

Quase 100 anos de cooperação em computação distribuída - Transcorrer, Resultados e Perspectivas

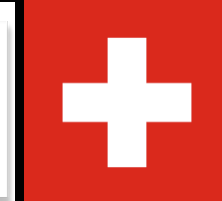
Workshop: Um Olhar Atual sobre Sistemas Distribuídos:
Da Pesquisa à Aplicação no Mundo Real

Fernando **Dotti**

fernando.dotti@pucrs.br

fldotti.github.io

PUCRS Pontifícia Universidade Católica do Rio Grande do Sul, Brazil



Sistemas Distribuídos

Desafios técnicos

- concorrência
- aspectos temporais
 - inexistência de relógio global
 - assincronia/sincronia/sincronia parcial
- falhas parciais

Sistemas Distribuídos

Requisitos

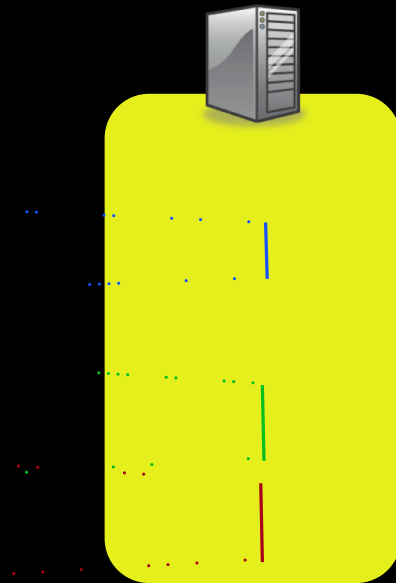
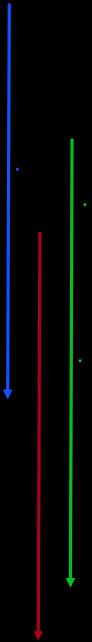
- Altamente disponíveis
 - Devem tolerar falhas
- Escaláveis
 - Capacidade de atender demandas maiores com a adição de recursos computacionais
- Consistentes
 - Oferecem modelo de consistência adequado para a aplicação

Sistemas Distribuídos

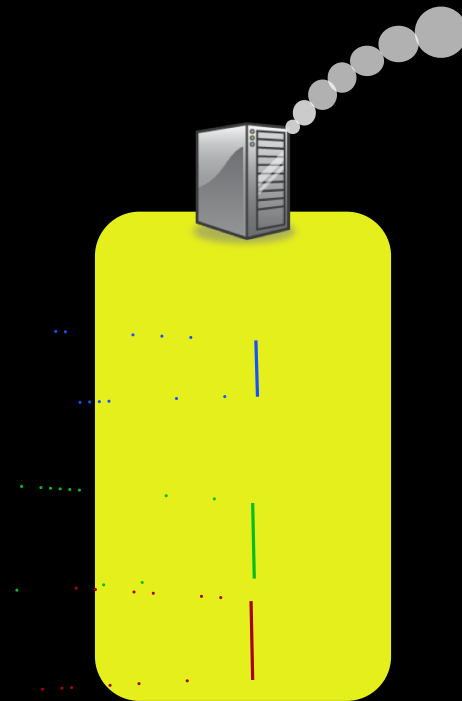
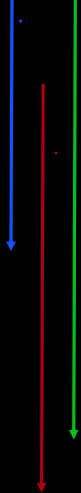
Abordagens básicas

- Alta disponibilidade
 - Tolerância a falhas -> Replicação
- Escalabilidade
 - Vertical -> Nodos computacionais mais rápidos
 - Horizontal -> Adição de nodos computacionais
- Consistência
 - Diferentes modelos -> Formas de coordenação entre processos

Sistema



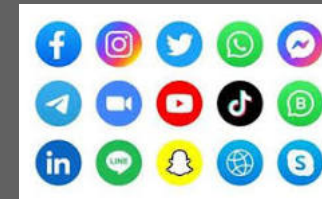
Sistema



Armazenamento



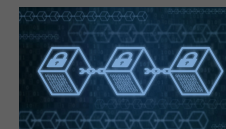
Bancos de dados
Data intensive systems



Redes Sociais



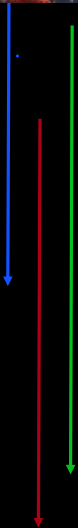
Bancos



Blockchains

Dynamo,
DynamoDB,
Redis,
Voldemort,
BigTable,
Cassandra,
MongoDB,
CouchDB,
AsterixDB,
Spanner,
CockroachDB,
Calvin, Spark
Streaming, Flink,
Kafka Streams,
...

Sistema



Armazenamento



Bancos de dados
Data intensive systems



Redes Sociais



Bancos



Blockchains

Dynamo,
DynamoDB,
Redis,
Voldemort,
BigTable,
Cassandra,
MongoDB,
CouchDB,
AsterixDB,
Spanner,
CockroachDB,
Calvin, Spark
Streaming, Flink,
Kafka Streams,
...

Sistema

Escalabilidade

Alta Disponibilidade

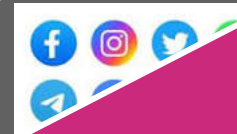
Consistência



Armazenamento



Bancos de dados
Data intensive



Redes Sociais



Bancos



Blockchains

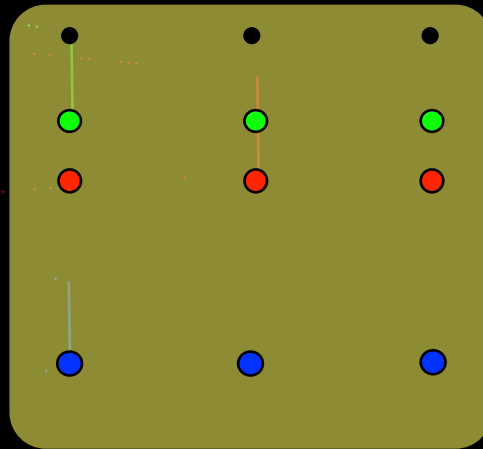
Dynamo,
DynamoDB,
Redis,
Voldemort,
BigTable,
Cassandra,
CouchDB,
HBase,
HDFS,
Hive,
MapR, Spark
Streaming, Flink,
Kafka Streams,
...

Sistemas Distribuídos

Alta disponibilidade -> **Replicação**

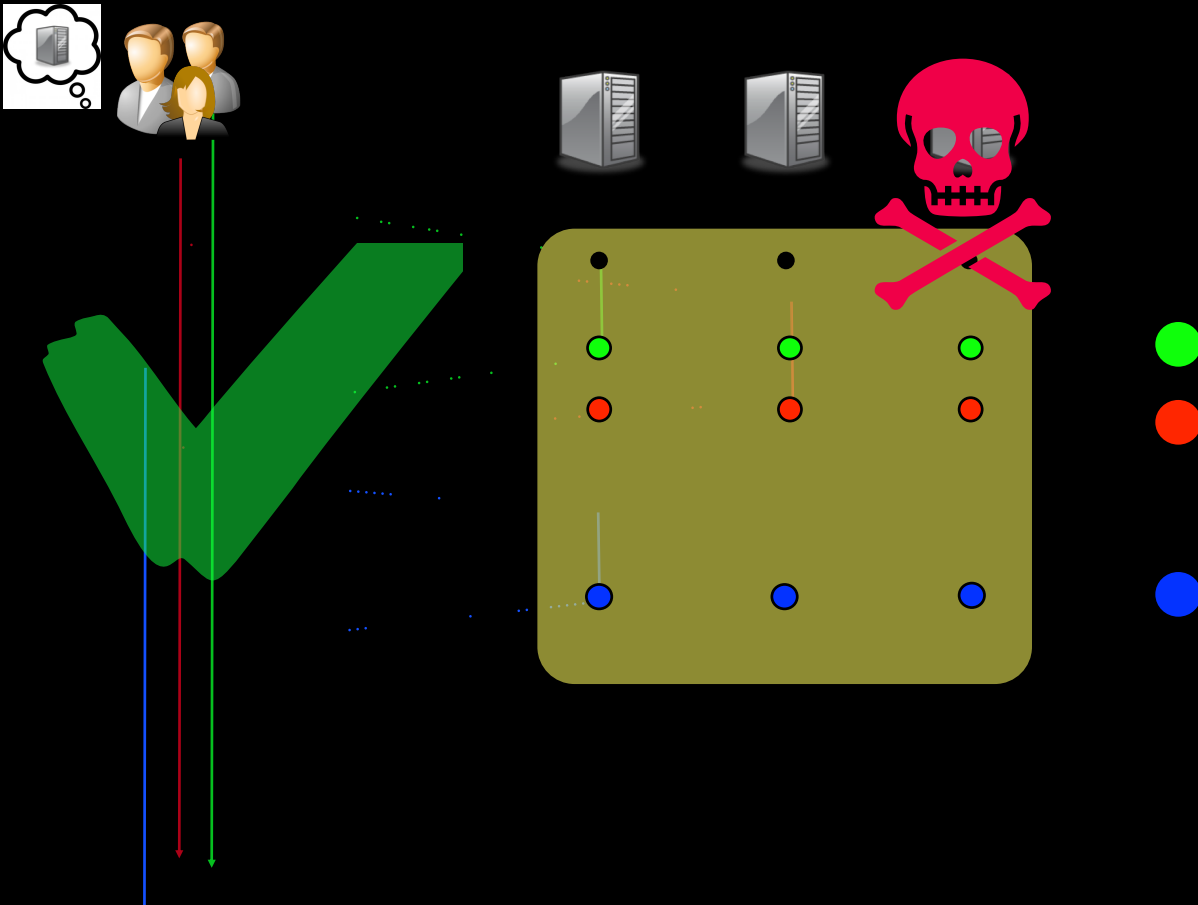


...
...
...
...
...



Sistemas Distribuídos

Alta disponibilidade -> **Replicação**



Sistemas Distribuídos

Replicação

Como manter réplicas consistentes?

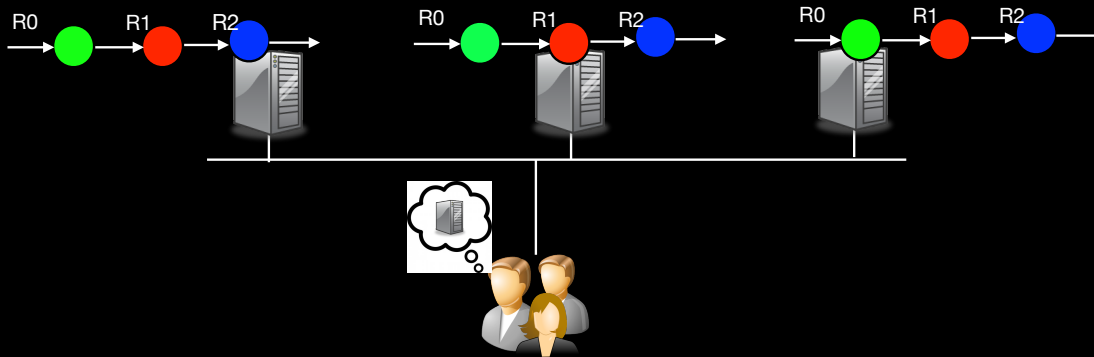
Replicação Máquina de Estados

Princípios

Lamport,
Schneider

- Réplicas iniciam no mesmo estado e ...
- processam deterministicamente ...
- a mesma ordem total de requisições

