8TH INTERNATIONAL CONFERENCE "DISTRIBUTED COMPUTING AND GRID-TECHNOLOGIES IN SCIENCE AND EDUCATION", GRID2018, 10-14 SEPT, DUBNA

Properties of The Parallel Discrete Event Simulation Algorithms on Small-World Communication Networks

Ziganurova Liliia

National Research University Higher School of Economics, Moscow
Science Centre in Chernogolovka, Chernogolovka, Moscow area

IN COLLABORATION WITH:

L.N. Shchur

1. Scientific Center in Chernogolovka, Moscow area, Russia
2. National Research University Higher School of Economics, Moscow, Russia
3. Landau Institute for Theoretical Physics, Moscow, Russia

M. Novotny

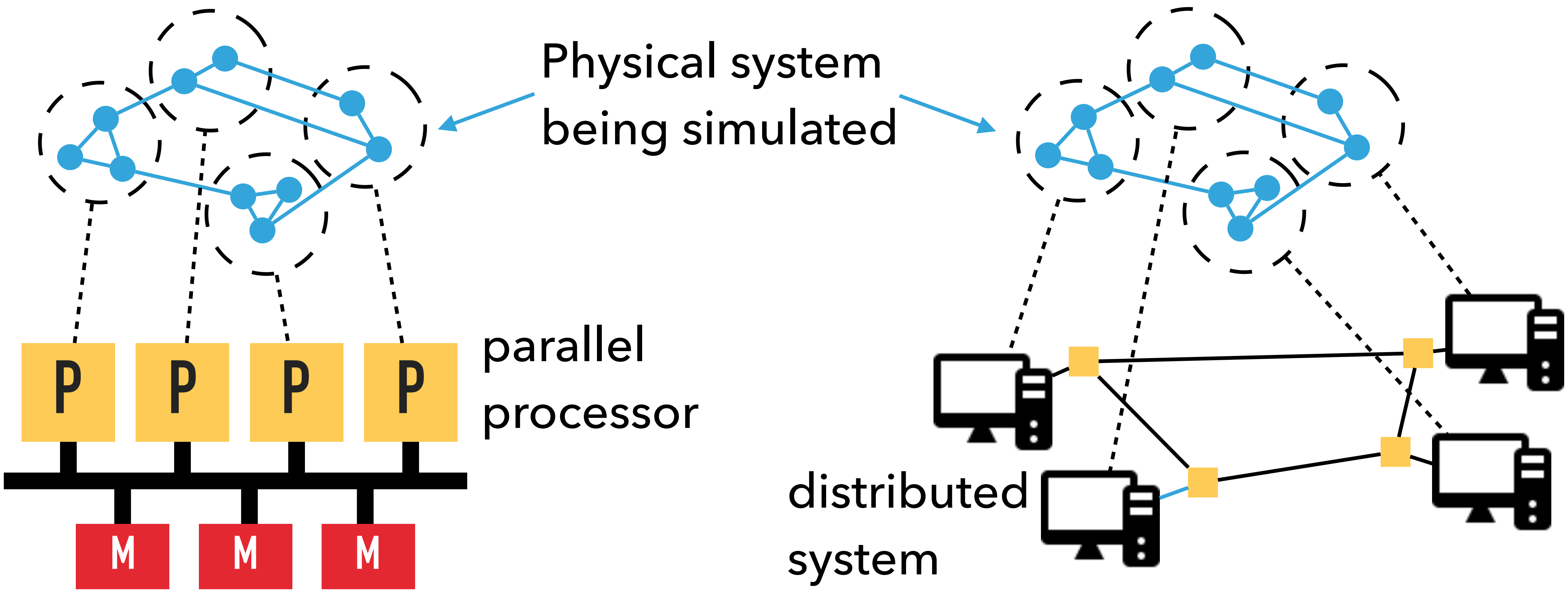
1. Mississippi State University, USA

MOTIVATION

- Modern computer systems: **10^4 of nodes**
- Each node may have many CPUs, cores, and numerical accelerator
- All CPU's (cores, threads, …) must be **synchronised** to efficiently execute one parallel program
- High performance computing requires new approaches to programming models

Parallel Discrete Event Simulation is a method of large-scale simulation which allows to execute a single program on a parallel computer.

PARALLEL AND DISTRIBUTED SIMULATION*



Parallel simulation involves the execution of a *single* simulation on a collection of **tightly** coupled processors (e.g. a shared memory multiprocessor)

Distributed simulation involves the execution of a *single* simulation on a collection of **loosely** coupled processors (e.g. PCs interconnected by a LAN or WAN)

**from R.Fujimoto*

ESSENTIAL PROPERTIES OF PDES:

- Changes in subsystems occur at some instant of time and are called **discrete events.**
- To preserve causality between dependent objects some **synchronisation protocol** is used.
- Using the **virtual time concept**.
- Communication between parallel processes goes via timestamped messages.
- No shared memory between subsystems.
- Developers may use special PDES frameworks (i.e. ROSS or TIMEWARP2 simulators)

VIRTUAL TIME CONCEPT (AN EXAMPLE)

PE = node/CPU/core/etc.	PE1	PE2	PE3	PE4
Lists of events with timestamps	1	3	5	2
	4	8	11	7
	9	14		13
	10			
Local virtual time of PEs	1	3	5	2

SYNCHRONISATION ALGORITHMS

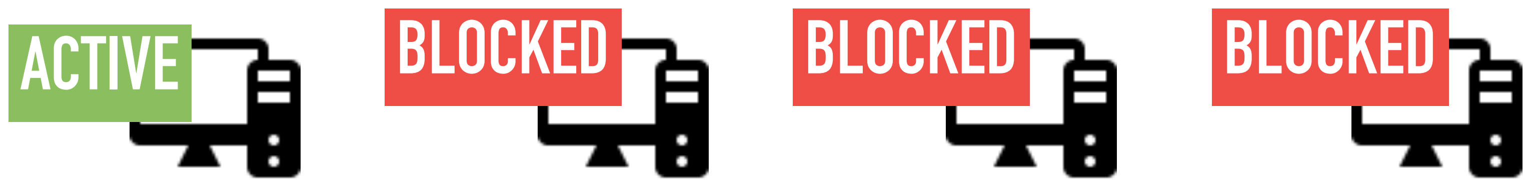
- **Conservative -** avoids all possible causality violations by checking the causality relations between dependent events at each discrete step of simulation
- **Optimistic -** allows some causality errors, has a roll-back mechanism
- **Freeze-and-Shift** - a combination of conservative and optimistic approaches

Conservative algorithm

Only those PEs, whose current time is lower than the time of their neighbours (i.e. the PEs which it is connected with), may proceed with computations. These PEs are called active. Such scheme guarantees that causality will be preserved.

CONSERVATIVE SYNCHRONISATION

PE = node/CPU/core/etc.



