Use of a Desktop Grid to Effectively Discover Hits in Virtual Drug Screening

Natalia Nikitina, Evgeny Ivashko

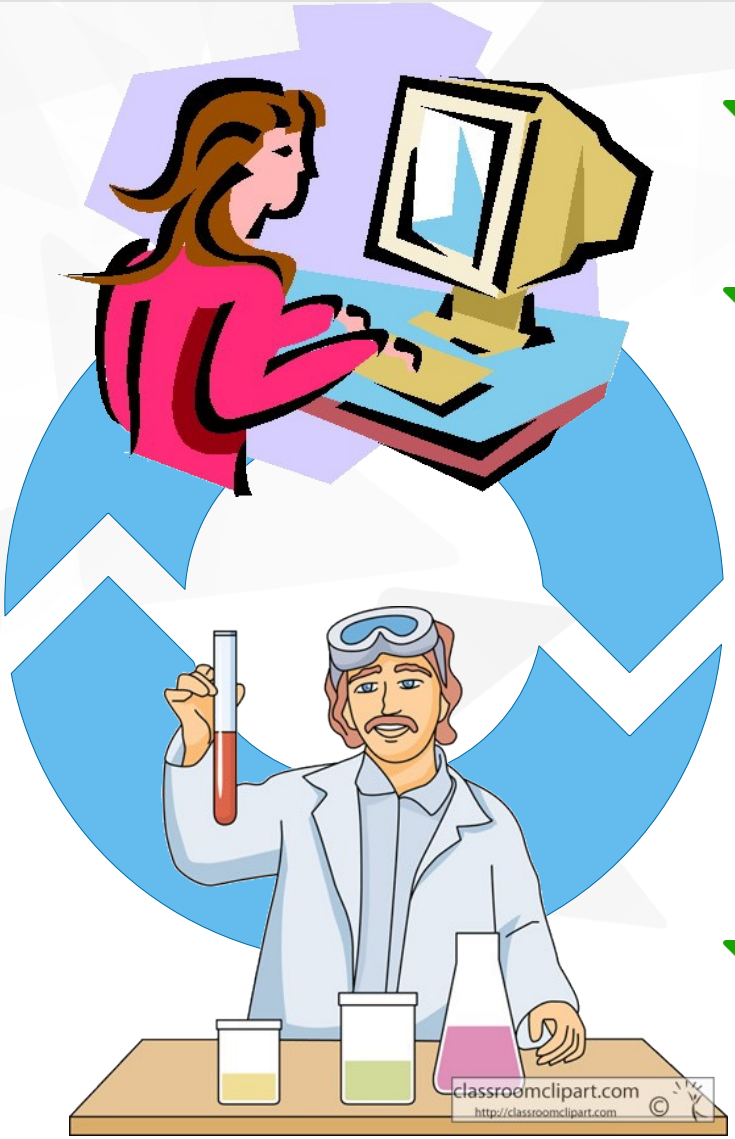
Institute of Applied Mathematical Research

Karelian Research Center of the RAS

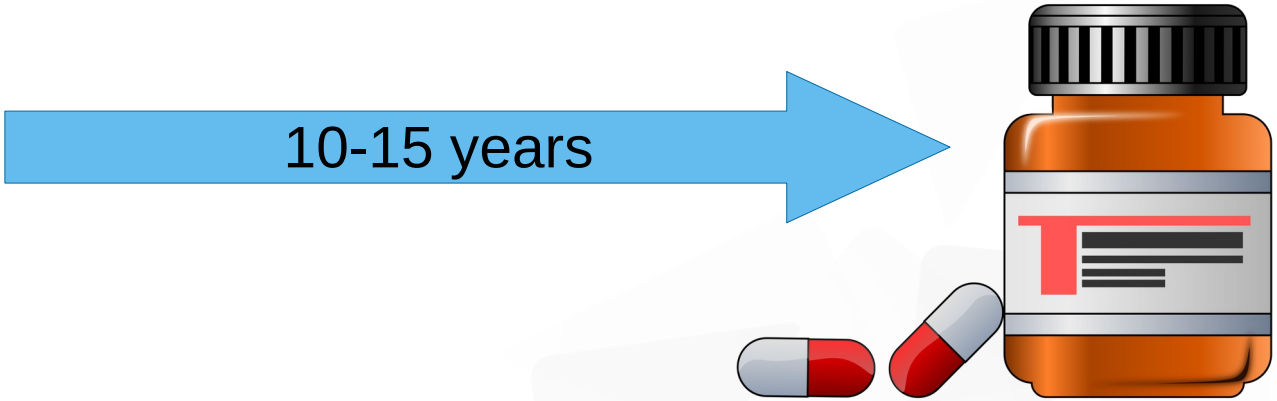
Moscow, Russia, 23-24 September 2019

1. Drug development and virtual screening
2. The problem of virtual screening
3. Mathematical model and solution
4. Desktop Grid and BOINC
5. Implementation and experiments