Consistência

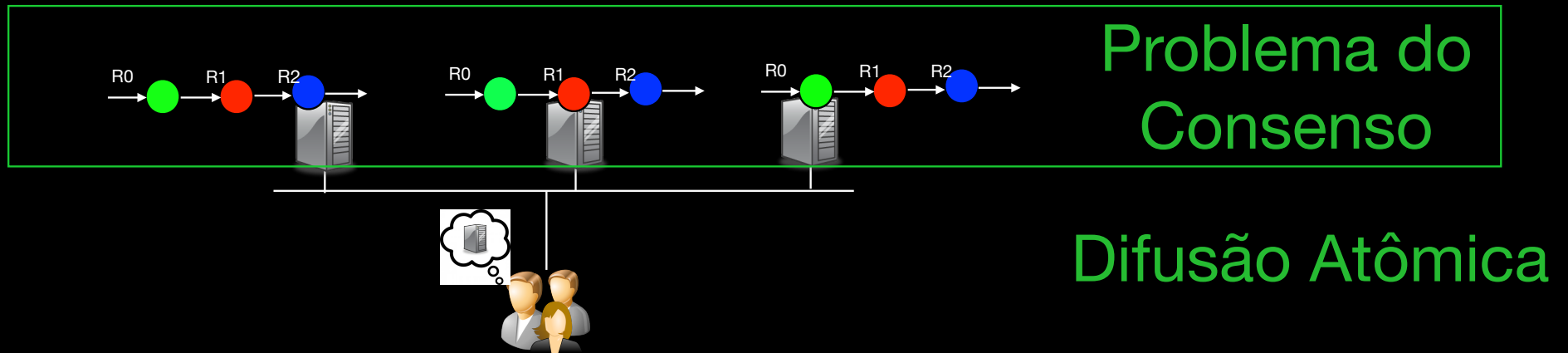


Replicação Máquina de Estados

Princípios

- Réplicas iniciam no mesmo estado e ...
- Processam deterministicamente ...
- a mesma ordem total de requisições

} Consistência



RME

Adoção

- 1. Distributed Databases & Key-Value Stores
 - Google Spanner (Uses Paxos/Raft for cross-node consistency)
 - Amazon DynamoDB (Uses a form of quorum-based replication)
 - CockroachDB (Built on Raft for strong consistency)
 - Etcd (Uses Raft for leader election and distributed consensus)
 - Consul (Uses Raft for service discovery and configuration management)
 - Zookeeper (Uses Zab, a variant of Paxos, for distributed coordination)
- 2. Blockchain & Cryptocurrencies
 - Ethereum (Previously Proof-of-Work, now Proof-of-Stake)
 - Tendermint (Byzantine Fault Tolerant consensus for Cosmos)
- 3. Cloud and Service Coordination
 - Google Borg & Kubernetes (K8s) (Kubernetes uses Etcd, which is based on Raft)
 - Amazon ECS & EKS (Uses distributed consensus mechanisms for service orchestration)
 - Apache Mesos (Uses Zookeeper for leader election and state consistency)
- 4. Distributed File Systems
 - Google Colossus (GFS successor) (Uses Paxos-based metadata management)
 - HDFS NameNode (Hadoop Distributed File System) (Uses Zookeeper for high availability)
 - GlusterFS (Uses a quorum-based mechanism for consistency)
- 5. Messaging Systems & Logs
 - Apache Kafka (Uses Zookeeper for leader election)
 - Google Cloud Pub/Sub (Uses Spanner for consistency)
 - RabbitMQ (Uses quorum queues for reliable message storage)
- 6. Configuration & Service Discovery
 - Netflix Eureka (Uses a leader election mechanism for availability)
 - Apache Zookeeper & Etcd (For distributed configuration and coordination)
- 7. Gaming Backends
 - Keep game state highly-available

RME

Adoção

- 1. Distributed Databases & Key-Value Stores
 - Google Spanner (Uses Paxos/Raft for cross-node consistency)
 - Amazon DynamoDB (Uses a form of quorum-based replication)
 - CockroachDB (Built on Raft for strong consistency)
 - Etcd (Uses Raft for leader election and distributed consensus)
 - Consul (Uses Raft for service discovery and configuration management)
 - Zookeeper (Uses Zab, a variant of Paxos, for distributed coordination)
- 2. Blockchain & Cryptocurrencies
 - Ethereum (Previously Proof of Work)
 - Tendermint (Byzantine Fault Tolerant)
- 3. Cloud and Service Coordination
 - Google Borg & Kubernetes
 - Amazon ECS & EKS (Uses Kubernetes)
 - Apache Mesos (Uses Zookeeper)
- 4. Distributed File Systems
 - Google Colossus (GFS successor)
 - HDFS NameNode (Hadoop)
 - GlusterFS (Uses a quorum-based replication)
- 5. Messaging Systems & Logs
 - Apache Kafka (Uses Zookeeper)
 - Google Cloud Pub/Sub (Uses a quorum-based replication)
 - RabbitMQ (Uses quorum-based replication)
- 6. Configuration & Service Discovery
 - Netflix Eureka (Uses a leader election mechanism for availability)
 - Apache Zookeeper & Etcd (For distributed configuration and coordination)
- 7. Gaming Backends
 - Keep game state highly-available



RME

E a escalabilidade ?



RME

E a escalabilidade ?



RME

Escalabilidade

(simples) aumento de réplicas não resolve escala

- Todas realizam mesma computação
[mesmo desempenho (ou pior)]

Como escalar RME ?

RME

Escalabilidade

Muitos avanços na área nos últimos 20+ anos ...

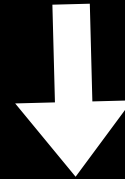
Nossas contribuições para

1. escalar verticalmente
2. escalar horizontalmente
3. recuperar rapidamente
4. sistemas BFT e blockchains

1) RME - Escalando Verticalmente

RME - Escalando Verticalmente

- Para manter consistência
Basta **respeitar a ordem total** para requisições **conflitantes**



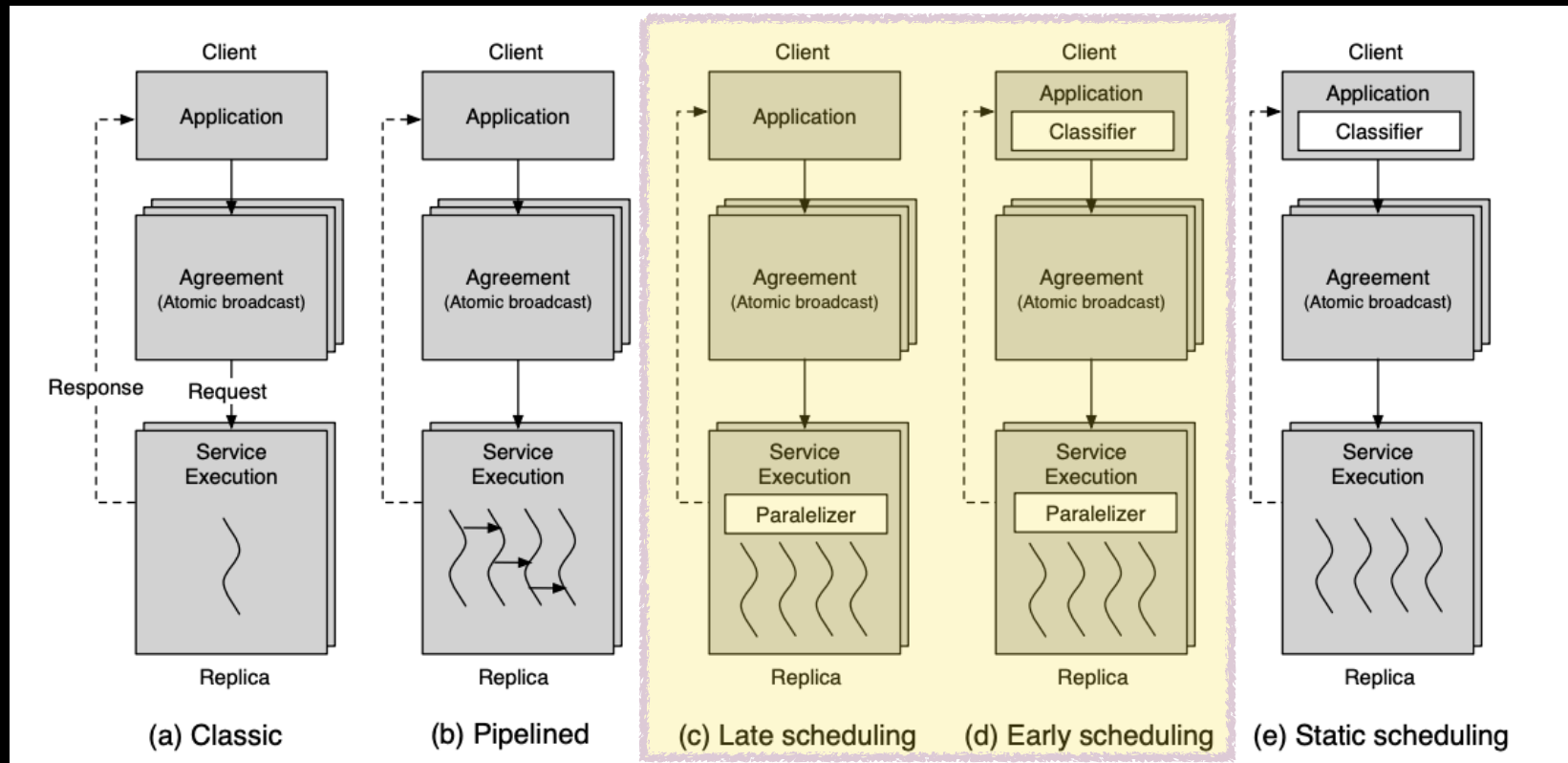
2 requisições conflitam se:
**utilizam mesmos dados e
ao menos uma os modifica**

- Ordem total —> **Ordem parcial**

Maior concorrência

RME - Escalando Verticalmente

Diversas formas de escalonamento de requisições em RME

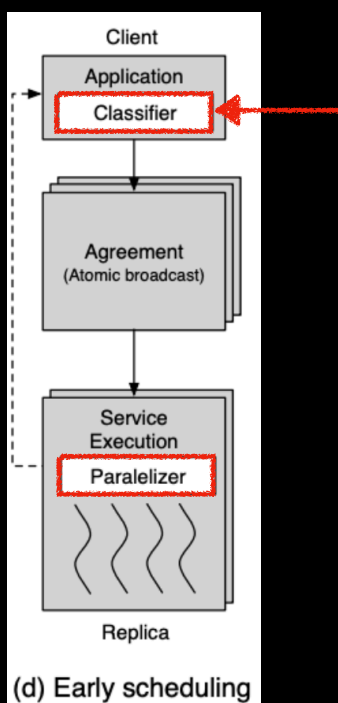


Alchieri, Dotti, Parisa, Odorico, Pedone:

Boosting State Machine Replication with Concurrent Execution. LADC 2018

RME - Escalando Verticalmente

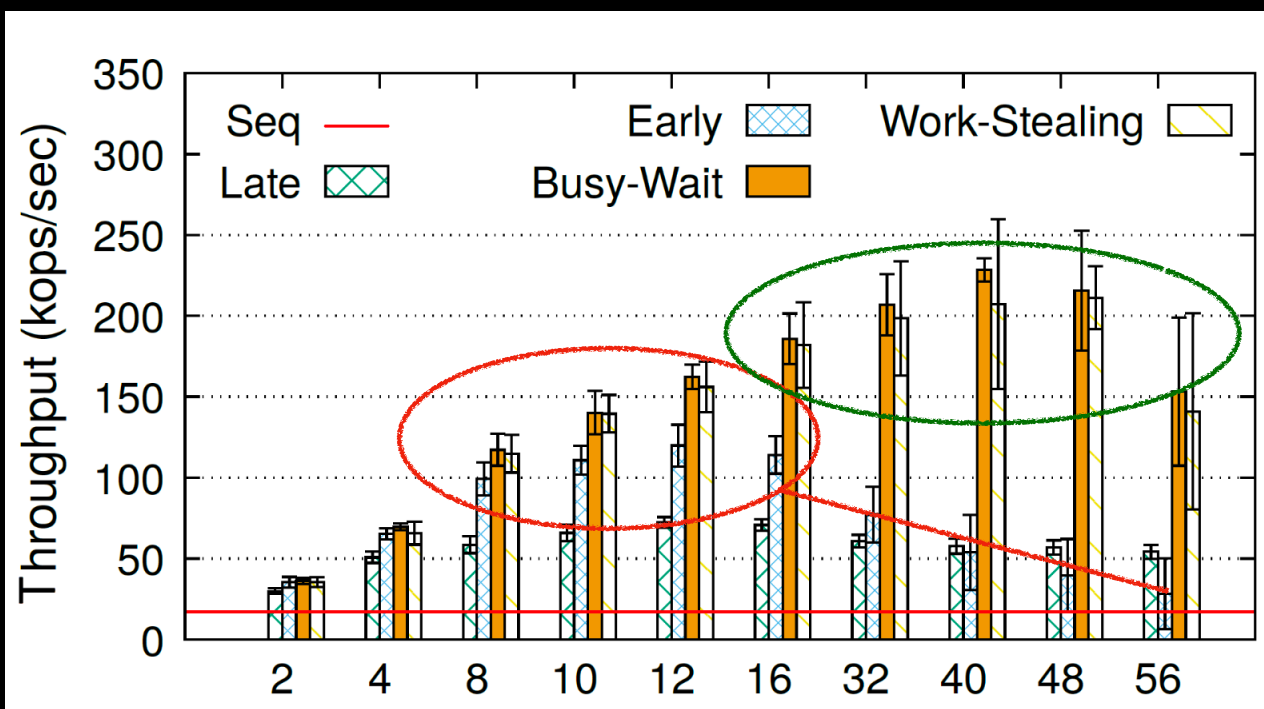
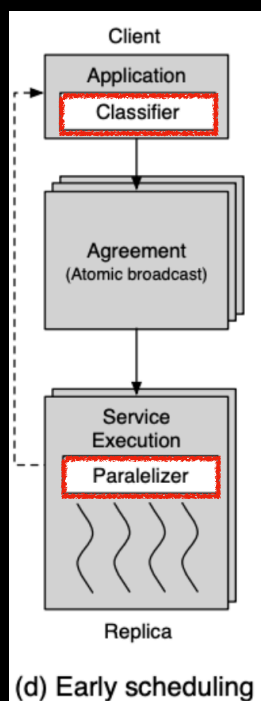
Escalonamento **antecipado**



Alchieri, Dotti, Pedone: SoCC 2018
Early Scheduling in Parallel State Machine Replication

RME - Escalando Verticalmente

Escalonamento **antecipado** - sincronização, work stealing

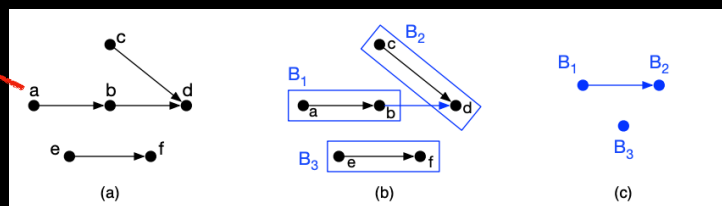
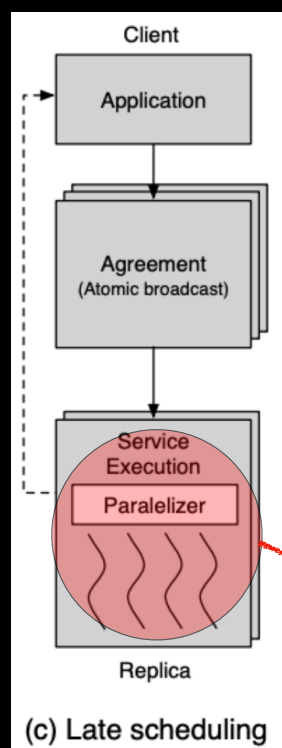


Eliã, Alchieri, Dotti, Pedone - Elsevier JPDC 2022

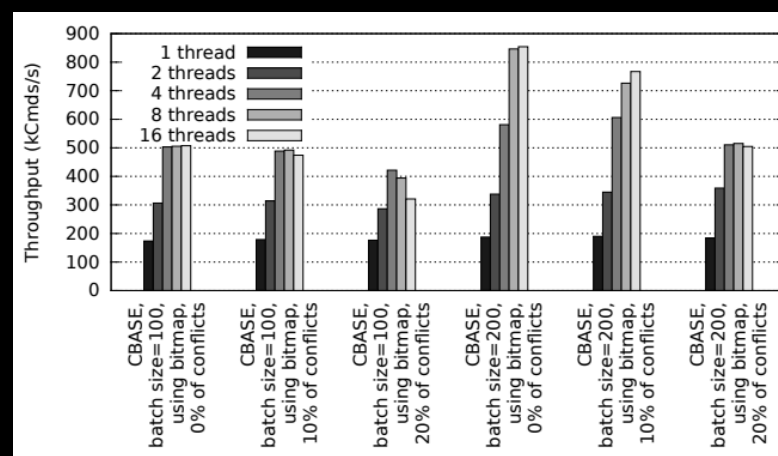
Early Scheduling on Steroids - Boosting Parallel State Machine Replication

RME - Escalando Verticalmente

Escalonamento **tardio** - grafo de dependências entre requisições (lotes)



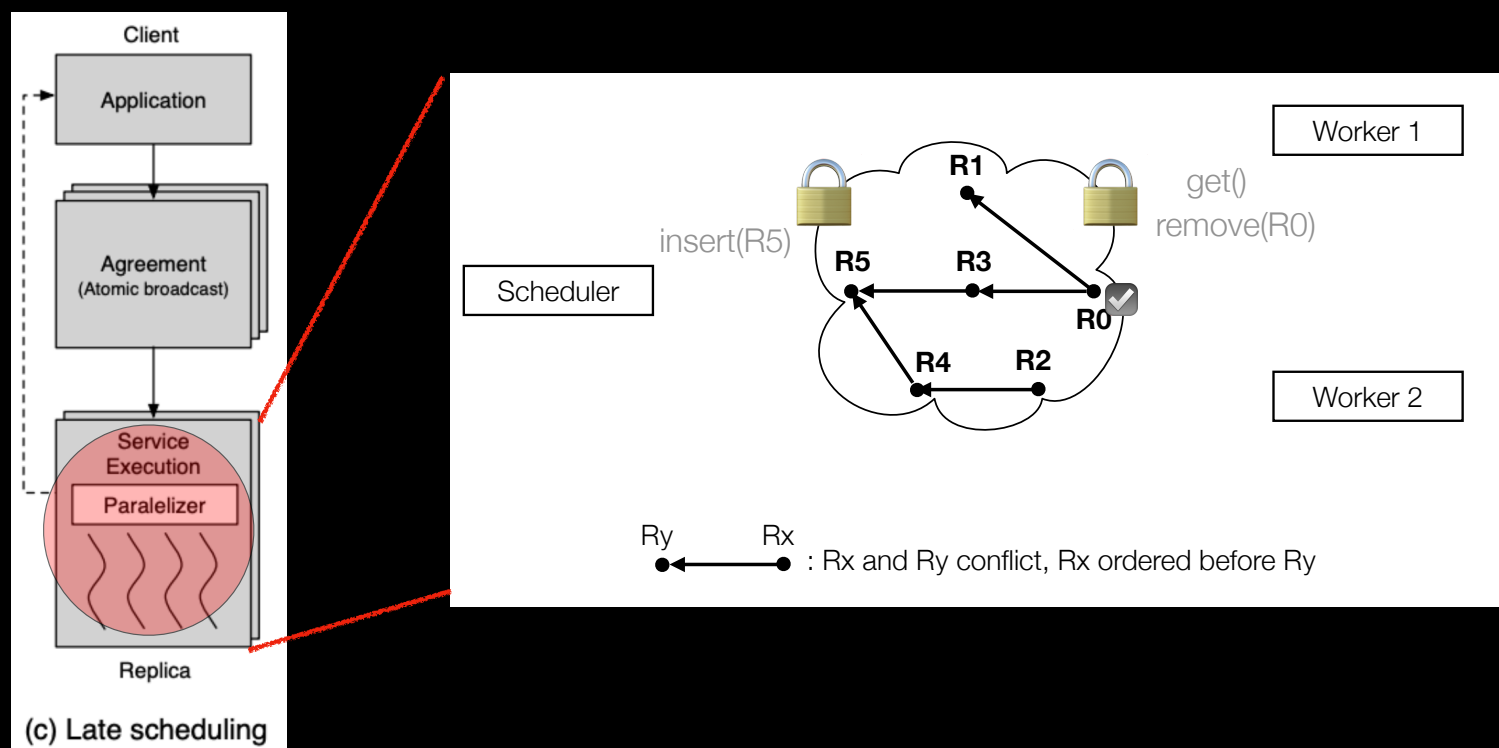
Tratamento de lotes



Odorico, Rudá, Dotti, Pedone - 2017 (IPDPS)

