Adaptive Scheduling for Adjusting Retrieval Process in BOINC-based Virtual Screening

Natalia Nikitina[0000−0002−0538−2939](✉) and Evgeny Ivashko[0000−0001−9194−3976]

Institute of Applied Mathematical Research, Karelian Research Center of the Russian Academy of Sciences, Petrozavodsk, Russia
nikitina@krc.karelia.ru, ivashko@krc.karelia.ru

Abstract. This work describes BOINC-based Desktop Grid implementation of adaptive task scheduling algorithm for virtual drug screening. The algorithm bases on a game-theoretical mathematical model where computing nodes act as players. The model allows to adjust the balance between the results retrieval rate and the search space coverage. We present the developed scheduling algorithm for BOINC-based Desktop Grid and evaluate its performance by simulations. Experimental analysis shows that the proposed scheduling algorithm allows to adjust the results retrieval rate and the search space coverage in a flexible way so as to reach the maximal efficiency of a BOINC-based Desktop Grid.

Keywords: Desktop Grid · BOINC · Scheduling · Virtual screening · Game theory · Congestion game

1 Introduction

High-performance computing (HPC) plays a significant role in implementing contemporary fundamental and applied research, developing new materials, new medicines, new types of industrial products. To perform HPC, computational clusters are often used. Should particularly large amounts of resources be required, one can also deploy Grid systems integrating computational clusters. Computing resources can be also provided on-demand using commercial cloud-based services. One more option is the use of Desktop Grids. The term stands for a distributed high-throughput computing system which uses idle time of non-dedicated geographically distributed computing nodes connected over low-speed (as opposed to supercomputer interconnect) regular network. In common case the nodes are either personal computers of volunteers connected over the Internet (volunteer computing) or organization desktop computers connected over local area network (Enterprise Desktop Grid). Desktop Grids can also be integrated into computational clusters or Grid systems (see [1,2] for examples).

As a Desktop Grid is a high-throughput computing tool, it is aimed at processing huge numbers of tasks. Usually, when solving such problems, one does not aim to find a particular answer, but rather to select among a large number of prospective solutions candidates for more detailed evaluation by the scientists,

for instance, in a laboratory. One of such problems is virtual screening where a HTC tool is used to perform computer modelling of interaction between a target protein and a prospective ligand; molecules with high predicted energy of interaction with the target are studied in detail in laboratories. For such problems, it is important not only to provide high performance when solving them, but also to organize computations in such way that would ensure the balance between the results retrieval rate and the search space coverage.

In this paper, we propose a task scheduling algorithm for adjusting the balance between the results retrieval rate and the search space coverage when performing computations in a BOINC-based Desktop Grid. The basis of the algorithm is a mathematical model which considers the heterogeneous Desktop Grid environment and the limited knowledge about the input dataset structure. The algorithm has been developed for solving the problem of virtual screening, but it can be also used for solving other computationally intensive search problems, where one needs to balance between the results retrieval rate and the search space coverage, probably with limited apriori knowledge about the input dataset.

The rest of the paper has the following structure. In Section 2, we provide the motivation for this work. In Section 3, we describe the methodology used. In Section 4, we provide and analyze the results of the computational experiments. In Section 5, we overview the related work. Finally, in Section 6, we conclude the paper with result discussion and directions of future work.

2 Motivation

2.1 Desktop Grids

There is a number of approaches to implement a Desktop Grid. There can be peer-to-peer, hierarchical, and other types of Desktop Grid. The diversity of high-level architectures of Desktop Grids has been described, for instance, in [3]. In our work, we consider the Desktop Grid which follows the server-client model, as shown in Fig. 1. The server holds a large number of tasks that are mutually independent pieces of a computationally heavy problem. When a computing node (or a client) is idle, it communicates with the server and requests work. The server replies by sending one or more independent tasks. The node processes them and reports results back to the server. The results are then processed and can be, for instance, stored in the database for further usage. Such architecture has been described in a number of works ([4,5] etc.).

The middleware systems for Desktop Grid operation also vary widely. However, the open source BOINC platform [4] is nowadays considered as *de facto* standard. Since 1990s, BOINC has been a framework for many independent volunteer computing projects. Today, it is the most actively developed Desktop Grid middleware, which supports the widest range of applications.

