# Local Virtual Times Analysis in PCS model

SCHOLARONE™
Manuscripts

# Local Virtual Times Analysis in PCS Model

Liliia Ziganurova, Lev Shchur *Member, IEEE, Fellow, APS*

*Abstract*—Efficient scalability and process synchronization are critical for achieving high performance in distributed computing environments. Analysis of the scalability is usually done using intensive case studies, which give an answer only for the particular set of model parameters. We found an efficient way to analyze the time evolution in models simulated with the Parallel Discrete Event Simulations (PDES) approach. The essential feature of PDES is the concept of local virtual time (LVT) associated with the evolution of each process of the model. The LVT of processes evaluates in simulations and forms a complicated profile. These profiles remind the profiles of the surface growth in the physical devices. In physics, researchers use the concept of universality, which helps to divide the different regimes of the class's surface growth—each class is described by some universal laws and does not depend on the details of the model. We demonstrate the applicability of this concept and present a model of LVT profile evolution in Personal Communication Service (PCS) model. The PCS network consists of a square grid of radio ports that serve users in their zone (cell). We build the LVT-PCS model, which describes the evolution of the LVT profile associated with the PCS model. We simulate the PCS model using the ROSS simulator (optimistic PDES) and compare results with those simulated by our LVT-PCS model. We found the profile demonstrates property, which is known in physics as roughening transition. We estimate the values of "critical" exponents for two models, which seem to belong to the same universality class. We believe that the similarity we found can be helpful for the preliminary analysis of the model scalability, process desynchronization, and possible deadlocks.

*Index Terms*—Local virtual time (LVT), LVT profile, optimistic synchronization, personal communication service (PCS), parallel discrete event simulation (PCS), ROSS

## I. INTRODUCTION

Parallel discrete event simulation (PDES) is a promising approach for the simulation of extreme-scale systems with a wide range of applications. Developing a new parallel simulation system or building it on top of an existing system is challenging due to the large set of parameters that affect the resulting performance. To achieve a parallel simulation performance, the developers should solve various fundamental problems: synchronization, load balancing, error recovery, and network overhead. Moreover, the PDES performance depends on the simulation environment, such as programming language, computer platform and infrastructure, processor type, and other characteristics. All these circumstances are biased, which makes prediction almost impossible. Each particular

The authors are with HSE University, 101000 Moscow, Russia

L. Ziganurova is with Science Center in Chernogolovka, 142432 Chernogolovka, Russia

L. Shchur is with Landau Institute for Theoretical Physics, 142432 Chernogolovka, Russia

system needs deep analysis and proper parameter selection before its large-scale simulation.

To gain more insight into the behavior of a simulation, developers commonly use *tracing*. Tracing records events of interest during program execution and store the history for later analysis. The method provides detailed information about a particular system running in a specific environment. However, tracing is not free from drawbacks. Firstly, tracing is memory-consuming. Each event must be recorded and analyzed while weighing terabytes of data in the typical simulation. Secondly, the problem is that tracing events adds perturbation to the simulation program and significantly slows down the simulation. In other words, tracing makes it hard to achieve an unbiased observation of the execution.

There are dozens of researches dedicated to enhancing and optimizing tools and simulation techniques [1], [2], performance analysis, and description of visualization tools [3], [4], [5], [6], [7]. The testing of tools with various sets of parameters and comparing them is a time-consuming and expensive technique.

In the paper, we present an approach that may predict the performance of a PDES system under given circumstances and without the direct investigation of the model behavior using the above-mentioned techniques. The approach is sometimes referred to as "simulation of simulation", or meta-simulation. We describe a general model of local virtual time evolution (LVT model) in optimistically synchronized parallel discrete-event simulation, capturing the essential properties of the actual simulation models.

For a given simulation PDES model, the LVT model may predict possible deadlocks, degree of scalability, and degree of desynchronization between parallel processes. It is important to stress that our approach gives a piece of complementary information to other performance analysis techniques rather than provides a very detailed view of the model under consideration.

The method of performance analysis using LVT profile has several advantages: 1) the LVT model describes a pattern of LVT profile growth in conservative or optimistic synchronization algorithm on a given communication topology, which make the model-independent on the context of the particular implementation of algorithms and simulation environment; 2) it makes possible to establish functional dependencies between the parameters of the model, such as means of distributions, number of logical processes (LPs), network topology, and the resulting efficiency of the computations. The theoretical predictions in the limit of an infinite number of LPs are reached for typical PDES models already for the size of $10^3$ – $10^4$ LPs.

The LVT models demonstrate similarities with the interface growth models in physics. The evolution of the LVT profile

in the conservative PDES algorithm belongs to Kardar-Parizi-Zhang universality class [8], and the LVT model's profile belongs to the universality class of directed percolation [9]. The present here the LVT-PCS, which belongs to the class of universality of random deposition [10] process, The theoretical predictions of such an approach can be highly applicable for a large class of existing models in PDES simulations.

The paper is organized as follows. A short description of the LVT is given in Section II. Works related to the efficiency analysis of the PDES models are reviewed in Section III. In Section IV, we present the validation of the LVT model by comparing simulation results in LVT and the model of Personal Communicational Service (PCS) for a widely used PDES simulator named ROSS [11]. Finally, the discussion and conclusion are presented in Section V.

## II. THE MODEL OF LOCAL VIRTUAL TIME PROFILE EVOLUTION

In the PDES framework [12], the model system is simulated as a set of independent logical processes that evolve their states along the time axis. The LPs generate time-stamped events and exchange them with each other. The events are stored in queues of receiving LPs, with time ordering. The LP processes an event, updating its Local Virtual Time (LVT) to the time of the processed event so that the LVT profile grows during the simulation. In the optimistically synchronized models, the time profile grows not only in the time direction but also goes back in time, using the rollback mechanism [13].

We use the following assumptions in the model of LVT profile evolution [9]:

- events are Poisson arrivals, so the time between them is exponentially distributed,
- graph of possible LP communications (dependences) is known in advance,
- number of rollbacks is given as a simulation parameter.

The simulation starts with a flat profile, $\tau_i = 0$, $i = 0..N-1$, where $N$ is a total number of LPs, $\tau$ is a local virtual time. Firstly, we increase LVTs of every LPs by exponentially distributed random value $\eta_i$ with unity mean:

$$\tau_i(t+1) = \tau_i(t) + \eta_i, i = 1, \ldots, N, \qquad (1)$$

$t$ is a simulation time step.

At this stage, we simulate the profile's growth, assuming that no causality violation has occurred. After this step, we simulate a rollback. The parameter $b$ is a mean rollback depth (number of rollbacks). We draw exponentially distributed random value $k$ with mean $b$, and then randomly choose $kN$ LPs, which local time will rollback. We simulate rollbacks as follows: if the LVT of an LP is less than the LVT of one of its neighbors (randomly chosen), then the LVT is reduced to the neighbor's time.

$$\tau_i = \begin{cases} \tau_r, \text{if } \tau_i > \tau_r \\ \tau_i, \text{otherwise,} \end{cases} \qquad (2)$$

where $r$ – the index of neighbouring LP.

After each time step we calculate the observables:

1) The average LVT $\tau(t)$:

$$\tau(t) = \frac{1}{N} \sum_{i=1}^{N} \tau_i(t). \qquad (3)$$

2) The average speed of the profile $v(t)$:

$$v(t) = \tau(t+1) - \tau(t). \qquad (4)$$

3) The average width of the profile $w^2(t)$:

$$w^2(t) = \frac{1}{N} \sum_{i=1}^{N} [\tau_i(t) - \tau(t)]^2. \qquad (5)$$

The final results are calculated as the average over $R$ independent runs of the random process:

$$\langle \tau(t) \rangle = \frac{\sum_R \tau(t)}{R}, \qquad (6)$$

$$\langle v(t) \rangle = \frac{\sum_R v(t)}{R}, \qquad (7)$$

$$\langle w^2(t) \rangle = \frac{\sum_R w^2(t)}{R}. \qquad (8)$$

In the language of computer science, the average speed of the profile $v$ is associated with the efficiency of the simulation, which is the average load of the parallel processes. The second observable, the average width of the profile $w^2$ reflects the desynchronization between the processing elements.

The model allows the analysis of the essential fundamental properties of the PDES algorithm. For example, the zero speed of the LVT profile illustrates the zero utilization of processing time or possible deadlocks. The growing width of the profile shows imperfect synchronization between LPs, which also decreases the efficiency. The more significant difference between LP's time is, the more rollbacks may occur. Divergent width may also illustrate an insufficient load balance (some LPs are always ahead of the others). The LVT profile model enables computation of the functional dependency between the number of rollbacks and parameters of communication topology between LPs. In [14], it was shown that the additional rare long-range communications between LPs in conservative PDES scheme significantly decreases the desynchronization.

