Serverless Dataflows: ...

Diogo Alexandre Ferreira de Jesus

Thesis to obtain the Master of Science Degree in

Computer Science and Engineering

Supervisor: Luis Manuel Antunes Veiga

Examination Committee

Chairperson: Prof. Name of the Chairperson
Supervisor: Luis Manuel Antunes Veiga
Member of the Committee: Prof. Name of First Committee Member

October 2025

Declaration
I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to thank my parents for their friendship, encouragement and caring over all these years, for always being there for me through thick and thin and without whom this project would not be possible. I would also like to thank my grandparents, aunts, uncles and cousins for their understanding and support throughout all these years.

Quisque facilisis erat a dui. Nam malesuada ornare dolor. Cras gravida, diam sit amet rhoncus ornare, erat elit consectetuer erat, id egestas pede nibh eget odio. Proin tincidunt, velit vel porta elementum, magna diam molestie sapien, non aliquet massa pede eu diam. Aliquam iaculis.

Fusce et ipsum et nulla tristique facilisis. Donec eget sem sit amet ligula viverra gravida. Etiam vehicula urna vel turpis. Suspendisse sagittis ante a urna. Morbi a est quis orci consequat rutrum. Nullam egestas feugiat felis. Integer adipiscing semper ligula. Nunc molestie, nisl sit amet cursus convallis, sapien lectus pretium metus, vitae pretium enim wisi id lectus.

Donec vestibulum. Etiam vel nibh. Nulla facilisi. Mauris pharetra. Donec augue. Fusce ultrices, neque id dignissim ultrices, tellus mauris dictum elit, vel lacinia enim metus eu nunc.

I would also like to acknowledge my dissertation supervisors Prof. Some Name and Prof. Some Other Name for their insight, support and sharing of knowledge that has made this Thesis possible.

Last but not least, to all my friends and colleagues that helped me grow as a person and were always there for me during the good and bad times in my life. Thank you.

To each and every one of you – Thank you.

Abstract

Serverless computing has become a suitable cloud paradigm for many applications, prized for its operational ease, automatic scalability, and fine-grained pay-per-use pricing model. However, executing workflows, which are compositions of multiple tasks, in Function-as-a-Service (FaaS) environments remains inefficient. This inefficiency stems from the stateless nature of functions, and a heavy reliance on external services for intermediate data transfers and inter-function communication.

In this document, we introduce a decentralized DAG engine that leverages historical metadata to plan and influence task scheduling. Our solution encompasses metadata management, static workflow planning, and a worker-level scheduling strategy designed to drive workflow execution with minimal synchronization. We compare our scheduling approach against WUKONG, another decentralized serverless DAG engine. Our evaluation demonstrates that utilizing historical information significantly improves performance and reduces resource utilization for workflows running on serverless platforms.

Keywords

Maecenas tempus dictum libero; Donec non tortor in arcu mollis feugiat;Cras rutrum pulvinar tellus.

Resumo

A computação serverless tornou-se um paradigma de nuvem adequado para muitas aplicações, valorizado pela sua facilidade operacional, escalabilidade automática e modelo de preços granular baseado na utilização. Contudo, a execução de workflows, que são composições de múltiplas tarefas, em ambientes Function-as-a-Service (FaaS) permanece ineficiente. Esta ineficiência resulta da natureza *stateless* (sem estado) destas funções e de uma forte dependência de serviços externos para transferências de dados intermédios e comunicação entre funções.

Neste documento, apresentamos um motor de workflows serverless descentralizado que utiliza métricas recolhidas durante a execução para planear e influenciar o *scheduling* de tarefas. A nossa solução abrange a gestão de metadados, o planeamento estático de workflows e uma estratégia de *scheduling* ao nível dos workers concebida para conduzir a execução de workflows de uma forma descentralizada e com sincronização mínima. Comparamos a nossa abordagem com o WUKONG, outro motor de workflows serverless descentralizado. A nossa avaliação demonstra que a utilização de informação histórica melhora significativamente o desempenho e reduz a utilização de recursos para workflows executados em plataformas serverless.

Palavras Chave

Cloud Computing; Serverless; FaaS; Serverless Workflows; Serverless DAGs; Metadata; Workflow Prediction

Contents

List of Figures

List of Tables

List of Algorithms

Listings

Acronyms

1

Introduction

Contents

Function-as-a-Service (FaaS) represents a serverless cloud computing paradigm that simplifies application deployment by abstracting away infrastructure management. It provides automatic, elastic scalability—potentially without limit—along with a fine-grained, pay-per-use pricing model. This has led to its widespread adoption for event-driven systems, microservices, and web services on platforms like AWS Lambda[1], Azure Functions[2], and Google Cloud Functions[3]. These applications typically benefit the most from FaaS because they are lightweight, stateless, and characterized by highly variable or unpredictable workloads, allowing them to leverage serverless platforms' on-demand scalability and cost-efficiency.

This paradigm is also increasingly used to execute complex scientific and data processing workflows,

[1] https://aws.amazon.com/pt/lambda/
[2] https://azure.microsoft.com/en-us/products/functions
[3] https://cloud.google.com/functions

such as the Cybershake [1] seismic hazard analysis or Montage [2], an astronomy image mosaicking workflow. These applications are structured as workflows—formally represented as Directed Acyclic Graphs (DAGs) of interdependent tasks. However, efficiently executing these complex workflows on serverless platforms remains a significant challenge.

