Monitoring System Architecture for the Multi-Scale Blockchain-based Logistic Network

Vladislav Kashansky, Radu Prodan
{vladislav.kashanskii,radu.prodan}@aau.at
Institute of Information Technology, University of Klagenfurt
Austria

Aso Validi, Cristina Olaverri-Monreal
{aso.validi,cristina.olaverri-monreal}@jku.at
Chair ITS-Sustainable Transport Logistics 4.0, Johannes Kepler University Linz
Austria

Gleb Radchenko
gleb.radchenko@silicon-austria.com
Silicon Austria Labs
Austria

ABSTRACT

Contemporary control processes and methods in multi-scale, cyber-physical systems require precise data collection at various levels, timely transmission, and analysis involving large number of computing and storage elements connected within high-performance permissioned consensus networks. For example, in transport networks, resources tend to form multi-scale dynamical systems with diverse operational requirements, including data exchange policies and consensus protocols. Apart from designing complete topology, chaincodes and consensus logic, effective monitoring of the applications and infrastructure of such complex systems remains a research challenge. In this paper, we discuss important aspects of the data-intensive applications monitoring investigated in the frames of the ADAPT project. We present highlights on the toolsets, architectures and details on possible optimization approaches for monitoring data collection. We introduce a dynamic multi-scale monitoring system architecture with preliminary workflow model. It allows obtaining effective low-latency publish-subscribe matching of the dynamically varying monitoring tasks and executing machines.

CCS CONCEPTS

• **Computer systems organization** → **Peer-to-peer architectures**; **Distributed architectures**; • **Applied computing** → **Transportation**.

KEYWORDS

Logistics, transportation, decentralization, blockchain, monitoring systems, optimization, data-intensive systems, hybrid systems

UCC '21 Companion, December 6–9, 2021, Leicester, United Kingdom

ACM ISBN 978-1-4503-9163-4/21/12…$15.00
https://doi.org/10.1145/3492323.3495633

ACM Reference Format:
Vladislav Kashansky, Radu Prodan, Aso Validi, Cristina Olaverri-Monreal, and Gleb Radchenko. 2021. Monitoring System Architecture for the Multi-Scale Blockchain-based Logistic Network . In *2021 IEEE/ACM 14th International Conference on Utility and Cloud Computing (UCC '21) Companion (UCC '21 Companion), December 6–9, 2021, Leicester, United Kingdom.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3492323.3495633

1 INTRODUCTION

Modern computational systems are complex hybrid networks of the converging [7, 21] cloud, cyber-physical, and HPC systems. These systems normally receive large volumes of data transmitted and analyzed by significantly heterogeneous processing architectures [17, 31, 32], with many computing and storage elements connected within high-performance permissioned blockchain networks. Moreover, transport networks tend to form multi-scale dynamical systems with diverse requirements on their operational processes, including data exchange policies and consensus protocols. Many results show that public transport networks have characteristics of complex networks. All parameters of the transport network at different scales significantly affect accessibility, performance, and reliability.

Besides, performance is one of the most important issues of various blockchain systems, especially when executing complex smart contracts, requiring adaptive continuous monitoring solutions. These systems collect data on the states of distributed infrastructure components (i.e., computing, virtual and telecommunications infrastructures, cloud environments), process them, and help to identify prediction models and root causes of the incidents.

Furthermore, when discussing transportation networks, it is crucially important to keep track of infrastructures and autonomous vehicles (e.g., fuel consumption, temperatures, and GPS data) that function independently inside platooning formations and blockchain systems (e.g., transaction rates, reliability, transaction data, etc.).

It is impossible and often impractical to globally measure the performance metrics of large-scale applications while preserving, for example, I/O limitation policies. Thus, it is critical to identify:

- The important parameters on the various system scales and their monitoring granularity;
- The measurement interval and the communication patterns concerning these intervals;
- The aggregation and pre-processing of performance metrics at a monitor granularity for further analysis.

This paper contributes on important aspects of data-intensive logistic blockchain monitoring investigated in the ADAPT project [22]. We present highlights on the toolsets and architectures. Further we introduce complete dynamic multi-scale monitoring architecture. Section 2 presents related work. Section 4 describes management aspects of monitoring data collection. Section 6 concludes the paper.

