Journal of
CHEMICAL INFORMATION
AND MODELING

Article

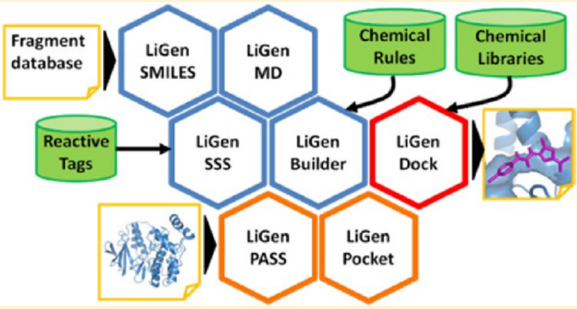
pubs.acs.org/jcim

LiGen: A High Performance Workflow for Chemistry Driven de Novo Design

Andrea R. Beccari,*[†,‡] Carlo Cavazzoni,[§] Claudia Beato,[‡] and Gabriele Costantino[‡]

[†]Dompé R&D Centre, Dompé SpA, Via Campo di Pile, 67100 L'Aquila, Italy
[‡]Dipartimento di Farmacia, Università degli Studi di Parma, Viale Area delle Scienze, 27/A, 43100 Parma, Italy
[§]CINECA, Via Manganelli 6/3, 40033 Casalecchio di Reno (BO), Italy

ABSTRACT: Tools for molecular de novo design are actively sought incorporating sets of chemical rules for fast and efficient identification of structurally new chemotypes endowed with a desired set of biological properties. In this paper, we present LiGen, a suite of programs which can be used sequentially or as stand-alone tools for specific purposes. In its standard application, LiGen modules are used to define input constraints, either structure-based, through active site identification, or ligand-based, through pharmacophore definition, to docking and to de novo generation. Alternatively, individual modules can be combined in a user-defined manner to generate project-centric workflows. Specific features of LiGen are the use of a pharmacophore-based docking procedure which allows flexible docking without conformer enumeration and accurate and flexible reactant mapping coupled with reactant tagging through substructure searching. The full description of LiGen functionalities is presented.



■ INTRODUCTION

Molecular de novo design is aimed at generating novel chemotypes endowed with a particular set of desired properties, in most cases pharmacological properties.[1] The de novo design process is ideally driven by knowledge, and this knowledge is used to define constraints which the novel molecular structures being generated are supposed to obey. As extensively discussed by Schneider and Fechner,[1] the search for novel chemotypes is a local and a multiobjective optimization process,[2−4] which can lead to several solutions according to the input constraints and to the generation rules chosen by the chemist. On the other hand, finding a globally optimal solution is unworkable given the enormous size of the available chemical space.[5] Therefore, computer programs have been actively sought during the past two decades to help (medicinal) chemists to find the best local solutions in terms of pharmacological properties, chemical feasibility, innovation, and patentability. As a result, several molecular de novo programs, such as LUDI,[6] LEGEND,[7] LeapFrog,[8] LigBuilder 2,[9] SPROUT,[10,11] HOOK,[12] PRO-LIGANDS,[13−15] and DOGS[16] have been developed and thorough reviews of the de novo programs are available.[1,17−19] Despite several successful applications having been reported,[20] many problems eventually prevented molecular de novo design approaches to become established tools in drug design, and, in fact, methods like virtual screening and molecular docking received higher attention, either in terms of successful applications or widespread use. The most common problems associated with these early de novo design methods are the following: (i) producing chemically invalid structures or structures that do not have drug-like properties; (ii) poor synthetic accessibility of the suggested ligand;[21] (iii) low structural diversity;[20] (iv) low potential for parallel synthesis applications (a part when combinatorial chemistry is directly addressed[22]); (v) generally low throughput if compared to docking programs.

The problem of synthetic feasibility, in particular, is a most relevant one, and its impact on the outcome of the de novo approach is highly dependent on the stage of the project and on the size of the library which has to be assembled. When the main goal is a primary screening setup, many docking as well as de novo design programs are able to handle combinatorial explosions. Under these conditions, the definition of a set of fragments (reagents) with simple anchor points, and the availability of a predefined set of reaction steps can lead to the generation of targeted libraries of arbitrary size which can be in principle synthesized in parallel. In this context, the ability of controlling chemistry, in terms of reaction steps and reagent's availability, is crucial to move the designed libraries from virtual to real.

Another common application of de novo design is lead optimization. In this case, the binding mode of the core structure of the lead has usually already been validated and the scope of the de novo approach is to optimize decoration toward increased affinity and/or improved ADME properties. The value of the class under study is therefore high, and the number of compounds to generate is reasonably low. Thus, the

Received: February 1, 2013

selection of the fragments is driven by the optimization process and not by the accessibility of the presynthesized reagents.

Generally used also for scaffold hopping, Ligbuilder[9] and FlexX[23,24] in combination with ReCore[25] are able to generate new scaffolds for the same pattern of substituents.

A relevant application is the design of ligand-based libraries without a predefined set of scaffolds. In this case, it is imperative to extensively map the available chemical space by defining a wide set of chemical rules.

Two approaches try to address the synthetic feasibility problem: the first one is to use a retrosynthetic analysis of the final ligand, and this is generally done by the implementation of bond breaking rules. LigBuilder2[9] is an example which uses an internal database of building blocks, while it is also possible to use standalone programs, like for example SYLVIA,[26] developed by Molecular Networks. The second approach, used by most of the available de novo design programs uses restricted sets of chemical rules which encode for the most common chemical reactions. Some of the most relevant reaction driven de novo software programs with an extended set of chemical reactions are SYNOPSIS[27] and DOGS.[16]

In this scenario, we decided to develop a de novo design program able to employ a user-defined set of true chemical reactions as connection rules and to use tangible commercial reagents as building blocks. Ligand Generator (LiGen), described in this paper, is a suite of programs able to define flexible de novo design workflows by using both pharmacophores and molecular docking engines, linking the fragments in situ, and directly scoring the drug likeness of the generated molecules.

LiGen consists of a set of tools which can be combined in a user-defined manner to generate projectcentric workflows. In a standard application, the modules work sequentially, from the generation of the input constraints (either structure-based, through active site identification, or ligand-based, through pharmacophore definition), to docking and de novo generation. Alternatively, the modules can be used as a standalone service according to the user's need. Figure 1 illustrates the general LiGen workflow.

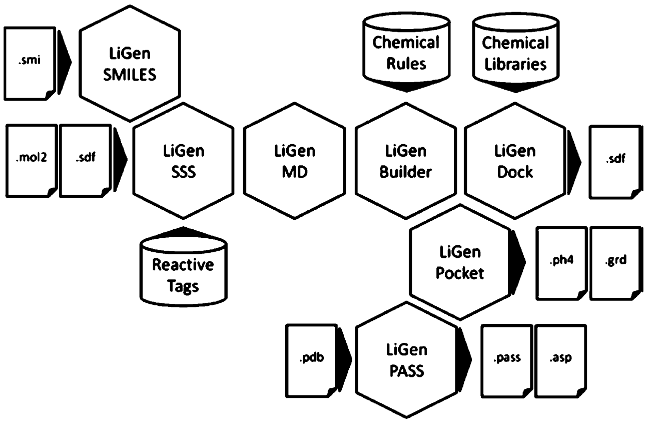


Figure 1. LiGen modules and data flow are represented. Input data of the overall process are the following: SMILES strings (representing fragments) from a database; the target protein(s); user queries; and chemical rules. Given these input data, the output of LiGen are novel compounds docked into the target protein. The description of the functionalities of each module are given in the Experimental Section.