Despite their advantages, serverless platforms present several limitations that complicate the execution of complex workflows. Since these platforms allow scaling down to zero resources to save costs, they can also introduce unpredictable latency, known as *cold starts* [3], particularly for short-lived functions, affecting overall workflow performance. The lack of *direct inter-function communication* [4] means that tasks often have to rely on external services, such as message brokers or databases to exchange intermediate data, which can increase overhead and reduce efficiency. Interoperability between platforms is further limited by the use of platform-specific workflow definition languages, which restricts the portability of workflows across different serverless environments. Additionally, while statelessness simplifies scaling and management, it can introduce overhead and complexity for applications that require continuity or coordination across multiple function invocations. Finally, developers have limited control over the underlying infrastructure, restricting the ability to optimize resource usage or tune performance for specific workloads.

Several solutions have emerged to address the limitations of serverless platforms. Stateful functions (e.g., AWS Step Functions[4], Azure Durable Functions[5], and Google Cloud Workflows[6]) expand the range of applications that can run on serverless platforms by maintaining state across multiple function invocations, coordinating complex workflows, and providing built-in fault tolerance. Other approaches tackle limitations at the runtime level, proposing extensions to FaaS platforms (e.g., Faa$T [5], Palette [6], Lambdata [7]) or entirely new serverless architectures (e.g., Apache OpenWhisk [8]). Finally, some workflow-focused solutions (e.g., WUKONG [9], Unum [10], DEWEv3 [11]) employ scheduling strategies and workflow-level optimizations to enhance efficiency, primarily by improving data locality to bring computation closer to the data and minimize reliance on external services.

These workflow-focused approaches, however, often use uniform resources for workers and rely on "one-step scheduling," making decisions based solely on the immediate workflow stage without considering the broader context or the downstream effects of their decisions. This combination of homogeneous worker configurations and limited scheduling foresight can lead to inefficient use of resources when tasks have diverse computational or memory requirements. Furthermore, the heuristic-based approaches used by other solutions can be inefficient in certain scenarios, as they lack mechanisms to adapt worker resource allocations to the specific needs of individual tasks. Moreover, we found no prior work that leverages metadata or historical metrics to inform scheduling decisions across an entire

[4] https://aws.amazon.com/pt/step-functions/
[5] https://learn.microsoft.com/en-us/azure/azure-functions/durable/durable-functions-overview?tabs=in-process%2Cnodejs-v3%2Cv1-model&pivots=csharp
[6] https://cloud.google.com/workflows

serverless workflow.

This limitation motivates the central research question of this work: if we have knowledge of the computation steps, collect sufficient metrics on their behavior, and understand how they are composed to form the full workflow, can we leverage this information to make smarter scheduling decisions that minimize makespan and maximize resource efficiency in a FaaS environment?

To answer this research question, we propose a decentralized serverless workflow execution engine that leverages historical metadata from previous workflow runs to generate informed task allocation plans, which are then executed by FaaS workers in a choreographed manner, without needing a central scheduler. By relying on such planning, our approach aims to minimize the usage of external cloud storage services, which are often employed by similar solutions for intermediate data exchange and synchronization, while also avoiding the inefficiencies of homogeneous worker resource allocations.

The main contributions of this work are as follows:

- Analysis of the serverless workflow orchestration research landscape;
- Propose a decentralized serverless workflow execution engine that overcomes the "one-step scheduling" and uniform-resource limitations of existing workflow-focused solutions by leveraging historical metadata to generate informed execution plans;
- Demonstrate how incorporating historical execution data can improve task allocation, reduce reliance on external cloud storage services, and enhance overall workflow efficiency on FaaS platforms.

