

# A Method for Automated Molecular Optimization Applied to Ames Mutagenicity Data

Ernst Ahlberg Helgee,\* Lars Carlsson, and Scott Boyer

Safety Assessment, AstraZeneca Research & Development, 43183 Mölndal, Sweden

Received June 24, 2009

The present work describes a method that optimizes a compound on the basis of the interpretation of quantitative structure–activity relationship models. The method has been applied to query compounds that have a mutagenicity liability. The substructure that contributes the most to the mutagenicity prediction is identified and replaced for each query compound. Replacement substructures have been generated in a deterministic fashion to produce a range of new, nonmutagen, compounds. A portion of the new compounds already exists in literature, but was unknown to the method during optimization. These results suggest that this method can substitute “toxic” substructures and produce libraries of compounds with lower liability for a given endpoint. This method is intended to complement or replace the database searches that chemists need to undertake when trying to avoid safety problems in compounds.

## 1. INTRODUCTION

Quantitative structure–activity relationship, QSAR, models are widely used for prediction of various endpoints, and the general inferences from these models guide chemists in making changes to their compounds. The concept of redesigning molecules by the use of inferences from QSAR models is not new, but the complexity of many problems addressed by QSAR models today renders highly complex models where a simple interpretation is oftentimes difficult. When a QSAR model predicts that a compound has an undesired property, the chemist needs to modify the compound to proceed. This work traditionally consists of database and literature searches, which together with inferences from the QSAR model aid the chemist in finding more promising compounds. The approaches used leave a difficult task to the chemists in finding new substituents that will result in more favorable properties. The process of improving compounds involves literature and database searches for possible substituents and fragments with desired physicochemical properties. This process is usually time-consuming. An additional issue is how chemists can extract relevant information from QSAR models when trying to redesign compounds.

In previous work, it has been shown how to identify active substructures in a compound<sup>1</sup> using the signature descriptor.<sup>2–4</sup> A query compound is predicted and in addition to the prediction an explanation as to which substructure contributes the most to the prediction. This knowledge can be used by the chemist to improve the query compound by replacing that part with something more favorable. The identification of bad substructures together with the algorithm of the generation of structures developed by Visco et al.<sup>5</sup> and Churchwell et al.<sup>6</sup> can replace large parts of the work that chemists do in this searching and enumeration of new molecules and fragments. Visco et al. use molecules in a data set that are disassembled into small building blocks

defined by the signature descriptor. This descriptor contains all necessary information for a reassembly, and the definition of which building blocks that fit together is described by a constraints equations. However, the reassembled compounds are much greater in numbers than the original compounds because the method allows for a deterministic and exhaustive recombination of those building blocks. Any information and particular fragments that the compounds of the data might contain are in a way transformed into new compounds. The generation algorithm is computationally intensive and, due to the complexity of the problem, slow for a complete regeneration of druglike compounds. The number of published applications thus far has been quite limited and usually describe limitations set on the constraints equations.

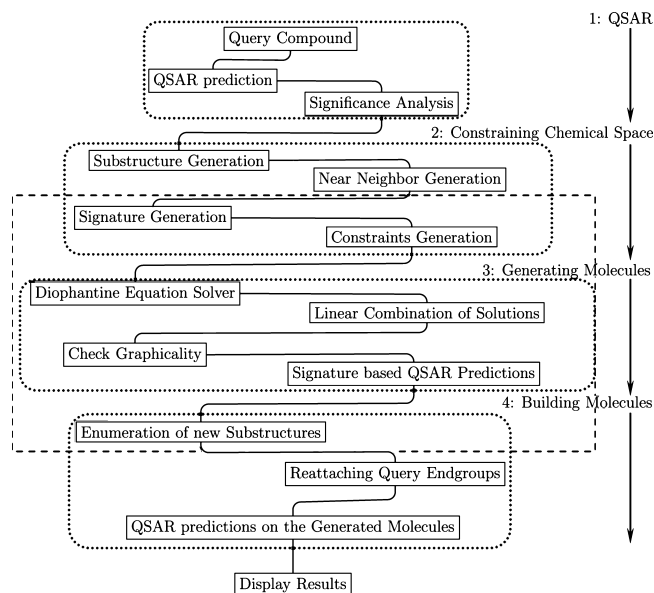
The remainder of this Article is organized as follows. In section 2, the combined algorithm is presented along with modifications necessary to adapt it to druglike compounds. Section 3 contains results, and this Article is concluded with discussions and conclusions in section 4.

