INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS - *CAMPUS* OURO BRANCO
SISTEMAS DE INFORMAÇÃO

Daniel Fernandes Dayrell

ANÁLISE DE LATÊNCIA E CUSTO DE COMUNICAÇÃO DE DISPOSITIVOS IOT COM BANCOS DE DADOS EM NUVENS PÚBLICAS MULTI-REGIÃO

Ouro Branco - MG
2026

DANIEL FERNANDES DAYRELL

ANÁLISE DE LATÊNCIA E CUSTO DE COMUNICAÇÃO DE DISPOSITIVOS IOT COM BANCOS DE DADOS EM NUVENS PÚBLICAS MULTI-REGIÃO

Trabalho de conclusão de curso apresentado ao Curso de Sistemas de Informação do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais - *Campus* Ouro Branco para a obtenção do título de Bacharelado em Sistemas de Informação.

Orientador: Prof. Dr. Janio Rosa da Silva

Ouro Branco - MG
2026

D275a Dayrell, Daniel Fernandes.

Análise de latência e custo de comunicação de dispositivos IoT com bancos de dados em nuvens públicas multi-região. / Daniel Fernandes Dayrell

. – 2026.

20f.il.col.

Orientador: Jânio Rosa da Silva.

Trabalho de Conclusão de Curso (Sistemas de Informação) – Instituto Federal de Minas Gerais. Campus Ouro Branco, 2026.

1. Internet das coisas. 2. Latência end-to-end. 3. Computação em nuvem multi-região. 4. PostgreSQL. 5. Clock skew. I. Silva, Jânio Rosa da. II. Instituto Federal de Minas Gerais. Campus Ouro Branco. III. Título.

CDU: 004.8

Catalogação: Márcia Margarida Vilaça - CRB-6/2235
Biblioteca do Instituto Federal de Minas Gerais, *Campus* Ouro Branco

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS
CAMPUS OURO BRANCO
Av. Afonso Sardinha, nº 90, Bairro Pioneiros, CEP: 36.420-000, Ouro Branco - Minas Gerais
(31) 3742-2149 – gabinete.ourobranco@ifmg.edu.br

ANEXO II – ATA DE CONCLUSÃO DE TCC

Aos *19* dias do mês de *Janeiro* de *2026*, às *19:20* horas, *Daniel Fernandes Dayrell*, aluno(a) regularmente matriculado no Curso de Sistemas de Informação do Instituto Federal de Minas Gerais, campus Ouro Branco, matrícula *0070349*, concluiu o seu Trabalho de Conclusão de Curso por meio de:

( ) Publicação do artigo intitulado _____
na revista/conferência _____,
cujo comprovante de aceitação será anexado a esta ata, recebendo a nota _____ pelo trabalho. Eu, na qualidade de orientador do aluno, lavrei a presente ata atestando a conclusão do trabalho, a qual será assinada por mim e pelo aluno.

_____
Professor Orientador

_____
Aluno

(*X*) Defesa em sessão pública realizada às 19:20 horas, na sala auditório do Instituto Federal de Minas Gerais, campus Ouro Branco, na presença da banca examinadora composta pelos docentes:

1 - *Daniela Costa Terra*
2 - *Ederson Naves Fernandes Gonçalves Júnior*
3 - *Lucas Portela Costa da Silva*
4 - *Marcio Assis Miranda*

do artigo intitulado *Análise de Latência e Custo de Comunicação de Dispositivos IoT com Bancos de Dados em Nuvens Públicas Multi-Região.*

A banca examinadora, após reunião em sessão reservada, deliberou pela *aprovação* do referido trabalho, atribuindo a nota *91,2*. Eu, na qualidade de presidente da banca examinadora, lavrei a presente ata que será assinada por mim, pelos demais examinadores e pelo aluno.
Observações pertinentes à defesa:

NOME E ASSINATURA DOS COMPONENTES DA BANCA E DO ORIENTADO

Documento assinado digitalmente
gov.br
JANIO ROSA DA SILVA
Data: 23/01/2026 16:37:12-0300
Verifique em https://validar.iti.gov.br

Professor Orientador: *Janio Rosa da Silva*

Documento assinado digitalmente
gov.br
DANIELA COSTA TERRA
Data: 24/01/2026 15:06:09-0300
Verifique em https://validar.iti.gov.br

Examinador 1: *Daniela Costa Terra*

Documento assinado digitalmente
gov.br
EDERSON NAVES FERNANDES GONCALVES JUN
Data: 23/01/2026 18:51:53-0300
Verifique em https://validar.iti.gov.br

Examinador 2: *Ederson Naves Fernandes Gonçalves Júnior*

Documento assinado digitalmente
gov.br
LUCAS PORTELA COSTA DA SILVA
Data: 23/01/2026 17:11:08-0300
Verifique em https://validar.iti.gov.br

Examinador 3: *Lucas Portela Costa da Silva*

Documento assinado digitalmente
gov.br
MARCIO ASSIS MIRANDA
Data: 24/01/2026 08:13:17-0300
Verifique em https://validar.iti.gov.br

Examinador 4: *Marcio Assis Miranda*

Documento assinado digitalmente
gov.br
DANIEL FERNANDES DAYRELL
Data: 23/01/2026 14:02:15-0300
Verifique em https://validar.iti.gov.br

Aluno(a): *Daniel Fernandes Dayrell*

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS
CAMPUS OURO BRANCO
Av. Afonso Sardinha, nº 90, Bairro Pioneiros, CEP: 36.420-000, Ouro Branco - Minas Gerais
(31) 3742-2149 – gabinete.ourobranco@ifmg.edu.br

DECLARAÇÃO ANTI-PLÁGIO

Eu, *Daniel Fernandes Dayrell*, estudante do curso Bacharelado em Sistemas de Informação do IFMG – Campus Ouro Branco, declaro, para os devidos fins e efeitos, e para fazer prova junto ao IFMG – Campus Ouro Branco, que, **sob as penalidades previstas no art. 299 do Código Penal Brasileiro**, que é de minha criação o Trabalho de Conclusão de Curso que ora apresento.