BOINC is based on server-client architecture, where the workflow proceeds as described above. The client part is able to work at an arbitrary number of computers with various hardware and software characteristics.

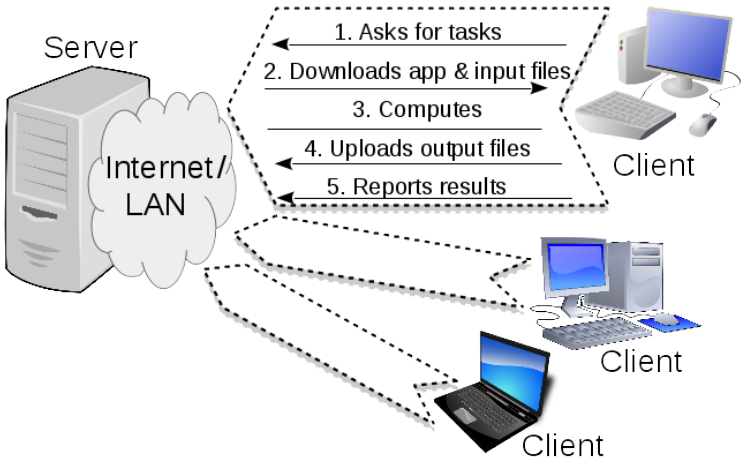


Fig. 1. Desktop Grid

With a variety of scheduling mechanisms implemented in BOINC, there is still a number of challenges one faces when solving a computationally intensive problem. Such challenges arise due to the heterogeneity and internal uncertainties present in Desktop Grid systems as well as specific requirements imposed by the field of research.

2.2 Virtual drug screening

Virtual drug screening [6] (further VS) refers to the creation of new medicines, a time-consuming process with high costs of research and development. It allows to bring *in silico* the first stage of drug development process, namely the identification of a set of chemical compounds called *hits* with predicted desired biochemical activity. 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*. In the course of VS, one performs computer modeling of the interaction of the candidate ligands with the target and scores the resulting molecular complexes. The ligands with high scores become hits.

In [7], we overview the problem of structure-based VS over large databases and illustrate the need for fast and efficient hits retrieval methods.

At the same time, discovery of novel chemotypes (e.g. essentially novel ligands for the given target) is considered to be one of the major drivers of VS progress for the next years [8]. The problem of finding novel chemotypes is tightly bound with the problem of retrieval of the most chemically diverse hits, fully covering the chemical space [9,10]. In general case, this objective can contradict the fastest hits retrieval, as the least promising areas of the chemical space must be explored along with the most promising ones.

To summarize, VS is a complex and resource-demanding process, and various challenges arise in its course. Depending on the stage of research process, one or another objective steps forward. With a possibility to adjust the balance between the hits retrieval rate and the search space coverage, one can direct the process of VS so as to achieve its maximal efficiency at the current stage of research.

3 Methodology

Being a computational technique to process large numbers of independent fine-grained tasks, VS essentially involves a set of computational nodes that may be seen as independent agents, each of them willing to maximize the reward they receive for computations. The reward may be expressed, for instance, in terms of virtual credit for CPU time (as it is implemented in BOINC), the number of found hits, or any other expression of useful work the node performs. At the same time, decisions of one node may influence the others in case they access limited shared resources, or compete for the fastest results retrieval, etc. Such presentation allows to apply the methods of mathematical game theory for modeling the computational process of VS.

The mathematical model described in this section has been elaborated in [7]. The model is based on a congestion game, which was first proposed by Rosenthal in 1973 [11]. Its important property is the existence of a deterministic Nash equilibrium. The convergence and finite-time convergence of the game iterations are well studied. Existence of a Nash equilibrium is ensured [12] even in the considered case of heterogeneous players and resources. In order to reach the equilibrium situation, we employ the best-response dynamics [12,13].

The idea lying in the basis of a model is as follows. 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 [14,15]. In contrast, molecules with very large number of atoms are less likely to become hits [16].

Thus, non-overlapping subsets of molecules in the library could be ranked beforehand by their estimated 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 model is designed so as to explore most prospective subsets first while keeping the desired level of diversity by restricting intensity of subsets exploration.