2 RELATED WORK

2.1 Blockchain for Transport and Logistics

Blockchain technology has emerged in the transportation and logistics industry to solve more efficient and cost-saving business operations [28]. Blockchain enables the coordination of documentation tasks on a shared distributed ledger, ensures trustworthy data across the transportation ecosystem and provides a scalable, immediate solution for authentication and tracking by design [15]. For example, shippers can post unique timestamped loads recorded and verified by the blockchain with preserved integrity.

Blockchain further enables transparency in the supply chain. Abeyratne et al. [5] reviewed the state-of-the-art and example applications, discussed the potential of technology, and proposed a vision for the future blockchain-ready manufacturing supply chain. A few supply chains are already using blockchain technology [14, 35]. For example, UbiMS start-up is the world's first patented cloud-based platform that reinvented the global supply chain process using blockchain to connect multiple goods providers with worldwide consumers. UbiMS is a shared supply chain infrastructure for small and medium enterprises, modeled as a three-dimensional globally interconnected e-marketplace and e-supply chain process for communication and distribution of material goods. Smart contracts coded through the blockchain and self-executed upon meeting certain conditions can reduce costs and eliminate the need for administrative steps, representing around 20% of the overall transportation costs. Casado-Vara et al. [12] combined smart contracts with a multi-agent system to improve the current logistics system's organization security and distribution times. ShipChain (https://blog.shipchain.io/) is one of the first start-ups that used smart contracts in maritime logistics to track goods from leaving the factory until customer delivery. Process automation allows stakeholders to purchase digital SHIP currency tokens to pay the cargo and execute transactions on the platform, enabling information sharing and platform transparency.

2.2 IoT-Blockchain Enabled Logistics Supply Chain Assurance, Auditing, and Monitoring

Sweetbridge (https://sweetbridge.com/) provides distributed protocols that integrate artificial intelligence, distributed ledger technology, and the Internet of Things (IoT) to improve the efficiency of settlement between parties throughout the supply chain in the freight and trucking industry. These protocols use decentralized inter-enterprise resource planning to perform continuous data assurance, auditing and real-time control monitoring. The IoT sensors in trucks and other shipping vehicles can help shippers and transportation companies detect the space used in cargo and determine individual freight costs stored in the blockchain.

The authors in [37] proposed a detailed and real-time performance monitoring framework for blockchain systems.

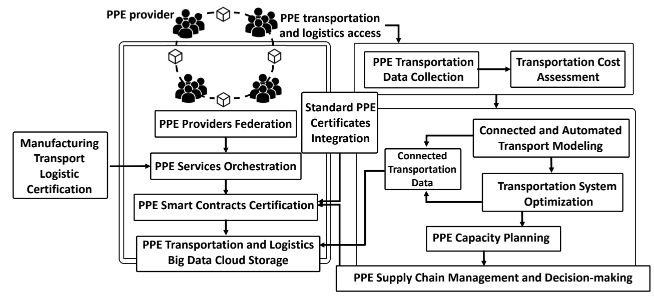

Figure 1: ADAPT architecture: decentralized transportation, logistic network, and transportation characteristics.

Peloton uses blockchain to store and validate the data exchanged among multiple freight vehicles forming a platoon and improving fuel efficiency and safety. Malik et al. [25] argue that blockchain cannot support the trust and reliability of data concerning physical commodities' quality and supply chain entities' trustworthiness. They provide an automated framework to associate trust to each supply chain event based on the trust of the participants and the quality of the commodity.

2.3 General Purpose Monitoring Architectures and Techniques

Understanding and optimizing data-intensive applications requires a comprehensive review of historical metrics. For example, it is difficult or impossible to tune applications on big data platforms with access to the coarse-grained system metrics only. Monitoring data is also critical for optimized job schedulers and auto-scalers [19, 20], even at a coarse-grain level [6].

Authors in [29] propose a dynamic architecture for monitoring large-scale distributed systems that use a specialized service to collect information and automatically improve task scheduling. Another work [30] presents a distributed resource monitoring architecture to determine the optimal set of machines to run an application and implements prediction methods to estimate the overall performance. The authors in [10] proposed a collaborative scheduling method for CPU and memory-intensive applications on a single node using monitoring information to improve overall throughput and energy efficiency. In centralized monitoring, all resource states and metrics are sent to a single monitoring server, continuously receiving them from each monitored component. This allows for more controlled management and data access but potentially sacrifices availability and elasticity by creating a single point of failure and eliminates horizontal scalability. Decentralized architectures solve most scalability and elasticity problems of centralized architectures without a single point of failure.

