Received: 3 January 2022 | Revised: 5 April 2022 | Accepted: 4 May 2022

DOI: 10.1002/wcms.1630

ADVANCED REVIEW

WIREs COMPUTATIONAL MOLECULAR SCIENCE WILEY

Network search algorithms and scoring functions for advanced-level computerized synthesis planning

Bartosz A. Grzybowski[1,2,3] | **Tomasz Badowski**[1] | **Karol Molga**[1] | **Sara Szymkuć**[1]

[1]Institute of Organic Chemistry, Polish Academy of Sciences, Warsaw, Poland

[2]Center for Soft and Living Matter, Institute for Basic Science (IBS), Ulsan, Republic of Korea

[3]Department of Chemistry, Ulsan National Institute of Science and Technology (UNIST), Ulsan, Republic of Korea

Correspondence
Bartosz A. Grzybowski, Department of Chemistry, Ulsan National Institute of Science and Technology (UNIST), 50 UNIST-gil, Ulsan 44919, Republic of Korea.
Email: nanogrzybowski@gmail.com

Edited by: Raghavan Sunoj, Associate Editor

Abstract

In 2020, a "hybrid" expert-AI computer program called Chematica (a.k.a. Synthia) was shown to autonomously plan multistep syntheses of complex natural products, which remain outside the reach of purely data-driven AI programs. The ability to plan at this level of chemical sophistication has been attributed mainly to the superior quality of Chematica's reactions rules. However, rules alone are not sufficient for advanced synthetic planning which also requires appropriately crafted algorithms with which to intelligently navigate the enormous networks of synthetic possibilities, score the synthetic positions encountered, and rank the pathways identified. Chematica's algorithms are distinct from *prêt-à-porter* algorithmic solutions and are product of multiple rounds of improvements, against target structures of increasing complexity. Since descriptions of these improvements have been scattered among several of our prior publications, the aim of the current Review is to narrate the development process in a more comprehensive manner.

This article is categorized under:

Data Science > Computer Algorithms and Programming
Data Science > Artificial Intelligence/Machine Learning
Quantum Computing > Algorithms

KEYWORDS

artificial intelligence, Chematica, expert systems, networks, synthesis

1 | INTRODUCTION

The first attempts to use computers to design multistep syntheses[1–7] date back to the 1960's and have, back then, met with considerable interest and anticipation that the challenge of computerized retrosynthesis would be addressed in short order. Unfortunately, time passed yet the vision of computers providing complete and chemically sound retrosynthetic plans remained unrealized[8]—in fact, the experimental validations of machine's suggestions for the synthesis of more complex targets failed[9,10] whereas those for simpler targets gave disappointing yields,[11] worse than those in the human-designed syntheses. In addition, the early programs were not really fully automated and, as in the pioneering LHASA, provided only lists of one-step suggestions, from which the human operator had to construct complete pathways. Given that the numbers of reactions potentially applicable at each step are in tens to hundreds, such manual navigation rapidly explodes the number of possibilities and the term "combinatorial explosion" has been coined[12] to, pretty much, deem the problem intractable, especially for the syntheses of complex targets requiring tens of steps.

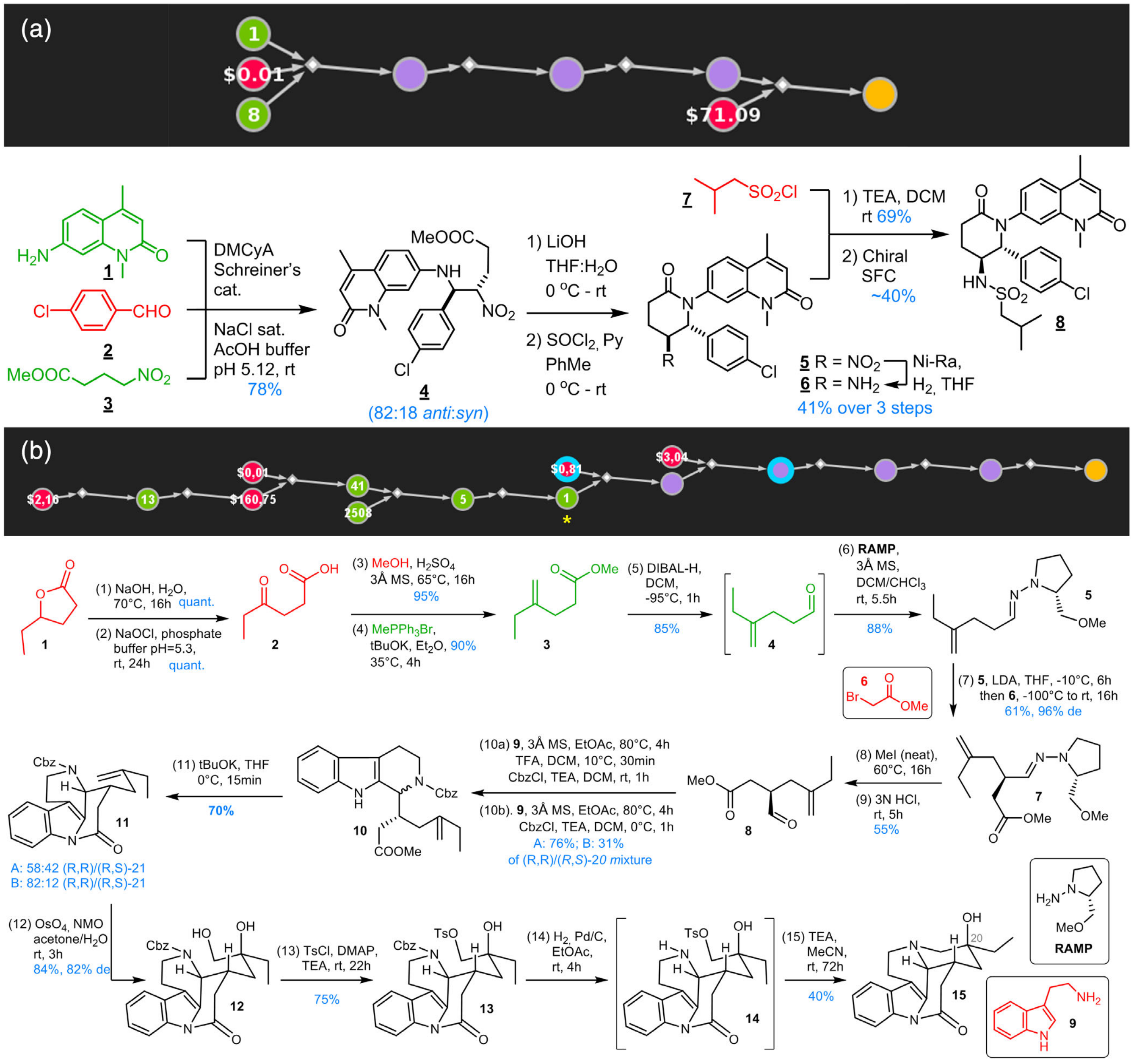
Notwithstanding, starting in the early 2000's, we revisited the problem from a perspective of large-scale networks[13–15] akin to those underlying chess-playing programs such as the then-popular DeepBlue. Namely, we envisioned the process of retrosynthetic design as a combination of (1) The "moves" defined by the reaction rules; (2) The large-scale networks, defined by the retron-to-synthon expansions according to these rules; and (3) The scoring functions that would guide the navigation over such networks to find reaction pathways starting from readily available and structurally simple materials. In the initial series of papers,[13–18] between 2005 and 2012, we described how large reaction datasets can be translated into the bi-partite network representation and how they can be navigated to trace and score reaction routes. These early attempts were based on "static" networks of the already published reactions but by ca. 2015, we have augmented them with large-enough set of reaction rules that allowed de novo synthesis planning for arbitrary and/or previously unknown targets (yielding, of course, syntheses not reported in the literature). These key elements were combined in our Chematica platform and were formalized in the 2016 publication[19] which, in many ways, revived broader interest in computational synthesis. In that article, we presented first completely autonomously designed and unprecedented routes to already nontrivial—though not yet complex—targets such as goniothalesdiol A or (+)-Cryptocarya diol.

From this point onward, two major lines of thinking have emerged. We continued to develop Chematica with the main focus on expert-coded reaction rules[20] and dual ("chemicals" and "reactions") scoring functions based on chemically interpretable variables[21,22] (numbers of rings or stereocenters, penalties for group incompatibilities, etc.). In parallel, we augmented these key components with various theoretical methods ranging from molecular mechanics, MM, to quantum mechanics, QM,[21] to Machine Learning, ML.[23,24] By contrast, virtually all other teams pursued purely data-driven AI strategy,[25–32] largely motivated by the assumption that only automated extraction of rules can keep the pace of the "exponentially expanding" body of chemical knowledge (although, as recently shown, the growth is not nearly as rapid, and only a very small fraction of reaction classes have enough literature examples to allow for meaningful machine learning).[33] These efforts have been based either on (i) automatic extraction of rules from large reaction databases and then scoring the reaction "moves" by various types of neural networks, NNs, trained on these databases; or (ii) template-free methods using the so-called transformer NNs trained on large reaction repositories.[29,31]

Given the recent feats of AI approaches in many other areas, the outcome of the "retrosynthesis rivalry" has been somewhat curious and not in AI's favor. On one hand, the success of our "hybrid" expert-MM-QM-ML approach has been documented by multiple experimental validations—by us[21,22,34] and others[35]—of complete, Chematica-designed reaction routes leading not only to medicinally relevant targets[21,35] (Figure 1a) but also complex natural products,[22,34] at the forefront of modern organic synthesis (Figure 1b). Moreover, in,[22] we showed that world's leading experts were no longer able to distinguish the machine-planned syntheses of very complex targets from those designed by humans. The program, rebranded as Synthia, is now commercialized and deployed by Merck KGaA in industry and academia[35,36] worldwide. On the other hand, the AI-based efforts have been very prolific in proposing various NN architectures but the synthetic routes they produce continue to be limited to simple targets for which, one could argue, a competent chemist would not really need any computational support. In particular, even when trained on reaction collections as vast as Reaxys or SciFinder, such programs fail to plan routes to targets incorporating demanding stereochemical or skeletal motifs (as, indeed, we verified in[22] for the natural products Chematica tackled). We are also unaware of any experimental validations of nontrivial syntheses suggested by these platforms.

While this dichotomy in performance is now becoming recognized,[36–39] the source of Chematica's successful synthetic planning has most often been attributed to its superior knowledge base of reaction rules which—based on the underlying reaction mechanism—are coded to delineate a broad spectrum of functional groups and structural fragments *incompatible* with a given reaction.[20] Indeed, in this way, the rules can indirectly map the space of "impossible" reactions to which literature-based AI methods have really no access (other than stipulating that putative reactions not reported in the literature are impossible, which is a dangerous oversimplification; see footnote #9 in [[40]]).

The quality of its chemical knowledge base is certainly important (and have been reviewed as such in [[19,20,41]]), but this is not the sole significant difference between Chematica and its AI contenders. The other key component are the algorithms with which these programs navigate the expanding networks of synthetic possibilities to find and then rank complete synthetic pathways. In the AI approaches, the key development—since then used in most other works on the topic—was the 2018 paper by Segler et al.[32] in which they adapted the so-called Monte Carlo Tree Search, MCTS, to expand the retrosynthetic tree by random sampling of the search space using rollouts starting from the tree's nodes. The origins of MCTS can be traced to the 2006 paper,[42] where the method (then, under a the name of "bandit-based Monte-Carlo") outperformed some alternatives, like asynchronous dynamic programming.[43] By contrast, Chematica has consistently used its own set of network-search algorithms with which it scores "synthetic positions" defined in



(a)
(b)
DMCyA Schreiner's cat.
NaCl sat. AcOH buffer pH 5.12, rt 78%
(82:18 anti:syn)
1) LiOH THF:H2O 0 °C - rt
2) SOCl2, Py PhMe 0 °C - rt
1) TEA, DCM rt 69%
2) Chiral SFC ~40%
5 R = NO2 Ni-Ra,
6 R = NH2 H2, THF
41% over 3 steps
(1) NaOH, H2O, 70°C, 16h quant.
(2) NaOCl, phosphate buffer pH=5.3, rt, 24h quant.
(3) MeOH, H2SO4 3Å MS, 65°C, 16h 95%
(4) MePPh3Br, tBuOK, Et2O, 90% 35°C, 4h
(5) DIBAL-H, DCM, -95°C, 1h 85%
(6) RAMP, 3Å MS, DCM/CHCl3 rt, 5.5h 88%
(7) 5, LDA, THF, -10°C, 6h then 6, -100°C to rt, 16h 61%, 96% de
(8) MeI (neat), 60°C, 16h
(9) 3N HCl, rt, 5h 55%
(10a) 9, 3Å MS, EtOAc, 80°C, 4h TFA, DCM, 10°C, 30min CbzCl, TEA, DCM, rt, 1h
(10b). 9, 3Å MS, EtOAc, 80°C, 4h CbzCl, TEA, DCM, 0°C, 1h
A: 76%; B: 31% of (R,R)/(R,S)-20 mixture
(11) tBuOK, THF 0°C, 15min 70%
A: 58:42 (R,R)/(R,S)-21
B: 82:12 (R,R)/(R,S)-21
(12) OsO4, NMO acetone/H2O rt, 3h 84%, 82% de
(13) TsCl, DMAP, TEA, rt, 22h 75%
(14) H2, Pd/C, EtOAc, rt, 4h
(15) TEA, MeCN, rt, 72h 40%
RAMP

