante as prioridades do sistema

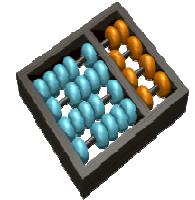
Condor



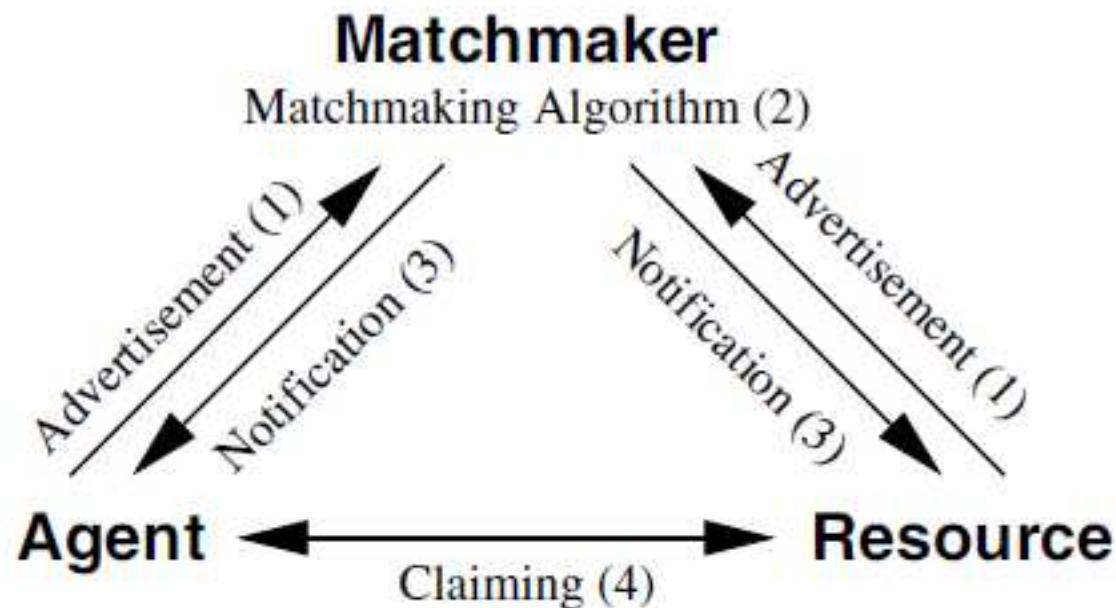
- Arquitetura
 - O exemplo a seguir ilustra quando um job da máquina 2 está sendo executado

Condor

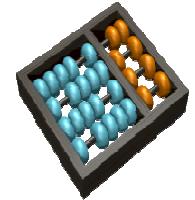




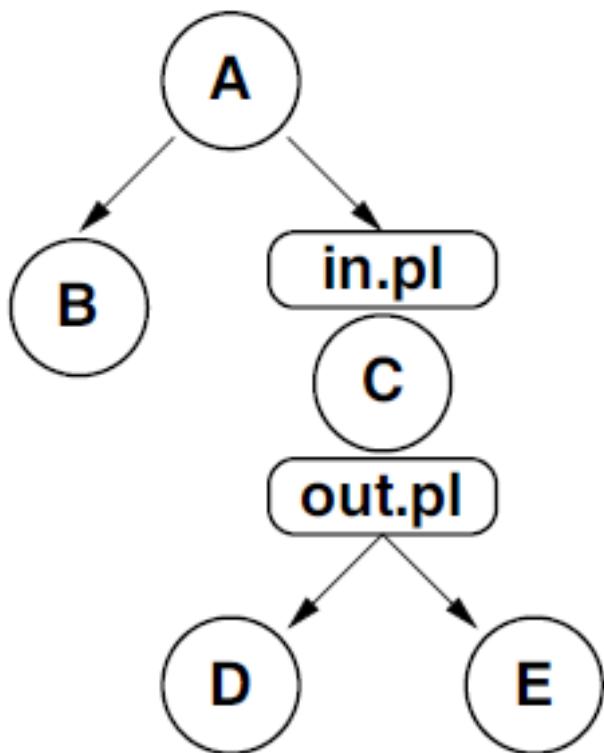
- Resumo da comunicação



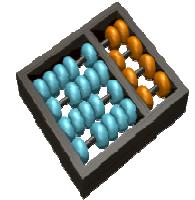
Condor



- Representação de um DAG



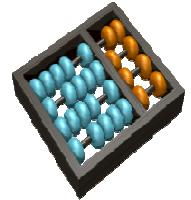
```
JOB A a.condor
JOB B b.condor
JOB C c.condor
JOB D d.condor
JOB E e.condor
PARENT A CHILD B C
PARENT C CHILD D E
SCRIPT PRE C in.pl
SCRIPT POST C out.pl
RETRY C 3
```



- Estudo de Caso

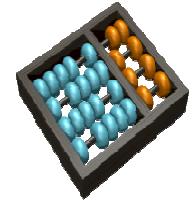
- C.O.R.E. Digital Pictures
 - Computer animation studio
 - Filme de efeitos especiais
 - Compute intensive process
 - Cada frame pode levar até meia hora
 - 1 segundo pode ter 30 frames ou mais
 - Condor foi usado pela C.O.R.E. e outras produções, tais como, X-men, Blade II, Nutty Professor II

Artigo



- . Cloud Scheduler: a resource manager for distribute compute clouds
- . Autores
 - . P. Armstrong et al
- . University of Victoria, Victoria, Canadá
 - . National Research Council Canada; Institute of Particle Physics of Canada
- . 2010

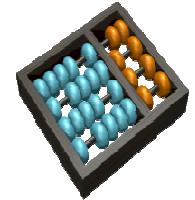
Armstrong et al (2010)