Authors in [18] describe generalized approaches and architectures to system performance analysis, including notable toolkits and visualization [11] techniques. Other works [1–3] provide detailed analysis of cloud and HPC monitoring techniques carried out in the frame of the European ASPIDE project [3].

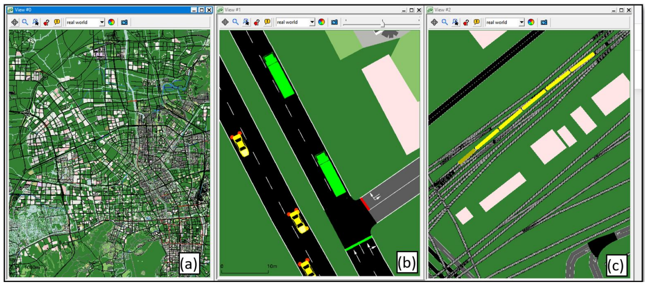

Figure 2: Visualization of part of the traffic network in Shanghai (China) (a), with the simulated vehicles, the trucks (b) railway and trains (c) .

3 ADAPT PROJECT

This section covers the requirements for the monitoring model and environment from the project's perspective.

3.1 ADAPT Architecture

This section presents the ADAPT architecture design (displayed in Figure 1), focusing on the decentralized blockchain-based PPE transportation and logistics network, followed by the autonomous transport and the Supply Chain decision making mechanism.

Automated Transport Control. ADAPT relies on real-time data using the open-source Simulation of Urban Mobility (SUMO) [8] program to mimic connected and automated solutions like platooning to reduce fuel consumption and CO_2 emissions. Figure 2 presents a part of the simulated network in Shanghai (China). ADAPT uses the Veins, an open-source vehicular network simulation framework [34] for vehicular communication. Veins relies on the OMNeT++ (https://omnetpp.org/) discrete event simulator to better represent, analyze, and implement vehicle platooning. Using real-time metrics within the integrated Veins, OMNeT++, and SUMO simulations allows autonomous vehicles to communicate wirelessly to exchange position, speed, and acceleration data.

Transportation System Optimization. The goal of ADAPT is to develop mobility models for connected and automated vehicles (e.g., platooning). The cooperative adaptive cruise control model used in the SUMO simulation, allows platooning without considering vehicular communication. We evaluate additional approaches for achieving and maintaining an efficient and sustainable connected and automated transportation network.

Connected transportation data. ADAPT uses the blockchain platform to store some of the data linked to connected and automated vehicles' speed, acceleration, and position. It also seeks to visualize the road network and traffic demand on a national scale by utilizing 3DCoAutoSim simulation [27]. This is to replicate the traffic network and its environment in a lab-controlled environment. The 3DCoAutoSim simulation framework simulates vehicle-to-vehicle communication from a driver-centric perspective [9], including automated capabilities and the ability to imitate cooperative driving assistance systems.

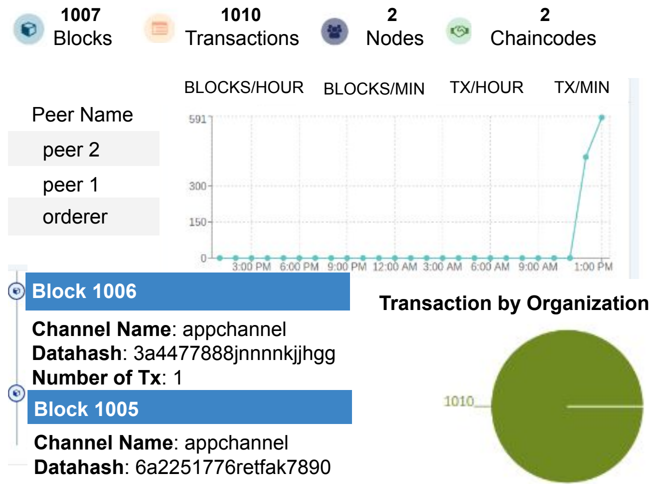

Figure 3: Aggregated statistics dashboard of the Hyperledger Fabric with Hyperledger Explorer in the ADAPT project.

3.2 Monitoring System Requirements

We analyzed the project use-cases and identified a set of critical requirements grouped in three levels, summarized in Table 1.

- *Local urban level* contains sensors measurements on vehicle position (GPS), fuel consumption and speed.
- *Organization level* contains aggregated information on physical and virtual execution nodes, such as CPU, RAM, and I/O, including read/write bytes, read/write operations, and I/O exceptions.
- *Cross organization level* contains information on the hyperledger operations.