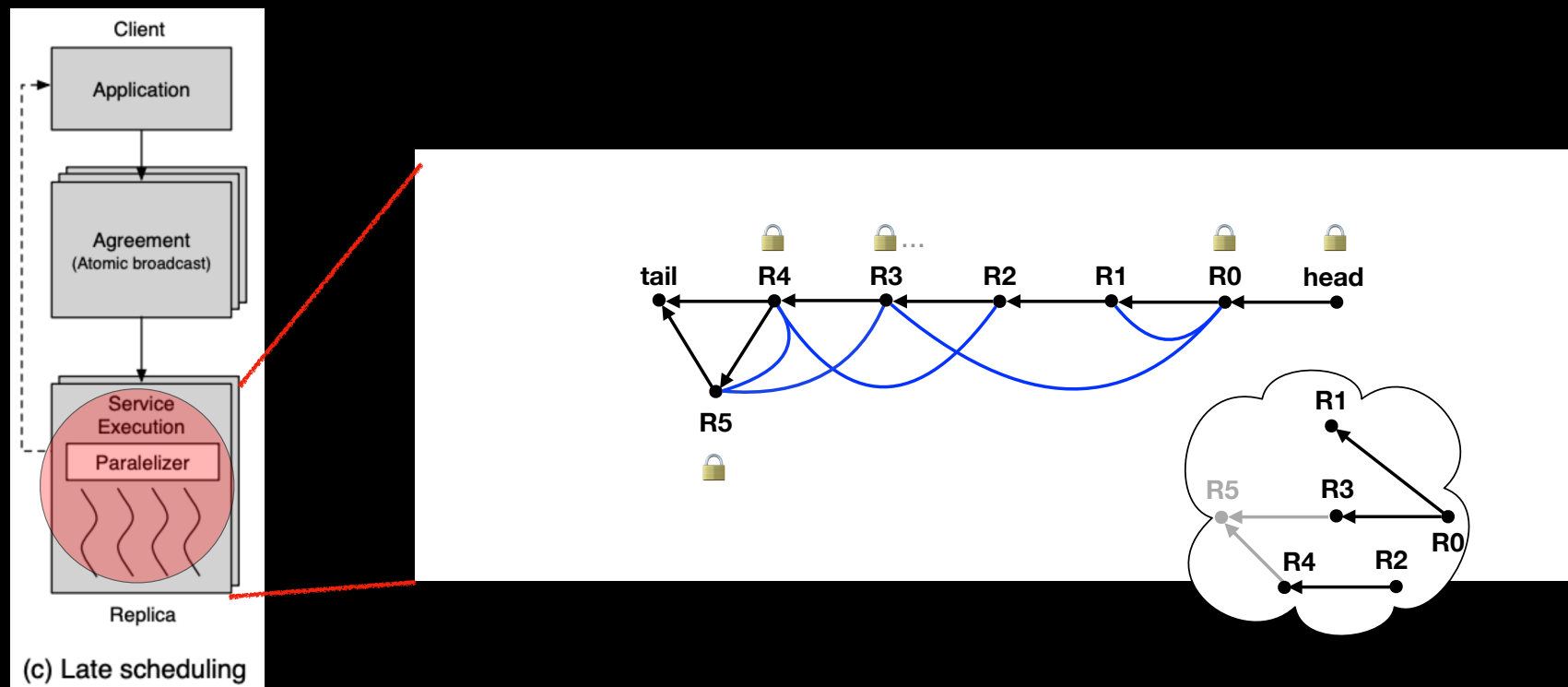
RME - Escalando Verticalmente

Grafo de dependências - **bloqueio de todo grafo** a cada operação



RME - Escalando Verticalmente

Grafo de dependências - **grão fino - bloqueio de nodos do grafo**

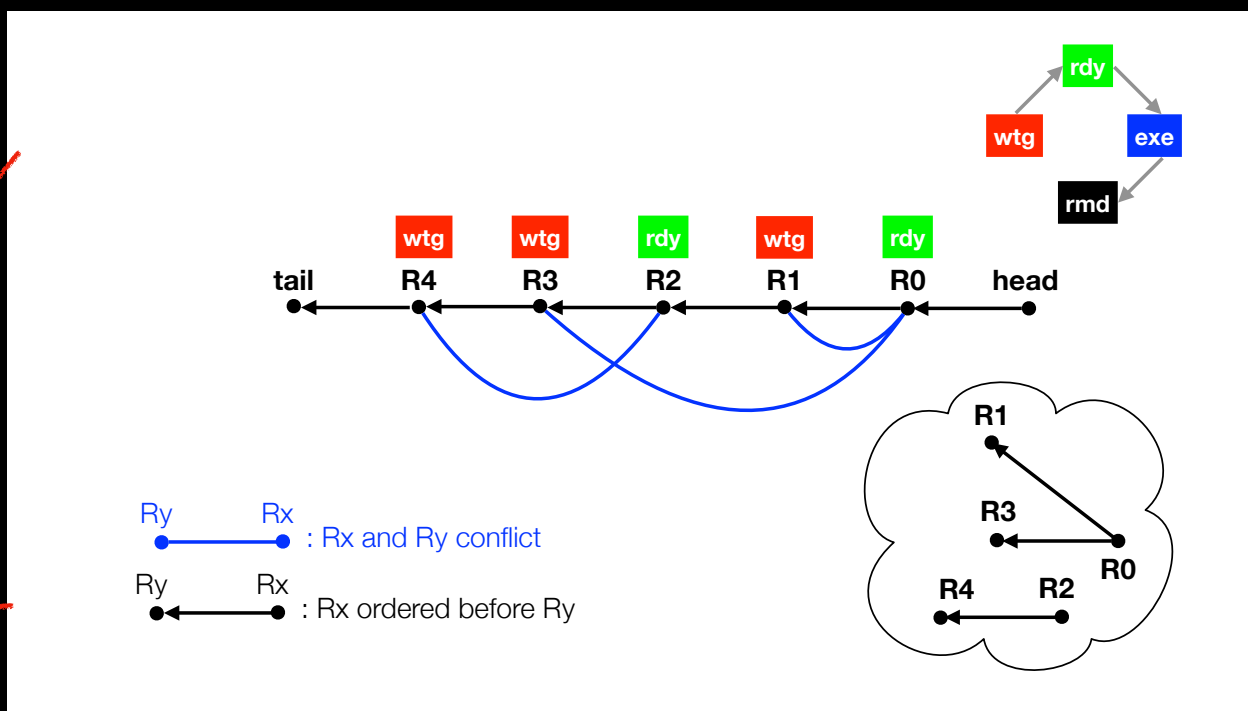
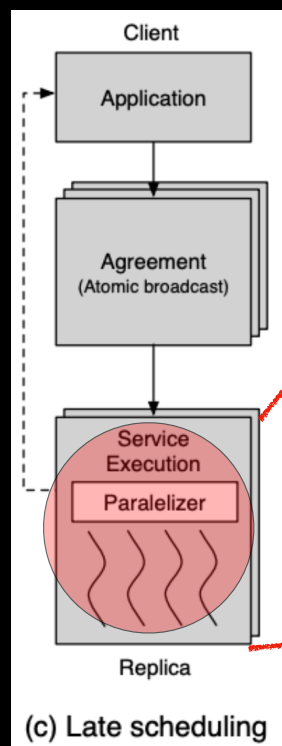


Ian Aragon Escobar, Alchieri, Dotti, Pedone:
Boosting concurrency in Parallel State Machine Replication. Middleware 2019



RME - Escalando Verticalmente

Grafo de dependências - **lock free!**



Ian Aragon Escobar, Alchieri, Dotti, Pedone:

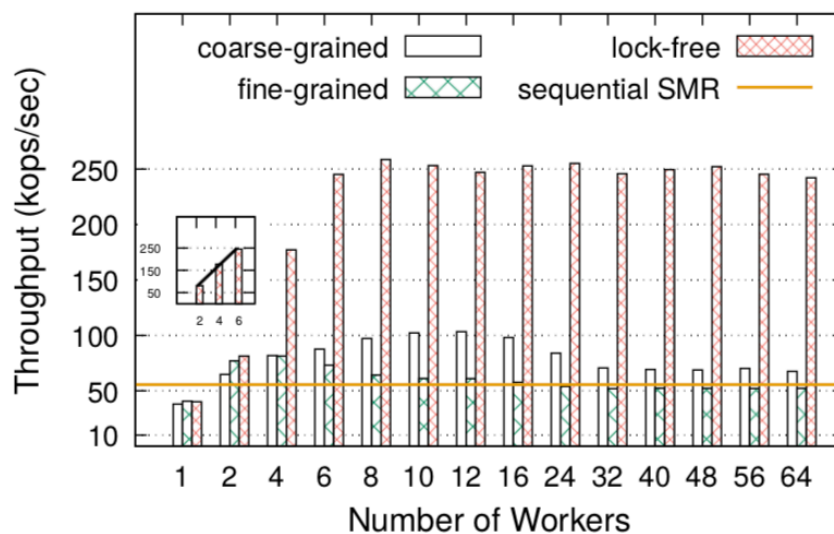
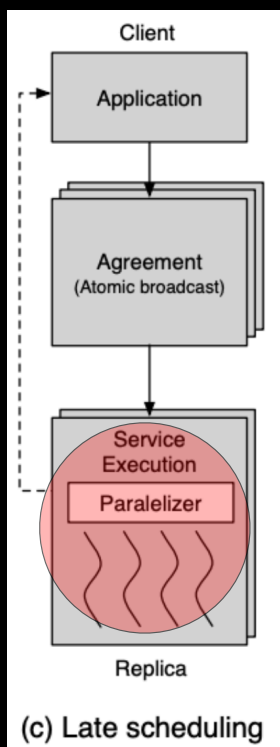
Boosting concurrency in Parallel State Machine Replication. Middleware 2019

(trabalho iniciou em prática em pesquisa no CC, depois IC de Ian)

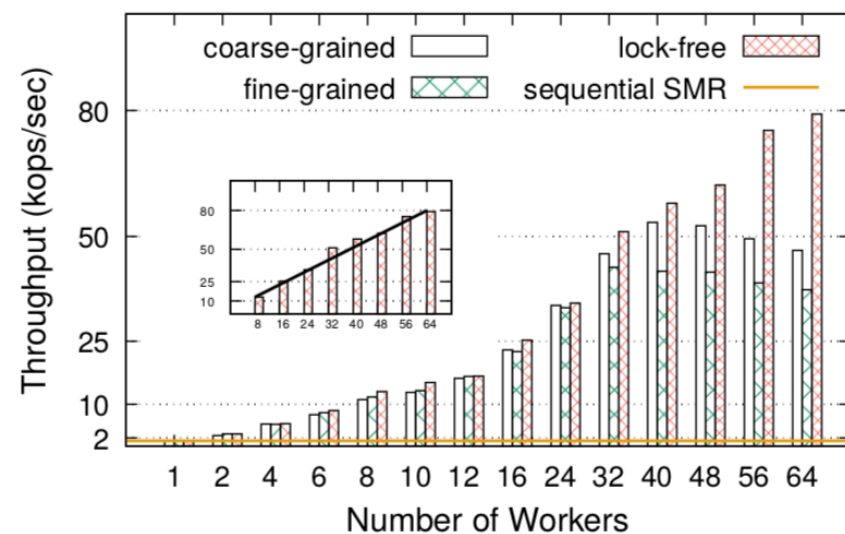


RME - Escalando Verticalmente

Grafo de dependências



Custo moderado de execução

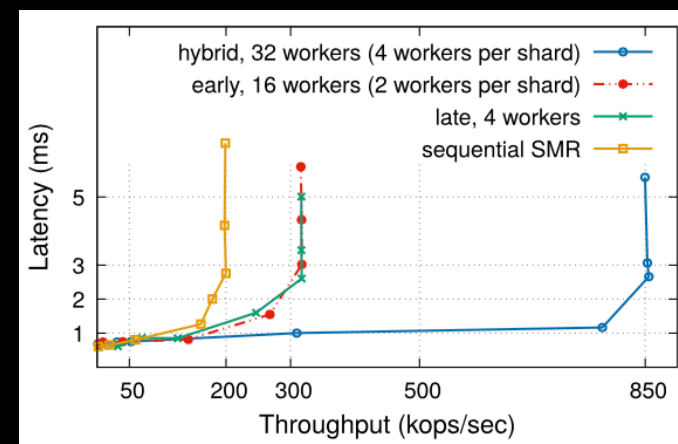
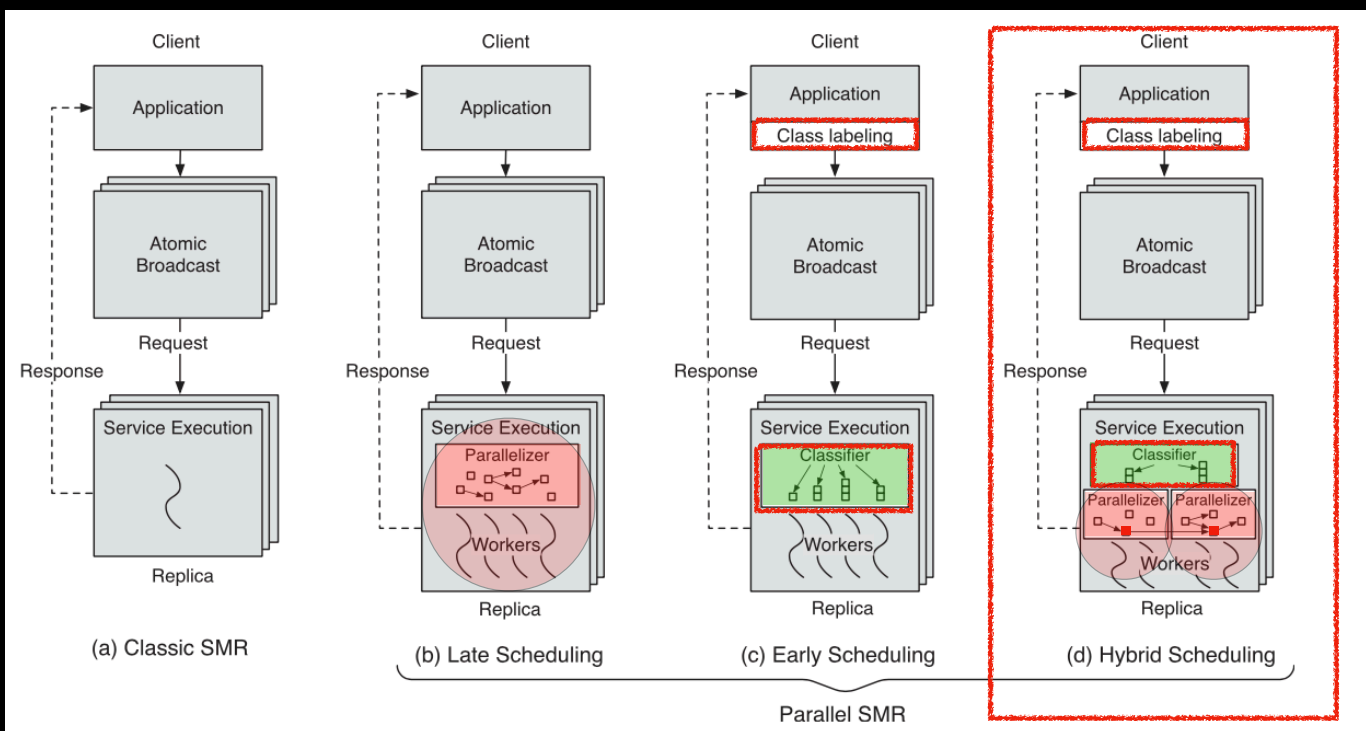