	PE1	PE2	PE3	PE4
Lists of events with timestamps	1	3	5	2
	4	8	11	7
	9	14		13
	10			
Local virtual time of PEs	1	3	5	2

CONSERVATIVE SYNCHRONISATION

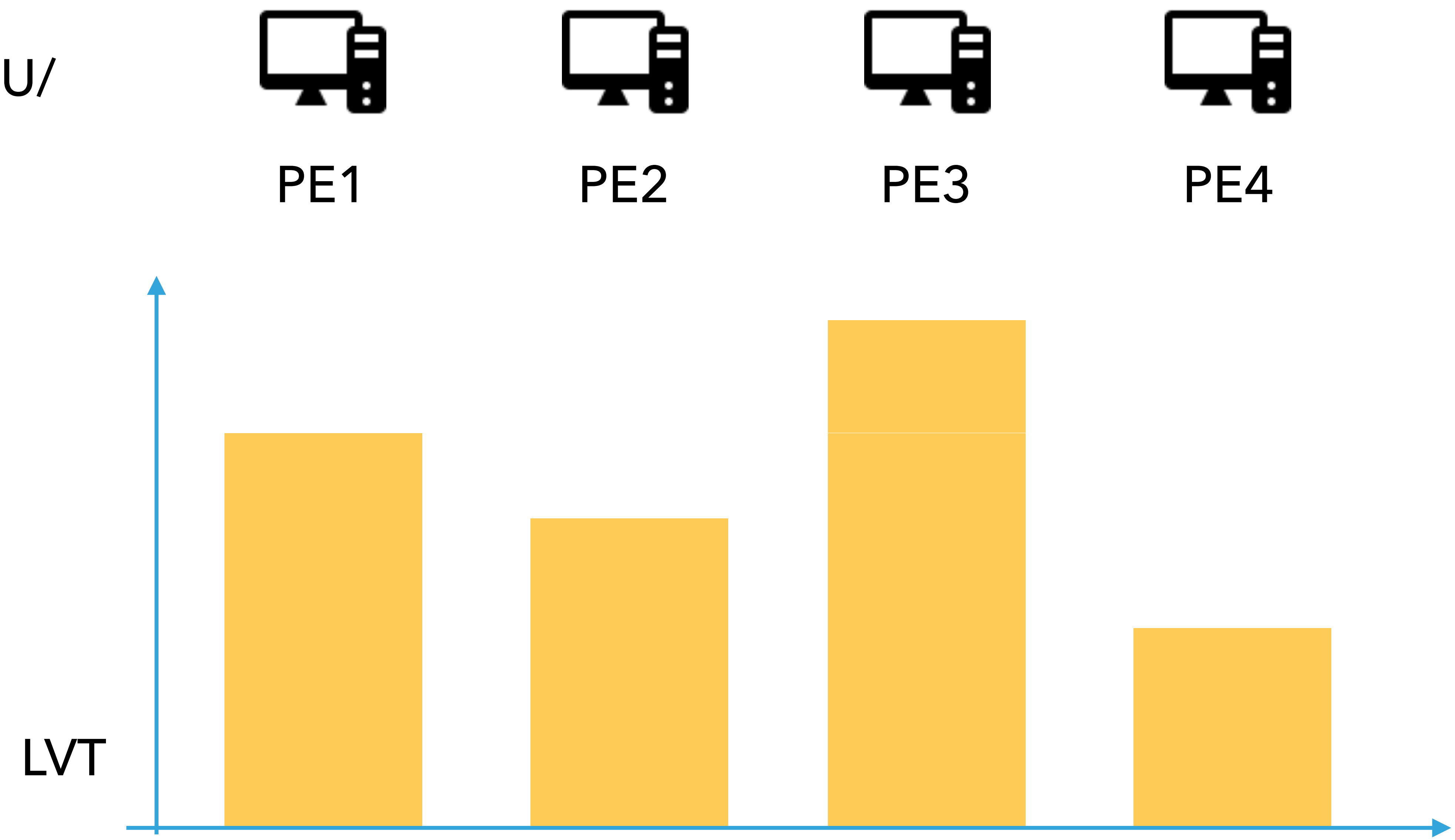
PE = node/CPU/core/etc.

	PE1	PE2	PE3	PE4
	BLOCKED	BLOCKED	BLOCKED	ACTIVE
Lists of events with timestamps	4	3	5	2
	9	8	11	7
	10	14		13
Local virtual time of PEs	4	3	5	2

THE CONCEPT OF VIRTUAL TIMES (EXAMPLE)

PE = node/CPU/core/etc.