Table 1 shows that system metrics such as CPU, RAM, and I/O with per-node granularity are most important for the organization level. We obtain these metrics with the Prometheus and Nagios monitoring systems combined with other cloud solutions. At the same time, we must consider underlying local urban level and cross-organization metrics too. We obtained coarse-grained (Fig. 3) cross-organization metrics associated with the hyperledger fabric using the Hyperledger Explorer [13]. Hyperledger Explorer is a web interface used to view, invoke, deploy, or query blocks, transactions and associated data, network information (e.g., name, status, list of nodes), chain codes and transaction families, and any other relevant information stored in the ledger.

The main complexity arises from the local level, where gathering important fine-grained metrics is necessary. Combining these metrics with dynamic profiling data (Fig. 4) and coarse-grained statistics (Fig. 3) for further analysis introduces additional complexity for system control software. Thus, it is important to provide a comprehensive workflow-based approach that allows aggregation and pre-processing of coarse-grained data subject to the required application scale.

3.3 ADAPT Monitoring System

We introduce a *dynamic multi-scale monitoring system* enabling distributed multi-scale monitoring and analysis, exposing and associating collected metrics with potential reliability or performance

Table 1: Monitoring requirements in the ADAPT project.

Scales	*Monitored system*	*Metrics*	*Latency*
Local urban level	Local self-organization (platooning)	Mostly sensors measurements: video, LiDAR, temperatures, GPS, fuel consumption	Sensitive, real-time data
Organization level	Data center (IaaS), Services (PaaS, SaaS)	CPU/RAM use, I/O latency heatmaps	Non-sensitive, coarse-grained statistics
Cross-organization level	Blockchain: hyperledger fabric	Adaptive profiling, failure monitoring, transaction rates	Non-sensitive, coarse-grained statistics

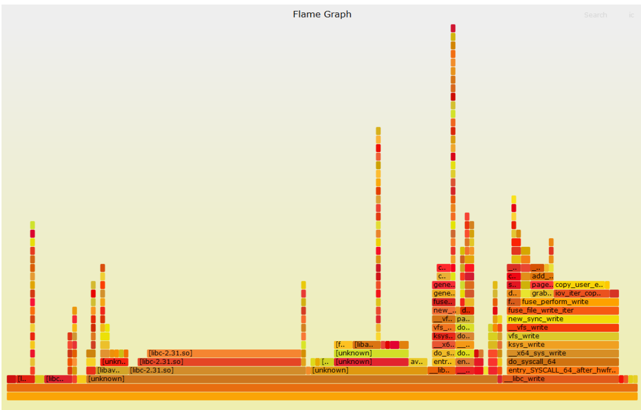

Figure 4: Dynamic profiling visualization flame graph [18] of the hyperledger fabric transaction execution operation illustrating the time spent in different code regions using Linux performance counters [4]. The X-axis represents the number of samples and the Y-axis the code region.

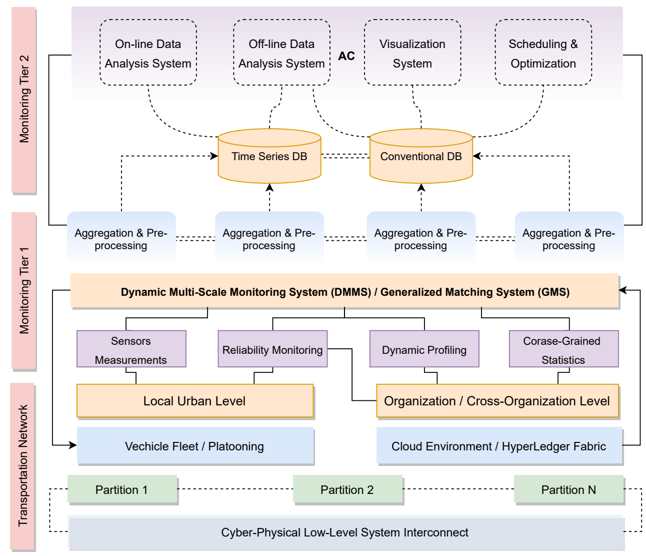

Figure 5: Monitoring system architecture split into two tiers: Tier 1 responsible for the low-latency fine-grained data collection, aggregation, and coarse-grained view of the cluster performance, and Tier 2 representing a high-level collection of the decision support tools.

issues. The conceptual monitoring system depicted in Figure 5 consists of the following components:

(1) Generalized matching system (*GMS*) uses a declarative approach with the SCIP optimization framework [16], based on the workflow model and generalized assignment problem [17, 26] relaxation to obtain publish-subscribe matching of the dynamically varying monitoring agents and aggregators.
(2) *Multi-scale aggregation and event detection* provides decentralized monitoring data aggregation from the agents and detects possible events.
(3) *Main analysis component* is centralized and provides a set of analytic tools, including smart monitoring of application performance and bottlenecks detection.
(4) *Application programming interface* enables access to the services provided by the monitoring platform.