Consider a computer system with m computational nodes — or players — $C_1, \ldots, C_m$, and a set of computational tasks T. Each node is characterized by its computational performance ops_i, which is the average number of operations performed in a time unit. The input set T is divided into non-overlapping blocks $T = T_1 \bigcup \ldots \bigcup T_n$ such that the estimated portion of VS hits in block T_j is p_j. We define priority of the block T_j as

$$\sigma_j = \frac{p_j}{p_1 + \ldots + p_n}. \quad (1)$$

The blocks with higher priority have to be chosen first for processing. We assume that all tasks in block T_j have the 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 $0, \tau, 2\tau, \ldots$. 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.

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 block T_j at the considered step, and $\delta(n_j)$ be the congestion coefficient for the block, which in the simplest case takes form

$$\delta(n_j) = \frac{1}{n_j + 1} \tag{2}$$

The utility of node C_i that chooses block T_j is

$$U_{ij} = (\alpha_i \, \delta(n_j) + (1 - \alpha_i) \, \sigma_j) \, \frac{ops_i}{\theta_j}. \tag{3}$$

Here, $\alpha_i \in [0; 1]$ is the parameter to control the balance between block congestion level $\delta(n_j)$ and block prospectivity σ_j. As it tends to zero, the player C_i gains maximal profit of getting the most possible number of hits at a step ("digger"). On the contrary, as α_i tends to one, the player gains maximal profit of selecting the blocks with minimal presence of other players ("explorer").

In such way, different players can have different preferences expressed by the value of parameter α_i. By fixing the preferences, one can direct the computational process as a whole. In Section 4, we consider the game where all players have the same preference $\alpha = \alpha_1 = \ldots = \alpha_m$, varying from 0 to 1.

Therefore, at each considered time step, we have a singleton congestion game $G = \langle C, T, U \rangle$, 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* $\boldsymbol{s} = (s_1, \ldots, s_m)$, where the component $s_i = j$ means that player C_i selects block T_j.

4 Experimental analysis

In order to perform computational experiments and evaluate the performance of the developed approach, we divide a molecules database into blocks and simulate VS. We prove the efficiency and flexibility of the proposed algorithm by showing the influence of scheduling parameter α on results retrieval rate and search space coverage.

As in [7,5], 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 hydrogen). GDB-9 represents about 320 thousand molecules with variety of chemical

properties. The chosen database is manageable for performing computational experiments and can be unambiguously divided into several non-overlapping blocks. At the same time, the set of molecules is rich enough to demonstrate the feasibility and practicability of proposed solutions.

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 $logP$. Basing on these properties, we divided the database into 16 non-overlapping task blocks. For the sake of time, we do not compute the predicted binding energy as it would be done in a real VS setting. Instead we use the pre-calculated value $logP$. As 0.99% of molecules in GDB-9 have $logP \geq x = 2.7765$, the value $x = 2.7765$ has been taken as a threshold to count a molecule as a hit.

We use two of the considered chemical properties, the total number of atoms and the calculated PSA, for charting the molecules database in two dimensions, and defining blocks of tasks. The resulting decomposition is provided in Table 1. In each cell, the upper number (in bold) stands for the total number of molecules/tasks in the corresponding block. The lower number stands for the hits fraction in this block.

Table 1. Decomposition of GDB-9 database into non-overlapping blocks.

		PSA			
		$[0.0, 24.6)$	$[24.6, 38.05)$	$[38.05, 52.04)$	$[52.04, 118.35]$
Number of atoms	$[4, 17)$	**9308** 0.00398	**18362** 0.00408	**24723** 0.00227	**29411** 0.00211
	$[17, 19)$	**16969** 0.00147	**20182** 0.00357	**20423** 0.00005	**20333** 0.00157
	$[19, 21)$	**24661** 0.01500	**20564** 0.00146	**19189** 0.00005	**15695** 0.00204
	$[21, 31]$	**30871** 0.06663	**22110** 0.01072	**15845** 0.00038	**10732** 0

We perform computational experiments, simulating virtual screening in a heterogeneous Desktop Grid consisting of 64 computing nodes. The simulations have been implemented within the Center for collective use "High-performance computing center" of Karelian Research Center. The parameters of the simulations are summarized in Table 2.