Custo alto de execução

Ian Aragon Escobar, Alchieri, Dotti, Pedone:
Boosting concurrency in Parallel State Machine Replication. Middleware 2019

RME - Escalando Verticalmente

Escalonamiento Híbrido = Antecipado + Tardio



Aldenio, Alchieri, Dotti, Pedone:
Exploiting Concurrency in Sharded Parallel State Machine Replication.
IEEE Trans. Parallel Distributed Syst. 33(9): 2133-2147 (2022)

RME - Escalando Verticalmente

Outros trabalhos

Alchieri, Dotti, Odorico, Pedone

Reconfiguring Parallel State Machine Replication - (SRDS), September 2017

Adaptação dinâmica do nível de concorrência à carga de trabalho

Tarcísio, Dotti, Pedone

Parallel State Machine Replication from Generalized Consensus - 2020 (SRDS)

Ordem parcial gerada por consenso generalizado é executada paralelamente

Erick, Dotti - E-Raft - Trabalho não publicado

Consenso Generalizado a partir de Raft

Vide palestra do Erick



RME - Escalando Verticalmente

Efeitos positivos

- disciplinas no PPGCC
 - estudo e desenvolvimento de estruturas de dados de baixa contenção levou a criar uma disciplina no PPGCC
Programação Concorrente, ou concorrência para Multi-Core
[estruturas concorrentes de baixa contenção]
 - Interesse por TLA+ levou a disciplina no PPGCC de
Construção de Algoritmos Distribuídos
- ... além de formação de pessoal, e das contribuições em si

RME - Escalando Verticalmente

PUCRS



Rudá

Tarcísio



Ian

Odorico



Dotti

Eliã



UNB



Aldenio



Alchieri

USI



Pedone

Parisa



2) RME - Escalando Horizontalmente

RME - Escalando Horizontalmente



RME - Escalando Horizontalmente

- Particionamento do estado

- *Sharding*



Partition Px



Partition Py



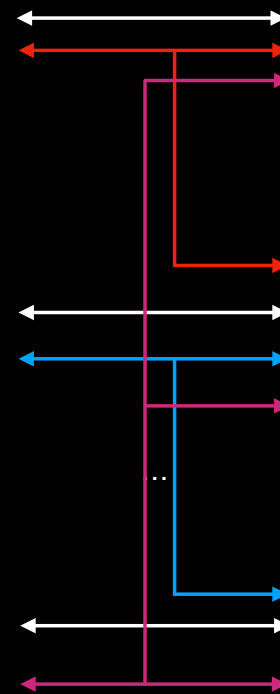
...



Partition P...



RME - Escalando Horizontalmente



Partition Px



Partition Py



Partition P...



RME - Escalando Horizontalmente

Particionamento

- Técnica fundamental para escalabilidade

Como realizar operações em múltiplas partições mantendo-as consistentes ?

RME - Escalando Horizontalmente

Particionamento

- Técnica fundamental para escalabilidade

Como realizar operações em múltiplas partições mantendo-as consistentes ?

Multicast Atômico

RME - Escalando Horizontalmente

- Validade, Acordo, Integridade
- **Ordem de Prefixo**
 - Mensagens aos mesmos nodos são entregues na mesma ordem
- **Ordem Acíclica**
 - A ordem de mensagens entregues em todos nodos é acíclica

Partition Px

m1 m3

Partition Py

m2 m1

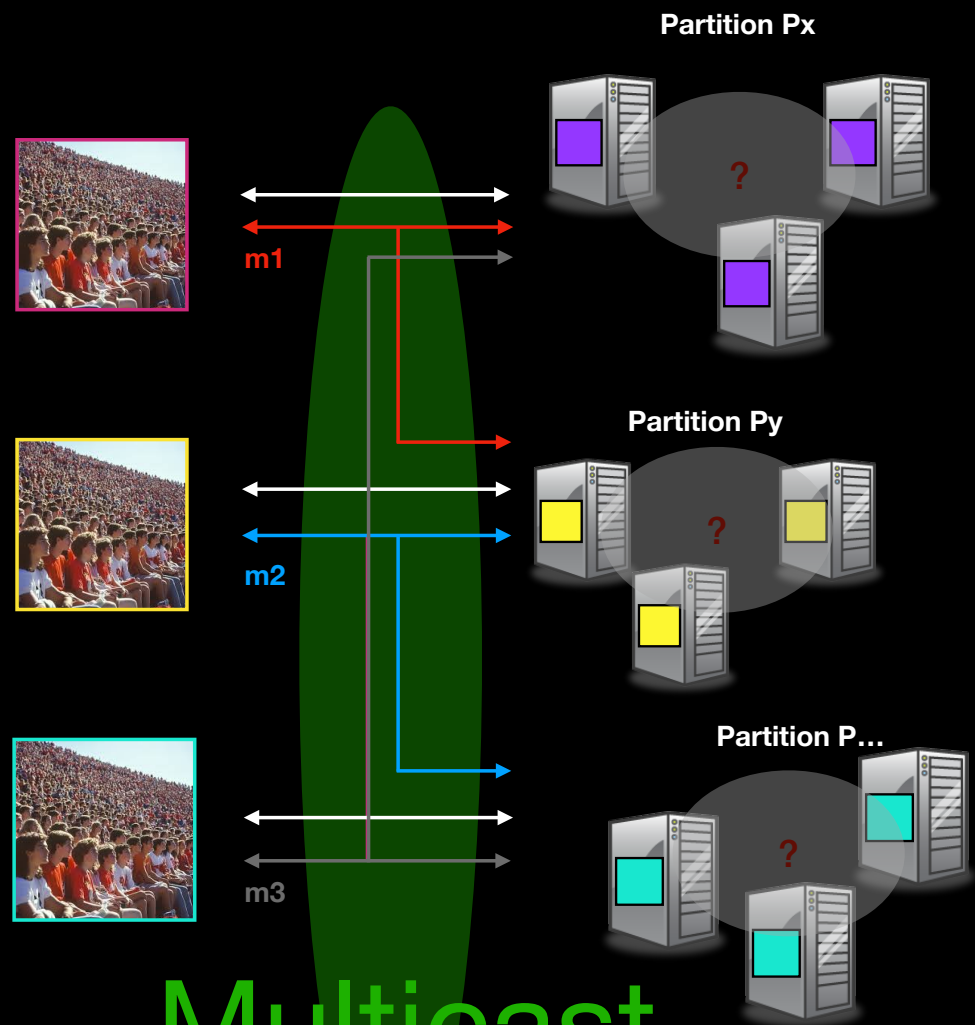
Partition P...

m3 m2

m1 m3

m2 m1

m2 m3



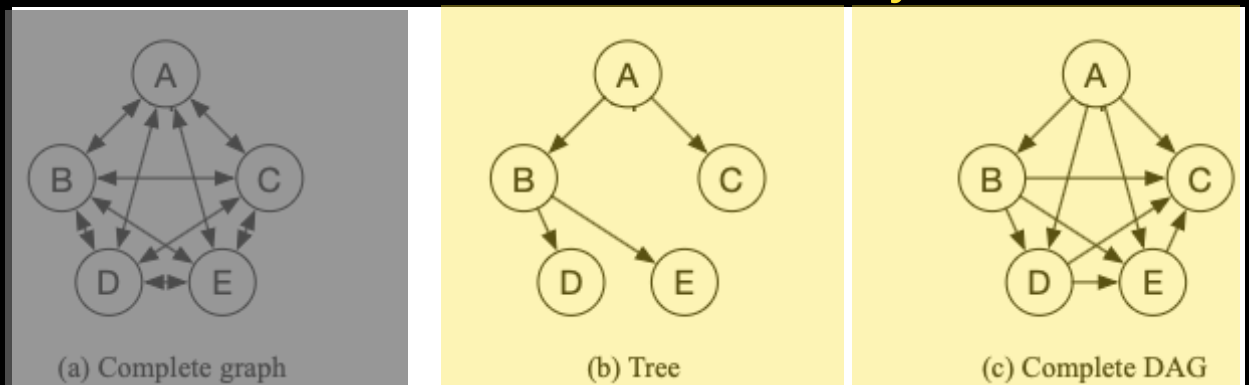
Multicast Atômico

RME - Escalando Horizontalmente

Multicast Atômico

- Nossas contribuições: uso de overlays
 - Diminui informação que cada nodo tem sobre o sistema
 - Reflete topologias reais onde conexões são limitadas
 - Permite algoritmos mais simples

Overlays

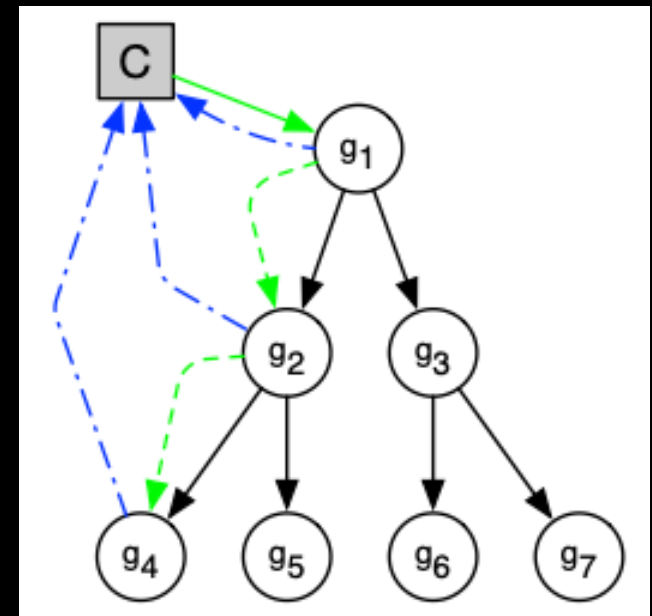


RME - Escalando Horizontalmente

Multicast Atômico

- ByzCast

- Tolerante a falhas Bizantinas
- Reuso de plataforma BFT
 - BFT-SMaRt
- Formação de overlay
 - **Árvore**
 - **Grupos são partições**
- Ordem total
 - Induzida pela hierarquia