Drug development



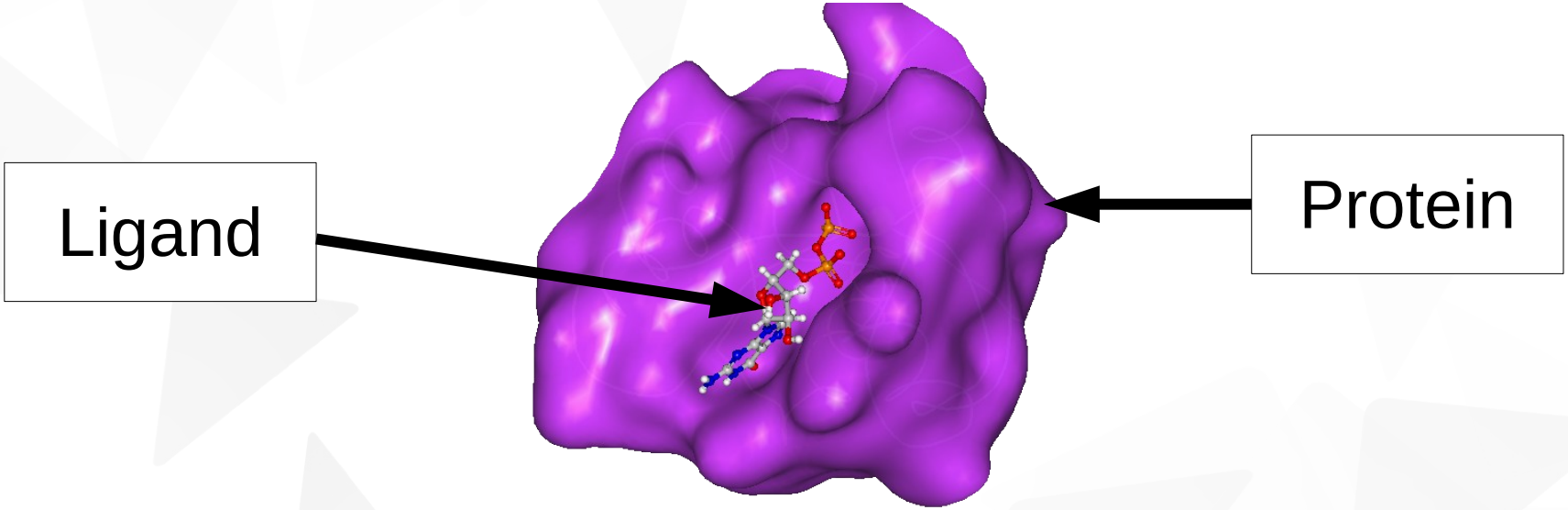
- Drug development is a time-consuming and resource-consuming process
- It takes up to 10-15 years to develop a new drug and bring it to the market



- At early stages, **high-performance computing and high-throughput computing** assist drug development