Local virtual time profile

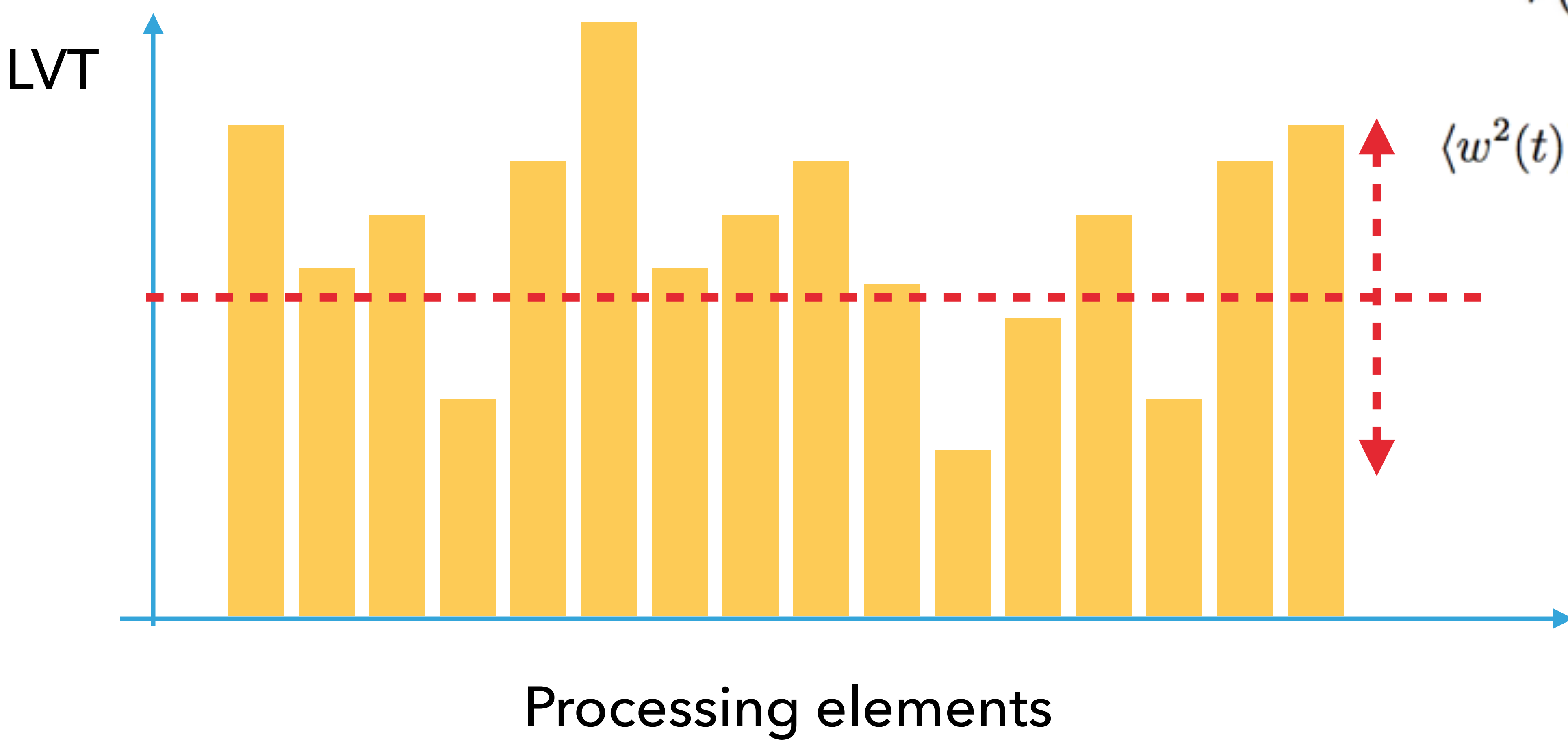


Local virtual time of PEs

PE1	PE2	PE3	PE4
4	3	5	2

THE OBJECT OF THE RESEARCH

We study the scalability properties of synchronisation algorithms on small-world communicational network.



$$\overline{\tau}(t) = 1/N \sum_{i=1}^{N} \tau_i$$

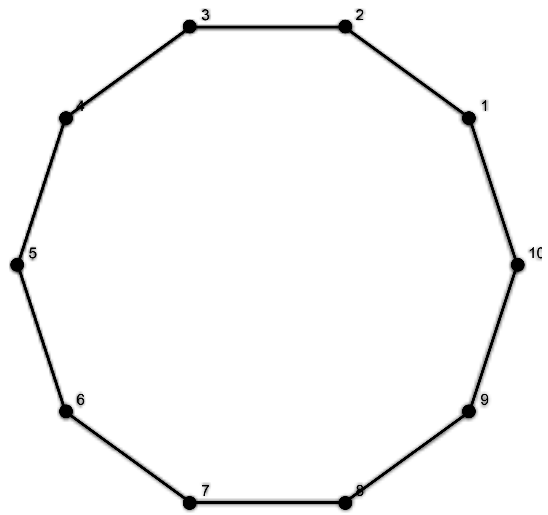
$$\langle w^2(t) \rangle = \langle \frac{1}{N} \sum_{i=1}^{N} [\tau_i(t) - \overline{\tau}(t)]^2 \rangle$$

desynchronisation

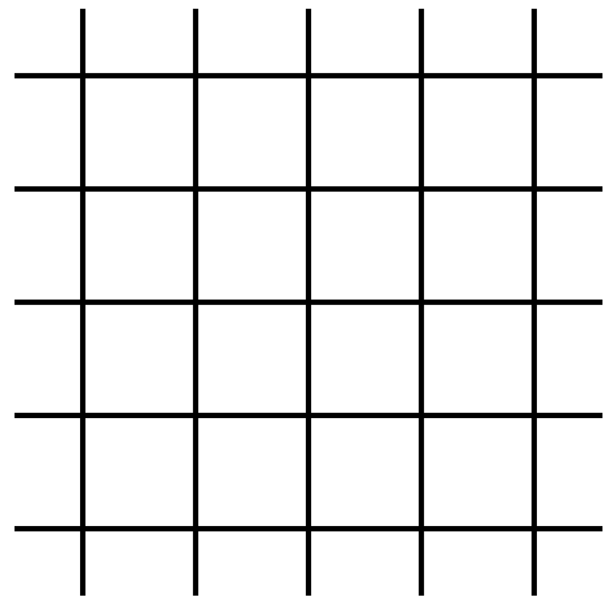
$$\langle u(t) \rangle = \frac{N_{active}}{N}$$

efficiency

HOW LONG-RANGE LINKS AFFECTS SYNCHRONISATION?