Further sections will contribute on details and model for points 1 and 2 of the list.

4 MULTI-SCALE MONITORING DATA COLLECTION WORKFLOW MODEL

Figure 6 depicts the general structure of dynamic data collection workflow:

$$\mathcal{V} = (V, E)\,, \tag{1}$$

consisting of the following task classes, we execute over the set of the machines M.

WP_1 optimization task, related to the computing of the optimal makespan,
SP_n activation signal propagation pseudo-task,
DC_n data collection tasks of different duration,
DF_n data filtration tasks (data dependent), which can be optional depending on the transaction instance.
AG aggregation of results,
DA data archiving and report preparation.
IN *and* OUT source and sink pseudo-tasks of transaction initiator.

4.1 Makespan Minimization Problem

General Formulation. After defining the workflow mapping:

$$\mu(i) \mapsto m, \quad \forall i \in V^* \equiv V \setminus \{SP_n, DC_n, IN, OUT\}\,, \tag{2}$$

a sequencing algorithm creates the schedule by computing the finish times $F_{i,\mu(i)}$ of every task $i \in \mathcal{V}$. The makespan is equal to the finish time $F_{n,\mu(n)}$ of the last OUT task, marked green in

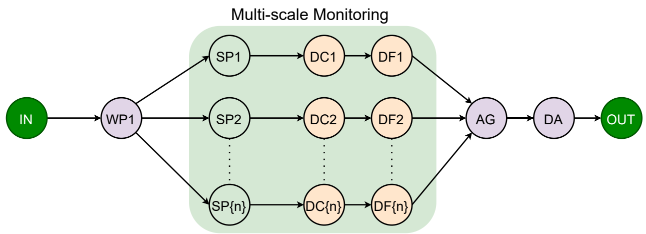

Figure 6: ADAPT multi-scale monitoring data collection workflow

Figure 6. The objective is to minimize it:

$$\min \quad T = F_{n,\mu(n)} \tag{3}$$

subject to the following precedence and resource constraints in the context of structural and computational models [23, 24]:

$$F_{j,\mu(j)} \leq F_{i,\mu(i)} - \tau^1_{i,\mu(i)} \qquad \forall i \in V \wedge j \in \mathcal{P}(i) \tag{4}$$

$$\sum_{i \in A(t)} r_{iz} \leq K_z \qquad \forall t \;\; \forall z \tag{5}$$

$$F_i \geq 0 \qquad \forall i \in V \tag{6}$$

where $\tau^1_{j,\mu(j)} \in \mathbb{Z}^+$ is the task execution time, $\mathcal{P}(i)$ set of the predecessor tasks of the i^{th} task, $r_{jz} \in \mathbb{Z}^+$ is the z^{th} renewable resource consumption by the j^{th} task in the set $A(t)$ of the *active* tasks at the given time instant t, and $K_z \in \mathbb{Z}^+$ is the upper bound on the available resource quantity. Specifically, Eq. 4 ensures the precedence constraints of the tasks finish times within the DAG structure, Eq. 5) limits overall resources usage at each time instant t and Eq. 6 prescribes non-negative execution times.

Finish and Execution Times. To compute finish times for each task, we define a data matrix D_{ij} that indicates the amount of data transmitted from task i to task j. Consequently, we obtain the *delay tensor* D^* [21] for transferring data from task i to task j assigned to the machines m and q:

$$d^*_{ijmq} = \overbrace{\mathcal{T}_{mq}}^{\text{Connection Delay}} + \overbrace{\mathcal{D}_{ij} \cdot \mathcal{B}^{-1}_{mq}}^{\text{Data Transfer Latency}} . \tag{7}$$

