Reactive Search Optimization: Learning while Optimizing

Roberto Battiti and Mauro Brunato

1 Introduction

The final purpose of Reactive Search Optimization (RSO) is to simplify the life for the final user of optimization. While researchers enjoy designing algorithms, testing alternatives, tuning parameters and choosing solution schemes — in fact this is part of their daily life — the final users' interests are different: solving a problem in the most effective way without requiring a costly adaptation and learning curve.

Reactive Search Optimization has to do with *learning for optimizing*, with the insertion of a machine learning component into a solution process so that algorithm selection, adaptation, integration, are done in an automated way, and a comprehensive solution is delivered to the final user. The diversity of tasks, stochasticity, dynamicity which is intrinsic in real-world tasks can be dealt with in a seamless manner. The interaction with the final user is simplified and made human: no complex technical questions are asked about parameters, but the focus is kept on the problem's detailed characteristics and user preferences. In fact, the user wants to maintain control of the problem definition, including of course hard and soft constraints, preferences, weights. This is the part which cannot be automated, while the user is happy to delegate issues related to algorithm choices and tuning.

Needless to say, studying and designing satisfactory solutions to the above final goal is a long-term enterprise with opportunities for PhD students and researchers of this century, but we feel that the road is clear and that preliminary results of interest abound.

Apart from the above concrete issues related to the final user, Reactive Search Optimization also addresses a scientific issue related to the reproducibility of results and to the objective evaluation of methods. In fact, if an intelligent user is actively

Roberto Battiti
LION Lab, Università di Trento, Italy, e-mail: roberto.battiti@unitn.it

Mauro Brunato
LION Lab, Università di Trento, Italy, e-mail: mauro.brunato@unitn.it

in the loop between a parametric algorithm and the solution of a problem, judging about an algorithm in isolation from its user — in some cases its creator — becomes difficult if not impossible. Are the obtained results a merit of the algorithm or a merit of its intelligent user? In some cases the second case holds, which explains why even some naïve and simplistic techniques can obtain results of interest if adopted by a motivated person, not to say by a researcher in love with his pet algorithm and under pressure to get something published.

Now that the long-term vision is given, let's come to a more detailed definition.

Reactive Search Optimization (RSO) advocates the integration of machine learning techniques into search heuristics for solving complex optimization problems. The word *reactive* hints at a ready response to events while alternative solutions are tested, through an internal online feedback loop for the self-tuning of critical parameters. Its strength lies in the introduction of high-level skills often associated to the human brain, such as learning from the past experience, learning on the job, rapid analysis of alternatives, ability to cope with incomplete information, quick adaptation to new situations and events.

If one considers the dictionary definition of *reactive*, see the box below, the "ready response to some treatment, situation, or stimulus" is the part of interest to us. The contrary in our context is: inactive, inert, unresponsive. For sure, its contrary is not proactive! In fact, when the level of automation increases, the final user wins, but the work becomes much more challenging for the researcher: he has to be fully proactive to anticipate the different adaptation needs of a Reactive Search Optimization algorithm.

re·ac·tive
1: of, relating to, or marked by reaction or reactance
2 a: readily responsive to a stimulus b: occurring as a result of stress or emotional upset
re·ac·tion
Date: circa 1611
1 a: the act or process or an instance of reacting b: resistance or opposition to a force, influence, or movement . . .
2: a response to some treatment, situation, or stimulus "her stunned reaction to the news" . . .
3: bodily response to or activity aroused by a stimulus: a: an action induced by vital resistance to another action ; especially : the response of tissues to a foreign substance (as an antigen or infective agent) . . .
4: the force that a body subjected to the action of a force from another body exerts in the opposite direction
5 a (1): chemical transformation or change : the interaction of chemical entities (2): the state resulting from such a reaction b: a process involving change in atomic nuclei
(derived from: *Merriam-Webster online dictionary*)

Before dwelling on the technical details, let's briefly mention some relevant characteristics of Reactive Search Optimization when applied in the context of local search based processes.

Learning on the job Real-world problems have a rich structure. While many alternative solutions are tested in the exploration of a search space, patterns and regularities appear. The human brain quickly learns and drives future decisions based on previous observations. This is the main inspiration source for inserting online machine learning techniques into the optimization engine of RSO.

Rapid generation and analysis of many alternatives Often, to solve a problem one searches among a large number of alternatives, each requiring the analysis of what-if scenarios. The search speed is improved if alternatives are generated in a strategic manner, so that different solutions are chained along a trajectory in the search space exploring wide areas and rapidly exploiting the most promising solutions.

Flexible decision support Crucial decisions depend on several factors and priorities which are not always easy to describe before starting the solution process. Feedback from the user in the preliminary exploration phase can be incorporated so that a better tuning of the final solutions takes the end user preferences into account.

Diversity of solutions The final decision is up to the user, not the machine. The reason is that not all qualitative factors of a problem can be encoded into a computer program. Having a set of diverse near-best alternatives is often crucial for the decision maker.

Anytime solutions The user decides when to stop searching. A first complete solution is generated rapidly, better and better ones are produced in the following search phases. The more the program runs, the bigger the possibility to identify excellent solutions, but if you want a solution fast you are going to get it!

Methodologies of interest for Reactive Search Optimization include machine learning and statistics, in particular neural networks, artificial intelligence, reinforcement learning, active or query learning.