The structure and the function of individual modules are described in details in the following paragraphs, along with a general description of the philosophy and of the functionalities and of the software.

GENERAL FEATURES OF LIGEN

As mentioned above, the LiGen suite has been designed to be, at the same time, a modular de novo design application and a computational tool box. The LiGen modules can be used independently from each other and combined with other tools typically used in drug design research.

This is made possible by the fact that modules communicate each other by using a few standard data formats, with a few minor extensions for intermodules communication not breaking the original standard format. Moreover the modular structure of LiGen allows the implementation of different workflows using the basic building block tools, depending on the specific computation or analysis required.

LiGen is written in C++, with the use STL (standard template library) at a high level, and its own algorithms (*functions*) and data structures (*classes*) at a low level. In this way the code is flexible, easily extensible, and fast on many architectures. Performance portability to different hardware has been a main priority in LiGen design. Using STL at all levels (e.g., basic vector for interatomic distances) may end-up with a code not equally efficient on all hardware or possibly not at all portable (think to GPU computing where CUDA or OpenCL kernels are required, and STL may not be fully supported). All LiGen modules share the same basic C++ classes representing the main biochemical objects—atoms, residues, molecules—or mathematical objects—vector, grid, ring, etc. LiGen modules can be compiled in standalone executable or built together in a single executable implementing a given workflow. The basic modules of LiGen are the following: LiGenPass, LiGenPocket, LiGenSmiles, LiGenMD, LiGenMinimizer, LiGenSSS, LiGenDock, LiGenScore, and LiGenBuilder.

Definition of the Input Constraints. The definition of the input constraints required to run a de novo design, given a biological target, is particularly relevant as it is directly connected to the quality evaluation of the candidate compounds which will eventually result from the run. In general, target constraints for the generated compounds are composed by a set of information related to ligand−receptor interactions. This information may equally derive from the known 3D structure of the ligand−receptor complex or from the structure of a set of ligands for that particular target. In the common case of receptor based approaches, input constraints are usually given in the form of shape constraints, derived from complementarity between ligand and binding pocket, and in the form of potential interaction points, usually classified as hydrogen bonds and electrostatic and hydrophobic interactions. The thus extracted constraints are used to score the candidate compounds. Scoring is clearly crucial for the design process, as it allows for a ranking of the candidate structures and for the selection of the most promising ones, and it also may guide the search through the available space by defining fitness values to the sampled space. Scoring can be achieved either by explicit force field methods, empiric scoring functions, or knowledge-based scoring functions, being explicit force fields methods usually more computationally expansive than the other two.

The way in which LiGen treats the definition of the input constraints, the docking, and the subsequent scoring of the

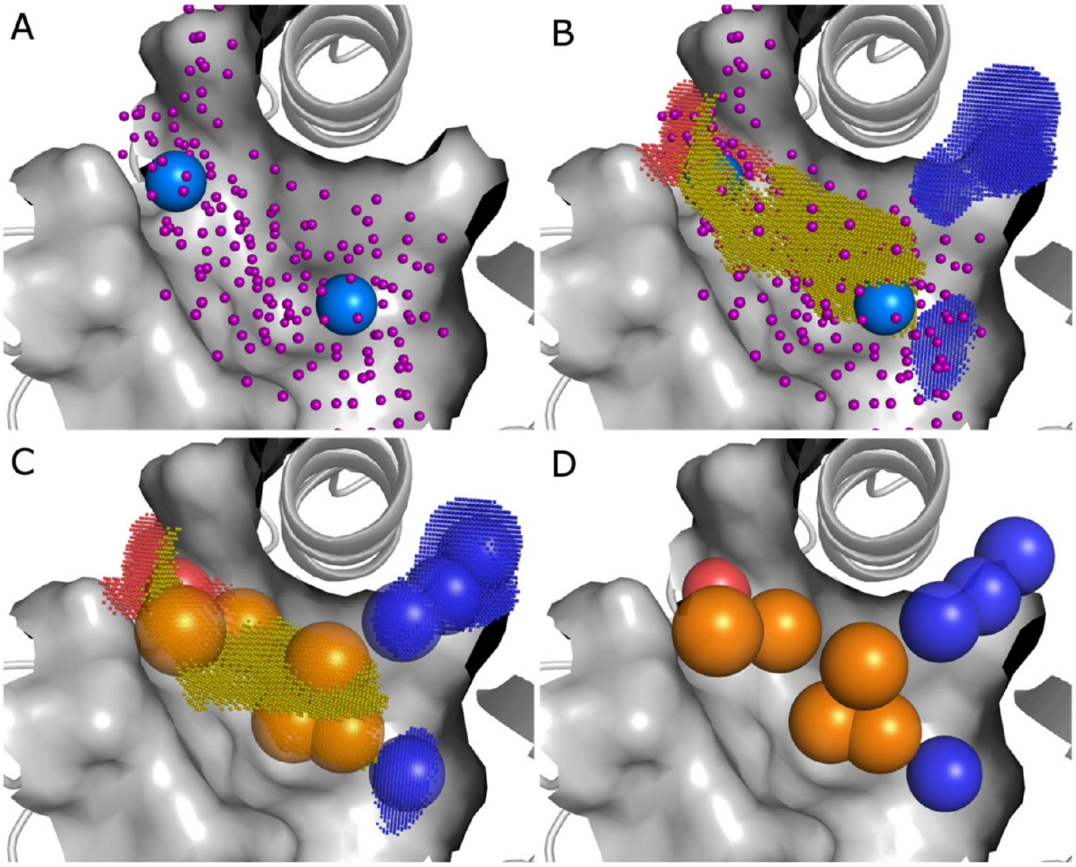


Figure 2. Identification of protein binding sites. (A) LiGenPass maps the protein surface filling cavities with probes (violet); probes are clustered in ASP, representing the center of mass of the cavity. (B) According with the tridimensional coordinates of ASP, the cavity surface is mapped in order to identify hydrophobic regions (yellow), H-bind donor (blue), and H-bond acceptor (red) regions (key sites). (C) Key sites just identified are clustered into a pharmacophore model (in orange are represented the hydrophobic features, in blue the H-bond donor features, in red the H-bond acceptor feature). (D) Pharmacophore model of the binding site. Protein PDB code: 1dfh. Figures are prepared with Pymol.[37]

candidate structures is exemplified by the diagram reported in Figure 1.

LiGenPass is based upon the algorithm used by PASS (putative active site with spheres), software developed by Brady et al.[28] to identify cavities in the target proteins. As described in the Experimental Section, LiGenPass is a new implementation of the algorithm, and it is used by the LiGen engine to define binding sites for the target protein. LiGenPass characterizes regions of buried volume in the target protein and identifies potential binding sites based upon the size, shape, and burial extent of these volumes (Figure 2A).