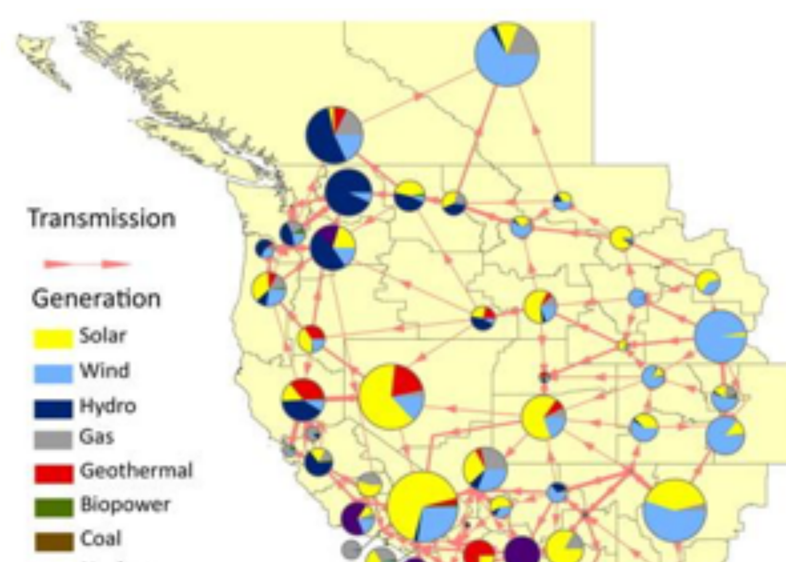
Regular ring lattice



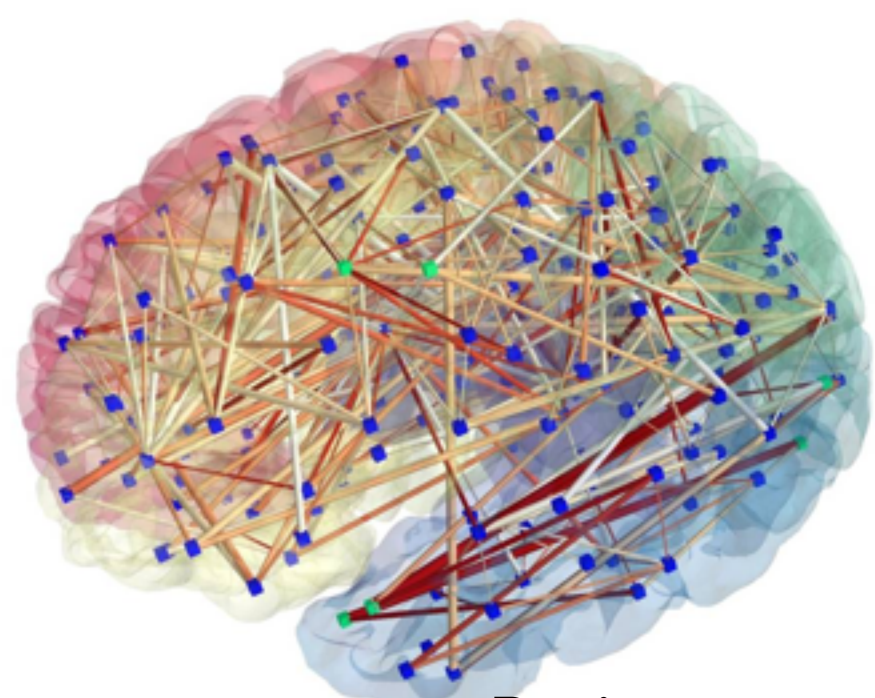
2d regular lattice

Social Networks

Vertices - PEs
Edges - communications (dependencies)