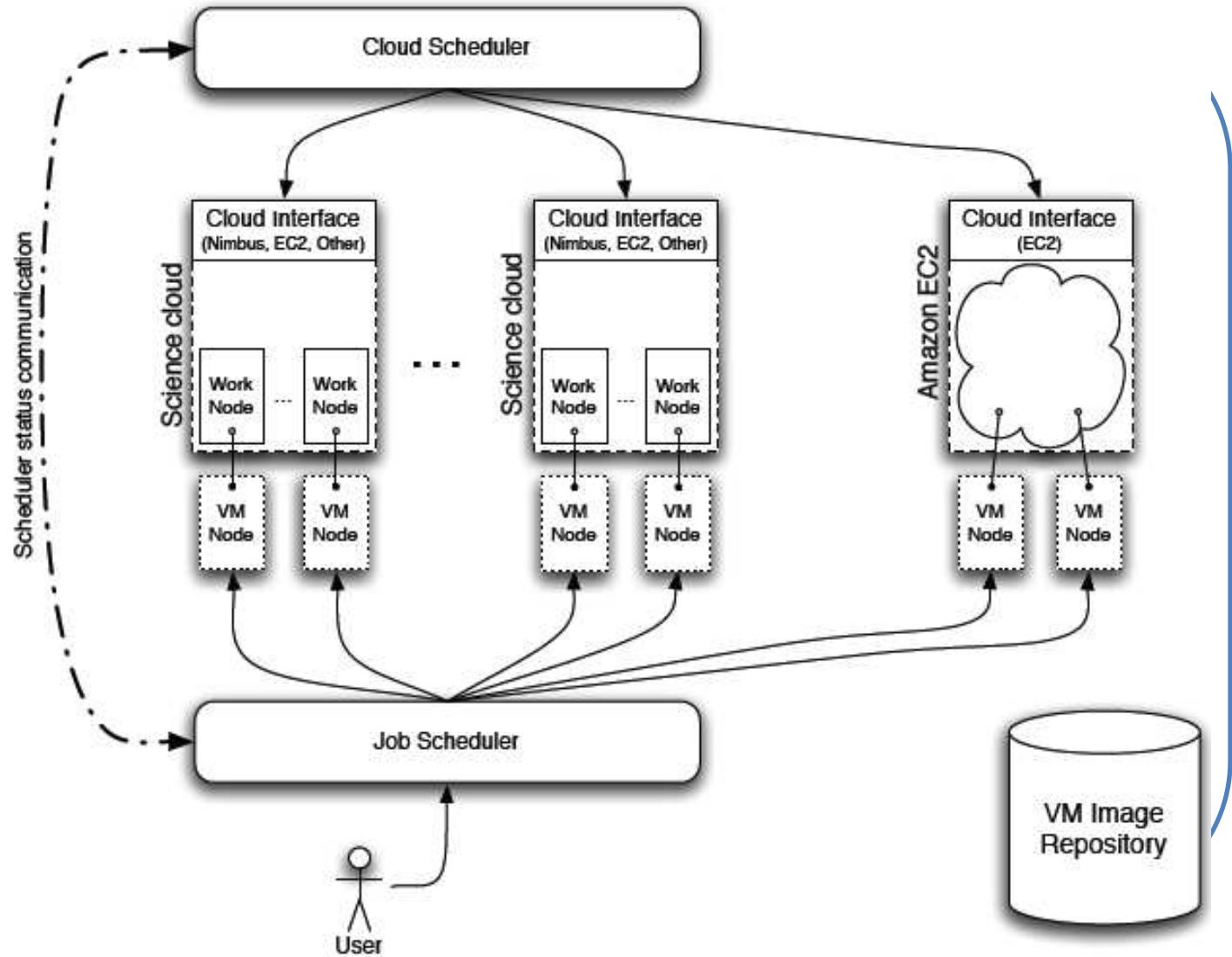
- Objetivo

- Prover um meio de gerenciar VMs customizadas pelo usuário em nuvens comerciais e científicas
- Nuvens comerciais incluem Amazon, RackSpace e IBM
- Nuvens científicas são nuvens estruturadas pelo governo para propostas científicos

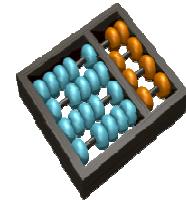
Armstrong et al (2010)



• Arquitetura

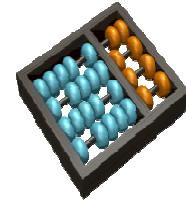


Armstrong et al (2010)



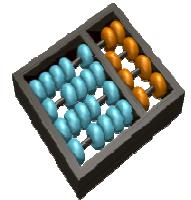
- Cloud scheduler
 - Rastreia jobs e recursos no conjunto de nuvens e jobs
- VM image repository
 - O usuário define sua VM e recarregando uma imagem base
- Cloud resources
 - Suporta Amazon, Nimbus, OpenNebula Eucalyptus
- Job scheduler
 - Condor HTC job scheduler
 - Foi escolhido porque trabalha bem com sistemas heterogêneos

Armstrong et al (2010)



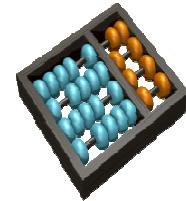
- Atualmente o sistema tem aplicações na área de:
 - Particle physics
 - Aplicações em C++ e Fortran
 - VM de 16 GB e 1 GB de RAM
 - Acesso a um banco de dados de 2 TB
 - Tipicamente a simulação gera arquivos de saída de 100 GB com 6 horas
 - 80 VMs
 - 3 nuvens
 - University of Victoria, National Research Council and Amazon EC2
 - O sistema processa 2000 *seven-hour jobs* em uma semana

Armstrong et al (2010)



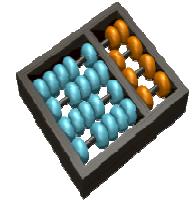
- Atualmente o sistema tem aplicações na área de:
 - Astronomy
 - Utiliza dois clusters
 - University of Victoria (25 máquinas e 200 cores)
 - Herzberg Institute of Astrophysics (6 máquinas e 32 cores)
 - Testado com sucesso com 9000 jobs utilizando 33000 **cores hours**

Tópicos



- Introdução
- Exemplo de Escalonador
 - Condor
- Escalonamento em Nuvens Híbridas
 - Artigos
- Conclusão

Artigo



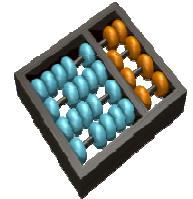
.HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds

.Autores

- Luiz Fernando Bittencourt
- Edmundo Roberto Mauro Madeira

.2011

Bittencourt e Madeira (2011)



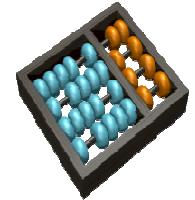
- Objetivo

- Minimizar o custo e o makespan
- Procura otimizar o custo da execução mantendo este menor que deadline

