

Congestion Game Scheduling for Virtual Drug Screening Optimization

Natalia Nikitina^{1}, Evgeny Ivashko¹, Andrei Tchernykh²*

¹Institute of Applied Mathematical Research, Karelian Research Center, Russian Academy of Sciences,
11 Pushkinskaya str., 185910 Petrozavodsk, Russia

²Computer Science Department, CICESE Research Center,
Carretera Ensenada-Tijuana No. 3918, Zona Playitas, Código Postal 22860, Apdo. Postal 360, Ensenada, Baja California,
Mexico

ORCID: Natalia Nikitina 0000-0002-0538-2939, Evgeny Ivashko 0000-0001-9194-3976, Andrei Tchernykh 0000-0001-5029-5212.

ACKNOWLEDGMENTS. This work is partially supported by the Russian Fund for Basic Research under grants no. 16-07-00622, 15-29-07974 and 16-51-55006, and CONACYT (Consejo Nacional de Ciencia y Tecnología, México) under grant no. 178415.

* Corresponding author; e-mail: nikitina@krc.karelia.ru, tel. +79317016511

Abstract: In virtual drug screening, the chemical diversity of hits is an important factor, along with their predicted activity. Moreover, interim results are of interest for directing the further research, and their diversity is also desirable. In this paper, we consider a problem of obtaining a diverse set of virtual screening hits in a short time. To this end, we propose a mathematical model of task scheduling for virtual drug screening in high-performance computational systems as a congestion game between computational nodes to find the equilibrium solutions for best balancing the number of interim hits with their chemical diversity. The model considers the heterogeneous environment with workload uncertainty, processing time uncertainty, and limited knowledge about the input dataset structure. We perform computational experiments and evaluate the performance of the developed approach considering organic molecules database GDB-9. The used set of molecules is rich enough to demonstrate the feasibility and practicability of proposed solutions. We compare the algorithm with two known heuristics used in practice and observe that game-based scheduling outperforms them by the hit discovery rate and chemical diversity at earlier steps. Based on these results, we use a social utility metric for assessing the efficiency of our equilibrium solutions and show that they reach greatest values.

Keywords: drug discovery, virtual drug screening, high-performance computing, high-throughput computing, desktop grid, game theory, congestion game

1. Introduction

Drug development can be time-consuming, difficult, and fraught with risks and setbacks. The Pharmaceutical Research and Manufacturers of America reports that it takes up to 10-15 years to develop a new market-available drug [1]. One of the first stages of this process is an identification of a set of chemical compounds called *hits* with measured desired biochemical activity which is predicted to translate into in vivo efficacy. Hits are identified among a set of *ligands*, low-molecular compounds able to form a biochemical complex with a protein molecule responsible for disease progression, called a *target*. At next stages, the hits are exposed to certain selection and optimization to form a set of *leads* that apparently have desired biochemical activity. The following optimization of leads is even more rigorous. Among many prospective compounds, only few proceed by the full cycle of drug development.

The automated method for exhaustive search of hits and leads among chemical compounds in a laboratory is High-Throughput Screening (HTS) [2]. This method requires expensive laboratory equipment and has certain physical restrictions undermining both the range of its applicability and experimentation cost.

With computer development of highly accurate specific disease models to validate targets and bind ligands to targets, *virtual screening* [3] (VS) has emerged as an *in silico* alternative to HTS.

In the course of VS, one can model the interaction of the candidate ligands with the target and score the resulting molecular complexes. The ligands with high scores become hits. VS is not restricted to molecule libraries of existing chemical compounds, but it can be implemented for molecules that have never been synthesized before, or for their fragments. The typical duration of this stage of a drug development is from about 0.5 to 5 years.

In general, VS methods can be divided into two main groups: ligand-based and structure-based. If there exists a known lead, then ligand-based screening can be applied

to find or construct structurally similar compounds. If the target structure and functions are well explored, the structure-based methods that search ligands in a library can be useful. In this paper, we restrict ourselves to the structure-based VS, which has many successful applications to discover novel ligands [4-6].

The chemical space of all small organic molecules, each of which might become a drug for studied disease, is estimated [7] to have the order of 10^{60} . Prepared libraries that are used in VS setups typically have size from hundreds of thousands to tens of millions of molecules [8-11].

The largest database with open access is GDB-17 [12]. It contains more than 166 billion known molecules of up to 17 atoms and allows performing fast ligand-based VS using molecular similarities [12]. Performing VS over such large databases essentially requires high-performance computational facilities such as computing clusters and grids. However, enormous size of databases makes them hardly suitable for exhaustive structure-based VS. It could take over five months at one of the world's fastest supercomputers Tianhe-2 [13].

With proper VS organization, the interim results can significantly contribute to the boost of research progress. Considering the long time interval to screen the whole ligand library, it is important to obtain a diverse set of hits as soon as possible, so that they could proceed to the next stage of research without waiting for the rest of them. Based on the feedback from laboratory tests, new knowledge about the disease target and search space can be obtained to accordingly refine VS algorithms and dataset structures.

In this paper, we propose a method for filtering the explored ligands space on the fly when performing structure-based VS to obtain diverse and useful results in a short time. Our algorithm is primarily useful for early iterations of VS. The method does not depend on the docking algorithm implementation and the quality of protein and ligands models used for VS. To this end, we introduce a new function of social utility to evaluate the

congestion game model and efficiency of equilibrium solutions.

The model considers the heterogeneous environment with uncertainty of processing time and workload [14], and limited knowledge about the input dataset structure.

The rest of the paper has the following structure. In Section 2, we overview the related work on achieving high quality VS results in a short time. In Section 3, we define the preliminaries, present the mathematical model, and illustrate the proposed model by an example. In Section 4, we provide the results of the computational experiments. In Section 5, we conclude the paper with result discussion and future work proposals.