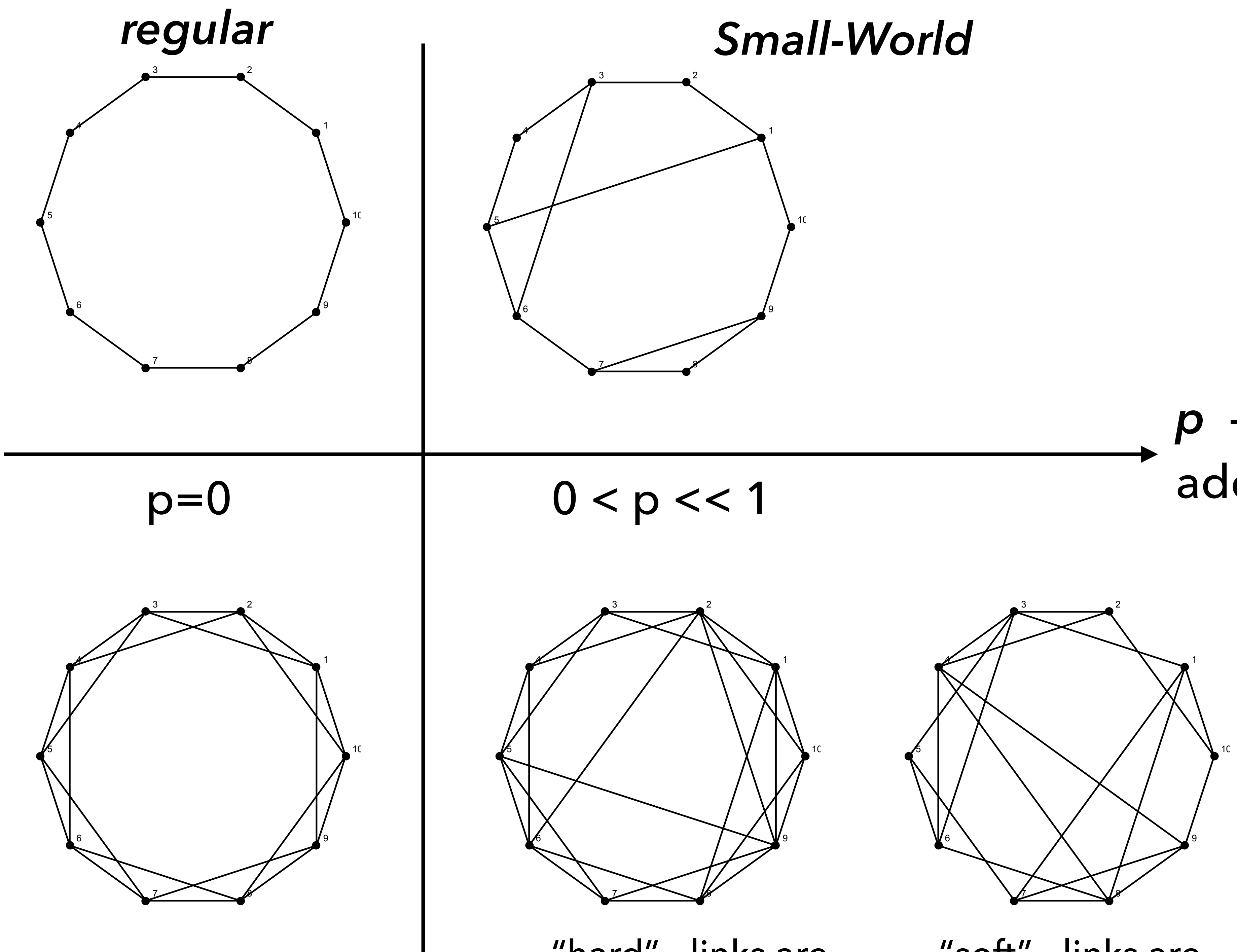


Electric Grid



Brain neurons

THREE TYPES OF NETWORKS



p - concentration of added links

1. Average shortest path ~ log(N)
2. Clustering coefficient ≈ 0

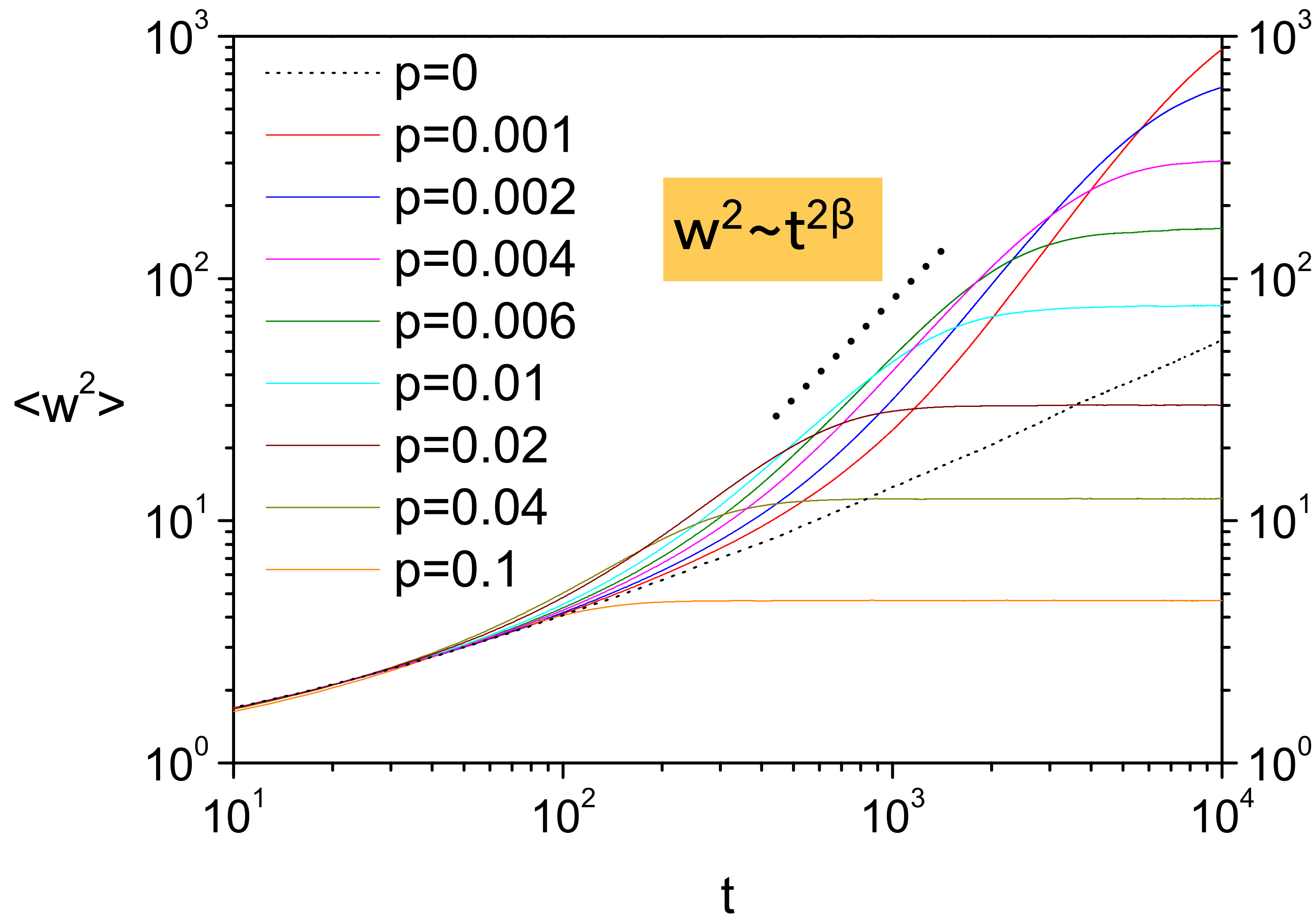
"hard" - links are added

"soft" - links are rewritten

1. Average shortest path ~ log(N)
2. High clustering coefficient

RESULT #1 GROWTH EXPONENT

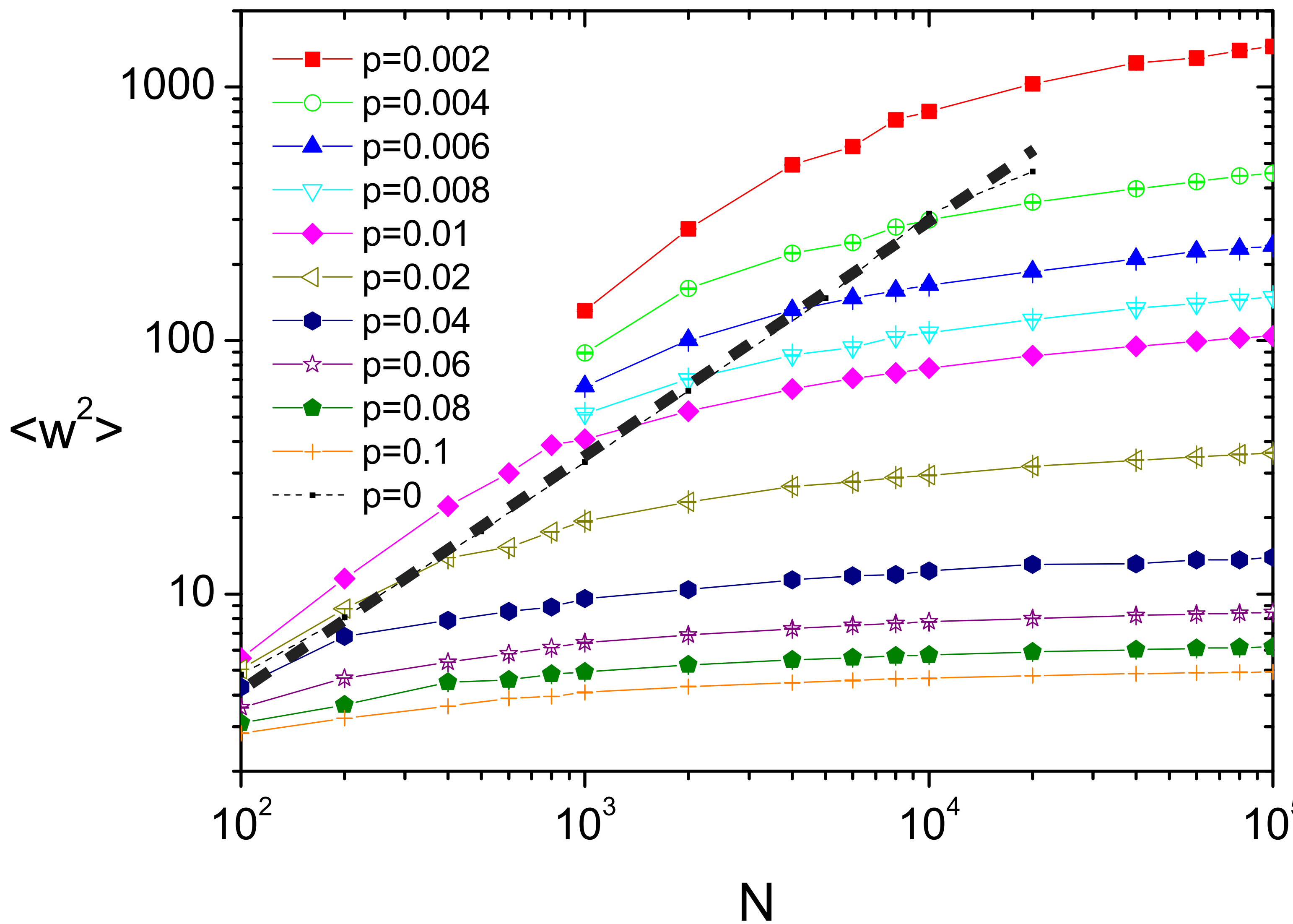
Growth exponent **β** logarithmically depends on the parameter ***p***