As mentioned above, the model's distinctive feature is similar to the models of growing interfaces in physics. Interestingly, the model with entirely different rules forms so-called universality classes, which are characterized by the set of universal critical exponents. These exponents describe the model behavior near the critical point. The models of LVT profiles are known to belong to such universality classes [15], [16]. One may suggest that most of the variations of the Time Warp algorithm will not change the universality behavior of the systems.

## III. Related Work

Various approaches have been employed to study the performance of parallel discrete event simulations or other HPC systems. All of them may be classified into three groups:

1) Performance analysis based on tracing and visualization,
2) Performance analysis of modified PDES algorithms (f.e., GVT computation, optimism control, cancellation strategy),
3) Meta-simulation and analytical models.

### A. Performance analysis based on tracing and visualization

One approach to the analysis of parallel program performance is to analyze execution traces and provide visualization of collected log files. Various tools have been developed to perform such analyses. Usually, they support major parallel programming methodologies, not only PDES programs. Examples of such tools are VAMPIR [3], Jumpshot [17], TAU [18], Scalasca [5], and EXPERT [19]. They may build state diagrams, activity charts, timeline displays, and other statistics by a given trace file. Some of the tools, e.g., Jumpshot and TAU, may generate event traces that can be displayed with other special tools.

Some of the software, as Projections [6] or special visualization tool for ROSS [7], [11], is designed specially for PDES applications. They provide an additional view of the PDES program's execution, which may be compared with the capabilities of our LVT model. For example, one of the capabilities of Projection is to draw real- and virtual-time active LPs (see Figs.12-15 in [6]). Our model also measures the number of active LPs, assuming that active LP is a process that increases its local virtual time at a given step of the simulation. In [7] Figures 7 and 8 show the difference between virtual time and GVT for best and worst efficiency configurations. These pictures are similar to the LVT profile in our model. Moreover, it is seen that the width of the profile (or the spreading near the mean value) is larger for inefficient configuration and negligible for the best efficiency configuration. So, this illustrates that the observables we study in our models are interesting for PDES developers.

Most of the described tools collect statistics of events in terms of real-time. As an alternative, Ravel traces an event history transferred into logical time, inferred from happened-before relationships [4]. Such an approach leads to a much scalable representation of PDES applications without loss of communication and dependency relationships between processes. Our approach, however, is closer to a real-time tracing rather than logical, even though it stimulates the growth of local virtual (logical) times.

Some tools focus on particular aspects of parallel programs, such as memory consumption or load balancing. ScalaMem-Trace collects memory traces in order to detect memory inefficiencies [20]. Such analysis is also quite crucial for PDES efficiency. However, we do not consider any questions regarding memory, load balancing, and robustness in our model.

We refer readers to a review [21] for more information on the visualization tools. As stated in the paper, one of the challenges of visualization approaches is scalability. The scale of collected performance data grows exponentially, and it is essential and challenging to create scalable performance visualization on the contrary. Our LVT model aims to provide a scaled view on the growth pattern of LVTs, and it does not require large memory consumption.

### B. Modification of PDES algorithms

Another group of researches aimed to increase the performance of PDES computations focuses on developing new mechanisms and modifications of the original Time Warp algorithm. Implementation of such aspects as computing of GVT, cancellation strategy, and optimism control may differ in different simulators.

Optimism control is an approach of adding the restrictions into message communication or event schedule to reduce the number of rollback events (using a simulated time window, limiting the number of events each LP may execute beyond GVT, and sending only guaranteed correct messages). In [22] information about future incoming events is collected via additional communication between neighboring LPs (what restricts the natural parallelization) and then used for better synchronization. Another examples of control mechanism are Moving Time Window [23], Breathing Time Window [24], Probabilistic Cost Expectation Function Protocol [25], PADOC (Probabilistic Adaptive Direct Optimism Control) [26], and Switch Time Warp mechanism [27], [28]. Additionally, to the time window, it is possible to use speculative computations. For example, in [29], in speculative computations, nodes are allowed to execute future events being in the "idle" phase. When the synchronization is complete, the speculative computation is tested for correctness and either committed or discarded.

In many PDES models, only a tiny subset of simulation threads are active at a given period. This fact is sufficiently exploited in Demand-Driven PDES [30]. The key idea behind DD-PDES is to identify threads that have no events to process and exclude them from the CPU and GVT computing until they receive a special message. DD-PDES mechanism is orchestrated by a particular "controller" thread.

Another aspect of Time Warp implementation is related to the computing of Global Virtual Time. The computation of GVT is an expensive operation because usually, during the collection of LVTs, LP should stop the simulation. With the increasing number of shared memory multicore machines, various non-blocking algorithms have become popular. For instance, using a multi-thread PDES environment, it is possible to compute GVT without blocking operations [31].

Most of the PDES modifications can be mapped on the LVT model for the optimistic algorithm. However, some mechanisms may significantly affect the appearance of the LVT profile. We do not analyze, for example, the possibility of mapping the PDES modification reported in [32], where authors developed the optimism control technique with rolling back of all processes to GVT at stochastically selected real-time intervals.

## C. Meta-simulation and analytical models

Meta-simulation is a performance prediction approach based on the simulation of a PDES execution, or "simulation of simulation". This method provides a performance prediction without the need to develop and run the actual parallel discrete-event simulation. Meta-simulation allows implementing different PDES algorithms in a common environment to avoid time-consuming testing in a realistic simulation environment. Thus, such factors as a programming language, network structure, or model properties are significant. On the other hand, as meta-simulation uses many abstractions (i.e., negligible roll-back cost, zero-delay communication), the results of performance predictions may differ from real ones.

For example, SimSim [33] is a sequential simulator for parallel and distributed simulation systems. A real simulation model is executed on a set of modeled processors in Sim-Sim connected via a modeled network. Then the events are executed, as usual (considering the order of the size of the messages). Some factor then scales the resulting execution time for the baseline system. Similar technique is used in the Scalable Simulation Framework (SSF) [34], [35] and special performance analyzers [36], [37].

There exist also a purely theoretical way to analyze the PDES performance, for example, Markov models. In [38] the Markov model of the Time Warp algorithm was used in order to estimate such performance measures as the fraction of time the processors remain idle, the expected length of rollback, the expected number of processed uncommitted events, the expected number of processed events above the GVT, the effective message density, and the probability of rollback. The model results are in good agreement with PHOLD simulation; however, the model is hardly adaptable to a particular simulated system.

Another exciting and worth mentioning here reported in [39]. The goal of the work was to estimate the influence of the communication topology, the lookahead, and the computation and communication delays on the simulation performance using the analytical model of the null-message conservative algorithm. The performance metric was defined as the ratio between the simulation end time and the total run time for all LPs. The approach is similar to the one we use in this paper because it focuses on estimating the speed of the LPs' progress.

Our LVT model lies somewhere between the meta-simulation and theoretical category because it simulates not the whole PDES execution but instead specifies the local virtual time growth in particular synchronization PDES algorithms, basing on the set of theoretical assumption.

## IV. VALIDATION OF LVT MODEL

This section describes how to establish a relationship between the LVT model and other PDES models. We use the Personal Communication Service network model (PCS) [40], [41], [42] as an example.

### A. PCS network model

A personal communication service network is a network of distributed radio ports, each having a set of radio channels and
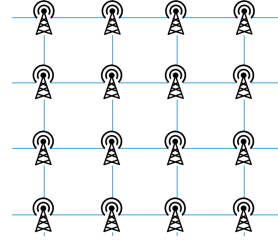


Fig. 1. Communication topology of PCS network model

users. The users send and receive phone calls by using these radio channels. When a user moves from one cell to another during a phone call, the network attempts to reallocate the call to the new cell (a port coverage area). If all channels in the new cell are busy, the phone call terminates.

The PCS model is implemented on ROSS simulator [11]. Let us describe the system in the language of logical processes and events. In this model, logical processes simulate the work of the radio ports (one LP = one radio port). Logical processes are located on the square lattice (Figure 1). Four different types of events are implemented in the system: 1) *NextCall* – a call arrival at a cell; 2) *CompletionCall* – a completed call at a cell; 3) *MoveOut* – a call moving out of the current cell (remote event); 4) *MoveIn*, which denotes the arrival of a hand-off call at a cell. In PCS, the time between events is exponentially distributed. The following parameters set the means of the distributions:

- MOVE_CALL_MEAN – the mean time between *MoveOut* events
- NEXT_CALL_MEAN – the mean time between *NextCall* events
- CALL_TIME_MEAN – the mean call duration.