In Fig. 2, we show the overall process of VS over GDB-9 database for 4 fixed values of parameter α. We observe that the difference in results retrieval rate can be drastic for different values of α, unless the VS process has entered the "tail" phase where all or nearly all hits have been found.

Table 2. Parameters of the simulations.

Parameter	Value	Description
n	16	Number of task blocks
m	64	Number of computing nodes
ops	15 (nodes C_1–C_{16})	
	20 (nodes C_{17}–C_{32})	Performance of a computing node
	25 (nodes C_{33}–C_{48})	(number of conditional operations per time unit)
	30 (nodes C_{49}–C_{64})	
θ	75 (blocks T_1–T_4)	
	100 (blocks T_5–T_8)	Complexity of a computational task
	125 (blocks T_9–T_{12})	(number of conditional operations)
	150 (blocks T_{13}–T_{16})	
τ	100	Maximal length of a step (number of time units)
x	2.7765	Threshold of $logP$ value for selecting a hit

Further, we investigate the influence of the scheduling parameter α value on the characteristics of the computational process.

Fig. 3 depicts the results retrieval rate and search space coverage, both averaged over steps 1–5 (a) and 21–25 (b), as α varies from 0.0 to 1.0. We observe that the search space coverage has a positive correlation with the scheduling coefficient α, while the results retrieval rate has a negative one.

With α varying from 0.0 to 1.0 with step 0.05, we expectedly observe the "scissors" of the two observed characteristics. The diagram shows that the case corresponding to zero influence of block congestion ($\alpha = 0.0$) is relatively inefficient both in terms of hits retrieval rate and search space coverage. This is due to the fact that at every step, all players select the same block which appeared to be most prospective at a previous step. Note that with $\alpha = 0.0$, the scheduling algorithm corresponds to the probabilistic heuristic described in [7].

Fig. 3 also shows that at full search space coverage, the results retrieval rate may decay with increasing α. Due to this reason, it may be inefficient to set the scheduling coefficient α close to 1.0. With $\alpha = 1.0$, all blocks are considered equal, and the scheduling algorithm corresponds to the uniform heuristic described in [7].

To summarize, the experiments show that with α changing in range [0.0, 1.0], it is possible to adjust both scheduling characteristics — the results retrieval rate and the search space coverage — as desired. Borderline values $\alpha = 0.0$ and $\alpha = 1.0$ are relatively inefficient, which follows from the mathematical model. As α increases within the borders (0.0, 1.0), the former characteristic decreases and the latter increases.

5 Related work

In this section we briefly overview the present state of the art of game-theoretical approaches to scheduling in Desktop Grids and scheduling for VS.

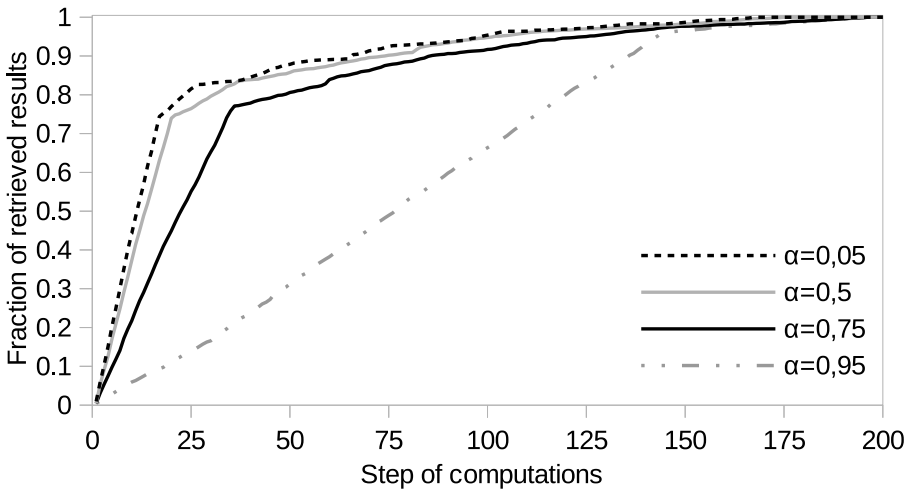
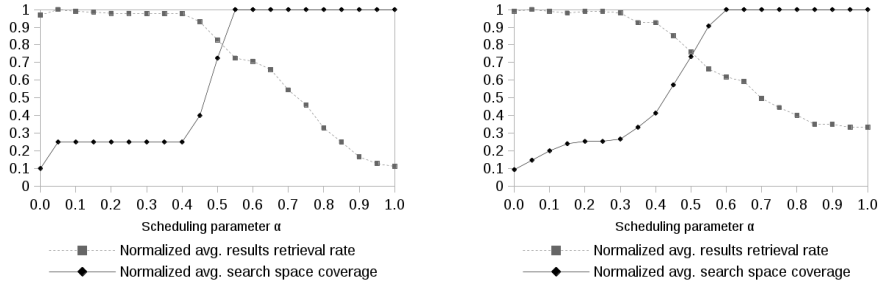