## 2. METHOD

The method presented in this study makes use of the data behind the QSAR model and the QSAR model itself and automates the previously described process by relating the local importance of the QSAR model to substructures in the molecule. The substructure with the highest importance for the prediction is located and replaced. The different steps in this method are visualized in Figure 1.

The main flow can be described as follows. A query compound is predicted with a QSAR model for a specific endpoint. If the compound is active, the most significant substructure is removed, and to replace it a chemical subspace is needed. A near neighbor search is made among known compounds, and a set of active and inactive compounds are selected. On the basis of the signatures describing these compounds and the substructure itself, constraints that restrict how the signatures can be combined are formed. The system of equations that these constraints form are solved with a Diophantine Equation Solver, because they have non-

\* Corresponding author phone: +46317064681; e-mail: ernst.ahlberghelgee@astrazeneca.com.



**Figure 1.** Flow chart displaying the different steps of the work flow, where the dashed box indicates the work of Visco et al.<sup>5</sup> and Churchwell et al.<sup>6</sup>

negative integer solutions. These solutions represent compounds not yet assembled, so new substructures are built from these solutions. The substructures are then combined with the remains of the query compound, if possible, and predicted with the QSAR model. These results are presented to the chemist.

The starting point is a query compound that needs to be predicted for a biological activity or a set of activities using multiple QSAR models. When the query is submitted, the AstraZeneca in-house tool LEATHERFACE<sup>7</sup> is used to generate canonical tautomers for the compound. Furthermore, chirality is also removed in this step using text substitution.

The descriptors used in the QSAR models are signatures,<sup>3,4</sup> and the model function is generated using any machine-learning method, such as random forests<sup>8</sup> or support vector machines.<sup>9</sup> Since the signatures for a compound represent each atom and its surroundings, up to a predefined height  $h$ , the local interpretation of machine-learning models described in ref 1 can be used to extract the signature with the most significant contribution to the QSAR prediction of the compound. This most significant signature corresponds to positions in the compound where changes possibly need to be made to get a different prediction from the QSAR model. The following procedure is only performed for compounds that receive unfavorable predictions.

The next step is to localize the atoms of the most significant signature in the compound and to retrieve a substructure generated by cutting bonds from the atoms at a specified distance from the center atom of the signature. If an atom at this distance belongs to a ring according to the OEChem<sup>10</sup> function `OEAtomBase::IsInRing()`, the path searched is extended to embed the ring. Each atom for which a bond is cut is kept as an anchor atom, and for each such atom a SMARTS<sup>11</sup> is generated that describes the atoms around the bond that was cut. To recombine generated substructures and the original end groups, these SMARTS must match. If the query compound cannot be cut, it will be treated as a substructure throughout the remainder of this

**Table 1.** Signatures for the Generation of Constraints Equations

	${}^h\tau_i$	${}^{h-1}\tau_i$	${}^{h-1}\sigma_{\tau_i}$
1	<chem>[c]([c]([c][o]))</chem>	<chem>[c]([c])</chem>	<chem>[c]([c][c][o])</chem>
2	<chem>[c]([c][c][o]([c]))</chem>	<chem>[c]([c][c][o])</chem>	<chem>[c]([c])</chem> <chem>[c]([c])</chem> <chem>[o]([c][c])</chem>
3	<chem>[o]([c]([c][c]) [c]([c][c]))</chem>	<chem>[o]([c][c])</chem>	<chem>[c]([c][c][o])</chem> <chem>[c]([c][c][o])</chem>

**Table 2.** Constraint Matrix

${}^{h-1}\tau_i \rightarrow {}^{h-1}\sigma_{\tau_j}$	$k_1$	$k_2$	$k_3$
<chem>[c]([c]) \rightarrow [c]([c][c][o])</chem>	1	-2	0
<chem>[c]([c][c][o]) \rightarrow [o]([c][c])</chem>	0	1	-2

algorithm. However, it will not go through the recombination step where SMARTS are used.

Neighbors to the substructure are retrieved on the basis of similarity computed from Daylight fingerprints.<sup>12</sup> The near neighbor search is conducted in a database of compounds for which measured activity is available for the specific endpoints and in particular the endpoint that the QSAR model approximates. From these neighbor compounds, a set is chosen that covers a range in activity around the query compound.