FIGURE 1 Examples of syntheses designed autonomously by Chematica and later validated in the laboratory. (a) In 2018, using its basic search algorithm and scoring functions, Chematica was able to design efficient routes to medicinally important drug molecules[21]—like the BRD 7/9 inhibitor shown here. In doing so, it was already showing some creativity: here, it skillfully applied the three-component aza-Henry reaction to design a concise and much higher yielding route than the previously published eight-step synthesis of the same target. (b) By 2020, with improved knowledge base and algorithms, the program was able to plan total syntheses of complex natural products. Shown here is the synthesis of (*R*,*R*,*S*)-tacamonidine. This target was chosen because it had not been synthesized before and because the multiple syntheses of its close analogue, tacamonine, are not amenable to the construction of the quaternary hydroxylated stereocentre present in tacamonidine. For even more complex targets and routes, see Reference 22 and its extended figures and supplementary information. Here, miniatures of Chematica's bi-partite graphs are shown above corresponding synthetic panels. In these graphs, *orange* circular node = target; *violet* nodes = unknown substances; *green* nodes = known substances (with numerals denoting synthetic popularity, i.e., the number of literature-reported syntheses in which a particular molecule was used as a substrate; *red* nodes = terminal, commercially available chemicals (with prices in \$/g); *blue* halo = protection needed. In the synthesis panels, substrates drawn/listed in *red* and *green* correspond to the red and green nodes in Chematica's scheme. Experimental yields are given in *blue* font below reaction arrows. In panel (b), *yellow* star denotes intermediate **4** that is already known and whose synthesis could be found from Chematica's literature-based Network of Organic Chemistry module[13–19]—however, upon reaching this starting material, the algorithm was allowed to continue the de novo search and navigate the synthesis all the way to very simple, commercially available building blocks (terminal *red* nodes). Figure and parts of the caption reproduced with permission (a) from Reference 21 (Copyright 2018 Elsevier) and (b) from Reference 22 (Copyright 2020 Springer Nature)

terms of *both* the molecules it creates and reaction operations it performs. In doing so, the program pursues multiple search strategies supported by beam-search-inspired[44] priority queues as well as multistep strategizing routines[22,33] to seek more far-sighted synthetic plans. Importantly, when MCTS and our algorithms were compared and used in conjunction with Chematica's other components, both algorithms performed well for simple targets (which is in line with a recent analysis of MCTS vs. an algorithm similar to Chematica's early versions [[45]]) but it was only the latter that was able to navigate routes to complex natural products—as opposed to MCTS which typically found no pathways[22] (likely because difficult syntheses cannot be finished by MCTS's rollouts).

The focus of the present Review, however, is neither to delve into such comparisons nor to exclude the possibility that in the future, the MCTS or other types of algorithms can be improved and adapted to advanced-level synthesis planning. Instead, our objective here is to summarize the principles on which Chematica's current and successful synthesis-planning search routines are based. This information has so far been scattered among several of our publications[19,21,22,34,41,46–48] and likely described in insufficient detail. In this light, we think it is timely and important to provide a more thorough and unifying description of our algorithms and scoring functions. In doing so, our hope is that they can be adapted to other synthesis-planning programs to enable synthetic designs more advanced than in the current AI-based methods, and that they can also foster development of even more efficient network-search and scoring routines.

2 | GENERAL CONSIDERATIONS FOR COMPUTERIZED RETROSYNTHESIS

The process of retrosynthesis aims to design at least one viable synthetic route to some target molecule of interest by iterative application of reaction rules encoding commonly accepted reaction types (say, S_N2, Diel-Alder, Michael addition, Pd-catalyzed couplings, and many more). The reaction-core atoms, scope of admissible substituents in the core's neighborhood, and also lists of incompatible groups can generally be expert-coded quite flexibly in the SMARTS notation, as described in detail in.[20] However, it is worth re-emphasizing that in many cases, such SMARTS-encoded rules need to be fine-tuned by more advanced models. For instance, for aromatic substitutions, SMARTS notation is ill-suited to encode all possible ring types and substitution patterns. Instead, the positions at which such substitutions are allowed are determined by algorithms utilizing, inter alia, Hammett constants and proton affinity calculations (Figure 2a). Hammett constants and steric indices are also used to vectorize certain reaction types for which statistics are abundant enough to allow for Machine Learning (cf. vectorization of the popular and synthetically powerful Diels-Alder reaction in Figure 2b,c). Moreover, when the conformations taken by the molecules matter for reaction outcomes, Molecular Mechanics calculations are useful (e.g., to determine which types of intramolecular cyclizations are plausible, Figure 2d).

Assuming the single-reaction rules are thus properly defined, they are first applied to only the target and produce the first generation of substrates. These substrates (a.k.a. "synthons") can then become the "targets" (a.k.a. "retrons") of progeny searches—that is, the reaction rules may be applied to them to produce the second-generation of substrates/synthons. The process is iterated until all substrates of a given reaction operation meet some stop criteria (e.g., they are commercially available). Such stop-substrates are no longer expanded and "solve" the retrosynthesis problem in that a path linking them to the target molecule has been established.

This general scheme merits four important comments that also outline the order of our subsequent discussion:

i. The iterative expansions create a network delineating the space of synthetic options.
ii. This network can be extremely large. As described in,[19] there are, on average, some 100 reaction rules that "match" any given retron molecule without any cross-reactivity conflicts, and the network is expanding extremely rapidly, with ~100^n molecule nodes within n synthetic steps from the target.
iii. Given the network size, its navigation must be guided in the sense that, at each step, some algorithm (per the nomenclature from Reference 19 a "scoring function") must evaluate the sets of substrates and allow further expansion of only the most promising ones. It should be noted that for advanced synthetic designs, the scoring scheme may need to be more far-sighted than just one-synthetic-step-at-the-time and should be able to strategize over multiple steps.[22]
iv. Assuming an adequately large collection of reaction rules (e.g., >100,000 expert-coded transforms currently in Chematica), smart and strategizing scoring functions, and large collection of stop-point substrates, a good network

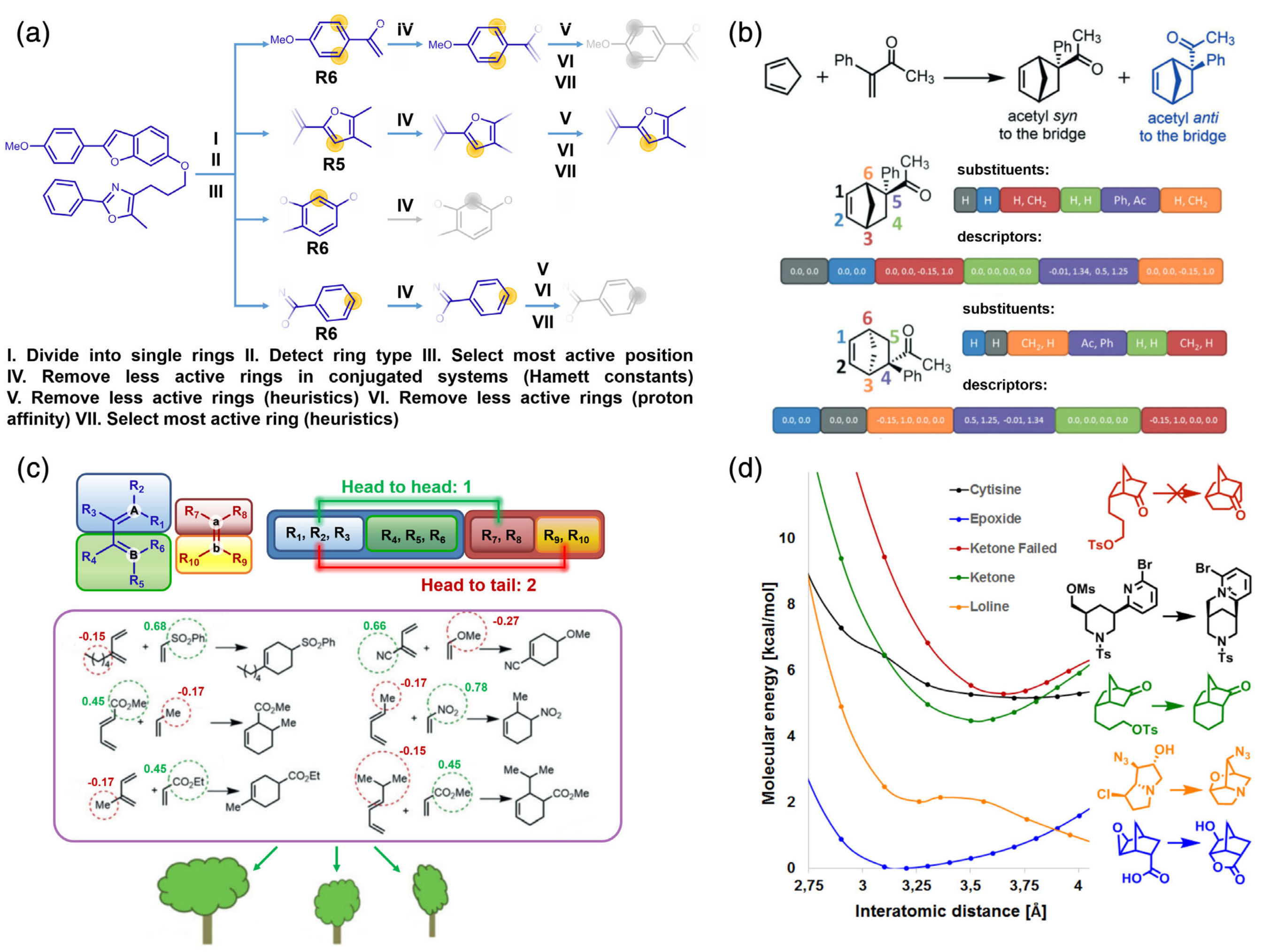

(a)
I. Divide into single rings II. Detect ring type III. Select most active position IV. Remove less active rings in conjugated systems (Hammett constants) V. Remove less active rings (heuristics) VI. Remove less active rings (proton affinity) VII. Select most active ring (heuristics)
(b)
acetyl syn to the bridge
acetyl anti to the bridge
substituents:
descriptors:
(c)
Head to head: 1
Head to tail: 2
(d)
Cytisine
Epoxide
Ketone Failed
Ketone
Loline
Molecular energy [kcal/mol]
Interatomic distance [Å]

FIGURE 2 Examples illustrating some of the methods to fine-tune SMARTS-encoded reaction rules. (a) An algorithm to predict regioselectivity of aromatic electrophilic substitution. First, the molecule is divided into individual rings. Rings are classified as benzene/heterocyclic, fused/single and the most active position within each ring is determined using Hammett constants and heuristics. Then, the algorithm performs three-step filtering to remove the less active rings using additional heuristics, Hammett constants and pre-calculated values derived from proton affinity. For details, see supplementary information to Reference 21. (b) To predict major diastereoisomers in Diels-Alder cycloaddition (with possible *syn* and *anti* products; top), it is necessary to vectorize the substituents—and their stereochemistry—at the reaction core (i.e., C=C–C=C and C=C). Here, this is illustrated for the *syn* product whereby substituents are sequentially added into the vector following counterclockwise numbering. In each block, substituent pointing down the plane is added first. The substituents are then featurized with electronic Hammett constants and steric TSEI indices. To ensure that the classifier does not depend on the order in which substituents are traversed, an equivalent vector (based on the ring rotated 180° around the axis bisecting the 1–2 double bond) is created (bottom). Desired classifiers are then trained on this composite and symmetric representation. (c) Substituent-based vectorization, this time for predicting of regioselectivity of Diels-Alder cycloaddition. Each substituent of the diene (R1-R6) and dienophile (R7-R10) is featurized with Hammett constants (listed outside the red and green circles) and TSEI indices and placed in the vector shown as "block" illustrated on the upper right. Internal sub-blocks represent sets of substituents associated with the "halves" of the molecules. The product is encoded as "1" if molecules react head-to-head and "2" if head-to-tail. With the representations described in (b) and (c), Random Forest classifiers trained on ~3000 literature examples achieved accuracy of ~90% (see Reference 23 from which part of the figure is adapted with permission). (d) Examples of possible (*black*, *green*, *orange*, and *blue*) and impossible (*red*) S_N2-type cyclizations reported in the literature. Energy profiles along approximate cyclization coordinate are calculated with MMFF. The threshold for permissible S_N2-based cyclizations is set below the energy values traced by the red curve (see [20,21])

search algorithm should be able to identify multiple and diverse synthetic plans. Indeed, in our practice with Chematica, it has not been unusual to retrieve hundreds of plausible plans for a given target—in such cases, it is also important to develop algorithms that present such pathways to the program's end user in some ranked order reflecting their realistic costs. As we will see, the algorithms for computing such costs[46] are also network-based but traverse the already-generated network in the "opposite" direction, from stop-point substrates to the target.

3 | GRAPHS, SYNTHETIC POSITIONS, AND HYPERGRAPHS

We first consider how to represent individual chemical reactions in a manner compatible with the creation of a retrosynthetic network. Say, a reaction produces product C from substrates A and B, $A + B \rightarrow C$. In a network/graph notation, A, B, and C will correspond to the "molecule nodes" that need to be connected by the so-called "directed edges." One way to do so is to draw such edges from A to C and from B to C (Figure 3a)—this method, however, is inappropriate as it may suggest that C can be made either from A or from B, whereas in reality it requires both A and B. The way to overcome this problem is to introduce another type of a node, the "reaction node" representing the reaction operation and denoted in Figure 3b by a small diamond. In this so-called bi-partite representation, all causal relationships between substrates and products are preserved—in our particular example, A and B *together* engage in reaction r_1 to yield C. With this notation, our retrosynthetic network can be represented as a directed graph (Figure 3c) of molecule nodes (containing molecular structures stored as canonical SMILES) and reaction nodes (containing details of the reaction operations).