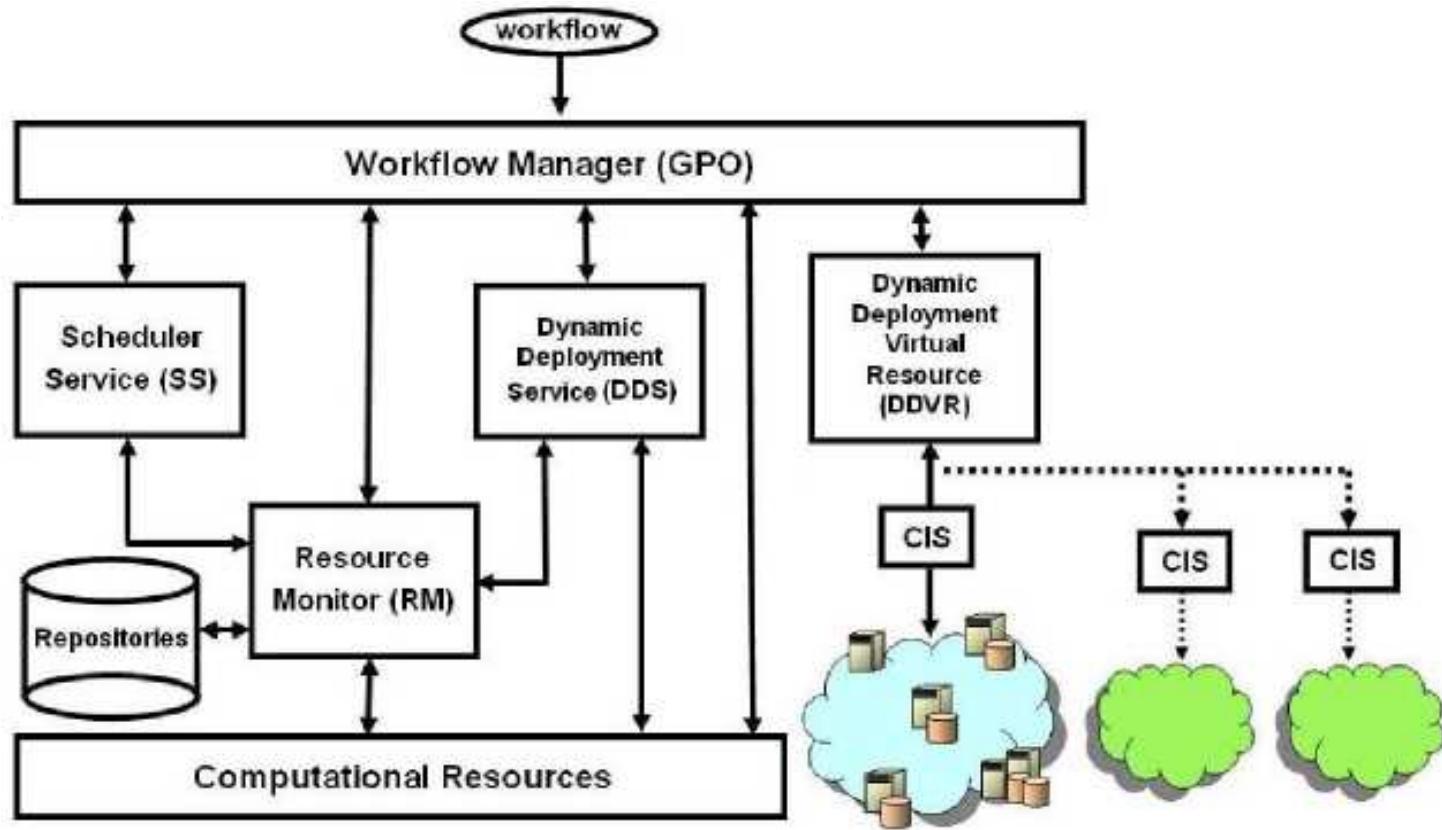
- As tarefas são dependentes

- Workflow

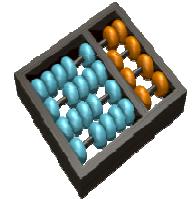
- Para aumentar o poder da nuvem privada, o algoritmo define quais recursos serão alugados da nuvem pública de forma a executar as tarefas dentro do prazo estabelecido



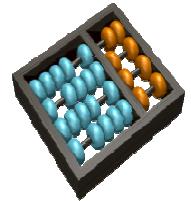
• Arquitetura



Bittencourt e Madeira (2011)



- Algoritmo de Escalonamento
 - Feito em duas fases
- Fase 1:
 - Escalona o workflow na nuvem privada
- Fase 2:
 - Enquanto o makespan for maior que o deadline
 - Selecionar tarefas a serem reescalonadas
 - Selecionar recursos da nuvem pública para compor a nuvem híbrida
 - Reescalonar as tarefas selecionadas na nuvem híbrida



- Definição de atributos

- Custo computacional

$$w_{i,r} = \frac{\text{instructions}}{p_r}$$

- Custo de comunicação

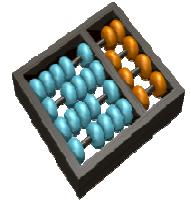
$$c_{i,j} = \frac{\text{data}_{i,j}}{l_{r,p}}$$

- Sucessores

- Conjunto de nós sucessores imediatos

- Predecessores

- Conjunto de nós predecessores imediatos



- Definição de atributos

- Prioridade

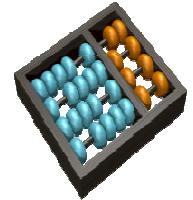
$$\mathcal{P}_i = \begin{cases} w_{i,r}, & \text{suc}(n_i) = \emptyset \\ w_{i,r} + \max_{\forall n_j \in \text{suc}(n_i)} (c_{i,j} + \mathcal{P}_j), & \text{otherwise} \end{cases}$$

- Menor tempo inicial

$$EST(n_i, r_k) = \begin{cases} Time(r_k), & \text{if } i = 1 \\ \max\{Time(r_k), ST_i\}, & \text{otherwise} \end{cases}$$

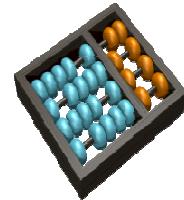
- Tempo final estimado

$$EFT(n_i, r_k) = EST(n_i, r_k) + w_{i,k}$$



• Escalonamento Inicial

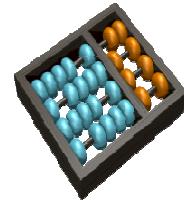
- Após o cálculo dos atributos, são criados grupos de nós (*cluster of tasks*)
 - Tais nós pertencem ao mesmo caminho no DAG
- As tarefas que pertencem ao mesmo *cluster* são escalonadas para o mesmo recurso
- O atributo Prioridade é utilizado para criar os *clusters*
- É escolhido um recurso que minimiza o tempo estimado de término



- Algoritmo de criação de *clusters*

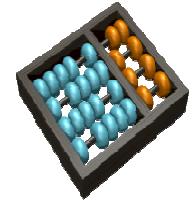
1. $n \in$ unscheduled node with highest Priority
2. $cluster \leftarrow cluster \cup n$
3. **while** (n has unscheduled predecessors) **do**
4. $n_{succ} \leftarrow$ successor_{*i*} of n with highest $P_i + EST_i$
5. $cluster \leftarrow cluster \cup n_{succ}$
6. $n \leftarrow n_{succ}$
7. **end while**
8. **return** $cluster$

Bittencourt e Madeira (2011)