The method used to generate new substructures has been described by Visco et al.<sup>5</sup> and Churchwell et al.<sup>6</sup> where compounds are decomposed into building blocks represented as signatures. These building blocks define a chemical space in which any possible combination is rebuilt under the restrictions imposed by the signatures. The implementation used here differs slightly from that used by Churchwell.<sup>6</sup> Most of the changes have been applied to constrain the size of the new substructures.<sup>13</sup> This part of the algorithm consists of setting up connectivity constraints, described using equations, for how the signatures of the compounds can be combined and then solving the resulting system of linear equations. The constraints are created by comparing parts of the signatures. The signatures themselves describe a center atom, its  $h$  layers of surrounding atoms, and the bond types connecting the atoms. By looking at the environment around the center atom, it is possible to see what the surroundings of an other atom must be in its  $h - 1$  layers to be able to connect to the center atom. For each signature, in the set of height  $h$  signatures spanning the chemical space, the height  $h - 1$  signature,  ${}^{h-1}\tau_i$ , is computed along with the  $h - 1$  signatures for the first layer neighbors,  ${}^{h-1}\sigma_{\tau_i}$ ; see Table 1. To form a bond between two atoms  $i, j$  described by the signatures  ${}^h\tau_i$  and  ${}^h\tau_j$ , at least one of the  ${}^{h-1}\sigma_{\tau_j}$  must match the  ${}^{h-1}\tau_i$  and vice versa. In this comparison, a direction is imposed on the bond  $i \rightarrow j$ . For each such pair of  $h - 1$  signatures, an equation is set up such that for  ${}^h\tau_i$  the number of possible connection pairs is counted and added as a coefficient,  $k_i$ , to the equation; see Table 2.

In the cases where the connection environments are identical for both atoms, a dummy variable has to be added to balance the equation. The coefficients of these equations form a constraints matrix that defines the molecular subspace in which new compounds are built. Because the representation of signatures in compounds is an enumeration of atom types and their neighboring environment, solutions to the system of equations must be of integer form, and the

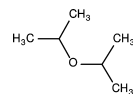
coefficients cannot be less than zero. Each equation can be described as  $\sum_{i=1}^n k_i x_i = 0$ , where  $k \in \mathbb{Z}$ . To solve the system of equations, a Diophantine Equation Solver algorithm developed by Devie and Contejean et al.<sup>14</sup> has been implemented. This algorithm is doing a stack-based search and retrieves the complete set of minimal solutions, where a minimal solution is a solution that cannot be obtained by combining other solutions. The algorithm starts from the origin and moves stepwise in attribute space. The endpoint of a valid step is pushed onto a stack, and each new step starts with a pop from that stack. For each pop, the algorithm evaluates possible steps in attribute space and pushes the points for which steps were allowed. A step in an attribute direction is only allowed if it represents a move closer toward the origin in constraints space. A minimal solution to the system of equations is found when a step reaches the origin in constraints space. The stack-based version of this algorithm prevents the search from finding the same minimal solution many times by blocking attribute directions in a way so that a particular subspace will only be searched once. This feature has been extended in our version such that a step in the attribute direction can be blocked once the attribute reaches a given threshold.

Another restriction has been imposed on the Diophantine Equation Solver to avoid the computation of solutions where the sum of signatures exceeds a predefined threshold. In a subsequent step, linear combinations of the solutions are made, where an upper and a lower bound are set on the size of a solution. These bounds are imposed to reduce computational time and to only generate solutions of size similar to that of the original substructure. The resulting solutions must also fulfill the graphicality equation,  $\sum_{i=1}^{\vartheta_{\max}} (i-2)g_i + 2 = 2c$ , which originates from graph theory.<sup>15</sup> The graphicality equation determines if a set of vertexes can establish a connected graph, and, if so, how many cycles it contains. A compound can be viewed as a connected graph where all vertexes (atoms) are saturated. If a compound *G* has *n* atoms and *m* bonds, then its cyclomatic number is  $c = m - n + 1$ . This is the number of independent cycles in *G*. Let  $g_i$  be the number of atoms in *G* with heavy atom valence  $\vartheta = i$ , then  $n = \sum_{i=1}^{\vartheta_{\max}} g_i$  and  $2m = \sum_{i=1}^{\vartheta_{\max}} i \cdot g_i$ . From this, the above graphicality equation can be derived.

At this point, it is possible to impose more restrictions on the solutions. If QSAR models have been built using signatures of height *h* or less, then the solutions representing nonbuilt substructures can be used to predict the properties of the new substructures even before they are rebuilt.

Next, in a recursive procedure, all possible substructures are created from the signatures according to the solutions. This is accomplished using an implementation of the algorithm proposed by Visco et al.<sup>5</sup> The algorithm recursively reassembles atoms from the solutions to form possible substructures, and it only allows the canonical structures to be built and thus reduces the construction time.