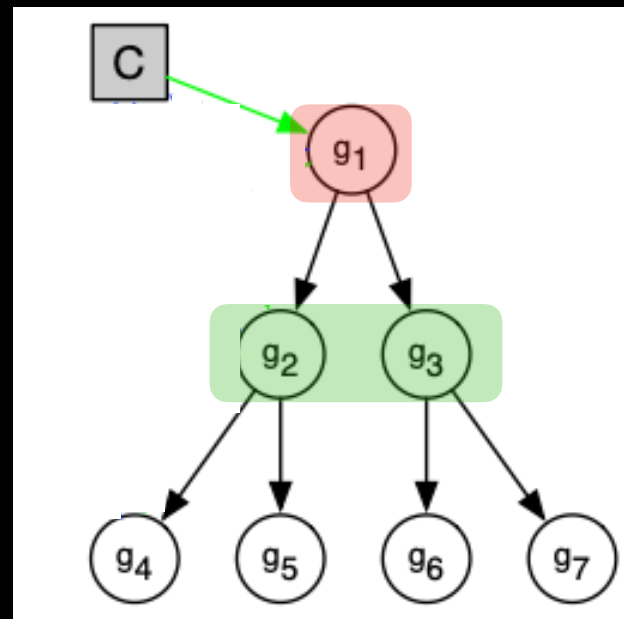
Paulo, Tarcisio, Alysson, Dotti, Pedone: DSN 2018
Byzantine Fault-Tolerant Atomic Multicast

RME - Escalando Horizontalmente

Multicast Atômico

- Árvore:
 - Multicast *parcialmente* genuíno

ByzCast



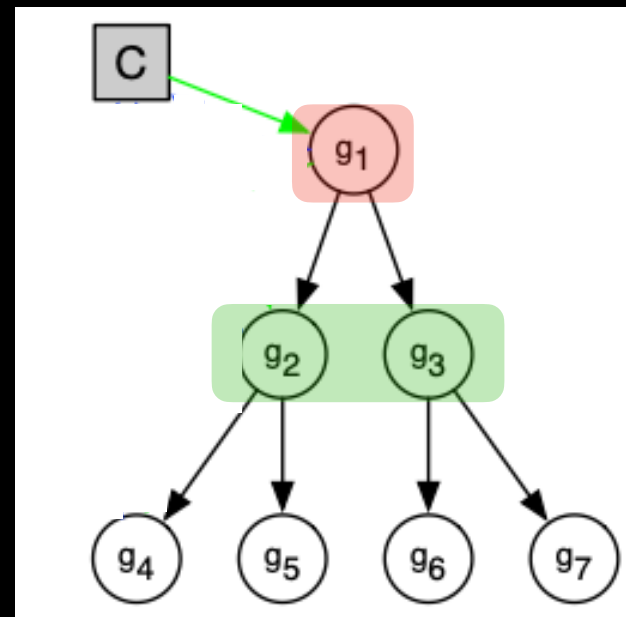
Paulo, Tarcisio, Alysson, Dotti, Pedone: DSN 2018
Byzantine Fault-Tolerant Atomic Multicast

RME - Escalando Horizontalmente

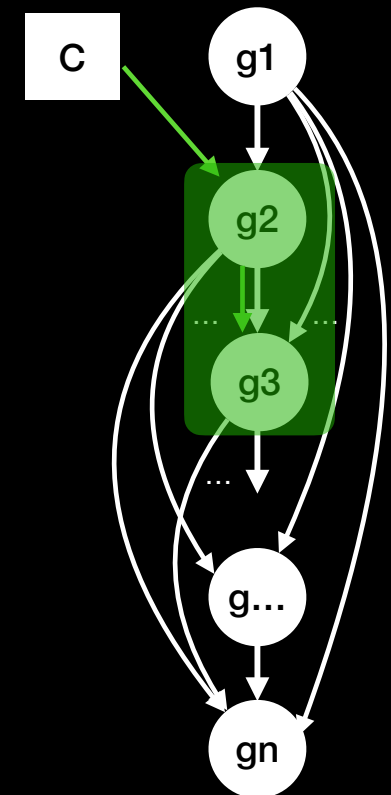
Multicast Atômico

- Árvore:
 - Multicast *parcialmente* genuíno
- Generalizar de árvore para GAD completo
 - FlexCast
 - Genuíno!

ByzCast



FlexCast



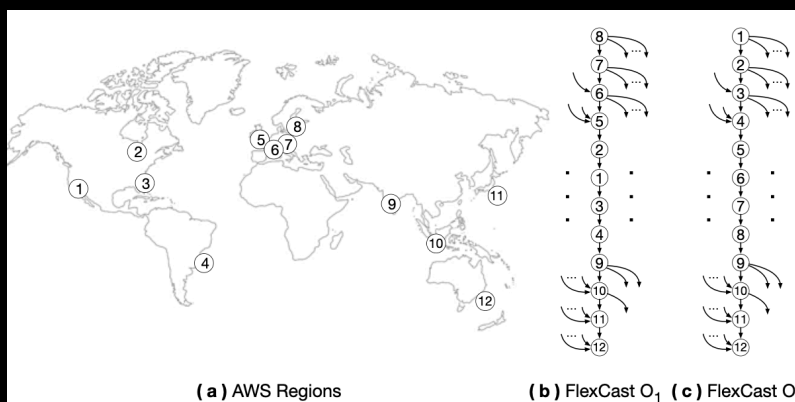
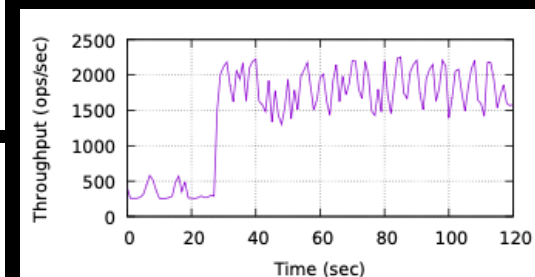
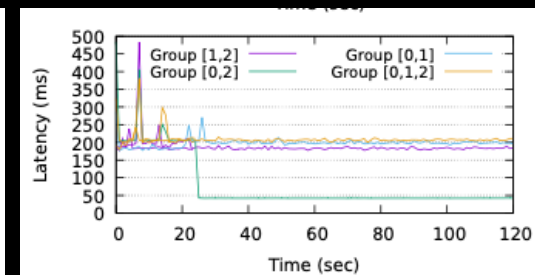
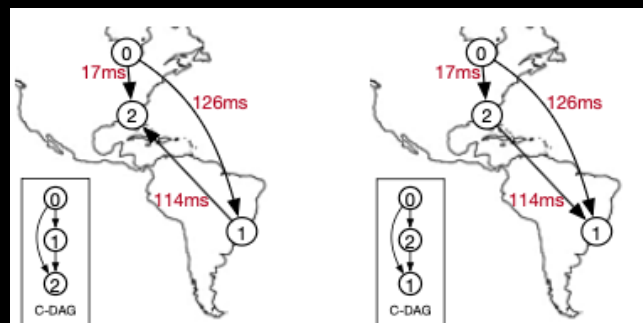
Eliã, Paulo, Alchieri, Dotti, Pedone: Middleware 2023

FlexCast: Genuine Overlay-based Atomic Multicast.

RME - Escalando Horizontalmente

Multicast Atômico

- FlexCast + Reconfiguração
 - Reconfiguração da topologia
 - Diminui latência entre nodos frequentemente endereçados juntos
 - Adapta-se ao workload
 - On-the-fly: mantém entregas

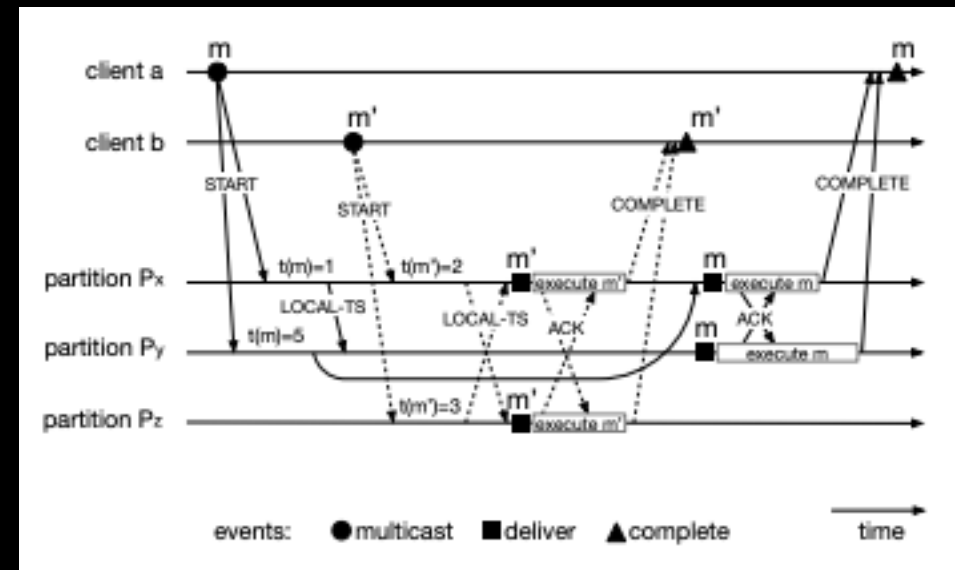


Eliã, Paulo, Alchieri, Dotti, Pedone: Reconfiguring Atomic Multicast.
Submetido a Journal em 2024 ...

RME - Escalando Horizontalmente

Multicast Atômico

- Linearizable atomic multicast
 - Difusão Atômica em RME
garante linearizabilidade
 - Multicast Atômico em RME particionada
não
garante linearizabilidade
- paper: razão, propostas, algoritmos
para garantir linearizabilidade em RME-Particionada



Pacheco, Dotti, Pedone: LADC 2022
posteriormente submetido a Journal em 2024 ...

RME - Escalando Horizontalmente

Multicast Atômico

- Muitos, excelentes autores
- Contribuições há décadas
- Ainda assim
 - Novos algoritmos, explorando novos aspectos, com melhor desempenho, tem surgido !!
- Vide palestra de Paulo!



RME - Escalando Horizontalmente

PUCRS

USI

UFU

UNB

Paulo



Paulo

Alchieri



Tarcísio



Pacheco



Universidade
de Lisboa

Eliã



Eliã

Pedone



Alysson



Dotti



Renan

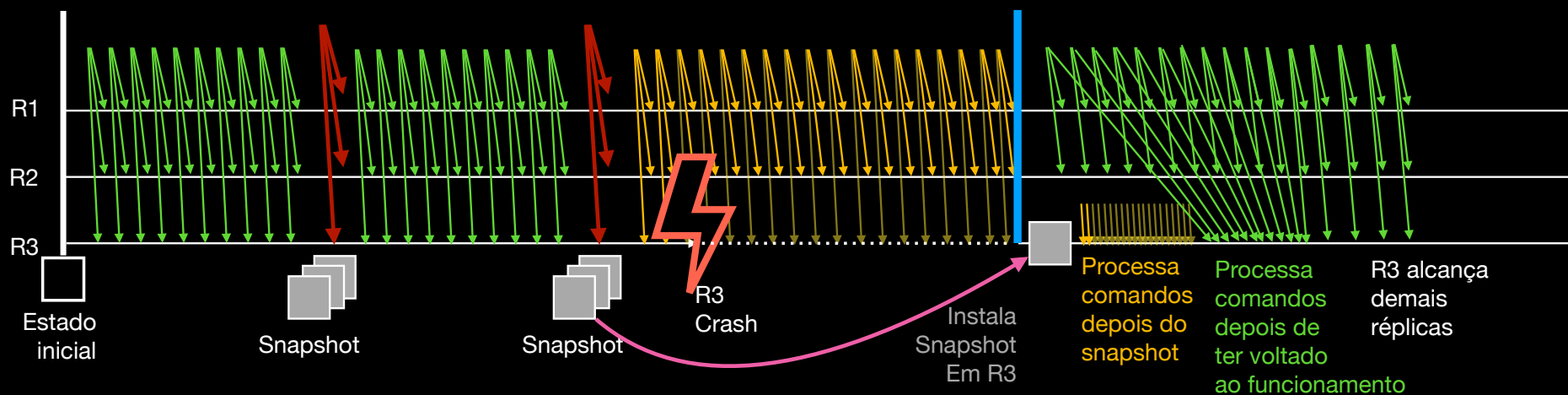


3) RME - Recuperar Rápidamente

RME - Recuperar Rapidamente

- Impactos no desempenho

- Durante **salvamento de estado**
 - vazão tipicamente baixa
- Durante **transferencia de estado**
 - demora - tamanho do estado
- Durante **Recuperação e execução do log**
 - quanto maior a vazão, maior o log
 - comandos entregues depois do retorno**
 - alcançar demais réplicas cfe. vazão



RME - Recuperar Rapidamente

Salvamento de Estado

Odorico, Parisa, Dotti, Pedone:

Checkpointing in Parallel State-Machine Replication. OPODIS 2014

- Checkpoint com coordenação ou sem coordenação de threads em RME paralela

Erick Pintor, Dotti: SBRC 2021

SMaRtTrie: Reducing Checkpoint's Impact in SMR Systems with a CTrie Data Structure.