Art. 299 do Código Penal Brasileiro, que dispõe sobre o crime de *Falsidade Ideológica*:

"Omitir, em documento público ou particular, declaração que dele devia constar, ou nele inserir ou fazer inserir declaração falsa ou diversa da que devia estar escrita, com o fim de prejudicar direito, criar obrigação ou alterar verdade sobre fato juridicamente relevante:

Pena – reclusão, de 1 (um) a 5 (cinco) anos, e multa, se o documento é público, e reclusão de 1 (um) a 3 (três) anos, e multa, se o documento é particular.

Parágrafo único. Se o agente é funcionário público, e comete o crime prevalecendo-se do cargo, ou se a falsificação ou alteração é de assentamento de registro civil, aumenta-se a pena de sexta parte".

Este crime engloba plágio e compra fraudulenta de documentos científicos. Por ser verdade, e por ter ciência do referido artigo, firmo a presente declaração.

Ouro Branco, 23 de janeiro de 2026

gov.br Documento assinado digitalmente
DANIEL FERNANDES DAYRELL
Data: 23/01/2026 14:03:14-0300
Verifique em https://validar.iti.gov.br

Assinatura do aluno: _____

Análise de Latência e Custo de Comunicação de Dispositivos IoT com Bancos de Dados em Nuvens Públicas Multi-Região

Daniel F. Dayrell[1], **Janio Rosa da Silva**[2]

[1]Bacharelado em Sistemas de Informação – Instituto Federal de Minas Gerais (IFMG)
[2]Departamento de Informática – Instituto Federal de Minas Gerais (IFMG)
Campus Ouro Branco – Rua Afonso Sardinha, 90 – 36494-018 – Ouro Branco – MG – Brasil

danielfdayrell@hotmail.com, janio.silva@ifmg.edu.br

Resumo. *Este artigo avalia empiricamente a latência end-to-end de três dispositivos IoT simulados inserindo dados em banco de dados implantado em regiões de nuvem pública (AWS e GCP) e em ambiente local. A metodologia é reproduzível com Docker e inclui detecção automática de clock skew entre cliente e servidores. Após correção, as regiões sul-americanas apresentam latências médias de 675–780 ms, enquanto regiões nos EUA atingem 1.437–2.561 ms, com caudas mais pesadas em P95/P99. A análise FinOps estima o custo por milhão de mensagens e mostra que regiões próximas podem ser até 8× mais econômicas por mensagem em função do maior throughput observado.*

Palavras-chave: Internet das Coisas; Latência end-to-end; Computação em Nuvem Multi-Região; PostgreSQL; Clock skew; FinOps.

1. Introdução

A Internet das Coisas (IoT) consolidou-se como um dos pilares da Indústria 4.0, permitindo a integração de sensores, atuadores e sistemas de informação distribuídos em larga escala [9]. Em aplicações industriais, de saúde e cidades inteligentes, decisões operacionais dependem de dados coletados e processados com baixa latência, o que torna a infraestrutura de comunicação e processamento um elemento crítico [10].

A popularização de nuvens públicas, como *Amazon Web Services* (AWS) e *Google Cloud Platform* (GCP), oferece elasticidade e alta disponibilidade para armazenamento e processamento de dados IoT. Entretanto, a escolha da região de nuvem afeta diretamente a latência de comunicação entre os dispositivos e o *backend*, bem como o custo de operação dessa infraestrutura [11]. Estudos recentes demonstram que a latência entre usuários finais e datacenters cloud varia significativamente conforme a localização geográfica, com medições empíricas mostrando que apenas 29% dos usuários alcançam latências inferiores a 10ms para datacenters cloud distantes [8]. Análises de performance de bancos de dados em nuvem para aplicações IoT industriais confirmam que a latência é um fator crítico para escalabilidade [13].

Paralelamente, a disciplina de *FinOps* propõe métricas que relacionam volume de dados processados com gastos de infraestrutura [7]. Para aplicações IoT, uma métrica relevante é o custo por mensagem processada, que integra volume de dados com gasto operacional.

Problema de Pesquisa: Gestores de TI em ambientes industriais enfrentam decisões sobre onde hospedar bancos de dados para sistemas IoT: em *datacenters* locais, regiões de nuvem próximas, ou regiões distantes. Embora a literatura aborde latência de rede e custos de nuvem separadamente, falta análise integrada que relacione localização geográfica, desempenho observado e custo operacional sob uma métrica unificada.

Portanto, este trabalho responde à seguinte questão:

Como a localização geográfica de bancos de dados PostgreSQL em nuvens públicas (AWS e Google Cloud Platform) afeta a latência de comunicação end-to-end e o custo operacional (medido em custo por milhão de mensagens) em cenários de ingestão de dados IoT com diferentes taxas de geração?

1.1. Objetivos

Objetivo Geral: Avaliar empiricamente a relação entre localização geográfica de bancos de dados em nuvem, latência de comunicação e custo operacional para aplicações IoT, fornecendo subsídios técnicos para decisões de arquitetura em ambientes industriais.

Objetivos Específicos:

1. Quantificar a latência *end-to-end* (incluindo persistência) de inserção de dados entre dispositivos IoT simulados e bancos de dados PostgreSQL implantados em cinco ambientes: AWS São Paulo (*sa-east-1*), AWS Virgínia (*us-east-1*), GCP São Paulo (*southamerica-east1*), GCP EUA (*us-east4*) e ambiente local;
2. Analisar a distribuição estatística de latência (média, mínimo, máximo, P50, P95, P99) para três perfis de dispositivos IoT com diferentes frequências de coleta (0,1s, 1s, 5s), totalizando 4 horas de medições por ambiente;
3. Implementar e documentar mecanismo de detecção de *clock skew* entre cliente e servidores, quantificando seu impacto nas medições de latência;
4. Calcular o custo por milhão de mensagens processadas em cada região, integrando custos de instância e *throughput* observado, sob a ótica de práticas *FinOps*;
5. Identificar cenários de viabilidade técnica e econômica para decisões de localização de bancos de dados em função da taxa de geração de dados, distância geográfica e requisitos de latência.