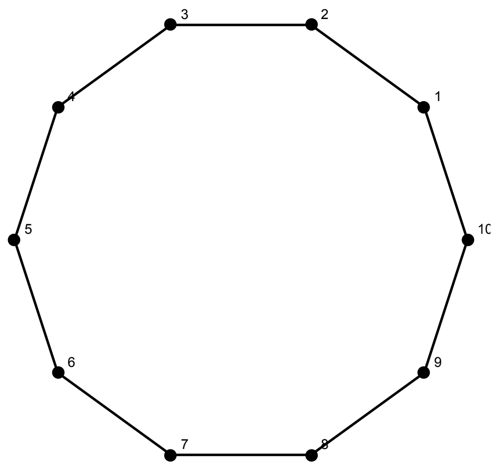
$$\beta \sim -0.162(2)\ln(p)$$

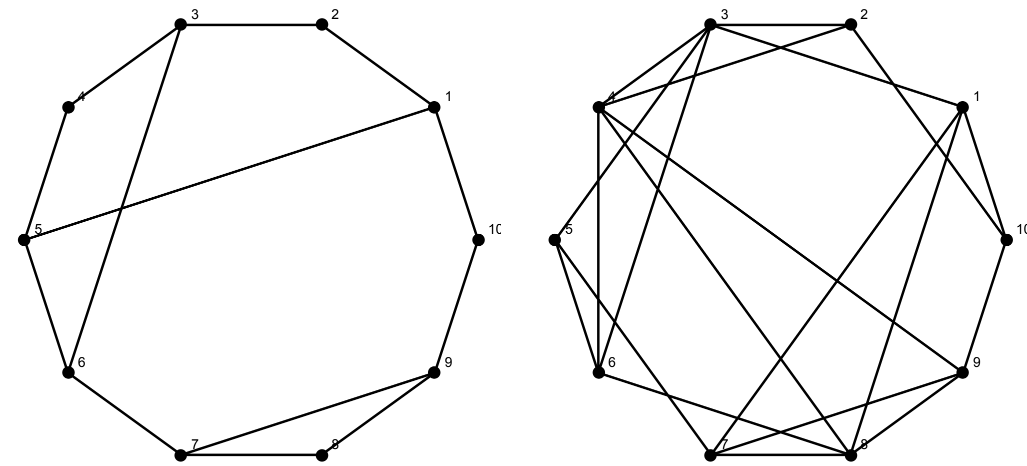
RESULT #2 ROUGHNESS EXPONENT

The width (i.e. desynchronisation) remains constant as the number of processes goes to infinity.



COMPARISON WITH REGULAR NETWORK





$\langle U_0 \rangle = 0.24647(1)$

$\langle w^2(t) \rangle \sim t^{2b}$, $b = 1/3$

$\langle w^2(N) \rangle \sim N^{2a}$, $a = 1/2$

$\langle U \rangle \sim \langle U_0 \rangle - p^B$, $B < 1$

$\langle w^2(t) \rangle \sim t^b$, $b \sim -\ln(p)$

$\langle w^2(N) \rangle \sim \text{const}$

1) the average progress rate remains positive - **no deadlocks**,

2) the desynchronisation degree of the LVT profile becomes **finite**, when the number of PEs goes to infinity.

Optimistic algorithm

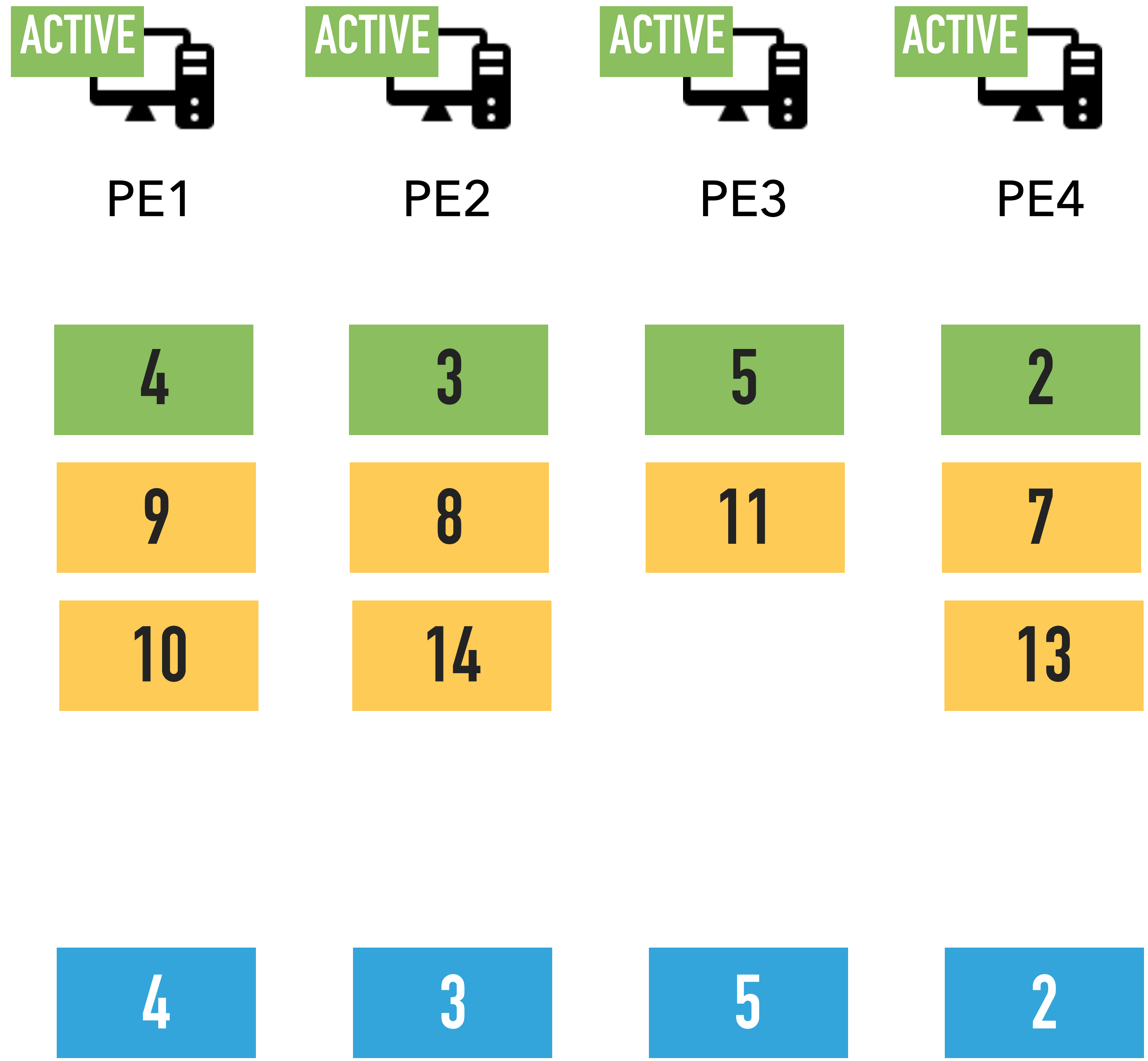
Allows emergence of causality errors but has a roll-back mechanism. The process, received a message with timestamp lower than its LVT, rolls-back to the state with lower time. It also sends anti messages to other processes to cancel previously sent messages.

