**Реализация алгоритмов MapReduce на моделях   
Python MapReduce и Python MapReduce Distributes**

1. Алгоритм вычисления TF-IDF для корпуса из восьми документов.

2. Алгоритм параллельного поиска в ширину (Breadth-First Search)

Найти кратчайший путь от вершины 0 до вершины 19

Список смежности графа:

**0**: 1, 3

**1**: 0, 2, 4

**2**: 1, 3, 5, 6

**3**: 0, 2, 8

**4**: 1, 10, 11

**5**: 2, 8, 10

**6**: 2, 7

**7**: 6, 12

**8**: 3, 5, 7, 9

**9**: 8, 10

**10**: 4, 5, 9, 11, 16

**11**: 4, 10, 13

**12**: 7, 14

**13**: 11, 14

**14**: 12, 13, 15

**15**: 14, 16

**16**: 10, 15, 17

**17**: 16, 18

**18**: 17, 19

**19**: 18

Документы для задания 1: (используйте приведённые ниже аннотации статей)

[An empirical analysis of stateful operator migration for online scheduling in distributed stream processing systems](https://www.sciencedirect.com/science/article/pii/S0141933123000571)

Streaming data is the data from sensors as well as other real-time surveillance systems. Distributed stream processing systems are the software that manages such data. Such frameworks have to deliver outcomes on the go instantly. They are susceptible to delay and malfunction or system failures. The system must be tolerant of faults and always accessible. Many variables, such as improved network arrival rates, node failures, and so on, disrupt the system's reliability. Some operators need to be relocated online from one physical resource to another to manage or reimburse a slow or failing node. In this study, we propose a co-location based systematic migration heuristic for live operator migration between physical resources using a migration map revised with costs for each migration. The suggested method evaluates continuous operator performance patterns and makes online scheduling decisions based on the same. The decisions include migrating operators during a node failure or straggling.

[Theodolite: Scalability Benchmarking of Distributed Stream Processing Engines in Microservice Architectures](https://www.sciencedirect.com/science/article/pii/S2214579621000265)

Distributed stream processing engines are designed with a focus on scalability to process big data volumes in a continuous manner. We present the Theodolite method for benchmarking the scalability of distributed stream processing engines. Core of this method is the definition of use cases that microservices implementing stream processing have to fulfill. For each use case, our method identifies relevant workload dimensions that might affect the scalability of a use case. We propose to design one benchmark per use case and relevant workload dimension. We present a general benchmarking framework, which can be applied to execute the individual benchmarks for a given use case and workload dimension. Our framework executes an implementation of the use case's dataflow architecture for different workloads of the given dimension and various numbers of processing instances. This way, it identifies how resources demand evolves with increasing workloads. Within the scope of this paper, we present 4 identified use cases, derived from processing Industrial Internet of Things data, and 7 corresponding workload dimensions. We provide implementations of 4 benchmarks with Kafka Streams and Apache Flink as well as an implementation of our benchmarking framework to execute scalability benchmarks in cloud environments. We use both for evaluating the Theodolite method and for benchmarking Kafka Streams' and Flink's scalability for different deployment options.

[A hybrid distributed batch-stream processing approach for anomaly detection](https://www.sciencedirect.com/science/article/pii/S0020025520306897)

Batch and stream processing are separately and efficiently applied in many applications. However, some newer data-driven applications such as the Internet of Things and cloud computing call for hybrid processing approaches in order to handle the speed and accuracy required for processing such complex data. In this paper, we propose a Hybrid Distributed Batch-Stream (HDBS) architecture for anomaly detection in real-time data. The hybrid architecture, while benefiting from the accuracy provided by batch processing, also enjoys the speed and real-time features of stream processing. In the proposed architecture, our focus is on the algorithmic aspects of hybrid processing including the interaction models between batch and stream processing units, the characteristics of batch and stream machine learning algorithms and the principles of merging the results of different processing units. The driving idea of such combination is that the results of batch and stream processing units are complementary with each other, as one of them constructs accurate models based on previous data, and the other one is capable of processing new stream data in real-time. Furthermore, we propose a generalized version of the HDBS with respect to its algorithms and communication policy levels. In the generalized HDBS architecture, we address the various aspects of the interaction between the batch and stream processing units, and the merging operations to produce the final results. the evaluations of the proposed architecture using various criteria (accuracy, space complexity, and time complexity) demonstrate that the accuracy of the proposed method is higher than the accuracy of the batch processing methods, its time complexity is also similar to one of the stream processing methods and much less than the batch processing methods, which makes our proposed architecture an efficient and practical solution for real-time anomaly detection.

