Hierarchical fog-cloud architecture to process priority-oriented health services with serverless computing

Gustavo André Setti Cassel[1], **Rodrigo da Rosa Righi**[1], **Marta Rosecler Bez**[2]

[1]Programa de Pós-Graduação em Computação Aplicada
Universidade do Vale do Rios dos Sinos (UNISINOS)
São Leopoldo, Rio Grande do Sul, Brasil

[2]Indústria Criativa – Universidade Feevale
Novo Hamburgo, Rio Grande do Sul, Brasil

gustavoasc@edu.unisinos.br, rrrighi@unisinos.br, martabez@feevale.br

Abstract. *This article presents SmartVSO - a computational model of a hierarchical, scalable fog-cloud architecture that processes vital signs with healthcare services implemented as serverless functions. Heuristics favor vital signs of people with health problems to obtain results with short response time even during usage peaks. We consider a recursive offloading mechanism of vital signs between fog nodes and cloud, in order to distribute processing based on health semantics. An experiment carried out with 80,000 vital signs indicates that the proposed model processes 60% of urgent vital signs in up to 5.3 seconds, while 60% of vital signs of healthy people are consumed in up to 1 hour and 54 min.*

Resumo. *Este artigo apresenta SmartVSO - um modelo computacional de uma arquitetura hierárquica, escalável, fog-cloud, que processa sinais vitais com serviços de saúde implementados como funções serverless. Heurísticas favorecem sinais vitais de pessoas com problemas de saúde, a fim de obterem resultados com baixo tempo de resposta mesmo durante picos de uso. Consideramos um mecanismo recursivo de offloading de sinais vitais sinais entre fog nodes e cloud, a fim de distribuir o processamento baseado em semânticas da saúde. Experimento realizado com 80.000 sinais vitais indica que o modelo proposto processa 60% dos sinais vitais urgentes em até 5,3 segundos, enquanto 60% dos sinais vitais de pessoas saudáveis são consumidos em até 1 hora e 54 minutos.*

1. Introdução

Serviços de saúde para cidades inteligentes vêm ganhando muita atenção nos últimos anos em função dos benefícios proporcionados por esse campo de pesquisa, os quais são significativos e melhoram a qualidade de vida das pessoas [Hartmann et al. 2022, Haghi Kashani et al. 2021]. Dispositivos vestíveis, como relógios inteligentes, continuamente coletam sinais vitais para que serviços de saúde os processem com uma abordagem preventiva, de modo a enviar notificações para parentes próximos ou inclusive chamar uma ambulância antes mesmo que o problema aconteça ou se agrave. Isso é especialmente importante para pessoas doentes ou com comorbidades, visto que respostas atrasadas no processamento de sinais vitais podem não ser aceitáveis. Com isso em mente, este artigo apresenta o modelo computacional SmartVSO (*Smart Vital Sign Offloading*) combinando camadas de *fog* e *cloud* para executar serviços de saúde com baixo tempo de resposta e

também lidar com picos de uso. Heurísticas são acionadas na *fog* para priorizar o processamento de sinais vitais de pessoas com problemas de saúde, de modo que tais sinais sejam consumidos mais rapidamente do que sinais de pessoas saudáveis, em função da maior urgência. As principais contribuições apresentadas por este artigo são: *i)* heurística que prioriza o processamento de sinais vitais críticos, e *ii)* estratégia recursiva para *offloading* de sinais vitais entre uma árvore de *fog nodes* e a *cloud*. Este artigo está assim estruturado: seguido desta introdução estão trabalhos relacionados, seguido do modelo SmartVSO, da metodologia utilizada para avaliação, seus resultados, e conclusão.

2. Trabalhos Relacionados

A Tabela 1 apresenta trabalhos relacionados no âmbito de *offloading* computacional para Internet das Coisas (IoT). Ao propormos um modelo computacional para processar sinais vitais com serviços de saúde, é importante entender como estudos relacionados propõem soluções para otimizar o uso de recursos na *fog*. Além disso, computação *serverless* é a principal tecnologia usada para implementar serviços de saúde no modelo SmartVSO, o qual também combina camadas *fog* e *cloud* com prioridades no âmbito da saúde. Em geral, trabalhos apresentam soluções genéricas de *offloading* sem considerar semânticas da saúde, embora incorporem prioridades para a decisão de *offloading*. A maior lacuna identificada foi a ausência de solução que combine prioridades (com semânticas de saúde) com mecanismo hierárquico de *offloading* de sinais vitais em formato de árvore, com *fog*, *cloud*, e computação *serverless* para implementar serviços escaláveis de saúde.