During retrosynthetic planning, this graph grows with every iterative expansion of "current" targets (retrons) into substrates (synthons). In principle, the expansion can continue almost indefinitely but in practice only up to several millions of nodes can be "expanded" and evaluated in realistic times and stored in machine's memory. Still, the graph may rapidly become very complex and may contain, for instance, synthetically unproductive cycles (e.g., in a simple example in Figure 4a, red reaction arrows trace a directed cycle involving molecules "*b*", "*g*", and "*j*"). The existence of cycles is, actually, a realistic problem encountered in several retrosynthetic platforms"—e.g., in Figure 4b, one such cycle in highlighted within a simple synthetic plan produced by MIT's MCTS-based ASKCOS planner.[49] Yet another problem to consider in the context of synthetic graphs is that the synthetic distance—measured by the number of reactions between any given molecules—may depend on the synthetic route. For instance, in Figure 4a, distance from the target "*a*" to molecule "*d*" is just one reaction step (*violet* arrows) if used as part of a three-component reaction involving also "*e*" and "*f*", but three steps (*brown* arrows) if used in a pathway though the "*i*" and "c" intermediates. Such nonuniqueness is problematic in assigning "synthesizability" scores to the nodes.

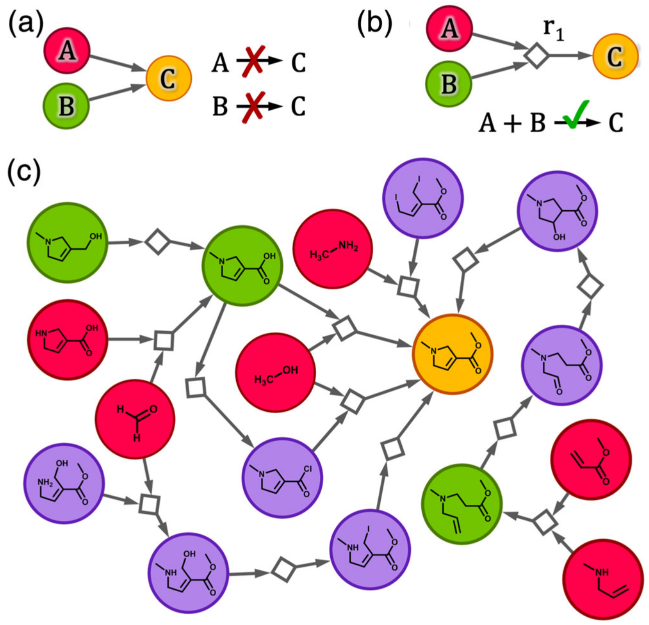


FIGURE 3 Examples of synthesis graphs. A simple $A + B \rightarrow C$ reaction represented as (a) a conventional graph and (b) a bi-partite graph. In the latter representation, there are two types of nodes (larger circular nodes correspond to molecules, smaller diamond nodes to reactions operations, here just one reaction node r_1). (c) A larger bipartite graph describing several methods of making a substituted 3-pyrroline target. *Yellow* = target; *red* = commercially available substrates; *green* = molecules already reported in the literature; *violet* - molecules unknown in the literature. Here, the coloring is just for illustration but reflects the color scheme used in realistic searches in Chematica (see Figures 11 and 12).

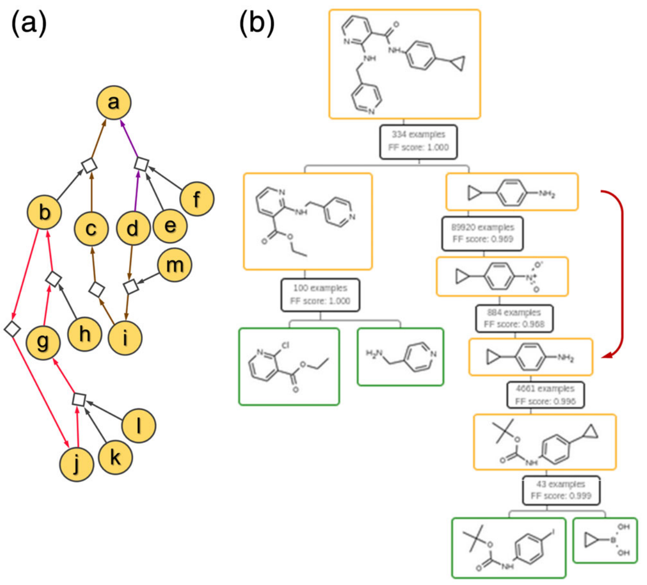

(a)
(b)
334 examples FF score: 1.000
89920 examples FF score: 0.969
100 examples FF score: 1.000
884 examples FF score: 0.968
4661 examples FF score: 0.996
43 examples FF score: 0.999

FIGURE 4 (a) An example illustrating problems in uniquely defining synthetic distance within a retrosynthetic graph and the existence of synthetically unproductive cycles (see main text for details). (b) A screenshot of a synthetic plan produced by MIT's ASCKOS program[49] to a relatively simple target (an analogue of Apatinib). A cycle present in this route is denoted by the red arrow. Panel (a) adapted with permission from Reference 21 (Copyright 2018 Elsevier)

Such considerations made us rethink the representation based on individual molecule nodes and led us to the concept of a "synthetic position."[19] In chess, the position is defined by the placement of *all* pieces on the chessboard. By analogy, in synthetic planning, *all* substrates of a given reaction need to be considered in order to evaluate whether this "retrosynthetic move" is promising—if only one substrate is "promising" but another is evidently hard to make or even structurally impossible, there is no point in seeking the syntheses of the former. What this means is that the "synthetic positions" we wish to evaluate should encompass *sets* of all substrates generated in a given reaction, and that the network of retrosynthetic options should be a directed "hypergraph" of such "hypernode" sets. The semantics of the hypergraph is illustrated in the right portion of Figure 5: The nodes of the hypergraph correspond to stages of synthesis, and edges correspond to reactions that transform one set of substances into another set that may be obtained via a one-step reaction operation. The problem of retrosynthesis therefore reduces to the *optimal path problem* widely studied in computational graph theory and here aimed at finding some "optimal" synthetic route leading from the singleton set of the target to any terminal nodes comprised of readily available starting materials.

4 | BASIC SCORING FUNCTIONS AND STOP POINTS

It should be emphasized that searching for the "optimal" synthetic pathway does not mean simply searching for the shortest route. In real-world synthetic design, many factors need to be taken into consideration—for instance, it may be preferable to perform an extra reaction but, by doing so, reach less expensive substrates; or one reaction may be preferable on account of being less technically cumbersome than some other type of chemistry, etc. In general, when scoring the synthetic positions, it is necessary to evaluate both the molecules the algorithm encounters and the reactions that yield them. In this spirit, Chematica has been using a scoring function that is intended to estimate and minimize the "cost" of synthesis in terms of both chemicals and reactions. This function is a sum of two major components, each based on a combination of chemically meaningful variables such as those summarized in Table 1:

4.1 | The Chemicals' scoring function

CSF evaluates substrate sets/synthetic positions leading to a given product/retron and promotes those sets that are composed of the simplest molecules. For instance, variable *RINGS* sums the rings while variable *STEREO* sums the

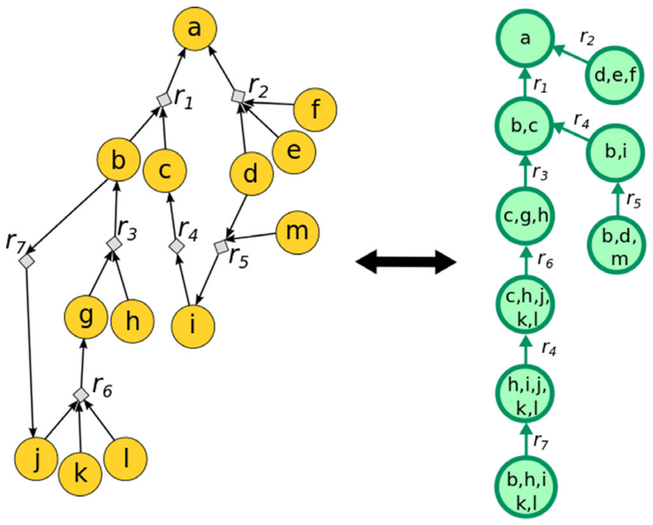


FIGURE 5 Scheme of a synthesis graph (*left*) and (a part of) the corresponding synthesis hypergraph (*right*). The labels near hypergraph edges indicate the corresponding reaction nodes in the synthesis graph. Note that more than one hypergraph edge might be related to a single reaction from the synthesis graph (as is the case here for reaction r_4). Figure reproduced with permission from Reference 21 (Copyright 2018 Elsevier)

TABLE 1 Key variables available in Chematica/Synthia to define CSF (*green*) and RSF (*red*) scoring functions

Variable	Description
SMALLER$^{\gamma}$	Summed lengths of the SMILES strings of all substrates, each raised to some user-specified power γ
HOOD$^{\gamma}$	More advanced version of the SMALLER variable. HOOD counts certain functional groups (whose atoms are no to be disconnected) within the SMILES as one atom (e.g., *O*-tosyl group is counted as a single 'O')
STEREO	Number of stereocenters in the substrates
RINGS	Number of rings in substrates
CONFLICTS	+1 of extra cost for each functional group incompatible with reaction conditions. It is recommended to use this variable with a very high coefficient
PROTECTIONS	+1 of extra cost for each functional group that needs to be protected
NONSELECTIVITY	+1 of extra cost for each reaction having multiple reaction sites on a molecule. It is recommended to use this variable with a very high coefficient
FILTERS	+1 of extra cost for each reaction in a sequence in which a fragile functional group is carried along the synthetic pathway. It is recommended to use this variable with very high coefficient
HIDE SMILES, HIDE SMARTS, HIDE NAME	Variables to exclude certain molecules, molecular patterns, or keywords (i.e., keywords in reaction names or reaction condition records)
TUNNEL_COEF	A coefficient between 0 and 1 that reduces cost of c.a. 100 special, two- or three-step reaction sequences transforming highly reactive functional groups into less reactive ones (e.g., a sequence of reactions from an alkyl iodide to a silyl ether; see Figure 8b)
FGI_COEF	A coefficient between 0 and 1 that reduces cost of reaction sequences that do not decrease molecular complexity but help find "bypasses" around reactivity conflicts; see Figure 8c
STEP	Variable increasing cost of reaction rules that are combination of several individual steps. For instance, SAMP/RAMP alkylation entails three individuals steps (installation of chiral auxiliary, alkylation and removal of chiral auxiliary) and so the cost increases thrice

stereocenters present in all molecules of a given substrate set (i.e., the hypergraph's node). Comparing different substrate sets leading to the same product, the one minimizing these variables is preferred—in other words, the *RINGS* and *STEREO* variables will promote substrate sets that, when all or some of their contents are reacted, increase the number of rings and/or stereocenters the most. Variable *SMALLER*, on the other hand, takes as argument the lengths of the *SMILES* ("*SMILES_LEN*") of all molecules in a given substrate set, raises them to some power $n > 1$, and then sums them up, $SMALLER = \sum_{\text{substrates}} SMILES\_LEN_i^n$. In particular, for a given number of substrates, *SMALLER* assumes minimal value when the disconnection is, per E.J. Corey's nomenclature,[50] "central" and into like-sized fragments. The preference for such central disconnections increases with increasing exponent n (e.g., if a retron is disconnected into two like-sized synthons, CSF will roughly scale as $1/2^n + 1/2^n$ which is equal to 0.5 for $n = 2$ and only 0.25 for $n = 3$). When different putative reactions disconnect the molecule into different numbers of fragments, multicomponent reactions are generally preferred (e.g., dividing into two vs. three like-sized fragments will translate into the *SMALLER* values scaling, respectively, as $\sim 1/2^n + 1/2^n$ vs. $\sim 1/3^n + 1/3^n + 1/3^n$, where the latter is smaller for $n > 1$). In addition, CSF algorithm is accompanied by routines that check if the substrate molecules contain any of the few hundred hard-coded structurally forbidden motifs, e.g., small rings with unsaturated bonds, systems with double bonds at bridgehead atoms, or highly strained bicyclic systems such as *trans*-cyclohexene oxide. Such molecules are eliminated from consideration. In a broader context, the CSF strives to approximate the "cost" of making a molecule (i.e., the real cost of substrates to which the algorithm is ultimately expected to navigate) and corresponds to the heuristic cost function of the vintage A*-type algorithms.[51] Its role it to guide the search toward less complex molecules, thus avoiding exponential branching and searching parts of the unpromising regions of the solution space.