Fig. 2. Results retrieval progress in the course of the simulated virtual screening run over GDB-9 database

When developing schedulers for Desktop Grids, game-theoretical methods seem potentially efficient. One can note the work [17]. Its authors describe a game-theoretical approach to the balancing of volunteer computing resources between several projects. Each BOINC client has three parameters: peak performance, resource availability history, and settings of resource sharing between BOINC projects. Each project has its own strategy consisting of the policy of sending tasks, the slack (the value that determines the order of deadlines appointment to the tasks), the intervals between connections to the server, and the replication strategy. The project's payoff function is equal to the cluster equivalence parameter. Based on this model, the authors of the paper find the equilibrium for high-performance computing projects and high-throughput computing projects.

In work [18], co-authored by the authors of the presented paper, an hierarchical game-theoretic model of task scheduling is presented. The model is used to reduce the total server load by creating optimal-sized task parcels instead of sending tasks to clients one by one. Each client decides which size of parcels they will request. The payoff function of the server determines its costs for the formation of task parcels, the time of waiting for the computations results and their processing. The payoff function of the client is the cost of computing and communications. Taking into account the fact that an error in the execution of a single task leads to an error in the entire task parcel, the presented model allows to decrease the number of requests to the server due to the increase in the amount of overhead expenses. The resulting solution can be used in projects with "short" computational tasks; the solution was tested on a VS setting where one task corresponded to a single ligand. In practice, the grouping of ligands



(a) Steps 1–5 (b) Steps 21–25

Fig. 3. Results retrieval rate and search space coverage depending on the scheduling coefficient α (normalized, averaged over 5 steps)

is being used in the course of VS, but the optimal parcel size is determined heuristically for specific computing systems (see, for example, [19,20]).

Due to the demand of resources, VS essentially involves high-performance computing 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 [8]. Among them, 25 of 82 papers published in 2008–2015 explicitly state that VS had been performed using high-performance computing such as computing clusters, supercomputers and grids.

There are works describing fruitful VS using Desktop Grids, as well. 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 [21], which employed the power of over 3.4 million personal computers to support 27 research projects, including drug searches against AIDS, cancer, malaria and other diseases. To name a few, the projects allowed to discover candidate treatments for neuroblastoma [22], tuberculosis [23], leishmaniasis [24] etc.

Thus, high-performance and high-throughput computing tools have proven to be instrumental for implementing VS.

At the same time, there are efforts to automate new VS runs using different types of computational resources [25,26] and to optimize the VS process. For example, the task of fast retrieval of maximally diverse hits is being solved using genetic algorithms and heuristics [9,27].

In work [28], the authors solve the task of fast retrieval of the most prospective ligands on early stages of VS. The method they use is dividing the search space into classes according to molecular properties, and calculating the Bayesian probability of new ligands falling into one or another class. The search space is restricted by well-known Lipinski's rules, which, in general, do not guarantee coverage of all prospective ligands. However, such charting of the database al-

lows to prioritize the ligands apriori, and the experiments prove the efficiency of the approach.

The authors of [20] consider the overall process of VS and investigate its performance in terms of wall-clock time to obtain the results. They divide the database into chunks of ligands and evaluate the performance for different chunk sizes.