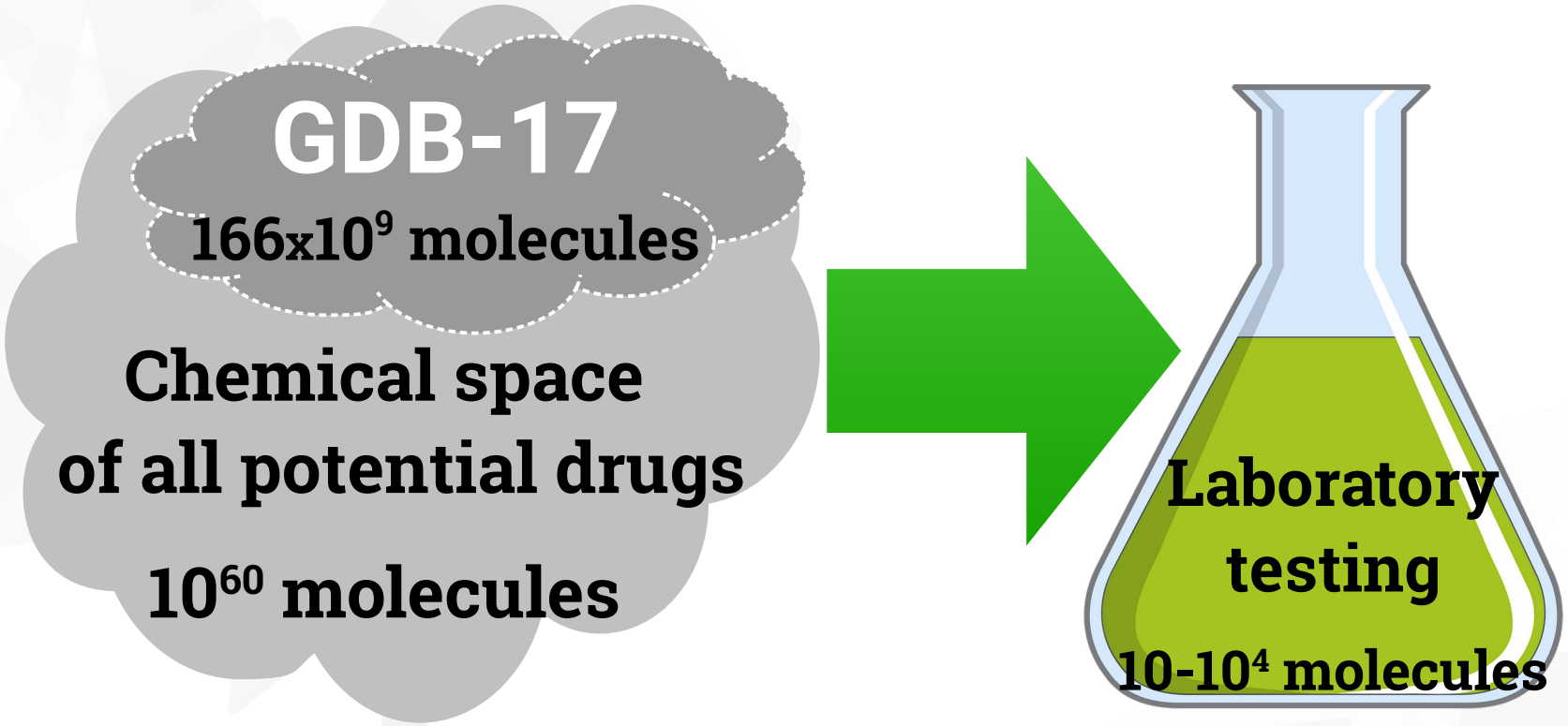
Drug development

- The aim of drug development is discovery of a small molecule (ligand) which binds to a target protein related to the disease development and has desired biochemical activity



Ligands able to bind to the target → **Hits** predicted to have desired biochemical activity → **Leads** evidently having desired biochemical activity

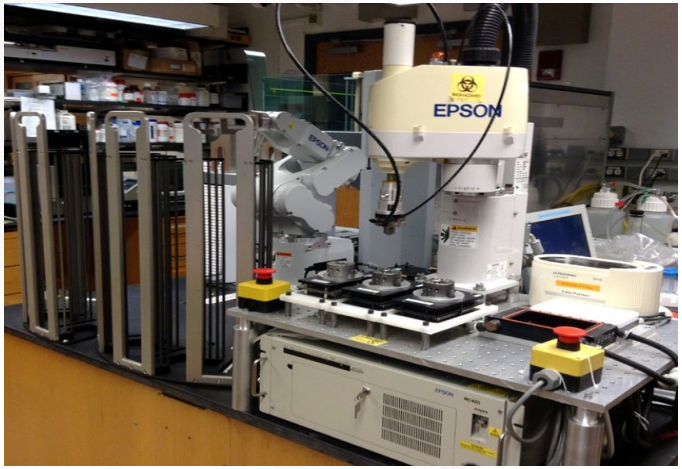
Chemical space



To reduce input dataset for drug search down to manageable size, the huge chemical space is pre-filtered, leaving only representative compounds that promise to show desired biological activities.