4.2 | Reaction scoring function

In its basic variant, the Reaction Scoring Function, RSF, is calculated for each reaction performed by the program and approximates the "cost" or difficulty of a particular reaction operation. Such assessments are possible because each of the expert-coded reaction rules in *Chematica* is accompanied by extensive lists of functional groups that, based on mechanistic considerations, are judged incompatible with a given reaction or groups that need to be protected before the reaction proceeds—as mentioned in the introduction, this is one of the key advantages of expert-coded rules over machine-extracted ones, because it considers impossible reactions that are virtually never reported in the literature. For example, an imine group present in any of the substrates is strictly incompatible with aldehyde reduction reaction because it would be reduced to an amine; an alcohol or phenol may be used in a Grignard-type reaction but only if it is first protected ("masked"), etc. To quantify such chemical "conditionals" and in addition to a certain standard cost of performing a reaction operation, RSF's variable *CONFLICTS* adds +1 to every incompatible group detected. When *CONFLICTS* is multiplied by some very high coefficient, it effectively removes any conflict-bearing reactions from consideration. Variable *NONSELECTIVITY* uses a similar scheme to penalize reactions that can proceed in more than one place of the same molecule, in effect decreasing the yield of reaction at only the desired location—typically, this variable is also used with highly penalizing coefficient, but its value sometimes can be lowered to allow for more synthetically "creative" solutions (cf. Sarpong's analyses in Reference 36) assuming that some nonselectivity problems can be circumvented by judicious choice of reagents and reaction conditions. Variable *FILTERS* is another variable with high-cost coefficient as it serves to penalize steps that drag along fragile functional groups, instead of reacting them immediately after formation. In contrast, variable *PROTECTIONS* (which adds +1 for every group that needs protection) is typically multiplied by a moderately valued coefficient and therefore assigns a smaller penalty/cost to reactions that require protection, in effect preferring—but not mandating—those that are protection-free (naturally, a very high value of the coefficient will remove such reactions altogether and the algorithm will then seek only the protection-free routes which are desirable[52] but not always possible). Algorithmically, the RSF mirrors the edge cost function of "classical" graph search algorithms such as A*[51] or Dijkstra.[53]

In *Chematica*, the scoring functions can be flexibly defined in a calculator-like panel as algebraic combinations of variables discussed above and in Table 1, and with weights reflecting the user's preferences. Irrespective of these functions' specifics, however, the search routines (cf. next section) will always seek syntheses that minimize the overall cost prescribed by the sum of CSF and RSF. As a rule of thumb, giving higher weight to the RSF will increase the cost of performing reactions relative to the cost of chemicals, resulting in the algorithm seeking shorter routes even at the expense of more expensive substrates.

As an example, a scoring function we have commonly used, $FUNCTION_1 = CSF_1 + RSF_1 = SMALLER^3 + 60 + 120 * PROTECTION + 1{,}000{,}000 * (CONFLICTS + NON\_SELECTIVITY + FILTERS)$ will strongly prefer central disconnections (SMALLER to the cube power), and eliminate any reactions suffering from cross-reactivity conflicts, nonselectivity issues, or those in which chemically unstable groups are dragged along (all such reactions are heavily penalized by the 1,000,000 coefficient). If reactions entail no such problems, this function will add an only moderate cost of performing a reaction operation (60 and, if a reaction requires protection, 120). In effect, this function will be able to seek longer routes starting from inexpensive substrates. By contrast, $FUNCTION_2 = CSF_1 + RSF_2 = SMALLER^3 + 60,000 + 120,000 * PROTECTION + 1,000,000 * (CONFLICTS + NON\_SELECTIVITY + FILTERS)$ will heavily promote the shortest routes on account of the 60,000 and 120,000 RSF costs of performing any problem-free reactions (such routes will stop at virtually *any* available substrates, even the ones that are very expensive).

On the last note, the stop-point substrates can be of two kinds—they can be chemicals that are either commercially available (in default version, from Sigma-Aldrich catalog although other catalogs can easily be configured) or those whose synthesis (or syntheses) is known. In the first case, the molecule comes with an actual catalog price rescaled to dollars per unit quantity and the user can specify that he/she is interested in stopping only at molecules that are below a certain threshold price. In the second case, the molecules are characterized by the so-called synthetic popularity measure[19,54,55] which quantifies the numbers of literature-reported reactions in which a given molecule has been used. The user can then select as stop points only the molecules for which the synthetic popularity measures are high, the rationale being that if such molecules are not in the Sigma-Aldrich catalog, they are widely used and can be readily synthesized. As could be expected, for molecules for which both measures are available, the price correlates with synthetic popularity—what this means is that the algorithm can stop at some commercially and some synthetically popular substrates, and the latter measures can be converted, at least approximately, to dollars per unit quantity.

5 | BASIC SEARCH ALGORITHM

Let us first illustrate the basic version of the A* search algorithm, one that Chematica used until ca. 2017/2018 and originally described in the supplementary information to[21] (see also the pseudocode of a sequential version of this algorithm in Figure 6). Starting from a desired target, the exploration of the synthesis graph and its induced hypergraph is guided by the constantly updated CSF + RSF sums. The hypergraphs' vertices (i.e., sets of currently available substances that further need to be synthesized) are CSF-RSF-scored and stored in the so-called priority queue, PQ, implemented as a binomial heap[56] (which is a data structure offering high efficiency of insert operations). The individual substances within the most promising (i.e., lowest CSF + RSF score) hypernodes are expanded by the application of all retrosynthetic reaction rules matching a given substance. A single expansion operation produces all synthetic options one reaction away from the molecule being "expanded"; in reality, the process is parallelized and many molecules are expanded simultaneously with the results collected and managed by a multi-processing service. This process is continued iteratively until the search reaches substrates that meet the stop-point criteria. When this happens, a synthesis pathway is established as a hyperpath through the hypergraph (though, of course, it is also retrievable from the synthesis graph).

This general scheme merits several comments. First, to avoid querying the same chemical node for retrosynthetic expansion several times and to know when a molecule's expansion into immediate synthons can be used for hypernode expansion, we keep track of the statuses of all nonterminal molecules, as illustrated in Figure 7 and defined as: *EXPLORED*, when already successfully expanded for possible retrosynthetic steps; *EXPLORATION-IN-PROGRESS*, when the node is being expanded but results have not yet recovered; and *UNEXPLORED*, when the node has not yet been queried for retrosynthetic expansion. In addition, we keep track of the "synthesizability status" of molecules which can be propagated "upwards" in the synthesis graph while the status of any node has changed: *COMPUTED* means that the node already has (at least one) computed synthesis—that is, the molecule is a terminal stop-point of the search (e.g., commercially available) or all substrates necessary to synthesize this molecule via some reaction are *COMPUTED*; *NONSYNTHESIZABLE* means that this node is *EXPLORED* and has no incoming reactions or all of the reactions producing it have some substrates which are *NONSYNTHESIZABLE*; *NOT-COMPUTED* otherwise. Hypernodes containing substrates which are NONSYNTHESIZABLE are not further expanded. Furthermore, keeping track of these statuses also allows us to detect when the algorithm identifies the first completed synthesis—this happens when the target becomes *COMPUTED*.

Second, when the algorithm reaches the stop-point (terminal) chemical nodes, their CSF values are replaced by actual/catalog costs, which are then propagated "upwards" through the graph of *COMPUTED* molecules, gradually replacing the CSFs (which, as we discussed before, serve only as estimates of the real cost during the searches). We will

```
def A_star_search():
  PQ.push(target_node)
  while True:
    node = PQ.pop()
    if node is not None: # if PQ is nonempty
      for sub in node.nonterminal_substrates:
        if not synthesis_graph.is_expanded(sub)
          spider = expand_spider(sub)
          synthesis_graph.add_spider(spider)
        # for decendants of node due to expansion of sub
        for descendant in get_descendant_nodes(node, sub):
          PQ.push(descendant)
    else: # if PQ is empty
       break # terminates the search
```

FIGURE 6 A Python language pseudocode of a sequential version of the basic A* search algorithm. Supplementary information to Reference 22 provides a more detailed pseudocode, describing one-step, retron-to-synthons expansions ("spiders"), and construction of descendant hypernodes of a given hypernode

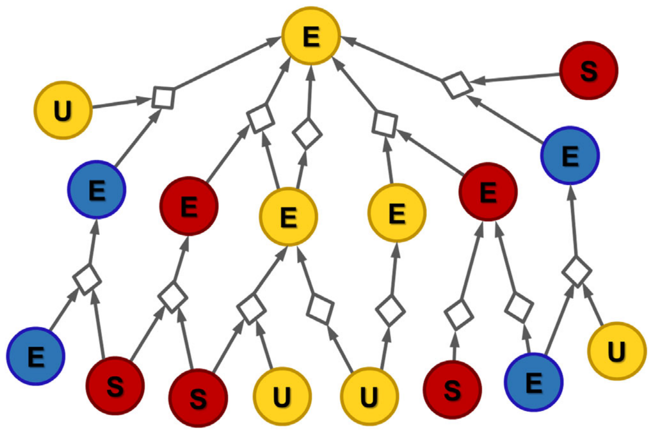


FIGURE 7 A sample synthesis graph illustrating various statuses of molecules updated during search. Nodes marked S denote "terminal" starting materials meeting the stop conditions; other nodes are nonterminal and can have statuses *EXPLORED* = E or *UNEXPLORED* = U. *NONSYNTHESIZABLE* molecules (i.e., ones which are expanded but have no incoming reactions or all reactions producing them have some *NONSYNTHESIZABLE* substrates) are in blue, *COMPUTED* (i.e., terminal or having at least one incoming reaction with all *COMPUTED* substrates) in red and *NOT-COMPUTED* in yellow.

discuss the upwards propagation in detail in Section 7, where this procedure will allow us to rank the pathways, if multiple such pathways are found.

Third, our search protocol avoids becoming stuck in the same regions of the synthetic space—specifically, as it keeps visiting hypernodes sharing the same "history" of expansions (i.e., similar fragments of already examined synthetic routes), penalties are being added to their CSF + RSF. In this way, these synthetic proposals are becoming less and less desirable, ultimately forcing the search to withdraw and explore elsewhere.

6 | MULTISTEP STRATEGIZING, ADVANCED SCORING, AND IMPROVED SEARCH ALGORITHM

With the algorithm and scoring functions described thus far, Chematica was able to construct syntheses to nontrivial drug molecules[21] such as in the example in Figure 1a. However, even upon significant expansion of its reaction knowledge base (to >100,000 expert-coded reaction rules), application of additional variables in the scoring function (e.g., those reflecting Corey's rules on synthesis logic[50]), or the use of state-of-the-art NN-based scoring functions,[40] the program was still not able to find routes to really complex targets, especially natural products, and if it happened to find a route, it was often roundabout and unremarkable.

Analysis of Chematica's failures in such advanced-level syntheses revealed that the program was not able to "think" in an expert-like, farsighted manner—in particular, it did not know when and how to strategize over multiple steps and sometimes pursue steps that, individually, offered no structural simplification but could open-up elegant synthetic possibilities at the later stages of planning. Accordingly, we set out[22] to furnish the program with "multistep logic" such that it would recognize when certain reaction choices imply *succession* (or elimination) of other transformations. Such causal relationships are key for the so-called Artificial Generalized Intelligence,[57,58] AGI, and we hoped that with their inclusion the program could mimic better the synthetic designs by human experts. Taking inspiration from classic total syntheses, we implemented four modalities of multistep strategizing:

(1) We taught Chematica when to pursue two-step sequences in which the first one (in the retrosynthetic direction) increases structural complexity but, by making such an "investment," enables a disconnection offering high degree of structural simplification (Figure 8a). Only few hundreds of such "tactical combinations," TCs, had been known before but we used big-data analysis of all possible reaction combinations to discover additional millions[34]—some 100,000 most useful of these TCs were then included in Chematica, allowing the program to efficiently (and often elegantly) overcome local maxima in structural complexity.