The mathematical model described in this paper has been proposed and elaborated in [7]. It was shown that the model allows to boost VS efficiency at early stages, probably at the sacrifice of its productivity at later stages. Its implementation for BOINC platform has been proposed in [5] as a pseudo code. In the present paper we investigate the ability of the model to adjust the results retrieval rate and search space coverage at any stage of VS.

6 Conclusions and future work

In this paper, we present an implementation of adaptive scheduling algorithm for virtual screening using BOINC-based Desktop Grid. It is based on the mathematical model of game theory, where task scheduling is considered as a congestion game with computing nodes as players, who choose specific subsets of data blocks for processing. We introduce a scheduling parameter α which expresses the balance between results retrieval rate and search space coverage. We conduct computational experiments and show that by varying α, one is able to set the desired balance value.

The computational experiments to evaluate the performance of the developed algorithm were performed in the Enterprise Desktop Grid based on resources of the Karelian Research Center, Russian Academy of Sciences.

The presented mathematical model and the scheduling algorithm are designed for BOINC-based Desktop Grid. Further study is required to investigate the impact that the overall search process experiences when individual players change their scheduling strategies according to their preferred behavior of either "digger" or "explorer". This is relevant for volunteer computing projects. This will be the subject of future work, as well as assessment of the algorithm performance and effectiveness in multi-objective domains.

Acknowledgements

This work was supported by the Russian Foundation of Basic Research, projects 18-07-00628 and 18-37-00094.

References

1. Afanasiev, A. P., Bychkov, I. V., Manzyuk, M. O., Posypkin, M. A., Semenov, A. A. and Zaikin, O. S.: Technology for integrating idle computing cluster resources into volunteer computing projects. In: 5th International Workshop on Computer Science and Engineering: Information Processing and Control Engineering, WCSE 2015-IPCE, pp. 109–114 (2015)

2. Kovács, J., Marosi, A., Visegrádi, Á., Farkas, Z., Kacsuk, P. and Lovas, R.: Boosting gLite with cloud augmented volunteer computing.Future Generation Computer Systems **43–44**, 12–23 (2015) https://doi.org/10.1016/j.future.2014.10.005
3. Choi, S., Kim, H., Byun, E., Baik, M., Kim, S., Park, C. and Hwang, C.: Characterizing and classifying desktop grid. In: Cluster Computing and the Grid. CCGRID 2007. Seventh IEEE International Symposium on, pp. 743–748. IEEE. (2007)
4. Anderson, D.P.: BOINC: A System for Public-Resource Computing and Storage. In: The Fifth IEEE/ACM International Workshop on Grid Computing (GRID'04), pp. 4–10. IEEE CS Press (2004)
5. Nikitina, N., Ivashko, E., Tchernykh, A.: Congestion Game Scheduling Implementation for High-Throughput Virtual Drug Screening Using BOINC-based Desktop Grid. In: Malyshkin, V. (ed.) Parallel Computing Technologies. PaCT 2017, LNCS, vol. 10421, pp. 480–491. Springer, Cham (2017) https://doi.org/10.1007/978-3-319-62932-2_46
6. Bielska, E. et al.: Virtual screening strategies in drug design — methods and applications. Journal of Biotechnology, Computational Biology and Bionanotechnology **92**(3), 249–264 (2011)
7. Nikitina, N., Ivashko, E., Tchernykh, A.: Congestion Game Scheduling for Virtual Drug Screening Optimization. Journal of Computer-Aided Molecular Design **32**(2), 363–374 (2018) https://doi.org/10.1007/s10822-017-0093-7
8. Irwin, J., Shoichet, B. K.: Docking screens for novel ligands conferring new biology. Journal of Medical Chemistry **59**(9), 4103–4120 (2016)
9. C. Rupakheti, A. Virshup, W. Yang, D. N. Beratan. Strategy to Discover Diverse Optimal Molecules in the Small Molecule Universe. Journal of Chemical Information and Modeling **55**(3), 529–537 (2015)
10. Harper, G., Pickett, S. D., Green, D. V.: Design of a compound screening collection for use in high throughput screening. Comb. Chem. High Throughput Screening **7**(1), 63–70 (2004)
11. Rosenthal, R.: A Class of Games Possessing Pure-Strategy Nash Equilibria. Int J Game Theory **2**(1), 65--67 (1973)
12. Milchtaich, I.: Congestion Games with Player-Specific Payoff Functions. Games Econ Behav. **13**, 111–124 (1996)
13. Ieong, S. et al.: Fast and Compact: A Simple Class of Congestion Games. In: AAAI'05 Proceedings of the 20th national conference on Artificial intelligence, Vol. 2, pp. 1--6. AAAI Press (2005)
14. Willet, P., Barnard, J. M., Downs, G. M.: Chemical Similarity Searching. Journal of chemical information and computer sciences **38**(6), 983–996 (1998)
15. Patterson, D. E., Cramer, R. D., Ferguson, A. M., Clark, R. D., Weinber, L. E.: Neighborhood behavior: a useful concept for validation of "molecular diversity" descriptors. Journal of Medicinal Chemistry **39**(16), 3049--3059 (1996)
16. Hann, M. M., Leach, A. R., Harper, G.: Molecular complexity and its impact on the probability of finding leads for drug discovery. Journal of Chemical Information and Computer Sciences **41**(3), 856--864 (2001)
17. Donassolo, B., Legrand, A., Geyer, C.: Non-cooperative scheduling considered harmful in collaborative volunteer computing environments. In: 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, pp. 144–153. (2011)
18. Mazalov, V., Nikitina, N., Ivashko, E.: Hierarchical Two-Level Game Model for Tasks Scheduling in a Desktop Grid. In: Applied Problems in Theory