HTS and virtual screening

- High-throughput screening is a robotized search for hits in libraries of real chemical compounds



✗ Expensive
✗ Not always feasible
✗ Hardly available to academic research
✗ Requires good preparation

- Virtual screening uses computational tools to search for hits in libraries of chemical compounds models

✔ Cheap
✔ Always feasible
✔ Available to academic research
✔ Prepares input set for HTS
✔ Perfectly suits distributed computing

Tools for virtual screening

- Libraries of molecule models
  - ZINC, ChemBridge, etc. – **commercially available**
  - GDB-13, GDB-17, GDB-21 – **potentially synthesizable**
- Software for molecular docking (>60 software products)
- Pipelines for virtual screening
  - boinc-server-autodock debian (Steffen Möller, Natalia Nikitina)
- Volunteer computing projects on virtual screening
  - Docking@Home
  - FightMalaria@Home
  - World Community Grid → FightAIDS@Home, OpenZika, Smash Childhood Cancer, Outsmart Ebola Together etc.
  - … and many others

The problem of virtual screening

- Huge size of libraries and computational cost of VS
- Pre-filtered libraries may be ineffective when studying new or rare diseases, as potentially good classes of molecules were filtered out. So the chemical diversity of results is limited

(Using high-throughput computing) how to perform virtual screening and

- ... provide high diversity of results in limited time
- ... provide first successful results ASAP

The problem of virtual screening

- Pre-filtering of the chemical space (clustering, Monte Carlo method, simulated annealing...)
  - Requires much knowledge about disease target and known ligands
  - Omits potentially interesting compounds for rare or novel diseases
  - Requires complex post-filtering of virtual screening results

- Genetic algorithms with stochastic search *(C. Rupakheti et al. "Strategy To Discover Diverse Optimal Molecules in the Small Molecule Universe". Journal of Chemical Information and Modeling, 2015, 55(3), pp. 529–537)*
  - Have good performance
  - Require redundant computations
  - In general case, do not guarantee results in appropriate time