Tabela 1. Trabalhos relacionados sobre *offloading* de carga de trabalho para IoT.

Artigo	Contexto	Objetivo	Prioridade?	Serverless?
[Cheng et al. 2019]	Geral	Reduzir latência e tempo de resposta	✓	✓
[Pelle et al. 2021]	Geral	Otimizar o *deploy* de serviços sensíveis à latência	-	✓
[Bermbach et al. 2020]	Geral	Escalabilidade	-	✓
[George et al. 2020]	Geral	Reduzir tempo de resposta e reduzir consumo de energia	-	✓
[Dehury et al. 2021]	Geral	Encontrar deploy ótimo de funções *serverless*	✓	✓
[Rausch et al. 2021]	Geral	Usar eficientemente recursos na infraestrutura de borda	-	✓
[Cicconetti et al. 2021]	Geral	Distribuir funções *serverless* de modo descentralizado	-	✓
[AlZailaa et al. 2021]	Saúde	Reduzir latência para tarefas críticas	✓	-
[Rezazadeh et al. 2019]	Saúde	Selecionar *fog nodes* com base em recursos e latência	-	-
[Arora and Singh 2021]	Geral	Distribuir serviços heterogêneos em *fog nodes* heterogêneos	✓	-
[Bukhari et al. 2022]	Geral	Maximizar QoS e uso de recursos com regressão logística	✓	-

3. Modelo SmartVSO

O modelo SmartVSO (*Smart Vital Sign Offloading*) executa serviços de saúde de maneira distribuída, com sinais vitais coletados por relógios inteligentes, ao mesmo tempo em que prioriza sinais vitais de pessoas com problemas de saúde para receberem respostas o mais rápido possível. Serviços de saúde são implementados como funções *serverless* para fins de escalabilidade. Este modelo emprega *fog* e *cloud* para, respectivamente, obter respostas com baixa latência e lidar com picos de uso, onde *fog nodes* são hierarquicamente interligados em formato de árvore e distribuídos nos bairros da cidade. Quando um determinado *fog node* recebe um sinal vital, mas está muito sobrecarregado, ou está levemente sobrecarregado e o sinal vital recebido foi coletado de uma pessoa saudável, é recursivamente enviado para o *fog node* pai através de uma operação de *offloading*, conforme indicado na Figura 1. Caso o sinal vital chegue na *cloud* por meio de operações de *offloading*, é armazenado em uma fila e processado assincronamente por funções *serverless*.