1.1 Problem/Motivation

1.2 Gaps in prior work

1.3 Proposed Solution

1.4 Document Organization

2

Related Work

Contents

2.1 Serverless Computing

2.1.1 Platforms

2.1.2 Use Cases

2.1.3 Limitations

2.2 Workflow Execution

2.3 Relevant Related Systems

2.3.1 Serverless Workflow Scheduling

2.4 Discussion/Analysis

3

Architecture

Contents

3.1 Workflow Definition Language

3.2 Overview

3.3 Metadata Management

3.4 Static Workflow Planning

3.4.1 Simulation Layer

3.4.2 Planners

3.4.3 Optimizations

3.5 Decentralized Scheduling

Algorithm 1 Worker Assignment Algorithm

```
Require: nodes, predictions, base_rc, SLA, MAX_CLUSTERING
assigned ← ∅
                                                ▷ nodes are topologically sorted
for all n ∈ nodes do
    if n ∈ assigned then
        continue
    end if
    if n.upstream = ∅ then                                          ▷ root nodes
        roots ← {r ∈ nodes | r.upstream = ∅ ∧ r ∉ assigned}
        AssignGroup(null, roots)
    else if |n.upstream| = 1 then                              ▷ 1→1 or 1→N
        u ← n.upstream[0]
        if |u.downstream| = 1 then
            AssignWorker([n], u.worker)                         ▷ reuse worker
        else                                                         ▷ 1→N
            fanout ← {d ∈ u.downstream | d ∉ assigned}
            AssignGroup(u.worker, fanout)
        end if
    else                                                             ▷ N→1
        outputs ← {u.worker : predictions.output_size(u) | u ∈ n.upstream}
        best ← arg max_{w∈outputs} outputs[w]
        AssignWorker([n], best)
    end if
end for

function AssignGroup(up_worker, tasks)
    if tasks = ∅ then return
    end if
    exec_t ← {t : predictions.exec_time(t) | t ∈ tasks}
    out_sz ← {t : predictions.output_size(t) | t ∈ tasks}
    median ← Median(exec_t.values())
    longs ← {t ∈ tasks | exec_t[t] > median}
    shorts ← SortLargerOutputFirst({t ∈ tasks | exec_t[t] ≤ median})

                 ▷ 1) cluster short tasks with bigger outputs on upstream worker
    if up_worker ≠ null ∧ shorts ≠ ∅ then
        cluster ← shorts[0 : MAX_CLUSTERING]
        AssignWorker(cluster, up_worker)
        shorts ← shorts[MAX_CLUSTERING :]
    end if

           ▷ 2) pair long tasks with remaining short tasks (1 long per group)
    while longs ≠ ∅ ∧ shorts ≠ ∅ do
        cluster ← [longs[0]] + shorts[0 : MAX_CLUSTERING − 1]
        worker_id ← NewWorkerId
        AssignWorker(cluster, worker_id)
        longs ← longs[1 :]
        shorts ← shorts[MAX_CLUSTERING − 1 :]
    end while

                                           ▷ 3) group remaining short tasks
    while shorts ≠ ∅ do
        worker_id ← NewWorkerId
        AssignWorker(shorts[0 : MAX_CLUSTERING], worker_id)
        shorts ← shorts[MAX_CLUSTERING :]
    end while

                                     ▷ 4) group remaining longs (half-size)
    half ← max(1, ⌊MAX_CLUSTERING/2⌋)
    while longs ≠ ∅ do
        worker_id ← NewWorkerId
        AssignWorker(longs[0 : half], worker_id)
        longs ← longs[half :]
    end while
end function
```

Algorithm 2 Resource Downgrading Algorithm

Require: $dag, nodes, critical\_path\_ids, original\_cp\_time, configs, predictions$

```
workers_outside ← ∅

                                  ▷ 1) Identify workers outside the critical path
for all n ∈ nodes do                              ▷ nodes are topologically sorted
    wid ← n.worker_id
    if n.id ∉ critical_path_ids ∧ ∀cp ∈ dag.critical_path_nodes : wid ≠ cp.worker_id then
        workers_outside ← workers_outside ∪ {wid}
    end if
end for
nodes_outside_cp ← {n ∈ nodes | n.id ∉ critical_path_ids}

                      ▷ 2) Attempt downgrade for each worker outside critical path
for all wid ∈ workers_outside do
    last_good_rc ← {n.id : n.config | n ∈ nodes_outside_cp ∧ n.worker_id = wid}

        ▷ Iterate through weaker configurations (skip strongest at index 0)
    for i ← 1 to |configs| − 1 do
        trial ← configs[i].CLONE(wid)

                        ▷ Apply trial configuration to all nodes of this worker
        for all n ∈ nodes_outside_cp do
            if n.worker_id = wid then
                n.config ← trial
            end if
        end for

                              ▷ Recompute workflow timing with predictions
        cp_time ← SIMULATECRITICALPATHTIME(dag)

        if cp_time = original_cp_time then
                        ▷ Downgrade acceptable, record as last good state
            for all n ∈ nodes_outside_cp do
                if n.worker_id = wid then
                    last_good_rc[n.id] ← n.config
                end if
            end for
        else
        ▷ Downgrade increases critical path, revert and move on to the next worker
            for all n ∈ nodes_outside_cp do
                if n.worker_id = wid then
                    n.config ← last_good_rc[n.id]
                end if
            end for
            break                                        ▷ move to next worker
        end if
    end for
end for
```

4

Evaluation

Contents

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Morbi commodo, ipsum sed pharetra gravida, orci magna rhoncus neque, id pulvinar odio lorem non turpis. Nullam sit amet enim. Suspendisse id velit vitae ligula volutpat condimentum. Aliquam erat volutpat. Sed quis velit. Nulla facilisi. Nulla libero. Vivamus pharetra posuere sapien. Nam consectetuer. Sed aliquam, nunc eget euismod ullamcorper, lectus nunc ullamcorper orci, fermentum bibendum enim nibh eget ipsum. Donec porttitor ligula eu dolor. Maecenas vitae nulla consequat libero cursus venenatis. Nam magna enim, accumsan eu, blandit sed, blandit a, eros.

4.1 Maecenas vitae nulla consequat

Aliquam aliquet, est a ullamcorper condimentum, tellus nulla fringilla elit, a iaculis nulla turpis sed wisi. Fusce volutpat. Etiam sodales ante id nunc. Proin ornare dignissim lacus. Nunc porttitor nunc a sem. Sed sollicitudin velit eu magna. Aliquam erat volutpat. Vivamus ornare est non wisi. Proin vel quam.

Vivamus egestas. Nunc tempor diam vehicula mauris. Nullam sapien eros 4.1, facilisis vel, eleifend non, auctor dapibus, pede.