The first term $\mathcal{T}_{mq}$ in Eq. 7 represents static connection delay, assumed as a *given* constant. Second term results from the time required to transfer the data from task i to j residing on the machines m and q, obtained by dividing the elements of data matrix D_{ij} by the corresponding bandwidth $\mathcal{B}_{mq}$. We compute the execution time of a given task $i \in V$ as follows:

$$\tau^1_{im} = \max_{j \in \mathcal{P}(i)} \{d^*_{ijmq}; \;\; \forall j \in \mathcal{P}_i\} + \tau^0_{im} \tag{8}$$

where $\mathcal{P}_i$ is the set of predecessors of task i, the upper indices give the order of the approximation, the term τ^1_{im} corresponds to the execution time perturbed by the transfer times and τ^0_{im} corresponds to the execution time measured in perfect conditions with no noise and data transfer delays. We assume a reasonable first-order approximation where the bandwidth is independent of the number of simultaneously transferring agents.

5 SOLUTION STATUS

Figure 7 depicts solution dynamics of the problem (Eq. 3) where we use two-phase SGS heuristic (see RCPSP and MRCPSP formulations [23, 24]) in combination with simulated annealing (SA) as first matching phase. It has polynomial complexity and provides the full knowledge about matching and schedule lifespan. However for the given system it is only important to compute effective matching and information about schedule is considered redundant. The reason is that those environments are chaotic and it is impossible to estimate finish times F_i with high precision. Moreover scalability of the solution becomes limited in high dimensions, for example, problems of 1000 tasks and 500 machines.

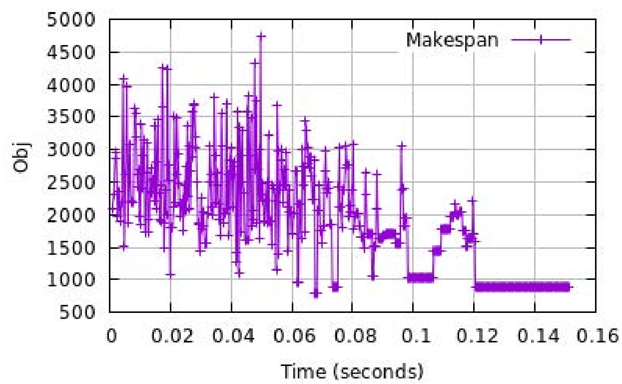

Figure 7: The results of the Simulated Annealing (SA) algorithm in the context of two-phase SGS heuristic [23, 24]. We can see that the primal bound converges to the local optimum of 900 seconds. The initial iterations of the algorithm show a certain variance of the objective function values, which is in agreement with theoretical estimates.

Therefore we currently architect generalized matching system (GMS) that uses a declarative mixed-integer linear programming formulation for publish-subscribe matching of dynamically varying monitoring tasks and executing machines. It is based on the generalized assignment problem (GAP) relaxation. In particular, we use by default the SCIP framework with the possibility to select different solution regimes (i.e., heuristic, optimality, enumeration) [17] in combination with heterogeneous earliest finish time (HEFT) [36] list scheduling heuristic to compute weights of the tasks and relax precedence constraints. The objective is to offer a nearly optimal controlling solution for monitoring agents' raw data aggregation while keeping the response time of the monitoring system and the network traffic induced by the data transfers from monitors to aggregators within the required range.

6 CONCLUSION AND FUTURE WORK

We have initially explored the architectural concepts of distributed multi-scale monitoring system control strategies, which are expected to enable near-optimal control systems performance in large-scale blockchain-based transport networks. We have explored different classes of monitoring tools and showed *one possible* way to perform low-latency data collection of monitoring metrics at different cyber-physical scales and granularity, including precedence relations of the process model. We have formulated formal workflow model, including preliminary vision on the particular workflow structure. Further important step will be finalization of the generalized matching system and comparative heuristic evaluation in realistic conditions. Moreover, several distributed optimization schemes also require further investigation to deal with large-scale MIPs beyond existing centralized architectural capabilities. Ubiquity Generator (UG) [33] is a generic framework to parallelize branch-and-bound based solvers in a distributed or shared memory computing environment. The ParaSCIP extension developed using UG consists of a set of base classes to instantiate parallel branch-and-bound based solvers. We instantiated ParaSCIP parallel solver and used SCIP as the black box MIP solver and MPI as the parallelization library.

ACKNOWLEDGMENTS

This work received financial support from the ADAPT project funded by the Austrian Research Promotion Agency (grant agreement 881703) and ASPIDE project funded by the European Union's Horizon 2020 Programme (grant agreement 801091). We also acknowledge Logoplan GmbH and Eteronix GmbH for consulting and cooperation.