When one considers the *source* of information that is used for the algorithm selection and tuning process, it is important to stress that there are at least three different possibilities:

1. *Problem-dependent information*. This is related to characteristics of the specific problem. For example, a local search scheme for the Traveling Salesman Problem needs a different neighborhood definition w.r.t. a scheme for the network partitioning problem.
2. *Task-dependent information*. A single problem consists of a set of instances or tasks with characteristics which can be radically different. For example, a Traveling Salesman task for delivering pizza among a set locations in Los Angeles can be very different from a pizza delivery task in Trento, a small and pleasant town in the Alps.

3. *Local properties in configuration space*. When one considers a local search scheme based on perturbation one builds a trajectory in configuration space given by successive sample points generated by selecting and applying the local moves. In poetic terms, one travels along a fitness surface with peaks and valleys which can vary a lot during the trip. For example, the size and depth of the attractors around local minimizers can vary from a reasonably flat surface, to one characterized by deep wells. If a scheme for escaping local minimizers is adapted also to the local characteristics, better results can be expected.

Now, the first possibility is the typical source of information for offline algorithm selection and parameter tuning, while the last two possibilities are the starting point for the online schemes of RSO, where parameters are dynamically tuned based on the current optimization state and previous history of the search process while working on a specific instance.

Intelligent optimization, a superset of Reactive Search Optimization, refers to a more extended area of research, including online and off-line schemes based on the use of memory, adaptation, incremental development of models, experimental algorithmics applied to optimization, intelligent tuning and design of heuristics.

The RSO approach of learning on the job is to be contrasted with off-line parameter tuning. This orthogonal approach is studied for example in [80, 79], that proposes methods to predict per-instance and per-parameter run-times with reasonable accuracy. These predictive models are then used to predict which parameter settings result in the lowest run-time for a given instance, thus automatically tuning the parameter values of a stochastic local search (SLS) algorithm on a per-instance basis by simply picking the parameter configuration that is predicted to yield the lowest run-time. An iterated local search (ILS) algorithm for the algorithm configuration problem is proposed in [81]. The approach works for both deterministic and randomized algorithms and can be applied regardless of tuning scenario and optimization objective.

On-line and off-line strategies are complementary: in fact, even RSO methods tend to have a number of parameters that remain fixed during the search and can hence be tuned by off-line approaches.

The following part of this chapter is organized as follows. First the different opportunities for RSO strategies are listed and briefly commented. Section 2 describes different RSO schemes that have been introduced in the literature. A much more extended presentation has been recently published in [14]. Then sample applications of Reactive Search Optimization principles are illustrated in Section 3.

The brevity of this chapter does not allow for a complete listing and examination: we ask the omitted authors for forgiveness, and encourage authors of novel work to get in touch with us. The Reactive Search Optimization community[1] and software[2] web sites are two additional sources of information which can be mined for more detailed interests.

[1] `http://reactive-search.org/`

[2] `http://reactive-search.com/`

2 Different reaction possibilities

The design principles of many superficially different techniques for diversifying the search in a responsive manner according to the RSO principles of learning while optimizing, are strongly related. The unifying principle is that of using online reactive learning schemes to increase the robustness and to permit more *hands-off* usage of software for optimization.

For brevity we concentrate this review chapter on reactive techniques applied to single local search streams. Other possibilities related to using more than one search stream, a.k.a. population-based methods, genetic algorithms, evolutionary techniques, etc. range from adaptive portfolios, to restart strategies, to racing techniques, to intelligent and reactive solver teams [14].

2.1 Reactive prohibitions

It is part of commonsense that the discovery of radically new solutions which is associated to real creativity demands departing from the usual way of doing things, avoiding known solutions. The popular concepts of "lateral thinking" and "thinking outside the box" are related to shifting the point of view, observing an old problem with new eyes, discarding pet hypotheses.

> Techniques that apply lateral thinking to problems are characterized by the shifting of thinking patterns, away from entrenched or predictable thinking to new or unexpected ideas. A new idea that is the result of lateral thinking is not always a helpful one, but when a good idea is discovered in this way it is usually obvious in hindsight, which is a feature lateral thinking shares with a joke.
>
> There are a number of mental tools or methods that can be used to bring about lateral thinking. These include the following:
>
> . . .
>
> Provocation: Declare the usual perception out of bounds, or provide some provocative alternative to the usual situation under consideration. . . .
>
> As an example see the provocation on cars having square wheels.
>
> Challenge: Simply challenge the way things have always been done or seen, or the way they are. This is done not to show there is anything wrong with the existing situation but simply to direct your perceptions to exploring outside the current area.
>
> For example you could challenge coffee cups being produced with a handle. There is nothing wrong with coffee cups having handles so the challenge is a direction to explore without defending the status quo. The reason for the

> handle seems to be that the cup is often too hot to hold directly. Perhaps coffee cups could be made with insulated finger grips . . .
>
> There are many other techniques . . . All these tools are practical matters for circumstances where our normal automatic perceptions and pattern matching tend to keep us trapped "within the box".
>
> (derived from Wikipedia "lateral thinking" voice, Jan 2008)

When one reflects about the above connections, it is not surprising to see ideas related to using "prohibitions" to encourage diversification and exploration (the technical terms for true creativity in the context of optimization heuristics) in different contexts and different times. For example, they can be found in the *denial* strategy of [121]: once common features are detected in many suboptimal solutions, they are *forbidden*.