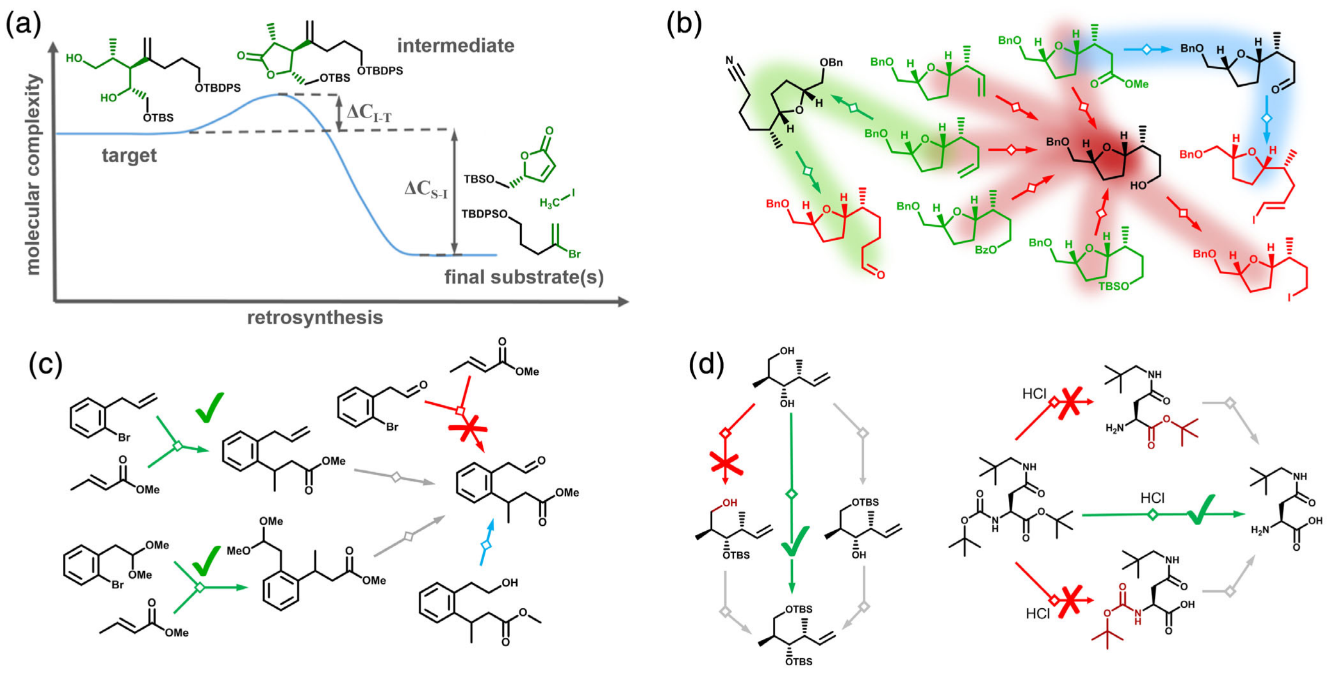


FIGURE 8 Multistep strategizing modalities to facilitate Chematica/Synthia's advanced-level planning. (a) Tactical combinations help navigate over intermediate synthons that are more complex than retrons. Overcoming such local complexity maxima allows the search to reach much simpler, downstream synthons. Here, the structural complexity of the diol target is first increased by introducing an additional ring (transforming the diol into a five-membered lactone). This "uphill" step, however, is not unproductive as it enables a stereoselective conjugate alkenylation-alkylation controlled by the structure of the substrate. This transformation disconnects the complex intermediate into three much simpler substrates, in the process removing two stereocenters. The parts of the molecules common to both reactions in the sequence (reaction core) are colored *green*. (b) Examples of "tunnel" sequences converting molecules (marked *red*) with highly reactive functional groups (here, alkyl and vinyl iodides, an aldehyde) into more stable ones (marked *green*) featuring an ester, an alkene or a silyl ether. Two-step "tunnels" shown here are traced by *red*, *green*, and *blue* reaction arrows over blurred backgrounds. (c) Navigation around reactivity conflicts. When Chematica encounters an elegant, structure-simplifying disconnection (here, Cu-catalyzed conjugated addition; crossed *red* arrow) but cannot execute this particular reaction because of a reactivity conflict, it first tries to perform a functional group interconversion to remove this conflict (here, by exchanging the highly electrophilic aldehyde into a ketal or a terminal alkene (arrows colored *gray*) and only then retries the promising disconnection (*green* arrows). Without taking such bypasses, the program would pursue a much less productive/simplifying step marked by the *blue* arrow. (d) Simultaneous and tandem reactions implemented in Chematica for, inter alia, the management of protecting groups. In the scheme on the left, silylation of both primary and secondary alcohols can be performed in a single step. If performed step-wise, the program would consider the unprotected primary alcohol as a reactivity conflict; sequential primary-then-secondary protection could be possible but inefficient. In the scheme on the right, the removal of *t*-butyl ester and Boc protection from the amine can be achieved simultaneously, in one step. If performed step-wise, the unprotected group would be (correctly) seen as also reactive, and Chematica would penalize such steps. Panel (a) adapted with permission from Reference 34 (Copyright 2020 Elsevier) and panel (b), from Reference 22 (Copyright 2020 Springer Nature)

(2) We implemented into the program a smaller number, ca. 100, of two- or three-step reaction sequences (Figure 8b) commonly employed in total synthesis on natural products.[59] Considered individually, the steps within these "tunnel" sequences appear futile but when taken together, they serve to convert highly reactive groups into more stable ones, overall reducing the numbers of reactivity conflicts and enabling additional synthetic options.

(3) We introduced routines that allowed Chematica to navigate around some types of reactivity conflicts. Previously, the program had no means of pursuing elegant, structure-simplifying options that entailed a cross-reactive group—we now taught the machine to probe "bypass" sequences to first try a functional group interconversion, FGI, and resolve the conflict, and only then retry the promising step (Figure 8c).

(4) Finally, we allowed the program to perform simultaneous and tandem reactions which, under given reaction conditions, could proceed in one rather than multiple reaction steps (Figure 8d). A sub-class of these improvements related to the all-important protection group management and, in particular, ability to perform multiple suitable protections or deprotections in one step (in this case, the protection and deprotection operations were treated explicitly, as actual reactions and not only as "need to protect" flags used in earlier versions of Chematica).

In Chematica, these multistep routines can be activated either as on/off options (e.g., use or do not use TCs) or as additional variables in the RSF, for example, *FGI_COEF* and *TUNNEL_COEF* variables (see Table 1) that take values between 0 and 1, and serve to reduce the cost per step and thus facilitate reactions included in "tunnels" or "bypasses."

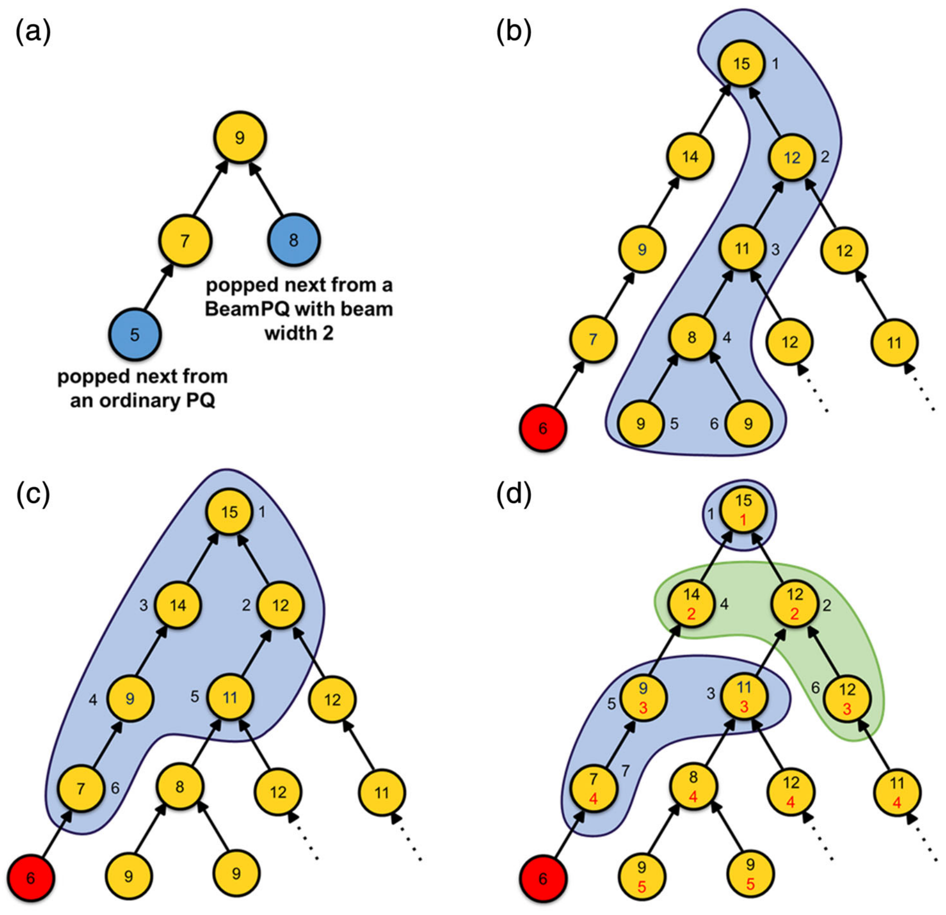

(a)
(b)
(c)
(d)
popped next from a BeamPQ with beam width 2
popped next from an ordinary PQ

FIGURE 9 Sample hypergraphs illustrating operation of a standard A* algorithm versus our modification using BeamPQ. Numbers inside the hypernodes denote their costs used by the queues. In (a), the hypernodes currently stored in a PQ are filled in *blue*. A* would pop from an ordinary PQ a hypernode with score 5, while from a BeamPQ with width $W = 2$, it would pop the node with a higher score, 8, but at lower depth. Panels (b–d) all have the same part of a hypergraph but searched using different methods: (b) a standard A* algorithm, (c) one using our BeamPQ with $W = 2$, (d) one using a BeamPQSet with two BeamPQs with $W = 1$. In each panel, hypernodes that were popped for expansion are on colored backgrounds. Numbers next to the hypernodes specify the order in which these nodes were selected. In (d), the costs used by the two queues for going "deep" and "wide" are shown inside the hypernodes in, respectively, black and red. The hypernodes popped from these queues and returned by BeamPQSet are on *blue* and *green* backgrounds, respectively. In (b–d), the only terminal node is shown in *red*

With these additions, a typical scoring function might look like $FUNCTION_3 = CSF_1 + RSF_3 = SMALLER^3 + 60 * TUNNEL\_COEF * FGI\_COEF * STEP + 1{,}000{,}000 * (CONFLICTS + NON\_SELECTIVITY + FILTERS)$ where the second term lowers the cost of reaction operation, 60, for "tunnels" and FGI "bypasses" (but increases it using the *STEP* variable if a reaction operation entails multiple sub-steps, e.g., SAMP/RAMP alkylation entails three individuals steps, installation of chiral auxiliary, alkylation, and removal of chiral auxiliary).

With the inclusion of multistep strategies, the program performed better but with the basic search algorithm described earlier, the searches were still prohibitively long (often timing out) and also not predictable in terms of the scoring functions to be used. In particular, we noticed that for some complex targets, CSF functions such as $SMALLER^3$—forcing more central disconnections and trying to reach simple substrates as soon as possible—were performing well but in some other cases, they also timed-out without finding any routes. In such failed searches, the algorithm rapidly decreased the CSF to reach molecules that could no longer be easily simplified—it then became stuck in some unpromising areas of the search space, typically at large search depths. These problems could sometimes be remedied by scoring functions allowing for more peripheral cuts and thus "broader" rather than "deeper" searches (e.g., $CSF = SMALLER^{1.5}$) but the times to find the solution were prohibitively long—the search depth increased too slowly for some longer pathways to be found on reasonable timescales.

Our first approach to improve performance was to force the algorithm to explore the search space more broadly, even when "deep"-searching CSF functions (like $SMALLER^3$) were used. To achieve this, we implemented a novel, beam-search-inspired priority queue (BeamPQ; for pseudocode, see SI to[22]) replacing standard PQ in the simple A* algorithm. As we have seen in Section 5, selecting a hypernode for expansion from a standard PQ always returned the lowest-scoring/"cheapest" hypernode stored in it. By contrast, BeamPQ, ensures that before the cheapest hypernode is

```
class BeamPQSet:
  """A queue using several BeamPQs with different scoring functions."""
  def __init__(self, node_scoring_functions, beam_width):
    # A list of functions for computing scores of nodes.
    self.node_scoring_functions = node_scoring_functions
    self.num_queues = len(self.node_scoring_functions)
    # the internal BeamPQs
    self.queues = [BeamPQ(beam_width=beam_width)  for i in range(self.num_queues)]
    # The index of the queue previously popped from, initially -1.
    self.i = -1
    # A set of pairs of form: (a tuple of nonterminal substrates, tuple of scores)
    # for all the already popped nodes.
    self.popped_subs_scores = set()

  def push(self, node):
    """Pushes the node and computed scores onto all of the internal queues."""
    for i in range(self.num_queues):
      self.queues[i].push(self.node_scoring_functions[i](node))

  def pop_from_single_queue(self, q):
    """Pops from q until a node with nonterminal substrates and
    node.scores not present in self.popped_subs_scores is found, in which case
    self.popped_subs_scores is updated and the node and score are returned or
    the queue becomes empty and None is returned."""

  def pop(self):
    """Pops and returns node from the next nonempty internal queue."""
    num_empty_queues = 0
    while num_empty_queues < self.num_queues:
      # Computes the index of the next queue to pop from.
      self.i = (self.i + 1) % self.num_queues
      node = self.pop_from_single_queue(self.queues[self.i])
      if node is not None:
        return node
      else: # the queue is empty
        num_empty_queues += 1
    return None # all self.queues were empty
```

FIGURE 10 A Python language pseudocode of a BeamPQSet priority queue that uses internally several BeamPQs. For details on the implementation of a BeamPQ priority queue, see the pseudocodes in the supplementary information to Reference 22

selected, a given number (called beam width, W) of the best-scoring hypernodes at each lower depth must be already tried (Figure 9a). In more detail, BeamPQ internally keeps a standard PQ, but also remembers W cheapest hypernodes pushed onto it at all search depths; it also keeps track of hypernodes that have not yet been selected for retrosynthetic expansion. The algorithm first chooses the cheapest hypernode n from the internal PQ. If there are no "remembered," not-yet-selected hypernodes at lower depths than the depth of hypernode n, then n is returned. Otherwise, the cheapest such remembered node m at the lowest depth is returned and n is pushed back onto the internal PQ. Of course, BeamPQ also keeps track of which nodes it has already returned and makes sure that the same node is not returned twice (e.g., once as m and then as n as above). The difference between our basic and BeamPQ algorithms are illustrated in Figures 9b,c using the same hypergraph and the first six hypernodes selected from a queue. In Figure 9b, the basic A* search with an ordinary PQ rapidly descends to higher search depths and, because it never expands the hypenode with nonminimal cost 14 at depth 1, it is unable to find the terminal hypernode (colored *red*). By contrast, in Figure 9c, A* using BeamPQ with width $W = 2$ expands both hypernodes at depth 1 before it proceeds deeper—as a result, it can "overcome" high-cost node 14 and ultimately reach the terminal *red* node.

Although this algorithm gave results chemically better than our basic searches, it was still prone to becoming stuck at higher search depths. In a further effort to improve it and analyze more like a human expert, we wanted it to (i) to simultaneously probe different synthetic options "widely" yet pursue more "deeply" those that appear most promising; and (ii) all along, allow the different search strategies to exchange their "experiences" and learn from each other's results.