Because the Diophantine Equation Solver is complete, it ensures that all possible substructures within the subspace defined by the constraints matrix are found. The substructures built are preprocessed in the same way as the query compound, and additional filters are applied to control ring size. The different filters applied can be chosen on the basis of the specific problem at hand. If the query compound has anchor atoms, SMARTS are generated that describe the



**Figure 2.** Example compound.

**Table 3.** Query Compounds Chosen from the CCRIS Data for Examples 1–4

1	2	3	4

anchor atoms and their required neighbors in the substructure. The new substructures are searched for the pattern, and when a substructure, compatible with the patterns of the side groups, is found, the side groups are attached to the substructure to form new compounds. For the case where a side group can be attached to several points in the substructure, all permutations are assembled. The generated compounds are preprocessed in the same way as the query compound, and any duplicates are removed. If the query compound could not be partitioned into a substructure and its corresponding side groups, substructures from the above step are the complete new compounds. A final filtering step can be applied to remove compounds with undesired properties. Examples of such filters are drug likeness, ring compositions, and molecular weight. For the new compounds (Figure 2), QSAR predictions are obtained for the different biological endpoints of interest.

### 3. RESULTS

The work flow is demonstrated using AMES mutagenicity data from CCRIS<sup>16</sup> from which compounds and corresponding activity have been collected according to the conditions described by Kazius et al.<sup>17</sup> The example queries were run on an HP workstation with an Intel Xeon 3 GHz processor and took about 50 h to run in total. The rebuilding part was submitted to a heterogeneous computer grid where about 60 cores were used for each query.

The selected compounds have been preprocessed in the same way as the query compound and then converted to a database where the SMILES and fingerprints have been added together with the activity. From the database, a set of query compounds have been selected; see Table 3.

The compounds were selected on the criteria that they should have a positive assay activity. With respect to each of these query compounds, 100 neighbors, based on fingerprint similarity, were selected from the database, the 50 nearest positive and the 50 nearest negative. If the query was in the database, it was removed. The selected compounds for each query can be found in the Supporting Information “QSAR Training Sets”, and the mean similarity among the selected positive and negative compounds is displayed in Table 4. This set of neighbors was the training set for the QSAR model that was built to predict the query compound. The QSAR model was built with height 2 signatures using translator and libSVM with a radial basis function kernel. The SVM model was selected from a 5-fold cross validation procedure where the parameters  $\gamma$  and  $C$  were optimized



**Table 4.** Modeling Data for the QSAR Model

query compound	1	2	3	4
$\gamma$	$4.88 \times 10^{-4}$	$1.95 \times 10^{-2}$	$4.88 \times 10^{-4}$	$6.25 \times 10^{-2}$
C	2048.00	8.00	32.00	32.00
cross validation rate	81.00	69.00	85.00	80.00
mean similarity, positive neighbors	0.34	0.42	0.38	0.52
mean similarity, negative neighbors	0.35	0.38	0.33	0.43

using the grid-search method included in libSVM.<sup>9</sup> Details for the models are shown in Table 4.

The most significant component of the compound was retrieved, see Table 5, and a substructure was generated around it.

Moving into “2: Constraining Chemical Space” from the outline in Figure 1, a set of 20 neighbors to the selected substructure was retrieved, 10 positive and 10 negative listed in the Supporting Information “Molecular Generation Training Set”. The mean similarities for the selected sets are displayed in Table 6 separated into positive and negative neighbors. The total set of signatures, from the neighbors and the substructure itself, were used to set up the connectivity constraints for the Diophantine Equation Solver.

The number of signatures and connectivity equations for each query compound can be found in Table 6. In the “3: Generating Molecules” part, the Diophantine Equation Solver was configured to only compute solutions with a restriction on the maximum number of atoms. In the generation of linear combinations of the minimal solutions, the maximum number of atoms was used together with a minimum number of atoms, both displayed in Table 6, to restrict the size of the final compounds.

From the solutions, all solutions that result in proper compounds, cyclic compounds are restricted to 5 and 6 member rings, are built as outlined in the “4: Building Molecules” of Figure 1. To reduce the time needed to generate the new compounds, the “Enumeration of New Substructures” was submitted to a heterogeneous computer grid using Sun Grid Engine.<sup>18</sup> The number of compounds presented in Table 6 as compounds built are compounds where the end groups could be added to the new substructures, and all generated compounds have been filtered to remove odd ring structures. All generated compounds have been predicted with the QSAR model, which was used for the respective query compound; all generated compounds can be found in the Supporting Information “Generated Compounds”. For query molecule 3, a subset of the generated compounds already existed in MCASE<sup>19</sup> or CCRIS; these 10 compounds were correctly predicted using the QSAR model.