2. Related Work

According to drug development principles, two primary characteristics of the compounds resulting set to be tested in the laboratory are efficiency of their interaction with the target protein and chemical diversity [15-19]. In general, these two objectives must be balanced depending on drug designers' preferences.

In structure-based VS, scoring and ranking of the compounds based on the estimated efficiency of interactions with the target is performed by molecular docking software that includes various scoring/ranking methods.

Different authors define the chemical diversity in different ways, but all of them use molecular descriptors to define a chemical space [23]. In this paper, we suggest that the chemical space can be divided into non-overlapping blocks following some rules, and we consider the diversity of a molecules set as the measure how well they cover all defined blocks. We focus on the task scheduling to provide the best diversity of results for a given scoring/ranking method.

It has been empirically proven [20] that, for a HTS stage, the set of compounds to screen should contain as many diverse clusters as possible, even, if a cluster contains only one or few molecules. This increases the probability of finding leads. However, preparation of, even, small diverse library for HTS and VS is a time- and resource-consuming problem.

The most conventional method to reduce input dataset for VS down to manageable size is pre-filtering of the chemical space, leaving only representative and chemically diverse compounds that promise to show desired biological activities.

Many compound libraries, used in a drug development, were created with proper filtering of the chemical space [9-11]. To do this, there should be sufficient knowledge about chemical and biological properties of disease target. Pre-filtered libraries typically have small size and high hit-rate, and are used straightforwardly in HTS rather than VS [21].

In other cases, the resulting library is large. VS results are redundant and require post-processing. Moreover, many compound classes that are potentially good for a specific target, but do not comply with general

considerations, can be filtered out before VS [15]. It undermines the quality of results. Hence, it is important to develop new VS methods efficient for large datasets.

The methods of pre-filtering a large database of compounds (as well as the methods of post-filtering VS results to attain high diversity) can be divided into four basic groups [22,23]:

1. Dissimilarity-based methods that iteratively fill the library with compounds least similar to already selected ones.
2. Clustering methods that define similarity clusters and fill the library with selected representatives from them.
3. Partitioning or cell-based methods that describe the chemical space of manageable dimensionality (typically, up to five-six dimensions) and fill the library to cover the largest subspace.
4. Optimization methods: genetic algorithms, simulated annealing, Monte-Carlo method etc.

Particularly, the third and fourth groups are proven to have higher speed than the first and second ones, and to be effective for very large libraries. In computational experiments, we use the third group of methods. But our algorithm is not restricted by them and can base, for example, on compounds clustering.

A recent example of effective genetic algorithms application [24] has been published in 2015. The authors explore the chemical space by creating new generations of molecules, filtering them by desirable properties, and selecting maximally diverse sets of results. Later, the method was extended [25] for performing structure-based VS. The algorithm demonstrates high efficiency in selecting diverse subsets of molecules with desired properties from large databases.

3. Congestion Game Model

3.1. Motivation

VS essentially involves high-performance computational tools. A recent review on the latest achievements and current state of VS lists more than a hundred successful examples of ligand discoveries in silico [26]. Among them, 25 of 82 papers published in 2008-2015 explicitly state that VS had been performed using High-Performance Computing (HPC) such as computing clusters, supercomputers and grids. Based on the review of 2016, in Figure 1, we illustrate the increase in the number of ligand discovery works accomplished due to VS.

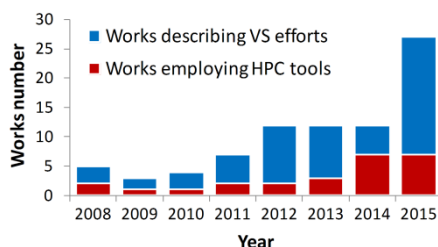


Fig. 1 Number of works describing successful VS examples

In VS, computational tasks are mutually independent, which makes application of High-Throughput Computing (HTC) for VS an intensive research field too. Examples of such systems are Desktop Grids, Enterprise Desktop Grids, volunteer computing systems etc. Several large-scale volunteer computing projects devoted to drug discovery have gathered and employed significant amounts of “gratis” computational resources due to high public interest to such projects.

The most prominent example of such an infrastructure is the World Community Grid [27], which employed the power of over 2.7 million personal computers to support 27 research projects, including drug searches against AIDS, cancer, malaria and other diseases.

In this section, we consider the use of HPC/HTC computational systems for VS. Each computational task contains an independent ligand as input data. With independent molecule models, VS procedure implies that a main node or a server distributes the data among computational nodes with no restrictions on the order of their processing.

Each computational node that requests new work receives a portion of tasks (one task or more) from the server, and processes them independently from other nodes. When the node finishes the tasks, it reports results back to the server. The results are then stored in the database for further usage. Figure 2 illustrates this process.

In general, each task is sent to a node ready to work. However, with heterogeneous nodes and heterogeneous tasks, the scheduling system has variety of options how to assign different tasks to different nodes or the groups of nodes.

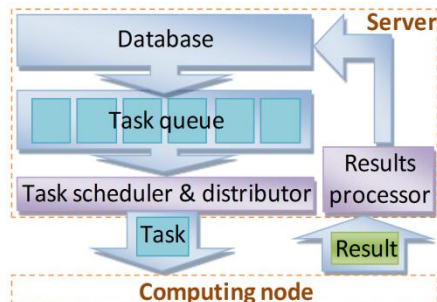


Fig. 2 General workflow of VS in an HPC/HTC system

The described workflow model is common for bag-of-tasks applications made of collections of independent tasks. We use a Desktop Grid simulator to perform

computational experiments considering system heterogeneity.