The full blossoming of "intelligent prohibition-based heuristics" starting from the late eighties is greatly due to the role of F. Glover in the proposal and diffusion of a rich variety of meta-heuristic tools under the umbrella of Tabu Search (TS) [68, 69], but see also [73] for an independent seminal paper. It is evident that Glover's ideas have been a source of inspiration for many approaches based on the intelligent use of memory in heuristics.

The main competitive advantage of TS with respect to alternative heuristics based on local search like Simulated Annealing (SA) lies in the intelligent use of the past history of the search to influence its future steps. Because TS includes now a wide variety of methods, we prefer the term *prohibition-based search* when the investigation is focussed onto the use of prohibition to encourage diversification.

Let us assume that the feasible search space is the set of binary strings with a given length L: $\mathscr{X} = \{0,1\}^L$. $X^{(t)}$ is the current configuration and $N(X^{(t)})$ the set of its neighbors, i.e., configurations that can be explored in the following step (Sec. 2.2 is mainly focused on neighborhoods). In prohibition-based search some of the neighbors are *prohibited*, a subset $N_A(X^{(t)}) \subset N(X^{(t)})$ contains the *allowed* ones. The general way of generating the search trajectory is given by:

$$X^{(t+1)} = \text{Best-Neighbor}(\, N_A(X^{(t)})\,) \tag{1}$$

$$N_A(X^{(t+1)}) = \text{Allow}(N(X^{(t+1)}), X^{(0)}, \ldots, X^{(t+1)}) \tag{2}$$

The set-valued function Allow selects a subset of $N(X^{(t+1)})$ in a manner that depends on the entire search trajectory $X^{(0)}, \ldots, X^{(t+1)}$.

By analogy with the concept of *abstract data type* in Computer Science [2], and with the related *object-oriented* software engineering framework [49], it is useful to separate the abstract concepts and operations of TS from the detailed implementation, i.e., realization with specific data structures. In other words, *policies* (that determine which trajectory is generated in the search space, what the balance of intensification and diversification is, etc.) should be separated from *mechanisms* that determine *how* a specific policy is realized. A first classification distinguishes between *strict-TS*, which prohibits only the moves leading back to previously visited configurations, and *fixed-TS*, which prohibits only the inverses of moves which have been applied recently in the search, their recency being judged according to a prohibition parameter T, also called tabu tenure.

Let μ^{-1} denote the *inverse* of a move, for example, if μ_i is changing the i-th bit of a binary string from 0 to 1, $\mu_i{}^{-1}$ changes the same bit from 1 to 0. A neighbor is allowed if and only if it is obtained from the current point by applying a move such that its inverse has not been used during the last T iterations. In detail, if $\textsc{LastUsed}(\mu)$ is the last usage time of move μ ($\textsc{LastUsed}(\mu) = -\infty$ at the beginning):

$$N_A(X^{(t)}) = \{X = \mu \circ X^{(t)} \text{ s. t. } \textsc{LastUsed}(\mu^{-1}) < (t-T)\} \tag{3}$$

If T changes with the iteration counter depending on the search status, and in this case the notation is $T^{(t)}$, the general dynamical system that generates the search trajectory comprises an additional evolution equation for $T^{(t)}$:

$$T^{(t)} = \textsc{React}(T^{(t-1)}, X^{(0)}, \ldots, X^{(t)}) \tag{4}$$

$$N_A(X^{(t)}) = \{X = \mu \circ X^{(t)} \text{ s. t. } \textsc{LastUsed}(\mu^{-1}) < (t-T^{(t)})\} \tag{5}$$

$$X^{(t+1)} = \textsc{Best-Neighbor}(N_A(X^{(t)})) \tag{6}$$

Rules to determine the prohibition parameter by reacting to the repetition of previously-visited configurations have been proposed in [26] (*reactive-TS*, *RTS* for short). In addition, there are situations where the single reactive mechanism on T is not sufficient to avoid long cycles in the search trajectory and therefore a second reaction is needed [26].

The prohibition parameter T used in equation (3) is related to the amount of *diversification*: the larger T, the longer the distance that the search trajectory must go before it is allowed to come back to a previously visited point. In particular, the following relationships between prohibition and diversification are demonstrated in [11] for a search space consisting of binary strings with basic moves flipping individual bits:

- The Hamming distance H between a starting point and successive point along the trajectory is strictly increasing for $T+1$ steps.

$$H(X^{(t+\Delta t)}, X^{(t)}) = \Delta t \quad \text{for} \quad \Delta t \leq T+1$$

- The minimum repetition interval R along the trajectory is $2(T+1)$.

$$X^{(t+R)} = X^{(t)} \;\Rightarrow\; R \geq 2(T+1)$$

In general, because a larger prohibition value implies a more limited choice of moves, it makes sense to set T to the smallest value that guarantees a sufficient degree of diversification

In *reactive-TS* [26] the prohibition T is determined through feedback (i.e., *reactive*) mechanisms during the search. T is equal to one at the beginning (the inverse of a given move is prohibited only at the next step), it increases only when there is *evidence* that diversification is needed, it decreases when this evidence disappears. The evidence that diversification is needed is signaled by the repetition of previously visited configurations. This criterion needs to be generalized when the search space dimension becomes very large, so that the exact repetition of configurations can become very rare even if the trajectory is confined. In this case, one can monitor an appropriate distance measure from a given starting configuration. An insufficient growth if the distance as a function of the number of steps can be taken as evidence of confinement, see for example [20].