- Avaliação de estrutura de dados específica para permitir snapshot consistente e concorrente (Trabalho de prática de pesquisa)

Everaldo, Alchieri, Dotti, Odorico: J. Internet Serv. Appl. 15(1): 194-211 (2024)

Reducing Persistence Overhead in Parallel State Machine Replication through Time-Phased Partitioned Checkpoint.

- Checkpoints em momentos diferentes, para partes diferentes, em réplicas diferentes

RME - Recuperar Rapidamente

Salvamento de Estado

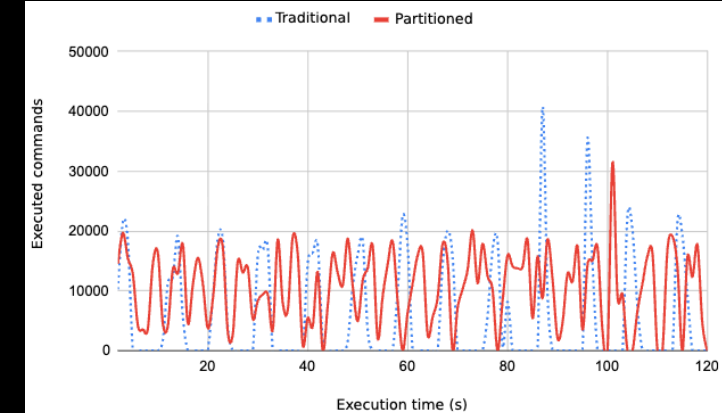
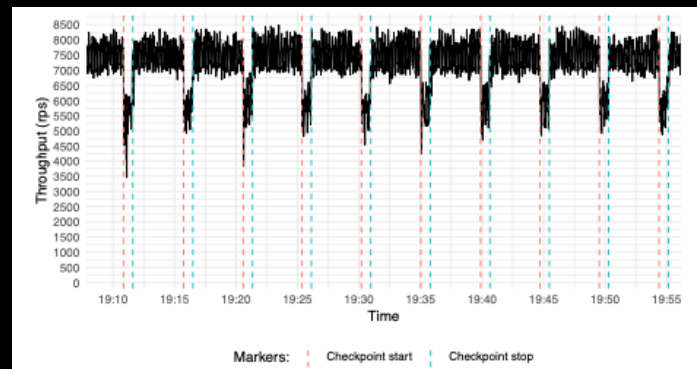
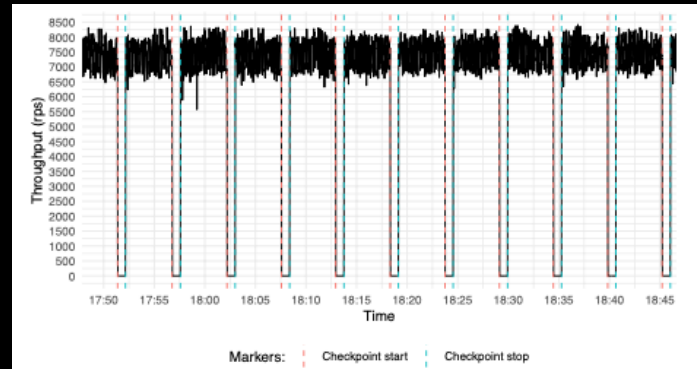
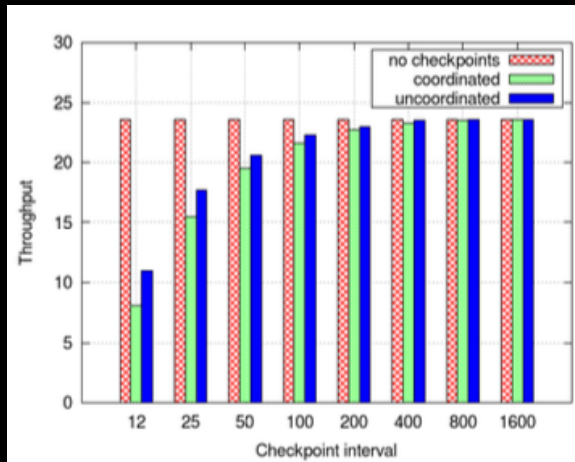


Figure 10. Throughput over time for a checkpoint interval of every 50,000 commands, using the workload 90r1c and 4 partitions.

RME - Recuperar Rapidamente

Transferência de estado e processamento do log

- Na recuperação:
 - Transfere estado: se acessado, sob demanda, em partes
 - Comandos novos processados imediatamente se não conflitam com log

Odorico, Dotti, Pedone: ICDCS 2017

High Performance Recovery for Parallel State Machine Replication.

- Redução do log

Luiz Gustavo Coutinho Xavier,

Dotti, Cristina, Odorico:

Shrinking Logs by Safely Discarding Commands. SBRC 2021

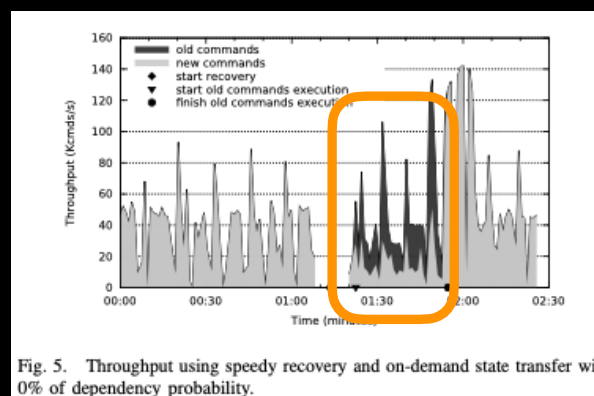


Fig. 5. Throughput using speedy recovery and on-demand state transfer with 0% of dependency probability.

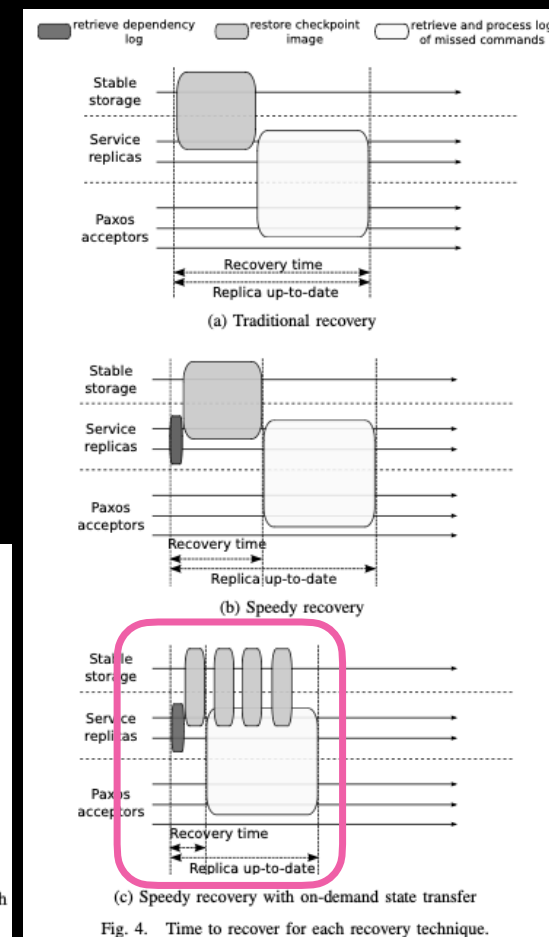


Fig. 4. Time to recover for each recovery technique.

RME - Recuperar Rapidamente

Em BFT

- Tendências
 - Sistemas com mais réplicas (ex.: blockchains)
 - Estado cresce em tamanho
- Necessidade de procedimentos para escala maior
 - Colaborativos, Paralelos
 - Estruturas auto-verificáveis
 - BFT
- Vide palestra do Eliã !
"SkipLists in SMR: Simplifying and Enhancing State Transfer and Validation"



RME - Recuperar Rapidamente

PUCRS

UFSC

UNB



Dotti

Odorico



Odorico

Eliã



Erick



Everaldo



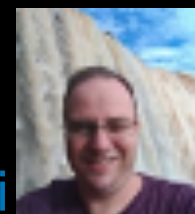
Luiz
Gustavo



Cristina M.



Alchieri



USI

Pedone



4) BFT e Blockchains

Blockchains

- Supõe **participantes** desconhecidos, **sem relação de confiança** entre si ... e ainda assim, **o sistema composto funciona de forma confiável**
- Para tal, usa mecanismos próprios:
 - incentivos (e.g. remuneração), provas,
 - “multas”, exclusão, ...
- Vide palestra de Enrique Fynn!
 - Perspectiva prática em PoW e PoS, smart-contracts



Blockchains

Como Replicação Máquina de Estados

Similaridades	RME	Blockchain
Estados	Estado	Cadeia de blocos
Nodos Computacionais	Réplicas	Proponentes/ Validadores
Acordo	Consenso	Decidir próximo bloco

Desafios	RME	Blockchain
Escala	Poucos nodos	Muitos
Reconfiguração	Por autoridade central	Descentralizada
Log	Mutável	Imutável, Auditável, Auto-verificável

Vide artigo

Alysson, Alchieri, João Sousa, André Oliveira, Pedone: DSN 2020
From Byzantine Replication to Blockchain: Consensus is only the Beginning

Blockchains

Processamento concorrente de transações de blocos

- Melhoria de vazão no processamento de transações
- Proposta: **Transações** de um bloco são **executadas concorrentemente**, se não conflitam
 - Adaptação de técnicas de escalonamento em RME para transações de um bloco

Aldenio Burgos, Alchieri, Dotti: LADC 2021

On the Performance of Using Parallel State Machine Replication to Implement Blockchains.

Blockchains

Processamento concorrente de transações de blocos

- Abordagens existentes
 - Proponente calcula DAG de dependências, envia com transações
 - Validadores confiam no DAG para executar transações
- Entretanto:
 - Proponente desonesto pode provocar atraso em validadores, informando dependências adicionais
- Como evitar ?
 - Conflitos verificáveis nos validadores
 - Verificadores devem chegar ao mesmo DAG do proponente
 - Incentivos e punições

Jefferson P. da Silva, Alchieri, Dotti, Pedone:

Parallel Execution of Transactions Based on Dynamic and Self-Verifiable Conflict Analysis. LADC 2023: 110-119

Blockchains

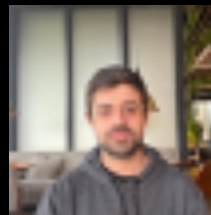
Consenso + Gossip

- Algoritmos de consenso podem usar difusão estilo gossip (e.g. Tendermint)
 - Consenso tem redundância:
não precisa de todo quorum para prosseguir
 - Gossip tem redundância:
mensagens chegam repetidas aos nodos
 - É possível diminuir a redundância,
melhorar desempenho, e
manter resiliência ?