REFERENCES

[1] [n.d.]. ASPIDE D3.1 Low overhead software instrumentation and monitoring. https://www.aspide-project.eu/wp-content/uploads/sites/62/2019/05/D3-1.pdf. Accessed: 2021-01-29.

[2] [n.d.]. ASPIDE D3.2 Extreme scale monitoring architecture – Initial. https://www.aspide-project.eu/wp-content/uploads/sites/62/2019/07/D3-2.pdf. Accessed: 2021-01-29.

[3] [n.d.]. ASPIDE D3.3 Monitoring data collection and mining for Exascale systems. https://www.aspide-project.eu/wp-content/uploads/sites/62/2020/01/D3-3.pdf. Accessed: 2021-01-29.

[4] [n.d.]. perf(1) — Linux manual page. https://man7.org/linux/man-pages/man1/perf.1.html. Accessed: 2021-06-10.

[5] Saveen A. Abeyratne and Radmehr P. Monfared. 2016. Blockchain Ready Manufacturing Supply Chain Using DLedger. *International Journal of Research in Engineering and Technology* 5, 9 (2016), 1–10.

[6] Vaggelis Antypas, Nikos Zacheilas, and Vana Kalogeraki. 2015. Dynamic Reduce Task Adjustment for Hadoop Workloads. In *Proceedings of the 19th Panhellenic Conference on Informatics* (Athens, Greece) *(PCI '15)*. ACM, New York, NY, USA, 203–208. https://doi.org/10.1145/2801948.2801953

[7] Pete Beckman, Jack Dongarra, Nicola Ferrier, Geoffrey Fox, Terry Moore, Dan Reed, and Micah Beck. 2020. Harnessing the Computing Continuum for Programming Our World. *Fog Computing: Theory and Practice* (2020), 215–230.

[8] Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. 2011. SUMO – Simulation of Urban Mobility: an Overview. In *Third International Conference on Advances in System Simulation*. IARIA, Barcelona, Spain, 23–28.

[9] Carlos Biurrun, Luis Serrano-Arriezu, and Cristina Olaverri-Monreal. 2017. Microscopic Driver-Centric Simulator: Linking Unity3D and SUMO. In *5th World Conference on Information Systems and Technologies*. Springer, Porto Santo Island, Madeira, Portugal, 851–860.

[10] J. Breitbart, J. Weidendorfer, and C. Trinitis. 2015. Case Study on Co-scheduling for HPC Applications. In *2015 44th ICPP Conference Workshops*. 277–285.

[11] Gregg Brendan. [n.d.]. CPU Flame Graphs - Manual. http://www.brendangregg.com/FlameGraphs/cpuflamegraphs.html. Accessed: 2021-01-29.

[12] Roberto Casado-Vara, Alfonso González-Briones, Javier Prieto, and Juan M. Corchado. 2018. Smart Contract for Monitoring and Control of Logistics Activities: Pharmaceutical Utilities Case Study. In *13th International Conference on Soft Computing Models in Industrial and Environmental Applications (AISC, 771)*. Springer, San Sebastian, Spain, 509–517.

[13] Vikram Dhillon, David Metcalf, and Max Hooper. 2017. The hyperledger project. In *Blockchain enabled applications*. Springer, 139–149.

[14] Mario Dobrovnik, David M. Herold, Elmar Fürst, and Sebastian Kummer. 2018. Blockchain for and in Logistics: What to Adopt and Where to Start. *Logistics* 2, 3 (Sept. 2018), 18.

[15] Jiang Duan, Chen Zhang, Yu Gong, Steve Brown, and Zhi Li. 2020. A Content-Analysis Based Literature Review in Blockchain Adoption within Food Supply Chain. *International Journal of Environmental Research and Public Health* 17, 5 (March 2020), 1784.

[16] Ambros Gleixner et al. 2018. *The SCIP Optimization Suite 6.0*. ZIB-Report 18-26. Zuse Institute Berlin. http://nbn-resolving.de/urn:nbn:de:0297-zib-69361

[17] Vladislav Kashansky et al. 2020. M3AT: Monitoring Agents Assignment Model for Data-Intensive Applications. In *2020 28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing*. IEEE, Västerås, Sweden, 72–79.

[18] Brendan Gregg. 2014. *Systems performance: enterprise and the cloud*. Pearson Education.