- . Segunda Fase do escalonamento
 - . Enquanto o makespan for maior que o deadline
 - . Selecionar tarefas a serem re-escalonadas
 - . Selecionar recursos da nuvem pública para compor a nuvem híbrida
 - . Re-escalonar as tarefas selecionadas na nuvem híbrida
- . Selecionar tarefa
 - . Maior Prioridade, Maior EFT ou a soma dos dois atributos
- . Selecionar recurso
 - . Considera o número de clusters, custo, número de cores e performance

Bittencourt e Madeira (2011)

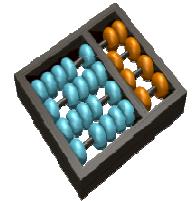


Algorithm 1 HCOC: the Hybrid Cloud Optimized Cost scheduling

```

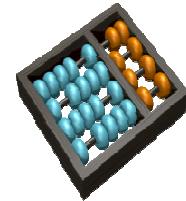
1:  $\mathcal{R}$  = all resources in the private cloud
2: Schedule  $\mathcal{G}$  in the private cloud using PCH
3: while  $makespan(\mathcal{G}) > \mathcal{D}$  AND  $iteration < size(\mathcal{G})$  do
4:    $iteration = iteration + 1$ 
5:   select node  $n_i$  such that  $\mathcal{P}_i$  is maximum and  $n_i \in \mathcal{T}$ 
6:    $\mathcal{H} = \mathcal{R}$ 
7:    $\mathcal{T} = \mathcal{T} \cup t$ 
8:    $num\_clusters =$  number of clusters of nodes in  $\mathcal{T}$ 
9:   while  $num\_clusters > 0$  do
10:    Select resource  $r_i$  from the public clouds such that
         $\frac{price_i}{num\_cores_i \times p_i}$  is minimum and  $num\_cores_i \leq num\_clusters$ 
11:     $\mathcal{H} = \mathcal{H} \cup \{r_i\}$ 
12:     $num\_clusters = num\_clusters - num\_cores_i$ 
13:   end while
14:   for all  $n_i \in \mathcal{T}$  do
15:     Schedule  $n_i$  in  $h_j \in \mathcal{H}$  such that  $EFT_i$  is minimum
16:     Recalculate  $ESTs$  and  $EFTs$ 
17:   end for
18: end while

```



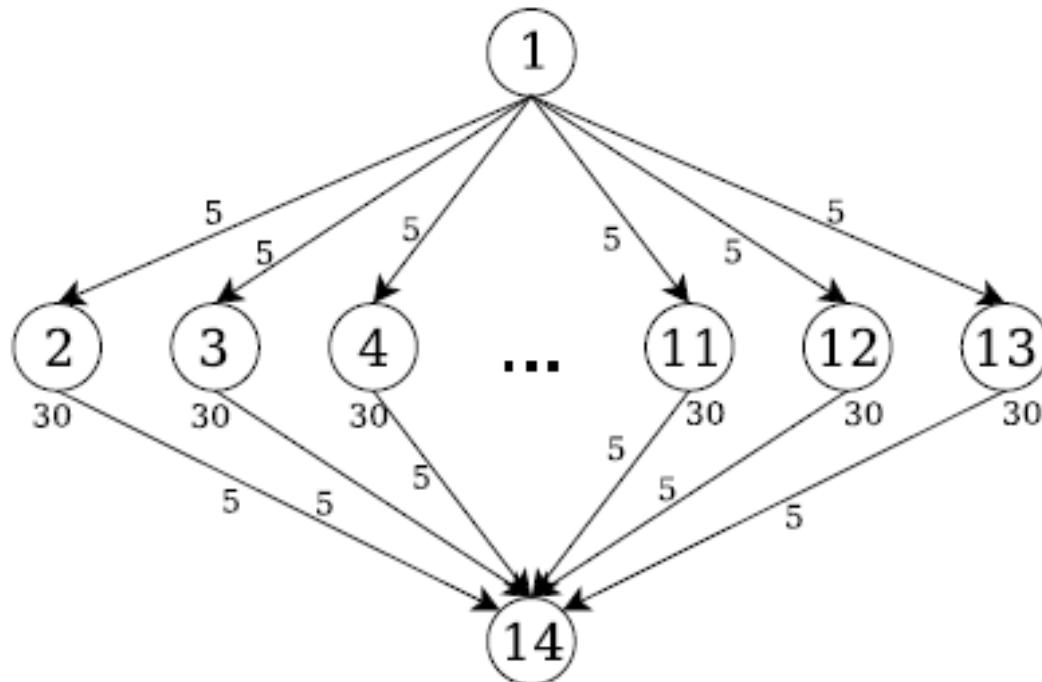
- Exemplo de execução
 - Tabela de recursos na nuvem privada

| Name | Cores | RAM | p_r per core |
|--------|-------|-------|----------------|
| Apolo | 2 | 2.5Gb | 1 |
| Cronos | 2 | 4Gb | 2 |