We achieved this by using a "composite" priority queue which internally keeps a set of BeamPQs (BeamPQSet), each using hypernode costs computed by a different CSF. When the algorithm selects hypernodes for retrosynthetic expansion (using a pop method according to the pseudocode in Figure 10), it does so from each internal queue in turn—for instance, in case of two queues, BeamPQ_1 and BeamPQ_2, the first selection is done from BeamPQ_1, the second from BeamPQ_2, the third again from BeamPQ_1, and so on. This said, if selection from an internal queue yields a hypernode that has already been selected by another internal queue and returned by the BeamPQSet, the algorithm rejects such a hypernode and tries selecting from the same queue until a hypernode not returned before is found or the queue becomes empty (in which case it tries to select from the next internal queue). Overall, the algorithm can follow several different search strategies defined by the CSFs of the internal queues. Moreover, when a new hypernode (resulting from expansion of its parent hypernode) is added to BeamPQSet, it is added to all of the inner BeamPQs with scores computed by the CSFs for these queues (see the push method of BeamPQSet in Figure 10)—this allows these individual search strategies to share their results.

In a typical scenario, two BeamPQs are used: $\mathrm{BeamPQ}_{\mathrm{wide}}$ with a CSF scoring function preferring wider exploration (e.g., $SMALLER^{1.5}$) and $\mathrm{BeamPQ}_{\mathrm{deep}}$ with a CSF preferring deeper searching (e.g., $SMALLER^{3}$). When used together,

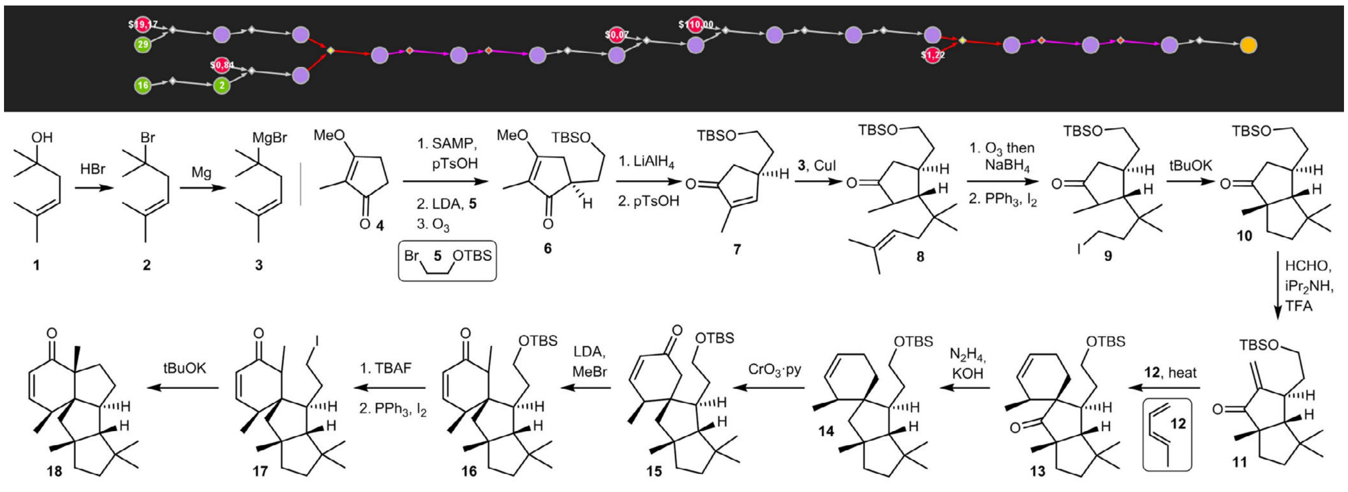


FIGURE 11 Chematica-planned synthesis of conidiogenone B, a complex cyclopiane diterpene featuring a challenging 6-5-5-5 ring system and six contiguous stereocentres (of which three are quaternary). This 14-step synthesis relies on intramolecular alkylations to construct five-membered rings and Diels–Alder cycloaddition to build the six-membered ring. When planning this route, it was essential for Chematica to employ multistep strategies—in the screenshot of the pathway's graph (shown on the top), these strategies are marked by *pink* and *red* reaction arrows. Figure reproduced and caption adapted with permission from Reference 22 (Copyright 2020 Springer Nature)

the first queue helps discover promising beginnings of routes (e.g., hypernodes which make it to the list of W best-scoring ones for $\text{BeamPQ}_{\text{deep}}$ at lower depths) whereas the second strives to "finish" these syntheses off. This is illustrated in Figure 9d where, for simplicity, the widths of both queues are assumed as $W = 1$ (in realistic searches, Chematica uses $W = 200–500$). The first selection (of the root hypernode) is arbitrarily done from $\text{BeamPQ}_{\text{deep}}$, followed by selection from $\text{BeamPQ}_{\text{wide}}$ which assigns the chosen hypernode with the score/cost of 2 (according to $\text{BeamPQ}_{\text{deep}}$, this node has score of 12, as indicated by the number in the upper portion of the node). The next hypernode is from $\text{BeamPQ}_{\text{deep}}$ and has score 11 (3 according to the "wide" search). This is followed by selection from $\text{BeamPQ}_{\text{wide}}$ of a hypernode with score 2 (14 according to the "deep" search). As this hypernode is expanded, and its descendants are pushed onto both queues, the previously cheapest hypernode at search depth 2 (i.e., one previously scored at 11/3) ceases to hold the "record" status and is supplanted by the hypernode for which the $\text{BeamPQ}_{\text{deep}}$ score is equal to 9. This illustrates the cooperation between the two strategies in the sense that a more promising beginning of a pathway sought by $\text{BeamPQ}_{\text{deep}}$ is found at depth 2 thanks to $\text{BeamPQ}_{\text{wide}}$. The next selection is from $\text{BeamPQ}_{\text{deep}}$ and is the hypernode with score 9 (rather than hypernode with lower score 8 but at depth 3). Then, after the hypernode with score 9 is expanded and $\text{BeamPQ}_{\text{wide}}$ returns the hypernode with scores 12/3, the hypernode with cost 7 is selected from $\text{BeamPQ}_{\text{deep}}$, and expanding it leads to a solution being found (red-colored terminal node).

A notable addition to this overall scheme was that in order to increase search diversity, we *limited* the number of retrosynthetic options suggested upon expansion of any molecule node (typically, to ca. 20 top-scoring reactions). While this might sound paradoxical, it is actually easy to rationalize—without such a limit, it could happen that the hundreds of synthetic options emanating from one or few specific retrons gave rise to enough hypernodes to "fill in" the entire width W of a BeamPQ. If this happened, the strategies probed by the program could all share the same few retrons and the diversity of solutions would be poor. By imposing the limit, we made it more likely that the W hypernodes at every search depth comprised structurally diverse molecules, ultimately translating into more diverse syntheses being probed.

With these improvements, the program finally became adept in identifying chemically sound routes to even very complex targets. We documented this in[22] where Chematica—using only its default scoring functions based on the abovementioned "deeper," $SMALLER^{3}$, and "wider," $SMALLER^{1.5}$, CSFs and a standard RSF—designed syntheses to ca. 30 complex natural products, at the forefront of modern synthesis (Figure 11). In addition to validating three total syntheses by experiment (as in the example shown in Figure 1b), we mixed the set of 20 Chematica-designed paths and 20 similar-complexity total syntheses from recent literature. We then asked 18 world's top-level synthetic experts to vote whether a particular path was of human versus machine origin, and also to rank its subjective elegance on a 1–10 scale. The experts' scores were similar for both human and machine syntheses, meaning that Chematica passed a form of a Turing test (for all routes, detailed metrics and results, see Reference 22 and its Supplementary information).

7 | SELECTION AND RANKING OF PATHWAYS

In the said work on natural products, we aimed to mimic the most common uses of Chematica by synthetic chemists whereby only default-scoring functions are applied and only the top-scoring solution is usually examined.[36] However, the stringent criterion of using default settings and analyzing just the best-scoring path does not harness one of the key advantages of computerized synthesis, namely, that the machine can produce *many* valid pathways, differing in the key disconnections and in the overall strategy of assembling the target. In our practice with Chematica, we have routinely seen the program produce tens of diverse and chemically valid plans to mid-complexity targets and often multiple materially different solutions for complex ones (see Figure 12 for examples of alternative syntheses of tacamonidine found by Chematica upon applying a variable penalizing specifically the Pictet-Spengler cyclization central to the route from the Figure 1b). This said, we also observed many roundabout or repetitive solutions which, if not properly scored and ranked, would only serve to pollute the list of useful results. Mindful of such considerations, we implemented[46] algorithms that (i) selects portions of the searched networks that span only the productive syntheses found during a given search; (ii) score and rank theses syntheses according to a metric more realistic than the CSF and RSF functions (which, as we discussed, only roughly approximate the actual cost while searching for routes); (iii) penalize pathways that are similar to each another; and (iv) present to the user a limited number of routes that are both economical and diverse.

In brief, within the post-search reaction network (Figure 13a), we retain only the "synthesizable" molecule nodes (i.e., molecules for which there exists a route connecting them with the stop-point substrates) and "viable" reaction nodes (i.e., reactions for which all substrates are synthesizable) (Figure 13b). We then retain only those nodes that are ancestors of the target and the target itself (Figure 13c). As defined in,[46] the cost of each molecule node within this

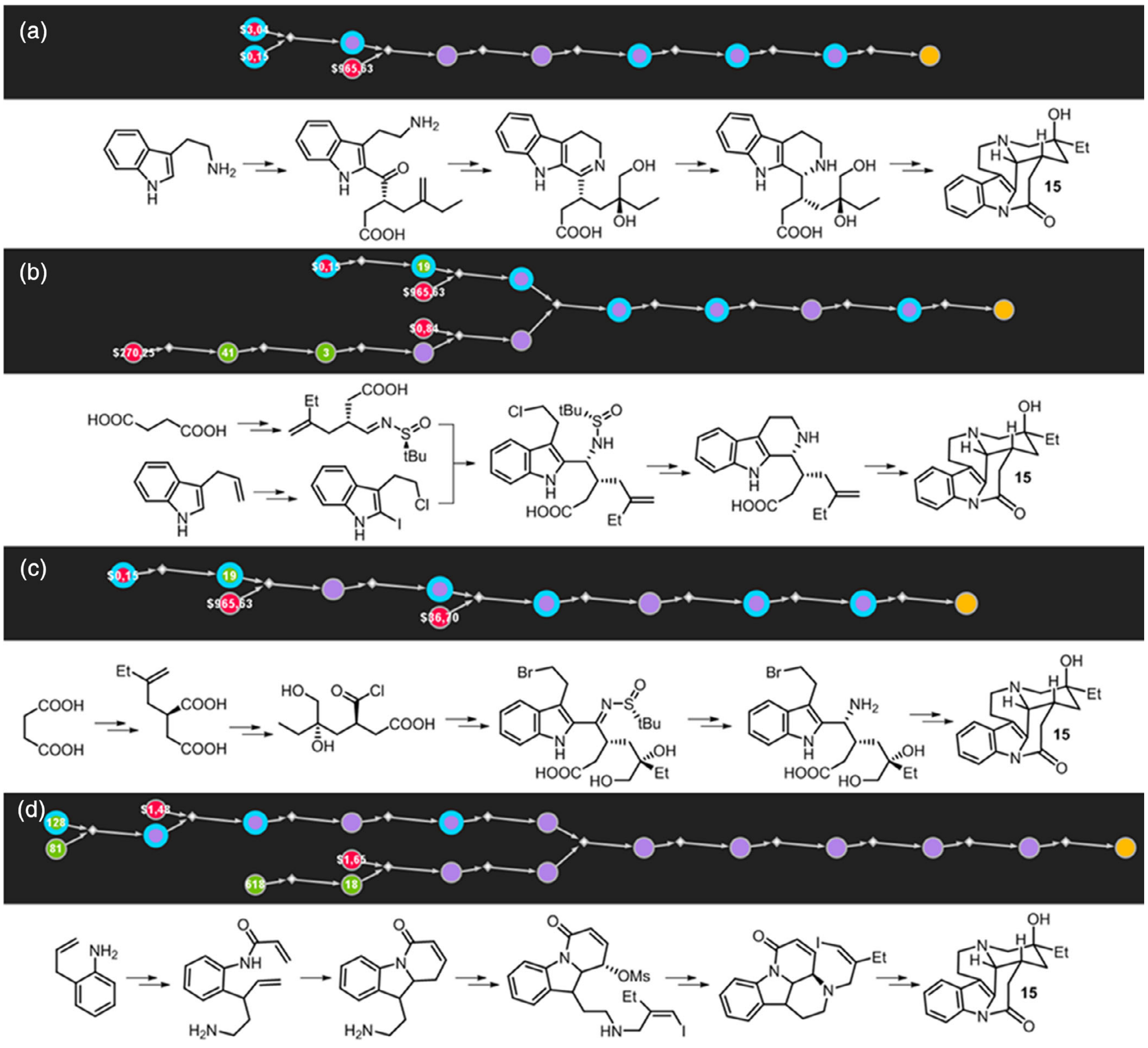


FIGURE 12 Examples of alternative syntheses of tacamonidine proposed by Chematica. Color coding of the nodes is the same as in Figure 1. Synthesis (a) starts with Friedel-Crafts acylation of tryptamine. This is followed by alkylation of the ketone[60] controlled by a chiral hydrazine, Sharpless' dihydroxylation, and reduction of the cyclic imine[61] (subsequent steps assemble the remaining rings). Routes in (b,c) rely on the formation of a chiral amine (controlled by Ellman's sulfonamide) via addition of an organometallic reagent derived from 2-iodoindole or reduction of an acyclic ketimine. We observe that in (b), indole addition to chiral sulfinylimine requires the formation of a dianion, or introduction of *N*-protecting group. In both plans, remaining stereocenters are formed using Evans-type auxiliary to control enolate alkylation[62] and Sharpless' asymmetric dihydroxylation. Synthesis in (d) starts from 2-allylaniline. The 6-5-6 ring system is formed via a copper-catalyzed carboamination of the alkene.[63] The key step is an intramolecular conjugate addition[64] followed by late-stage indoline and alkene oxidations. For pathway details and discussion, see section 3.12 of the supplementary information to Reference 22 from which the figure is reproduced and caption adapted by permission (Copyright 2020 Springer Nature)

"solutions graph" is the smallest cost of all syntheses that can make some desired quantity of this chemical; the cost of a reaction node is the smallest cost of all synthetic routes containing this particular reaction and giving this reaction's product at a desired scale.