Solution

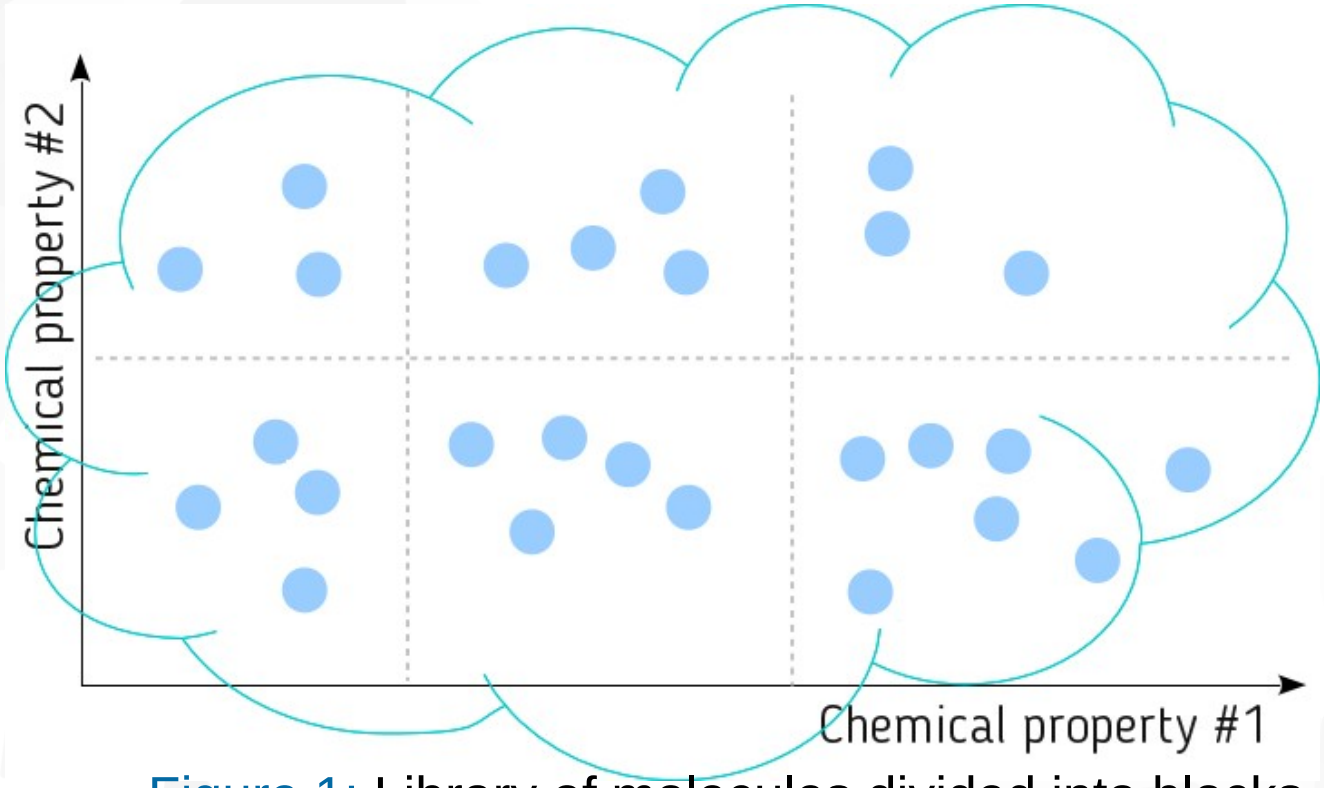


Figure 1: Library of molecules divided into blocks

Blocks priorities to search in:

- molecules of very simple/complex shape are less likely to become drugs
- molecules highly similar to a known ligand are more likely to become drugs
- etc.