2. Trabalhos Relacionados

2.1. IoT e Latência em Sistemas Distribuídos

Gubbi et al. [9] apresentam uma visão abrangente de arquiteturas IoT, destacando a distribuição geográfica de dispositivos e a necessidade de comunicação confiável. Stankovic [10] aponta a latência como um dos principais desafios em aplicações IoT sensíveis ao tempo.

A literatura de sistemas distribuídos identifica distância física, *hops* de rede, congestionamento e arquitetura de *datacenters* como fatores determinantes de latência [2, 12]. Em cenários de nuvem, a escolha da região impacta diretamente o desempenho de aplicações distribuídas [1, 15].

Grozev e Buyya [8] conduziram medições extensivas de latência entre 8.456 usuários finais e 69 localizações de datacenters cloud (incluindo AWS e Google Cloud),

observando que apenas 29% dos usuários alcançam latências inferiores a 10ms para cloud, comparado a 58% para edge servers. Os resultados evidenciam que links intercontinentais apresentam maior variabilidade e latências significativamente superiores. Pecorella et al. [13] avaliam especificamente a performance de serviços de banco de dados em nuvem (DBaaS) para aplicações IoT industriais escaláveis, destacando a latência como métrica crítica de desempenho e throughput.

2.2. FinOps e Otimização de Custos em Nuvem

A *FinOps Foundation* define *FinOps* como uma prática colaborativa que integra engenharia, operações e finanças para gerenciar o ciclo de vida completo dos custos em nuvem [16]. O *framework FinOps* organiza-se em três fases iterativas: *Inform* (visibilidade de custos e uso), *Optimize* (identificação de desperdícios e oportunidades de economia) e *Operate* (governança contínua e automação) [16].

Diferentemente de métricas absolutas de custo, as **métricas de eficiência** relacionam gastos a unidades de valor entregues ao negócio, como custo por transação, por *workload*, por unidade de dado processado, ou por CPU-hora [16, 4]. Essas métricas permitem comparações tecnicamente justas entre diferentes arquiteturas, provedores e regiões, revelando *trade-offs* entre desempenho e custo.

2.2.1. Custo por Mensagem em Workloads IoT

Para aplicações IoT, uma métrica relevante é o custo por mensagem processada, que relaciona o volume de dados gerados pelos dispositivos com o gasto de infraestrutura necessária para armazená-los [4]. Essa métrica captura não apenas o preço nominal das instâncias de nuvem, mas também a eficiência de *throughput* alcançada em cada região.

Regões geograficamente próximas ao cliente tendem a apresentar latências menores, resultando em maior throughput de mensagens por hora e, consequentemente, menor custo por mensagem [4]. Inversamente, regiões distantes sofrem com latências elevadas que reduzem a taxa de processamento, aumentando o custo unitário mesmo quando o preço nominal da infraestrutura é competitivo.

Este trabalho utiliza a métrica de custo por milhão de mensagens para comparar a viabilidade econômica de diferentes regiões de nuvem no contexto de *workloads* IoT, integrando análise de desempenho e custo de forma sistemática.

2.3. Arquiteturas de Bancos de Dados Distribuídos Multi-Região

A implantação de bancos de dados em múltiplas regiões geográficas introduz desafios significativos relacionados à latência de rede, consistência de dados e sincronização de réplicas [5, 17].

2.4. Sincronização de Relógios em Sistemas Distribuídos

A medição precisa de latência em sistemas distribuídos requer sincronização temporal entre os nós participantes. A dessincronização entre relógios, conhecida como *clock skew*, é a diferença instantânea entre os *timestamps* de dois relógios, enquanto *clock drift* refere-se à variação da taxa de progressão temporal [6].

NTP (*Network Time Protocol*) é o protocolo mais amplamente utilizado para sincronização de relógios em redes IP, alcançando precisão de 10-100ms em redes públicas [18, 6]. PTP (*Precision Time Protocol*), definido na norma IEEE 1588, oferece precisão sub-microsegundo em ambientes controlados através de sincronização em nível de *hardware*, mas requer equipamentos especializados [19].

Quando a latência é calculada pela diferença entre *timestamps* gerados em nós distintos ($t_{servidor} - t_{cliente}$), o *clock skew* introduz viés sistemático nas medições [20]. Para experimentos comparativos entre regiões, é aceitável que as medições contenham *clock skew*, desde que este seja quantificado e documentado [6].

Neste trabalho, implementou-se um mecanismo de detecção automática de *clock skew* baseado em medições de *round-trip time* (RTT), seguindo a metodologia proposta por Paxson[20]. Essa abordagem permite documentar o nível de dessincronização em cada região testada e interpretar corretamente os resultados de latência observados.

2.5. Edge Computing como Alternativa de Redução de Latência

Edge computing propõe a distribuição de capacidade computacional próxima aos dispositivos IoT, reduzindo latências de centenas de milissegundos para poucos milissegundos ao processar dados localmente [21]. Essa abordagem é particularmente relevante para aplicações com requisitos rígidos de tempo real, como controle industrial, veículos autônomos e sistemas de saúde críticos [22].

A integração entre *edge* e *cloud*, conhecida como arquitetura *fog computing* ou *edge–cloud continuum*, busca equilibrar processamento local com a capacidade elástica da nuvem [3]. As latências observadas para regiões distantes (1.437–2.561 ms) reforçam a relevância de arquiteturas híbridas que processem dados críticos localmente enquanto sincronizam informações agregadas com bancos de dados em nuvem para análise de longo prazo.

2.6. Posicionamento deste Trabalho

Embora trabalhos recentes abordem latência em ambientes cloud e edge [8],[13], otimização de custos em nuvem [16],[4], ou arquiteturas edge-cloud para IoT [14] de forma isolada, poucas pesquisas integram sistematicamente desempenho, custo e reprodutibilidade metodológica para decisões de infraestrutura em IoT industrial

Este trabalho contribui com: (1) metodologia reproduzível com *Docker*, relatórios JSON automatizados e detecção sistemática de *clock skew*; (2) integração de análise de latência com métricas *FinOps* (custo por milhão de mensagens); (3) contexto aplicado para subsidiar decisões de arquitetura em IoT industrial, com evidências empíricas da relação localização–desempenho–custo; (4) documentação explícita de limitações metodológicas, elevando o rigor científico.