Following these definitions, for each nonstarting-material chemical, *c*, in the network, its cost is calculated recursively as $cost(c) = min_{r \in pred(c)}(cost(r))$, while for each reaction node, $cost(r) = fixed\_cost(r) + \sum_{c \in pred(r)} \frac{cost(c)}{yield(r)}$, where

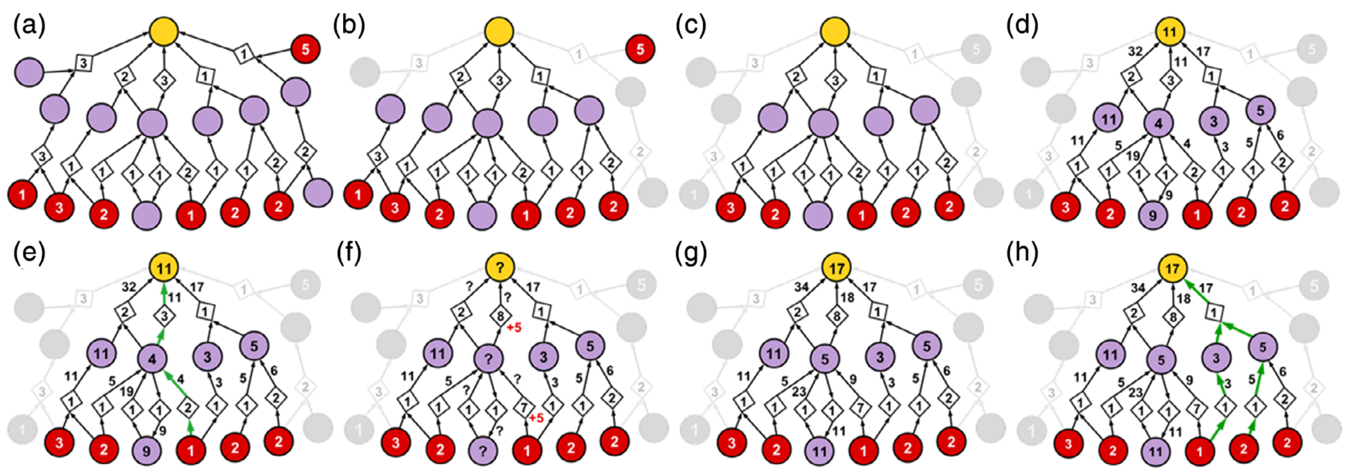


FIGURE 13 Selection of cost-effective yet diverse pathways from a graph explored during synthetic search. For the sake of illustration, we assume here 50% yield for all reactions. (a) The entire postsearch reaction network. *Yellow* = target, *red* = commercially available substrates with numbers denoting hypothetical costs converted to "per mmol." Costs of reaction operations are indicated in the diamond-shaped reaction nodes. Some of the space probed by the search algorithm involves nodes that did not ultimately lead to complete syntheses of the target (in reality, the majority of the network is of this type). In (b,c) such molecule and reaction nodes are removed. In (d), the costs of nodes in the remaining "solutions" subgraph are recursively propagated from starting materials to the target. In (e), the lowest-cost synthesis (*green*) is found and in (f) its reactions are penalized by $P = +5$ (*red* numbers) to increase diversity of additional solutions. Question marks denote nodes whose costs increase upon addition of the penalty. Costs of these nodes are recalculated as in (g). In panel (h), the new "best" synthesis is identified. Additional routes can subsequently be found by repeating the penalization-selection cycle. Figure reproduced and caption adapted from Reference 46 with permission (Copyright 2019 Royal Society of Chemistry)

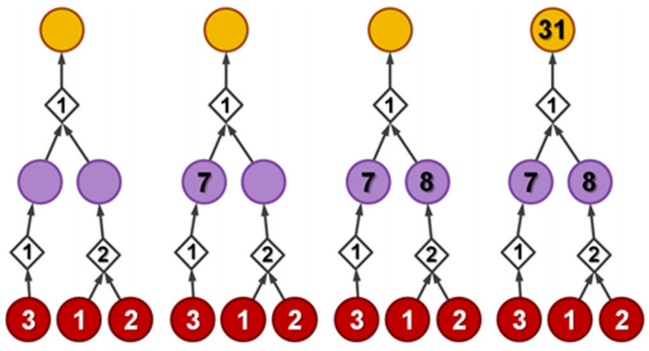


FIGURE 14 An example illustrating recursive propagation of costs (assuming, for simplicity, all yields to be 50%). *White* numbers inside the *red* molecule nodes denote costs of these starting materials (per mmol). Numbers inside diamond-shaped reaction nodes specify fixed costs of synthetic operations at unit scales. Figure reproduced and caption adapted from Reference 46 with permission (Copyright 2019 Royal Society of Chemistry).

fixed_cost(r) is the cost of performing synthetic operations at some unit scale, *pred* denotes the set of predecessors of a given node in the network, and *yield*(r) denotes estimated reaction yield (of note, for diverse reaction types, the yield values are hard to predict accurately by either ML or thermodynamic models[65–67] and in the context of route selection we consider here, it is more practical to use some average values, like, the 75–80% average yield of all organic reactions reported in the literature[67]).

The working of the recursive formula is illustrated by the example in Figure 14. Assuming for simplicity that all reactions therein have the same yield (say, 50%), we first consider the (left) branch of the network starting from a substrate whose cost per mmol is 3. This cost is propagated upwards, to the violet-node intermediate as $1 + 3/50\% = 7$, meaning that in order to make 1 mmol of this intermediate, it is necessary to use 2 mmol of the starting material and pay the cost of 1 for performing the reaction (reaction costs are shown inside the diamond-shaped reaction nodes). For the right branch of the network, the two substrates cost, respectively, 1 and 2 per mmol, and the unitary cost of reacting them is 2. Accordingly, making 1 mmol of the violet-node intermediate will cost $2 + (1 + 2)/50\% = 8$. Next, the costs of the two intermediates thus calculated are propagated further upwards, to derive the cost of making 1 mmol of the target, $1 + (7+8)/50\% = 31$. It is easy to confirm that the result of this recursive calculation is the same as for traditional chemical balancing, whereby to make 1 mmol of target, 4 mmols of each substrate need to be used (chemicals' cost $4 \cdot 3 + 4 \cdot 2 + 4 \cdot 1 = 24$), and the two reactions using the substrates are performed at double the scale of the final reaction yielding the target (reactions' cost $2 \cdot 1 + 2 \cdot 2 + 1 \cdot 1 = 7$).

Returning to Figure 13d, once the costs of all nodes are recursively propagated, the least expensive synthetic path is identified by tracing from the target to the substrates and, at each depth, choosing the lowest-cost reaction (e.g., 11 followed by four in Figure 10e). Although the graph can now be re-searched to return *n* best-scoring pathways, this approach does not ensure the chemical diversity among the selected routes. To enforce such diversity, the algorithm iterates the following loop until it identifies a desired number of routes (or until no more syntheses can be found):

(i) Add penalty *P* (e.g., +5 in Figure 13f) to the fixed costs of each reaction from the most-recently found synthesis;
(ii) Identify all molecule and reaction nodes that are affected by the penalization (nodes marked by question marks in Figure 13f);
(iii) Recalculate the costs of all affected nodes (Figure 13g)
(iv) Select the newest lowest-scoring route and (Figure 13h) and go back to (i).

Examples of pathways found with and without the diversification routine can be found in.[46] Therein, the reader will also find additional details of the algorithm (e.g., avoidance of cycles, some limiting behaviors) as well as performance metrics. On the latter topic, we note that the times to select a given number of ranked syntheses scale approximately linearly with the size of the synthesis graph. Even for large graphs and on a standard machine (e.g., one with 2.5 GHz AMD Opteron 6380 processors), these times are on the order of seconds, which is insignificant compared to the times of synthesis searches (minutes to hours).

8 | SUMMARY AND OUTLOOK

The selection and diversification routines complete the portfolio of graph-based algorithms we have used in Chematica/Synthia's advanced-level synthesis planning. Development of these algorithms has spanned over a decade of research and has, to a large extent, followed a cycle of algorithm development and testing in increasingly more complex syntheses. The solution thus created rests on many pillars, each of which is essential for Chematica's synthetic prowess—the conflict-aware reaction rules, the MM, QM, and ML routines fine-tuning applicability of rules, the graphs, hypergraphs and scoring functions to represent and evaluate synthetic positions, the multistep routines to plan further than just one step ahead, the network searches querying the space of synthetic solutions according to multiple search strategies, and the selection-diversification algorithms.

With this "hybrid" approach, computers have finally achieved the level of human synthesis experts. This said, we do recognize the need (and opportunity) for additional improvements, though mostly in some technical aspects—for instance, further (and ongoing) augmentation of the reaction rule set, acceleration of routines that examine stereochemistry of complex molecules (which has been the slow component of Chematica's searches), or further fine-tuning of reaction rules with routines to rapidly evaluate transition states (e.g., to eliminate unfeasible cyclizations[36]). In a broader context, we hope the solutions we described here will also resonate with colleagues developing purely data-driven, ML retrosynthesis algorithms. We anticipate that Chematica-like search and scoring schemes could help improve the performance of these programs and allow them to tackle targets more complex than currently. This focus on complex targets is important. Indeed, our experience tells us that mostly complex targets are of interest to the demanding audience of synthesis experts who are unlikely to use any retrosynthetic planners unless they offer help them with advanced—and not the trivial—problems.

AUTHOR CONTRIBUTIONS

Bartosz Grzybowski: Conceptualization (equal); formal analysis (equal); funding acquisition (lead); investigation (equal); project administration (lead); supervision (equal); validation (equal); writing – original draft (lead). **Tomasz Badowski:** Conceptualization (equal); formal analysis (equal); investigation (equal); writing – original draft (supporting). **Karol Molga:** Conceptualization (equal); data curation (equal); formal analysis (equal); investigation (equal); writing – review and editing (supporting). **Sara Szymkuć:** Conceptualization (equal); data curation (equal); investigation (equal); writing – review and editing (supporting).

FUNDING INFORMATION

Development of Chematica was partly supported by US DARPA under the Make-It Award, 69461-CH-DRP #W911NF1610384. B.A.G. acknowledges personal support from the Institute for Basic Science Korea, project code IBS-R020-D1.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