Each cavity is defined by an Active Site Point (ASP) as the center of mass of the cavity itself. The number of possible ASPs can be defined by the user as the threshold distance at which they are considered distinct units. Once identified a number of cavities through the corresponding ASPs, the user may want to characterize them in terms of key interaction points and suggest a pharmacophore which can subsequently be used for to 3D database searching, docking procedure or de novo ligand computation. This task is operated by the LiGenPocket module. In essence, LiGenPocket works by creating a regularly spaced grid around ASPs, if the target protein has no cocrystallized ligands, or by defining a sphere which covers the ligand and the protein atoms surrounding the ligand, and then creating a grid within the sphere, if there is a cocrystallized ligand. Then, the program places a hydrogen atom as a probe on each grid point to check its accessibility. If the probe bumps into the protein, that grid point will be labeled as "not free". A bump is counted when the interatomic distance is less than the sum of van deer Waals radii reduced by 0.5 Å. If the probe does not bump into the protein, that grid point will be labeled as "free". If a grid point is farther than 5 Å from any atom of the protein, it will be labeled as "outside" rather than "free". The assembly of free grid points forms the body of the binding pocket in which each newly designed ligand will be built up. As the next step, the program will derive key interaction sites within the binding pocket. Such information is necessary for the subsequent ligand construction or docking process. In the current implementation, the program uses three different types of probe atoms, based on the Tripos Force Field, to screen the binding pocket: a positively charged sp3 nitrogen atom (ammonium cation), representing a hydrogen bond donor; a negatively charged sp2 oxygen atom (as in a carboxyl group), representing a hydrogen bond acceptor; a sp3 carbon atom (methane), representing a hydrophobic group. In our hands, the choice of this triplet of atom types yielded the best results, but this can be changed by the user and additional atom types can be included. For each free grid point, the binding energies between the probes and the protein are calculated by using an in house developed scoring function based on the paper by Wang et al.[29] Each grid point will be labeled as donor, acceptor, or hydrophobic according to the highest score achieved by one of the three probes on that particular point. The program will then filter all the grid points to derive the key interaction sites in a two-step process. In the first step, the program calculates the average score for all the grid points labeled as "donor". Then, the program identifies grid points having a score lower than the average and labels them back as free. The same process is repeated for the "acceptor" grid points and for the "hydrophobic" grid points (Figure 2B). At this stage, LiGenPocket attempts a definition of pharmacophoric points. Each survived grid points, either donor, acceptor, or hydrophobic, is characterized for the number of "neighbors" grid points. In this context, neighbors are defined as grid points with the same definition (donor, acceptor, or hydrophobic), falling less than 2 Å from that particular point. The average of neighbors for each type of grid points is calculated, and those points having a number of neighbors lower than the average are labeled as free. After this step, only those grid points that aggregate survive and can be defined as the key interaction sites within the binding pocket. The geometric center of each aggregation is calculated and finally defined as a pharmacophoric point (donor, acceptor, or hydrophobic; Figure 2C).

This binding site-derived pharmacophore model can then be used for the subsequent docking of candidate compounds, or directly as a query structure to perform 3D database searching, which provides an additional way to find novel ligand molecules that fit to the target protein.

Docking Procedures. In a standard application of a de novo design approach, placement of fragments into the target protein according to a set of input constraints usually follows the characterization of the binding pocket(s). LiGen accomplishes this task by using the module LiGenDock, which can also be used as a standalone module to perform classical docking experiments. The main feature of LiGenDock is the use of the pharmacophore scheme generated by LiGenPocket as the driver for the docking procedure, including a nonenumerative flexible docking algorithm (Figure 3).

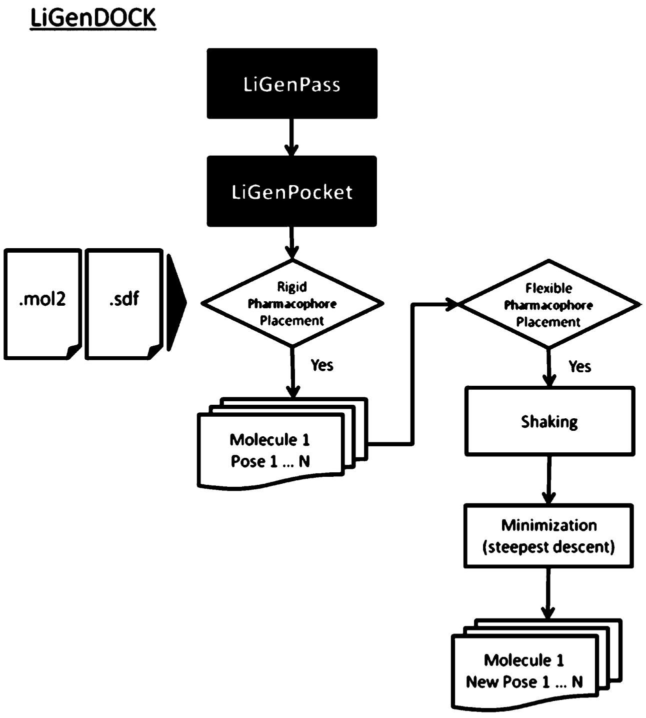

LiGenDOCK
LiGenPass
LiGenPocket
.mol2
.sdf
Rigid Pharmacophore Placement
Yes
Molecule 1 Pose 1 ... N
Flexible Pharmacophore Placement
Yes
Shaking
Minimization (steepest descent)
Molecule 1 New Pose 1 ... N

Figure 3. LiGenDock module. Black boxes represents modules before LiGenDock.

A thurough description of LiGenDock, along with its full validation and with the optimization of the docking parameters, is presented in the accompanying paper.[30] Here, we discuss the most relevant features. In essence, LiGenDock first computes all possible combination pairs between the pharmacophoric points defined by LiGenPocket and the interacting atoms on the ligand. Then, LiGenDock translates the ligand in such a way that its interacting atoms coincide with the pharmacophore counterparts. After that, LiGenDock tries to match a second pair by rotating the ligand around the first anchor point. The rotation angle is chosen in such a way that the second ligand atom lay in a line connecting the first pair and the second pharmacophoric point. All the pairs laying within a user defined distance cutoff are taken into account to establish if a match exists or not. If this second ligand atom is hydrophobic, the pair is taken into account only if the hydrophobicity of the pharmacophore feature is larger than a threshold value, which again can be specified by the user. At this point, a conformer search is performed, allowing all torsional angles to rotate freely. The conformers with the best scores are retained for the successive step. Finally, LiGenDock loops over all ligand poses that match two pairs of pharmacophore feature–ligand atom, trying to match a third pair by rotating the ligand around the axis connecting the two pairs already found. At the end LiGenDock exits the procedure by keeping the best poses of the ligand which are then passed to an energy minimization routine used to refine the position of the ligand inside the binding pocket (Figure 4).

Two algorithms for the energy minimization have been implemented in the module LiGenMinimizer. The first one is a full featured rigid body energy minimization (shacking). It uses spring-elastic constraints (defined by a force constant that is a function of the interaction energy of the original grid point) between the pharmacophore points and the ligand atoms. In principle the system should evolve to accommodate as many pharmacophore–ligand pairs as possible, to lower the free energy. The second step of energy minimization is a steepest descent minimization, where the algorithm is directed by the scoring-function instead of the force-field potential. Instead of a real minimization process, in this case the position of the molecules are changed by a discrete value of 0.25 Å in seven directions of the tridimensional space (three axes and four quadrant bisectors) and then the direction of diminishing score energy is taken.

Structure Generation and Synthetic Accessibility. Several strategies can be adopted to sample and generate structures by de novo approaches. They are usually referred to as linking,[6,9] growing,[31] lattice-based sampling, random structure mutation, transitions driven by molecular dynamics simulations, and graph-based sampling,[1] the former two having received particular attention. The linking approach starts with fragments positioned or predocked at different interaction sites of the binding pocket. Then, the fragments are linked together to give a molecule able to satisfy to the greatest extent all the interaction points. Also the growing approach starts with a positioned fragment which is grown to maximize favorable interactions with the key interaction sites or with receptor regions in between key interaction sites.