of Probabilities and Mathematical Statistics Related to Modeling of Information Systems, 2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), pp. 641–645. (2014) https://doi.org/10.1109/ICUMT.2014.7002159
19. Jaghoori, M. et al.: A multi-infrastructure gateway for virtual drug screening. Concurrency Computat.: Pract. Exper. **27**, 4478–4490 (2015)
20. Krüger, J. et al.: Performance Studies on Distributed Virtual Screening. Hindawi Publishing Corporation, BioMed Research International **2014**, Article ID 624024, 1–7 (2014)
21. World Community Grid, https://www.worldcommunitygrid.org. Last accessed 12 Apr 2018
22. Nakamura, Y., Suganami, A., Fukuda, M., Hasan, Md. K., Yokochi, T., Takatori, A., Satoh, S., Hoshino, T., Tamura, Y., Nakagawara, A.: Identification of novel candidate compounds targeting TrkB to induce apoptosis in neuroblastoma. Cancer Medicine **3**(1), 25–35 (2014) https://doi.org/10.1002/cam4.175
23. Perryman, A. L., Yu, W., Wang, X., Ekins, S., Forli, S., Li, S.-G., Freundlich, J. S., Tonge, P. J., Olson, A. J.: A Virtual Screen Discovers Novel, Fragment-Sized Inhibitors of Mycobacterium tuberculosis InhA. Journal of Chemical Information and Modeling **55**(3) 645–659 (2015) https://doi.org/10.1021/ci500672v
24. Ochoa, R., Watowich, S.J., Flórez, A. et al.: Drug search for leishmaniasis: a virtual screening approach by grid computing. Journal of Computer-Aided Molecular Design **30**(7), 541–552 (2016) https://doi.org/10.1007/s10822-016-9921-4
25. Ellingson, S. R. and Baudry, J.: High-throughput virtual molecular docking with AutoDockCloud. Concurrency Computat.: Pract. Exper. **26**, 907–916 (2014) https://doi.org/10.1002/cpe.2926
26. Forli, S. et al.: Computational protein-ligand docking and virtual drug screening with the AutoDock suite. Nature Protocols **11**(5), 905–919 (2016)
27. Rupakheti, C. R.: Property Biased-Diversity Guided Explorations of Chemical Spaces. PhD dissertation, Duke University, 138 p. (2015)
28. Pradeep, P., Struble, C., Neumann, T., Sem D. S. and Merrill, S. J.: A Novel Scoring Based Distributed Protein Docking Application to Improve Enrichment. IEEE/ACM transactions on computational biology and bioinformatics **12**(6), 1464–1469 (2015)