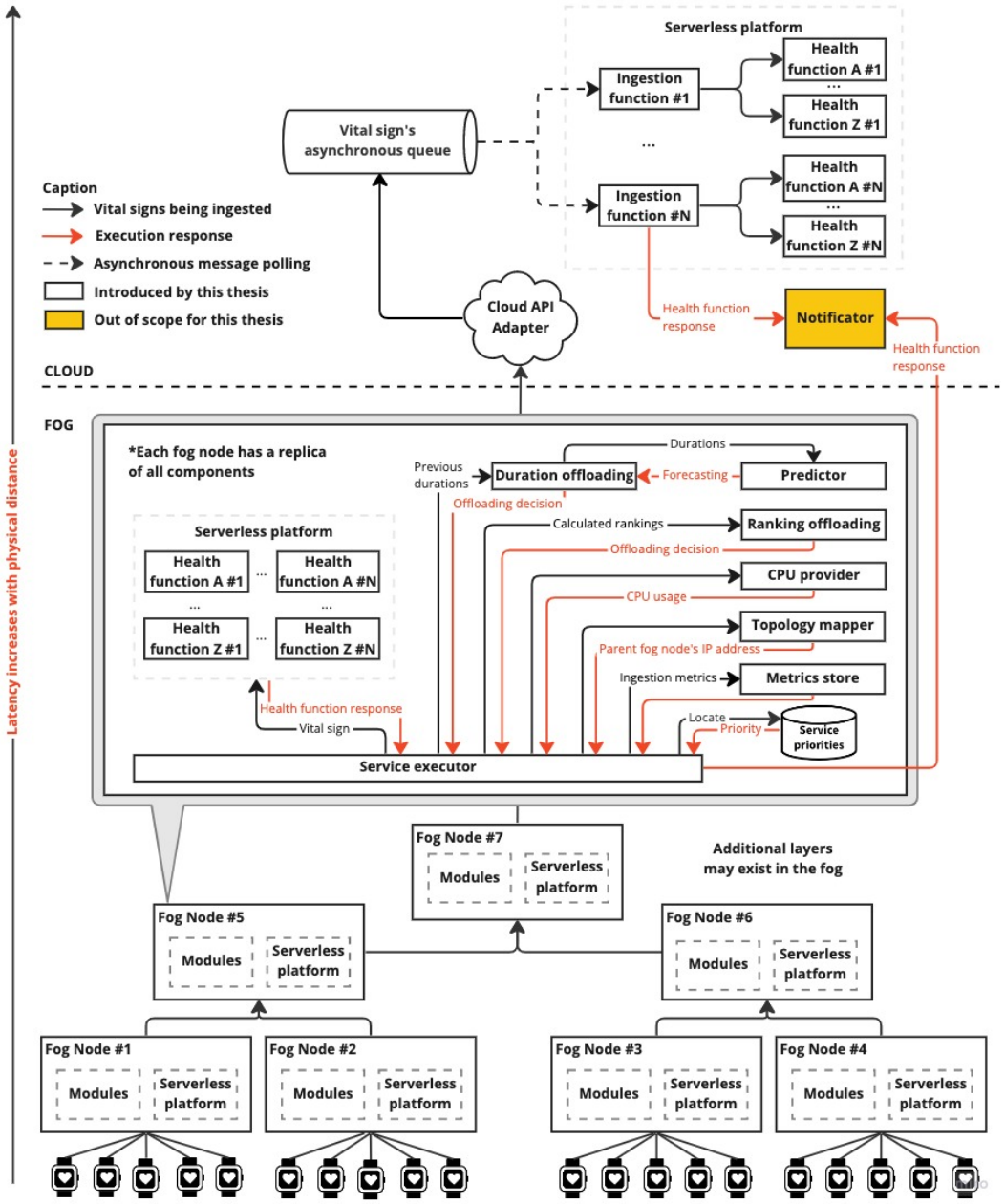

Figura 1. Arquitetura com interação entre *fog nodes* hierárquicos e a *cloud*. Cada *fog node* executa uma réplica dos módulos propostos e é conectado ao seu pai, e o *fog node* raiz é conectado à *cloud* para obter recursos virtualmente infinitos.

A Figura 2 *a)* elucida como o uso de CPU afeta a decisão de *offloading*. Todos os sinais vitais podem ser processados localmente quando o uso de CPU está abaixo do limiar inferior, independente da prioridade do sinal vital. No entanto, são enviados para o *fog node* pai quando o uso de CPU excede o limiar superior, independente da prioridade. Por fim, apenas sinais vitais prioritários são processados localmente quando o uso de CPU no *fog node* atual está entre os limiares inferior e superior, enquanto sinais vitais de baixa prioridade são enviados para o *fog node* pai e processados remotamente. Neste momento é acionada a heurística que verifica se o ranking do sinal vital recebido é maior ou menor do que demais sinais sendo processados. Dois tipos de prioridade são combinados em

um único valor numérico chamado *ranking*, apresentado na Figura 2 *b)* e resultante da equação $ranking = 2 * user\_priority + service\_priority$. Quanto maior o ranking, maior a criticidade do sinal vital. A prioridade do usuário é dinâmica e recebida juntamente com o sinal vital, visto que a pessoa pode estar saudável mas depois ficar doente, enquanto a prioridade do serviço de saúde é estática e configurada manualmente por especialistas da saúde. Quando a heurística de ranking não é capaz de efetuar a decisão de *offloading*, uma segunda heurística é acionada para decidir com *forecasting* da duração dos serviços.

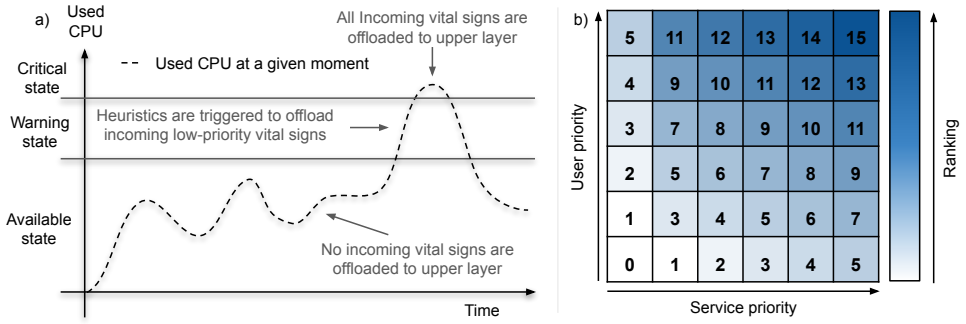

Figura 2. a) Limiares para *offloading* de sinais vitais. b) Ranking combinando prioridades de usuário e de serviço de saúde, de modo que, quanto maior maior a prioridade, mais rápido o sinal vital será atendido nos *fog nodes* hierárquicos.