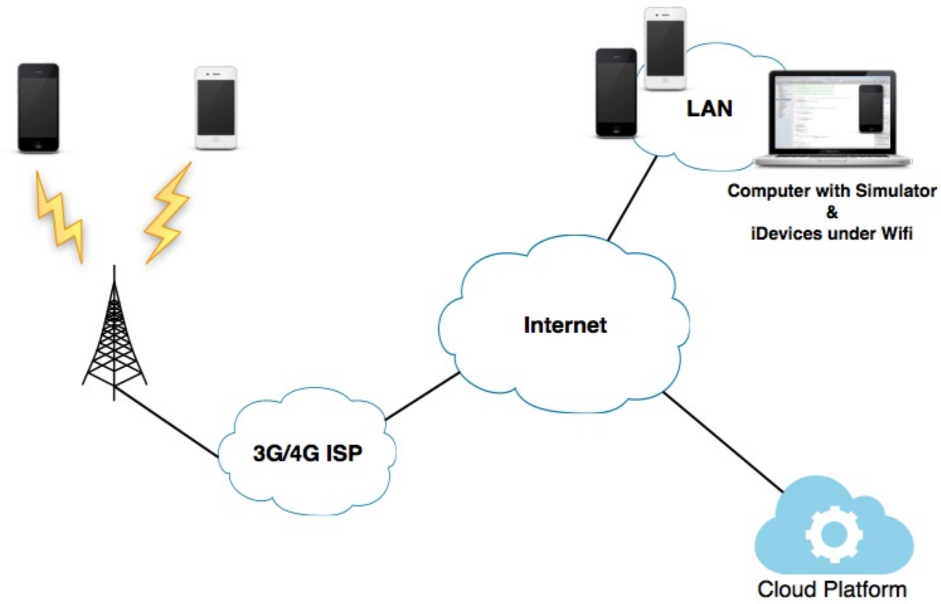


Figure 4.1: Test Environment

Aliquam aliquet, est a ullamcorper condimentum, tellus nulla fringilla elit, a iaculis nulla turpis sed wisi. Fusce volutpat. Etiam sodales ante id nunc. Proin ornare dignissim lacus. Nunc porttitor nunc a sem. Sed sollicitudin velit eu magna. Aliquam erat volutpat. Vivamus egestas. Nunc tempor diam vehicula mauris. Nullam sapien eros, facilisis vel, eleifend non, auctor dapibus, pede 4.1 used in the tests. The Network Link Conditioner allows to force/simulate fluctuations in fixed network segments.

Table 4.1: Network Link Conditioner Profiles

Network Profile	Bandwidth	Packets Droped	Delay
Wifi	40 mbps	0%	1 ms
3G	780 kbps	0%	100 ms
Edge	240 kbps	0%	400 ms

Aliquam aliquet, est a ullamcorper condimentum, tellus nulla fringilla elit, a iaculis nulla turpis sed wisi. Fusce volutpat. Etiam sodales ante id nunc. Proin ornare dignissim lacus. Nunc porttitor nunc a sem. Sed sollicitudin velit eu magna. Aliquam erat volutpat. Vivamus ornare est non wisi. Proin vel quam. Vivamus egestas. Nunc tempor diam vehicula mauris. Nullam sapien eros, facilisis vel, eleifend non, auctor dapibus, pede.

4.2 Proin ornare dignissim lacus

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et sapien ullamcorper pharetra. Vestibulum erat wisi, condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentum, elit eget tincidunt condimentum, eros ipsum rutrum orci, sagittis tempus lacus enim ac dui. Donec non enim in turpis pulvinar facilisis. Ut felis.

Et “optimistic” nulla dui purus, eleifend vel, consequat non, dictum porta, nulla. Duis ante mi, laoreet ut, commodo eleifend, cursus nec, lorem. Aenean eu est. Etiam imperdiet turpis. Praesent nec augue. Curabitur ligula quam, rutrum id, tempor sed, consequat ac, dui G_j, nec ligula et lorem consequat ullamcorper p ut mauris eu mi mollis luctus j, porttitor ut, 4.1, uctus posuere justo:

N_j Is the number of times peer j has been optimistically unchoked.

n_j Among the N_j unchokes, the number of times that peer j responded with unchoke or supplied segments to peer p.

$C_{r[j]}$ The cooperation ratio of peer j. If peer j never supplied peer p, the information of $C_{r[j]}$ may not be available.

$C_{r(max)}$ The maximum cooperation ratio of peer p's neighbors, i.e., $C_{r(max)} = max(C_r)$.

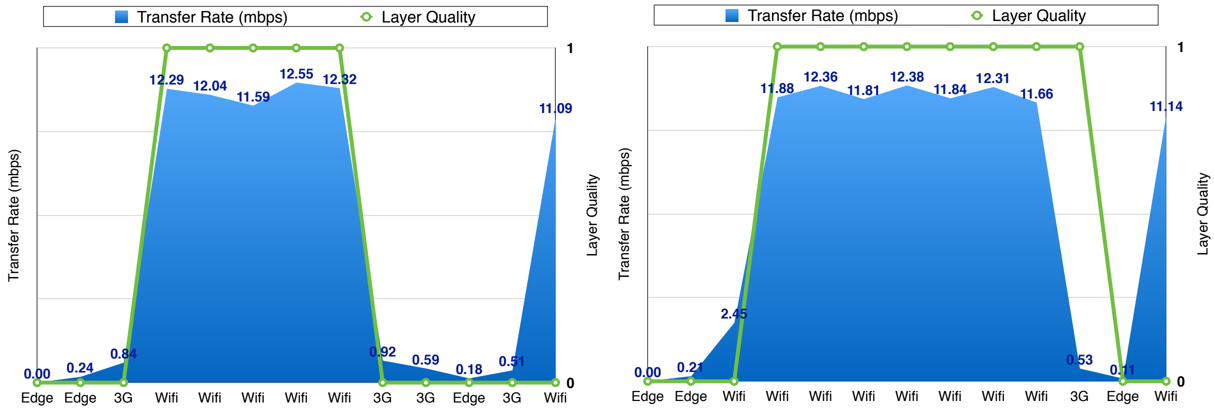
$$G_j = \begin{cases} \dfrac{n_j C_{r[j]}}{N_j} & \text{if } n_j > 0 \\ \dfrac{C_{r(max)}}{N_j + 1} & \text{if } n_j = 0 \end{cases} \tag{4.1}$$

Cursus $C_{r(max)}$ conubia nostra, per inceptos hymenaeos j gadipiscing mollis massa $N_j = 0$, unc ut dui eget nulla venenatis aliquet $G_j = C_{r(max)}$.

