risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido/Submission: 22/01/2021
Aceitação/Acceptance: 10/04/2021

Escalonamento de pedidos no paradigma *Fog Computing:* proposta de um modelo sensível ao contexto e avaliação do seu desempenho

Celestino Barros[1], Vítor Rocio[2], André Sousa[3], Hugo Paredes[4]

celestino.barros@docente.unicv.edu.cv; vitor.rocio@uab.pt; andresousa@utad.pt; hparedes@utad.pt

[1] Faculty of Science and Technology of University of Cape Verde, Praia, Cabo Verde

[2] INESC TEC and Open University of Portugal, Lisbon, Portugal

[3] Critical TechWorks, Porto, Portugal

[4] INESCT TEC and University of Trás-os-Montes and Alto Douro, Vila Real

DOI: 10.17013/risti.42.93–119

Resumo: os pedidos de execução de aplicações na arquitetura *cloud* e no paradigma *fog* são geralmente heterogéneos e o escalonamento nessas arquiteturas é um problema de otimização com múltiplas restrições. Neste artigo, fizemos um levantamento sobre os trabalhos relacionados com o escalonamento na arquitetura *cloud* e no paradigma *fog*, identificamos as suas limitações, explorarmos perspetivas de melhorias e propomos um modelo de escalonamento sensíveis ao contexto para o paradigma *fog*. A solução proposta utiliza a normalização *Min-Max*, para resolver a heterogeneidade e normalizar os diferentes parâmetros de contexto. A prioridade dos pedidos é definida através da aplicação da técnica de análise de Regressão Linear Múltipla e o escalonamento é feito utilizando a técnica de Otimização de Programação Não Linear Multiobjetivo. Os resultados obtidos a partir de simulações no kit de ferramentas *iFogSim*, demonstram que a nossa proposta apresenta um melhor desempenho em comparação com as propostas não sensíveis ao contexto.

Palavras-chave: sensibilidade ao contexto; qualidade de experiência; escalonamento sensível ao contexto; modelo de escalonamento; arquitetura *cloud* e paradigma *fog computing*.

Task scheduling in the Fog Computing paradigm: proposal of a context-aware model and evaluation of its performance

Abstract: Application execution requests in cloud architecture and fog paradigm are generally heterogeneous and scheduling in these architectures is an optimization problem with multiple constraints. In this paper, we conducted a survey on the related works on scheduling in cloud architecture and fog paradigm, we identify their limitations, we explore some prospects for improvements and we propose a context-aware scheduling model for fog paradigm. The proposed solution uses Min-Max normalization, to solve heterogeneity and normalize the different

context parameters. The priority of requests is set by applying the Multiple Linear Regression analysis technique and the scheduling is done using the Multiobjective Nonlinear Programming Optimization technique. The results obtained from simulations on iFogSim toolkit, show that our proposal performs better compared to the non-context-aware proposals.

Keywords: context-aware; quality of experience; context-aware task scheduling; scheduling model; cloud architecture and fog paradigm.

1. Introdução

As técnicas atuais de computação que utilizam a *cloud* estão-se a tornar insustentáveis, visto que bilhões de dispositivos, impulsionados sobretudo pelo rápido crescimento da *Internet of Things* (IoT) estão conectados à Internet. Os dados obtidos pelos sensores e aplicações têm aumentado exponencialmente e muitos destes dispositivos permitem agregar num único aparelho funcionalidades de execução de aplicações, comunicação, entretenimento, jogos, entre outros. Além disso, uma das suas principais características é a sua capacidade de identificar e partilhar diferentes tipos de informações ao nível de utilizadores, dispositivo, aplicações e rede. Por outro lado, eles possuem várias limitações como: capacidade de processamento reduzida, escassez de recursos, autonomia reduzida da bateria, baixa conetividade, entre outras. Estas limitações impõem aos analistas e desenvolvedores a adoção de serviços que ampliam a capacidade dessas aplicações em execução nesses dispositivos através da utilização de serviços hospedados na *cloud* (Fernando, Loke, & Rahayu, 2013).

Não obstante as vantagens, em algumas situações, não é benéfica a utilização da arquitetura *cloud*. Ela é centralizada e consequentemente o processamento é feito em *data centers* concentrados, para otimização de custos energéticos e de comunicações. Diferentes técnicas que minimizam a execução na *cloud* através do processamento local em elementos periféricos que permitem resolver limitações deste modelo têm sido propostas. E uma dessas técnicas passa pela utilização do paradigma *fog* (Bonomi *et al.,* 2014) que Segundo OpenFog (2017), é uma extensão da *cloud* e tem sido proposta para colmatar as inconveniências deste, visto que visa reduzir, entre outros, o tempo de resposta, consumo de energia e latência. No entanto, devido a sua densidade e heterogeneidade o escalonamento de tarefas precisa ser tratado com perícia e continuam a apresentar alguns desafios aliciantes que nos levam a questionar a forma como as tarefas são encaminhadas entre os diferentes dispositivos físicos, nós da *fog* e *cloud*.

Na *fog*, devido à maior densidade e heterogeneidade de dispositivos, o escalonamento é muito complexo e, na literatura, ainda existem poucos estudos. Contrariamente, o escalonamento na *cloud* que é amplamente estudado. Muitas pesquisas abordam, no entanto, essa questão na perspetiva de provedores de serviço ou otimizam os níveis da qualidade de serviço (QoS) da aplicação. Ignoram, porém, informações contextuais ao nível do dispositivo e dos utilizadores finais e as suas experiências de utilização (QoE).

Neste artigo, fizemos um levantamento dos trabalhos relacionados tanto na arquitetura *cloud* como no paradigma *fog*, explorámos as suas limitações e sugerimos algumas perspetivas de melhorias. Com base nas sugestões, propomos um modelo

de escalonamento sensível ao contexto para o paradigma *fog* e analisamos o seu desempenho.

Sendo o objetivo principal deste artigo, a elaboração de uma proposta que tenha como âmbito a criação de conhecimento na forma de técnicas, métodos, modelos e teoria, a metodologia de investigação utilizada na sua conceção foi o *Design Science Research (DSR)*. Está dividido em seis secções sendo a primeira, a introdução, a segunda, a contextualização, a terceira, o levantamento dos trabalhos relacionados. Na quarta, são descritos os contextos previstos, o modelo proposto e as suas arquiteturas. Na quinta, é avaliado o desempenho do modelo proposto e feita a comparação com propostas não sensíveis ao contexto e na sexta, é feita a conclusão do artigo.

Como resultado e com base no levantamento dos trabalhos relacionados e na identificação das suas limitações, propomos um modelo de escalonamento de tarefas sensível ao contexto para a arquitetura *fog* e analisamos o seu desempenho comparando-o com propostas não sensíveis ao contexto.

2. Contextualização

Escalonamento de tarefas refere-se à atribuição de recursos necessários para a execução de uma tarefa. Assume-se como um processo essencial para melhorar a confiabilidade e a flexibilidade dos sistemas e exige algoritmos capazes de escolher o recurso mais adequado disponível para executar a tarefa (Swaroop, 2019).

Pretende-se que os pedidos sejam executados tendo em consideração as restrições definidas, como: tempo, custo, duração da bateria, níveis de sinal da rede, QoS da aplicação, entre outros (Swaroop, 2019). Avança ainda, que devido sua a complexidade, o paradigma *fog*, apresenta alguns desafios aliciantes que nos leva a questionar a forma como as tarefas são encaminhadas entre dispositivos clientes, nós da *fog*, servidores *cloud*, entre outros.

Sistemas lidam com pedidos prioritários, tarefas prioritárias e/ou com requisitos restritos de QoS. Para garantir o seu bom funcionamento e a sua execução dentro dos limites de tempo definidos, o escalonamento precisa ser tratado com perfeição. Escalonamento eficiente de tarefa deve garantir processamento simultâneo e eficiente independentes do seu fluxo. No paradigma *fog*, o escalonamento constitui um desafio e exige algoritmos avançados capazes de escolher o recurso mais adequado e disponível para executar a tarefa (Mahmud *et al.,* 2016).

2.1. Conceção de algoritmos de escalonamento

Conforme Swaroop (2019), na conceção de um algoritmo de escalonamento, devem ser obedecidas restrições como: custo de tarefas; dependências entre as tarefas e a sua localização. Avança ainda, que as decisões de escalonamentos podem ser estáticas - onde decisão é tomada durante a compilação. Ou dinâmicas - onde são utilizadas informações sobre o estado do fluxo de tarefas num determinado instante durante a execução para a tomada de decisões de escalonamento. Constitui a melhor abordagem, porque possibilita que vários problemas tenham solução que podem ser representados numa árvore de

pesquisa. Por outro lado, esses problemas são exigentes computacionalmente, requerem estratégia de paralelização e balanceamento de carga dinâmico.

2.2. Contexto

Segundo Sousa (2017), o termo "contexto" é algo que, à partida, pode parecer simples, mas à medida que refletimos sobre ela, a sua definição torna-se cada vez mais complexa. Em computação móvel o contexto de um utilizador é muito dinâmico. Ao utilizar aplicações nesse ambiente, o comportamento dessa aplicação deve ser personalizado para a situação atual do utilizador. Para promover uma utilização efetiva do contexto, muitos autores disponibilizam as suas definições de contexto. Bazire & Brézillon (2005), examinaram 150 definições de contexto, nas diferentes áreas de investigação e concluíram que a criação de uma única definição é um esforço árduo e provavelmente, impossível visto que a mesma varia com a área científica e depende principalmente do campo em que ela está a ser aplicada. No entanto, avançam que contexto é um conjunto de restrições que influenciam o comportamento relativo a uma determinada tarefa.