In the PCS model the most events are processed locally by the LPs, but some events, i.e., of type *MoveOut*, are generated by LPs for other LPs. Such events, traversing between LPs, are called remote events. The percentage of remote events is available in the output statistics. We denote this percentage by $p'$. In other words, the LPs are located on the square lattice with periodic boundary conditions, and the interaction between the closest LPs occurs with the probability $p'$.

Besides the number of remote events $p'$, the ROSS statistics provides such data as total events processed, the number of rolled back events, and the event rate. Moreover, to compare precisely the behavior of the LVTs, it is possible to trace the LVTs during the simulation.

It is important to note that the statistics of the computations is collected not by each LP but based on specially introduced Kernel Processes (KPs), which aggregate a group of LPs in order to process event-list for those LPs as a single list, and make the GVT computation and fossil collection more effective [11].

### B. Parameters mapping

In the LVT model, we have two major parameters: the growth rate $q$ and the parameter of communication topology

$p$, which regulates the number of interprocess communications (remote events). The growth rate $q$ is defined by the formula:

$$q = \frac{1}{1+b}, \qquad (9)$$

$b$ is the average "length" of rollbacks provided that the average increase of the profile is equal to unity. The parameter may be interpreted as the number of processed events divided by the sum of processed and rollbacked events.

Most of the events are processed locally on the generated LP in the PCS network model, but sometimes LPs may send events to the neighboring cells. Such remote interactions between LPs occur naturally when and events of type *MoveIn* or *MoveOut* are generated. The percentage of remote events is not a parameter of the system but the statistical output of the simulation. We denote it by $p'$ to emphasize the connection with the parameter $p$ of the LVT model. Fraction of remote events $p'$ depends on the number of *MoveOut* events, and it is inversely with the parameter MOVE_CALL_MEAN.

The analogue of the growth rate $q$, in the PCS network model we define a number $q'$:

$$q' = 1 - \frac{\text{number of rollbacks}}{\text{total number of processed events}} \qquad (10)$$

Both $q'$ and $p'$ are biased variables, depending on the parameter MOVE_CALL_MEAN.

### C. LVT-PCS model

We define the LVT-PCS model as the particular case of the LVT model described in section II. The central point in mapping the PCS model to the LVT-PCS model is identifying the parameters $p, q$ necessary for the LVT-PCS model simulation with the parameters $p', q'$ extracted from the PCS model simulation.

The protocol of simulations is following

1) simulate the PCS network model with parameter MOVE_CALL_MEAN choosing from the range [300, 4500],
2) trace the LVTs during the simulation,
3) save the corresponding values of $q'$ and $p'$,
4) simulate the LVT model with $p = p'$ and $q = q'$,
5) compare the average speed of the profile in LVT with the simulation performance in PCS,
6) compares the average speed and the average width of the profile in both models.

We simulate the PCS network model using 64 cores of a computing node with two Intel(R) Xeon Platinum 8164 2.0 GHz processors and 2x768 GB 2666MHz DDR4 onboard memory. The number of LPs and KPs is equal to 256. We study how the average local virtual time grows during the simulation. Thus, the average speed of the profile is calculated as a slope of the function $LVT(t)$, where $t$ is a batch of processed events equal to 256 per LP. In ROSS, the statistic is produced at the moment of GVT computation and other serving computations. The average width of the profile is calculated by formula (5) and then averaged over $t$ after some cutoff time. The final result is also average over ten independent runs.

### D. Results

The main result of our simulation is that the behavior of LVT profiles in both models is qualitatively similar. The more remote events, the lower the speed of the LVT growth, event rate, and efficiency, and the average width of the profile. The comparison of the average speed of the profile in LVT and PCS models and event rate in the PCS model is given in Figures 3 and 2. The average speed in the LVT model reflects the utilization of events in PCS models (i.e., event rate).

We simulate the PCS model with ROSS simulator [11]. The dependence of the event rate $ER$ as a function from the value of parameter $q'$ is shown in the Figure 2. We fit event rate data using expression $ER = ER_0(q' - q'_c)^\nu + \text{const}$, and estimate values of $q'_c = 0.100 \pm 0.001$ and $\nu = 1.5 \pm 0.1$.
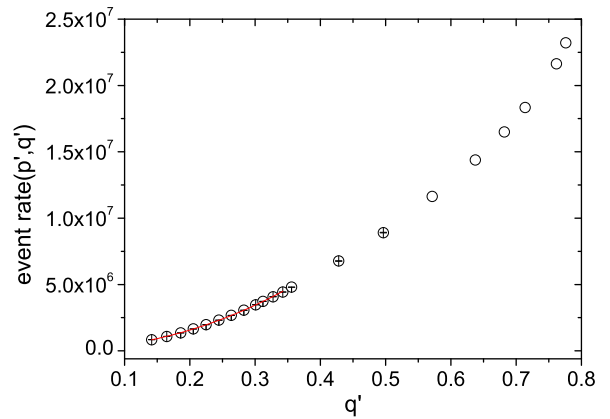


Fig. 2. The event rate in PCS model as a function of rollback frequency.

We use pairs of values $p', q'$ extracted from PCS model simulation to simulate LVT-PCS model with pairs of parameters $p, q$ equal to $p', q'$.

Figure 3 demonstrate the resulting dependence of the profile speed average as a function of the parameter $q$. There are three ranges of $q$ with different speed dependence. Firstly, it is zero velocity below value $q_c \approx 0.132 \pm 0.005$, which is the roughening critical point [43]. Secondly, we use approximation of the profile using expression $v = v_0(q - q_c)^\nu + \text{const}$ of the data in the range $q_c < q < 0.3$. We estimate value of the exponent $\nu = 1.5 \pm 0.01$, which is smaller than in the directed percolation model, $\nu_{DP} \approx 1.73$) Ref. [43].

Thirdly, it is the regime of the random deposition in close to value of 0.8, in which profile growth randomly at each LPs, and average speed reach value one, the maximum possible profile speed.

We found that value of exponent $\nu$ in PCS model and LVT-PCS model is close to 1.5, and the critical value of $q_c$ is around 0.1. Therefore, the LVT-PCS model reflects critical properties of PCS model which was now known before our analysis using similarity of the profile with those in the statistical physics of surface growth. We have to note the value of exponent is smaller than those of the physical models because of the particular topology of PCS model.
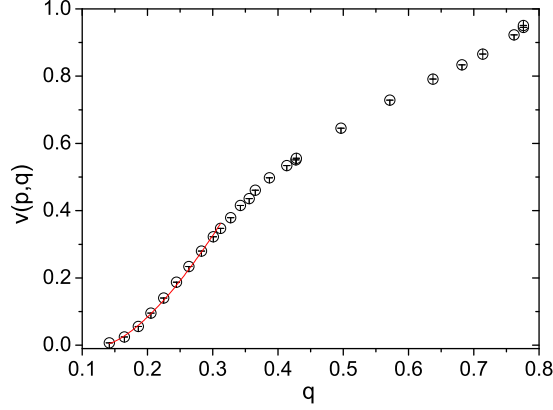
Fig. 3. The average speed of the LVT profile in LVT model as a function of $q$.



a) The average speed as a function of $p'$.

We also check how values of the profile speed and event rate depend on the value of the parameter $p'$ which reflects the random selection of the neigbour cite (cell) in PCS model. Figures **??** and 4 show the dependence of the average LVT speed and event rate on the fraction of remote events $p$ and $p'$. The plots are looks similar.

So, the analysis shows that the average speed of the profile in the LVT model indeed corresponds to such popular measure of PDES efficiency as event rate [44], [45]. Slight differences in the behavior of these observables may be explained by the presence of additional correlations in the actual models, which are not taken into account in the LVT model.

The average width of the profile in both models is qualitatively similar (Figures 5, 6), even though in the PCS network mode. As was stated above, the average width of the LVT profile reflects the desynchronization between logical processes. In actual systems, it is possible to estimate the desynchronization by tracing the local virtual times. This characteristic is not very meaningful itself. However, it correlates with the efficiency of the simulation. When the average width of the profile is low, the speed of the profile (i.e., the efficiency) is high, and vise versa. Our results confirm this fact.

## V. Discussion

We described a possible approach to performance analysis of PDES systems based on the local virtual time profile evolution simulation. The model captures the essential properties of PDES and may predict the possible deadlocks, efficiency behavior, and desynchronization depending on the distribution means and communication topology.