A more radical *escape* mechanism can be triggered when the basic prohibition mechanism is not sufficient to guarantee diversification. In [26] the escape (a number of random steps) is triggered when too many configurations are repeated too often. Further details about applications, implementation and data structures can be found in [14].

A reactive determination of the T value can change the process of escaping from a local minimum in a qualitative manner: one obtains an (optimistic) logarithmic increase in the *strict-TS* algorithm, and a (pessimistic) increase that behaves like the square root of the number of iterations in the reactive case [14].

Robust stochastic algorithms related to the previously described deterministic versions can be obtained in many ways. For example, prohibition rules can be substituted with *probabilistic generation-acceptance rules* with large probability for allowed moves, small for prohibited ones, see for example the *probabilistic-TS* [68]. Asymptotic results for TS can be obtained in probabilistic TS [56]. In a different proposal (*robust-TS*) the prohibition parameter is randomly changed between an upper and a lower bound during the search [122].

If the neighborhood evaluation is expensive, the exhaustive evaluation can be substituted with a partial *stochastic sampling*: only a partial list of candidates is examined before choosing the best allowed neighbor.

Finally, other possibilities which are softer than prohibitions exist. For example the HSAT [67] variation of GSAT introduces a tie-breaking rule into GSAT: if more moves produce the same (best) Δf, the preferred move is the one that has not been applied for the longest span. HSAT can be seen as a "soft" version of Tabu Search: while TS prohibits recently-applied moves, HSAT discourages recent moves if the same Δf can be obtained with moves that have been "inactive" for a longer time.

2.2 Reacting on the neighborhood

Local search based on perturbing a candidate solution is a first paradigmatic case where simple online adaptation and learning strategies can be applied. Let $\mathscr{X}$ be the search space, $X^{(t)}$ the current solution at iteration ("time") t. $N(X^{(t)})$ is the neighborhood of point $X^{(t)}$, obtained by applying a set of basic moves $\mu_0, \mu_1, \ldots, \mu_M$ to the current configuration:

$$N(X^{(t)}) = \{X \in \mathscr{X} \text{ such that } X = \mu_i(X^{(t)}), i = 0, \ldots, M\}$$

Local search starts from an admissible configuration $X^{(0)}$ and builds a *search trajectory* $X^{(0)}, \ldots, X^{(t+1)}$. The successor of the current point is a point in the neighborhood with a lower value of the function f to be minimized. If no neighbor has this property, i.e., if the configuration is a local minimizer, the search stops.

$$Y \leftarrow \textsc{Improving-Neighbor}(\, N(X^{(t)})\,) \tag{7}$$

$$X^{(t+1)} = \begin{cases} Y & \text{if } f(Y) < f(X^{(t)}) \\ X^{(t)} & \text{otherwise (search stops)} \end{cases} \tag{8}$$

IMPROVING-NEIGHBOR returns an improving element in the neighborhood. In a simple case this is the element with the lowest f value, but other possibilities exist, as we will see in what follows.

Online learning strategies can be applied in two contexts: selection of the neighbor or selection of the neighborhood. While these strategies are part of the standard bag of tools, they in fact can be seen as simple forms of reaction to the recent history of evaluations.

When the neighborhood is fixed, one can modify the unresponsive strategy which considers all neighbors before selecting one of the best moves (*best-improvement local search*) and obtain a very simple reactive strategy like FIRSTMOVE. FIRSTMOVE accepts the first improving neighbor if one is found before examining all candidates. The simple adaptation is clear: the exact number of neighbors evaluated before deciding the next move depends not only on the instance but on the particular local properties in the configuration space around the current point. On the average, less neighbors will need to be evaluated at the beginning of the search, when finding an improving move is simple, more neighbors when the trajectory goes deeper and deeper into a given local minimum attractor.

When the neighborhood is changed depending on the local configuration one obtains for example the Variable Neighborhood Search (VNS) [72]. VNS considers a *a set of neighborhoods*, defined *a priori* at the beginning of the search, and then uses the most appropriate one during the search.

Variable Neighborhood Descent [74] (VND), see Fig. 1, uses the default neighborhood first, and the ones with a higher number only if the default neighborhood fails (i.e., the current point is a local minimum for N_1), and only until an improving move is identified, after which it reverts back to N_1. When VND is coupled with an ordering of the neighborhoods according to the *strength* of the perturbation, one realizes the principle *use the minimum strength perturbation leading to an improved solution*, which is present also in more advanced RSO methods. The consideration of neighborhoods of increasing diameter (distance of its members w.r.t. the starting configuration) can be considered as a form of *diversification*. A strong similarity with the design principle of Reactive Tabu Search is present, see later in this chap-

```
function VariableNeighborhoodDescent (N_1, ..., N_{k_max})
    repeat until no improvement or max CPU time elapsed
        k ← 1 // index of the default neighborhood
        while k ≤ k_max:
            X' ← BestNeighbor (N_k(X)) // neighborhood exploration
            if f(X') < f(X)
                X ← X' ; k ← 1 // success: back to default neighborhood
            else
                k ← k + 1 // try with the following neighborhood
```

Fig. 1 The VND routine. Neighborhoods with higher numbers are considered only if the default neighborhood fails and only until an improving move is identified. X is the current point.

ter, where diversification through prohibitions is activated when there is evidence of entrapment in an attraction basin and gradually reduced when there is evidence that a new basin has been discovered.