A definição de contexto mais utilizada atualmente, mesmo noutros campos, como na vertente da operacionalização foi proferida em Dey (2001), onde ela é descrita como: *"Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between user and an application, including the user and applications themselves."*

3. Trabalhos relacionados

Segundo o conhecimento dos autores, pesquisas sobre algoritmos de escalonamento sensíveis ao contexto no paradigma *fog* são muito recentes e, na arquitetura *cloud*, encontramos algumas propostas. Os autores em Deng *et al.* (2016), Li *et al.* (2017), Lawanyashri *et al.* (2017) e Yang *et al.* (2018) propuseram abordagens que enfatizam a redução do consumo de energia dos *data centers* e nota-se uma grande preocupação com a eficiência energética.

Em Zhou *et al.* (2015), é proposto um algoritmo que se baseia na QoS para a *Mobile Cloud Server*, onde os atributos de tarefas como: privilégios dos utilizadores; tamanho da tarefa; expetativa e o tempo suspenso na fila são utilizados para calcular a prioridade.

Cardellini *et al.* (2015), propuseram um escalonador distribuído sensível ao QoS para o processamento de fluxo de dados no paradigma *fog*. No entanto, em topologias que envolvem muitos operadores funciona de forma instável.

Em Stavrinides & Karatza (2019), é proposto um algoritmo leva em consideração o custo da comunicação decorrente da transferência de dados dos sensores e dispositivos da camada da *fog* durante o processo de escalonamento e em Aazam *et al.* (2016), é proposto um algoritmo que visa otimizar a utilização dos recursos e QoS.

Skarlat *et al.* (2017), definiram uma política de escalonamento que visa a colocação de aplicações sensíveis à QoS em nós da *fog*. Em Intharawijitr, Iida & Koga (2016), tentam garantir a melhor utilização da largura de banda disponibilizada.

Em Sheikhalishahi *et al.* (2015), Deng *et al.* (2016), Lawanyashri *et al.* (2017), há uma evidente preocupação dos autores com a eficiência energética dos recursos. O algoritmo proposto em Shojafar *et al.* (2015) e Fun *et al.* (2017), pretende cumprir o *deadline* das tarefas e aumentar os lucros do provedor de serviços da *fog*. Nestas cinco últimas propostas, há uma evidente preocupação dos autores em focar e defender principalmente os interesses dos provedores de serviços, ao invés de considerarem os contextos dos utilizadores finais e as suas experiências de utilização.

Li *et al.* (2017), desenvolveram um algoritmo de escalonamento cooperativo focado em melhorar as recompensas associado as recolhas de dados nas regiões suburbanas.

O algoritmo proposto por Ghouma e Jaseemuddin (2015), explora contexto como níveis da conetividade da rede do dispositivo e o nível de bateria, para a definição das tarefas a serem escalonadas. Enquanto que o proposto por Zhou *et al.* (2017), exploram os níveis da conectividade da rede e os recursos de Maquinas Virtuais (MV) alugados para disponibilizar decisões de *code offloading*. Mahmud *et al.* (2016), propõem uma política de escalonamento de aplicação sensíveis ao contexto para *Mobile Cloud Computing* que é executado em uma *Cloudlet*.

O algoritmo proposto em Zhu *et al.* (2015), é utilizado em tarefas agrupadas com objetivo de minimizar o tempo de execução. O proposto em Oueis, Strinati & Barbarossa (2015), apesar de proporcionar ganho de latência e baixo consumo de energia, tal como os propostos em Cardellini *et al.* (2015), e Zhu *et al.* (2015), o desempenho degrada-se quando é utilizado em arquitetura *fog* de grande escala.

Em Gill, Garraghan & Buyya (2019), é proposta uma técnica de gestão de recursos para ambientes *cloud* e *fog* sensíveis ao QoS que, contrariamente ao escalonador padrão da *fog*, aproveita da otimização *Particle Swarm* para otimizar em simultâneo vários parâmetros do contexto.

Em Aazam *et al.* (2016), é proposto um algoritmo que objetiva estimar os recursos da *fog* com intuito de otimizar a QoE. Em vez disso, Skarlat *et al.* (2017), propuseram uma abordagem que visa escalonar aplicações sensíveis à QoS em recursos virtualizados da *fog*. Contrariamente, em Zhu *et al.* (2015), Oueis, Strinati & Barbarossa (2015), Mahmud *et al.* (2016) e Aazam *et al.* (2016), foram tidas em conta as questões relacionadas com o aprimoramento da QoE do utilizador final.

Muitas propostas, como as descritas em Sheikhalishahi *et al.* (2015), Lawanyashri *et al.* (2017), Tiwary *et al.* (2018) e Shinde *et al.* (2018), abordam o problema da otimização sob a perspetiva dos provedores de serviço e ignoram questões contextuais dos utilizadores finais e as suas experiências de utilização. Outras, como as definidas em: Cardellini *et al.* (2015), Aazam *et al.* (2016), Skarlat *et al.* (2017) e Gill, Garraghan & Buyya (2019), pretendem sobretudo otimizar os níveis de QoS da aplicação e alguns concentram apenas no escalonamento de tarefas na arquitetura *cloud* e no paradigma *fog*. Outros, ainda, preocupam-se com a eficiência energética.

3.1. Limitações dos trabalhos relacionados

Diferentes escalonadores têm as suas próprias deficiências, seguidamente destacaremos algumas limitações dos trabalhos relacionados:

- *Análise de políticas na perspetiva de serviços*, a maioria dos escalonadores analisa as políticas apenas na perspetiva de serviço. A otimização dos custos para utilizadores, bem como, o aperfeiçoamento da QoE dos utilizadores não é tido em consideração.
- *Desconhecimento do contexto do utilizador final*, nas técnicas de escalonamento sensíveis ao contexto estudados, os pedidos dos utilizadores finais são analisados num escopo restrito. A força do sinal associado a um pedido, por exemplo, não é tida em consideração. Como consequência, qualquer dispositivo pode ficar desconectado antes ou enquanto obtém resposta de um pedido. O nível da bateria do dispositivo dos utilizadores finais também é ignorado. Para garantir que um pedido sempre tenha respostas em tempo oportuno, um limite para o nível da bateria deve ser preservado.
- *Escalonamento deficiente de tarefas*, escalonadores de tarefas básico como as definidas em Yang *et al.* (2018), Li *et al.* (2017), Lawanyashri (2017), Deng *et al.* (2016) e Qiu *et al.* (2015), são aqueles que privilegiam a eficiência energética e não consideram a sensibilidade ao contexto, a QoS e a QoE.
- *Priorização inadequada de tarefas*, alguns escalonadores baseados em prioridade foram estudados. No entanto, muitos não descrevem a forma como a prioridade é definida e outros não explicam claramente a metodologia utilizada para a priorização de tarefas.
- *Aumento do tempo médio de espera,* geralmente à medida que os pedidos aumentam, o tempo médio de espera tende também a aumentar proporcionalmente. Nos escalonadores analisados, nenhuma compensação para esse problema é proposta.
- *Qualidade de experiência subtil,* apesar de existirem alguns algoritmos de escalonamentos na arquitetura *cloud* e no paradigma *fog* que privilegiam a QoS para a priorização das tarefas, elas não se focalizam em otimizar a QoE do utilizador.
- *Supervisão para a preservação da QoS*, os escalonadores analisados não fazem supervisão com objetivo de preservar adequadamente a qualidade de serviço. Isto é, o tempo máximo permitido para a obtenção de respostas não é considerado em algumas propostas e em outras, é considerado de forma inadequada.

3.2. Perspetivas de melhoria dos trabalhos relacionados

Alguns aspetos podem ser explorados por forma a melhorar as estratégias existentes:

- *Consciencialização do contexto no escalonamento de tarefas:* várias pesquisas asseguram que os escalonadores sensíveis ao contexto são eficientes no aperfeiçoamento da QoS e otimizam os custos, tanto do ponto de vista dos utilizadores como dos provedores de serviço.
- *Priorização de tarefas sensíveis ao contexto:* devem ser introduzidos modelos de escalonamento onde as prioridades são definidas com base no contexto dos pedidos.
- *Preservação da restrição de energia:* a restrição energética deve ser levada em consideração durante o escalonamento de tarefas. O nível da bateria do dispositivo do utilizador final associado a um pedido deve disponibilizar um

nível limite da bateria, para que o dispositivo solicitante seja preservado até ao fim da execução.
- *Conservação da força do sinal da rede:* a intensidade do sinal associada a um pedido deve assegurar a força mínima do sinal de forma a possibilitar ao utilizador solicitar recursos e ter respostas em tempo hábil.
- *Salvaguarda da QoS:* atendendo que as tarefas descarregadas na *fog* são heterogéneas, o escalonador de tarefas deve considerar que o tempo necessário para a obtenção de resposta deve estar confinado dentro dos limites temporais previstos.
- *Redução do tempo médio de espera*: um mecanismo de compensação relativamente ao tempo médio de espera deve ser introduzido pelo escalonador de tarefas de forma a que o aumento do tempo de espera seja proporcionalmente menor em relação à chegada das tarefas.
- *Otimização da QoE:* os algoritmos de escalonamentos devem também concentrar em otimizar tanto a QoE dos utilizadores finais como a QoS.