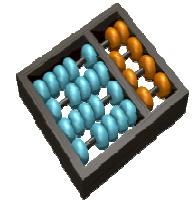


- Exemplo de execução

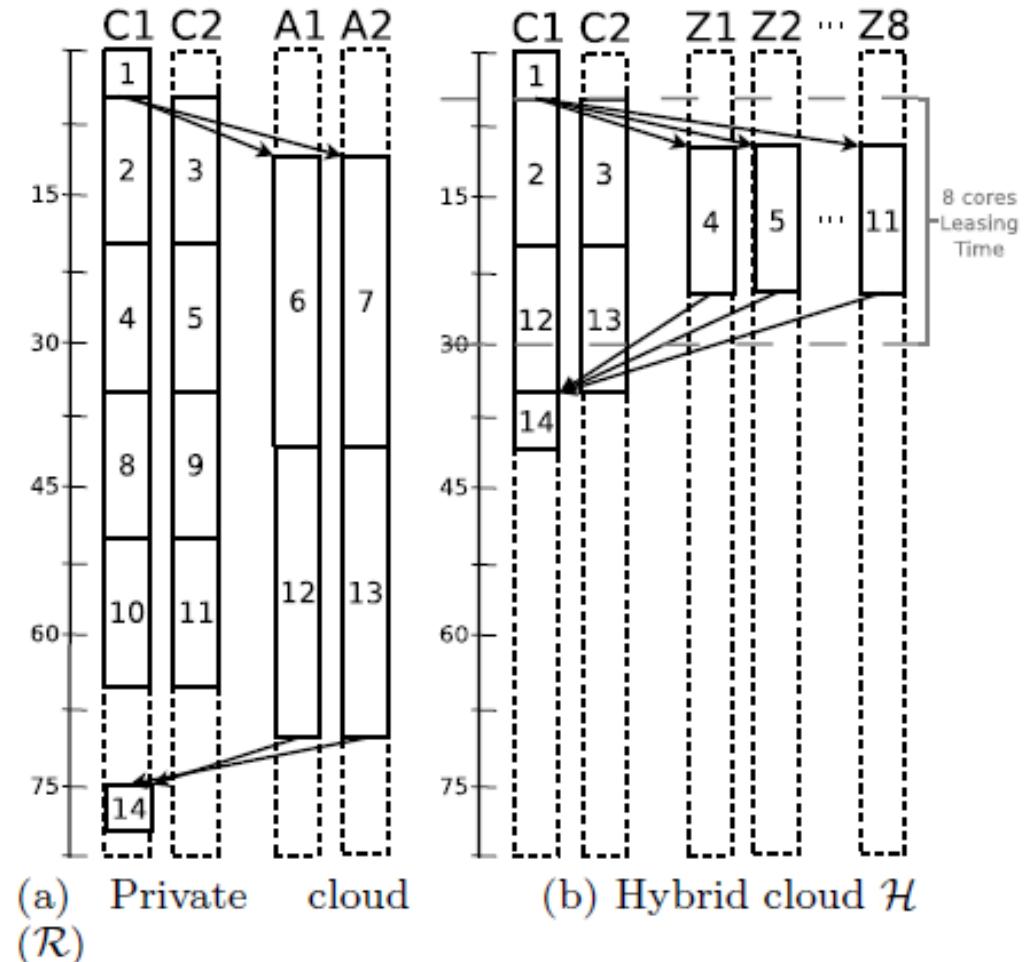
- DAG



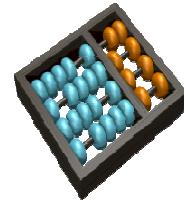
Bittencourt e Madeira (2011)



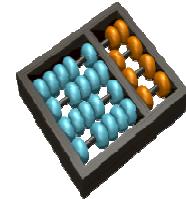
- Exemplo de execução
 - Resultado



Bittencourt e Madeira (2011)



- . De forma geral, o algoritmo proposto reduziu o custo de execução em nuvens públicas
- . Nos casos em que o tempo de execução desejado é muito pequeno, o algoritmo encontrou melhor escalonamento do que outros algoritmos propostos.



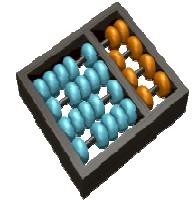
.Cost-effective Provisioning and Scheduling of Deadline-constrained Applications in Hybrid Clouds

- Autores

- Rodrigo N. Calheiros
- Rajkumar Buyya

- 2012

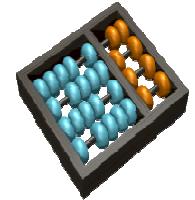
Calheiros e Buyya (2012)



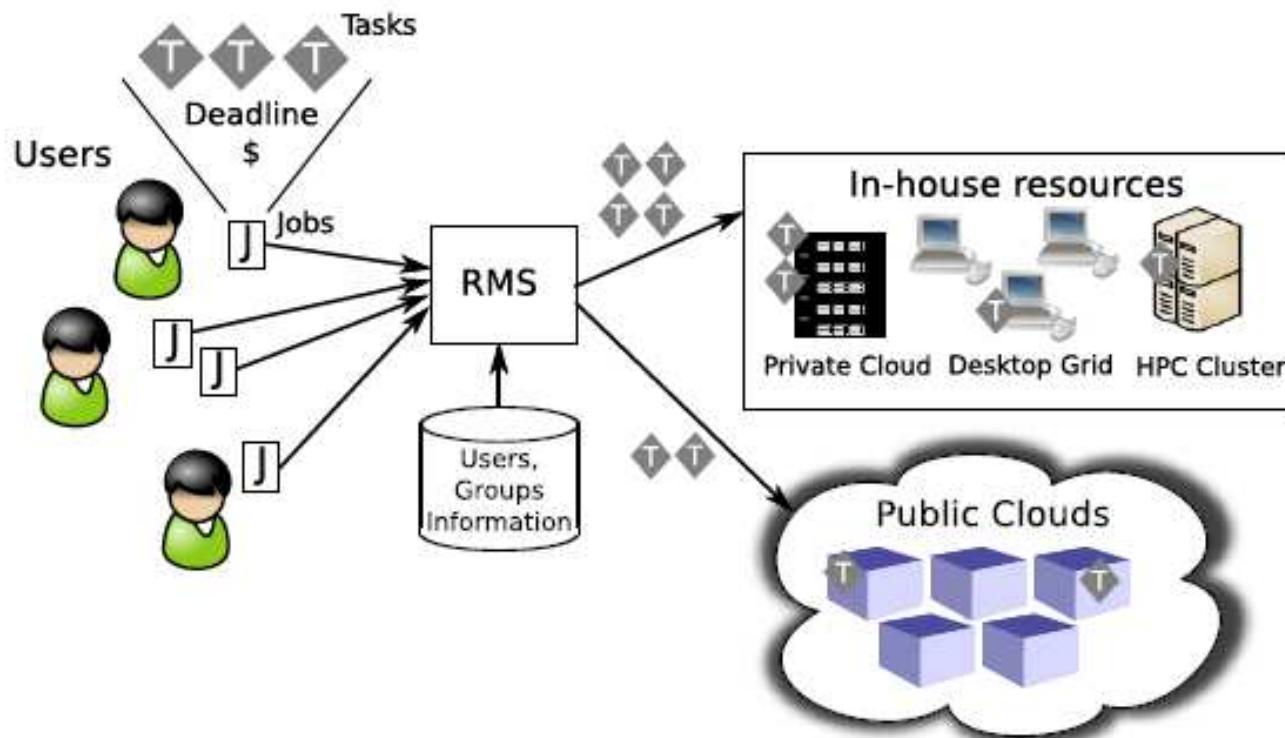
. Objetivos

- Apresentar uma arquitetura para coordenar fornecer e escalar recursos de forma a atender os requisitos das aplicações
- Utilizar a infraestrutura de uma nuvem pública para **acelerar** a execução de aplicações distribuídas
 - *Cloud bursting*
 - Alcançar o deadline
- Resource Management System (RMS)

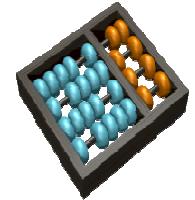
Calheiros e Buyya (2012)



- Modelo do Sistema

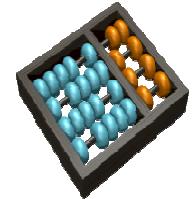


Calheiros e Buyya (2012)