The basic building blocks for the assembly of candidate structures can be either single atoms or fragments. Atom-based approaches are superior to fragment-based methods in terms of the structural variety that can be generated, but this increase in potential solutions makes it harder to find feasible and

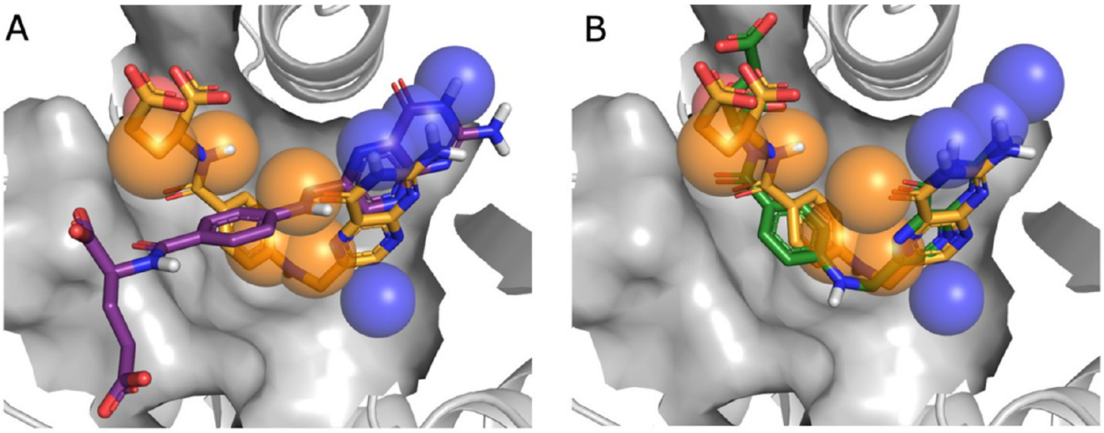


Figure 4. Exemplification of the ligand conformation generation inside the binding pocket. (A) Starting conformation for a given ligand is represented in violet and its optimized conformation in bright orange. To produce the flexible pose, once two pharmacophore features are matched, the dihedral angles are allowed to rotate to match the other pharmacophore features. (B) Optimized pose superposed to the crystallographic ligand conformation (green). Figures are prepared with Pymol.[37]

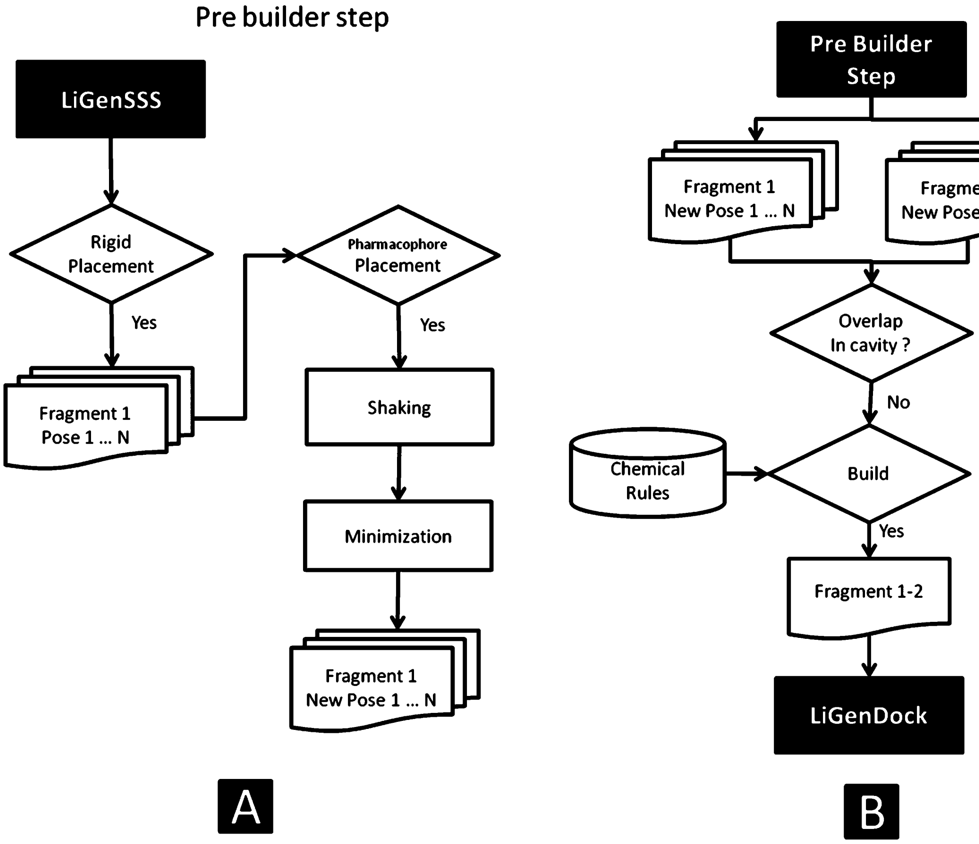


Figure 5. de Novo design process of LiGen. Black boxes represents modules before LiGenBuilder.

druggable candidate compounds among the ones that have been identified. LiGen implements the structure building in a way to effectively ensure synthetic feasibility of the candidate compounds. The building model of LiGen, called LiGenBuilder, is at the heart of LiGen and is the main engine implementing our denovo algorithm. In particular, LiGenBuilder is combined with the docking (LiGenDock) and the force field engine (LiGenMD) modules to cooperate in order to perform in situ de novo growth of novel ligands. The building algorithm implemented in LiGenBuilder is described in detail in the next paragraph.

de Novo Design. In Figure 5 the overall process of ligand growth is presented.

The process starts with a prebuilding step (Figure 5A) where a set of fragments is docked into the target active site and a number of poses that match given criteria are retained into a set which will be used as a basis for the successive growth process. In detail, seed building blocks are assigned force field parameters (MMFF94[32−36]) and minimized (geometry optimization using MMFF94 force field). The building block atoms are also tagged for all possible reactions defined in a *chemical rules database*. Seed building blocks are then passed through