Propomos na secção 4, um modelo e arquitetura de escalonamento de tarefas sensível ao contexto para o paradigma *fog*, que incorporam estas sugestões.

4. Modelo e arquitetura proposta

Segundo Fun *et al.* (2017), ao contrário da arquitetura *cloud*, o paradigma *fog* é composta por, no mínimo, três camadas, isto é, nós da borda adicionais são introduzidos entre o utilizador e a *cloud*.

No nosso modelo assumimos que uma técnica de *code offloading* adequado (por exemplo, MAUI definido em Bahl *et al.* (2010), COMET apresentado em Gordon *et al.* (2012), entre outros) esteja a ser executada nos dispositivos móveis a fim de tomar a melhor decisão em relação ao descarregamento ou não de códigos e em quais nós da *fog* (Berg, Durr & Rothermel, 2014).

Consideramos que um pedido inclui o atraso máximo permitido para executar a aplicação, o nível de bateria do dispositivo e os valores da força do sinal da rede. Também assumimos, tal como em Deng *et al.* (2016), que a *fog* disponibiliza capacidades computacionais muito maiores que os dispositivos móveis e deve ter a capacidade de extrair os contextos associados aos pedidos e tomar a decisão de escalonamento em conformidade.

Musumba e Nyongesa (2013), definiram como os principais contextos que podem ser explorados em qualquer ambiente de computação móvel: a ligação a rede; processadores disponíveis; nível de bateria, a localização; a largura de banda da rede; o tráfego na rede; as maquinas virtuais alugadas e requisitos de QoS da aplicação.

No nosso domínio do problema, os contextos dos provedores de serviços foram ignorados. Também, depois de descarregado a tarefa na *fog*, torna-se desnecessário considerar os processadores no dispositivo móvel. A localização do dispositivo também não afetará a tarefa do escalonamento, bem como o tráfego na rede e a largura de banda que é igual para todos os utilizadores. Com base nestas considerações, previmos três parâmetros

de contexto: nível da bateria; relação sinal-interferência-ruído da rede (SIN) e QoS da aplicação.

4.1. Modelo proposto

Os nós da *fog*, com a nossa proposta ativada, é constituída por várias unidades de trabalho: *Unidade de recuperação de informações de contexto* - compreende uma arquitetura, conforme definido em La e Kim (2010). Ela recupera as informações de contexto (C_i) de cada pedido ($r \in R$). As informações de contexto recuperadas são encaminhadas para a unidade de priorização de tarefas sensíveis ao contexto; *Unidade de priorização de tarefas sensíveis ao contexto* - estima o valor da prioridade de contexto (P_r) para cada pedido individual $r \in R$ e o encaminha para a unidade de QoE e escalonamento sensível ao contexto. A *Unidade de QoE e escalonamento sensível ao contexto* - escalona as tarefas para serem executadas nas *MVs em servidores* de forma que a QoE do utilizador final seja otimizada.

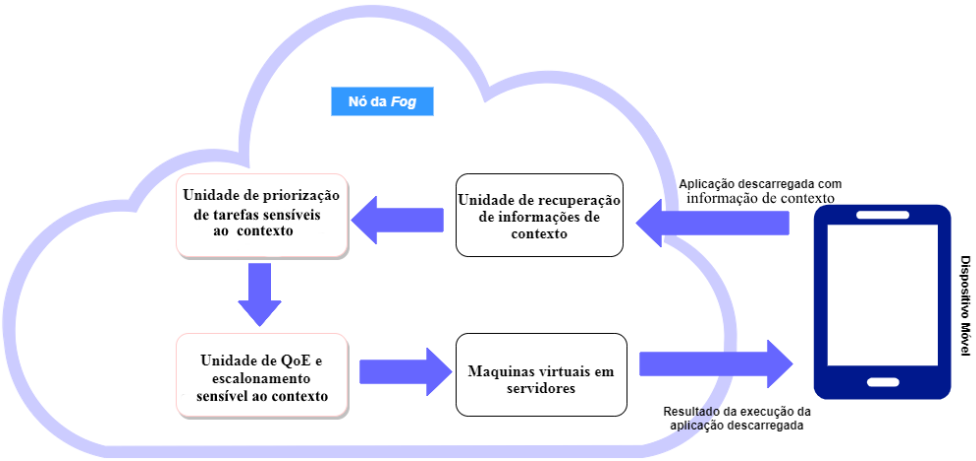

Figura 1 – Modelo proposto.

A figura 1, ilustra as diferentes unidades do modelo proposto. As notações relevantes utilizadas na *unidade priorização de tarefas sensíveis ao contexto* e na *unidade de QoE e escalonamento sensíveis ao contexto* estão listadas na tabela 1.

Símbolo	Definição
C	Conjunto de todos os parâmetros de contexto
α_i	Quantidade de diferentes tipos de níveis abstratos de um parâmetro de contexto, $C_i \in C$
γ_i	Diferença lógica entre dois níveis abstratos $C_i \in C$
η_i	Intervalo normalizado de valores $C_i \in C$
L_i	Conjunto de todos os níveis concretos de $C_i \in C$ no instante t
θ_i	Conjunto de valores tendenciosos $C_i \in C$

Símbolo	Definição
F	Conjunto de todas as combinações possíveis de um elemento de cada L_i
M	Conjunto múltiplo de valores mínimos $\forall \mu \in F$
R	Conjunto de execuções de aplicações em um dado instante t
V	Conjunto de máquinas virtuais disponíveis em um dado instante t
β	Conjunto dos coeficientes de regressão linear múltipla
P_r	Prioridade de qualquer pedido $r \in R$
$\psi_{r,v}$	Tempo necessário para a execução de um pedido $r \in R$ em uma máquina virtual $v \in V$
I_r	Quantidade de interrupções de uma requisição $r \in R$

Tabela 1 – Notações e definições da arquitetura do modelo proposto.

Assumimos que algumas MVs estejam criadas com diferentes configurações o que permite minimizar as sobrecargas relativas aos processos de criação e eliminação de MVs, conforme referido em Skarlat *et al.* (2017). A provisão ótima de MVs para os pedidos e as suas afetações energeticamente eficientes estão fora do escopo deste artigo. Foram, no entanto, discutidas em Li *et al.* (2017) e Yang *et al.* (2018).

4.2. Arquitetura do modelo proposto

Em Han (2011), foi proposto um resolvedor de heterogeneidade de contexto, que processa vários parâmetros de contexto, num intervalo normalizado, através da normalização *Min-Max*. Nas subsecções seguintes são definidas a arquitetura do modelo proposto, a começar pela unidade de priorização das tarefas sensíveis ao contexto.

4.2.1. Unidade de priorização de tarefas sensíveis ao contexto

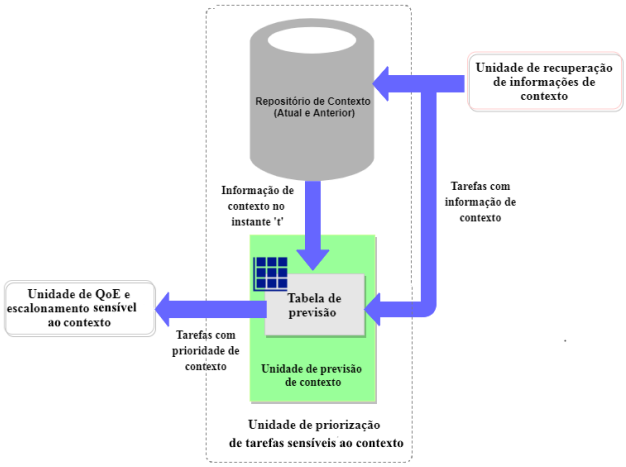

Figura 2 – Arquitetura da unidade de priorização de tarefas.

Esta unidade é composta pelo *repositório de contexto,* que armazena as informações de contexto das tarefas atuais e anteriormente recebidas e pela *unidade de previsão de contexto* que explora a informação de contexto num determinado instante de tempo e alimenta a *tabela de previsão*. Assim, conseguimos eliminar a heterogeneidade das informações de contexto na alimentação da *tabela de previsão*.

A *tabela de previsão*, disponibiliza um conjunto de dados para a análise da Regressão Linear Múltipla (RLM) que visa definir a prioridade de contexto dos pedidos atuais. A figura 2, ilustra arquitetura da *unidade de priorização de tarefas sensíveis ao contexto* do modelo proposto.

4.2.2. Criação da tabela de previsão de prioridade de contexto

Para a resolução de problemas relacionados com a heterogeneidade de informações do contexto, com base na normalização *Min-Max*, foi concebido um modelo que alimenta a tabela de previsão de prioridade de contexto. O repositório de contexto disponibiliza informações do contexto de tarefas previamente recebidas num determinado instante *t*. $\forall C_i \in C$, todas as informações do contexto são normalizadas em relação aos extremos, os valores ficam concentrados num intervalo entre 0 e 1. Esta abordagem permite minimizar a heterogeneidade dos diferentes parâmetros do contexto em termos de valores e unidades. Definimos o intervalo normalizado, η_i, para um parâmetro de contexto, $C_i \in C$, conforme a equação 1.

$$\eta_i = \frac{max(C_i) - min(C_i)}{max(C_i)} \tag{1}$$

Dentro deste intervalo normalizado de um $C_i \in C$, assumimos que existem α_i níveis abstratos. Estes níveis abstratos representam a medição qualitativa do parâmetro de contexto correspondente. Permitem que as variações de valores de um determinado parâmetro de contexto possam ser classificadas internamente. As diferenças lógicas entre dois níveis abstratos consecutivos, γ_i, de um contexto $C_i \in C$ são definidas conforme a equação 2.

$$\gamma_i = \frac{\eta_i}{\alpha_i} \tag{2}$$

A representação numérica desta abstração compreende um conjunto em concreto de níveis, L_i, para qualquer $C_i \in C$. Esta abordagem transforma a medição qualitativa em quantitativa, necessários para cálculos posteriores. O L_i é definido conforme a equação 3.

$$L_i = \left\{ x : x = \frac{min\ C_i}{max\ C_i} + z\gamma_i \right\}, \forall C_i \in C. \tag{3}$$

Onde $z = 0,1,2,\ldots,(\alpha_i - 1)$.