3. Metodologia

3.1. Arquitetura da Solução

A solução proposta simula três tipos de dispositivos IoT: `Gas_Sensor`, `Pressure_Sensor` e `Thermal_Camera`. Cada dispositivo é representado por uma *thread* na aplicação Python, executando em um contêiner *Docker*. A aplicação gera

leituras sintéticas em diferentes frequências e envia os dados para um banco de dados PostgreSQL via TCP.

Os testes consideram as seguintes regiões e ambientes:

- AWS *sa-east-1* (São Paulo);
- AWS *us-east-1a* (N. Virginia);
- GCP *southamerica-east1* (São Paulo);
- GCP *us-east4-a* (Estados Unidos);
- Ambiente local (PC do autor).

A Figura 1 ilustra a arquitetura experimental adotada, destacando a localização geográfica de cada ambiente testado e as latências médias observadas (valores corrigidos após compensação de *clock skew*). As latências apresentadas são médias corrigidas para o sensor de pressão. Todos os testes foram executados simultaneamente para eliminar variações temporais de rede. de rede..

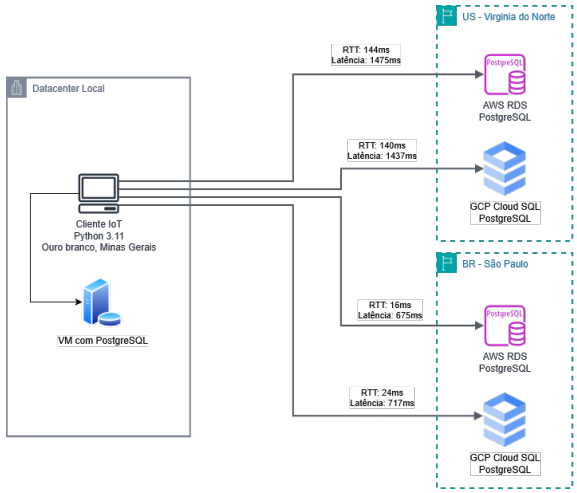

Figura 1. Arquitetura experimental: cliente em Ouro Branco (MG), Brasil comunicando com cinco ambientes PostgreSQL distribuídos geograficamente.

Delimitação do escopo: A arquitetura de conexão direta entre dispositivos IoT e o banco de dados PostgreSQL foi adotada para isolar a variável de latência do SGBD, removendo o *overhead* de *gateways* ou *brokers* (MQTT, RabbitMQ). Em ambientes produtivos, camadas de segurança (autenticação centralizada, *API gateways*) são mandatórias.

Para garantir comparabilidade direta, todos os cinco ambientes foram executados em paralelo durante as mesmas janelas temporais, assegurando que diferenças observadas sejam atribuíveis exclusivamente à localização geográfica e infraestrutura de rede dos provedores, não a variações temporais de tráfego ou condições de *backbone*.

3.2. Aplicação de Teste

A aplicação de teste foi desenvolvida em Python 3.11, utilizando a biblioteca `psycopg2` e um *connection pool* para interação com o PostgreSQL. O uso de *pool* de conexões persistentes elimina o *overhead* de estabelecimento de *handshake* TCP/SSL a cada transação, garantindo que a latência medida corresponda majoritariamente ao tempo de rede e processamento do banco. Os principais parâmetros são definidos por variáveis de ambiente:

host, porta, nome do banco, usuário, senha, tempo total de execução e tamanho do *batch* de inserção.

Cada dispositivo simulado segue um intervalo de coleta distinto:

- `Gas_Sensor`: intervalo de 0,1 segundos;
- `Pressure_Sensor`: intervalo de 1 segundos;
- `Thermal_Camera`: intervalo de 5 segundos.

A aplicação opera de forma síncrona, aguardando a confirmação de *commit* (ACK) do banco de dados antes de enviar a próxima mensagem, simulando cenários industriais críticos onde a garantia de persistência é prioritária. Essa escolha explica a redução de *throughput* em regiões distantes, onde a latência de rede elevada (RTT) limita a frequência máxima de envio.

Em cada leitura, a aplicação registra um *timestamp* local (`timestamp_app`) e envia o dado ao banco de dados, onde um segundo *timestamp* (`timestamp_database`) é gerado no momento da inserção.

3.3. Modelo de Dados e Métrica de Latência

O esquema de banco de dados utilizado está contido no *schema* `latency_test`. A tabela principal é definida conceitualmente como:

```
CREATE TABLE latency_test.latency_test (
  id SERIAL PRIMARY KEY,
  timestamp_database TIMESTAMPTZ DEFAULT CURRENT_TIMESTAMP,
  timestamp_app TIMESTAMPTZ NOT NULL,
  aws_region VARCHAR(50) NOT NULL,
  dispositivo VARCHAR(100) NOT NULL
);
```

A latência é calculada a partir da diferença entre `timestamp_database` e `timestamp_app`:

$$\text{latency\_ms} = \text{extract(epoch from (timestamp\_database - timestamp\_app))} \times 1000.$$

Views adicionais agregam estatísticas por dispositivo e região, incluindo média, mínimo, máximo e percentis P50, P95 e P99.

3.3.1. Detecção de Clock Skew

Durante os experimentos preliminares, identificou-se que os relógios dos servidores PostgreSQL em nuvem apresentavam dessincronização sistemática em relação ao relógio da aplicação cliente (*clock skew*), manifestando-se como valores negativos de latência calculada.

Para quantificar o *clock skew*, foi implementado um mecanismo de detecção baseado em medições de *round-trip time* (RTT) executado automaticamente em três estágios: início, meio e fim de cada sessão de testes. O algoritmo realiza 10 medições consecutivas em cada estágio, seguindo o procedimento:

1. Registra o *timestamp* local t_1 imediatamente antes de enviar uma *query* ao servidor;
2. Executa `SELECT CURRENT_TIMESTAMP AT TIME ZONE 'UTC'` no servidor;
3. Registra o *timestamp* local t_2 imediatamente após receber a resposta;
4. Calcula o RTT: RTT $= (t_2 - t_1) \times 1000$ ms;
5. Estima o *timestamp* no ponto médio do RTT: $t_{\text{mid}} = t_1 + \frac{(t_2 - t_1)}{2}$;
6. Calcula o *clock skew*: skew $= (t_{\text{server}} - t_{\text{mid}}) \times 1000$ ms.