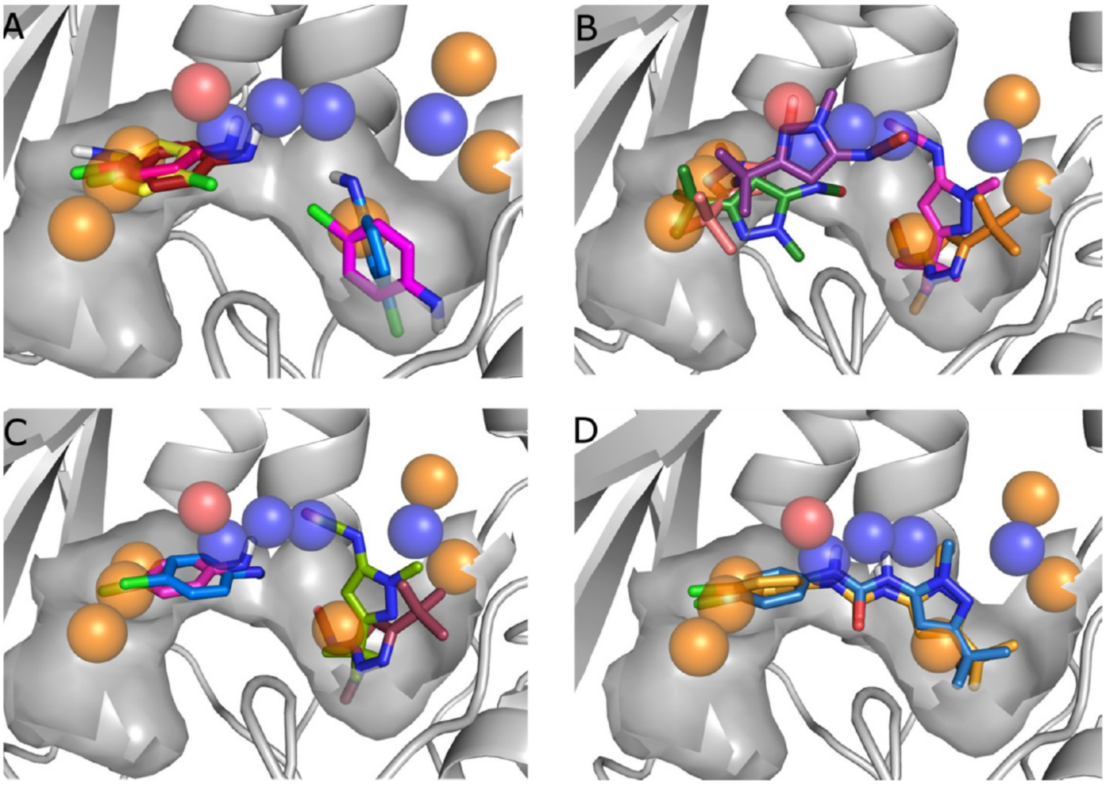
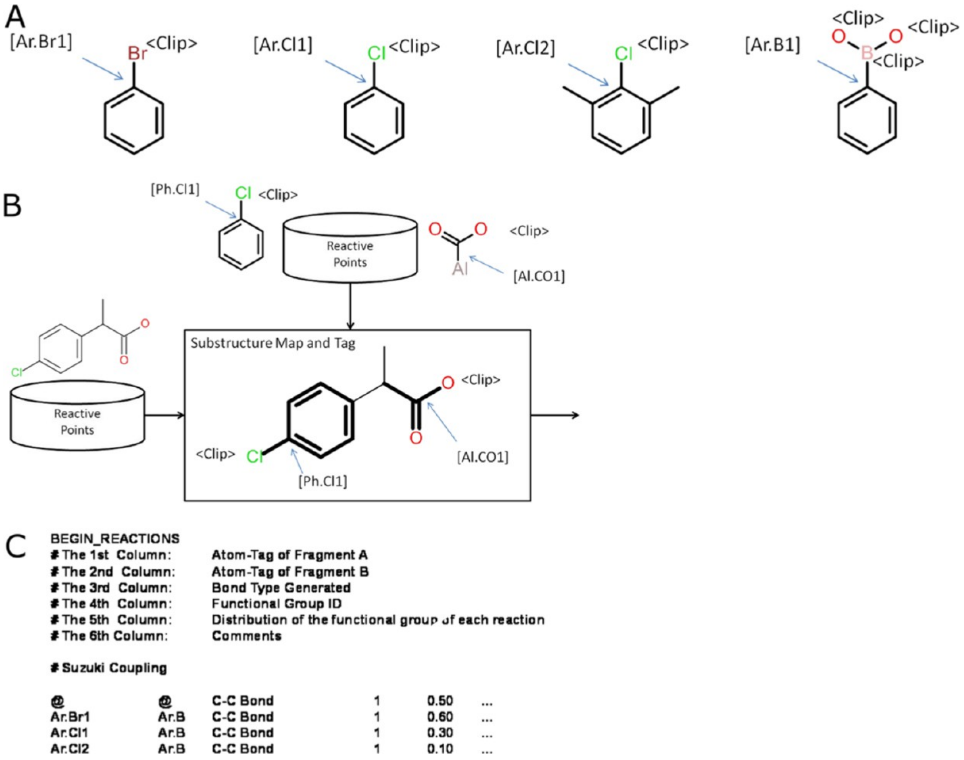


Figure 6. Example of ligand growing process. (A and B) Placement of the fragments inside the binding site in order to match the features of the pharmacophore. (C) Different fragments shown together in the binding site, matching different pharmacophore features. (D) Final generated ligand is minimized inside the binding site in order to catch the most favorable interactions (in orange is represented the generated ligand, in blue the cocrystallize ligand pose). PDBcode of the example complex: 1kv1. Figures are prepared with Pymol.[37]



```
BEGIN_REACTIONS
# The 1st  Column:        Atom-Tag of Fragment A
# The 2nd  Column:        Atom-Tag of Fragment B
# The 3rd  Column:        Bond Type Generated
# The 4th  Column:        Functional Group ID
# The 5th  Column:        Distribution of the functional group of each reaction
# The 6th Column:         Comments

# Suzuki Coupling

@              @        C-C Bond         1      0.50    ...
Ar.Br1         Ar.B     C-C Bond         1      0.60    ...
Ar.Cl1         Ar.B     C-C Bond         1      0.30    ...
Ar.Cl2         Ar.B     C-C Bond         1      0.10    ...
```

Figure 7. (A) Examples of tagged chemical structures. (B) Example of how LiGenSSS works. It transfers tags from the reactive point database to the fragment database used in the de novo design process. (C) Example of a reaction file (a Suzuki coupling reaction).

three subphases. The first one is rigid docking, where the fragments are placed inside the pocket using the scoring function and the best poses kept in the seed set of poses; then the torsional angle of the poses are allowed to move so that the

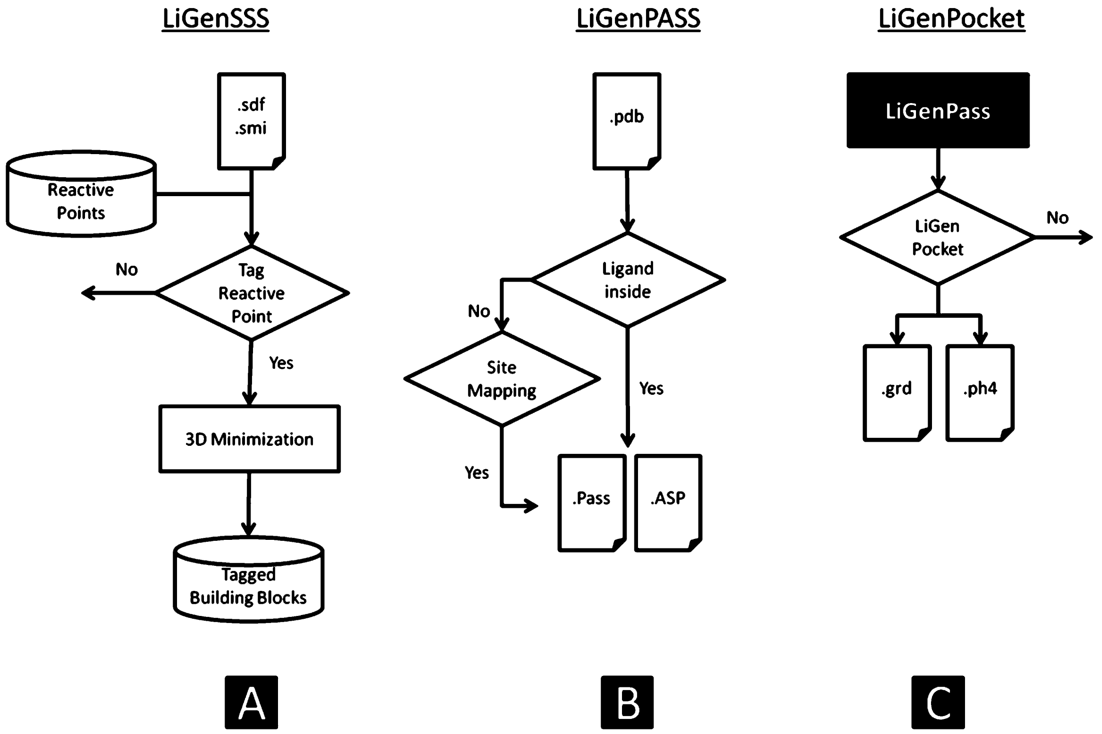


Figure 8. Flowchart of LiGenSSS, LiGenPass, and LiGenPocket modules. The black box represents module before LiGenPocket.

poses can be further optimized to lower the binding energy with the target protein; finally the complexes between poses and the target protein are minimized to find a local energy minimum. A pose is kept inside the set of best poses if the difference between internal energy before and after the minimization inside the pocket is less than a given threshold (defined by the user).