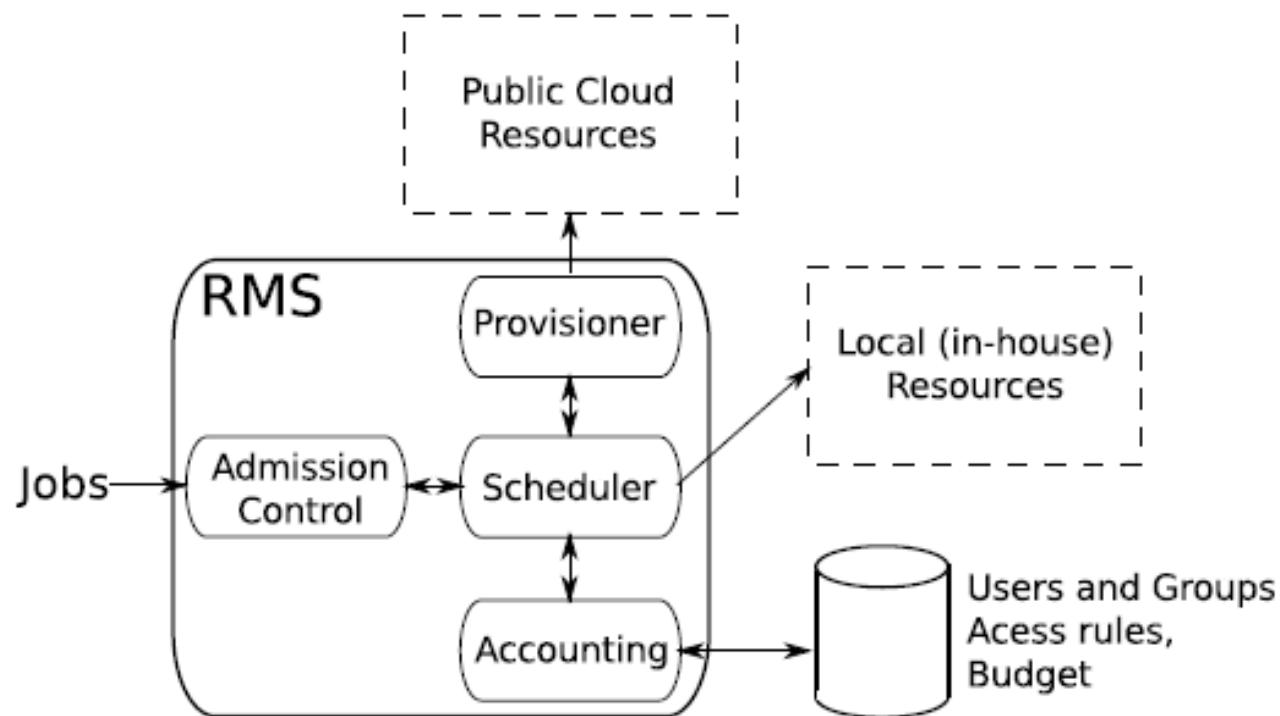


- User request (Job):
 - Descrição de cada tarefa do job
 - Arquivos necessários, tempo estimado
 - Opcionalmente requisitos de QoS e orçamento
- O modelo não considera dependência entre as tarefas
- Recursos são adquiridos quando há o risco de perder a tarefa por causa do deadline
 - Acelerar a execução

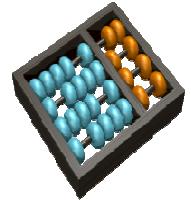
Calheiros e Buyya (2012)



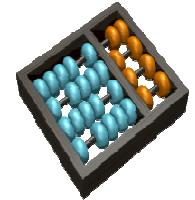
- Arquitetura proposta



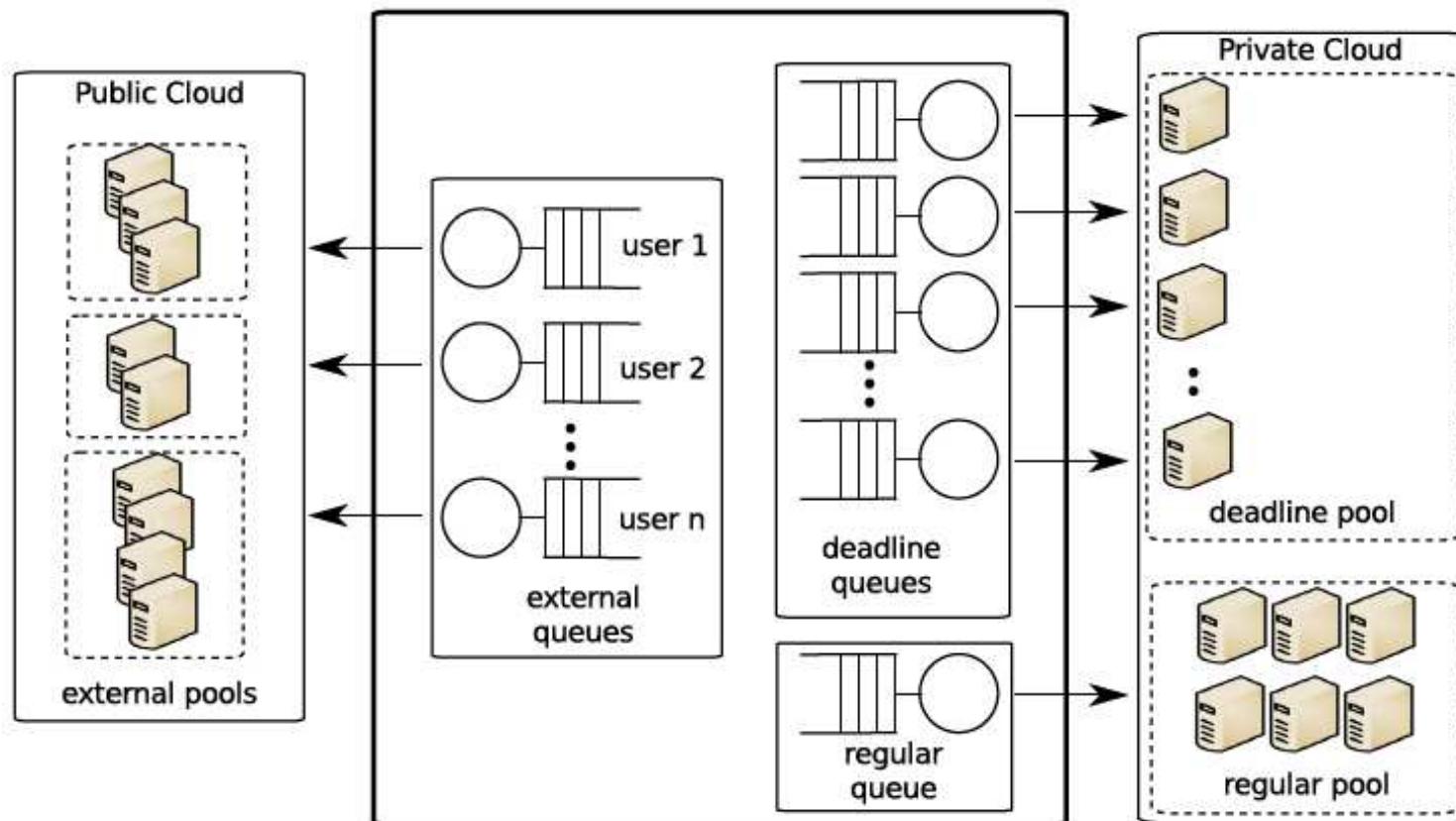
Calheiros e Buyya (2012)