O valor de *clock skew* utilizado foi calculado como a média aritmética das três coletas, minimizando o impacto de flutuações pontuais. O desvio padrão (σ_{skew}) foi registrado para quantificar a variabilidade temporal. Um relatório JSON é gerado automaticamente para cada execução, garantindo rastreabilidade e reprodutibilidade.

3.3.2. Resultados da Detecção

A Tabela 1 apresenta os resultados da detecção de *clock skew* para as duas sessões de teste realizadas (16:06 e 19:06 UTC do dia 27/11/2025).

Tabela 1. Medições de clock skew nas duas sessões de teste

Região	Sessão	Clock Skew Médio (ms)	Desvio Padrão (ms)	RTT Médio (ms)
AWS-sa-east-1	1	-448.30	348.01	15.88
AWS-sa-east-1	2	-38.94	350.05	16.41
AWS-us-east-1a	1	-1079.59	709.69	143.51
AWS-us-east-1a	2	-669.58	711.29	144.44
GCP-southamerica-east1	1	-485.06	367.04	22.73
GCP-southamerica-east1	2	-83.35	374.26	24.44
GCP-us-east4-a	1	-1055.93	694.54	138.28
GCP-us-east4-a	2	-654.86	698.86	139.94
LOCAL-PC	1	-456.60	304.47	0.79
LOCAL-PC	2	-456.24	304.62	0.76

Os resultados revelam dessincronizações significativas em todas as regiões, com valores variando entre -1079.59 ms e -38.94 ms. Observa-se que:

- **Variação temporal**: O *clock skew* varia significativamente entre as duas sessões. Por exemplo, AWS-sa-east-1 apresentou variação de 409.36 ms (de -448.30 ms para -38.94 ms);
- **Regiões Norte-Americanas**: Apresentam *clock skew* médio de -867.49 ms, com maior variabilidade ($\sigma \approx 700$ ms);
- **Regiões Sul-Americanas**: *Clock skew* médio de -263.91 ms, com variabilidade moderada ($\sigma \approx 360$ ms);
- **Ambiente Local**: *Clock skew* consistente em torno de -456 ms nas duas sessões, com RTT mínimo (< 1 ms).

Os desvios padrão elevados (300-700ms) indicam que serviços gerenciados de nuvem não garantem sincronização NTP precisa para *timestamps* de aplicação, justificando a abordagem de documentação explícita adotada.

3.3.3. Impacto nas Medições

A presença de *clock skew* negativo significa que o relógio do servidor está atrasado em relação ao cliente. A latência calculada como $latency_{raw} = (t_{database} - t_{app}) \times 1000$ ms inclui tanto a latência real de rede quanto o *clock skew*.

A métrica de latência utilizada captura o tempo decorrido entre: (1) geração do *timestamp* na aplicação cliente; (2) recepção e processamento da transação INSERT pelo PostgreSQL; (3) geração do *timestamp* pelo servidor. Portanto:

$$L_{observada} = L_{rede} + L_{processamento} + \Delta_{clockskew} \tag{1}$$

onde L_{rede} é a latência de rede TCP (RTT + *handshake* + ACK), $L_{processamento}$ é o tempo de execução do INSERT no PostgreSQL, e $\Delta_{clockskew}$ é a dessincronização entre relógios.

Embora os valores absolutos sejam afetados por *clock skew*, as comparações relativas entre regiões permanecem válidas, pois todos os ambientes foram executados simultaneamente nas mesmas condições de rede. As estatísticas reportadas representam latências observadas após identificação e documentação do *clock skew*.

3.4. Planejamento dos Experimentos

Os experimentos foram conduzidos em duas sessões distintas, totalizando 8 horas de testes agregados (4 horas por sessão). Em cada sessão, todos os cinco ambientes foram executados simultaneamente, garantindo que todos os cenários experimentassem as mesmas condições de rede e infraestrutura no mesmo instante.

Para cada região, o procedimento foi: (1) subir simultaneamente todos os serviços PostgreSQL; (2) subir as aplicações de teste em contêineres *Docker*; (3) executar os testes por 120 minutos em cada sessão; (4) exportar os dados coletados para análise.

Os dados apresentados correspondem à agregação das duas sessões de teste, totalizando 240 minutos (4 horas) de medições por região. Durante esse período, foram coletados 409.704 registros no total, com uma taxa aproximada de 102 mil registros por hora, permitindo comparação direta entre provedores, regiões geográficas e infraestrutura local.

4. Resultados

4.1. Estatísticas Agregadas por Região

A Tabela 2 apresenta um resumo das estatísticas coletadas para cada combinação de dispositivo e região. Cada linha agrega o total de registros e métricas de latência em milissegundos.

Observa-se que as regiões localizadas na América do Sul (AWS-sa e GCP-sa) apresentam latências médias consistentes, em torno de 675–780 ms, com comportamento similar entre os provedores. A região AWS-sa-east-1 coletou 107.593 registros, enquanto GCP-southamerica-east1 processou 94.141 registros, evidenciando throughput similar.