After this starting set has been completed, the process continues with a sequence of building steps (Figure 5B) where the poses contained in the starting set are grown using building blocks selected randomly (according to a probability function that depend on the chemical rules) among all the fragments that can react with the pose.

The new fragments thus formed are then docked again inside the protein pocket and the growing cycle continues. The process ends when either all the possible combinations of fragments have been explored (very rare event) or the molecular weight of the candidate ligand (pose) is larger than a user-defined value (Figure 6).

Chemical Rule Database. To perform the linking of two fragments or building blocks, a database of the most frequently established chemical reactions in parallel and combinatorial chemistry is used. The database has been encoded using the MDL mol format tagging the atoms with labels.

Two kinds of labels have been implemented: the part of the molecule that will be not present in the product is tagged with "<Clip>" and the reactive point for the reaction is labeled with an atom environment description tag. For example a reactive chlorine on a phenyl ring is tagged as "<Ph.Cl1>". Other examples of tagged structures are presented in Figure 7A

The key step of the de novo design process is now the transfer of the set of reactive points to the set of fragments or building blocks. Differently from other software like LigBuilder,[9] LiGen does not need a pretagged database of fragments. On the contrary, an arbitrary database of fragments can be used without manipulation in the de novo process. This feature is provided by the LiGenSSS module (Figure 7B).

LiGenSSS carries out a substructure search in the database of fragments and identifies the reactive part of the building block. Once a match between a fragment substructure and a reactive point is found, LiGenSSS transfers the tag(s) from the reactive point database to the matched structure in the fragment database. As a result, a database of tagged reagents (building blocks) is available.

The assembly of reactants is carried out, in a stepwise process, by LiGenBuilder. LiGenBuilder reads a text file where a simple linear notation for all the possible defined chemical reactions is encoded. Figure 7C shows the part of this reaction file for a Suzuki coupling reaction.

The user can edit the file to encode pairs of reacting groups. It is also possible to control the distribution of the chemical reactions in the library.

This can be done by the user by modifying column five of the reaction file (See Figure 7C). Column five controls the probability that a given reaction will be cast by the LiGenBuilder grow engine. According to the example given in Figure 7C, a 50% of probability to be selected has been given to Suzuki coupling reaction. Furthermore, among all the possible Suzuki couplings, the aryl-bromine derivative has been assigned a 60% chance to be selected versus the other two reactants. On the contrary, the sterically hindered chlorine derivatives [Ar.Cl2], with a 10% of probability, will be used only if reagents with a higher reactivity are not available. The distribution of the functional groups in the Reaction File is therefore chosen by the user according to his/her knowledge of the underlying chemistry. Here, the objective is not to cover all the possible building blocks within a particular reaction class (the Suzuki's coupling in our example), but rather, to select and give them a probability weight according to their synthetic accessibility not only in terms of general chemical feasibility but also in terms of specific know-how owned by that specific user or laboratory.

EXPERIMENTAL SECTION

Module Description. *LiGenPass.* The LiGenPass (LiGen putative active site with sphere) module is a simple computational tool that uses geometry to characterize regions of buried volume in proteins and to identify positions likely to represent binding sites based upon the size, shape, and burial extent of these volumes.

LiGenPocket. The LiGenPocket module takes a list of active sites and a target molecule, computes the volume, shape, and physicochemical characteristics (donor, acceptor, hydrophobic, etc.), and proposes a pharmacophore related to each active site given in the list. The volume and shape of the pocket is computed by using a regular Cartesian grid (the grid spacing is user-defined), and the pocket physicochemical characterization and the pharmacophore schemes are computed by using a in house derived version of XSCORE.[29] The protein atoms are characterized using TRIPOS force field.

LiGenSSS. LiGenSSS is a substructure search tool, used to retrieve a database of a flat file (.sdf or mol2) of all the molecules containing a given substructure. LiGenSSS carries out a substructure search in the database of fragments and identifies the reactive part of the building block. Once a match between a fragment substructure and a reactive point is found, LiGenSSS transfers the tag(s) from the reactive point database to the matched structure in the fragment database. As a result, a database of tagged reagents (building blocks) is available.

LiGenSmiles. LiGenSmiles is a simple parser used to convert smile string to 3D molecular structure. LiGenSmiles uses MMFF94 force field parameters to arrange atoms locally (distances and bond angle). LiGenSmiles is used to interface the fragment database (containing smile strings) to the LiGen suite.

LiGenMD. LiGenMD is a module performing molecular dynamics and energy minimization of small molecules by using MMFF94 force field. It is used in combination with LiGenSmiles to refine 3D structures obtained from smile string and to refine ligands geometry built inside protein pocket by LiGenBuilder.

LiGenScore. LiGenScore is the module implementing a consensus scoring algorithm similar to those described in the paper of Wang et al.[29] by using basic parameters from TRIPOS force field.

LiGenDock. LiGenDock is a docking module using LiGenScore to compute the scoring function and the LiGenPass and LiGenPocket modules to obtain the 3D structure of the binding site. The docking algorithm works as follows: (a) Take one or more ligands. (b) For each ligand define a pool of best poses. (c) Start matching a ligand's feature (i.e., a hydrogen bond donor site), with all the previously identified pharmacophore features. (d) Rotate the docked ligand of an appropriate solid angle to match a second pharmacophore feature with a second ligand's feature (or to make them as close as possible). (e) Rotate the ligand around the axis an appropriate angle passing between the two pharmacophore features and try to match a third feature. (f) Unlock the torsional angle and build conformers in situ trying to match as many features as possible (some angle may be selectively locked by the user, i.e. amide bond). (g) At every step the score is compared with the scores previously stored in a list. If this actual score is better than the stored one with the lowest score, the new pose is retained and the pose with the lowest score is removed. (h) Some constraints are implemented: one may require that poses contained in the best poses list should not occupy the same position in space, so that the best pose list will fill the entire protein pocket. (i) Finally the score is optimized with a simple score minimization algorithm that treats the docket ligand as a rigid body inside the pocket. The ligand is displaced and rotated in space around the docking pose until a minimum score is reached. Usually a steepest descent minimization algorithm gives satisfactory results, but here, one can adopt also a simulated annealing or a simple metropolis algorithm. Experience shows that it is not worth using sophisticated algorithms; otherwise, one ends up with "measuring" the scoring function rather than obtain a chemically acceptable result. It is by far more profitable to leave a simple algorithm and work to optimize scoring parameters.

LiGenBuilder. LiGenBuilder is the structure generator module of LiGen. LiGenBuilder first places fragments into the active site. Atoms potentially involved in structure generation are tagged and a set of chemical rules is applied to link fragments together. LiGenBuilder does not need a library of pretagged fragments. Rather, any arbitrary database of fragments can be used without manipulation in the de novo process

Parallel Computing. The basic algorithm of HTVS implemented in LiGen is the computation of a score between a target protein and a geometrical configuration of a trial ligand. Considering that normally HTVS deal with multiple target proteins, a large database of fragments to be combined into ligands and a considerable amount of possible ligand geometries, both internal atomic positions (conformers) and different docking configuration (poses), HTVS results particularly suited for massively parallel computing.

Architectures: windows, linux

Parallelization: MPI for more traditional HPC architectures, or bash scripting for cloud environment.

Accelerators/Coprocessors: when available the interatomic distances can be computed offloading the computation to other processors such as GPUs.