O cálculo do produto cartesiano, permitem definir o conjunto combinatório de todos os níveis de contexto a partir dos diferentes parâmetros de contexto. Este produto cartesiano é formulado conforme indicado na equação 4.

$$F = \prod_{i=1}^{|C|} L_i \cdot \tag{4}$$

Todas as combinações possíveis dos diferentes parâmetros do contexto de uma requisição ($r \in R$) são determinadas.

Também é criado um multiconjunto *M* com os valores mínimos de cada combinação de *F*, definido conforme a equação 5.

$$\mathrm{M} = \left\{ \mathrm{q} : \mathrm{q} = \min\left(\mu\right) \right\}, \forall \mu \in \mathrm{F}. \tag{5}$$

Este multiconjunto identifica o estrangulamento de todas as possíveis combinações do contexto de um pedido $r \in R$ que influência a priorização das combinações simbólicas. Além dos parâmetros de estrangulamento, os valores de enviesamento, associados aos diferentes níveis dos vários parâmetros de contexto, utilizados na priorização, também influenciam a priorização. Este valor de enviesamento permite enfatizar os outros parâmetros. Basicamente, é um mapeamento *um-para-um* entre os elementos de L_i e o conjunto enviesado, θ_i para um determinado parâmetro de contexto, $C_i \in C$.

Seja, $\theta_{i,j}$ refere ao valor mínimo de enviesamento de C_i , no seu *j-ésimo* nível associado a uma combinação candidata em *F*. Para mapear, θ_i para L_i, $\theta_{i,o}$, presume-se, $\forall C_i \in C$. $\forall C_i \in C$, $\theta_{i,j}$ deve satisfazer a seguinte condição:

$$\theta_{i,j} * max(q \in M) < \theta_{i,j+1} * min(q \in M) \tag{6}$$

As prioridades destas combinações são definidas de forma que a informação do contexto de qualquer pedido possa ser mapeada sobre ela mesma a fim de prever a prioridade desse pedido.

O cálculo da prioridade é feito utilizando os valores relevantes de enviesamento $\forall C_i \in C$ e o seu parâmetro de contexto de estrangulamento. Assumimos que $\delta_i\left(\mu_k\right)$ define a prioridade do contexto $\mu_k \in F$ enviesado em $C_i \in C$. $\delta_i\left(\mu_k\right)$ é representado conforme a equação 7.

$$\delta_i(\mu_k) = \frac{\theta_{ij} * q_k}{\sum q}, \forall q \in M. \tag{7}$$

Onde, $0 \leq k \leq \left(|F|-1\right)$ e $q_k \in M$ está associada à μ_k.

A prioridade estimada, $\hat{\delta}_i(\mu_k)$ de μ_k é calculada através da equação 8.

$$\delta_i(\mu_k) = \sum_{C_i \in C} \delta_i(\mu_k). \tag{8}$$

μ_k e $\hat{\delta}(\mu_k)$, $\forall \mu_k \in F$ alimentam a tabela de previsão com um conjunto de dados. Estes dados da tabela de previsão permitem prever a prioridade de qualquer pedido em fila de espera.

4.2.3. Previsão da prioridade do contexto

A RLM é um dos métodos estatísticos multivariados cuja a principal preocupação consiste em estabelecer as relações entre várias variáveis independentes ou preditoras e uma variável dependente ou critério. Ao identificar como estas múltiplas variáveis independentes se relacionam com a variável dependente, as informações sobre as variáveis independentes podem ser utilizadas para fazer previsões precisas e poderosas (Quinn & Keough, 2002). Consideremos, *m* observações de um conjunto de *p* número do variável preditor *Xs* e um variável critério *Y* associado a elas. O modelo de RLM, que geralmente se ajusta a este cenário, é definido conforme a equação 9.

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_j x_{ij} + \cdots + \beta_p x_{ip} \in_{i,} \tag{9}$$

onde, para a *i-ésima* observação, $y_i = Y$ e $x_{ji} = X_j$, $\forall X_j \in X$. O coeficiente do RLM, β_0, é a interceção da população e β_j, é a variação de *Y* em uma unidade de variação em X_j, $\forall X_j \in X \, e \, 1 \leq j \leq p,$ mantendo as outras variáveis independentes constantes, $\in_i$ é o erro aleatório ou inexplicável associado à *i-ésima* observação. Os valores estimados dos coeficientes da RLM são calculados através dos valores conhecidos da equação (8) de cada observação, e resolvidos algebricamente. Estimando $\beta = \{\beta_0, \beta_1, ..., \beta_p\}$, $\forall \beta_0 \in \beta$ e σ_ϵ^2 (erro da variância), a reta de regressão ajustada, que prevê Y para qualquer observação desconhecida, é expressa conforme a equação 10:

$$\hat{\boldsymbol{y}}_i = \boldsymbol{b}_0 + \boldsymbol{b}_1 \boldsymbol{x}_{i1} + \ldots + \boldsymbol{b}\boldsymbol{x}_{ij} + \ldots + \boldsymbol{b}_p \boldsymbol{x}_{ip} \in_i \tag{10}$$

onde, $\hat{y}_i$ é o valor previsto para qualquer observação desconhecida, b_j é a estimativa da amostra de $\beta_j \forall \beta_j \in \beta$.

Ao invés de calcular as regressões para cada variável preditora individualmente, a RLM utiliza informações de todas as variáveis independentes simultaneamente para prever uma única variável critério. Como resultado, ela é naturalmente mais rápida do que os outros métodos de análise multivariada (Mertler & Reinhart, 2016).

Fazemos o mapeamento da tabela de previsão de contexto para o modelo RLM, considerando cada *tuplo* da tabela de previsão como uma observação de um conjunto de dados da RLM em que, o conjunto de variáveis independentes, $X = C_i, \forall C_i \in C$ de cada combinação $\mu \in F$, a variável dependente, $Y = \hat{\delta}_i(\mu_k)$ é a prioridade prevista de qualquer requisição desconhecida, $P_r = \hat{y}_i$.

Para qualquer pedido $r \in R$, quanto menor for o valor P_r, maior será a prioridade.

A tabela de previsão de prioridade de contexto é criada através dos valores quantitativas $\forall C_i \in C$ e dos parâmetros de contexto que são independentes entre si. Após a definição da prioridade do contexto, as informações do contexto desta tarefa são armazenadas no repositório de contexto.

4.3. Otimização do escalonamento das aplicações

A *unidade de QoE e escalonamento sensível ao contexto* do sistema proposto escalona as tarefas prioritárias para serem executadas em MVs na *fog* após um intervalo de escalonamento (IE), τ.

De modo a otimizar a QoE dos utilizadores, o escalonador proposto explora a prioridade de contexto (P_r) de um pedido $r \in R$ e a sua duração estimada do tempo de execução ($T_{r,v}$) para definir o escalonamento desse pedido $r \in R$ em uma MV, $v \in V$. Se em virtude da quantidade limitada de MVs e à baixa prioridade, um pedido $r \in R$ não poder ser escalonada num intervalo de escalonamento, ela deverá ser escalonada nos próximos intervalos. Na nossa proposta, a execução de tarefa com menores prioridades é interrompida em virtude da chegada de tarefa com maiores prioridades. Considerando que essa preempção pode propiciar a espera por tempo indeterminado para a execução de um determinado pedido $r \in R$, exploramos também o número de intervalos de escalonamento (I_r), em que um pedido de escalonamento numa MV é adiado desde a sua chegada, com o objetivo de evitar a condição de *starvation*. Com vista a otimizar a QoE dos utilizadores foi utilizada a técnica de otimização de Programação Não Linear Multiobjetivo (MONLP) para escalonar os pedidos.

Segundo Miettinen (1998), a técnica MONLP é geralmente aplicada aos problemas de otimização onde existem mais de uma função objetivo (FO) não lineares a serem otimizada simultaneamente. Ela possui alguns elementos como: conjunto de funções

objetivos não lineares que devem ser otimizadas; variáveis de decisão que constituem o domínio no qual cada FO deve operar; restrições que delimitam o espaço de pesquisa.

Nas secções 4.3.1 e 4.3.2, são definidas as funções objetivos e as restrições para o escalonamento otimizado de aplicações.

4.3.1. Definição da função objetivo

Um dos objetivos desta artigo consiste em escalonar pedidos, $r \in R$ em MV, $v \in V$, visando otimizar a QoE dos utilizadores para todos os pedidos num determinado intervalo de escalonamento. Assumindo que a execução de todos os pedidos, $r \in R$ não são preemptivas e que todas as MVs, $v \in V$ estão criadas na *fog*, a FO é definida conforme a equação 11.

$$min_{r,v} \sum_{r \in R} \sum_{v \in V} \frac{P_r * \psi_{r,v}}{I_r} \tag{11}$$

E está sujeita a algumas restrições, (inequações (12) à (17)).