OPTIMISTIC SYNCHRONISATION

PE = node/CPU/core/etc.

Lists of events with timestamps

Local virtual time of PEs



OPTIMISTIC SYNCHRONISATION

PE = node/CPU/core/etc.	PE1	PE2	PE3	PE4
	ACTIVE	ACTIVE	ACTIVE	ACTIVE
Lists of events with timestamps	9	8	11	7
	10	14		13
Local virtual time of PEs	9	8	11	7

OPTIMISTIC SYNCHRONISATION

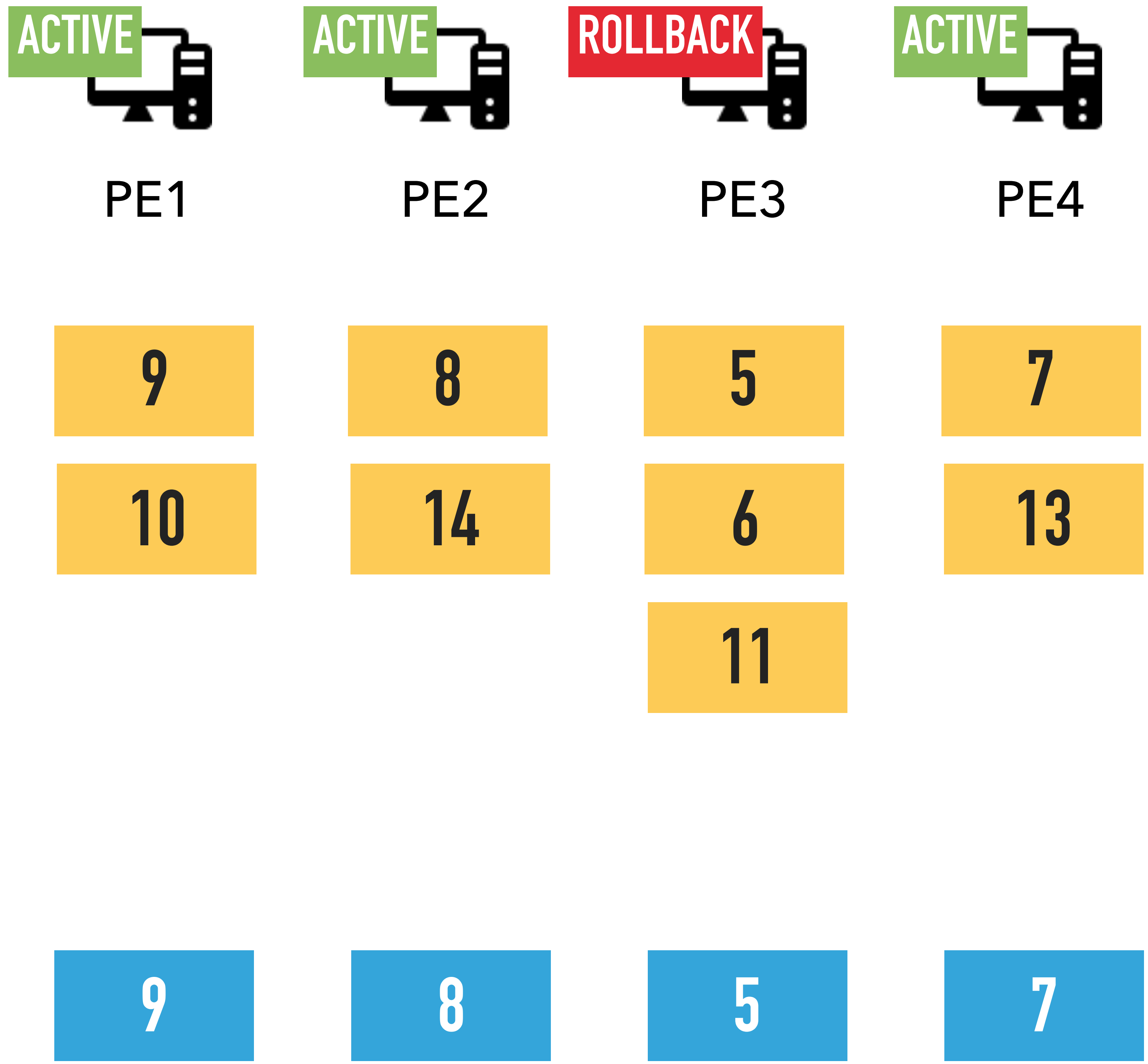
PE = node/CPU/core/etc.	ACTIVE PE1	ACTIVE PE2	ACTIVE PE3	ACTIVE PE4
Lists of events with timestamps	9	8	11	7
	10	14	6	13
			CAUSALITY ERROR!	
Local virtual time of PEs	9	8	11	7

OPTIMISTIC SYNCHRONISATION

PE = node/CPU/core/etc.