- Vide palestra do Ricardo

Ricardo, Daniel, Nenad, Dotti, Pedone (et.al):

Byzantine Consensus and Semantic Gossip: Scaling Decentralized Systems - em submissão



Blockchains

Avaliação de desempenho

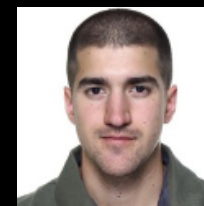
- Dada a busca por desempenho em blockchains
 - É importante avaliar o desempenho de forma
 - Sistemática
 - Extraíndo métricas significativas
- Vide palestra do Lucca -
Iniciativa conjunta Informal/PUCRS para avaliar o
desempenho do protocolo de consenso Tendermint



Consenso BFT

Diferentes suposições da camada de comunicação

- Consenso sobre camada de comunicação com um serviço seguro de ordenação de mensagens
 - **Palestra do Alchieri !!**
- Observações levam a possibilidade de suposições temporais adicionais em algoritmos de consenso, explorando estas em novos algoritmos
 - **Palestra do Nenad !!**



Blockchains

PUCRS

USI

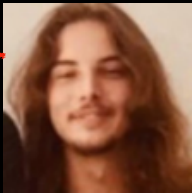
UNB



Dotti



Ricardo



Lucca



Pedone



Nenad



Daniel



Alchieri



Aldenio



Jefferson

Nosso Workshop

Hoje

14:40 "Uma Carreira Consistente em Computação Distribuída"

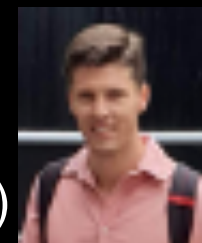
Lasaro Camargos



15:10 Coffee-break

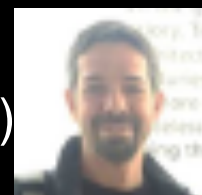
15:40 "Atomic Multicast: do Skeen ao Pacheco"

Paulo Coelho (UFU)



16:10 "Composing State Machine Replicas"

Odorico Mendizabal (UFSC)



16:40 "A different form of consensus: Proof of Work & Proof of Stake"

Enrique Fynn (Chorus One)



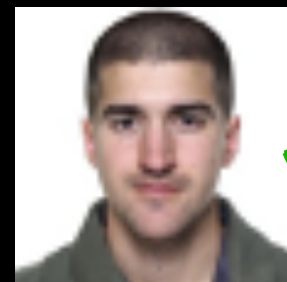
Nosso Workshop

16 de Abril

9:00 "Cloud-of-Clouds Storage: from Theory to Production"
Alysson Bessani (Universidade de Lisboa)



9:30 "How Far Can Synchronous BFT Consensus Go?"
Nenad Milošević (USI - Informal Systems)



10:00 "Consenso Bizantino Baseado em uma Camada de Rede
com Ordenação de Mensagens Tolerante a Intrusões"
Eduardo Alchieri (UFSC)



10:30 Coffee Break

Nosso Workshop

16 de Abril

11:00 As Múltiplas Faces de um Sistema Distribuído

Erick Pintor



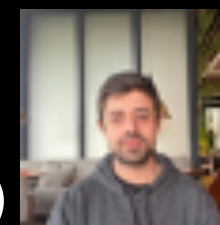
11:15 "SkipLists in SMR: Simplifying and Enhancing State Transfer and Validation"

Eliã Batista (USI / PUCRS)



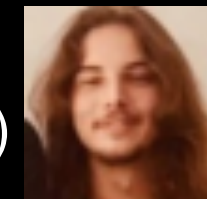
11:30 "Byzantine Consensus and Semantic Gossip: Scaling Decentralized Systems."

Ricardo Guimarães (PUCRS)



11:45 "Performance Evaluation of a Consensus Algorithm"

Lucca Dornelles Cezar (PUCRS)



A Colaboração

Suporte

- CAPES -
Pesquisador Visitante Especial



- CNPq - Projetos Universais



- Capes - Pesquisador Gaúcho



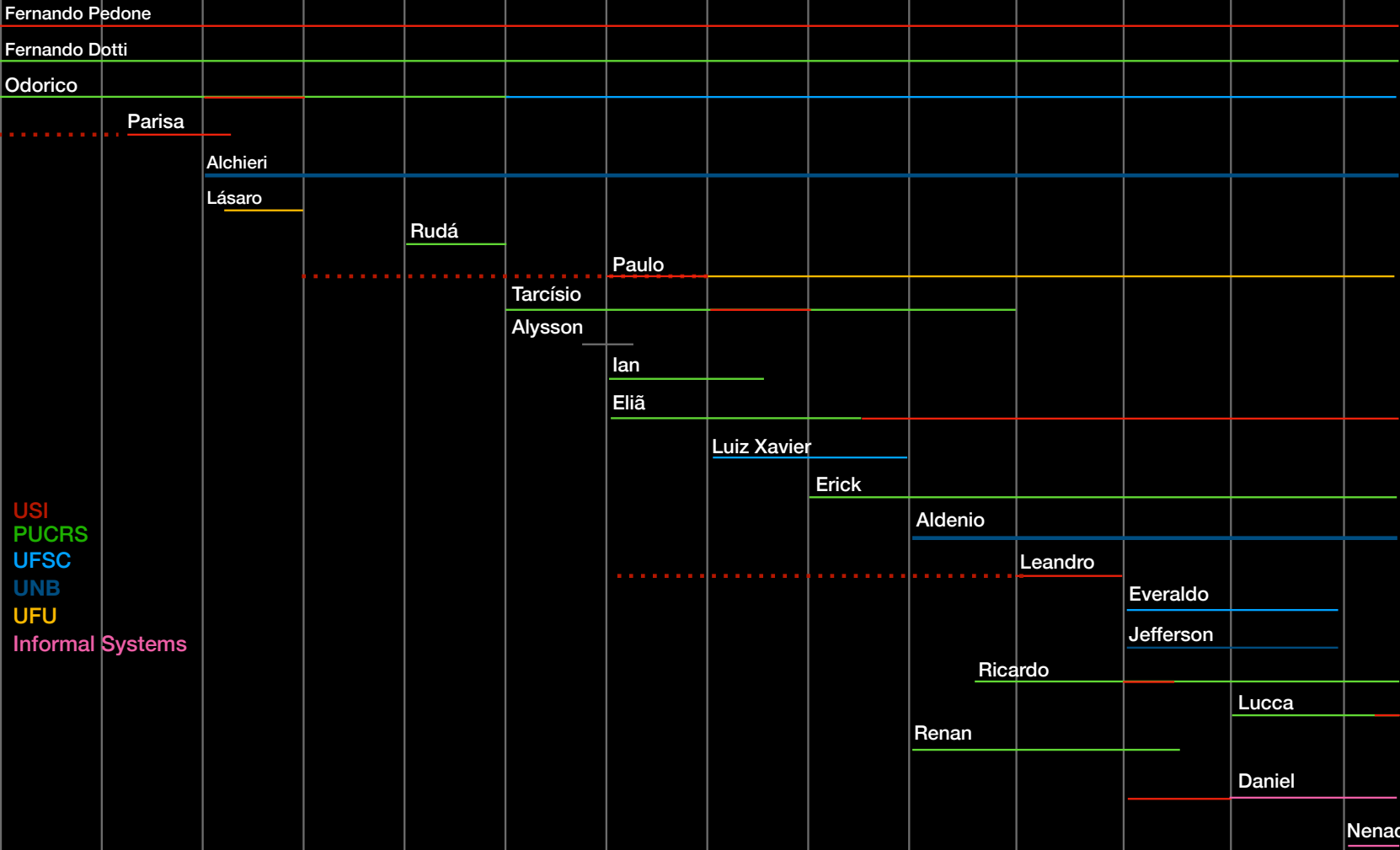
- USI/Suíça - Programa MARS



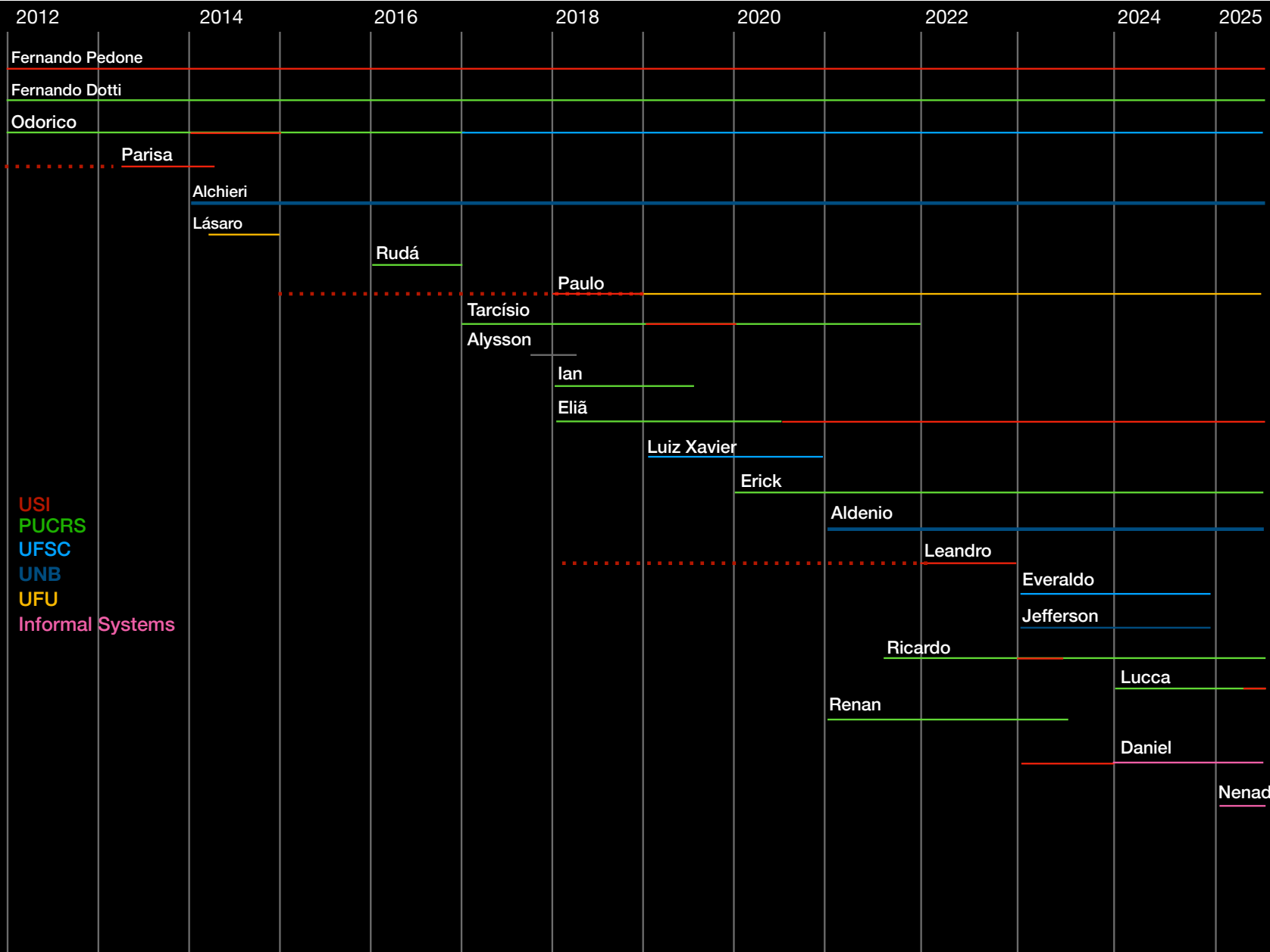
- Governo da Suíça -
Bolsas de Doutorado SW



2012 2014 2016 2018 2020 2022 2024 2025



USI
PUCRS
UFSC
UNB
UFU
Informal Systems



**Quase 100
anos de
cooperação
em
computação
distribuída**



Obrigado!

fernando **dotti**

fernando.dotti @ pucrs.br

fldotti.github.io

PUCRS Pontifícia Universidade Católica do Rio Grande do Sul, Brazil