Esta equação indica que a QoE de todos os pedidos de aplicações dos utilizadores podem ser otimizadas através da minimização da soma dos seus tempos de execução. Ela também leva em consideração a execução prioritária das tarefas com maiores prioridades através da minimização da soma das prioridades de todas os pedidos, dado que, quanto menor for o resultado, maior será a prioridade obtida. Além disso, a soma dos valores inversos do (I_r), $\forall r \in R$ mostra que os pedidos, em que foram adiados os seus escalonamentos num determinado intervalo, terão maior prioridade para serem escalonadas nos atuais intervalos, atenuando assim a situação de *starvation*.

4.3.2. Definição das restrições

Para o escalonamento ótimo definimos as seguintes restrições:

- *Restrição de capacidade:* é apresentada conforme a inequação 12.

$$\sum_{v \in V} \eta_v \leq \eta \tag{12}$$

Onde, η_v representa o tamanho da máquina virtual e η é a capacidade total do nó da *fog*.

- *Restrição de afetação de MV:* é apresentada conforme a inequação 13.

$$\sum_{v \in V} K_{v,r} \leq 1, \forall v \in V, \forall r \in R \tag{13}$$

$K_{v,r}$ é uma variável booleana, que é igual a um, se uma MV $v \in V$ for afetada a um pedido $r \in R$; caso contrário é zero.

- *Restrição de escalonamento de pedidos:* é escrita conforme a inequação 14.

$$\sum_{r \in R} \chi_{r,v} \leq \forall r \in R, \forall v \in V. \tag{14}$$

$\chi_{r,v}$ é uma variável booleana, é igual a um, se um pedido $r \in R$ estiver escalonada em uma MV $v \in V$; caso contrário, é zero.

- *Restrição de QoS:* é representada conforme a inequação 15.

$$\psi_{r,v} + Q_r \leq T_r, \forall r \in R. \tag{15}$$

onde, $\psi_{r,v}$ representa o tempo necessário para executar um pedido $r \in R$ em uma máquina virtual $v \in V$, Q_r é o tempo de espera na fila do pedido *r*. e T_r corresponde à QoS da aplicação.

- *Restrição de consumo de energia:* é apresentada conforme a inequação 16.

$$\boldsymbol{B_r \geq B_{th}}. \tag{16}$$

onde, B_r representa o nível de bateria do dispositivo do utilizador final associado a um pedido, $r \in R$ e B_{th} indica o nível mínimo de bateria, para que o dispositivo requisitante se permaneça ligado até ao final da execução.

Restrição da qualidade do sinal: é escrita conforme a inequação 17.

$$\boldsymbol{SIN_r \geq SIN_{th}}, \tag{17}$$

onde, SIN_r representa a intensidade do sinal associado a um pedido $r \in R$ e SIN_{th} indica a intensidade mínima do sinal necessário para a submissão de um pedido do dispositivo do utilizador final.

4.4. Definição dos parâmetros

Comparado com os outros parâmetros de contextos, o requisito de QoS da aplicação tem maior impacto no aperfeiçoamento da QoE dos utilizadores finais. Por isso, calculamos os valores de enviesamento, θ associados principalmente a este parâmetro.

O desempenho da solução proposta é influenciado grandemente pelo intervalo de escalonamento, τ. Atendendo que o tempo médio de execução de diferentes pedidos

$r \in R_t$ nas diferentes MVs $v \in V_t$ são conhecidos a partir das experiências anteriores. Podemos calcular a previsão do escalonamento estático (τ_s) conforme definido na equação 18.

$$\tau_s = \frac{1}{|R_t|} \sum_{r \in R_t} \sum_{v \in V_t} \psi_{r,v} \,. \tag{18}$$

O valor τ estático nem sempre permite uma boa utilização dos recursos. Um valor de τ alto reduz a sobrecarga do escalonamento, contudo, poderá não conseguir manter a exigência de QoS da aplicação do utilizador nem garantir uma melhor utilização dos recursos computacionais. Do mesmo modo, um valor de τ muito baixo pode provocar uma grande sobrecarga no escalonamento, assim como poderá possibilitar a existência de alguns intervalos de escalonamento, sem a chegada de pedidos. Portanto, um valor τ dinâmico permite aumentar o desempenho da nossa proposta, assumindo que a taxa média de chegada dos pedidos varia muito ao longo do tempo. No entanto, na prática, nem a sequência da chegada dos pedidos, nem os respetivos tempos de execução nas MVs obedecem o intervalo de escalonamento estático (τ_s), que é calculada como média aritmética dos valores das experiências anteriores. Por isso, visando aumentar o desempenho do modelo proposto optamos, pelo cálculo do intervalo de escalonamento dinâmico (τ_d), que utiliza a fórmula do *Exponentially Weighted Moving Average* (EWMA) (Wold, 1994), conforme a equação 19:

$$\hat{\tau}_{d(m)} = (1-\omega)\hat{\tau}_{d(m-1)} + \omega\tau_{d(m)} \tag{19}$$

onde, ω é a constante de ponderação, ou o fator de alisamento, onde $0 < \omega < 1$ e $\tau_{d(m)}$, é a média aritmética dos tempos de execução dos pedidos em todas as MVs no intervalo de escalonamento *m-éssimo*. Conforme a equação 20.

$$\tau_{d(m)} = \frac{1}{|S|} \sum_{s \in S} \frac{1}{|V_s|} \left(\sum_{v \in S} \frac{N_{v,r}}{\phi_v} \right).$$

onde, S e V_s significam respetivamente o conjunto de servidores e de MVs em cada servidor. $N_{v,r}$ representa o número de instruções de um pedido $r \in R$ executada numa MV $v \in V_s$ com velocidade de processamento ϕ_v.

Na secção seguinte, é feita a avaliação do desempenho do modelo proposto.

5. Avaliação de desempenho do modelo proposto

O ambiente de simulação do modelo proposto foi desenvolvido no *kit* de ferramentas de simulação *iFogSim* e modela uma *fog* composta por três *hosts* com parâmetros conforme a tabela 2.

Parâmetros	Valores
Quantidade de MV	1 ~ 15
Capacidade do disco de cada MV	10,000 MB
Memória de cada MV	512 MB
QoS da Aplicação	6 ~ 15 s
Tamanho do pedido	1000 ~ 2000 MI
B_{th}	15%
SIN_{th}	15 dB
$\alpha_{Bateria}$	3
α_{sin}	2
α_{QoS}	4
Duração da Simulação	500 s

Tabela 2 – Parâmetros de simulação

5.1. Métricas de desempenho da análise do modelo proposto

As seguintes métricas de desempenho foram utilizadas para comparar a nossa proposta de escalonamento (com Intervalo de Escalonamento (IE) estático e dinâmico) com abordagens de escalonamento não sensíveis ao contexto como: *FCFS, SJF* e *QoS-based*:

- *Percentagem de execução dos pedidos bem-sucedidos:* é calculada através da razão entre a quantidade de tarefas que preservam os diferentes parâmetros de contexto e a totalidade de tarefas solicitadas. Quanto maior for esta percentagem, maior será a quantidade de tarefas que preservam os diferentes parâmetros de contexto.
- *Tempo médio de espera de uma tarefa:* é o tempo decorrido desde a sua chegada até a sua disponibilização numa MV. Q_r indica o tempo de espera de um pedido $r \in R$. O tempo médio de espera de uma tarefa é calculado conforme a equação 21.

$$\hat{Q} = \frac{1}{|R|}\sum_{r \in R}(Q_r) \tag{21}$$

onde *R* representa a totalidade dos pedidos recebidos durante o período de simulação. Quanto menor for o valor de $\hat{Q}$ maior será o desempenho.

- *Qualidade da Experiência (QoE):* é o grau de satisfação global dos utilizadores relativamente à utilização de um produto ou serviço. Pode ser aperfeiçoada através da diferença entre o tempo máximo permitido para a obtenção da resposta (requisito QoS da aplicação), T_r e o tempo de resposta de um pedido.

Calculamos a média de QoE de todos os pedidos R expedidos durante a simulação conforme a equação 22.

$$QoE = \frac{1}{|R|} \sum_{r \in R} \sum_{v \in V} \left(T_r - \left(\psi_{r,v} + Q_r \right) \right) \tag{22}$$

Quanto maior for este valor, melhor será a capacidade do sistema em otimizar a QoE do utilizador.

Os resultados da avaliação, obtidos através da simulação implementada com base nos parâmetros da tabela 2, são descritos na subsecção 5.2.

5.2. Resultados e discussões

Uma análise detalhada ao ficheiro da simulação permite-nos concluir que a taxa de sucesso no escalonamento das aplicações sensíveis ao contexto é superior em comparação com os escalonamentos não sensíveis ao contexto. No modelo proposto, os pedidos dos dispositivos em situações de vulnerabilidades, com baixos valores de contextos, possuem maior precedência de execução. Isto é, é-lhes assegurado o funcionamento prioritário por forma a poderem receber o *feedback* da *fog*. No entanto, como a *fog* possui recursos limitados, o que afeta a performance de serviço, a taxa de sucesso diminui quando aumentam a quantidade dos pedidos.