Solution

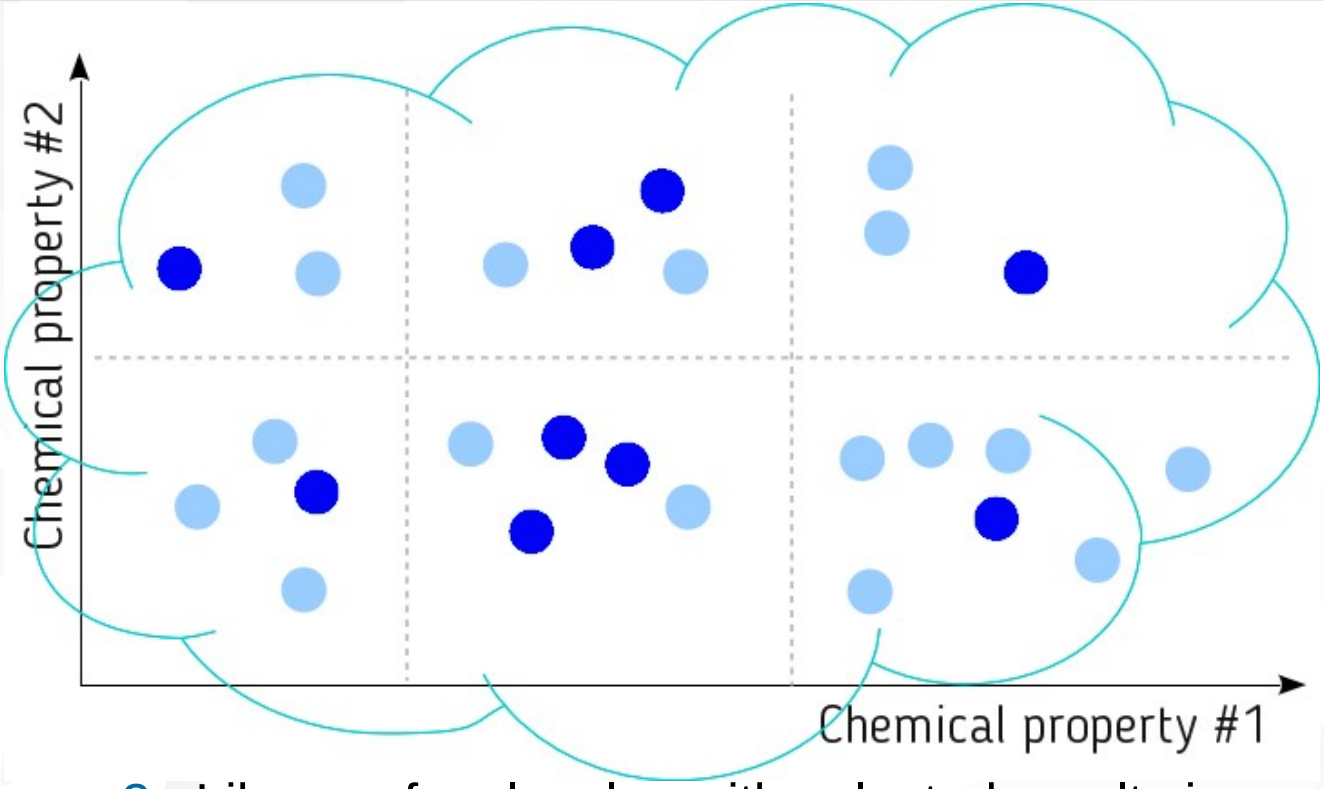


Figure 2: Library of molecules with selected results in each block

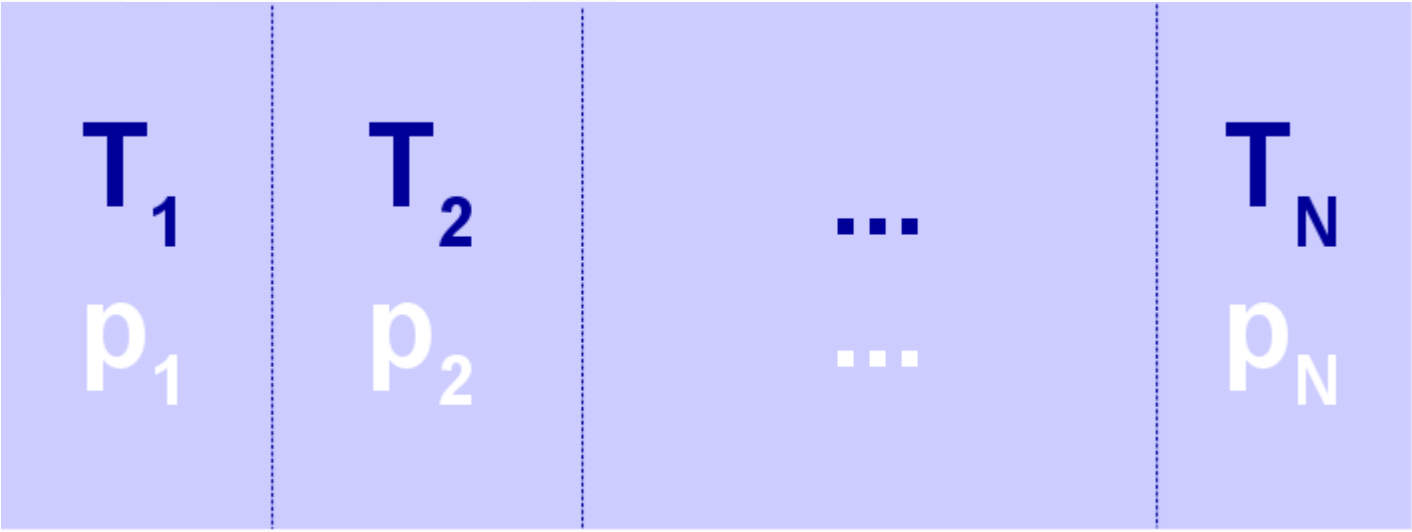
Task scheduling:

- explore different blocks → obtain chemically diverse results in limited time
- explore prospective blocks first → successful results in short time

Mathematical model

$C_1, \ldots, C_M$ are the computational nodes (players), $M \geq 2$

T is the set of computational tasks



p_r is the expected fraction of useful results in block T_r

$\sigma_r = \dfrac{p_r}{p_1 + \ldots + p_N}$ is the priority of block T_r

Mathematical model

ops_i is the performance of the computational node (number of operations per second)

θ_j is the complexity of a computational task (number of operations)