Due to variations in chemical characteristics, molecules have different chances to show high predicted binding affinity. One can expect that these chances are higher for molecules close in topology to a known ligand [28,29]. In contrast, molecules with large molecular complexity are less likely to become hits [30]. Thus, non-overlapping subsets of molecules in the library could be ranked by their prospectivity for VS.

Once specified, the estimated probabilities can be updated in the course of VS according to interim results. At the same time, results originating from the same subset might be redundant. The idea behind this work is to explore most prospective subsets first while keeping the desired level of diversity by restricting intensity of subsets exploration.

Mathematical game theory studies models of conflict and cooperation between rational intelligent decision-makers. The congestion game [31], first proposed by Rosenthal, 1973, is a useful tool to analyze and solve problems described in the previous paragraph. Its important property is the existence of a deterministic Nash equilibrium that not every strategic game has, which is equivalent to a local-extrema.

In the congestion game, there are players and resources. The payoff of each player depends on the resources he chooses, and the number of players choosing the same resource. As the congestion game is a special case of potential game, the convergence and finite-time convergence of the game iterations are well studied. Heterogeneity of both players and resources complicates the model, but as we show in the next section, existence of a Nash equilibrium is ensured.

3.2. Mathematical Model

Let us consider a computer system with m computational nodes or players $C = C_1, \dots, C_m$, and a set of computational tasks T (data) for VS processing. Each node C_i is characterized by its computational performance ops_i , which is an average number of operations performed in a time unit.

The data set T is divided into n non-overlapping blocks $T = T_1 \cup \dots \cup T_n$ of sizes $N_1^{total}, \dots, N_n^{total}$ such that the estimated portion of VS hits in block T_j will be p_j . We define priority of the block T_j as (Eq. 1).

$$\sigma_j = \frac{p_j}{p_1 + \dots + p_n}, 0 \leq \sigma_j \leq 1. \quad (1)$$

The blocks with higher priority have to be chosen first for processing. We assume that all tasks in block T_j have average computational complexity θ_j , i. e., a number of operations to process one task. Each node selects exactly one block.

The nodes make their decisions at time steps $0, \tau, 2\tau, \dots$. After a node has processed its portion of tasks, it sends the results to the server and is ready for the next portion. Let the utility of node C_i at time step τ express

the amount of useful work performed during this step. This amount depends on the number of executed tasks from the chosen block, its computational complexity, priority, and the number of other nodes who have also chosen this block.

As the VS progresses, the variables p_j and θ_j can be updated according to the interim results. Moreover, the blocks structure may be updated too: blocks can be merged together or divided into smaller blocks, as necessary.

The fewer nodes explore block T_j simultaneously, the more valuable their work is. This condition ensures diversification of the interim set of hits. Let n_j be the number of the players who have chosen the same block T_j at the considered step, and $\delta(n_j)$ be the congestion coefficient for the block (Eq. 2):

$$\delta(n_j) = \frac{1}{n_j}. \quad (2)$$

The utility of node C_i that chooses block T_j is (Eq. 3)

$$U_{ij} = \left(\sigma_j \times \delta(n_j) \right) \frac{ops_i}{\theta_j} \tau, \quad (3)$$

where

- σ_j is the priority of the block T_j (1);
- n_j is the number of the players who have chosen block T_j at the considered step;
- $\delta(n_j)$ is the congestion coefficient (2);
- ops_i is the computational performance of the node C_i ;
- θ_j is the average computational complexity of a task from the block T_j ;
- τ is the length of a time step.

Note that (Eq. 4) holds.

$$U_{ir}(x) = \frac{ops_i}{ops_j} U_{jr}(x) \quad \forall r \in T, \forall x \in N \quad (4)$$

Therefore, at each considered time step, we have a singleton congestion game $G = (C, T, U)$, where C is the set of players (computational nodes), T is the set of data blocks of which each node selects exactly one, and U is the set of utility functions. A strategy profile is a schedule $\bar{s} = (s_1, \dots, s_m)$, where the component $s_i = j$ equals to the block T_j chosen by player C_i .

A strategy profile $\bar{s}^* = (s_1, \dots, s_m)$ is called the Nash equilibrium if no player can benefit from unilaterally changing their strategy. In the considered game, the Nash equilibrium situation means that no node can increase amount of its useful work by unilaterally deviating from the schedule.

So, at each step the schedule \bar{s} is selected so as to satisfy the following inequations:

$$\begin{aligned} \forall i \quad & U_{i,s_i}(s_1^*, \dots, s_{i-1}^*, s_i^*, s_{i+1}^*, \dots, s_m^*) \geq \\ & \geq U_{i,s_i}(s_1^*, \dots, s_{i-1}^*, \dots, s_i, s_{i+1}^*, \dots, s_m^*) \end{aligned} \quad (5)$$

Congestion games have been thoroughly studied in literature. The existence of at least one Nash equilibrium in pure strategies has been proven for the case of identical players [31] and identical task blocks [32]. Moreover, better- and best-response dynamics are guaranteed to converge to equilibrium in polynomial time [32,33].

Heterogeneity complicates the model: utility of each player depends both on its own performance, and on the task complexity of the chosen block. But due to the form of utility functions, as (4) holds, the game G has at least one Nash equilibrium in pure strategies [34]. The Nash equilibrium situation means the best amount of useful work given the current estimates of probabilities and the whole available set of nodes and molecules. Further in this paper we show that the Nash equilibrium, in the majority of cases, corresponds to the best proportion between the chemical diversity and hits number.

3.3. Illustration

Let us consider the following simplified example. Suppose that we have a set of molecules projected onto two-dimensional plane. Each molecule is characterized by two chemical properties x and y (horizontal and vertical axis correspondingly). In this example, we divide the given set of molecules into six equal-size blocks containing 9000 tasks each, and suppose the true fractions of hits in each block are 0.4, 0.01, 0.01, 0.01, 0.25 and 0.25 (Figure 3). This true distribution of hits is not known a priori. For simplicity, we suppose that hits are distributed uniformly in each block.