Seguidamente, apresentamos e discutimos a variação dos pedidos executados com sucesso em relação ao aumento de pedidos, MVs e requisitos QoS da aplicação.

5.2.1. Impactos do aumento de pedidos

As percentagens dos pedidos executados com sucesso preservando os diferentes parâmetros de contexto (mantendo constante a quantidade de MVs), são apresentadas nos gráficos 1, 2 e 3. O tempo médio de espera e a QoE neste cenário são apresentadas nos gráficos 4 e 5, respetivamente.

Na nossa proposta, os pedidos provenientes de dispositivos com nível de bateria baixo têm maior prioridade. São executadas com maior precedência. Como resultado, os dispositivos obtêm as respostas aos pedidos antes do dispositivo se desligar devido a insuficiência da bateria. Consequentemente, o sistema proposto possui um melhor desempenho em comparação com as outras abordagens em estudo, conforme ilustrado no gráfico 1. Isto deve-se ao facto de que nenhum dos outros algoritmos estudados se preocupa com o contexto do nível da bateria. Além disso, o sistema proposto (com IE dinâmico) diminui o tempo de inatividade dos recursos da MV nos servidores o que permite aumentar as hipóteses para os pedidos com contextos críticos serem executadas com maior prioridade.

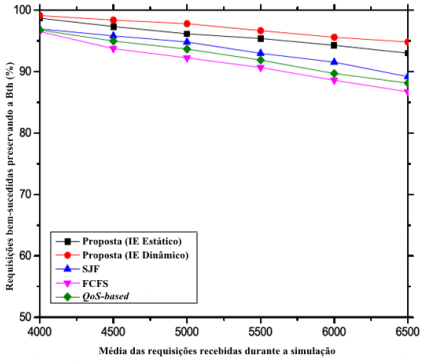

Gráfico 1 – Tarefas executadas com sucesso preservando o nível da bateria em relação ao aumento de pedidos.

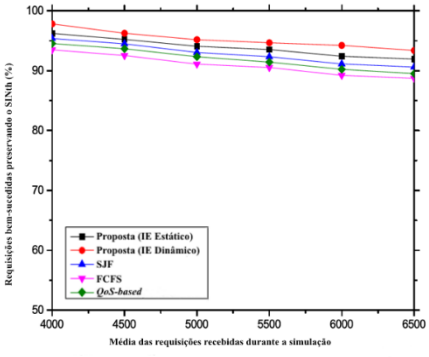

Gráfico 2 – Tarefas executadas com sucesso preservando o nível do sinal em relação ao aumento de pedidos.

Da mesma forma, podemos observar através do gráfico 2 que em todas as abordagens estudadas, a percentagem de pedidos de aplicações executados com sucesso preservando o nível do sinal, também diminui gradualmente devido ao aumento de receção dos pedidos. Como teoricamente esperado, a quantidade de pedidos com valores críticos do SIN aumentam em função do aumento da quantidade de pedidos e, como consequência, o desempenho total diminui ao longo do tempo.

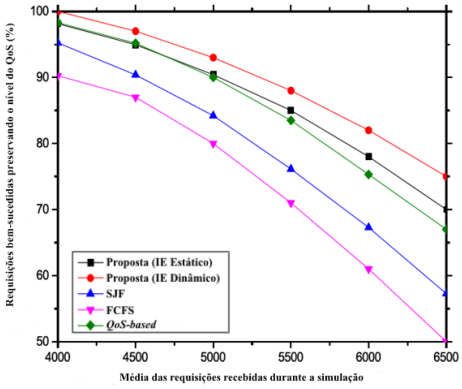

Gráfico 3 – Tarefas executadas com sucesso preservando o nível da QoS em relação ao aumento da quantidade de pedidos.

O gráfico 3 ilustra que a percentagem dos pedidos executados com sucesso, preservando as exigências da QoS, diminui drasticamente em todas as abordagens de escalonamento estudadas em função do aumento da quantidade de pedidos. A razão subjacente a este resultado é explicada pelo facto de que o aumento dos pedidos também permite aumentar o tempo de espera, o que obriga muitas tarefas a violarem os requisitos de QoS. No entanto, o desempenho da nossa proposta supera o dos outros algoritmos (*FCFS, SJF, QoS-based*).

No modelo proposto, o impacto dessa diminuição é significativamente menor, dado que ela privilegia o escalonamento de acordo com os seus requisitos de QoS. Por fim, o resultado demonstra que, para todos os casos, o intervalo de escalonamento dinâmico produz um impacto ainda melhor de desempenho em comparação com o seu equivalente estático.

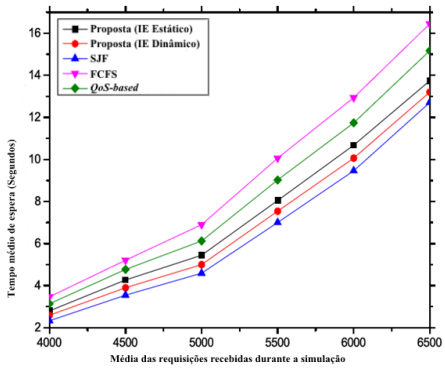

Gráfico 4 – Tempo médio de espera em relação ao aumento da quantidade de pedidos.

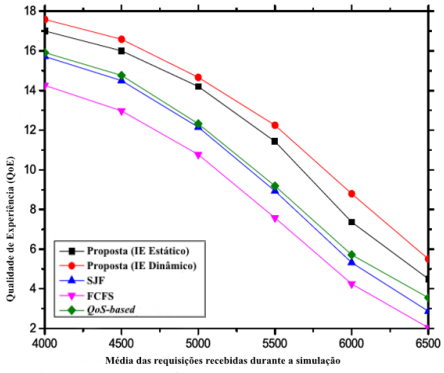

Gráfico 5 – QoE dos utilizadores em relação ao aumento da quantidade de pedidos.

O gráfico 4 ilustra que o tempo médio de espera no modelo proposto é inferior em comparação com a maioria das outras técnicas não sensíveis ao contexto.

Isto deve-se ao facto de o escalonamento no modelo proposto não considerar apenas o contexto das tarefas, mas também as interrupções dos mesmos enquanto estão na fila de espera para serem escalonadas e ao seu tamanho individual. Por conseguinte, uma tarefa deve ser penalizada por um período curto devido a outros fatores e uma tarefa trivial não deve esperar muito tempo para ser executada, devido à execução de uma tarefa maior.

O gráfico 5 ilustra a otimização da QoE dos utilizadores finais.

O sistema proposto privilegia a execução de tarefas com valores de contexto mais baixos. Como resultado elas são executadas respeitando um tempo mínimo e atraso de resposta tolerável. Relativamente às outras tarefas com parâmetros de contexto adequados, este cenário é suportado de forma intrínseca. Assim, a QoE dos utilizadores é otimizada em todos os cenários possíveis. Em comparação com a maioria das outras técnicas não sensíveis ao contexto, elas muitas vezes não conseguem otimizar a QoE, porque de acordo com os seus critérios de escalonamento, enquanto satisfazem um pedido com maior prioridade, o pedido com menor prioridade pode não conseguir manter a sua tolerância de QoS. Nos piores cenários, poderá resultar em valores de QoE negativos, enquanto que, no modelo proposto, o valor de QoE é sempre positivo.

5.2.2. Impactos do aumento de MVs na fog

As percentagens das tarefas executadas com sucesso preservando os diferentes parâmetros de contexto, assumindo que a quantidade de pedidos recebidos durante a simulação é constante em relação ao aumento da quantidade de MVs, são ilustrados nos

gráficos 6, 7 e 8. Os gráficos 9 e 10 ilustram respetivamente o tempo médio de espera e QoE dos utilizadores no critério impactos do aumento de MVs.

Se o número de MVs aumentar, também aumenta a taxa de sucesso na execução dos pedidos que satisfazem os vários condicionalismos contextuais. Este cenário é apresentado nos gráficos 6 e 7 tanto em termos de níveis da bateria como em relação à força do sinal da rede.

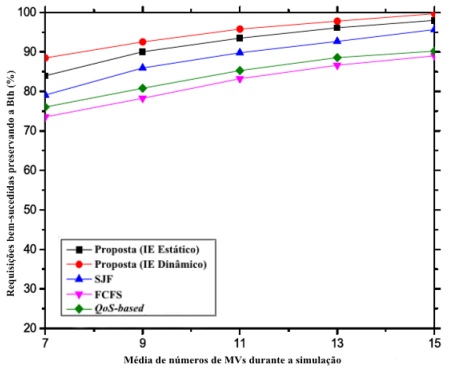

Gráfico 6 – Pedidos executados preservando o nível de bateria em relação ao aumento de MV.

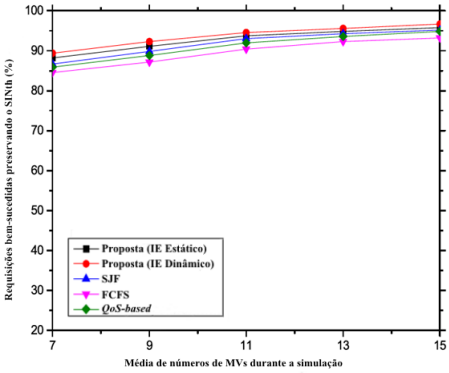

Gráfico 7 – Pedidos executados preservando o nível do SIN em relação ao aumento de MV.