An explicitly reactive-VNS is considered in [35] for the VRPTW problem (vehicle routing problem with time windows), where a construction heuristic is combined with VND using first-improvement local search. Furthermore, the objective function used by the local search operators is modified to consider the waiting time to escape from a local minimum. A preliminary investigation about a self-adaptive neighborhood ordering for VND is presented in [78]. The different neighborhoods are ranked according to their observed benefits in the past.

Let's also note some similarities between VNS and the adaptation of the search region in stochastic search technique for continuous optimization, see the discussion later in this chapter. Neighborhood adaptation in the continuous case, see for example the Affine Shaker algorithm in [25], is mainly considered to speed-up convergence to a local minimizer, not to jump to nearby valleys.

A related possibility to cause a more radical move when simple ones are not sufficient to escape from a local minimum is *iterated local search* (ILS). ILS is based on building a sequence of locally optimal solutions by perturbing the current local minimum and applying local search after starting from the modified solution. The work about large-step Markov chain of [96, 94, 95, 126] contains very interesting results coupled with a clear description of the principles.

In VNS minimal perturbations maintain the trajectory in the starting attraction basin, while excessive ones bring the method closer to a random sampling, therefore loosing the boost which can be obtained by the problem structural properties. A possible solution consists of perturbing by a short random walk of a length which is *adapted* by statistically monitoring the progress in the search. Memory and reactive learning can be used in a way similar to that of [20] to adapt the *strength* of the perturbation to the local characteristics in the neighborhood of the current solution for the considered instance. Creative perturbations can be obtained by temporarily changing the objective function with penalties so that the current local minimum is displaced, like in [31, 45], see also the later description about reactively changing the objective function, or by *fixing* some configuration variables and optimizing sub-parts of the problem [92].

2.3 Reacting on the annealing schedule

A widely popular stochastic local search technique is the Simulated Annealing (SA) method [88] based on the theory of Markov processes. The trajectory is built in a randomized manner: the successor of the current point is chosen stochastically, with a probability that depends only on the difference in f value w.r.t. the current point and not on the previous history.

$$Y \leftarrow \textsc{Neighbor}(\, N(X^{(t)})\,)$$

$$X^{(t+1)} \leftarrow \begin{cases} Y & \text{if } f(Y) \leq f(X^{(t)}) \\ Y & \text{if } f(Y) > f(X^{(t)}), \text{ with probability } p = e^{-(f(Y)-f(X^{(t)}))/T} \\ X^{(t)} & \text{if } f(Y) > f(X^{(t)}), \text{ with probability } (1-p). \end{cases} \tag{9}$$

SA introduces a *temperature* parameter T which determines the probability that worsening moves are accepted: a larger T implies that more worsening moves tend to be accepted, and therefore a larger diversification occurs. An analogy with energy-minimization principles in physics is present, and this explains the “temperature term”, as well as the term “energy” to refer to the function f.

If the local configuration is close to a local minimizer and the temperature is already very small in comparison to the upward jump which has to be executed to escape from the attractor, the system will *eventually* escape, but an enormous number of iterations can be spent around the attractor. The memoryless property (current move depending only on the current state, not on the previous history) makes SA look like a dumb animal indeed. It is intuitive that a better performance can be obtained by using memory, by self-analyzing the evolution of the search, by developing simple models and by activating more direct *escape* strategies aiming at a better usage of the computational resources devoted to optimization.

Even if a vanilla version of a cooling schedule for SA is adopted (starting temperature T_{start}, geometric cooling schedule $T_{t+1} = \alpha\, T_t$, with $\alpha < 1$, final temperature T_{end}), a sensible choice has to be made for the three involved parameters T_{start}, α, and T_{end}. The work [130] suggests to estimate the distribution of f values. The standard deviation of the energy distribution defines the maximum-temperature scale, while the minimum change in energy defines the minimum-temperature scale. These temperature scales tell us where to begin and end an annealing schedule.

The analogy with physics is further pursued in [89], where concepts related to *phase transitions* and *specific heat* are used. The idea is that a phase transition is related to solving a sub-part of a problem. After a phase transition corresponding to the big reconfiguration occurs, finer details in the solution have to be fixed, and this requires a slower decrease of the temperature.

When the parameters T_{start} and α are fixed *a priori*, the useful span of CPU time is practically limited. After the initial period the temperature will be so low that the system *freezes* and, with large probability, no tentative moves will be accepted anymore in the remaining CPU time of the run. For a new instance, guessing appropriate parameter values is difficult. Furthermore, in many cases one would like to use an *anytime algorithm*, so that longer allocated CPU times are related to possibly better and better values until the user decides to stop. *Non-monotonic cooling schedules* are a reactive solution to this difficulty, see [46, 105, 1]. The work [46] suggests to reset the temperature once and for all at a constant temperature high enough to escape local minima but also low enough to visit them, for example, at the temperature T_{found} when the best heuristic solution is found in a preliminary SA simulation.