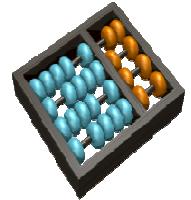
- . Admission Control
 - . Recebe e aceita as requisições do usuários
- . Scheduler
 - . Baseado nas informações das filas, dos deadlines e dos recursos disponíveis, requisita novos recursos
- . Provisioner
 - . Responsável por adquirir novos recursos das nuvens públicas
- . Accounting
 - . Define se o usuário tem créditos e autorização para requisitar novos recursos



- Organização do Escalonador

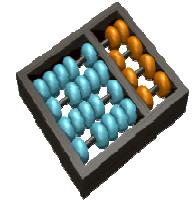


Calheiros e Buyya (2012)



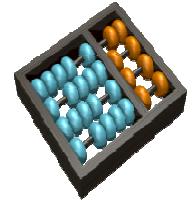
- . Regular pool previne que as tarefas fiquem sem recursos
 - . Garante o mínimo de recursos
- . Tarefas são realocadas quando as filas ficam vazias
- . Formação das filas
 - . Deadline pool => Round Robin, Worst Fit, Best Fit, ...
 - . Regular pool => Ordena por utilização

Calheiros e Buyya (2012)



• Resultados

- As simulações feitas apontam para a redução de 20% na utilização dos recursos da nuvem pública

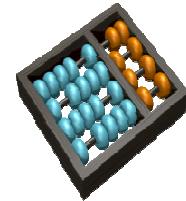


.Cost-optimal Scheduling in Hybrid Clouds for Deadline Constrained Workloads

. Autores

- . Ruben V.D. Bossche
- . Kurt Vanmechelen
- . Jan Broeckhove

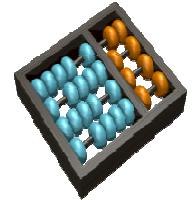
.2010



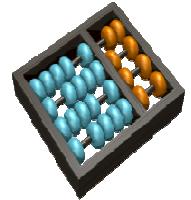
. Objetivo

- Propor um escalonador para maximizar a utilização dos recursos internos e minimizar o custo de rodar as tarefas em nuvens externas considerando as restrições das aplicações

Bossche, Vanmechelen e Broeckhove (2010)

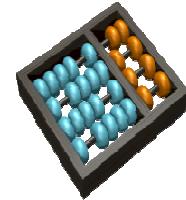


- A implementação do escalonador é feita a partir de uma formulação utilizando Programação Inteira Binária
- O escalonador considera preempção
- As tarefas são independentes



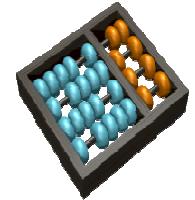
- Formulação em PIB

- A aplicações: a_1, \dots, a_A
- Deadline: dl_k
- T_k tarefas: t_{k1}, \dots, t_{Tk}
- C provedores: c_1, \dots, c_C
 - Para cada c_j
 - preços p_i e p_o (Tráfego de entrada e saída, em Gb)
 - p_{ij} : preço por hora de utilização da instância
 - I tipos de instâncias: it_1, \dots, it_l
 - Cada tipo possui cpu_i , mem_i
 - Nuvem privada: $maxcpu_j$, $maxmem_j$
 - Nuvem pública: “recursos ilimitados”



- Formulação em PIB

- A tarefa t_{kl} da aplicação a_k tem um tempo de execução r_{kli} em cada it_i
- A tarefa t_{kl} é associada com volume de tráfego de entrada n_{kli} e de saída n_{kli}
- $x_{klijs} = 1$; se a tarefa t_{kl} da aplicação a_k está executando o tipo de instância it_i da nuvem c_j ; 0 caso contrário!
- $y_{klijs} = 1$; se a tarefa t_{kl} finalizou



. Formulação em PIB

minimize

$$\begin{aligned} Cost = & \sum_{k=1}^A \sum_{l=1}^{T_k} \sum_{i=1}^I \sum_{j=1}^C y_{klij} \cdot (n_{ikl} \cdot p_{ij} + n_{okl} \cdot p_{oj}) \\ & + \sum_{k=1}^A \sum_{l=1}^{T_k} \sum_{i=1}^I \sum_{j=1}^C \sum_{s=1}^S (p_{ij} \cdot x_{klij_s}) \end{aligned} \quad (1)$$

subject to

$$\forall k \in [1, A], l \in [1, T_k]:$$

$$\sum_{i=1}^I \sum_{j=1}^C y_{klij} = 1 \quad (2)$$

$$\forall k \in [1, A], l \in [1, T_k], i \in [1, I], j \in [1, C]:$$

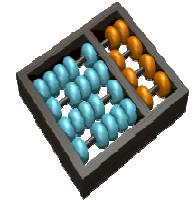
$$\sum_{s=1}^{dl_k} x_{klij_s} = y_{klij} \cdot r_{kli} \quad (3)$$

$$\forall j \in [1, C], s \in [1, S]:$$

$$\sum_{k=1}^A \sum_{l=1}^{T_k} \sum_{i=1}^I cpu_i \cdot x_{klij_s} \leq maxcpu_j \quad (4)$$

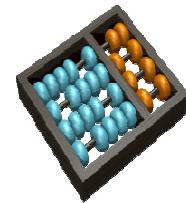
$$\forall j \in [1, C], s \in [1, S]:$$

$$\sum_{k=1}^A \sum_{l=1}^{T_k} \sum_{i=1}^I mem_i \cdot x_{klij_s} \leq maxmem_j \quad (5)$$