Vestibulum accumsan eros nec magna. Vestibulum vitae dui. Vestibulum nec ligula et lorem consequat ullamcorper. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Phasellus eget nisl ut elit porta ullamcorper. Maecenas tincidunt velit quis orci. Sed in dui. Nullam ut mauris eu mi mollis luctus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Sed cursus cursus velit. Sed a massa.

Both **??** Phasellus eget nisl ut elit porta “perfect” tincidunt. Class aptent taciti sociosqu ad litora torquent per conubia nostra.

Cras sed ante. Phasellus in massa. Curabitur dolor eros, gravida et, hendrerit ac, cursus non, massa. Aliquam lorem. In hac habitasse platea dictumst. Cras eu mauris. Quisque lacus. Donec ipsum. Nullam vitae sem at nunc pharetra ultricies. Vivamus elit eros, ullamcorper a, adipiscing sit amet, porttitor ut, nibh. Maecenas adipiscing mollis massa. Nunc ut dui eget nulla venenatis aliquet.



(a) Adaptation System Test 4

(b) Adaptation System Test 5

Figure 4.2: Adaptation System Behavior Test

Sed luctus posuere justo. Cras vehicula varius turpis. Vivamus eros metus, tristique sit amet, molestie dignissim, malesuada et, urna.

5

Conclusion

Contents

Pellentesque vel dui sed orci faucibus iaculis. Suspendisse dictum magna id purus tincidunt rutrum. Nulla congue. Vivamus sit amet lorem posuere dui vulputate ornare. Phasellus mattis sollicitudin ligula. Duis dignissim felis et urna. Integer adipiscing congue metus.

Rui Cruz
You should always start a Chapter with an introductory text

5.1 Conclusions

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Morbi commodo, ipsum sed pharetra gravida, orci magna rhoncus neque, id pulvinar odio lorem non turpis. Nullam sit amet enim. Suspendisse id velit vitae ligula volutpat condimentum. Aliquam erat volutpat. Sed quis velit. Nulla facilisi. Nulla libero. Vivamus pharetra posuere sapien. Nam consectetuer. Sed aliquam, nunc eget euismod ullamcorper, lectus nunc ullamcorper orci, fermentum bibendum enim nibh eget ipsum. Donec porttitor ligula eu dolor. Maecenas vitae nulla consequat libero cursus venenatis. Nam magna enim, accumsan eu, blandit sed, blandit a, eros.

Quisque facilisis erat a dui. Nam malesuada ornare dolor. Cras gravida, diam sit amet rhoncus ornare, erat elit consectetuer erat, id egestas pede nibh eget odio. Proin tincidunt, velit vel porta elementum, magna diam molestie sapien, non aliquet massa pede eu diam. Aliquam iaculis. Fusce et ipsum et nulla tristique facilisis. Donec eget sem sit amet ligula viverra gravida. Etiam vehicula urna vel turpis. Suspendisse sagittis ante a urna. Morbi a est quis orci consequat rutrum. Nullam egestas feugiat felis. Integer adipiscing semper ligula. Nunc molestie, nisl sit amet cursus convallis, sapien lectus pretium metus, vitae pretium enim wisi id lectus. Donec vestibulum. Etiam vel nibh. Nulla facilisi. Mauris pharetra. Donec augue. Fusce ultrices, neque id dignissim ultrices, tellus mauris dictum elit, vel lacinia enim metus eu nunc.

Proin at eros non eros adipiscing mollis. Donec semper turpis sed diam. Sed consequat ligula nec tortor. Integer eget sem. Ut vitae enim eu est vehicula gravida. Morbi ipsum ipsum, porta nec, tempor id, auctor vitae, purus. Pellentesque neque. Nulla luctus erat vitae libero. Integer nec enim. Phasellus aliquam enim et tortor. Quisque aliquet, quam elementum condimentum feugiat, tellus odio consectetuer wisi, vel nonummy sem neque in elit. Curabitur eleifend wisi iaculis ipsum. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. In non velit non ligula laoreet ultrices. Praesent ultricies facilisis nisl. Vivamus luctus elit sit amet mi. Phasellus pellentesque, erat eget elementum volutpat, dolor nisl porta neque, vitae sodales ipsum nibh in ligula. Maecenas mattis pulvinar diam. Curabitur sed leo.