A non-monotonic schedule aims at: exploiting an attraction basin rapidly by decreasing the temperature so that the system can settle down close to the local minimizer, *increasing the temperature* to diversify the solution and visit other attraction basins, decreasing again after reaching a different basin. The implementation details have to do with determining an *entrapment* situation, for example from the fact that no tentative move is accepted after a sequence t_{max} of tentative changes, and determining the detailed temperature decrease-increase evolution as a function of events occurring during the search [105, 1]. Enhanced versions involve a learning process to choose a proper value of the heating factor depending on the system state. Let's note that similar "strategic oscillations" have been proposed in tabu search, in particular in the reactive tabu search of [26], see later in this chapter, and in variable neighborhood search.

Modifications departing from the exponential acceptance rule and other adaptive stochastic local search methods for combinatorial optimization are considered in [99, 100]. The authors appropriately note that the optimal choices of algorithm parameters depend not only on the problem but also on the particular instance and that a proof of convergence to a globally optimum is not a selling point for a specific heuristic: in fact a simple random sampling, or even exhaustive enumeration (if the set of configurations is finite) will eventually find the optimal solution, although they are not the best algorithms to suggest. A simple adaptive technique is suggested in [100]: a perturbation leading to a worsening solution is accepted if and only if a fixed number of trials could not find an improving perturbation. The temperature parameter is eliminated. The positive performance of the method in the area of design automation suggests that the success of SA is "due largely to its acceptance of bad perturbations to escape from local minima rather than to some mystical connection between combinatorial problems and the annealing of metals."

"Cybernetic" optimization is proposed in [60] as a way to use probabilistic information for feedback during a run of SA. The idea is to consider more runs of SA running in parallel and to aim at *intensifying the search* (by lowering the temperature parameter) when there is evidence that the search is converging to the optimum value.

The application of SA to continuous optimization (optimization of functions defined on real variables) is pioneered by [48]. The basic method is to generate a new point with a random step along a direction $\mathbf{e}_h$, to evaluate the function and to accept the move with the exponential acceptance rule. One cycles over the different directions $\mathbf{e}_h$ during successive steps of the algorithm. A first critical choice has to do with the range of the random step along the chosen direction. A fixed choice obviously may be very inefficient: this opens a first possibility for *learning* from the local f surface. In particular a new trial point $\mathbf{x}'$ is obtained from the current point $\mathbf{x}$ as:

$$\mathbf{x}' = \mathbf{x} + \text{RAND}(-1,1) v_h \mathbf{e}_h$$

where $\text{RAND}(-1,1)$ returns a random number uniformly distributed between -1 and 1, $\mathbf{e}_h$ is the unit-length vector along direction h, and v_h is the step-range parameter, one for each dimension h, collected into the vector $\mathbf{v}$. The v_h value is adapted during the search to maintain the number of *accepted* moves at about one-half of the total number of tried moves. Although the implementation is already reactive and based on memory, the authors encourage more work so that a “good monitoring of the minimization process” can deliver precious feedback about some crucial internal parameters of the algorithm.

In Adaptive Simulated Annealing (ASA), also known as very fast simulated re-annealing [82], the parameters that control the temperature cooling schedule and the random step selection are automatically adjusted according to algorithm progress. If the state is represented as a point in a box and the moves as an oval cloud around it, the temperature and the step size are adjusted so that all of the search space is sampled at a coarse resolution in the early stages, while the state is directed to promising areas in the later stages.

A reactive determination of parameters in an advanced simulated annealing application for protein folding is presented in [75].

2.4 Reacting on the objective function

In the above methods, the objective function f remains the guiding source of information to select the next move. Reactive diversification to encourage exploration of areas which are distant from a locally optimal configuration has been considered though an adaptive selection of the neighborhood or of the neighbor based on the local situation and on the past history of the search process. A more direct way to force diversification is to directly prohibit configurations or moves to create a pressure to reach adequate distances from a starting point.

This part considers a different way to achieve similar results, by reactively changing the function guiding the local search. For example, the act of visiting a local minimum may cause a local increase of the evaluation function value so that the point becomes less and less appealing, until eventually the trajectory is gently pushed to other areas. Of course, the real objective function values and the corresponding configurations are saved into memory before applying the modification process. The physics analogy is that of pushing a ball out of a valley by progressively raising the bottom of the valley.

A relevant problem for which objective function modifications have been extensively used is maximum satisfiability (MAX-SAT): the input consists of logic variables — with false and true values — and the objective is to satisfy the maximum number of clauses (a clause is the logical OR of literals, a literal is a variable or its negation). The decision version is called SAT, one searches for a variable assignment, if any exists, which makes a formula true.

The influential algorithm GSAT [117] is based on local search with the standard basic moves flipping the individual variables (from false to true and *vice versa*). Different noise strategies to escape from locally optimal configurations are added to GSAT in [115, 116]. In particular, the GSAT-with-walk algorithm introduces random walk moves with a certain probability. A prototypical evaluation function modification algorithm is the breakout method proposed in [98] for the related constraint satisfaction problem. The cost is measured as the sum of the weights associated to the violated constraints. Each weight is one at the beginning, at a local minimum the weight of each nogood is increased by one until one escapes from the given local minimum (a breakout occurs). Clause-weighting has been proposed in [114] for GSAT. A positive weight is associated to each clause to determine how often the clause should be counted when determining which variable to flip. The weights are dynamically modified during problem solving and the qualitative effect is that of “filling in” local optima while the search proceeds. Clause-weighting and the breakout technique can be considered as “reactive” techniques where a repulsion from a given local optimum is generated in order to induce an escape from a given attraction basin.