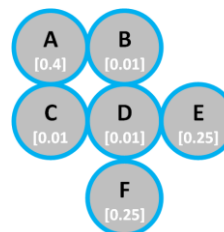


Fig. 3 Blocks and fractions of hits

We suppose that there is a known ligand belonging to block A. We know nothing more about the molecules set. We estimate the expected fractions of hits accordingly: 0.25 for block A (molecules similar to a known ligand by considered chemical properties), and 0.1 for each other block from B to F.

Let us consider a computational system consisting of six identical nodes, and illustrate the work of the algorithm described in this paper.

In this example and further in the paper, performance of the proposed game-theoretical algorithm is compared with two scheduling strategies described in Table 1.

The probabilistic scheduling strategy represents the case when the selection of a task block does not depend on the congestion level of the block, but only on the probability to find a hit.

On the contrary, the uniform scheduling strategy ensures the least possible level of congestion, or the

highest diversity, by distributing the nodes across task blocks as uniformly as possible.

Strategy	Description
1 Game scheduling	Each node selects a task block defined by the Nash equilibrium.
2 Probabilistic scheduling	Each node selects a task block with a given probability of finding a hit.
3 Uniform scheduling	As many diverse blocks as possible are explored simultaneously.

Table 1. Scheduling strategies.

In Figure 4, we show this computational process step-by-step for three considered algorithms provided equilibrium, probabilistic and uniform task distribution.

The numbers inside circles denote predicted fractions of hits calculated after each step. These predicted fractions are updated at each step as more task blocks get explored. The filled part of the circle indicates the fraction of all computed results in a task block.

The first one is the proposed algorithm, where computational nodes select task blocks according to the equilibrium schedule (Figure 4a). At each step, the equilibrium is constructed based on the current estimates of hit probabilities for each block and number of available tasks, according to the mathematical model described in Subsection 3.2.

In order to illustrate the Nash equilibrium concept, let us consider the game-theoretical schedule obtained at the first step. The algorithm (Figure 4a) computes the schedule, where four nodes (say, nodes 1-4) select block A, one node (say, node 5) selects block B, and one node (say, node 6) selects block C. In this schedule, no node will benefit from selecting any other block, if all other nodes do not change their decisions. Indeed, if node 5 changes its decision in favor of block D, E, or F, its expected utility stays the same. If it selects block C joining node 6, the utilities of both nodes decrease. If it selects block A joining nodes 1-4, its utility decreases as well. Thus, the schedule shown in Fig. 4a corresponds to the Nash equilibrium. The second algorithm is probabilistic scheduling: each computational node selects a task block to work on with a given probability of finding a hit (Figure 4b).

The third heuristic is to allocate all nodes to work uniformly on the blocks (Figure 4c).

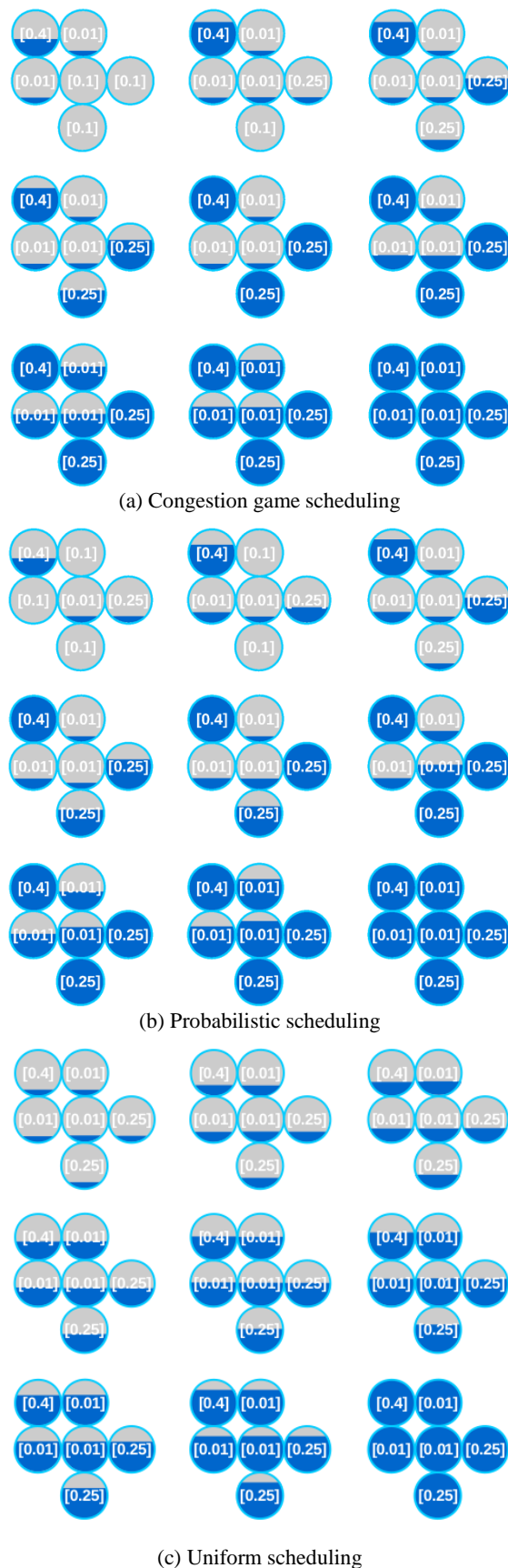


Fig. 4 Computational process step-by-step (from left to right, from top to bottom)

In Figure 5, we show the number of hits obtained at each step, using the proposed algorithm and two alternative heuristics. The diagram illustrates that, starting from the third step, the probabilistic algorithm outperforms two other algorithms, but Figure 6 illustrates that the proposed game scheduling algorithm provides the best quality of interim results sets due to better diversity.