The LVT-PCS model was validated by comparison with the PCS network model running on the ROSS simulator. The simulations of LVT-PCS and PCS models with the same input parameters demonstrate qualitative similarities in LVT evolution. The average speed of the profile in the LVT-PCS model indeed displays the efficiency of the PCS model or the event rate. The PCS model's efficiency approaches zero when the parameter $q'$ is close to its critical value $q'_c$. The same behavior is observed for the average speed in the LVT-PCS
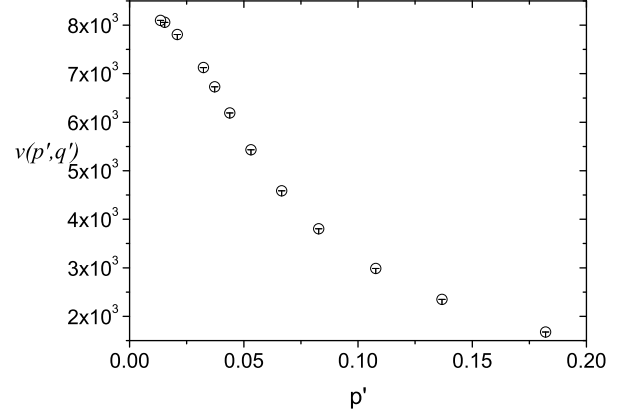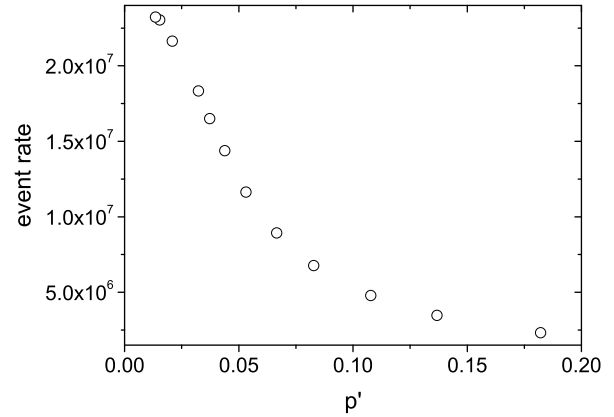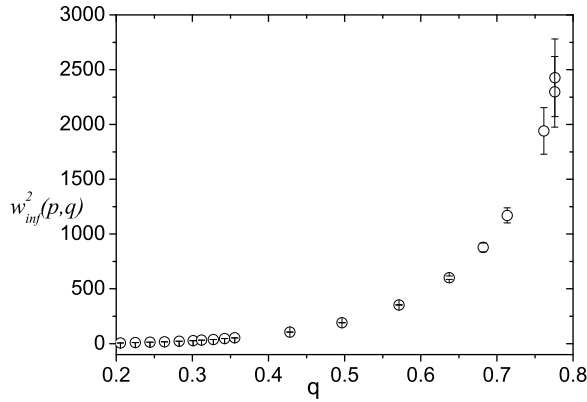


b) Event rate as a function of $p'$.

Fig. 4. The average LVT speed and event rate as a function of remote events $p'$ in PCS model.
miem

model. Both models undergo the roughening phase transition characterized with the approximately same valued of the critical exponent.
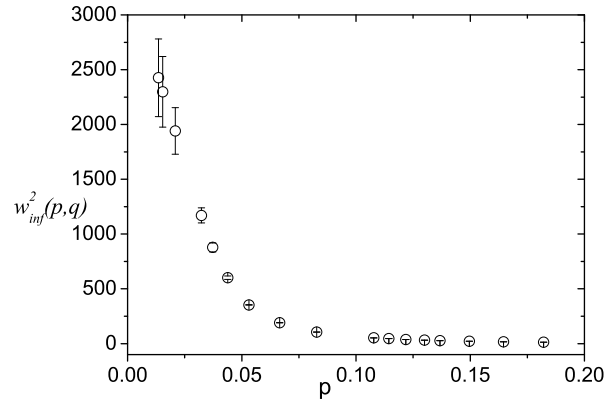
The research showed the consistency of the LVT-PCS model. To analyze particular PDES model performance with the LVT modeling, one needs to choose the proper network parameters and distribution means and then build the mapping to the LVT model and construct the proper simulation protocol. Thus build model can be used for the analysis of properties with a wide range of parameters. Of course, matching the parameters between the PDES model and LVT models is nontrivial and requires accurate analysis of a particular PDES model. We hope our approach can be helpful for the community, both in practice and in theory.
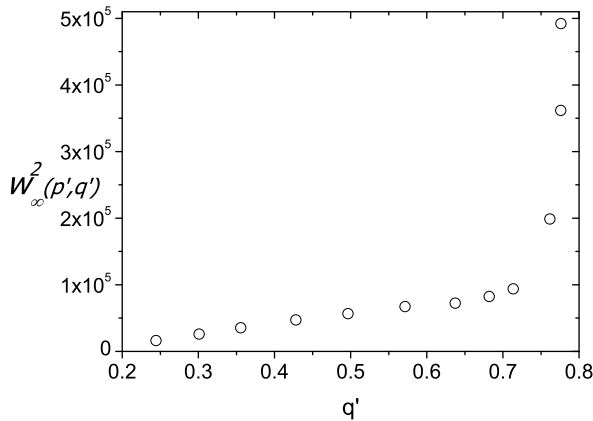
## VI. Acknowledgements

a) The average width of the profile in the LVT model as a function of $q$.



b) The average width of the profile in the PCS network model as a function of $q'$.

Fig. 5. Comparison of the average width in LVT and PCS network models depending on the rollback probability.
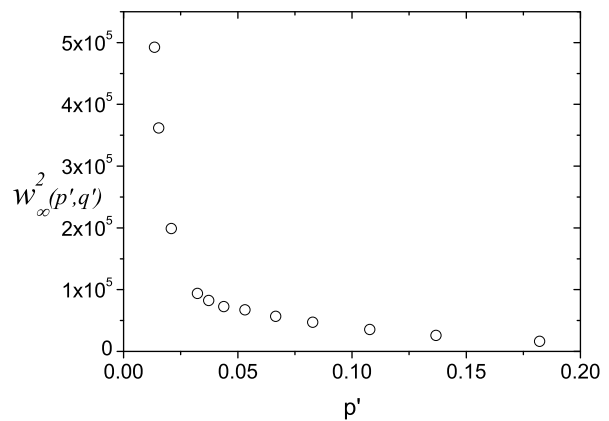


a) The average width of the profile in the LVT model as a function of $p$.



b) The average width of the profile in the PCS network model as a function of $p'$.

Fig. 6. Comparison of the average width in LVT and PCS network models depending on the fraction of remote events.

## References

[1] S. R. Das, "Adaptive protocols for parallel discrete event simulation," *Journal of the operational Research Society*, vol. 51, no. 4, pp. 385–394, 2000.

[2] S. Rahman, N. Abu-Ghazaleh, and W. Najjar, "Pdes-a: Accelerators for parallel discrete event simulation implemented on fpgas," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 29, no. 2, pp. 1–25, 2019.

[3] W. E. Nagel, A. Arnold, M. Weber, H.-C. Hoppe, and K. Solchenbach, "Vampir: Visualization and analysis of mpi resources," 1996.

[4] K. E. Isaacs, P.-T. Bremer, I. Jusufi, T. Gamblin, A. Bhatele, M. Schulz, and B. Hamann, "Combing the communication hairball: Visualizing parallel execution traces using logical time," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 2349–2358, 2014.

[5] M. Geimer, F. Wolf, B. J. Wylie, E. Ábrahám, D. Becker, and B. Mohr, "The scalasca performance toolset architecture," *Concurrency and Computation: Practice and Experience*, vol. 22, no. 6, pp. 702–719, 2010.

[6] C. W. Lee, T. L. Wilmarth, and L. V. Kalé, "Performance visualization and analysis of parallel discrete event simulations with projections," *Parallel Programming Laboratory, Department of Computer Science, University of Illinois at Urbana-Champaign, Tech. Rep.*, pp. 05–19, 2005.

[7] C. Ross, C. D. Carothers, M. Mubarak, P. Carns, R. Ross, J. K. Li, and K.-L. Ma, "Visual data-analytics of large-scale parallel discrete-event simulations," in *2016 7th International Workshop on Performance Mod-*

*eling, Benchmarking and Simulation of High Performance Computer Systems (PMBS)*. IEEE, 2016, pp. 87–97.