[19] Alexey Ilyushkin, Ahmed Ali-Eldin, Nikolas Herbst, Alessandro V Papadopoulos, Bogdan Ghit, Dick Epema, and Alexandru Iosup. 2017. An experimental performance evaluation of autoscaling policies for complex workflows. In *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering*. 75–86.

[20] Alexey Ilyushkin, Bogdan Ghit, and Dick Epema. 2015. Scheduling workloads of workflows with unknown task runtimes. In *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 606–616.

[21] Vladislav Kashansky, Gleb Radchenko, and Radu Prodan. 2021. Monte Carlo Approach to the Computational Capacities Analysis of the Computing Continuum. In *International Conference on Computational Science*. Springer, 779–793.

[22] Vladislav Kashansky et al. 2021. The ADAPT Project: Adaptive and Autonomous Data Performance Connectivity and Decentralized Transport Network. In *Proceedings of the Conference on Information Technology for Social Good (GoodIT'21)*. ACM, New York, NY, USA.

[23] Rainer Kolisch. 1996. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. *European Journal of Operational Research* 90, 2 (1996), 320–333.

[24] Rainer Kolisch and Andreas Drexl. 1997. Local search for nonpreemptive multi-mode resource-constrained project scheduling. *IIE transactions* 29, 11 (1997), 987–999.

[25] Sidra Malik, Volkan Dedeoglu, Salil S. Kanhere, and Raja Jurdak. 2019. TrustChain: Trust Management in Blockchain and IoT Supported Supply Chains. In *International Conference on Blockchain*. IEEE, Atlanta, USA, 184–193.

[26] Silvano Martello and Paolo Toth. 1981. A bound and bound algorithm for the zero-one multiple knapsack problem. *Discrete Applied Mathematics* 3, 4 (1981), 275–288.

[27] Florian Michaeler and Cristina Olaverri-Monreal. 2017. 3D Driving Simulator with VANET Capabilities to Assess Cooperative Systems: 3DSimVanet. In *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, IEEE, Redondo Beach, CA, USA, 999–1004.

[28] Guido Perboli, Stefano Musso, and Mariangela Rosano. 2018. Blockchain in logistics and supply chain: A lean approach for designing real-world use cases. *Ieee Access* 6 (2018), 62018–62028.

[29] Florin Pop, Ciprian Dobre, Corina Stratan, Alexandru Costan, and Valentin Cristea. 2009. Dynamic Meta-Scheduling Architecture Based on Monitoring in Distributed Systems. In *2009 International Conference on Complex, Intelligent and Software Intensive Systems*. IEEE.

[30] S. Rajkumar, N. Rajkumar, and V. G. Suresh. 2014. Hybrid approach for monitoring and scheduling the job in heterogeneous system. In *International Conference on Information Communication and Embedded Systems (ICICES2014)*. IEEE.

[31] Albert Reuther, Chansup Byun, William Arcand, David Bestor, Bill Bergeron, Matthew Hubbell, Michael Jones, Peter Michaleas, Andrew Prout, Antonio Rosa, et al. 2018. Scalable system scheduling for HPC and big data. *J. Parallel and Distrib. Comput.* 111 (2018), 76–92.

[32] Zahra Najafabadi Samani, Nishant Saurabh, and Radu Prodan. 2021. Multilayer Resource-aware Partitioning for Fog Application Placement. In *2021 IEEE 5th International Conference on Fog and Edge Computing (ICFEC)*. IEEE, 9–18.

[33] Yuji Shinano, Tobias Achterberg, Timo Berthold, Stefan Heinz, and Thorsten Koch. 2011. ParaSCIP: a parallel extension of SCIP. In *Competence in High Performance Computing 2010*. Springer, 135–148.

[34] Christoph Sommer, Reinhard German, and Falko Dressler. 2011. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing* 10, 1 (January 2011), 3–15.

[35] Edvard Tijan, Saša, Aksentijević, Katarina Ivanić, and Mladen Jardas. 2019. Blockchain Technology Implementation in Logistics. *Sustainability* 11, 4 (Feb. 2019), 1185.

[36] Haluk Topcuoglu, Salim Hariri, and Min-You Wu. 2002. Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE transactions on parallel and distributed systems* 13, 3 (2002), 260–274.

[37] Peilin Zheng, Zibin Zheng, Xiapu Luo, Xiangping Chen, and Xuanzhe Liu. 2018. A detailed and real-time performance monitoring framework for blockchain systems. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP)*. IEEE, 134–143.