3.4. Social utility

The social utility is one of the key concepts of non-cooperative games. In contrast with individual benefits, it evaluates the quality of a game situation for the whole set of players. To estimate the efficiency of equilibrium solutions, we describe the quality of resulting set obtained at each step in terms of the social utility.

The social utility function reflects the balance between diversity and the number of hits discovered at a step, i.e. it shows the effectiveness of a schedule at each step. According to the proposed model, it depends on the expected number of discovered hits h_r and their chemical diversity under the schedule \bar{s} which all players have agreed on.

The proposed social utility function (Eq. 6) has the following form:

$$SOC(\bar{s}) = \gamma(\bar{s}) \sum_{r=1}^n h_r - \frac{\max(h_r)}{\sum_{r=1}^n h_r} + \frac{\min(h_r)}{\sum_{r=1}^n h_r}, \quad (6)$$

where

- \bar{s} is the selected schedule;
- $\gamma(\bar{s})$ is the fraction of non-empty blocks chosen by at least one player under the schedule \bar{s} ;
- $h_r = \frac{\tau p_r}{\theta_r} \sum_{i:s_i=r} ops_i$ is the expected number of hits to be obtained from block r using schedule \bar{s} at the considered step.

Let us consider the case with identical computational nodes ($ops_1 = \dots = ops_m = ops$), and tasks of identical complexities ($\theta_1 = \dots = \theta_n = \theta$). Under such assumptions, a schedule can be represented more simply by a vector \bar{n} , where each component n_i is the number of identical players choosing block T_i .

Let $\gamma(\bar{n})$ denote the number of positive components of the vector \bar{n} . The social utility function (Eq. 7) takes then the following form:

$$SOC(\bar{n}) = \frac{ops}{\theta} \frac{\tau}{m} \left(\gamma(\bar{n}) \sum_{i=1}^n p_i n_i - \frac{\max(p_i n_i)}{\sum_{i=1}^n p_i n_i} + \frac{\min(p_i n_i)}{\sum_{i=1}^n p_i n_i} \right), \quad (7)$$

where

- \bar{n} is the selected schedule;
- $\gamma(\bar{n})$ is the fraction of non-empty blocks chosen by at least one player under the schedule \bar{n} ;

- $p_i n_i$ is the expected number of hits to be obtained from block i using schedule \bar{s} at the considered step.

In Figure 6, we provide accumulated social utility values at each step of the computational process described in Section 3.3. The diagram illustrates that, for the considered example, the proposed game scheduling algorithm outperforms two others by the quality of interim resulting set. However, the equilibrium solution is not always the optimal one. Moreover, in certain cases, game-theoretical principles of the system operation are not the best one.

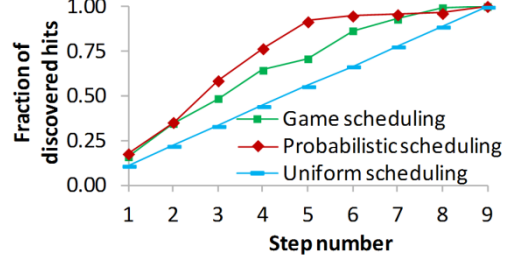


Fig. 5 Fraction of hits discovered at each step

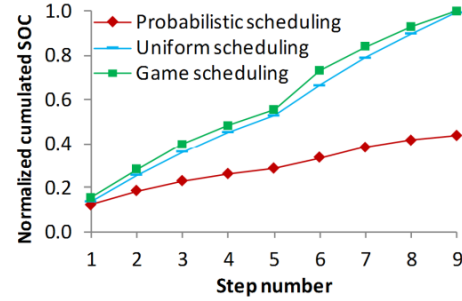


Fig. 6 Accumulated social utility during the computational process

Next, we consider the efficiency of equilibria.

The Price of Anarchy (PoA) (Eq. 8) and the Price of Stability (PoS) (Eq. 9) evaluate the efficiency of Nash equilibria in comparison with non-equilibrium solutions. The price of anarchy is the estimation of a loss in the social utility due to selecting the worst equilibrium schedule. The price of stability is the estimation of the gain obtained due to selecting the best schedule.

$$PoA(\bar{s}) = \frac{\min_{s \text{ is a NE}} SOC(\bar{s})}{\max_s SOC(\bar{s})} \quad (8)$$

$$PoS(\bar{s}) = \frac{\max_{s \text{ is a NE}} SOC(\bar{s})}{\max_s SOC(\bar{s})} \quad (9)$$

PoA and PoS can be considered as estimations of the total gain and loss obtained due to game-theoretical approach, when players act independently. Our simulations show that PoA and PoS differ very slightly or do not differ at all in the proposed game model. Thus, all equilibria are approximately equal in each game instance from the social point of view. In particular, equilibrium solutions

are optimal, when the variance between blocks priorities is close to zero.

4. Experimental Setup

4.1. Database preparation

In order to perform computational experiments and evaluate the performance of the developed approach, we have to consider a molecules database, divide it into blocks and simulate VS. The efficiency of our congestion game approach can be shown by earlier acquisition and higher diversity of hits at early VS stages comparing with known heuristics used in practice.

We perform simulations on a real yet small database and assume a relatively high fraction of hits, for the purpose of illustration. In real settings, a database is much larger, and the fraction of hits can be smaller. The experiment is aimed at verifying the approach and illustrating the results obtained so far.