4. Metodologia de Avaliação

Para avaliar este modelo foram implementados módulos[1] nas linguagens Python e Java. Emulamos o ambiente de uma cidade inteligente com *fog nodes* em São Paulo e recursos da *cloud* em Londres, consumindo sinais vitais gerados artificialmente. Instâncias *Elastic Compute Cloud* (EC2) representam *fog nodes*, enquanto a fila *Simple Queue Service* (SQS) armazena sinais vitais para processamento assíncrono. A plataforma *serverless* AWS Lambda serve para executar serviços de saúde na *cloud*, assim como OpenFaaS é a plataforma serverless utilizada na *fog*. Foram feitos 9 experimentos modificando limiares inferior e superior para decisões de *offloading*, além de considerar diferentes números de sinais vitais e técnicas para coleta de CPU, com dois serviços de saúde implementados como funções *serverless* para os testes. Estes serviços recebem como entrada o sinal vital recém coletado no formato JSON e identificam, respectivamente, se a pessoa está com febre e se terá insuficiência cardíaca nos próximos minutos. O segundo serviço considera um histórico fixo de sinais vitais para fins de predição, sendo que este segundo serviço é *CPU-bound* e faz uso mais intenso de processador do que o primeiro. A saída de ambos é uma mensagem JSON indicando se deve ser enviada notificação para o usuário (embora não envie de fato durante o experimento). Apenas o resultado de um dos experimentos é apresentado neste artigo, o qual considera 80.000 sinais vitais com limiares de 60% (inferior) e 98% (superior) de CPU, de modo que heurísticas sejam acionadas quando o uso de CPU no *fog node* está entre tais limiares. É utilizada árvore composta por 3 *fog nodes*.

5. Análise dos Resultados

A Figura 3 apresenta os tempos de resposta para o experimento detalhado na Seção 4. Todos os percentis de tempo de resposta para pessoas em estado *muito crítico* são menores

[1]https://github.com/GustavoASC/vital-signs-ingestion

do que os mesmos percentis para pessoas em estado *muito saudável*. Isso é desejado, pois prioridades maiores devem ser processadas rapidamente, tipicamente na *fog*. 60% dos sinais vitais com prioridade *muito crítica* foram processados em até 5,3 segundos, enquanto o mesmo percentil para pessoas em estado *muito saudável* é de 1 hora e 54 minutos. Isso representa 0,07% do tempo e evidencia a eficácia das heurísticas. No entanto, para percentis maiores, como 90%, o tempo de resposta é argumentavelmente alto mesmo para pessoas em estados de saúde urgentes, pois os *fog nodes* ficaram sobrecarregados durante o experimento e diversos sinais vitais foram enviados para o fim da fila na *cloud*. Quanto mais sinais vitais são enviados para o fim da fila na *cloud*, maior o tempo de resposta para percentis altos, visto que nesta versão do SmartVSO não são utilizadas heurísticas para priorização de mensagens na *cloud*. Também pode ser visto que para percentis menores, como de 10%, 20%, e 30%, o tempo de resposta é inversamente proporcional à prioridade: quanto maior a prioridade, menor o tempo de resposta, o que é desejado.

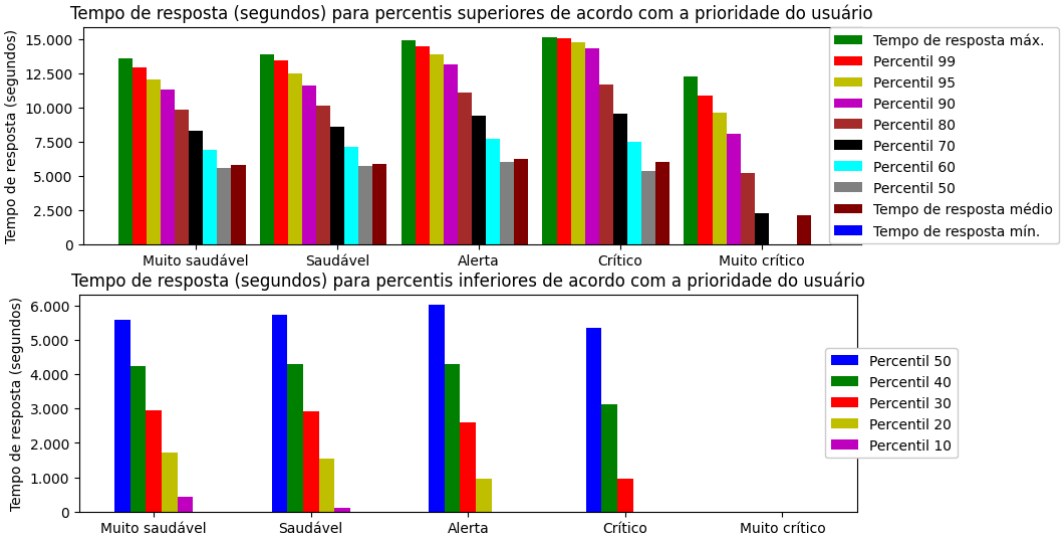

Figura 3. Tempo de resposta *versus* prioridade do usuário. Prioridades altas tendem para um tempo de resposta menor, principalmente para o percentil 60 e inferiores, o que destaca a eficácia da heurística de priorização dos sinais vitais.

