The abstract provides a clear overview of the paper and its main contributions. It highlights the main motivation for exploring serverless computing for data processing, and outlines the design and analysis performed by the authors to evaluate its suitability. The CCS Concepts and Keywords sections provide relevant information for indexing and searching the paper in academic databases, and the ACM Reference Format provides proper citation information.

Overall, the abstract effectively summarizes the content and significance of the paper, making it easy for potential readers to understand its purpose and scope.

This introduction provides a good overview of the evolution of cloud computing and the different services offered by cloud providers. The explanation of the progression from IaaS to SaaS and finally to FaaS is clear and concise. The distinction between the two ways of using IaaS for a data analytics task is also well explained.

However, there are some minor issues that could be improved. The repetition of the introduction section is a mistake and should be removed. Additionally, the use of technical terms such as "elasticity" and "billing" could benefit from a brief explanation for readers who may not be familiar with these concepts. Finally, it would be helpful to have more specific examples of use cases for FaaS in data analytics to help readers better understand the context.

The overview of Lambada provides a clear description of the architecture of the data analytics engine. It explains how the system uses serverless components from Amazon Web Services, such as AWS Lambda, DynamoDB, and S3, to complement serverless functions and preserve their advantages. The use of other cloud services that incur costs for idle infrastructure is discouraged. The system implements a data-parallel query plan that is executed by serverless workers. These workers run as a function in AWS Lambda and communicate with the driver through shared serverless storage. The architecture is similar to a traditional shared storage database architecture but is implemented in a purely serverless environment. The authors of the overview also discuss some of the challenges they faced while implementing the system and their proposed solutions to overcome these challenges. Overall, the overview provides a good understanding of the system architecture and the challenges faced while developing the system.

This section describes a solution for invoking serverless workers in a batch process to minimize the latency of network round-trip. The authors propose a two-level invocation approach where the first-level workers receive a list of IDs and input data and invoke a second-generation worker for each ID/input pair in that list. This approach is shown to be faster than a sequential invocation, as the last worker is initiated after about 2.5 seconds as opposed to the expected 13-18 seconds for sequential invocation. The authors also analyze the network characteristics of accessing S3 from the serverless workers and provide design principles for implementing scan operators based on the performance and cost of accessing S3. The results of the microbenchmarks are presented in a graphical format to show the distribution of the median, minimum, and maximum bandwidth.

The authors are proposing a family of exchange operators for serverless workers as a building block for data processing. The exchange operator transfers data among workers to support data parallel query processing in joins, sorting, and grouping. However, the basic exchange algorithm has two major issues. The first issue is that the total number of files is quadratic in the number of workers, which can cause throttling by the cloud provider due to a rate limit on requests. The second issue is that the costs of the requests grow quadratically with the number of workers and can be prohibitively expensive.

To address these issues, the authors present two optimizations. The first optimization is to do the exchange through multiple levels, where each level only involves a small subset of the workers. The second optimization involves reducing the number of requests by batching them and sending them in parallel.

Overall, the authors present a useful idea for data processing in a serverless worker environment. However, the limitations of the basic algorithm and the cost issues need to be addressed and addressed effectively.

This is an evaluation of a system (Lambada) in comparison to two Query-as-a-Service systems (Google BigQuery and Amazon Athena). The evaluation focuses on scan-heavy queries from the TPC-H benchmark, which are used to study the performance and cost characteristics of the system. The experiments vary the amount of memory allocated to each worker, as well as the number of files processed by each worker. The results show that increasing the worker size improves the execution time and reduces the cost, but the gains diminish as the worker size increases. The paper also studies the effect of pushing down selections and projects into the scan operator. The results of these experiments are not discussed in this summary.

This paper presents the evaluation of a serverless computing system called Lambada, comparing it with two well-established cloud-based data processing systems, Athena and BigQuery. The evaluation involves running a variety of workloads including TPC-H queries and scientific workloads from hydrology and high-energy physics. The results of the TPC-H queries show that Lambada is competitive with Athena and BigQuery, with a slightly reduced advantage over Athena at the smaller scale factor. The scientific workloads from hydrology and high-energy physics demonstrate that Lambada is able to run these queries interactively and at a cost-effective price.

The authors argue that serverless computing is well suited for interactive data analytics on cold data, which is a common pattern in the initial exploratory phase of data analysis. The results of the evaluation support this argument as Lambada is able to run the hydrology and high-energy physics workloads interactively and at a low cost. The authors also highlight the limitations of the usage-based pricing model of serverless functions, when compared to the size-based pricing model of QaaS. However, they also point out that further optimizations of Lambada could improve its performance and price.

Overall, the evaluation provides a thorough and informative comparison of Lambada with two established data processing systems, and demonstrates its potential as a cost-effective solution for interactive data analytics on cold data.

The related work section provides a comprehensive overview of related work in the fields of data analytics on cold data and serverless computing. The authors clearly state the main lines of related work and provide examples and references to support their claims. They also compare their work with other similar systems, such as Flint and Starling, and highlight the differences and strengths of their approach. The authors also mention previous work on data exchange in the serverless context, but argue that their solution addresses some of the limitations of these approaches.

One area for improvement in this related work section could be to provide a more in-depth analysis of the limitations of the previous approaches and how their work addresses these limitations. Additionally, the authors could include a comparison of their system with more recent work in the field to provide a more up-to-date overview of the state-of-the-art.

Overall, the related work section provides a solid foundation for understanding the context and significance of the authors' work, but could benefit from a more in-depth analysis of the limitations and strengths of previous approaches.