In addition, 303 of the positive compounds from the Ames data set by Kazius et al.<sup>17</sup> have been tested. These computations took roughly 1 month using six CPUs for building models and solving the System of Diophantine Equations. The rebuilding process was distributed on a heterogeneous grid using at maximum 100 nodes for rebuilding compounds. Out of the 303 compounds, 181 were predicted to be positive by the model, and the corresponding statistics (covering number of generated compounds, percentage generated compounds predicted positive, computational time, number of solutions, and the number of steps taken by the Diophantine Equation Solver) for those computations are provided

in the Supporting Information, Section B. The rest of the compounds were predicted to be negative by the model and therefore not optimized. In the set of optimized compounds, 18 of the 28 approved toxicophores described by Kazius et al. were covered; see the Supporting Information for details. The computational time varies between a few minutes for aliphatic compounds to several days for some of the bicyclic aromatic compounds. Up to 15 000 compounds have been generated for a single query, whereas in some cases only a handful of new compounds have been generated, which affects the computational time.

#### 4. DISCUSSION AND CONCLUSIONS

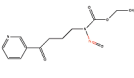
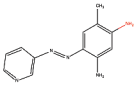
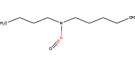
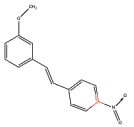
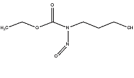
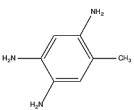
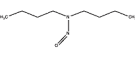
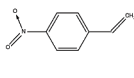
This Article illustrates a complete automated work-flow for molecular optimization. The method builds new molecules deterministically and incorporates valuable information from available data and models regarding the field where it is applied. Information handling is very flexible, and the method can be applied to any problem of interest and any source of structure activity data that can be represented using signatures or any descriptor set that allows rebuilding of compounds. The QSAR model that is built using signatures mines the underlying data and extracts the locally most important substructures. This provides the chemist with an extensive platform for understanding the chemical–biological topography around the compound of interest as well as gives suggestions on possible changes and their respective effects.

In contrast to molecular evolution algorithms, which make use of random search algorithms to find new compounds,<sup>20</sup> the deterministic search of this method allows for exploration of unknown chemistry close to known chemistry. This is useful in finding ways around problems due to the chemical structure and can aid the generation of fast followers where a part of the molecule needs to be replaced to avoid patent infringement. In a case like this, the substructures would have to be picked manually.

In this Article, Ames mutagenicity has been used as an endpoint to illustrate the method. However, the method can be applied to any biological endpoint and in combination with other endpoints in a multiobjective optimization. Various QSAR models that are built on arbitrary descriptors can also be used in the rebuilding stage of the proposed method, as additional filters that restrict the possibility of generating compounds.

The computational time needed for the method increases dramatically with the number of signatures due to the combinatorial explosion of possible compounds as the number of unique signatures increases. The reason for this is that the method is a deterministic search method, meaning that all signature combinations that may result in new compounds will be found, but it may take awhile to find them. With a higher number of unique signatures, the chemical diversity among the generated molecules will be higher, and therefore a trade-off between computational time and chemical diversity has to be made. There are different approaches that could be applied to reduce the computational time needed. Further investigation of how the connectivity constraints affect the computational time may be of value. There is also a possibility to generate compounds from the solutions as they appear instead of waiting until the Diophantine Equation Solver has returned all of them. This

**Table 5.** Most Significant Component and Most Significant Substructure for Examples 1–3

Query Compound	1	2	3	4
significant component				
significant substructure				

**Table 6.** Data for the Generation of New Compounds

query	1	2	3	4
mean similarity, positive neighbors	0.39	0.77	0.63	0.74
mean similarity, negative neighbors	0.32	0.66	0.46	0.55
number of signatures	124	28	42	45
number of connectivity constraint equations	81	14	25	24
minimum number of atoms	7	5	6	6
maximum number of atoms	17	15	16	16
number of diophantine solutions	737	139	186	1380
number of compounds built	2730	294	1851	290
wall-clock time used (h)	8	2	29	8

way, a user could start almost immediately to examine new compounds and also be able to assess the validity of the proposed compounds early on without having to occupy too much of the computational resources.