[8] G. Korniss, M. A. Novotny, A. Kolakowska, and H. Guclu, "Statistical properties of the simulated time horizon in conservative parallel discrete-event simulations," in *Proceedings of the 2002 ACM symposium on Applied computing*, 2002, pp. 132–137.

[9] L. Ziganurova, M. Novotny, and L. Shchur, "Model for the evolution of the time profile in optimistic parallel discrete event simulations," in *Journal of Physics: Conference Series*, vol. 681, no. 1. IOP Publishing, 2016, p. 012047.

[10] J. W. Evans, "Random-deposition models for thin-film epitaxial growth," *Physical Review B*, vol. 39, no. 9, p. 5655, 1989.

[11] C. D. Carothers, D. Bauer, and S. Pearce, "Ross: A high-performance, low-memory, modular time warp system," *Journal of Parallel and Distributed Computing*, vol. 62, no. 11, pp. 1648–1669, 2002.

[12] R. M. Fujimoto, "Parallel discrete event simulation," *Communications of the ACM*, vol. 33, no. 10, pp. 30–53, 1990.

[13] D. R. Jefferson, "Virtual time," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 7, no. 3, pp. 404–425, 1985.

[14] G. Korniss, M. A. Novotny, H. Guclu, Z. Toroczkai, and P. A. Rikvold, "Suppressing roughness of virtual times in parallel discrete-event simulations," *Science*, vol. 299, no. 5607, pp. 677–679, 2003.

[15] L. Ziganurova and L. N. Shchur, "Synchronization of conservative parallel discrete event simulations on a small-world network," *Physical Review E*, vol. 98, no. 2, p. 022218, 2018.

[16] L. Shchur and L. Ziganurova, "Synchronization of processes in parallel

[16] discrete event simulation," *Journal of Experimental and Theoretical Physics*, vol. 129, no. 4, pp. 722–732, 2019.

[17] O. Zaki, E. Lusk, W. Gropp, and D. Swider, "Toward scalable performance visualization with jumpshot," *The International Journal of High Performance Computing Applications*, vol. 13, no. 3, pp. 277–288, 1999.

[18] S. S. Shende and A. D. Malony, "The tau parallel performance system," *The International Journal of High Performance Computing Applications*, vol. 20, no. 2, pp. 287–311, 2006.

[19] F. Wolf and B. Mohr, "Automatic performance analysis of hybrid mpi/openmp applications," *Journal of Systems Architecture*, vol. 49, no. 10-11, pp. 421–439, 2003.

[20] S. Budanur, F. Mueller, and T. Gamblin, "Memory trace compression and replay for spmd systems using extended prsds," *The Computer Journal*, vol. 55, no. 2, pp. 206–217, 2012.

[21] K. E. Isaacs, A. Giménez, I. Jusufi, T. Gamblin, A. Bhatele, M. Schulz, B. Hamann, and P.-T. Bremer, "State of the art of performance visualization." in *EuroVis (STARs)*, 2014.

[22] J. Lindén, P. Bauer, S. Engblom, and B. Jonsson, "Exposing interprocess information for efficient parallel discrete event simulation of spatial stochastic systems," in *Proceedings of the 2017 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 2017, pp. 53–64.

[23] L. M. Sokol, J. B. Weissman, and P. A. Mutchler, "Mtw: An empirical performance study," Institute of Electrical and Electronics Engineers (IEEE), Tech. Rep., 1991.

[24] J. S. Steinman, "Breathing time warp," in *Proceedings of the seventh workshop on Parallel and distributed simulation*, 1993, pp. 109–118.

[25] A. Ferscha and J. Luthi, "Estimating rollback overhead for optimism control in time warp," in *Proceedings of Simulation Symposium*. IEEE, 1995, pp. 2–12.

[26] A. Ferscha, "Probabilistic adaptive direct optimism control in time warp," in *Proceedings of the ninth workshop on Parallel and distributed simulation*, 1995, pp. 120–129.

[27] R. Suppi, F. Cores, M. SERRANO, and E. Luque, "Stw: Switch time warp: A model for rollback reduction in optimistic pdes," in *Parallel Computing: Fundamentals and Applications*. World Scientific, 2000, pp. 400–407.

[28] R. Suppi, F. Cores, and E. Luque, "Pdes: A case study using the switch time warp," in *European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting*. Springer, 2001, pp. 327–334.

[29] M. Venu and I. Joe, "Improving performance of optimistic simulation for distributed simulation system using speculative computation," in *2014 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2014, pp. 428–432.

[30] A. Eker, B. Williams, K. Chiu, and D. Ponomarev, "Demand-driven pdes: Exploiting locality in simulation models," in *Proceedings of the 2020 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 2020, pp. 39–48.

[31] M. Ianni, R. Marotta, A. Pellegrini, and F. Quaglia, "A non-blocking global virtual time algorithm with logarithmic number of memory operations," in *2017 IEEE/ACM 21st International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*. IEEE, 2017, pp. 1–8.

[32] K. Jones and S. R. Das, "Combining optimism limiting schemes in time warp based parallel simulations," in *1998 Winter Simulation Conference. Proceedings (Cat. No. 98CH36274)*, vol. 1. IEEE, 1998, pp. 499–505.

[33] R. Ewald, J. Himmelspach, A. M. Uhrmacher, D. Chen, and G. K. Theodoropoulos, "A simulation approach to facilitate parallel and distributed discrete-event simulator development," in *2006 Tenth IEEE International Symposium on Distributed Simulation and Real-Time Applications*. IEEE, 2006, pp. 209–218.

[34] J. Liu, D. Nicol, B. Premore, and A. Poplawski, "Performance prediction of a parallel simulator," in *Proceedings Thirteenth Workshop on Parallel and Distributed Simulation. PADS 99.(Cat. No. PR00155)*. IEEE, 1999, pp. 156–164.

[35] D. M. Nicol, D. Jin, and Y. Zheng, "S3f: The scalable simulation framework revisited," in *Proceedings of the 2011 Winter Simulation Conference (WSC)*. IEEE, 2011, pp. 3283–3294.

[36] C.-C. Lim, Y.-H. Low, B.-P. Gan, S. Jain, W. Cai, W. J. Hsu, and S. Y. Huang, "Performance prediction tools for parallel discrete-event simulation," in *Proceedings Thirteenth Workshop on Parallel and Distributed Simulation. PADS 99.(Cat. No. PR00155)*. IEEE, 1999, pp. 148–155.

[37] Z. Juhasz, S. Turner, K. Kuntner, and M. Gerzson, "A performance analyser and prediction tool for parallel discrete event simulation," *Simulation*, vol. 8, p. 4, 2003.

[38] A. Gupta, I. F. Akyildiz, and R. M. Fujimoto, "Performance analysis of time warp with multiple homogeneous processors," *IEEE Transactions on Software Engineering*, vol. 17, no. 10, p. 1013, 1991.

[39] C.-H. Li, A. J. Park, and E. Schenfeld, "Analytical performance modeling for null message-based parallel discrete event simulation," in *2011 IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*. IEEE, 2011, pp. 349–358.

[40] C. D. Carothers, R. M. Fujimoto, Y.-B. Lin, and P. England, "Distributed simulation of large-scale pcs networks," in *Proceedings of International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*. IEEE, 1994, pp. 2–6.

[41] Y.-B. Lin and P. A. Fishwick, "Asynchronous parallel discrete event simulation," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 26, no. 4, pp. 397–412, 1996.

[42] C. D. Carothers, R. M. Fujimoto, and Y.-B. Lin, "A case study in simulating pcs networks using time warp," *ACM SIGSIM Simulation Digest*, vol. 25, no. 1, pp. 87–94, 1995.

[43] U. Alon, M. Evans, H. Hinrichsen, and D. Mukamel, "Roughening transition in a one-dimensional growth process," *Physical review letters*, vol. 76, no. 15, p. 2746, 1996.

[44] P. A. Wilsey, "Some properties of events executed in discrete-event simulation models," in *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 2016, pp. 165–176.

[45] P. Crawford, S. J. Eidenbenz, P. D. Barnes, and P. A. Wilsey, "Some properties of communication behaviors in discrete-event simulation models," in *2017 Winter Simulation Conference (WSC)*. IEEE, 2017, pp. 1025–1036.

**Liliia Ziganurova** received the MCs degree in applied mathematics from the HSE University in 2015 and the PhD degree in computer science from the HSE University in 2020. She is an associate professor in the Applied Mathematics Department of HSE University and a postdoctoral fellow at the Scientific Center in Chernogolovka, Chernogolovka, Russia. Her research interests include parallel and distributed systems and molecular dynamics. She has published more than ten journal and conference papers.