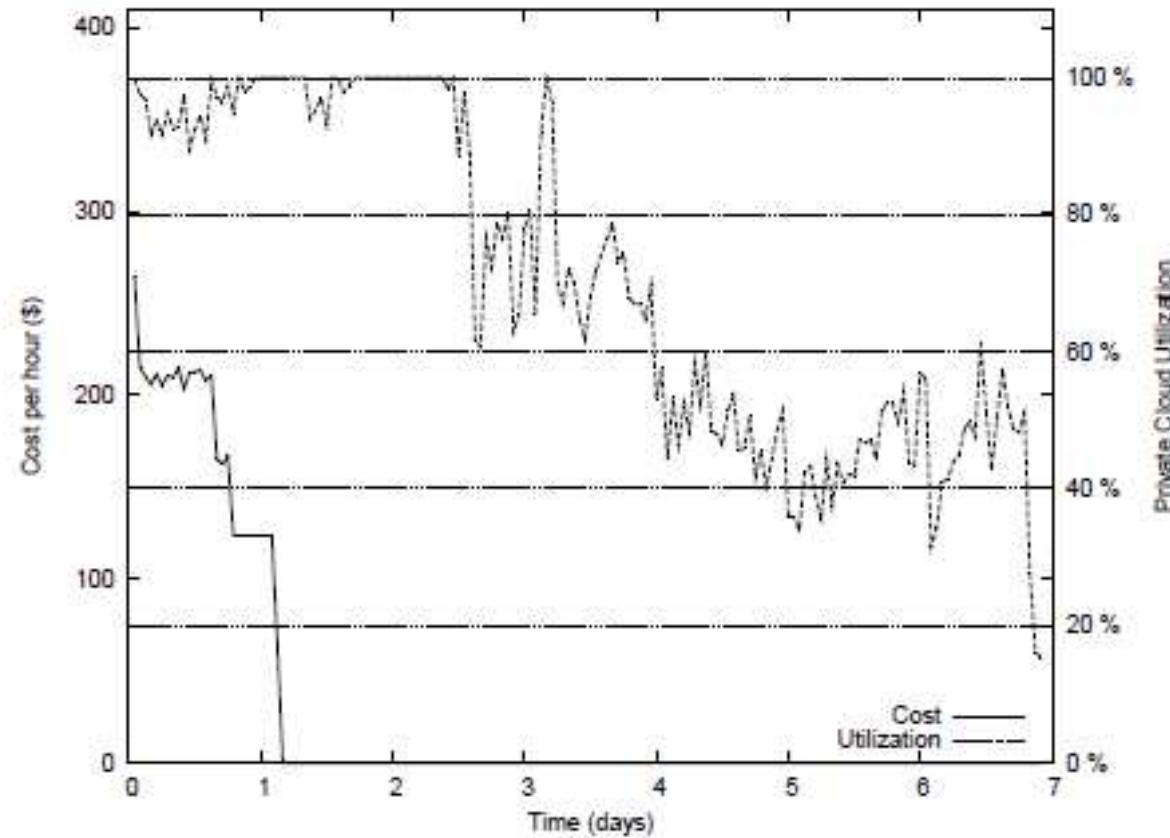


- Formulação em PIB

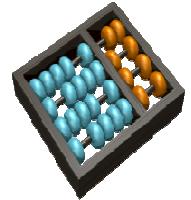
- (1) Representa o custo do tráfego de dados somado ao custo computacional
- (2) Garante que cada tarefa é escalonada em uma única instância e provedor
- (3) Garante que todas as tarefas de uma aplicação finalizaram antes do deadline
- (4) e (5) Limitam a utilização de CPU e memória
 - Aplica-se apenas a nuvem privada



• Resultados

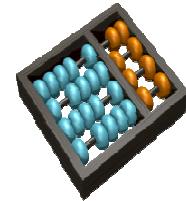


Resumo



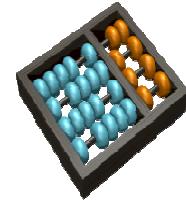
| Artigo | Alvo | Criterio | Multi-core | On-demand | Workflow |
|--|-------------|--|-------------------|------------------|-----------------|
| Bittencourt e Madeira (2011) | Nuvens | Min. cost com deadline | Sim | Sim | Sim |
| Calheiros e Buyya (2012) | Nuvens | Min. Makespan | Não | Sim | Não |
| Bossche, Vanmechelen e Broeckhove (2010) | Nuvens | Maximizar utilização e minimizar custo | Não | Não | Sim |

Tópicos



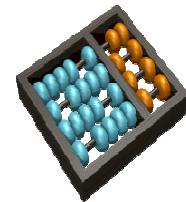
- Introdução
- Exemplo de Escalonador
 - Condor
- Escalonamento em Nuvens Híbridas
 - Artigos
- Conclusão

Conclusão

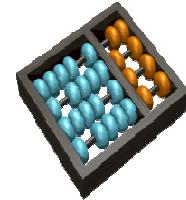


- . Este trabalho discutiu o problema de escalonamento e mostrou os conceitos, a estrutura e o algoritmos de alguns escalonadores para Nuvens Híbridas
- . Embora essa área seja largamente explorada, o escalonador ideal está longe de ser desenvolvido
- . Novos aspectos podem ser adicionados/tratados para propor novas soluções. Logo, essa área ainda deve ser muito explorada por pesquisadores

Perguntas / Dúvidas

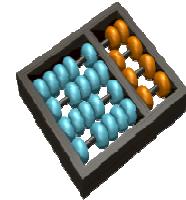


Referências



- (Boutaba, 2010) Q. Zhang, L. Cheng, e R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.
- (Bittencourt e Madeira, 2012) L. F. Bittencourt, E. R. M. Madeira e N. L. S. Fonseca, "Scheduling in Hybrid Clouds," *IEEE Communications Magazine*, pp. 42-47, 2012.
- (Armstrong et al, 2010) P. Armstrong, A. Agarwal, A. Bishop, A. Charboneum, R. Desmarais, "Cloud Scheduler: a resource manager for distribute compute clouds". 2010
- (Tannenbaum, Wright, Miller e Livny, 2001) Todd Tannenbaum, Derek Wright, KAREN Miller e Miron Livny, "Condor: A distributed Job Scheduler", 2001.
- (Bittencourt e Madeira, 2011) L. F. Bittencourt and E. R. M. Madeira, "HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds," *J. Internet Svcs. and Apps.*, vol. 2, no. 3, Dec 2011, pp. 207–27
- (Calheiros e Buyya, 2012) Rodrigo N. Calheiros, Rajkumar Buyya: Cost-Effective Provisioning and Scheduling of Deadline-Constrained Applications in Hybrid Clouds. *WISE 2012*: 171-184
- (Bossche, Vanmechelen e Broeckhove, 2010) Ruben Van den Bossche, Kurt Vanmechelen, Jan Broeckhove: Cost-Efficient Scheduling Heuristics for Deadline Constrained Workloads on Hybrid Clouds. *CloudCom 2011*: 320-327
- (Bossche, Vanmechelen e Broeckhove, 2011) Ruben Van den Bossche, Kurt Vanmechelen, Jan Broeckhove: Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds. *Future Generation Comp. Syst.* 29(4): 973-985 (2013)

Pergunta



Todas as informações e estimativas fornecidas como entrada para o escalonamento são consideradas corretas. Definir tais informações pode não ser uma tarefa fácil.

Qual o impacto no escalonamento caso essas informações sejam imprecisas? De forma resumida, cite uma estratégia para tratar tal problema.