We use the database GDB-9 of enumerated organic molecules consisting of at most nine atoms of C, N, O, S and Cl (not counting the hydrogen atoms). Being a subset of the database GDB-13 [35], GDB-9 represents about 320 thousand molecules with variety of chemical properties. GDB-9 is manageable for performing computational experiments and can be unambiguously divided into several non-overlapping blocks. Nevertheless, the set of molecules is rich enough to demonstrate the feasibility and practicability of proposed solutions.

Using the *obprop* utility from the Open Babel library [36] for processing chemical data, we calculate several chemical properties for every molecule of the database. After that, we simulate a virtual screening setting without docking programs. The pre-calculated chemical characteristic of our interest are stored in prepared files. As long as we consider a task scheduling algorithm to obtain diverse set of hits in a short time, we can assume

that the specific implementation of the docking algorithm unimportant for performance analysis. The expected time of one docking task can vary in a real VS setting, but, for the sake of simplicity, we consider it equals for all tasks.

For the experiments, we consider three pre-calculated chemical properties of each molecule: the total number of atoms including hydrogens, polar surface area (PSA), and partition coefficient logP. We do not consider the time taken for the calculation of properties and clustering, because it is equal for all three algorithms. In practice, this time can be important to consider.

Over GDB-9, the total number of atoms including hydrogens varies from 4 to 31. The PSA values vary in range from 0 to 118.35.

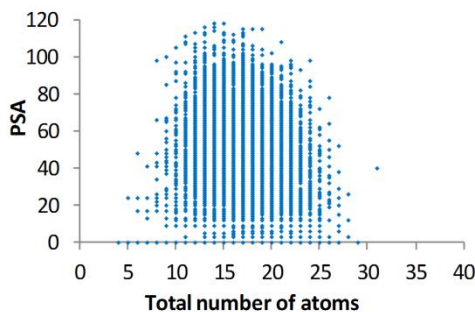
As 3160 (or 0.99%) of molecules in GDB-9 have $\log P \geq x = 2.7765$, the value $x = 2.7765$ is taken as a threshold to count a molecule as a hit.

In Figure 7a and Figure 7b, we show that considered properties are distributed inhomogeneously over the database. If the molecule space is divided into smaller blocks for further search of molecules, then frequencies of hits in those blocks would differ.

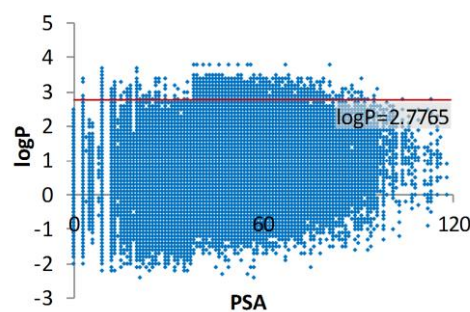
Two other chemical properties, the total number of atoms and the PSA, are used for charting the molecules database into two dimensions, and defining the blocks for search (see Figure 8).

According to the model, the molecules in different blocks should have different combinations of chemical properties. We set block borders in such a way that there is minimal variance between block sizes, taking into account repetitive values.

The upper number in bold, in each block, defines the number of its elements, and the lower one defines the fraction of hits in the block. The white letter in the upper right corner is the block name.



(a) Values of PSA for different number of atoms



(b) Values of logP for different polar surface area

Fig. 7 Distribution of molecule properties in GDB-9

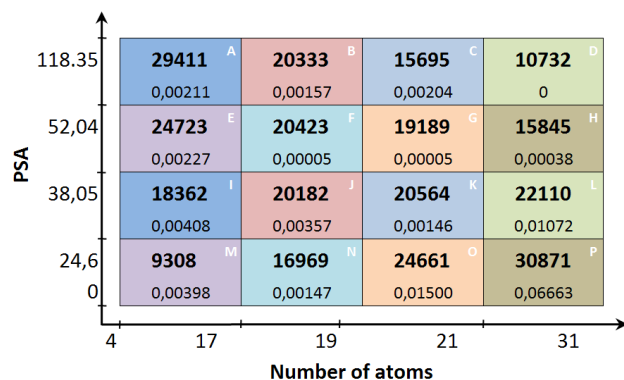


Fig. 8 Decomposition of GDB-9 database into non-overlapping blocks

4.2. Experimental setup

As a first test case, we consider the case with identical computational nodes, tasks of identical complexities, and VS settings listed in Table 2.

Parameter	Value	Description
n	16	Number of task blocks
m	64	Number of computational nodes
ops	25	Performance of a computational node (number of conditional operations per time unit)
θ	100	Complexity of a computational task (number of conditional operations)
τ	1000	Maximal length of a step (number of time units)
x	2.7765	Threshold of logP value for selecting a hit

Table 2. Parameters of the first series of simulations (identical nodes, identical tasks).

At each VS step, the optimal schedule is computed based on the current knowledge about expected fractions of hits in blocks, as the first equilibrium schedule in lexicographical order. Blocks are selected for processing according to the schedule. After completion of all blocks in the step, the expected fractions of hits are updated according to the number of hits discovered in each block. Then a next step is performed, etc.

The number of tasks to be executed by the computational system at each step has been calculated in advance. In such a way, we consider the length of each step fixed and defined in advance. Also, in the simulation, we do not take into account the overhead for making scheduling decisions.

At the second test case, we consider a Desktop Grid system with heterogeneous computational nodes and heterogeneous tasks. The parameters of the simulations are provided in Table 3.