[Topology-aware task allocation for online distributed stream processing applications with latency constraints](https://www.sciencedirect.com/science/article/pii/S0378437119311604)

There have been increasing demands for real time processing of the ever-growing data. In order to meet this requirement and ensure the reliable processing of streaming data, a variety of distributed stream processing architectures and platforms have been developed, which handles the fundamental task of allocating processing tasks to the currently available physical resources and routing streaming data between these resources. However, many stream processing systems lack an intelligent scheduling mechanism, in which their default schedulers allocate tasks without taking resource demands and availability, or the transfer latency between resources into consideration. Besides, stream processing has a strict request for latency. Thus it is important to give latency guarantee for distributed stream processing. In this paper, we propose two new algorithms for stream processing with latency guarantee, both the algorithms consider transfer latency and resource demand in task allocation. Both algorithms can guarantee latency constraints. Algorithm AHA reduces more than 21.3% and 58.9% resources compared with the greedy and the round-robin algorithms, and algorithm PHA further improves the resource utilization to 32.1% and 73.2%.

[Improvement Design for Distributed Real-Time Stream Processing Systems](https://www.sciencedirect.com/science/article/pii/S1674862X19300023)

In the era of Big Data, typical architecture of distributed real-time stream processing systems is the combination of Flume, Kafka, and Storm. As a kind of distributed message system, Kafka has the characteristics of horizontal scalability and high throughput, which is manly deployed in many areas in order to address the problem of speed mismatch between message producers and consumers. When using Kafka, we need to quickly receive data sent by producers. In addition, we need to send data to consumers quickly. Therefore, the performance of Kafka is of critical importance to the performance of the whole stream processing system. In this paper, we propose the improved design of real-time stream processing systems, and focus on improving the Kafka’s data loading process. We use Kafka cat to transfer data from the source to Kafka topic directly, which can reduce the network transmission. We also utilize the memory file system to accelerate the process of data loading, which can address the bottleneck and performance problems caused by disk I/O. Extensive experiments are conducted to evaluate the performance, which show the superiority of our improved design.

[Comprehensive analysis of the heterogeneous computing performance of DNNs under typical frameworks on cloud and edgecomputing platforms](https://www.sciencedirect.com/science/article/pii/S0957417423009776)

In this paper, nearly 40 commonly used deep neural network(DNN) models are selected, and their cross-platform and cross-inference frameworks are deeply analysed. The main metrics of accuracy, the total number of model parameters, the computational complexity, the accuracy density, the inference time, the memory consumption and other related parameters are used to measure their performance. The heterogeneous computing experiment is implemented on both the Google Colab cloud computing platform and the Jetson Nano embedded edge computing platform. The obtained performance is compared with that of two previous computing platforms: a workstation equipped with an NVIDIA Titan X Pascal and an embedded system based on an NVIDIA Jetson TX1 board. In addition, on the Jetson Nano embedded edge computing platform, different inference frameworks are investigated to evaluate the inference efficiency of the DNN models. Regression models are established to characterize the variation in the computing performance of different DNN classification algorithms so that the inference results of unknown models can be estimated. ANOVA methods are proposed to quantify the differences between models. The experimental results have important guiding significance for the better selection, deployment and application of DNN models in practice. Codes are available at this https URL https://github.com/Foreverzfy/Model-Test.

[Low-power deep learning edge computing platform for resource constrained lightweight compact UAVs](https://www.sciencedirect.com/science/article/pii/S2210537922000609)

Unmanned Aerial Vehicles (UAVs), which can operate autonomously in dynamic and complex environments, are becoming increasingly common. Deep learning techniques for motion control have recently taken a major qualitative step since vision-based inference tasks can be executed directly on edge. The goal is to fully integrate the machine learning (ML) element into small UAVs. However, given the limited payload capacity and energy available on small UAVs, integrating computing resources sufficient to host ML and vehicle control functions is still challenging. This paper presents a modular and generic system that can control the UAV by evaluating vision-based ML tasks directly inside the resource-constrained UAV. Two different vision-based navigation configurations were tested and demonstrated. The first configuration implements an autonomous landing site detection system, tested with two models based on LeNet-5 and MobileNetV2, respectively. This allows the UAV to change its planned path accordingly and approach the target to land. Moreover, a model for people detection based on a custom MobileNetV2 network was evaluated in the second configuration. Finally, the execution time and power consumption were measured and compared with a cloud computing approach. The results show the ability of the developed system to dynamically react to the environment to provide the necessary maneuver after detecting the target exploiting only the constrained computational resources of the UAV controller. Furthermore, we demonstrated that moving to the edge, instead of using cloud computing inference, decreases the energy requirement of the system without reducing the quality of service.

[Heterogeneous edge computing open platforms and tools for internet of things](https://www.sciencedirect.com/science/article/pii/S0167739X19312841)

With the continuous development of Internet of Things (IoT) and the overwhelming explosion of Big Data, edge computing serves as an efficient computing mode for time stringent data processing, which can bypass the constraints of network bandwidth and delay, and has been one of the foundation of interconnected applications. Although edge computing has gradually become one of bridges between cloud computing centers and mobile terminals, the literature still lacks a thorough review on the recent advances in edge computing platforms. In this paper, we firstly introduce the definition of edge computing and advantages of edge computing platform. And then, we summarize the key technologies of constructing an edge computing platform, and propose a general framework for edge computing platform. The role of distributed storage management systems in building edge computing platform is elaborated in detail. Furthermore, we give some applications to illustrate how to use third-party edge computing platforms to build specific applications. Finally, we briefly outline current open issues of edge computing platform based on our literature survey.