**Lev Shchur** (Member, IEEE), Computational Physics Group, Landau Institute for Theoretical Physics, Chernogolovka, Russia. Lev Shchur received the MCs degree in physics and computer sciences from Gorky State University in 1976 and the PhD degree in physics from the Landau Institute for Theoretical Physics in 1983. He is a principal researcher and Landau Institute and a professor at HSE University (Moscow). He is the Commission's secretary on Computational Physics (C20) of IUPAP (International Union of Pure and Applied Physics). His current research focused on high-performance algorithms and methods in computational physics, machine learning in physical problems, and software for high-performance computing in physics. He has co-authored more than 150 journal and conference research papers. He is a fellow of the American Physical Society.

**PAPER • <span style="color:red">OPEN ACCESS</span>**

# Local time evolution in Personal Communication Service model

To cite this article: Liliia Ziganurova and Lev Shchur 2021 *J. Phys.: Conf. Ser.* **1740** 012005

View the article online for updates and enhancements.
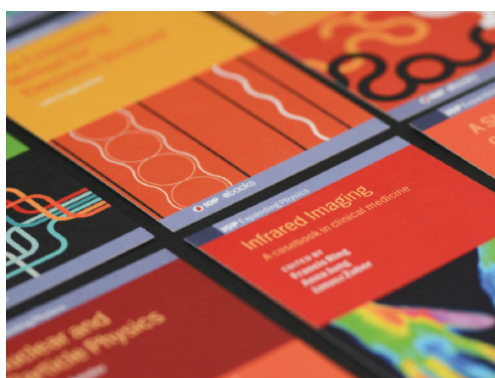
# Local time evolution in Personal Communication Service model

**Liliia Ziganurova[1,2] and Lev Shchur[1,3]**

[1]National Research University Higher School of Economics, 101000 Moscow, Russia
[2]Science Center in Chernogolovka, 142432 Chernogolovka, Russia
[3]Landau Institute for Theoretical Physics, 142432 Chernogolovka, Russia

**Abstract.** We investigate the local time evolution in the Personal Communication Service (PCS) model simulated with the parallel discrete event simulation method's optimistic algorithm. We propose a model for the optimistic local virtual time evolution (OLVT) in PCS, which is reminiscent of statistical physics's surface growth. We use Rensselaer's optimistic simulation system with the Time Warp implementation. We compare the results of the simulations of both PCS and OLVT models and found good agreement. We discuss the highlights of our approach in the analysis of scalability and synchronization using the OLVT model.

## 1. Introduction

Parallel Discrete Event Simulation (PDES) [1] is a large-scale simulation technique used in a wide range of fields, from physics and engineering to biology and social sciences. The discrete-event model contains a description of the system's status and the events that can occur in it. Events occur at random time moments and change the system's state (for example, in a magnetic system, an event is a spin-flip; and in the logistics, the event is a change of the status of delivery).

Today, many software platforms (simulators) for PDES facilitate using the method for solving applied problems. Among them, the most famous are ROSS [2], WARPED2 [3], Simian [4], SPEEDS [5], PDEVS [6], and NS-3 [7, 8]. The main functionality of PDES simulators is the event planner, i.e., organizing the execution of events by logical processes and receiving and transmitting messages containing the event information, and providing the other details of the simulation process. The successful application of the optimistic PDES simulator was demonstrated on almost two million cores [9].

Simulators' critical features are the scalability of parallel simulations and the synchronization of the processes [1, 10, 11]. Analysis of modeling profiles is successfully used to optimize modeling tools and algorithms.

In the paper, we present *the model of the model*, i.e., the model of local virtual time evolution (OVLT) of the optimistic simulator running the particular model of interest. The OLVT model can capture the essential properties of the real simulation model and provide information on the — i) possible lockdown; ii) on the scalability, and iii) on the desynchronization of parallel processing units depending on the parameters of simulations. We should stress that information is valid for a particular model and a wide range of parameters. In contrast to the widely used modeling profiles for each particular model and particular values of parameters, the proposed

approach provides information on the particular model's behavior in a wide range of parameters. We do not pretend on a very detailed analysis of the algorithm performance and provide additional information on the simulator's general behavior running the particular model, which can be informative for the general understanding performance. In other words, we do not pretend that our approach is superior to any other approaches or can substitute them. Instead, our point is to present the new and complementary way of algorithm analysis.

We present the preliminary results of applying our approach to the Personal Communication Service (PCS) model. The results of the PCS simulations are in good agreement with the OLVT model. Both models undergo a roughening transition.

Our approach is based on the previous analysis of the models of conservative algorithm [12] and optimistic algorithm [13] and models on the small-world networks [14, 15, 16]. In contrast to these studies, we build the OLVT model of optimistic simulation of a particular model — the PCS model. The new ingredient is the simulation of the model for the specific pair of two parameters in the model, which is correlated with the PCS model.

## 2. PCS model

PCS (personal communication service) model simulates a wireless communication network that provides communication services for mobile units [17, 18, 19]. The service area of the network is partitioned into checkerboard areas (cells). The simulation model consists of cell objects and portable objects associated with mobile devices. Each cell object represents a cellular receiver/transmitter with a fixed number of channels. The portable object represents a mobile unit that resides within the cell for some time and can move during the call to one of the four neighboring cells. The portable object's events are call arrival, call completion, and move to a neighboring cell. When a portable device moves to the neighboring cell, the currently allocated channel must be freed, and the destination cell object must allocate a channel to the portable object. It happens practically instantaneously.

The model has several customizable parameters, and typically the number of portables is much larger than the number of cells. The most important are the parameters that specify the mean values of the distributions:

- `MOVE_CALL_MEAN` - the average time between mobile devices moving from one cell to another,
- `NEXT_CALL_MEAN` - the next call average,
- `CALL_TIME_MEAN` - the average call length.

We run the PCS model using the ROSS simulator (Rensselaer's optimistic simulation systems) [2]. The ROSS simulator is a parallel discrete event simulation system based on the Time Warp synchronization algorithm [20] and is widely used in the scientific community [21, 22, 23, 24, 25].

The ROSS simulator's output contains the total number of events processed, the total number of rollbacks, the number of events processed per second (event rate), the number of events with the portable moves to the neighboring cell. In our case, it is essential to get in the output the local virtual times of processes. However, it can be done when calculating the global virtual time (GVT), which happens after 256 events processed by the default setting.

## 3. OLVT model

In this section, we introduce the *model* of the local virtual (LVT) time evolution in PCS *model* simulated with an optimistic algorithm - the OLVT model. It is based on the ideas of the models of local time evolution in the conservative PDES algorithm [12], FaS algorithm [26], and optimistic algorithm [13].

In the OLVT model, we associate each cell object with the logical process ($LP_i$) and initiate the flat profile of the local virtual time $\tau_i$ of each $LP_i$ ($i = 1, 2, \ldots, N$).

The simulation of the LVT with an optimistic algorithm consists of two essential parts. The first part is the evolution of the system for some time, assuming there is no causality violation – it is the forward computations. The second part is the check of causality violation with rolling back those events, which violates the time order – it is the rollback computations.

To model the forward part in OLVT we use the same approach as in Refs. [12, 26, 13] computing the increment of LVT with the exponentially distributed random variable $\eta_i$ with the mean equal to unity

$$\tau_i(t + 1) = \tau_i(t) + \eta_i, \tag{1}$$

where $i = 1, 2, \ldots, N$. The wall time $t$ in the OLVT model is measured in the units of the Time Warp window [2].

To model the rollback part in OLVT, we choose randomly from the exponential distribution with the mean $b$ the rollback depth value $k$ and choose $kN$ LPs randomly. For each chosen LP, check the rollback condition with the following law.

With probability $p$ the LVT $\tau_j(t+1)$ of the chosen $LP_j$ will remain the same with probability $(1 - p)$ and will be updated with probability $p$

$$\tau_j(t + 1) = \quad \begin{array}{l} \tau_r(t + 1), \ \ \text{if } \tau_j(t + 1) > \tau_r(t + 1) \\ \tau_j(t + 1), \ \ \text{otherwise,} \end{array} \tag{2}$$

where $\tau_r(t + 1)$ is LVT of the one of the nearest neighbors chosen randomly. The LVT update corresponds to the event of the active mobile device moving from the cell $r$ to the cell $j$.

We should stress that the rollback law in the OLVT model is different from the one in the optimistic LVT model [13], and use some ideas of the LVT model defined on the small-world network in Ref. [16].