Lists of events with timestamps

Local virtual time of PEs



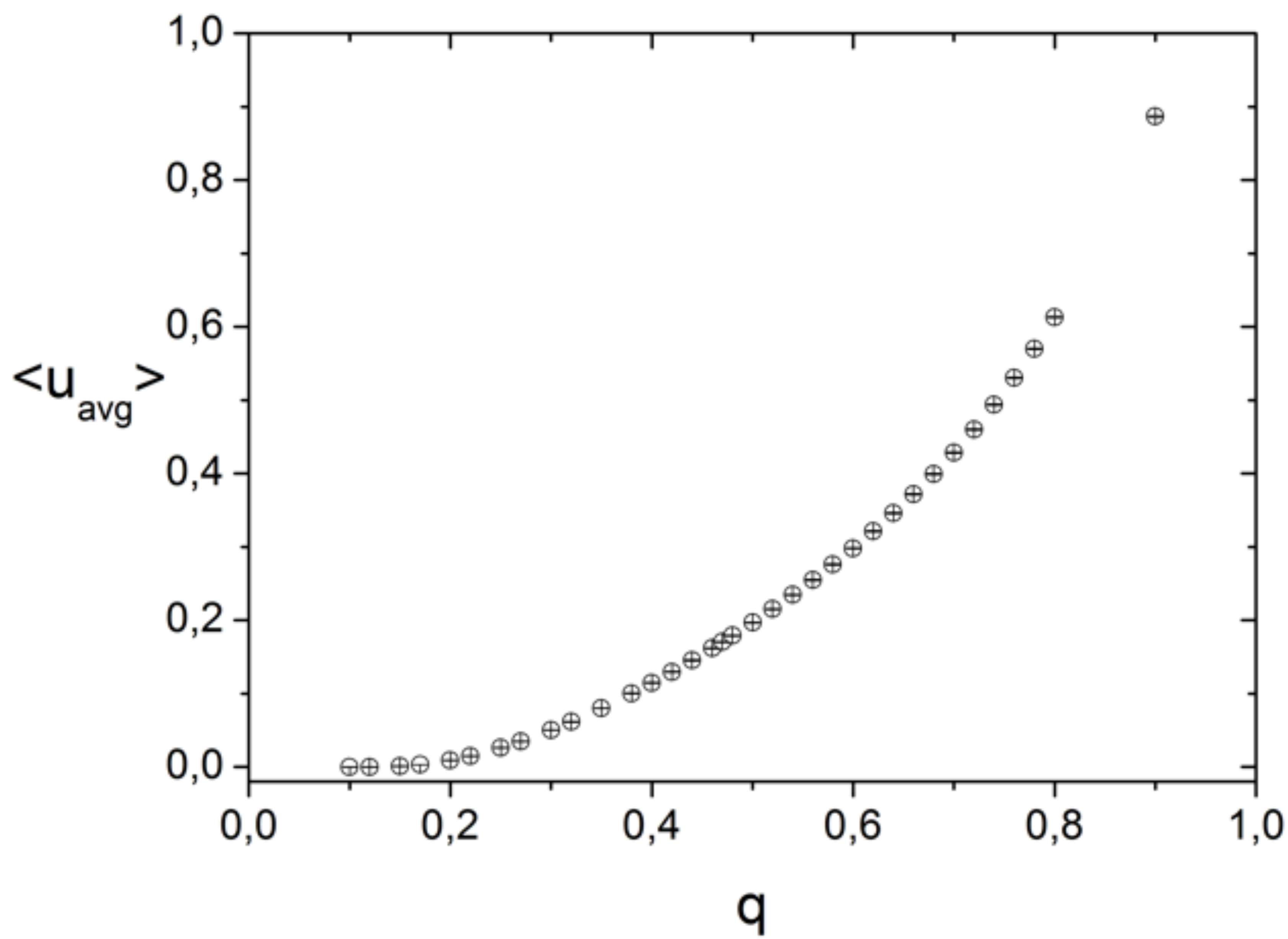
RESULT: AVERAGE SPEED

Let's introduce the parameter (progress rate):

$$q = \frac{1}{1+b}$$

where b is a mean avalanche length (the number of PEs, which rolled back during one simulation step)

$$u = u_0(q - q_c)^\nu$$



RESULT: AVERAGE SPEED

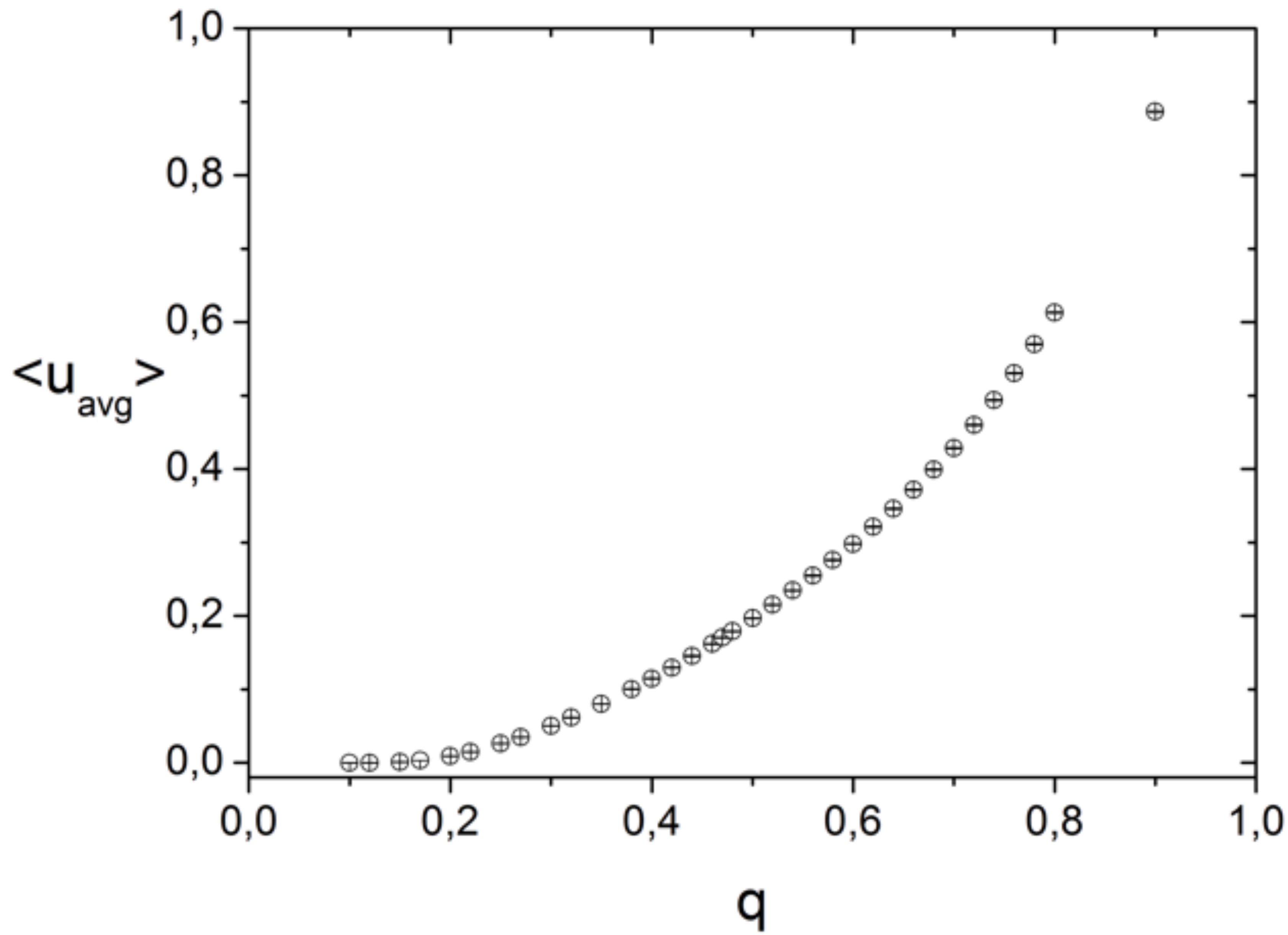
- **Critical exponent**

$$\nu \sim p$$

- **Critical point**

$$q_c = 0.233(1)$$

$$u = u_0(q - q_c)^{\nu}$$



FUTURE WORK

1. Run test models (transport, epidemiological models) on ROSS simulator (http://carothersc.github.io/ROSS/)

CONCLUSION

- Synchronisation in the PDES algorithm better in systems with some amount of long-range interactions (on Small-world networks)
- Synchronisation in the PDES algorithm depends only on average shortest path and does not depend on clustering degree of a network

- **PDES is a promising method of large-scale simulations, because it is well scalable and relatively easy to implement.**