τ is the considered time interval

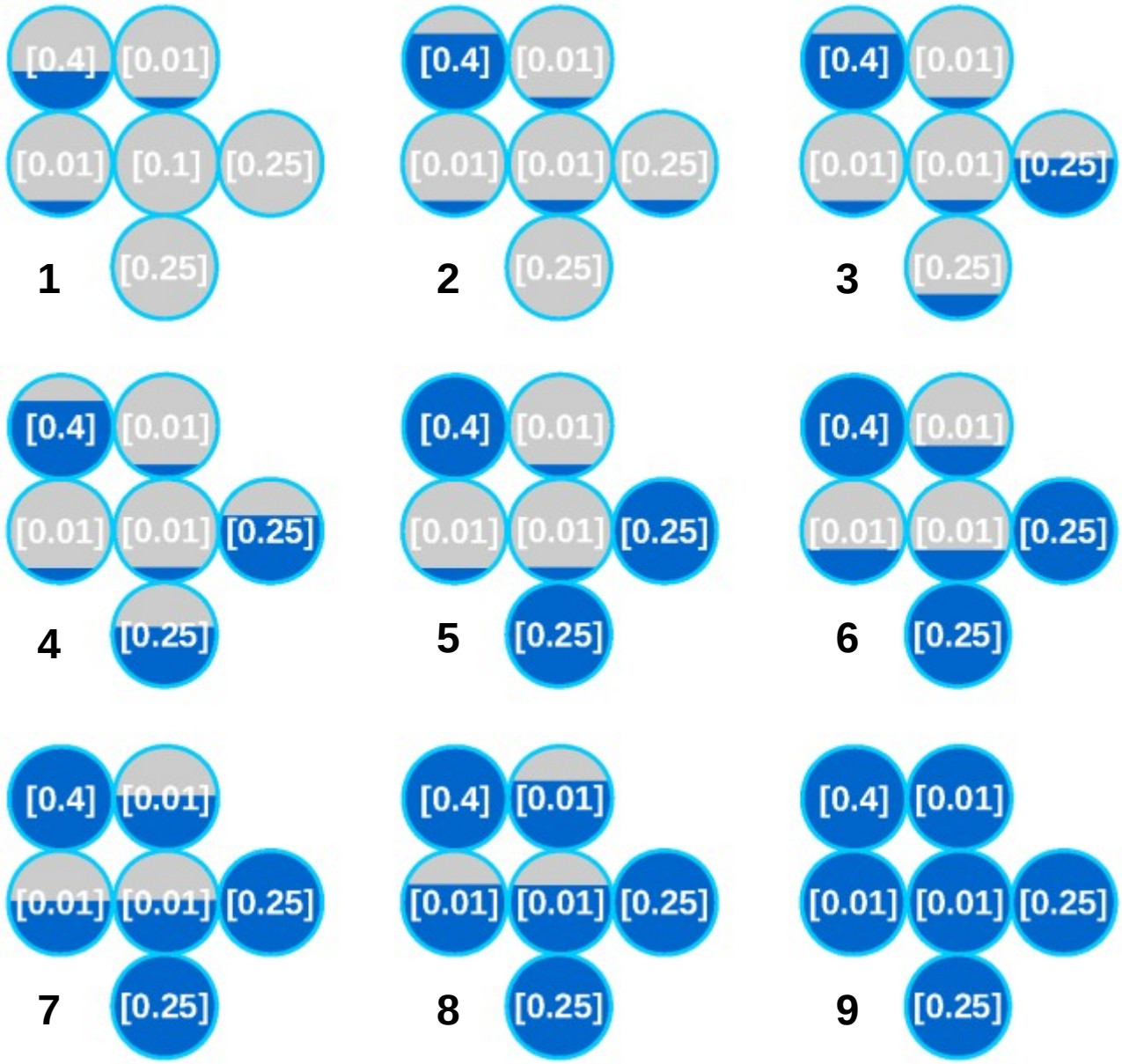
n_j is the number of players that have chosen block T_j

$\delta(n_j) = \frac{M+1-n_j}{M}$ is the congestion coefficient of block T_j

$U_{ij} = \sigma_j \delta(n_j) \frac{ops_i}{\theta_j} \tau$ is the usefulness of node C_i which chooses block T_j

$\vec{s} = (s_1, \ldots, s_M)$ is the strategy profile (blocks selected by each player)