Nulla facilisi. In vel sem. Morbi id urna in diam dignissim feugiat. Proin molestie tortor eu velit. Aliquam erat volutpat. Nullam ultrices, diam tempus vulputate egestas, eros pede varius leo, sed imperdiet lectus est ornare odio. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Proin consectetuer velit in dui. Phasellus wisi purus, interdum vitae, rutrum accumsan, viverra in, velit. Sed enim risus, congue non, tristique in, commodo eu, metus. Aenean tortor mi, imperdiet id, gravida eu, posuere eu, felis. Mauris sollicitudin, turpis in hendrerit sodales, lectus ipsum pellentesque ligula, sit amet scelerisque urna nibh ut arcu. Aliquam in lacus. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nulla placerat aliquam wisi. Mauris viverra odio. Quisque fermentum pulvinar odio. Proin posuere est vitae ligula. Etiam euismod. Cras a eros.

Nunc auctor bibendum eros. Maecenas porta accumsan mauris. Etiam enim enim, elementum sed, bibendum quis, rhoncus non, metus. Fusce neque dolor, adipiscing sed, consectetuer et, lacinia sit amet, quam.

5.2 System Limitations and Future Work

Aliquam aliquet, est a ullamcorper condimentum, tellus nulla fringilla elit, a iaculis nulla turpis sed wisi. Fusce volutpat. Etiam sodales ante id nunc. Proin ornare dignissim lacus. Nunc porttitor nunc a sem.

Sed sollicitudin velit eu magna. Aliquam erat volutpat. Vivamus ornare est non wisi. Proin vel quam. Vivamus egestas. Nunc tempor diam vehicula mauris. Nullam sapien eros, facilisis vel, eleifend non, auctor dapibus, pede.

Bibliography

[1] R. Graves, T. H. Jordan, S. Callaghan, E. Deelman, E. Field, G. Juve, C. Kesselman, P. Maechling, G. Mehta, K. Milner, D. Okaya, P. Small, and K. Vahi, “Cybershake: A physics-based seismic hazard model for southern california,” *Pure and Applied Geophysics*, vol. 168, no. 3, pp. 367–381, 2011. [Online]. Available: https://doi.org/10.1007/s00024-010-0161-6

[2] J. C. Jacob, D. S. Katz, G. B. Berriman, J. C. Good, A. Laity, E. Deelman, C. Kesselman, G. Singh, M.-H. Su, T. Prince, and R. Williams, “Montage: A grid portal and software toolkit for science-grade astronomical image mosaicking,” *International Journal of Computational Science and Engineering*, vol. 4, no. 2, pp. 73–87, 2009. [Online]. Available: https://doi.org/10.1504/IJCSE.2009.026999

[3] M. Golec, G. K. Walia, M. Kumar, F. Cuadrado, S. S. Gill, and S. Uhlig, “Cold start latency in serverless computing: A systematic review, taxonomy, and future directions,” *ACM Computing Surveys*, vol. 57, no. 3, pp. 1–36, 2024.

[4] J. M. Hellerstein, J. Faleiro, J. E. Gonzalez, J. Schleier-Smith, V. Sreekanti, A. Tumanov, and C. Wu, “Serverless computing: One step forward, two steps back,” *arXiv preprint arXiv:1812.03651*, 2018.

[5] F. Romero, G. I. Chaudhry, I. n. Goiri, P. Gopa, P. Batum, N. J. Yadwadkar, R. Fonseca, C. Kozyrakis, and R. Bianchini, “Faa$t: A transparent auto-scaling cache for serverless applications,” in *Proceedings of the ACM Symposium on Cloud Computing*, ser. SoCC ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 122–137. [Online]. Available: https://doi.org/10.1145/3472883.3486974

[6] M. Abdi, S. Ginzburg, C. Lin, J. M. Faleiro, I. Goiri, G. I. Chaudhry, R. Bianchini, D. S. Berger, and R. Fonseca, “Palette load balancing: Locality hints for serverless functions,” in *EuroSys*. ACM, May 2023. [Online]. Available: https://www.microsoft.com/en-us/research/publication/palette-load-balancing-locality-hints-for-serverless-functions/

[7] Y. Tang and J. Yang, “Lambdata: Optimizing serverless computing by making data intents explicit,” in *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*. IEEE, 2020, pp. 294–303.

[8] Apache openwhisk. [Online]. Available: https://openwhisk.apache.org/

[9] B. Carver, J. Zhang, A. Wang, A. Anwar, P. Wu, and Y. Cheng, “Wukong: A scalable and locality-enhanced framework for serverless parallel computing,” in *Proceedings of the 11th ACM symposium on cloud computing*, 2020, pp. 1–15.

[10] D. H. Liu, A. Levy, S. Noghabi, and S. Burckhardt, “Doing more with less: Orchestrating serverless applications without an orchestrator,” in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, 2023, pp. 1505–1519.

[11] Q. Jiang, Y. C. Lee, and A. Y. Zomaya, “Serverless execution of scientific workflows,” in *International Conference on Service-Oriented Computing*. Springer, 2017, pp. 706–721.

A

Code of Project

Nulla dui purus, eleifend vel, consequat non, dictum porta, nulla. Duis ante mi, laoreet ut, commodo eleifend, cursus nec, lorem. Aenean eu est. Etiam imperdiet turpis. Praesent nec augue. Curabitur ligula quam, rutrum id, tempor sed, consequat ac, dui. Vestibulum accumsan eros nec magna. Vestibulum vitae dui. Vestibulum nec ligula et lorem consequat ullamcorper.