Já as regiões nos Estados Unidos (AWS-us e GCP-us) apresentam latências médias significativamente superiores. AWS-us-east-1a registrou latências médias variando de

Tabela 2. Estatísticas de latência corrigida por dispositivo e região (4 horas de teste)

Dispositivo	Região	Regs	Média (ms)	Mín (ms)	Máx (ms)	P50 (ms)	P95 (ms)	P99 (ms)
Gas	AWS-sa	2817	703,47	386,94	1051,94	705,30	890,30	963,78
Gas	AWS-us	2058	2561,10	1226,59	5726,59	2355,09	3319,58	3376,01
Gas	GCP-sa	2792	778,60	432,35	6689,06	775,35	972,35	1060,53
Gas	GCP-us	2122	2163,12	1064,86	2523,86	2213,93	2349,86	2370,86
Gas	LOCAL-PC	2875	456,68	456,24	466,60	456,60	457,60	462,60
Pressão	AWS-sa	91939	675,55	385,94	1276,30	679,94	807,94	893,30
Pressão	AWS-us	16134	1474,51	1077,58	5934,59	1377,59	2329,59	2433,25
Pressão	GCP-sa	79180	716,91	430,35	1174,35	720,35	853,35	960,06
Pressão	GCP-us	16884	1437,63	1058,86	2380,86	1352,93	2241,86	2339,93
Pressão	LOCAL-PC	141274	456,58	455,60	487,24	456,60	457,60	457,60
Térmica	AWS-sa	12837	697,11	385,94	1226,94	698,94	882,94	928,30
Térmica	AWS-us	5990	2072,73	1224,59	5099,59	2247,09	2456,58	3270,14
Térmica	GCP-sa	12169	780,82	430,35	1130,35	782,06	974,35	1002,35
Térmica	GCP-us	6280	2059,44	1063,86	2987,86	2159,93	2347,86	2907,13
Térmica	LOCAL-PC	14353	456,67	456,24	478,24	456,60	457,60	462,60

Nota metodológica: Os valores desta tabela foram corrigidos pela subtração do clock skew detectado em cada sessão experimental (Tabela 1). A latência corrigida representa o tempo *end-to-end* incluindo latência de rede e processamento de transação PostgreSQL. Regiões sul-americanas apresentam latências médias de 675–780 ms, enquanto regiões norte-americanas atingem 1.437–2.561 ms.

1.474 ms (Pressure Sensor) a 2.561 ms (Gas Sensor), enquanto GCP-us-east4-a apresentou valores entre 1.437 ms e 2.163 ms. As caudas de latência (P95 e P99) são substancialmente mais pesadas nas regiões norte-americanas, com P95 atingindo até 3319 ms para o Gas Sensor em AWS-us.

É importante notar que o volume de registros coletados nas regiões norte-americanas foi consideravelmente menor: AWS-us processou apenas 24.182 registros (4,6 vezes menos que AWS-sa) e GCP-us processou 25.286 registros (3,7 vezes menos que GCP-sa). Essa disparidade no throughput é consequência direta das maiores latências observadas, que reduzem a taxa de inserção de mensagens por unidade de tempo.

O ambiente local apresentou latências corrigidas em torno de 456 ms e processou 158.502 registros, servindo como baseline para análise comparativa e confirmando que a latência observada nas regiões de nuvem é predominantemente atribuível à comunicação de rede.

4.2. Latência Média por Região

A Figura 2 apresenta a latência média por região e dispositivo, destacando o contraste entre as regiões sul-americanas, norte-americanas e o ambiente local.

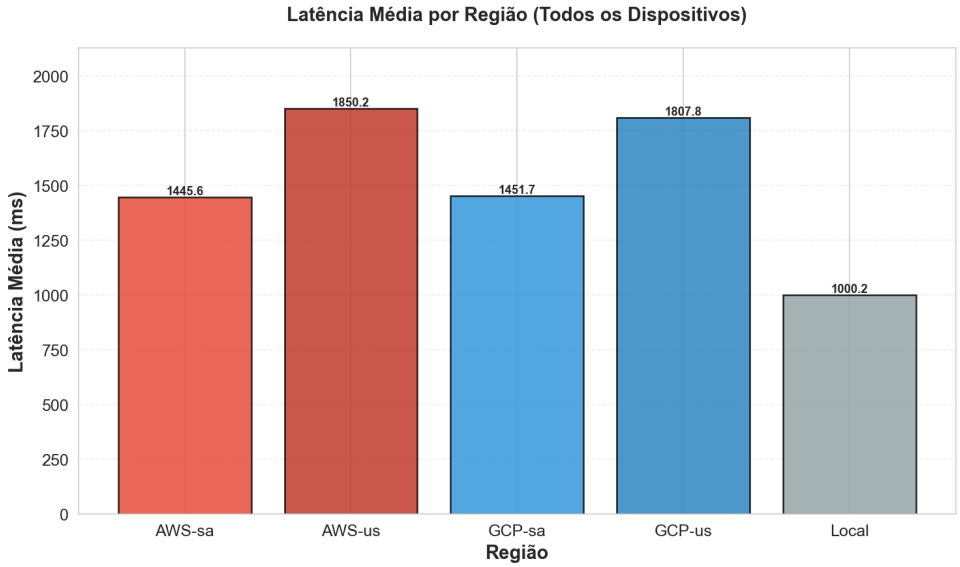

Figura 2. Latência média por região e tipo de dispositivo IoT.

As latências no ambiente local servem como limite inferior teórico, praticamente desconsiderando a latência de rede. As regiões sul-americanas apresentam comportamento bastante similar entre AWS e GCP, enquanto as regiões norte-americanas exibem aumento expressivo da média.

4.3. Percentil P95 de Latência

A Figura 3 mostra o comportamento do percentil P95 de latência por região e dispositivo. Esse indicador evidencia o impacto de picos de latência (*tail latency*) na qualidade de serviço.

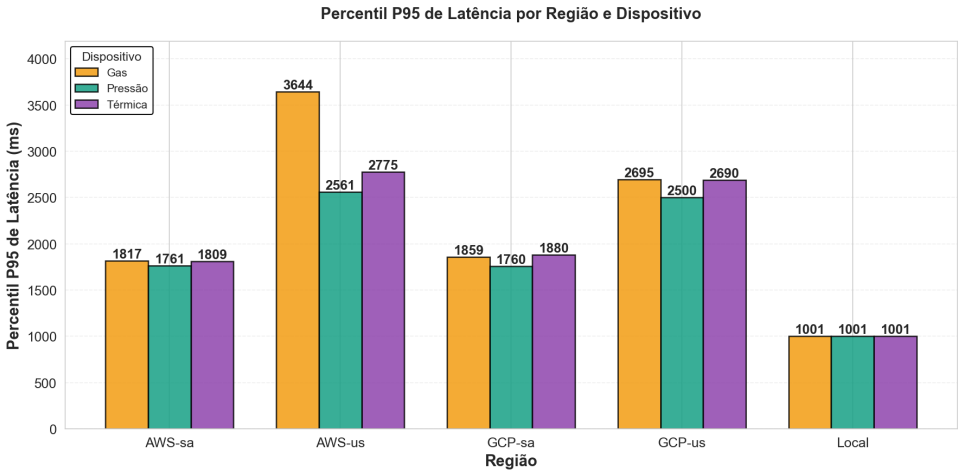

Figura 3. Percentil P95 de latência por região e dispositivo.