6. Conclusão

Este artigo apresentou o modelo SmartVSO, o qual conecta *fog nodes* na forma de árvore para processar sinais vitais com serviços de saúde implementados como funções *serverless*, juntamente com heurísticas na *fog* para priorizar sinais vitais de pessoas em estado de saúde crítico, e computação assíncrona na *cloud* para lidar com picos de uso. Resultados indicam que a heurística de prioridades é eficaz para priorizar sinais vitais de pessoas com problemas de saúde, visto que 60% dos sinais vitais urgentes foram processados em até 5,3 segundos, enquanto 60% dos sinais vitais de pessoas saudáveis foram processados em até 1 hora e 54 minutos. Técnicas para reduzir ainda mais o tempo de resposta incluem aumentar a capacidade computacional de *fog nodes* ou solicitar aumento no número de funções *serverless* para o provedor *cloud*. Trabalhos futuros incluem aplicar heurísticas de priorização na *cloud* e calibrar automaticamente limiares de CPU na *fog*.

Referências

AlZailaa, A., Chi, H. R., Radwan, A., and Aguiar, R. (2021). Low-latency task classification and scheduling in fog/cloud based critical e-health applications. In *ICC 2021 - IEEE International Conference on Communications*, pages 1–6.

Arora, U. and Singh, N. (2021). Iot application modules placement in heterogeneous fog–cloud infrastructure. *International Journal of Information Technology*, 13(5):1975–1982.

Bermbach, D., Maghsudi, S., Hasenburg, J., and Pfandzelter, T. (2020). Towards auction-based function placement in serverless fog platforms. In *2020 IEEE International Conference on Fog Computing (ICFC)*, pages 25–31.

Bukhari, M. M., Ghazal, T. M., Abbas, S., Khan, M. A., Farooq, U., Wahbah, H., Ahmad, M., and Adnan, K. M. (2022). An intelligent proposed model for task offloading in fog-cloud collaboration using logistics regression. *Computational Intelligence and Neuroscience*, 2022:3606068.

Cheng, B., Fuerst, J., Solmaz, G., and Sanada, T. (2019). Fog function: Serverless fog computing for data intensive iot services. In *2019 IEEE International Conference on Services Computing (SCC)*, pages 28–35.

Cicconetti, C., Conti, M., and Passarella, A. (2021). A decentralized framework for serverless edge computing in the internet of things. *IEEE Transactions on Network and Service Management*, 18(2):2166–2180.

Dehury, C. K., Poojara, S. R., Domanal, S. G., and Srirama, S. N. (2021). Def-drel: Systematic deployment of serverless functions in fog and cloud environments using deep reinforcement learning. *CoRR*, abs/2110.15702.

George, G., Bakir, F., Wolski, R., and Krintz, C. (2020). Nanolambda: Implementing functions as a service at all resource scales for the internet of things. In *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 220–231.

Haghi Kashani, M., Madanipour, M., Nikravan, M., Asghari, P., and Mahdipour, E. (2021). A systematic review of iot in healthcare: Applications, techniques, and trends. *Journal of Network and Computer Applications*, 192:103164.

Hartmann, M., Hashmi, U. S., and Imran, A. (2022). Edge computing in smart health care systems: Review, challenges, and research directions. *Transactions on Emerging Telecommunications Technologies*, 33(3):e3710. e3710 ett.3710.

Pelle, I., Paolucci, F., Sonkoly, B., and Cugini, F. (2021). Latency-sensitive edge/cloud serverless dynamic deployment over telemetry-based packet-optical network. *IEEE Journal on Selected Areas in Communications*, 39(9):2849–2863.

Rausch, T., Rashed, A., and Dustdar, S. (2021). Optimized container scheduling for data-intensive serverless edge computing. *Future Generation Computer Systems*, 114:259–271.

Rezazadeh, Z., Rezaei, M., and Nickray, M. (2019). Lamp: A hybrid fog-cloud latency-aware module placement algorithm for iot applications. In *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*, pages 845–850.