Listing A.1: Example of a XML file.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<StreamInfo version="2.0">
    <Clip duration="PT01M0.00S">
        <BaseURL>videos/</BaseURL>
        <Description>svc_1</Description>
        <Representation mimeType="video/SVC" codecs="svc" frameRate="30.00" bandwidth="401.90"
            width="176" height="144" id="L0">
            <BaseURL>svc_1/</BaseURL>
            <SegmentInfo from="0" to="11" duration="PT5.00S">
                <BaseURL>svc_1-L0-</BaseURL>
            </SegmentInfo>
        </Representation>
```

```
        <Representation mimeType="video/SVC" codecs="svc" frameRate="30.00" bandwidth="1322.60"
            width="352" height="288" id="L1">
            <BaseURL>svc_1/</BaseURL>
            <SegmentInfo from="0" to="11" duration="PT5.00S">
                <BaseURL>svc_1-L1-</BaseURL>
            </SegmentInfo>
        </Representation>
    </Clip>
</StreamInfo>
```

Etiam imperdiet turpis. Praesent nec augue. Curabitur ligula quam, rutrum id, tempor sed, consequat ac, dui. Maecenas tincidunt velit quis orci. Sed in dui. Nullam ut mauris eu mi mollis luctus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Sed cursus cursus velit. Sed a massa. Duis dignissim euismod quam.

Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Phasellus eget nisl ut elit porta ullamcorper. Maecenas tincidunt velit quis orci. Sed in dui. Nullam ut mauris eu mi mollis luctus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos.

This inline MATLAB code `for i=1:3, disp('cool'); end;` uses the `\mcode{}` command.[1]

Nullam ut mauris eu mi mollis luctus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Sed cursus cursus velit. Sed a massa. Duis dignissim euismod quam. Nullam euismod metus ut orci.

Listing A.2: `Matlab Function`

```
for i = 1:3
    if i >= 5 && a ~= b       % literate programming replacement
        disp('cool');         % comment with some LaTeX in it: $\pi x^2$
    end
    [:,ind] = max(vec);
    x_last = x(1,end) - 1;
    v(end);
    ylabel('Voltage (μV)');
end
```

Nullam ut mauris eu mi mollis luctus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Sed cursus cursus velit. Sed a massa. Duis dignissim euismod quam. Nullam euismod metus ut orci.

[1] MATLAB Works also in footnotes: `for i=1:3, disp('cool'); end;`

Listing A.3: `function.m`

```matlab
%   Copyright 2010 The MathWorks, Inc.
function ObjTrack(position)
% #codegen
% First, setup the figure
numPts = 300;               % Process and plot 300 samples
figure;hold;grid;           % Prepare plot window
% Main loop
for idx = 1: numPts
    z = position(:,idx);    % Get the input data
    y = kalmanfilter(z);    % Call Kalman filter to estimate the position
    plot_trajectory(z,y);   % Plot the results
end
hold;
end   % of the function
```

Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Phasellus eget nisl ut elit porta ullamcorper. Maecenas tincidunt velit quis orci. Sed in dui. Nullam ut mauris eu mi mollis luctus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Sed cursus cursus velit. Sed a massa. Duis dignissim euismod quam. Nullam euismod metus ut orci. Vestibulum erat libero, scelerisque et, porttitor et, varius a, leo.

Listing A.4: HTML with CSS Code

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Listings Style Test</title>
    <meta charset="UTF-8">
    <style>
      /* CSS Test */
      * {
        padding: 0;
        border: 0;
        margin: 0;
      }
    </style>
    <link rel="stylesheet" href="css/style.css" />
  </head>
```

```
<header> hey </header>
<article> this is a article </article>
<body>
  <!-- Paragraphs are fine -->
  <div id="box">
    <p>
      Hello World
    </p>
    <p>Hello World</p>
    <p id="test">Hello World</p>
    <p></p>
  </div>
  <div>Test</div>
  <!-- HTML script is not consistent -->
  <script src="js/benchmark.js"></script>
  <script>
    function createSquare(x, y) {
      // This is a comment.
      var square = document.createElement('div');
      square.style.width = square.style.height = '50px';
      square.style.backgroundColor = 'blue';

      /*
       * This is another comment.
       */
      square.style.position = 'absolute';
      square.style.left = x + 'px';
      square.style.top = y + 'px';

      var body = document.getElementsByTagName('body')[0];
      body.appendChild(square);
    };

    // Please take a look at +=
    window.addEventListener('mousedown', function(event) {
      // German umlaut test: Berührungspunkt ermitteln
      var x = event.touches[0].pageX;
      var y = event.touches[0].pageY;
```

```
        var lookAtThis += 1;
      });
    </script>
  </body>
</html>
```

Nulla dui purus, eleifend vel, consequat non, dictum porta, nulla. Duis ante mi, laoreet ut, commodo eleifend, cursus nec, lorem. Aenean eu est. Etiam imperdiet turpis. Praesent nec augue. Curabitur ligula quam, rutrum id, tempor sed, consequat ac, dui. Vestibulum accumsan eros nec magna. Vestibulum vitae dui. Vestibulum nec ligula et lorem consequat ullamcorper.

Listing A.5: HTML CSS Javascript Code

```

@media only screen and (min-width: 768px) and (max-width: 991px) {

  #main {
    width: 712px;
    padding: 100px 28px 120px;
  }

  /* .mono {
    font-size: 90%;
  } */

  .cssbtn a {
    margin-top: 10px;
    margin-bottom: 10px;
    width: 60px;
    height: 60px;
    font-size: 28px;
    line-height: 62px;
  }
```

Nulla dui purus, eleifend vel, consequat non, dictum porta, nulla. Duis ante mi, laoreet ut, commodo eleifend, cursus nec, lorem. Aenean eu est. Etiam imperdiet turpis. Praesent nec augue. Curabitur ligula quam, rutrum id, tempor sed, consequat ac, dui. Vestibulum accumsan eros nec magna. Vestibulum vitae dui. Vestibulum nec ligula et lorem consequat ullamcorper.