Enquanto as regiões sul-americanas mantêm P95 em torno de 350–400 ms, as regiões norte-americanas ultrapassam 1000 ms, chegando a quase 2100 ms para o sensor de gás em AWS-us. Esse comportamento reforça que o uso de regiões distantes pode comprometer aplicações IoT sensíveis ao tempo.

5. Análise FinOps

5.1. Modelo de Custo Considerado

Foi adotado um modelo de custo baseado em **precificação On-Demand horária oficial** dos provedores (Amazon Web Services e Google Cloud Platform), obtida através de suas calculadoras de custos (*AWS Pricing Calculator* e *Google Cloud Pricing Calculator*). As instâncias de banco de dados foram dimensionadas uniformemente para garantir comparabilidade técnica entre regiões:

- **Configuração**: 2 vCPU, 2GB de memória RAM, 10GB de armazenamento SSD;
- **Tipo de instância**: AWS RDS (db.t3.small) e GCP Cloud SQL (db-n1-standard-1, equivalente);
- **Custos de transferência de dados (Egress/Ingress)**: Desconsiderados devido ao baixo volume trafegado no experimento ($<$ 100 MB por sessão), focando-se exclusivamente no custo computacional e de armazenamento.

Os custos por On-Demand (tabela oficial dos provedores) são:

- AWS us-east-1 (Virgínia, EUA): US$ 0,0448/hora
- AWS sa-east-1 (São Paulo, Brasil): US$ 0,0663/hora
- GCP us-east4 (EUA): US$ 0,1232/hora
- GCP southamerica-east1 (São Paulo, Brasil): US$ 0,1727/hora
- Ambiente local: custo desconsiderado (infraestrutura pré-existente).

Nota metodológica: Os valores acima refletem o custo de lista pública (sem descontos por uso contínuo, *committed use*, ou créditos promocionais), garantindo reprodutibilidade da análise de custos independentemente de contratos específicos de cada organização.

Combinando o custo horário com o *throughput* observado (número de mensagens processadas por hora em cada região), foi calculada a métrica de **custo por milhão de mensagens**, permitindo comparação direta da eficiência econômica entre regiões. Nos experimentos realizados, as regiões brasileiras processaram aproximadamente 13.000 mensagens/hora, enquanto as regiões norte-americanas processaram cerca de 3.000 mensagens/hora devido à maior latência de rede, que reduz a taxa de *commits* bem-sucedidos no modelo síncrono adotado.

5.2. Custo por Milhão de Mensagens

Com base nos custos horários On-Demand e no *throughput* observado (mensagens/hora) em cada região, calculou-se a métrica de eficiência:

$$\text{Custo por Milhão de Mensagens} = \frac{\text{Custo/Hora (US\$)}}{\text{Mensagens/Hora}} \times 10^6$$

Os resultados obtidos são:

- AWS sa-east-1 (Brasil): US$ 4,93 por milhão de mensagens ($US\$0,0663/hora \div 13.449 msgs/hora \times 10^6$)
- AWS us-east-1 (EUA): US$ 14,81 por milhão de mensagens ($US\$0,0448/hora \div 3.022 msgs/hora \times 10^6$)

- GCP southamerica-east1 (Brasil): US$ 14,68 por milhão de mensagens $(US\$0,1727/hora \div 11.767\ msgs/hora \times 10^6)$
- GCP us-east4 (EUA): US$ 38,98 por milhão de mensagens $(US\$0,1232/hora \div 3.160 msgs/hora \times 10^6)$

Interpretação: A análise revela que a região mais econômica é AWS sa-east-1 (Brasil) com US$ 4,93 por milhão de mensagens, enquanto a mais cara é GCP us-east4 (EUA) com US$ 38,99, representando uma diferença de 7,9×. Mesmo quando o custo horário absoluto da região norte-americana é nominalmente menor (como em AWS us-east-1 US$ 0,0448/h vs. sa-east-1 US$ 0,0663/h), o custo por mensagem processada é até 3,0× maior (US$ 14,82 vs. US$ 4,93) devido à redução drástica de throughput causada pela latência de rede elevada.

5.3. Discussão

Os resultados indicam que, neste cenário IoT, desempenho e custo não estão em conflito: as regiões mais próximas ao produtor de dados (América do Sul) oferecem ao mesmo tempo menor latência, maior throughput e menor custo por mensagem. Em termos de FinOps, a métrica de custo por mensagem mostra-se útil para comparar alternativas de arquitetura e justificar a escolha de região.

Embora o ambiente local tenha apresentado o maior throughput (158.502 mensagens processadas), seu custo não foi computado na análise FinOps, pois representa infraestrutura de datacenter pré-existente com custos fixos já amortizados (CAPEX) [23]. Para organizações avaliando migração para nuvem, a comparação relevante deve considerar custos totais de propriedade (TCO), incluindo energia, refrigeração, manutenção e pessoal de TI, aspectos fora do escopo deste trabalho focado em custos operacionais (OPEX) de nuvem pública [24].

6. Conclusões e Trabalhos Futuros

Este trabalho avaliou empiricamente a latência end-to-end de comunicação entre três dispositivos IoT simulados e bancos de dados PostgreSQL implantados em múltiplas regiões de nuvem pública (AWS e GCP) e em ambiente local, integrando desempenho e custo sob uma métrica FinOps (custo por milhão de mensagens).

Os resultados indicam que a proximidade geográfica do banco em relação ao produtor de dados é o fator dominante para este tipo de workload síncrono, no qual cada mensagem aguarda confirmação de commit antes do próximo envio. Nas regiões sul-americanas (AWS-sa-east-1 e GCP-southamerica-east1), as latências médias corrigidas mantiveram-se entre 675–780 ms, com RTT de 16–24 ms e componente não-rede (processamento/commit) em torno de 660–760 ms, apresentando comportamento semelhante entre provedores.