ORCID

Bartosz A. Grzybowski https://orcid.org/0000-0001-6613-4261

REFERENCES

1. Corey EJ, Wipke WT. Computer-assisted design of complex organic syntheses. Science. 1969;166(3902):178–92. https://doi.org/10.1126/science.166.3902.178
2. Corey EJ, Wipke WT, Cramer RD, Howe WJ. Computer-assisted synthetic analysis. Facile man-machine communication of chemical structure by interactive computer graphics. J Am Chem Soc. 1972;94(2):421–30. https://doi.org/10.1021/ja00757a020
3. Gelernter HL, Sanders AF, Larsen DL, Agarwal KK, Boivie RH, Spritzer GA, et al. Empirical explorations of SYNCHEM. Science. 1977; 197(4308):1041–9. https://doi.org/10.1126/science.197.4308.1041
4. Hanessian S, Franco J, Larouche B. The psychobiological basis of heuristic synthesis planning - man, machine and the chiron approach. Pure Appl Chem. 1990;62(10):1887–910. https://doi.org/10.1351/pac199062101887
5. Hendrickson JB. Systematic synthesis design. 6. Yield analysis and convergency. J Am Chem Soc. 1977;99(16):5439–50. https://doi.org/10.1021/ja00458a035
6. Ugi I, Bauer J, Bley K, Dengler A, Dietz A, Fontain E, et al. Computer-assisted solution of chemical problems: the historical development and the present state of the art of a new discipline of chemistry. Angew. Chem. Int. Ed. 1993;32(2):201–27. https://doi.org/10.1002/anie.199302011
7. Wipke WT, Ouchi GI, Krishnan S. Simulation and evaluation of chemical synthesis: SECS—an application of artificial intelligence techniques. Artif Intell. 1978;11(1–2):173–93. https://doi.org/10.1016/0004-3702(78)90016-4
8. Judson P. Knowledge-based expert systems in chemistry: not counting on computers. Theoretical and computational chemistry series. Cambridge, UK: Royal Society of Chemistry; 2009. https://doi.org/10.1039/9781847559807
9. van Rozendaal ELM. Some approaches to the synthesis of taxol and its derivatives: total-synthesis based on a LHASA analysis and semi-synthesis starting from taxine B. Nijmegen, Netherlands. Available from:: Radboud University Nijmegen; 1994. https://repository.ubn.ru.nl/bitstream/handle/2066/30052/mmubn000001_181282496.pdf
10. van Rozendaal ELM, Ott MA, Scheeren HW. A LHASA analysis of taxol. Recl. Trav. Chim. Pays-Bas. 1994;113(5):297–303. https://doi.org/10.1002/recl.19941130507
11. Tanaka A, Kawai T, Takabatake T, Oka N, Okamoto H, Bersohn M. Synthesis of an azaspirane via Birch reduction alkylation prompted by suggestions from a computer program. Tetrahedron Lett. 2006;47(38):6733–7. https://doi.org/10.1016/j.tetlet.2006.07.100
12. Corey E, Long A, Rubenstein S. Computer-assisted analysis in organic synthesis. Science. 1985;228(4698):408–18. https://doi.org/10.1126/science.3838594
13. Bishop KJM, Klajn R, Grzybowski BA. The core and most useful molecules in organic chemistry. Angew Chem Int Ed. 2006;45(32): 5348–54. https://doi.org/10.1002/anie.200600881
14. Fialkowski M, Bishop KJM, Chubukov VA, Campbell CJ, Grzybowski BA. Architecture and evolution of organic chemistry. Angew Chem Int Ed. 2005;44(44):7263–9. https://doi.org/10.1002/anie.200502272
15. Grzybowski BA, Bishop KJM, Kowalczyk B, Wilmer CE. The "wired" universe of organic chemistry. Nat Chem. 2009;1(1):31–6. https://doi.org/10.1038/nchem.136
16. Fuller PE, Gothard CM, Gothard NA, Weckiewicz A, Grzybowski BA. Chemical network algorithms for the risk assessment and management of chemical threats. Angew Chem Int Ed. 2012;51(32):7933–7. https://doi.org/10.1002/anie.201202210
17. Gothard CM, Soh S, Gothard NA, Kowalczyk B, Wei Y, Baytekin B, et al. Rewiring chemistry: algorithmic discovery and experimental validation of one-pot reactions in the network of organic chemistry. Angew Chem Int Ed. 2012;51(32):7922–7. https://doi.org/10.1002/anie.201202155
18. Kowalik M, Gothard CM, Drews AM, Gothard NA, Weckiewicz A, Fuller PE, et al. Parallel optimization of synthetic pathways within the network of organic chemistry. Angew Chem Int Ed. 2012;51(32):7928–32. https://doi.org/10.1002/anie.201202209
19. Szymkuć S, Gajewska EP, Klucznik T, Molga K, Dittwald P, Startek M, et al. Computer-assisted synthetic planning: the end of the beginning. Angew Chem Int Ed. 2016;55(20):5904–37. https://doi.org/10.1002/anie.201506101
20. Molga K, Gajewska EP, Szymkuć S, Grzybowski BA. The logic of translating chemical knowledge into machine-processable forms: a modern playground for physical-organic chemistry. React Chem Eng. 2019a;4(9):1506–21. https://doi.org/10.1039/c9re00076c
21. Klucznik T, Mikulak-Klucznik B, McCormack MP, Lima H, Szymkuć S, Bhowmick M, et al. Efficient syntheses of diverse, medicinally relevant targets planned by computer and executed in the laboratory. Chem. 2018;4(3):522–32. https://doi.org/10.1016/j.chempr.2018.02.002

22. Mikulak-Klucznik B, Gołębiowska P, Bayly AA, Popik O, Klucznik T, Szymkuć S, et al. Computational planning of the synthesis of complex natural products. Nature. 2020;588(7836):83–8. https://doi.org/10.1038/s41586-020-2855-y
23. Beker W, Gajewska EP, Badowski T, Grzybowski BA. Prediction of major regio-, site-, and diastereoisomers in Diels-Alder reactions by using machine-learning: the importance of physically meaningful descriptors. Angew Chem Int Ed. 2019;58(14):4515–9. https://doi.org/10.1002/anie.201806920
24. Moskal M, Beker W, Szymkuć S, Grzybowski BA. Scaffold-directed face selectivity machine-learned from vectors of non-covalent interactions. Angew Chem Int Ed. 2021;60(28):15230–5. https://doi.org/10.1002/anie.202101986
25. CAS SCIFINDERn. (2021). Available from: https://scifinder-n.cas.org
26. Coley CW, Green WH, Jensen KF. Machine learning in computer-aided synthesis planning. Acc Chem Res. 2018;51(5):1281–9. https://doi.org/10.1021/acs.accounts.8b00087
27. Coley CW, Thomas DA, Lummiss JAM, Jaworski JN, Breen CP, Schultz V, et al. A robotic platform for flow synthesis of organic compounds informed by AI planning. Science. 2019;365(6453):eaax1566. https://doi.org/10.1126/science.aax1566
28. Genheden S, Thakkar A, Chadimová V, Reymond J-L, Engkvist O, Bjerrum E. AiZynthFinder: a fast, robust and flexible open-source software for retrosynthetic planning. J Chem. 2020;12(1):70. https://doi.org/10.1186/s13321-020-00472-1
29. Lee AA, Yang Q, Sresht V, Bolgar P, Hou X, Klug-McLeod JL, et al. Molecular transformer unifies reaction prediction and retrosynthesis across pharma chemical space. Chem Commun. 2019;55(81):12152–5. https://doi.org/10.1039/C9CC05122H
30. Reaxys Synthesis Planner. (2021). Available from: https://www.reaxys.com/#/search/synthesis-planner
31. Schwaller P, Petraglia R, Zullo V, Nair VH, Haeuselmann RA, Pisoni R, et al. Predicting retrosynthetic pathways using transformer-based models and a hyper-graph exploration strategy. Chem Sci. 2020;11(12):3316–25. https://doi.org/10.1039/C9SC05704H
32. Segler MHS, Preuss M, Waller MP. Planning chemical syntheses with deep neural networks and symbolic AI. Nature. 2018;555(7698):604–10. https://doi.org/10.1038/nature25978
33. Szymkuć S, Badowski T, Grzybowski BA. Is organic chemistry really growing exponentially? Angew Chem. 2021;133(50):26430–6. https://doi.org/10.1002/ange.202111540
34. Gajewska EP, Szymkuć S, Dittwald P, Startek M, Popik O, Mlynarski J, et al. Algorithmic discovery of tactical combinations for advanced organic syntheses. Chem. 2020;6(1):280–93. https://doi.org/10.1016/j.chempr.2019.11.016
35. Lin Y, Zhang Z, Mahjour B, Wang D, Zhang R, Shim E, et al. Reinforcing the supply chain of umifenovir and other antiviral drugs with retrosynthetic software. Nat Commun. 2021;12(1):7327. https://doi.org/10.1038/s41467-021-27547-3
36. Hardy MA, Nan B, Wiest O, Sarpong R. Strategic elements in computer-assisted retrosynthesis: a case study of the pupukeanane natural products. Tetrahedron. 2022;104:132584. https://doi.org/10.1016/j.tet.2021.132584
37. Matsubara S. Digitization of organic synthesis. How synthetic organic chemists use AI technology. Chem Lett. 2021;50(3):475–81. https://doi.org/10.1246/cl.200802
38. Shen Y, Borowski JE, Hardy MA, Sarpong R, Doyle AG, Cernak T. Automation and computer-assisted planning for chemical synthesis. Nat Rev Methods Primers. 2021;1(1):23. https://doi.org/10.1038/s43586-021-00022-5
39. Williams CM, Dallaston MA. The future of retrosynthesis and synthetic planning: algorithmic, humanistic or the interplay? Austral J Chem. 2021;74(5):291–326. https://doi.org/10.1071/CH20371
40. Badowski T, Gajewska EP, Molga K, Grzybowski BA. Synergy between expert and machine-learning approaches allows for improved retrosynthetic planning. Angew Chem Int Ed. 2020;59(2):725–30. https://doi.org/10.1002/anie.201912083
41. Molga K, Szymkuć S, Grzybowski BA. Chemist ex machina: advanced synthesis planning by computers. Acc Chem Res. 2021;54(5):1094–106. https://doi.org/10.1021/acs.accounts.0c00714
42. Kocsis L, Szepesvári C. Bandit based Monte-Carlo planning. In: Fürnkranz J, Scheffer T, Spiliopoulou M, editors. Machine learning: ECML 2006. Berlin Heidelberg: Springer; 2006. p. 282–93.
43. Barto A, Bradtke S, Singh S. Real-time learning and control using asynchronous dynamic programming. Amherst, MA: University of Massachusetts at Amherst, Department of Computer and Information Science; 1991. p. 1991.
44. Sammut C. Beam search. Encyclopedia of machine learning and data mining. US: Springer; 2017. p. 120. https://doi.org/10.1007/978-1-4899-7687-1_68
45. Genheden S, Bjerrum E. PaRoutes: a framework for benchmarking retrosynthesis route predictions. ChemRxiv. 2022. https://doi.org/10.26434/chemrxiv-2022-wk8c3
46. Badowski T, Molga K, Grzybowski BA. Selection of cost-effective yet chemically diverse pathways from the networks of computer-generated retrosynthetic plans. Chem Sci. 2019;10(17):4640–51. https://doi.org/10.1039/c8sc05611k
47. Molga K, Dittwald P, Grzybowski BA. Computational design of syntheses leading to compound libraries or isotopically labelled targets. Chem Sci. 2019b;10(40):9219–32. https://doi.org/10.1039/c9sc02678a
48. Molga K, Dittwald P, Grzybowski BA. Navigating around patented routes by preserving specific motifs along computer-planned retrosynthetic pathways. Chem. 2019c;5(2):460–73. https://doi.org/10.1016/j.chempr.2018.12.004
49. ASKCOS. Software tools for organic synthesis. Cambridge, MA: MIT Department of Chemical Engineering. (2021). Available from: http://askcos.mit.edu
50. Corey EJ, Cheng X-M. The logic of chemical synthesis. New York, NY: John Wiley & Sons; 1995.
51. Hart P, Nilsson N, Raphael B. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybernet. 1968;4(2):100–7. https://doi.org/10.1109/TSSC.1968.300136

52. Young IS, Baran PS. Protecting-group-free synthesis as an opportunity for invention. Nat Chem. 2009;1(3):193–205. https://doi.org/10.1038/nchem.216
53. Dijkstra EW. A note on two problems in connexion with graphs. Numer Math. 1959;1(1):269–71. https://doi.org/10.1007/BF01386390
54. Kowalczyk B, Bishop KJM, Smoukov SK, Grzybowski BA. Synthetic popularity reflects chemical reactivity. J Phys Org Chem. 2009; 22(9):897–902. https://doi.org/10.1002/poc.1535
55. Soh S, Wei Y, Kowalczyk B, Gothard CM, Baytekin B, Gothard N, et al. Estimating chemical reactivity and cross-influence from collective chemical knowledge. Chem Sci. 2012;3(5):1497–502. https://doi.org/10.1039/c2sc00011c
56. Cormen TH, Leiserson CE, Rivest RL, Stein C. Binomial heaps. Introduction to algorithms. 2nd ed. Cambridge, MA: The MIT Press; 2001. p. 455–75.
57. Bergstein B. What AI still can't do. MIT Technol Rev. 2020;123(2):1–7. Available from:. https://www.technologyreview.com/2020/02/19/868178/what-ai-still-cant-do/
58. Yi, K., Gan, C., Li, Y., Kohli, P., Wu, J., Torralba, A., & Tenenbaum, J. B. (2019). CLEVRER: CoLlision events for video REpresentation and reasoning. Ithaca, NY: arxiv. Available from: http://arxiv.org/abs/1910.01442
59. Serratosa F. Organic chemistry in action: the design of organic synthesis. Amsterdam, The Netherlands: Elsevier; 1996.
60. Job A, Janeck CF, Bettray W, Peters R, Enders D. The SAMP-/RAMP-hydrazone methodology in asymmetric synthesis. Tetrahedron. 2002;58(12):2253–329. https://doi.org/10.1016/S0040-4020(02)00080-7
61. Blank N, Opatz T. Enantioselective synthesis of tetrahydroprotoberberines and bisbenzylisoquinoline alkaloids from a deprotonated α-aminonitrile. J Org Chem. 2011;76(23):9777–84. https://doi.org/10.1021/jo201871c
62. Lelais G, Campo MA, Kopp S, Seebach D. Enantioselective preparation of β2-amino acids with aspartate, glutamate, asparagine, and glutamine side chains. Helv Chim Acta. 2004;87(6):1545–60. https://doi.org/10.1002/hlca.200490142
63. Fuller PH, Chemler SR. Copper(II) carboxylate-promoted intramolecular carboamination of alkenes for the synthesis of polycyclic lactams. Org Lett. 2007;9(26):5477–80. https://doi.org/10.1021/ol702401w
64. Piers E, Harrison CL, Zetina-Rocha C. Intramolecular conjugate addition of alkenyl and aryl functions to enones initiated by lithium−iodine exchange. Org Lett. 2001;3(21):3245–7. https://doi.org/10.1021/ol016288u
65. Emami FS, Vahid A, Wylie EK, Szymkuc S, Dittwald P, Molga K, et al. A priori estimation of organic reaction yields. Angew Chem Int Ed. 2015;54(37):10797–801. https://doi.org/10.1002/anie.201503890
66. Schwaller P, Vaucher AC, Laino T, Reymond J-L. Prediction of chemical reaction yields using deep learning. Mach Learn Sci Technol. 2021;2(1):015016. https://doi.org/10.1088/2632-2153/abc81d
67. Skoraczyński G, Dittwald P, Miasojedow B, Szymkuć S, Gajewska EP, Grzybowski BA, et al. Predicting the outcomes of organic reactions via machine learning: are current descriptors sufficient? Sci Rep. 2017;7(1):3582. https://doi.org/10.1038/s41598-017-02303-0

How to cite this article: Grzybowski BA, Badowski T, Molga K, Szymkuć S. Network search algorithms and scoring functions for advanced-level computerized synthesis planning. WIREs Comput Mol Sci. 2022. e1630. https://doi.org/10.1002/wcms.1630