To enumerate and rebuild complete compounds with the algorithms proposed by Visco et al. and Churchwell et al. is a costly procedure when it comes to druglike compounds containing multiple cyclic systems. In drug discovery, small changes in the chemical structure can have a considerable effect on the activity. By adding the identification of significant substructures and limiting the size of similar substructures, as proposed here, it is possible to regenerate compounds that are valid in a drug-design context. This is clearly shown by the examples in section 3 and by the extensive run where 181 compounds were optimized with Ames mutagenicity as the endpoint.

**Supporting Information Available:** Tables displaying compounds used and compounds generated for the four examples in the Article (first section) and overview results for a larger set of test compounds run through the proposed method (second section).

The first section has three subsections “QSAR Training Sets” displaying the compounds to build the QSAR models for each test case, “Molecules used for Constraints Generation” displaying the compounds used when setting up the constraints, and “Generated Compounds” where the generated compounds are presented for each test case, respectively.

The second section shows the results from a test set of 303 Ames positive compounds from the data set in Kazius et al.,<sup>17</sup> run through the presented method. The first 122 compounds were predicted to be negative and therefore not optimized. The other 181 compounds were optimized, and

a table presents the smiles of the optimized compounds together with the number of compounds generated, the fraction of generated compounds predicted to be positive, the computational time, the number of solutions to the system of equations, and the number of steps taken by the Diophantine Equation Solver. The set of optimized compounds has been matched against the approved toxicophores in Kazius et al.,<sup>17</sup> and the result of the SMARTS matching is presented in a table containing SMARTS name, the number of hits among the positive compounds in the Kazius data set, and the number of hits among the 181 optimized compounds. This material is available free of charge via the Internet at <http://pubs.acs.org>.

## REFERENCES AND NOTES

- (1) Carlsson, L.; Ahlberg Helgee, E.; Boyer, S. *J. Chem. Inf. Model.* **2009**, doi: 10.1021/ci9002206.
- (2) Faulon, J.-L. *Translator*; <http://www.cs.sandia.gov/jfaulon/QSAR/translator.tar.gz> (accessed Jun 2008).
- (3) Faulon, J.-L.; Visco, D. P. J.; Pophale, R. S. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 707–720.
- (4) Faulon, J.-L.; Churchwell, C. J. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 721–734.
- (5) Visco, D. P. J.; Pophale, R. S.; Rintoul, M. D.; Faulon, J.-L. *J. Mol. Graphics Modell.* **2002**, *20*, 429–438.
- (6) Churchwell, C. J.; Rintoul, M. D.; Shawn, M.; Visco, D. P. J.; Kotu, A.; Larson, R. S.; Sillerud, L. O.; Brown, D. C.; Faulon, J.-L. *J. Mol. Graphics Modell.* **2004**, *22*, 263–273.
- (7) Kenny, P.; Sadowski, J. *Chemoinformatics Drug Discovery* **2005**, 271–285.
- (8) Breimann, L. *Machine Learning* **2001**, *45*, 5–32.
- (9) Chang, C.-C.; Lin, C.-J. *LIBSVM: a library for support vector machines*; 2001; software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- (10) *Openeye Scientific Software*, <http://www.eyesopen.com> (accessed Aug 30, 2005).
- (11) *Daylight Theory: SMARTS - A Language for Describing Molecular Patterns*, <http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html> (accessed Jan 13, 2008).
- (12) *Fingerprints*, <http://daylight.com/dayhtml/doc/theory/theory.finger.html> (accessed Jan 20, 2008).
- (13) Helgee, E. A. M.Sc. thesis, Chalmers University of Technology, 2005.
- (14) Contejean, E.; Devie, H. *Inf. Comp.* **1994**, *113*, 143–172.
- (15) Berge, C. *Graphs and Hypergraphs*; Elsevier Science Ltd.: New York, 1985.
- (16) *TOXNET - Chemical Carcinogenesis Research Information System*, <http://toxnet.nlm.nih.gov> (accessed Nov 22, 2006).
- (17) Kazius, J.; McGuire, R.; Bursi, R. *J. Med. Chem.* **2005**, *48*, 312–320.
- (18) Gentzsch, W. *IEEE Int. Symp. Cluster Computing Grid* **2001**, 35.
- (19) *MultiCASE Inc.*, <http://www.multicase.com/products/prod0910.htm> (accessed Jan 13, 2008).
- (20) Willett, P. *Trends Biotechnol.* **1995**, *13*, 516–521.

CI900221R