O gráfico 8 ilustra que a percentagem dos pedidos executados com sucesso preservando o nível de QoS em relação ao aumento das máquinas virtuais. Excetuando o comportamento normal para valores mais altos no eixo X, a taxa de sucesso é consideravelmente mais alta no sistema proposto em comparação com os outros algoritmos, o mesmo também se verifica em relação aos valores mais baixos do eixo X.

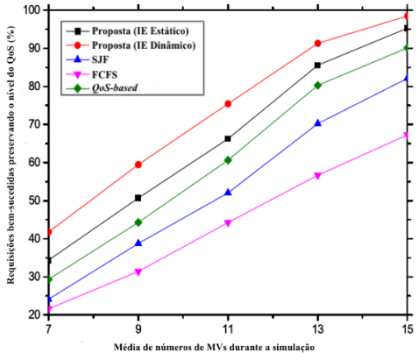

Gráfico 8: Pedidos executados preservando a QoS em relação ao aumento de MV.

Quanto ao tempo de espera e QoE em relação ao aumento de MV, conforme os gráficos 9 e 10, apresentam cenários teoricamente comprovados. Mesmo assim, o desempenho do modelo proposto é sempre superior em comparação com as outras políticas.

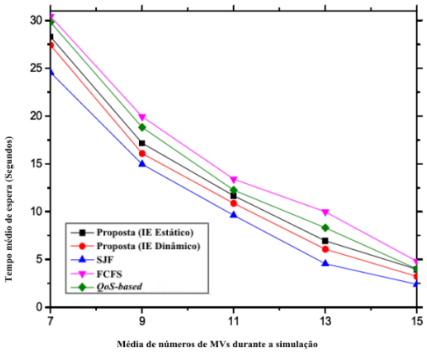

Gráfico 9 – Tempo médio de espera em relação ao aumento de MV.

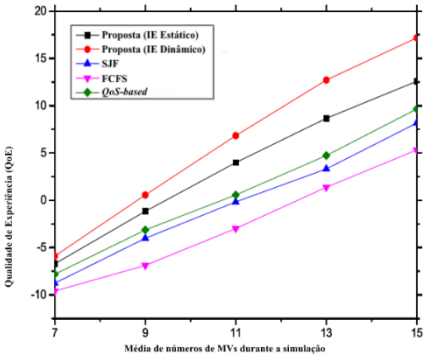

Gráfico 10 – QoE do utilizador em relação ao aumento de MV.

5.2.3. Impactos do requisito QoS da aplicação

Nos gráficos 11 e 12 é ilustrado o impacto do requisito QoS dos pedidos no desempenho do modelo proposto. Ilustram que à medida que aumenta o tempo médio admissível de resposta aos pedidos, aumentam também as percentagens das tarefas satisfeitas em termos de QoS e QoE. Isto acontece porque o tempo de execução dos pedidos diminui. Consequentemente, é possível executar tarefas de acordo com a sua exigência de QoS e dentro do tempo mínimo estabelecido. Este facto permite o aperfeiçoamento da QoE dos utilizadores. As outras técnicas de escalonamento não sensíveis ao contexto, em comparação com o modelo proposto, revelam menos eficiência neste cenário.

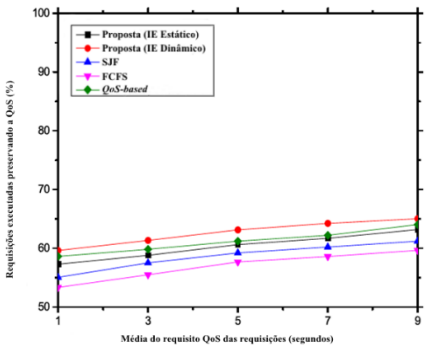

Gráfico 11 – Pedidos executados em relação ao aumento da QoS da aplicação.

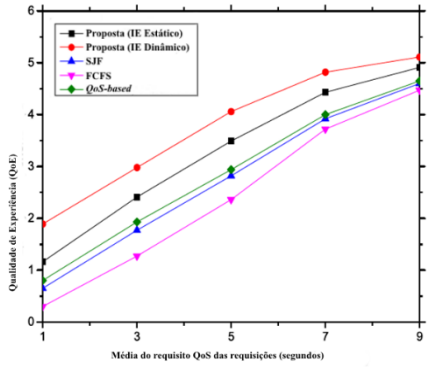

Gráfico 12 – QoE do utilizador em relação ao aumento da QoS da aplicação.

5.3. Resumo das simulações

Em síntese a esta experiência, afirmamos sem dúvida que a política de escalonamento sensível ao contexto proposta, é eficiente em relação à quantidade de tarefas escalonadas e que satisfazem a QoS, diminuição do tempo de resposta, otimização de

QoE, redução do tempo médio de espera, entre outros, em comparação com as políticas de escalonamentos não sensíveis ao contexto. Tendo em consideração as observações fundamentadas apresentadas, podemos afirmar que a nossa proposta de escalonamento possui condições para ser implementada em ambientes práticos.

6. Conclusões

O objetivo principal deste artigo consiste em definir um modelo de escalonamento de tarefas sensíveis ao contexto para o paradigma *fog*. Como apresentado, o modelo proposto é suficientemente eficiente para priorizar tarefas independentemente da heterogeneidade das suas informações de contextos. Ademais, consegue otimizar a QoE dos utilizadores e permite situações vantajosas em termos de utilização de recursos e tempo de execução.

As principais conclusões alcançadas com base no levantamento de trabalhos relacionados tanto na arquitetura *cloud* como no paradigma *fog* foram que o escalonamento na *cloud*, foi amplamente estudado. Enquanto que na *fog*, devido densidade e heterogeneidade de dispositivos, o escalonamento é complexo e ainda existem poucos estudos. Muitos dos algoritmos estudados possuem várias limitações: não descrevem a forma como a prioridade é definida; não explicam o método utilizado na priorização de tarefas e nem definem a priorização de tarefas com base em informações do contexto; muitos defendem a perspetiva dos provedores de serviços; outros são aplicados em tarefas agrupadas para diminuir o tempo de execução; alguns otimizam apenas a QoS. Outros, exploram alguns contextos. Identificamos e sugerimos algumas perspetivas de melhorias podem ser explorados e melhorados como: a utilização do contexto no escalonamento; priorização de tarefas sensíveis ao contexto; consideração da restrição energética no escalonamento; preservação da força do sinal da rede; preservação da QoS; redução do tempo médio de espera e otimização da QoE.

Com base nessas sugestões de melhorias propomos um modelo de escalonamento de tarefas sensíveis ao contexto para o paradigma *fog* que utiliza a normalização *Min-Max*, para resolver o problema da heterogeneidade dos diferentes parâmetros de contexto, o método da RLM para a definição das prioridades dos pedidos e a técnica MONLP para o escalonamento ótimo visando otimizar a QoE dos utilizadores. A definição dos parâmetros do sistema permitiu um escalonamento mais eficiente, que garante a boa utilização dos recursos, bem como a execução eficaz das tarefas.

A validação do modelo e da arquitetura proposta, foi feita com base em simulações realizadas no kit de ferramentas *iFogSim*, que nos possibilitou fazer comparações experimentais e teóricas da nossa proposta sensível ao contexto, tanto para intervalos de escalonamento estáticos como dinâmicos com escalonadores não sensíveis ao contexto (*FCFS, SJF, QoS-based*) com base nas seguintes métricas: taxa de sucesso dos pedidos; tempo médio de espera e QoE dos utilizadores.

Apesar de considerarmos vários parâmetros de contextos na nossa proposta, acreditamos que outros ainda podem ser escolhidos por forma a influenciarem ainda mais significativamente o escalonamento. Concluímos afirmando que todos os objetivos propostos foram alcançados.

Agradecimentos

Os autores agradecem à Fundação Calouste Gulbenkian pelo financiamento desta investigação através da bolsa de doutoramento sob a referência n.º 234242, 2019-Bolsas de Pós-Graduação para estudantes dos PALOP e de Timor-Leste.

Referências

Aazam, M., Hilaire, M. St., Lung, Ch, & Lambadaris, I. (2016). MeFoRE: Resource Estimation QoE based at Fog to Enhance QoS in IoT. In: Proc. of the 23rd International Conference on Telecommunications, ICT '16, IEEE, pp. 1-5. https://doi.org/10.1109/ICT.2016.7500362.

Bahl, P., Cuervo, E., Balasubramanian, A., Cho, D. K., Wolman, A., Saroiu, S., & Chandra, R. (2010). Maui: making smartphones last longer with code offload. In Proceedings of the 8th international conference on Mobile systems, applications, and services. pp. 49-62. https://doi.org/10.1145/1814433.1814441.

Berg, F., Durr, F., & Rothermel, K. (2014). Increasing the efficiency and responsiveness of mobile applications with preemptable code offloading., IEEE International Conference on Mobile Services (MS), IEEE, pp. 76–83, https://doi.org/10.1109/MobServ.2014.20.

Cardellini, V., Grass,i V., Presti, F.L., & Nardelli, M. (2015). On QoS-Aware Scheduling of Data Stream Applications over Fog Computing Infrastructures, IEEE Symposium on Computers and Communication (ISCC), pp. 271-276. https://doi.org/10.1109/ISCC.2015.7405527.