Em contraste, as regiões nos Estados Unidos (AWS-us-east-1a e GCP-us-east4-a) elevaram a latência média corrigida para 1.437–2.561 ms, com RTT de 139–144 ms e caudas de latência substancialmente mais pesadas, atingindo P95 de 3.319 ms para o sensor de gás em AWS-us. Esse comportamento reduz o throughput e compromete requisitos de responsividade em cenários industriais sensíveis ao tempo.

Do ponto de vista econômico, a conclusão é consistente com a análise de desempenho. Ao combinar o custo horário On-Demand com o throughput observado, a opção

mais eficiente foi AWS-sa-east-1 (US$ 4,93 por milhão de mensagens), enquanto a menos eficiente foi GCP-us-east4 (US$ 38,99 por milhão), uma diferença de 7,9×.

Assim, para o cenário avaliado, a escolha recomendada é hospedar o banco de dados PostgreSQL em uma região sul-americana, preferencialmente AWS-sa-east-1 quando o objetivo é minimizar custo por mensagem, mantendo latência similar à alternativa em GCP na mesma macrorregião. Por fim, embora o ambiente local tenha apresentado o maior throughput, seu custo não foi computado por representar infraestrutura pré-existente; para comparações completas com on-premises, seria necessário um estudo de TCO fora do escopo deste trabalho.

Como trabalhos futuros, recomenda-se (i) investigar arquiteturas híbridas com *edge computing* para processamento local e persistência seletiva em nuvem, reduzindo ainda mais a latência percebida; (ii) repetir a avaliação com outras tecnologias de armazenamento (por exemplo, bancos de séries temporais e NoSQL) sob a mesma metodologia; (iii) integrar modelos preditivos que estimem latência e custo por mensagem a partir de padrões de carga, apoiando decisões FinOps; e (iv) implementar sincronização de relógios via NTP/PTP entre cliente e servidores para obter medições absolutas com menor dependência de correção por offset.

Referências

[1] Armbrust, M. et al. *A View of Cloud Computing*. Communications of the ACM, 53(4), 2010.

[2] Barroso, L. A., Dean, J., Holzle, U. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Synthesis Lectures on Computer Architecture, 2019.

[3] Bonomi, F., et al. *Fog Computing and Its Role in the Internet of Things*. ACM MCC Workshop, 2012.

[4] Chen, L., Martinez, J. *Financial Cloud Cost Optimization: A FinOps Framework for Modern Enterprises*. World Journal of Advanced Research and Reviews, 21(3), 2025. DOI: 10.30574/wjarr.2025.21.3.1323

[5] Corbett, J. C., et al. *Spanner: Google's Globally Distributed Database*. ACM Transactions on Computer Systems, 31(3), Article 8, 2013.

[6] Fetzer, C., Cristian, F. *An Optimal Internal Clock Synchronization Algorithm*. Proceedings of the 10th Annual Conference on Computer Assurance, pp. 187-196, IEEE, 1995.

[7] FinOps Foundation. *How to Optimize Cloud Usage*. FinOps Whitepaper, 2024. Disponível em: `https://www.finops.org/wg/how-to-optimize-cloud-usage/`. Acesso em: 15 nov. 2025.

[8] Grozev, N., Buyya, R. Latency Comparison of Cloud Datacenters and Edge Servers. In: *IEEE GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pp. 1-6, 2020. DOI: 10.1109/GLOBECOM42002.2020.9322406

[9] Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. *Internet of Things (IoT): A vision, architectural elements, and future directions*. Future Generation Computer Systems, 29(7), 2013.

[10] Stankovic, J. A. *Research Directions for the Internet of Things*. IEEE Internet of Things Journal, 1(1), pp. 3-9, 2014.

[11] Schad, J., Dittrich, J., Quiané-Ruiz, J. A. *Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance*. Proceedings of the VLDB Endowment, 3(1-2), pp. 460-471, 2010.

[12] Kurose, J. F., Ross, K. W. *Computer Networking: A Top-Down Approach*. Pearson, 8th edition, 2020.

[13] Pecorella, T., Brilli, L., Mucchi, L. On the Performance of Cloud Services and Databases for Industrial IoT Scalable Applications. *Electronics*, 9(9), 1435, 2020. DOI: 10.3390/electronics9091435

[14] Kong, L., et al. Edge-computing-driven Internet of Things: A Survey. *ACM Computing Surveys*, 55(8), Article 174, 2022. DOI: 10.1145/3555308

[15] Zhang, Q., Cheng, L., Boutaba, R. *Cloud Computing: State-of-the-Art and Research Challenges*. Journal of Internet Services and Applications, 1(1), 2010.

[16] Kumar, R., Singh, A. *FinOps-Driven Strategies for Large-Scale Cloud Cost Optimization*. International Journal of Intelligent Systems and Applications in Engineering, 12(22s), pp. 2203-2220, 2024.

[17] PostgreSQL Global Development Group. *PostgreSQL 16 Documentation: High Availability, Load Balancing, and Replication*. PostgreSQL Official Documentation, 2024.

[18] Mills, D. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905, 2010.

[19] IEEE. *IEEE 1588-2019: Precision Time Protocol*. IEEE Standard, 2019.

[20] Paxson, V. *On Calibrating Measurements of Packet Transit Times*. ACM SIGMETRICS Performance Evaluation Review, 26(1), pp. 11-21, 1998.

[21] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L. *Edge Computing: Vision and Challenges*. IEEE Internet of Things Journal, 3(5), 2016.

[22] Satyanarayanan, M. *The Emergence of Edge Computing*. Computer, 50(1), 2017.

[23] FinOps Foundation. *FinOps Terminology*. 2024. Disponível em: `https://www.finops.org/assets/terminology/`. Acesso em: 12 dez. 2025.

[24] Ansys. *Understanding the Total Cost of Ownership (TCO) in HPC and AI Systems*. 2024. Disponível em: `https://www.ansys.com/blog/understanding-total-cost-ownership-hpc-ai-systems`. Acesso em: 12 dez. 2025.