Xeon Phi

In agreement with Intel, the code has been part of the beta testing program for new Intel Many Integrated Core architectures, first on code named Knight Ferry processor and then on the code named Knight Corner processor (now commercialized with Xeon Phi). The results of experimental porting to this new kind of processor have shown that they represent an extremely promising platform for HTVS application.

DISCUSSION

Despite their theoretical very high relevance, de novo design methods did not become established tools in drug design because of the presence of several drawbacks that limited their application to real world cases. LiGen, the suite of programs described in this paper, aims at addressing most of the issues associated with available de novo design approaches. Key features of LiGen are the following: (i) Pharmacophoric constraints implemented in the pose identification. This allows obtaining only the poses endowed with the correct pattern of interactions with the binding pocket and, therefore, to significantly reduce the time for filtering them out once generated, as it is done by most of the docking programs and the time needed for visual inspection. (ii) Flexible docking

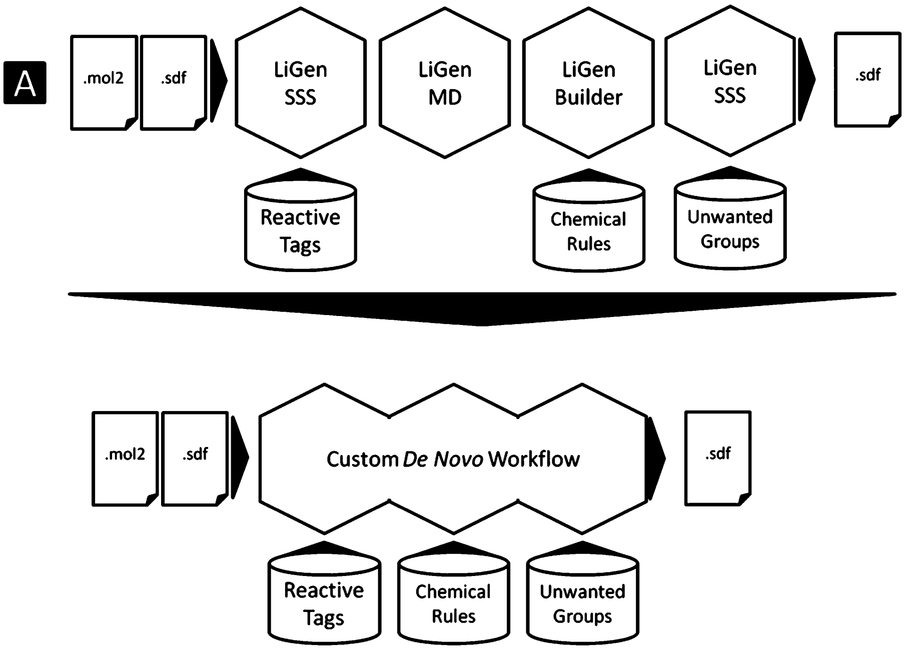


Figure 9. Example of a user-defined LiGen workflow. In this example the workflow was composed to enumerate a de novo library according to user-defined chemical rules and to exclude from this library unwanted combination of building blocks (i.e., possible new molecules) running again, after LiGenBuilder, the LiGenSSS module.

without conformer enumeration. Even for molecules with a few rotatable bonds, the coverage of the conformational space accomplished by enumerative methods, is only partial. Thus, docking methods based on conformational sampling are biased by the degree of coverage of the conformational space. Flexible alignment of the molecule on the pharmacophore points eliminates this bias and produces the bioactive conformation of the molecule, as long as the interaction points are correctly defined. If energetically allowed, this conformation is retained. If not, the molecule is discarded as unable to match the possible pharmacophoric mappings. This minimizes the number of molecular conformations which are evaluated for each ligand. A detailed description and optimization of the docking routines is presented in an accompanying paper. (iii) Accurate and flexible reactant mapping. Although many excellent de novo design software programs effectively handle chemical rules (for example DOGS[16]), LiGen steps forward by allowing the user to prioritize the chemical reactions through the definition of a probability of a reaction class (in the example, Suzuki coupling has a 50% of chance to be selected among all other reaction) and the definition of a probability for the reactive groups, meaning that the most reactive one will have more chance to be selected by the system to be incorporated in the final molecule. The user-defined choice of chemical reactions to be considered can in principle limit the scope of the being created de novo library, since particular reactions can simply be not considered. On the other hands, this fine-tuning allows the user to get results which are manageable by his/her lab in terms of synthetic accessibility and yields. (iv) Reactant tagging through substructure searching. In this way, the library of building blocks, fragments, or reagents does not need to be manually tagged. Once the reactive chemical points are defined, a substructure search routine automatically transfers the tag(s) to the fragment library. Also the identification of novel reactive points of the definition of subgroups is straightforward. (v) Modular architecture. Although LiGen was intended to be a suite of programs finalized to de novo design, it can also be used in a modular fashion, thus allowing each module to be used as a standalone tool, or to combine different modules to assemble task-specific workflows. For example, it is possible to define a workflow to just enumerate a de novo library according to defined chemical rules and use again the LiGenSSS to filter out unwanted combination of building blocks (Figure 9).

Interestingly, the LiGen source code has been written in such a way that a given workflow can be composed either by a pipe of different executables or can be compiled in a single embedded binary executable (Figure 9).

In conclusion we presented a new method for de novo design based on pharmacophore-driven docking and accurate chemical reaction mapping and implemented to be suitable for HPC applications and portable to many hardware architectures.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: a.beccari@dompe.it.

Notes

The authors declare no competing financial interest.

■ ABBREVIATIONS

STL, standard template library; GPU, graphics processing unit; CUDA, compute unified device architecture; OpenCL, open computing language; PASS, putative active site search; ASP, active site point

■ REFERENCES

(1) Schneider, G.; Fechner, U. Computer-based de novo design of drug-like molecules. *Nat. Rev. Drug Discovery* **2005**, *4*, 649−663.

(2) Nicolaou, C. A.; Apostolakis, J.; Pattichis, C. S. De novo drug design using multiobjective evolutionary graphs. *J. Chem. Inf. Model.* **2009**, *49*, 295−307.

(3) Nicolaou, C. A.; Kannas, C.; Loizidou, E. Multi-objective optimization methods in de novo drug design. *Mini. Rev. Med. Chem* **2012**, *12*, 979−87.

(4) Dey, F.; Caflisch, A. Fragment-based de novo ligand design by multiobjective evolutionary optimization. *J. Chem. Inf. Model.* **2008**, *48*, 679−690.

(5) Ruddigkeit, L.; van Deursen, R.; Blum, L. C.; Reymond, J. L. Enumeration of 166 Billion Organic Small Molecules in the Chemical Universe Database GDB-17. *J. Chem. Inf. Model.* **2012**, *52*, 2864−75.

(6) Bohm, H. J. The computer program LUDI: a new method for the de novo design of enzyme inhibitors. *J. Comput. Aided Mol. Des.* **1992**, *6*, 61−78.

(7) Honma, T.; Hayashi, K.; Aoyama, T.; Hashimoto, N.; Machida, T.; Fukasawa, K.; Iwama, T.; Ikeura, C.; Ikuta, M.; Suzuki-Takahashi, I.; Iwasawa, Y.; Hayama, T.; Nishimura, S.; Morishima, H. Structure-based generation of a new class of potent Cdk4 inhibitors: new de novo design strategy and library design. *J. Med. Chem.* **2001**, *44*, 4615−27.

(8) Tan, J. J.; Zhang, B.; Cong, X. J.; Yang, L. F.; Liu, B.; Kong, R.; Kui, Z. Y.; Wang, C. X.; Hu, L. M. Computer-aided design, synthesis, and biological activity evaluation of potent fusion inhibitors targeting HIV-1 gp41. *Med. Chem.* **2011**, *7*, 309−16.