New clause-weighing parameters are introduced and therefore new possibilities for tuning the parameters based on feedback from preliminary search results. The algorithm in [113] suggests to use weights to encourage more priority on satisfying the “most difficult” clauses. One aims at *learning how difficult a clause is to satisfy*. These hard clauses are identified as the ones which remain unsatisfied after a try of local search descent followed by plateau search. Their weight is increased so that future runs will give them more priority when picking a move. More algorithms based on the same weighting principle are proposed in [63, 64], where clause weights are updated after each flip: the reaction from the unsatisfied clauses is now immediate as one does not wait until the end of a try (weighted GSAT or WGSAT). If weights are only increased, after some time their size becomes large and their relative magnitude will reflect the overall statistics of the SAT instance, more than the local characteristics of the portion of the search space where the current configuration lies. To combat this problem, two techniques are proposed in [64], either *reducing* the clause weight when a clause is satisfied, or storing the weight increments which took place recently, which is obtained by a weight decay scheme (each weight is reduced by a factor ϕ before updating it). Depending on the size of the increments and decrements, one achieves “continuously weakening incentives not to flip a variable” instead of the strict prohibitions of Tabu Search. The second scheme takes the *recency of moves* into account, the implementation is through a weight decay scheme updating so that each weight is reduced before a possible increment by δ if the clause is not satisfied:

$$w_i \leftarrow \phi\, w_i + \delta$$

where one introduces a decay rate ϕ and a "learning rate" δ. A faster decay (lower ϕ value) will limit the temporal extension of the context and imply a faster forgetting of old information. A critique of some *warping* effects that a clause-weighting dynamic local search can create on the fitness surface is present in [123]: in particular let's note that the fitness surface is changed in a global way after encountering a local minimum. Points which are very far from the local minimum, but which share some of the unsatisfied clauses, will also see their values changed.

A more recent proposal of a dynamic local search (DLS) for SAT is in [124]. The authors start from the Exponentiated Sub-Gradient (ESG) algorithm [112], which alternates search phases and weight updates, and develop a scheme with low time complexity of its search steps: Scaling and Probabilistic Smoothing (SAPS). Weights of satisfied clauses are multiplied by α_{sat}, while weights of unsatisfied clauses are multiplied by α_{unsat}, then all weights are smoothed towards their mean $\overline{w}$: $w \leftarrow w\,\rho + (1-\rho)\,\overline{w}$. A *reactive version* of SAPS (RSAPS) is then introduced that adaptively tunes one of the algorithm's important parameters.

A similar approach of dynamically modifying the objective function has been proposed with the term of Guided Local Search (GLS) [127, 128] for other applications. GLS aims at enabling intelligent search schemes that exploit problem- and search-related information to guide a local search algorithm. Penalties depending on solution features are introduced and dynamically manipulated to distribute the search effort over the regions of a search space. A penalty formulation for TSP including memory-based trap-avoidance strategies is proposed in [129]. One of the strategies avoids visiting points that are close to points visited before, a generalization of the previously described STRICT-TS strategy. A recent algorithm with an *adaptive clause weight redistribution* is presented in [83], it adopts resolution-based preprocessing and reactive adaptation of the total amount of weight to the degree of stagnation of the search.

Let us note that the use of a dynamically modified (learned) evaluation function is related to the machine learning technique of *reinforcement learning* (RL). Early applications of RL in the area of local search is presented in [34, 33]. Some reinforcement learning approaches for optimization are also discussed in [8]. Recent work includes [15], on-the-fly parameter tuning for evolutionary algorithms in [55], and the presentation in [14].

3 Applications of Reactive Search Optimization

It must be noted that Reactive Search Optimization is not a rigid algorithm but a general framework: specific algorithms have been introduced for unconstrained and constrained tasks, with different stochastic mechanisms and rules for selecting neighbors. As it usually happens in heuristics, the more specific knowledge about a problem is used, the better the results. Nonetheless, it was often the case that simple RSO versions realized with very limited effort could duplicate the performance of more complex schemes because of their simple embedded feedback loop, and without intensive parameter and algorithm tuning. A long-term goal of RSO is the progressive shift of the learning capabilities from the algorithm user to the algorithm itself, through machine learning techniques.

The RSO framework and related algorithms and tools have been and are being applied to a growing number of "classical" discrete optimization problems, continuous optimization tasks, and real-world problems arising in widely different contexts. The Web, see for example Google scholar, lists thousands of citations to the seminal papers, the following list is a selection of some applications we are aware of. We are always happy to hear from users and developers interested in RSO principles and applications.

In the following we summarize some applications in "classical" combinatorial tasks in Section 3.1, where by classical we mean abstract definitions of problems which have been extensively studied in the Computer Science and Operations Research community.