Listing A.6: PYTHON Code

```
class TelgramRequestHandler(object):
    def handle(self):
        addr = self.client_address[0]         # Client IP-adress
        telgram = self.request.recv(1024)     # Recieve telgram
        print "From: %s, Received: %s" % (addr, telgram)
        return
```

B

A Large Table

Aliquam et nisl vel ligula consectetuer suscipit. Morbi euismod enim eget neque. Donec sagittis massa. Vestibulum quis augue sit amet ipsum laoreet pretium. Nulla facilisi. Duis tincidunt, felis et luctus placerat, ipsum libero vestibulum sem, vitae elementum wisi ipsum a metus. Nulla a enim sed dui hendrerit lobortis. Donec lacinia vulputate magna. Vivamus suscipit lectus at quam. In lectus est, viverra a, ultricies ut, pulvinar vitae, tellus. Donec et lectus et sem rutrum sodales. Morbi cursus. Aliquam a odio. Sed tortor velit, convallis eget, porta interdum, convallis sed, tortor. Phasellus ac libero a lorem auctor mattis. Lorem ipsum dolor sit amet, consectetuer adipiscing elit.

Nunc auctor bibendum eros. Maecenas porta accumsan mauris. Etiam enim enim, elementum sed, bibendum quis, rhoncus non, metus. Fusce neque dolor, adipiscing sed, consectetuer et, lacinia sit amet, quam. Suspendisse wisi quam, consectetuer in, blandit sed, suscipit eu, eros. Etiam ligula enim, tempor ut, blandit nec, mollis eu, lectus. Nam cursus. Vivamus iaculis. Aenean risus purus, pharetra in, blandit quis, gravida a, turpis. Donec nisl. Aenean eget mi. Fusce mattis est id diam. Phasellus faucibus interdum sapien. Duis quis nunc. Sed enim. Nunc auctor bibendum eros. Maecenas porta accumsan mauris. Etiam enim enim, elementum sed, bibendum quis, rhoncus non, metus. Fusce neque dolor, adipiscing sed, consectetuer et, lacinia sit amet, quam.

As B.1 shows, the data can be inserted from a file, in the case of a somehow complex structure. Notice the Table footnotes.

Table B.1: Example table

Benchmark: ANN	**#Layers** (1)	**#Nets** (2)	#Nodes* $(3)=8\cdot(1)\cdot(2)$	Critical path $(4)=4\cdot(1)$	**Latency** **(T_{iter})** (5)
A1	**3–1501**	1	**24–12008**	**12–6004**	4
A2	501	1	4008	2004	**2–2000**
A3	10	**2–1024**	**160–81920**	40	60†
A4	10	50	4000	40	**80–1200**
Benchmark: FFT	**FFT size‡** (1)	*#Inputs* $(2)=2^{(1)}$	*#Nodes** $(3)=10\cdot(1)\cdot(2)$	*Critical path* $(4)=4\cdot(1)$	**Latency** **(T_{iter})** (5)
F1	**1–10**	2–1024	**20–102400**	4–40	6–60†
F2	**5**	32	1600	20	**40 – 1500**
Benchmark: Random networks	**#Types** (1)	**#Nodes** (2)	*#Networks* (3)	*Critical path* (4)	**Latency** **(T_{iter})** (5)
R1	3	10–2000	500	*variable*	(4)
R2	3	50	500	*variable*	$(4)\times[1;\cdots;20]$

* *Excluding constant nodes.*
† *Value kept proportional to the critical path:* $(5)=(4)*1.5$.
‡ *A size of x corresponds to a 2^x point FFT.*
Values in bold indicate the parameter being varied.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Morbi commodo, ipsum sed pharetra gravida, orci magna rhoncus neque, id pulvinar odio lorem non turpis. Nullam sit amet enim. Suspendisse id velit vitae ligula volutpat condimentum. Aliquam erat volutpat. Sed quis velit. Nulla facilisi. Nulla libero. Vivamus pharetra posuere sapien. Nam consectetuer. Sed aliquam, nunc eget euismod ullamcorper, lectus nunc ullamcorper orci, fermentum bibendum enim nibh eget ipsum. Donec porttitor ligula eu dolor. Maecenas vitae nulla consequat libero cursus venenatis. Nam magna enim, accumsan eu, blandit sed, blandit a, eros.

And now an example (**??**) of a table that extends to more than one page. Notice the repetition of the Caption (with indication that is continued) and of the Header, as well as the continuation text at the bottom.

An example of a large Table that autofits the size to the page margins is illustrated in B.2. Please notice the text size that is shrunken in order fot the table to adjust to the page:

Table B.2: Sample Table.

URL	First Time Visit	Last Time Visit	URL Counts	Value	Reference
https://web.facebook.com/	1521241972	1522351859	177	56640	[facebook-2021]
http://localhost/phpmyadmin/	1518413861	1522075694	24	39312	database-management
https://mail.google.com/mail/u/	1516596003	1522352010	36	33264	Google-Gmail-2021
https://github.com/shawon100	1517215489	1522352266	37	27528	Code-Repository
https://www.youtube.com/	1517229227	1521978502	24	14792	Youtube-video-2021