Parameter	Value	Description
n	16	Number of task blocks
m	64	Number of computational nodes
ops	15 (nodes C ₁ -C ₁₆)	Performance of a

	20 (nodes C ₁₇ -C ₃₂)	computational node
	25 (nodes C ₃₃ -C ₄₈)	(number of conditional
	30 (nodes C ₄₉ -C ₆₄)	operations per time unit)
θ	100 (nodes T ₁ -T ₄)	Complexity of a
	125 (nodes T ₅ -T ₈)	
	125 (nodes T ₉ -T ₁₂)	
	150 (nodes T ₁₃ -T ₁₆)	
τ	1000	Maximal length of a step
		(number of time units)
x	2.7765	Threshold of logP value for
		selecting a hit

Table 3. Parameters of the second series of simulations (heterogeneous nodes, heterogeneous tasks).

4.3. Experimental analysis

In Figure 9 (a) and (b), we show the rate of discovery hits during the first and second test cases, respectively. Each discrete point represents the fraction of the total amount of hits discovered at the corresponding step. The quality measure of such curves is their proximity to the upper left corner of the chart.

The probabilistic scheduling results are, in both cases, averaged over 100 runs. The red whiskers denote the observed ranges of the measured values for probabilistic scheduling. The lower point of each whisker corresponds to the worst observed value, the upper one to the best one.

Presented results indicate that at the first step, all three scheduling strategies show the same performance, due to the same apriori assumptions about the probabilities of finding hits. Starting from the second step, with the updated probabilities, performances differ.

In terms of the fraction of discovered hits, the uniform scheduling has proven to be very inefficient at steps 2-20 (first test case) and 2-19 (second test case). The game scheduling and the probabilistic scheduling have approximately equal performances on the average, but the whiskers on the plots illustrate that indeterminations make the probabilistic algorithm inefficient.

In the "tail" phase of computations, starting from step 21 (both test cases), three algorithms show approximately equal performances again. This is connected, in the first place, with the fact that about 100% of hits have been found before, and the least prospective blocks (such as

blocks D, F, G, and H in Fig. 8) are being explored last of all in all three scheduling algorithms.

The performances in terms of obtained chemical diversity differ more significantly. To illustrate this, in Figure 10 (a) and (b), we provide the normalized cumulated social utility obtained during the first and the second series of simulations, respectively. Throughout the process, intrinsic indeterminations of the probabilistic algorithm lead to substantial reduction in its performance. The game scheduling algorithm is preferable at the early steps, namely in the first half of the computational process (first test case) and in the first third of the computational process (second test case). In the remaining steps, uniform scheduling becomes the most efficient strategy.

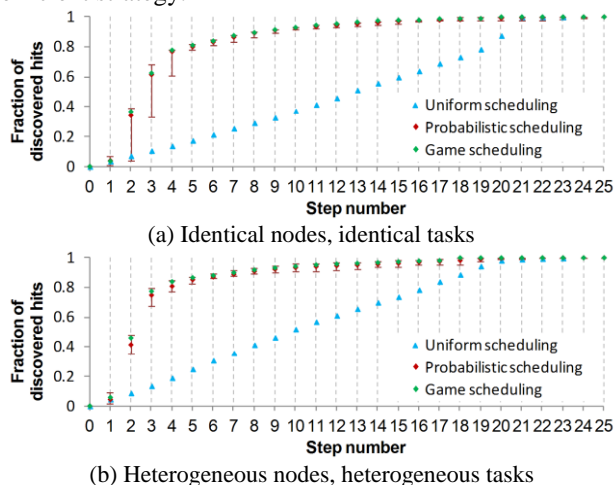


Fig. 9 Fraction of hits discovered at each step of simulations

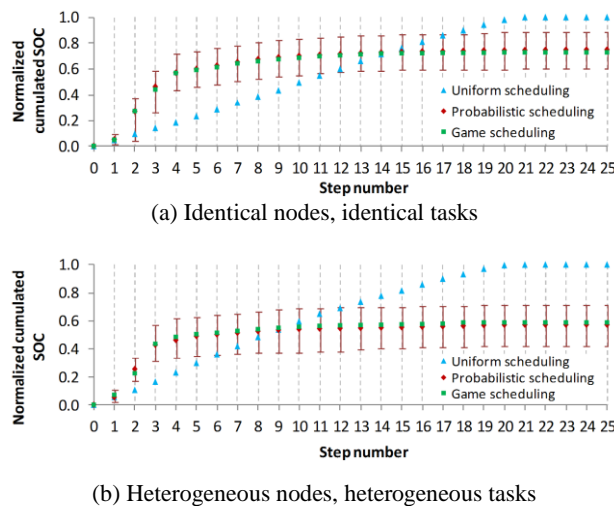


Fig. 10 Cumulated social utility obtained at each step of simulations

5. Conclusion and discussion

We address a problem of task scheduling and optimization of virtual drug screening in HPC systems. The objective is to obtain a diverse set of virtual screening hits in the short time. In order to achieve this,

we define task subsets and construct a congestion game model with computational nodes used as players, who choose specific subsets for processing. The model considers heterogeneous environment with resource availability and processing time uncertainties. We show that, with limited knowledge about the input dataset structure, the equilibrium solution corresponds to the best balance between the number of interim hits and their chemical diversity.

We compare the results of the proposed scheduling algorithm with two known heuristics used in practice, and demonstrate that game-based scheduling outperforms both heuristics by the rate of discovery hits and their chemical diversity at early steps. We also use the social utility metric to show that our game-based schedules reach its greatest values.

The proposed method can be used by different HPC systems. For example, in [37] we present the implementation of this algorithm in BOINC-based Desktop Grid and provide a pseudocode.

However, further study is required to assess its actual performance and effectiveness in a laboratory. This will be the subject of future work. Moreover, the improvement of game equilibrium solutions stability and selection of the most efficient algorithms among all equilibria are other important issues to be addressed. The developed scheduling algorithms will be implemented as open software.