## 4. Parameters and observables
### 4.1. Parameters in OLVT model
The parameter of OLVT model defined in line with the optimistic LVT model [13] as the ratio of the forward increment to the sum of the forward and rollback increments

$$q = \frac{1}{1 + b}, \tag{3}$$

which takes the limiting values 1 and 0 while $b$ varies from 0 to $\infty$.

Analyzing the simple LVT evolution model in the paper [13] we found that at a low enough value of parameter $q$, the LVT profile becomes flat and does not increase the mean value. For a large enough value of the parameter $q$, the LVT profile becomes rough and propagates with the wall time $t$, and the mean value of the LVT profile becomes proportional to the wall time $t$. There is some critical value $q_c$ separating the two regimes, the value of the roughening transitions known in statistical physics [27] (for details, we refer the reader to reviews [28, 29]). The critical feature of the roughening transition is that the LVT profile behavior near the critical value $q_c$ is universal, giving the same law of the mean LVT profile dependence from the parameter $q$, and the same law of the mean LVT profile width dependence from the number of cells $N$. In the language of computer science, the first dependence is connected to the efficiency of simulations (the loading of the processing elements), and the second dependence is connected to the synchronization of the processing elements.

In addition to the parameter $q$, we already defined the parameter $p$ for the OLVT model as the probability of the event describing the active mobile device moving to the nearest cell.

*4.2. Parameters in PCS model*

Let us introduce the parameter $q'$

$$q' = 1 - \frac{\#\,\texttt{rollbacks}}{\#\,\texttt{events}} \tag{4}$$

which defined through the relative number of rollbacks ($\#\,\texttt{rollbacks}$) to the number of the processed events ($\#\,\texttt{events}$).

As we will see, the parameter $q'$ plays the same role as the parameter $q$ in the OLVT model.

In the PCS model, the parameter $p'$ has the same meaning as the parameter $p$ in the OLVT model. We keep different notations for $p, p'$ and $q, q'$ because in PCS model parameters $p'$ and $q'$ is not set before simulations as in OLVT model and rather computed in PCS simulation. The parameters $p'$ and $q'$ depend on the value of ROSS parameter `MOVE_CALL_MEAN`, which is the average time between calls that leave the tower's coverage area.

*4.3. Observables*

We calculate time-dependent averages at each time step $t$ and after simulations average them over $M = 10$ realizations of the process in the PCS model and $M = 1000$ in the OLVT model using the following definitions

(i) The mean LVT profile

$$\langle \tau(t) \rangle = \frac{1}{M} \sum_{j=1}^{M} \left( \frac{1}{N} \sum_{i=1}^{N} \tau_i(t) \right)_j, \tag{5}$$

(ii) The mean velocity

$$\langle v(t) \rangle = \frac{1}{M} \sum_{j=1}^{M} \left( \tau(t+1) - \tau(t) \right)_j, \tag{6}$$

(iii) The mean LVT profile square width

$$\left\langle w^2(t) \right\rangle = \frac{1}{M} \sum_{j=1}^{M} \left( \frac{1}{N} \sum_{i=1}^{N} [\tau_i(t) - \tau(t)]^2 \right)_j. \tag{7}$$

**5. Simulations**

*5.1. The profile evolution*

The PCS model was simulated on 64 cores of the computing node R2D26 computer. The number of logical processes and the number of kernel processes was $N = 256$. The parameter `MOVE_CALL_MEAN` sets the average time between calls moving from one network coverage area to another.

We use values of $p', q'$ calculated in the PCS model simulations as the values of parameters $p, q$ in OLVT simulations. The data presented in Table 1. Below, we use the notations $v(p, q)$ for the profile velocity in the OLVT model and similar notations for the profiles' square width.

The OLVT model was simulated on the R2D26 computer. We run each model with several logical processes $N = 256$ and different parameters $p$ and $q$ independently on the available cores to average over random realizations.

Mean OLVT profile velocity $\langle v(t) \rangle$ saturated after some transition time, forgetting the initial flat state. We name this velocity the steady-state velocity $v(p, q)$. The steady-state velocity does depend on the value of parameter $q$ defined by expression 3) and is shown in Figure 1-a. In the figures, each point supplied the error bars, computed with the number of realizations. The size of error bars is typically less than the symbols. The velocity vanishes at some value

**Table 1.** Parameters of the simulations.

| MOVE_CALL_MEAN | $p'$ | $q'$ |
|---|---|---|
| 300 | 0.1821 | 0.24 |
| 450 | 0.1368 | 0.30 |
| 600 | 0.1078 | 0.36 |
| 800 | 0.0828 | 0.43 |
| 1000 | 0.0666 | 0.50 |
| 1250 | 0.0531 | 0.57 |
| 1500 | 0.0439 | 0.64 |
| 1750 | 0.0373 | 0.68 |
| 2000 | 0.0323 | 0.71 |
| 3000 | 0.0209 | 0.76 |
| 4000 | 0.0154 | 0.7759 |
| 4500 | 0.0135 | 0.7761 |
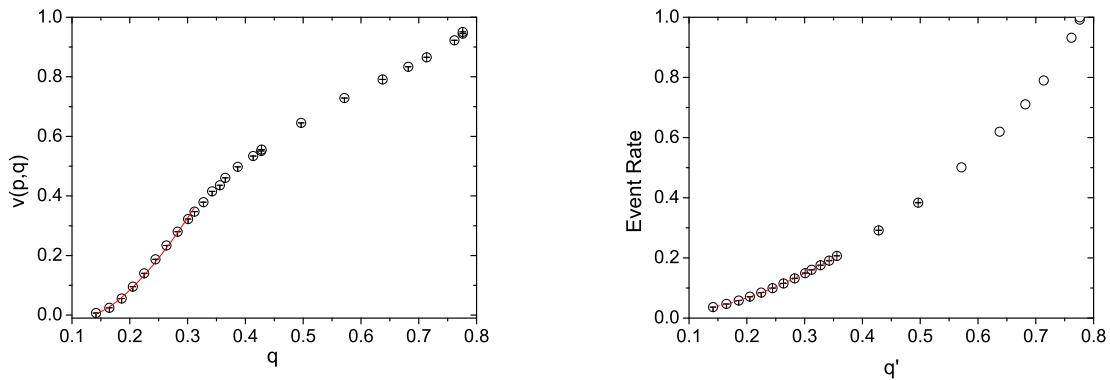


a) Steady-state velocity in OLVT as function of parameter $q$.



b) Normalized event-rate in PCS model as function of parameter $q'$.

**Figure 1.** Critical behaviour of the a) steady-state velocity $v(p,q)$ in OLVT model and b) normalized event-rate in PCS model.

$q_c$, which is the critical point of the roughening transition [28, 29]). The approximation of the $v$ near the $q_c$ for the values of $q < 0.33$ gives the following expression

$$v \sim (q - q_c)^{\nu}, \tag{8}$$

with $q_c \approx 0.132(5)$ (the number in parentheses is the statistical error of the fitting procedure) and $\nu \approx 1.50(1)$.

The normalized event-rate in the PCS model can be estimated as the number of events in the unity time divided by the number of maximum event rates. In a qualitative analogy with the steady-state velocity in the OLVT model, the normalized event-rate in the PCS model demonstrates the roughening transition. The fit to the data in Figure 1-b for the values of $q' < 0.33$ give the following approximation of the event-rate behavior near the roughening point

$$\text{Event rate} \sim (q' - q_c')^{\nu'}, \tag{9}$$

with $q_c' \approx 0.101(2)$ and $\nu' \approx 1.54(3)$.

The qualitative analogy of the steady-state velocity in the OLVT model and the event rate become even better while plotting depends on them as functions of $p$ and $p'$, correspondingly. In Figure 2 both the left and right curves drops done about five times in the same variation range of the parameters $p$ and $p'$.
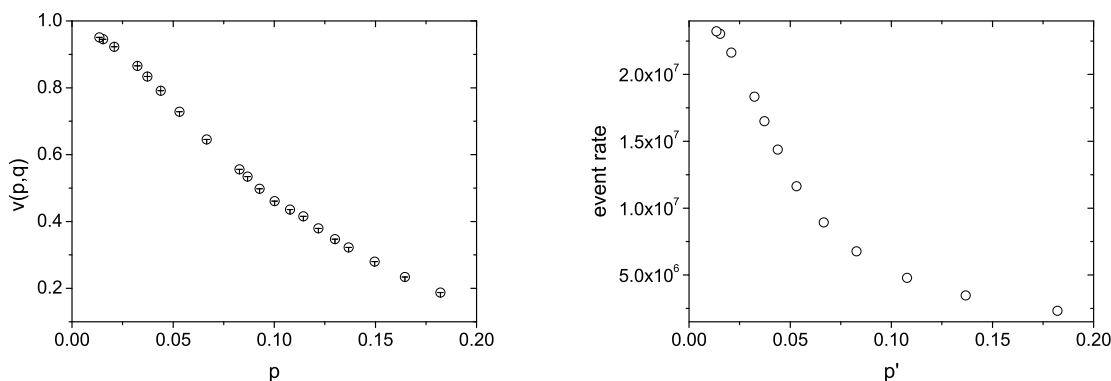


a) Steady-state velocity in OLVT as function of parameter $p$.

b) Event-rate in PCS model as function of parameter $p'$.