(9) Yuan, Y.; Pei, J.; Lai, L. LigBuilder 2: A Practical de Novo Drug Design Approach. *J. Chem. Inf. Model.* **2011**, *51*, 1083−1091.

(10) Gillet, V. J.; Newell, W.; Mata, P.; Myatt, G.; Sike, S.; Zsoldos, Z.; Johnson, A. P. SPROUT: recent developments in the de novo design of molecules. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 207−17.

(11) Sova, M.; Cadez, G.; Turk, S.; Majce, V.; Polanc, S.; Batson, S.; Lloyd, A. J.; Roper, D. I.; Fishwick, C. W.; Gobec, S. Design and synthesis of new hydroxyethylamines as inhibitors of D-alanyl-D-lactate ligase (VanA) and D-alanyl-D-alanine ligase (DdlB). *Bioorg. Med. Chem. Lett.* **2009**, *19*, 1376−9.

(12) Eisen, M. B.; Wiley, D. C.; Karplus, M.; Hubbard, R. E. HOOK: a program for finding novel molecular architectures that satisfy the chemical and steric requirements of a macromolecule binding site. *Proteins* **1994**, *19*, 199−221.

(13) Westhead, D. R.; Clark, D. E.; Frenkel, D.; Li, J.; Murray, C. W.; Robson, B.; Waszkowycz, B. PRO-LIGAND: an approach to de novo molecular design. 3. A genetic algorithm for structure refinement. *J. Comput. Aided Mol. Des.* **1995**, *9*, 139−48.

(14) Clark, D. E.; Frenkel, D.; Levy, S. A.; Li, J.; Murray, C. W.; Robson, B.; Waszkowycz, B.; Westhead, D. R. PRO-LIGAND: an approach to de novo molecular design. 1. Application to the design of organic molecules. *J. Comput. Aided Mol. Des.* **1995**, *9*, 13−32.

(15) Waszkowycz, B.; Clark, D. E.; Frenkel, D.; Li, J.; Murray, C. W.; Robson, B.; Westhead, D. R. PRO_LIGAND: an approach to de novo molecular design. 2. Design of novel molecules from molecular field analysis (MFA) models and pharmacophores. *J. Med. Chem.* **1994**, 37, 3994−4002.

(16) Hartenfeller, M.; Zettl, H.; Walter, M.; Rupp, M.; Reisen, F.; Proschak, E.; Weggen, S.; Stark, H.; Schneider, G. DOGS: reaction-driven de novo design of bioactive compounds. *PLoS Comput. Biol.* **2012**, *8*, e1002380.

(17) Roe, D. C. Computer-Aided Molecular Design: De Novo Design. In *Handbook of Chemoinformatics Algorithms*, first ed.; Faulon, J.-L., Bender, A., Eds.; Chapman and Hall/CRC Taylor & Francis Group: Boca Raton, FL, 2010; pp 295−315.

(18) Ji, H. Fragment-Based Drug Design: Considerations for Good ADME Properties. In *ADMET for Medicinal Chemists: A Practical Guide*, first ed.; Tsaioun, K., Kates, S. A., Eds.; John Wiley & Sons, Inc.: Hoboken, NJ, 2011; pp 417−485.

(19) Proschak, E.; Tanrikulu, Y.; Schneider, G. Fragment-based De Novo Design of Drug-like Molecules. In *Chemoinformatics Approaches to Virtual Screening*, first ed.; Varnek, A., Tropsha, A., Eds.; The Royal Society of Chemistry: Cambridge, UK, 2008; Vol. *0*, pp 217−239.

(20) Kutchukian, P. S.; Shakhnovich, E. I. De novo design: balancing novelty and confined chemical space. *Exp. Opin. Drug Disc.* **2010**, *5*, 789−812.

(21) Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *J. Cheminf.* **2009**, *1*, 8.

(22) Jia, Y.; Chiu, T. L.; Amin, E. A.; Polunovsky, V.; Bitterman, P. B.; Wagner, C. R. Design, synthesis and evaluation of analogs of initiation factor 4E (eIF4E) cap-binding antagonist Bn7-GMP. *Eur. J. Med. Chem.* **2010**, *45*, 1304−13.

(23) Stiefl, N.; Zaliani, A. A knowledge-based weighting approach to ligand-based virtual screening. *J. Chem. Inf. Model.* **2006**, *46*, 587−96.

(24) Gastreich, M.; Lilienthal, M.; Briem, H.; Claussen, H. Ultrafast de novo docking combining pharmacophores and combinatorics. *J. Comput. Aided Mol. Des.* **2006**, *20*, 717−34.

(25) Maass, P.; Schulz-Gasch, T.; Stahl, M.; Rarey, M. Recore: a fast and versatile method for scaffold hopping based on small molecule crystal structure conformations. *J. Chem. Inf. Model.* **2007**, *47*, 390−399.

(26) Zaliani, A.; Boda, K.; Seidel, T.; Herwig, A.; Schwab, C. H.; Gasteiger, J.; Claussen, H.; Lemmen, C.; Degen, J.; Parn, J.; Rarey, M. Second-generation de novo design: a view from a medicinal chemist perspective. *J. Comput. Aided Mol. Des.* **2009**, *23*, 593−602.

(27) Vinkers, H. M.; de Jonge, M. R.; Daeyaert, F. F.; Heeres, J.; Koymans, L. M.; van Lenthe, J. H.; Lewi, P. J.; Timmerman, H.; Van Aken, K.; Janssen, P. A. SYNOPSIS: SYNthesize and OPtimize System in Silico. *J. Med. Chem.* **2003**, *46*, 2765−73.

(28) Brady, G. P., Jr.; Stouten, P. F. Fast prediction and visualization of protein binding pockets with PASS. *J. Comput. Aided Mol. Des.* **2000**, *14*, 383−401.

(29) Wang, R.; Lai, L. H.; Wang, S. M. Further development and validation of empirical scoring functions for structure-based binding affinity prediction. *J. Comput. Aided Mol. Des.* **2002**, 16.

(30) Beato, C.; Beccari, A. R.; Cavazzoni, C.; Lorenzi, S.; Costantino, G. The use of experimental designs to optimize docking performances. The case of LiGenDock, the docking module of LiGen, a new de novo design program. *J. Chem. Inf. Model.* **2013**, DOI: 10.1021\ci400079k.

(31) Rotstein, S. H.; Murcko, M. A. GroupBuild: a fragment-based method for de novo drug design. *J. Med. Chem.* **1993**, *36*, 1700−10.

(32) Halgren, T. A. Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94. *J. Comput. Chem.* **1996**, *17*, 490−519.

(33) Halgren, T. A.; Nachbar, R. B. Merck molecular force field. IV. conformational energies and geometries for MMFF94. *J. Comput. Chem.* **1996**, *17*, 587−615.

(34) Halgren, T. A. Merck molecular force field. V. Extension of MMFF94 using experimental data, additional computational data, and empirical rules. *J. Comput. Chem.* **1996**, *17*, 616−641.

(35) Halgren, T. A. Merck molecular force field. II. MMFF94 van der Waals and electrostatic parameters for intermolecular interactions. *J. Comput. Chem.* **1996**, *17*, 520−552.

(36) Halgren, T. A. Merck molecular force field. III. Molecular geometries and vibrational frequencies for MMFF94. *J. Comput. Chem.* **1996**, *17*, 553−586.

(37) DeLano, W. L. *PyMol Molecular Graphic System*, version1.2r1; DeLano Scientific: South San Francisco, CA, 2009.