Barros, C., Rocio, V., Sousa, A., & Paredes, H. (2020). Survey on Job Scheduling in Cloud-Fog Architecture. 2020 15th Iberian Conference on Information Systems and Technologies (CISTI), Sevilla, Spain, 2020, pp. 1-7. https://doi.org/10.23919/CISTI49556.2020.9141156.

Bazire, M., & Brézillon, P. (2005). Understanding Context Before Using It. In A. Dey, B. Kokinov, D. Leake, & R. Turner (Eds.), Modeling and Using Context (pp. 29–40). Springer Berlin Heidelberg. https://doi.org/10.1007/11508373_3.

Bonomi, F., Milito, R., Natarajan, P., & Zhu, J. (2014). Fog Computing: A Platform for Internet of Things and Analytics. In N. Bessis & C. Dobre (Eds.), Big Data and Internet of Things: A Roadmap for Smart Environments (pp. 169–186). Springer International Publishing. https://doi.org/10.1007/978-3-319-05029-4_7.

Deng, R., Luan, T. H, Lu, R., Liang, H., & Lai, C. (2016). Optimal Allocation Workload in Fog-Cloud Computing Towards Balanced Delay and Power Consumption, IEEE Internet Things J. X(X), 1171-1181. https://doi.org/10.1109/JIOT.2016.2565516.

Dey, A. K. (2001). Understanding and Using Context. Personal and Ubiquitous Computing, 5(1), 4–7. https://doi.org/10.1007/s007790170019.

Fan, J., Wei, X., Wang, T., Lan, T., & Subramaniam, S. (2017). Deadline-Aware Task Scheduling in a Tiered IoT Infrastructure. In GLOBECOM 2017 - 2017 IEEE Global Communications Conference (pp. 1–7). https://doi.org/10.1109/GLOCOM.2017.8255037.

Fernando N., Loke S. W., & Rahayu, W. (2013). Mobile cloud computing: The survey, Future Generation Computer Systems, 29(1), 84-106. https://doi.org/10.1016/j.future.2012.05.023.

Ghouma, H., & Jaseemuddin, M. (2015). Context aware resource allocation and scheduling for mobile cloud, 2015 IEEE 4th International Conference on Cloud Networking (CloudNet), Niagara Falls, ON, (pp. 67-70). https://doi.org/10.1109/CloudNet.2015.7335282.

Gill, S. S., Garraghan, P., & Buyya, R. (2019). ROUTER: Fog enabled cloud based intelligent resource management approach for smart home IoT devices. Journal of Systems and Software, 154, 125-138. https://doi.org/10.1016/j.jss.2019.04.058.

Gordon, M., Jamshidi, D., Mahlke, S., Mao, Z., & Chen, X. (2012). COMET: code offload by migrating execution transparently, Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation, (pp. 93-106). https://dl.acm.org/doi/10.5555/2387880.2387890.

Han, J. (2011). Data Mining: Concepts and Techniques (3th Ed.). Morgan Kaufmann Publishers Inc..

Intharawijitr, K., Iida, K., & Koga, H. (2016). Analysis of Fog Model considering Computing and Communication Latency in 5G Cellular Networks, IEEE International Conference on Pervasive Computing and Communication Workshops (Workshops Percom), (pp. 1-4). https://doi.org/10.1109/PERCOMW.2016.7457059.

La, H. J., Kim, S. D. (2010). A conceptual framework for provisioning context-aware mobile cloud services, in Cloud Computing (CLOUD, IEEE 3rd International Conference on. IEEE, (pp. 466-473). https://doi.org/10.1109/CLOUD.2010.78.

Lawanyashri, M., Balusamy, B., & Subha, S. (2017). Energy-Aware fruitfly hybrid optimization for load balancing in cloud environments is EHR applications, Informatics Med. Unlocked, 8, 42-50. https://doi.org/10.1016/j.imu.2017.02.005.

Li, Q., Novak, E., Yi, S., & Hao, Z. (2017). Challenges and Software Architecture for Fog Computing, in IEEE Internet Computing, 21(2), 44-53. https://doi.org/10.1109/MIC.2017.26.

Li, T., Liu, Y., Gao, A. L. & Liu, A. (2017). A for Cooperative - based Smart Sensing Tasks in Fog-Computing in IEEE, Access, vol. 5, pp. 21296-21311, https://doi.org/10.1109/ACCESS.2017.2756826.

Mahmud, M. R., Afrin, M., Razzaque, M. A., Hassan, M. M., Alelaiwi, A., & Alrubaian, M. (2016). Maximizing Quality of Experience through Context-Aware Mobile Application Scheduling in Cloudlet Infrastructure. Software: Practice and Experience 46 (11), 1525-1545. https://doi.org/10.1002/spe.2392.

Mertler, C., & Reinhart, V. (2016). Advanced and Multivariate Statistical Methods (6th Editio). Routledge. https://doi.org/10.4324/9781315266978.

Miettinen, K. (1998). Nonlinear Multiobjective Optimization (1998th Edition). Springer editors.

Musumba, G.W., & Nyongesa, H. O. (2013). Context awareness in mobile computing: a review. International Journal of Machine Learning and Applications 2(1): 1-5. https://doi.org/10.4102/ijmla.v2i1.5.

OpenFog. (2017). OpenFog Reference Architecture for Fog Computing, OpenFog Consortium Architecture Working Group, Available at: https://www.iiconsortium.org/pdf/OpenFog_Reference_Architecture_2_09_17.pdf. (Accessed 9 march 2021).

Oueis, J., Strinati, E. C., & Barbarossa, S. (2015). The Fog Balancing: Load Cell Distribution for Small Cloud Computing, IEEE 81st Vehicular Technology Conference (VTC Spring), Glasgow, 2015, (pp. 1-6). https://doi.org/10.1109/VTCSpring.2015.7146129.

Qiu, M., Chen, Z., Yang, L. T., Qin, X., & Wang, B. (2012). Towards Power-Efficient Smartphones by Energy-Aware Dynamic Task Scheduling. In 2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (pp. 1466–1472). IEEE. https://doi.org/10.1109/HPCC.2012.214.

Quinn, G. P., & Keough, M. J. (2002). Experimental Design and Data Analysis for Biologists. Cambridge University Press. https://doi.org/10.1017/CBO9780511806384.

Sheikhalishahi, M., Grandinetti, L., Guerriero, F., Wallace, RM, & Vazquez-Poletti, JL. (2015). Multi-dimensional job scheduling, Future Generation Computer Systems, 54, 123-131. https://doi.org/10.1016/j.future.2015.03.014.

Shinde, S.K., & Gawali, M. B. (2018). Task scheduling and resource allocation in the cloud using heuristic approach. Jornal Cloud Computing, 7(1), https://doi.org/10.1186/s13677-018-0105-8.

Shojafar, M., Javanmardi, S., Abolfazli, S. (2015). FUGE: The joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and the genetic method. Cluster Computing, 18, 829-844. https://doi.org/10.1007/s10586-014-0420-x.

Skarlat, O., Nardelli, M., Schulte, S., & Dustdar, S., (2017). Towards QoS-aware Service Placement Fog, in: Procedure of the First IEEE International Conference on Fog and Edge Computing, ICFEC '17, IEEE, https://doi.org/10.1109/ICFEC.2017.12.

Sousa, A. (2017). Adaptação dinâmica e sensível ao contexto de interfaces móveis em ambiente ubíquo. (Ph.D Thesis, Universidade de Trás-os-Montes e Alto Douro). Retrieved from https://catalogo.biblioteca.utad.pt/cgi-bin/koha/opac-detail.pl?biblionumber=72842.

Stavrinides, G. L., & Karatza, H. D. (2019). A hybrid approach to real-time scheduling IoT workflows in fog and cloud environments. Multimedia Tools and Applications 78, 24639-24655. https://doi.org/10.1007/s11042-018-7051-9.

Swaroop, P. (2019). Cost Based Job Scheduling In Fog Computing, (PhD thesis, DTU, India), Available at: http://dspace.dtu.ac.in:8080/jspui/handle/repository/16722.

Tiwary, M., Sahoo, B., Yang, D., & Puthal, K. S. (2018). Response time for optimization cloudlets in Mobile Computing Edge Jornal of Parallel Distributed Computing, Vol. 119, pp. 81-91, https://doi.org/10.1016/j.jpdc.2018.04.004.

Yang Y., Zhao S., Zhang W., Chen Y., Luo X. and Wang J. (2018). DEBTS: Delay Balanced Energy Task Scheduling in Homogeneous Fog Networks. IEEE Internet of Things Journal, 5(3), 2094-2106. https://doi.org/10.1109/JIOT.2018.2823000.

Zhou, B., Dastjerdi, A. V., Calheiros, R. N., Srirama, S. N., & Buyya, R. (2017). Mcloud: The Context-Aware Offloading Framework for Heterogeneous Mobile Cloud. IEEE Transactions on Services Computing, 10(5), 797-810. https://doi.org/10.1109/TSC.2015.2511002.

Zhou, X., Sun, M., Wang, Y., & Wu, X., (2015). The New QoE-driven Video Cache Allocation Scheme for Mobile Cloud Server. In: Procedure of the 11th Conference on International Heterogeneous Networking for Quality, Reliability, Security and Robustness, QSHINE 15, IEEE, pp. 122-126.

Zhu, C., Li, X., Leung, V., Hu, X., & Yang, T.L. (2015). Towards Integration of Wireless Sensor Networks and Cloud Computing, IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom), IEEE, Singapore, (pp. 62-69). https://doi.org/10.1109/CloudCom.2015.27.