**Figure 2.** Behaviour of the a) steady-state velocity $v$ in OLVT model and b) event-rate in PCS model.

*5.2. The width of the profile evolution*

The width of the profile can be associated with the level of processing elements synchronization. Figure 3 shows the dependence of the square width of the OLVT profile and the square width of the PCS profile. The name roughening of the transitions comes from the fact that below $q_c$ $(q'_c)$, both models have the flat profile $\langle w^2(\infty) \rangle$ after saturation, and for larger values of parameter $q_c$ $(q'_c)$ the width is finite, reflecting the rough random profile in both models.
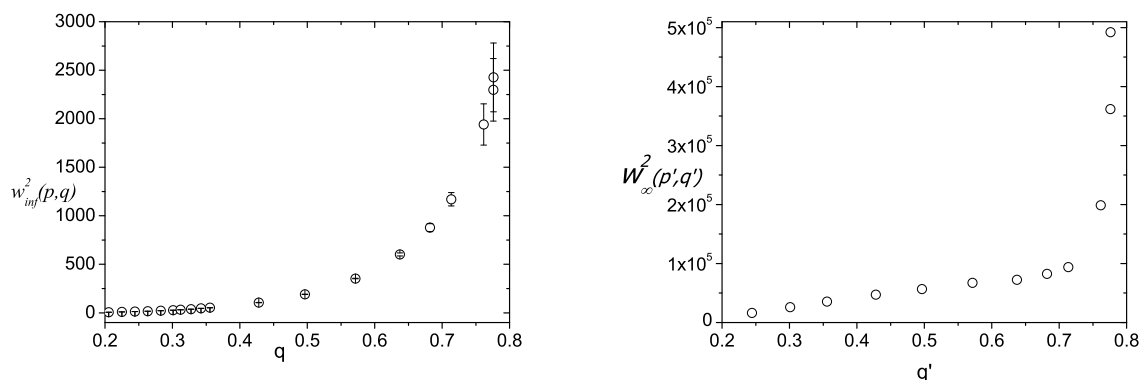


a) The square width of LVT profile in OLVT model as function of parameter $q$.

b) The square width of LVT profile in PCS model as function of parameter $q'$.

**Figure 3.** The mean square width of LVT profiles in OLVT and PCS models.

We should note the similar qualitative behavior in the left and right panel of Figure 3.

## 6. Discussions

We introduce the model of the local virtual time (LVT) profile evolution in the PCS model, the OLVT model. The model reflects the essentials of the LVT evolution. The comparison of the direct simulation of the PCS model with the ROSS simulator and OLVT model demonstrates the qualitative analogy in the LVT profile evolution. Moreover, both models have the roughening transition, and the behavior of models near the transition belongs to the same universality class of the transition.

The paper's main results are 1) introduction of the OLVT model and 2) validation of the model by comparing simulation results for the LVT profile of both models.

## 7. Acknowledgments

## References

[1] Fujimoto R M 1990 *Commun. ACM* **33** 30
[2] Carothers C D, Bauer D and Pearce S 2002 *J. Parallel Distr. Com.* 2002 **62** 1648–69
[3] Weber D Time warp simulation on multi-core processors and clusters 2016 Master's thesis, University of Cincinnati, Cincinnati, OH
[4] Santhi M, Eidenbenz S and Liu J 2015 *Proc. of the 2015 Winter Simulation Conf. (Huntington Beach)* ed. Yilmaz L et al. (New York City: IEEE press) pp 3013–24
[5] Steinman J 1991 *Proc. SCS Western Multi-conf. on Advances in Parallel and Discrete Simulation (Anaheim)* vol 23, ed. Madisetti V et al. p 1111
[6] Cardoen B, Manhaeve S, Van Tendeloo Y and Broeckhove J 2018 *Simulation* **94(4)** 281–300
[7] Henderson T R, Lacage M, Riley G F, Dowell C and Kopena J 2008 *SIGCOMM demonstration* **14(14)** 527
[8] Riley G F and Henderson T R 2010 *Modeling and tools for network simulation* 15–34.
[9] Barnes Jr P D, Carothers C D, Jeferson D R and LaPre J M 2013 *Proc. of the 1st ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation (Montreal)* (Association for Computing Machinery: New York) pp 327–36
[10] Fujimoto R M, Carothers C D, Ferscha A, Jefferson D, Loper M, Marathe M and Taylor S J 2017 *Proc. of the 2017 Simulation Conf. (Las Vegas)* (New York: IEEE) p 431
[11] Balci O, Fujimoto R M, Goldsman D, Nance R E, and Zeigler B P 2017 *Proc. of the 2017 Simulation Conf. (Las Vegas)* (New York: IEEE) pp 821–836
[12] Korniss G, Toroczkai Z, Novotny M A and Rikvold P A 2000 *Phys. rev. lett.* **84(6)** 1351
[13] Ziganurova L, Novotny M A and Shchur L 2016 *J. Phys. Conf. Ser.* **681** 012047
[14] Korniss G, Novotny M A, Guclu H, Toroczkai Z and Rikvold P A 2003 *Science* **299(5607)** 677-79
[15] Ziganurova L and Shchur L N 2018 *Phys. Rev.* E **98(2)** 022218
[16] Shchur L N and Ziganurova L F 2019 *J. Exp. Theor. Phys.* **129(4)** 722–32
[17] Carothers C D, Fujimoto R M, Lin Y B and England P 1994 *Proc. of International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (Durham)* (New York City: IEEE press) pp 2-6
[18] Carothers C D, Fujimoto R M and Lin Y-B 1995 *ACM SIGSIM Simulation Digest* **25(1)** 87–94
[19] Lin Y-B and Fishwick P A A 1996 *IEEE Trans. Syst. Man Cybern. Syst.* **26(4)** 397–412
[20] Jefferson D R 1985 *ACM Transactions on Programming Languages and Systems (TOPLAS)* **7** 404
[21] Wilsey P A 2016 *Proc. of the 2016 ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation (Banff Alberta)* (New York: ACM) pp 165–76
[22] Bauer Jr D W, Carothers C D and Holder 2009 *Proc. of the 2009 ACM/IEEE/SCS 23rd Workshop on Principles of Advanced and Distributed Simulation (Lake Placid, New York)* (Washington: IEEE Computer Society) pp 35–44
[23] Williams B, Ponomarev D, Abu-Ghazaleh N and Wilsey P 2017 *Proc. of the 2017 ACM SIGSIM Conf. on Principles of Advanced Discrete Simulation (Singapore)* (New York: ACM) pp 121–32
[24] Jagtap D, Abu-Ghazaleh N and Ponomarev D. 2012 *IEEE 26th Int. Parallel and Distributed Processing Symposium (Shanghai)* (New York: IEEE Press) pp 520–53

[25] Eker A, Williams B, Mishra N, Thakur D, Chiu K, Ponomarev D and Abu-Ghazaleh N 2018 *Proc. of the 22nd International Symposium on Distributed Simulation and Real Time Applications (Madrid)* (New York: IEEE Press) pp 87–96

[26] Shchur L N and Novotny M A 2004 *Phys. Rev, E* **70(2)** 026703

[27] Grassberger P 1982 *Zeitschrift für Physik B Condensed Matter* **47** 365

[28] Odor G 2004 *Rev. Mod. Phys.* **76** 663

[29] Hinrichsen H 2000 *Brazilian J. Phys.* **30** 69

Part of the results was published in the Conference Proceedings (L. Ziganurova and L. Shchur, Local time evolution in Personal Communication Service model, Journal of Physics: Conference Series, v. 1740 (2021) 012005, doi:10.1088/1742-6596/1740/1/012005).

Difference between the current paper with the published proceedings

1. The paper addresses the researchers working in the field of Parallel and Distributed computing.
2. Extended Abstract.
3. Extended Introduction.
4. New section – III. Related Work.
5. The extended section IV. Validation of OLVT model with a new presentation in subsections C and D.
6. New discussion section.