Example: Filling the blocks of molecules

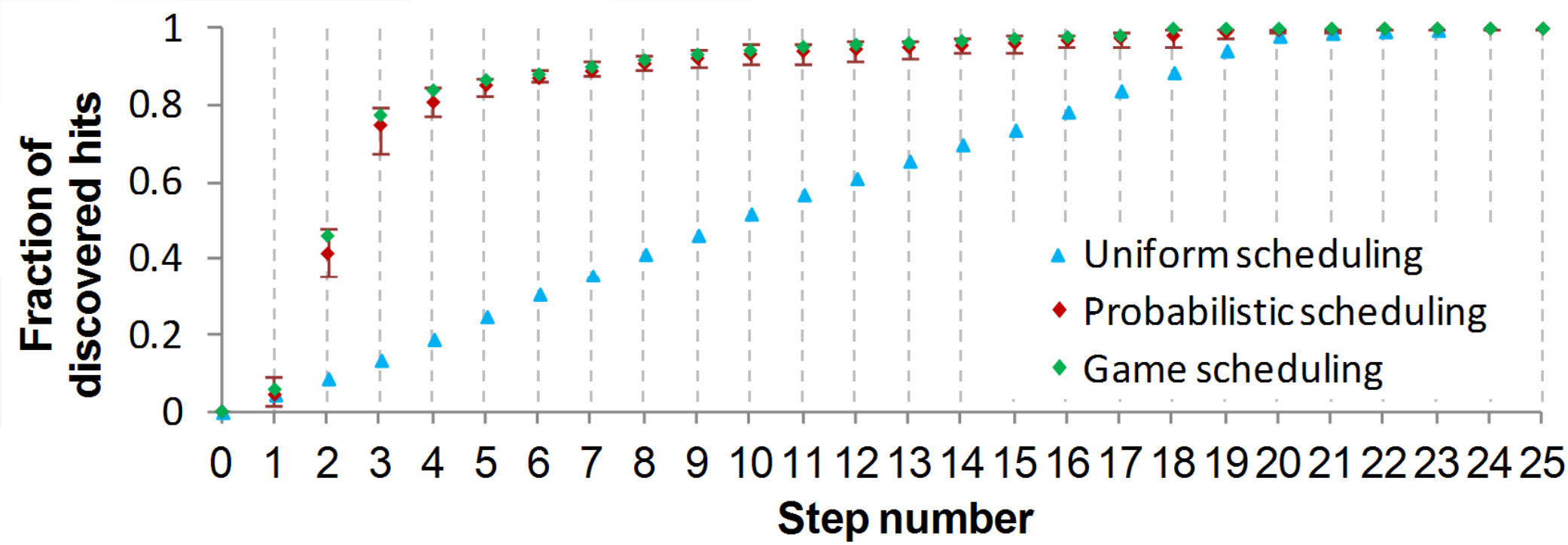


processing rate

fraction of good results

5 step number

Example: Desktop Grid model, heterogeneous nodes, heterogeneous tasks



The proposed solution outperforms probabilistic scheduling about 10% in the early stages.

Congestion Game Scheduling for Virtual Drug Screening Optimization. N. Nikitina, E. Ivashko, A. Tchernykh. Journal of Computer-Aided Molecular Design, 2018.

New mathematical model

ops_i is the performance of the computational node (number of operations per second)

θ_j is the complexity of a computational task (number of operations)

τ is the considered time interval

n_j is the number of players that have chosen block T_j

$\delta(n_j) = \dfrac{1}{1 = n_j}$ is the congestion coefficient of block T_j

$U_{ij} = (\alpha_i \delta(n_j) + (1 - \alpha_i)\sigma_j)\dfrac{ops_i}{\theta_j}$ usefulness of node C_i which chooses block T_j

$\vec{s} = (s_1, \ldots, s_M)$ is the strategy profile (blocks selected by each player)

α is the balance parameter between block congestion level and prospectivity

Limitations

- Necessity to solve the optimisation problem by the project server → affects its performance
- Discretisation of the moments of task distribution → the endings of all computations performed by all nodes must be synchronised

Algorithm

Deployment of a specific procedure of information exchange between the nodes(not present in BOINC by default) allows using a shared file resource in order to transfer the necessity of decision making to the computing nodes themselves. With such enrichment, the solution of the optimisation problem is simplified,the form of utility functions allows to get rid of discretisation and the necessity of synchronisation of the computing nodes. At the same time, the server keeps the capability of dynamic balance between the number of useful results and the search scope.

Server

0. Input data: apriori values of parameters; generated tasks.
1. If a client I asks for a task, solve equation (5), send a task from the block r*.
2. Update w*r. Update α if necessary.
3. If a client sends a result, receive it and update parameters. Update α if necessary.

Client

0. Set resources for the project.
1. Ask for tasks.
2. Perform computations.
3. Send the result to the server

Conclusion

- A congestion game is proposed to model task scheduling in a Desktop Grid for virtual drug screening
- The equilibrium solution describes the balance between the number of hits and their chemical diversity

Thank you for your attention!