REFERENCES

1. Pharmaceutical Research and Manufacturers of America (PhRMA). 2016 Biopharmaceutical Industry Profile (2016) <http://phrmaconnectionsmedia.com/sites/default/files/pdf/biopharmaceutical-industry-profile.pdf>. Accessed 21 November 2017
2. Bleicher K et al. (2003) Hit and lead generation: beyond high-throughput screening. *Nat. Rev. Drug Discov.* 2:369-378
3. Bielska E et al. (2011) Virtual screening strategies in drug design — methods and applications. *BioTechnologia* 92(3):249-264
4. Sosić I et al. (2016) Nonpeptidic selective inhibitors of the chymotrypsin-like ($\beta 5$ i) subunit of the immunoproteasome. *Angew. Chem. Int. Ed.* 55(19):5745-5748
5. Cavasotto CN, Palomba D (2015) Expanding the horizons of G protein-coupled receptor structure-based ligand discovery and optimization using homology models. *Chem. Commun.* 51(71):13576-13594
6. Leung CH et al. (2011) A natural product-like inhibitor of NEDD8-activating enzyme. *Chem. Commun.* 47(9):2511-2513
7. Bohacek RS, McMartin C, Guida WC (1996) The art and practice of structure-based drug design: a molecular modeling perspective. *Med Res Rev* 16(1):3-50
8. Irwin J et al. (2012) ZINC: a free tool to discover chemistry for biology. *J. Chem. Inf. Model.* 52:1757-1768
9. Bento AP et al. (2014) The ChEMBL bioactivity database: an update. *Nucleic Acids Res.* 42:1083-1090
10. Pence HE, Williams A (2010) ChemSpider: an online chemical information resource. *J. Chem. Educ.* 87(11):1123-1124
11. Bolton EE et al. (2008) In: Wheeler RA and Spellmeyer DC (eds) *Annu Rep Comput Chem.* 4:217-241

12. Ruddigkeit L et al. (2012) Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. *J. Chem. Inf. Model.* 52:2864-2875
13. Liu T et al. (2016) Applying high performance computing in drug discovery and molecular simulation. *Natl Sci Rev.* 3(1):49-63
14. Tchernykh A et al. (2015) Towards understanding uncertainty in cloud computing resource provisioning. *Procedia Comput Sci.* 51:1772-1781
15. Muegge I, Oloff S (2006) Advances in virtual screening. *Drug Discovery Today Technol.* 3(4):405-411
16. Wale N, Karypis G (2007) Methods for effective virtual screening and scaffold-hopping in chemical compounds. *Comput. Syst. Bioinformatics Conf.* 6:403-416
17. Krüger DM, Evers A (2010) Comparison of structure- and ligand-based virtual screening protocols considering hit list complementarity and enrichment factors. *ChemMedChem* 5(1):148-158
18. Tanrikulu Y, Krüger B, Proschak E. (2013) The holistic integration of virtual screening in drug discovery. *Drug Discovery Today* 18(7/8):358-364
19. Lionta E et al. (2014) Structure-based virtual screening for drug discovery: principles, applications and recent advances. *Curr. Top. Med. Chem.* 14:1923-1938
20. Harper G, Pickett SD, Green DV (2004) Design of a compound screening collection for use in high throughput screening. *Comb. Chem. High Throughput Screening* 7(1):63-70
21. Harris CJ et al. The design and application of target-focused compound libraries. *Comb. Chem. High Throughput Screening* 14:521-531
22. Böhm M (2011) In: Sottriffer C (ed) *Virtual screening: principles, challenges, and practical guidelines*, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, Germany
23. Leach AR, Gillet VJ (2007) *An introduction to chemoinformatics*. Springer Science & Business Media
24. Rupakheti CR et al. (2015) Strategy to discover diverse optimal molecules in the small molecule universe. *J. Chem. Inf. Model.* 55:529-537
25. Rupakheti CR (2015) Property biased-diversity guided explorations of chemical spaces. PhD dissertation, Duke University
26. Irwin J, Shoichet BK (2016) Docking screens for novel ligands conferring new biology. *J. Med. Chem.* 59(9):4103-4120
27. World Community Grid. <http://www.worldcommunitygrid.org>. Accessed 21 November, 2017
28. Patterson DE et al. (1996) Neighborhood behavior: a useful concept for validation of "molecular diversity" descriptors. *J. Med. Chem.* 39:3049-3059
29. Willet P, Barnard JM, Downs GM (1998) Chemical Similarity Searching. *J. Chem. Inf. Comput. Sci.* 38(6):983-996
30. Hann MM, Leach AR, Harper G (2001) Molecular complexity and its impact on the probability of finding leads for drug discovery. *J. Chem. Inf. Comput. Sci.* 41:856-864
31. Rosenthal R (1973) A class of games possessing pure-strategy Nash equilibria. *Int J Game Theory* 2(1):65-67
32. Milchtaich I (1996) Congestion games with player-specific payoff functions. *Games Econ Behav* 13:111-124
33. Jeong S et al. (2005) Fast and compact: a simple class of congestion games. *AIII Proceedings* 1-6
34. Gairing M, Klimm M (2013) Congestion games with player-specific costs revisited. *International Symposium on Algorithmic Game Theory*, Springer Berlin Heidelberg
35. Blum LC, Raymond J-L (2009) 970 Million druglike small molecules for virtual screening in the chemical universe database GDB-13. *J. Amer. Chem. Soc.* 131(25):8732-8733
36. O'Boyle NM et al. (2011) Open Babel: an open chemical toolbox. *J. Cheminform* 3(1):33
37. Nikitina N, Ivashko E, Tchernykh A. (2017) Congestion Game Scheduling Implementation for High-Throughput Virtual Drug Screening Using BOINC-based Desktop Grid. 14th Int. Conf. PaCT 2017, LNCS 10421:480-491