Then we present applications in the area of neural networks in Section 3.2, where RSO has been used to solve the optimization problems related to machine learning. Let's note that the synergy between optimization and machine learning is explored in the opposite direction in this case, i.e., of using optimization to solve machine learning tasks.

Then we discuss versions of RSO for continuous optimization tasks in Section 3.3.

Finally, in Section 3.4, we present some applications to problems which are closer to the real application areas. These problems are of course related to their abstract and clean definitions but usually contains more details and require more competence in the specific area to make substantial progress.

3.1 Classic combinatorial tasks

The seminal paper about RSO for Tabu Search (Reactive Tabu Search) presented preliminary experimental results on the 0-1 Knapsack Problem, and on the Quadratic Assignment Problem [26]. A comparison with Simulated Annealing on QAP tasks is contained in [27]. An early experimental comparison of Reactive Search Optimization with alternative heuristics (Repeated Local Minima Search, Simulated Annealing, Genetic Algorithms and Mean Field Neural Networks) is presented in [28].

An application of a self-controlling software approach to Reactive Tabu Search is presented in [57] with results on the QAP problem.

A reactive local search-based algorithm (adaptive memory search) for the 0/1-Multidemand Multidimensional knapsack problem (0/1-MDMKP) is proposed in [5]. The 0/1-MDMKP represents a large class of important real-life problems, including portfolio selection, capital budgeting, and obnoxious and semi-obnoxious facility location problems. A different application is considered in [76] for the disjunctively constrained knapsack problem (DCKP), a variant of the standard knapsack problem with special disjunctive constraints. A disjunctive constraint is a couple of items for which only one item is packed.

A reactive tabu search algorithm for Minimum Labeling Spanning Tree is considered in [39, 40], together with other meta-heuristics. The problem is as follows: Given a graph G with a color (label) assigned to each edge one looks for a spanning tree of G with the minimum number of different colors. The problem has several applications in telecommunication networks, electric networks, multi-modal transportation networks, among others, where one aims at ensuring connectivity by means of homogeneous connections.

The graph partitioning problem has been a test case for advanced local search heuristics starting at least from the seminal Kernighan and Lin paper [84], which proposes a variable-depth schemes. This is is fact a simple prohibition-based (tabu) scheme where swaps of nodes among the two sets of the partitions are applied, and the just swapped nodes are kept fixed during a sequence of tentative moves in search of an improving chain. Greedy, Prohibition-based, and Reactive Search Optimization Heuristics for Graph Partitioning are proposed and compared in [11], Multilevel Reactive Tabu Search techniques, based on producing coarse versions of very large graphs are considered for Graph Partitioning in [10].

A reactive tabu search version for the vehicle routing problem with time windows is designed in [44], while a version of the vehicle routing problem with back-hauls is considered in[50] and [104]. A Reactive Variable Neighborhood Search for the Vehicle-Routing Problem with Time Windows is proposed in [35]. Vehicle routing with soft time windows and Erlang travel times is studied in [109].

An RSO scheme is applied to the maximum clique problem in graphs in [19] [23]. A clique is a subset of nodes which are mutually interconnected, the problem is related to identifying densely interconnected communities and, in general, to clustering issues. A relaxed quasi-clique version of the problem where some edged may be missing is addressed in [38].

The work in [132] designs Reactive Prohibition-Based Ant Colony Optimization (RPACO): A New Parallel Architecture for Constrained Clique Sub-Graphs. This paper introduces a new algorithm that combines the stigmergic capabilities of Ant Colony Optimization (ACO) with local search heuristics to solve the maximum and maximum-weighted clique problem. The introduced Reactive Prohibition-based Ant Colony Optimization (RPACOMCP) complements the intelligent ant colony search with a prohibition-based diversification technique, where the amount of diversification is determined in an automated way through a feedback (history-sensitive) scheme.

Maximum satisfiability is considered in [21], [20] [97], reactive SAPS (scaling and probabilistic smoothing) [124]. Constraint satisfaction in [102]. Reactive local search techniques for the maximum *k*-conjunctive constraint satisfaction problem (MAX-*k*-CCSP) in [22]. A worst-case analysis of tabu search as a function of the tabu list length for the MAX-TWO-SAT problem is presented in [97], with applications also to a reactive determination of the prohibition.

In [111] the authors address the problem of computing a graph similarity measure which is generic in the sense that other well known graph similarity measures can be viewed as special cases of it. They propose and compare two algorithms: an Ant Colony Optimization based algorithm and a Reactive Search Optimization and show that they obtain complementary results.

A classification of methods to manage the prohibition period (Tabu tenure) in the literature is presented in [54] together with a new reactive Tabu tenure adaptation mechanism. The generic method is tested on the k-coloring problem.

A Reactive Tabu Search algorithm with variable clustering for the Unicost Set Covering Problem is proposed in [86]. Unicost SCPs arise in graph theory when one must select a minimum covering of edges by nodes or nodes by cliques. In addition, in many practical applications (crew scheduling, political redistricting, conservation biology, etc.) the relative variation in the weights may be small enough to warrant a unicost model.

3.2 Neural networks and VLSI systems with learning capabilities

While derivative-based methods for training from examples have been used with success in many contexts (error back-propagation is an example in the field of neural networks), they are applicable only to differentiable performance functions and are not always appropriate in the presence of local minima. In addition, the calculation of derivatives is expensive and error-prone, especially if special-purpose VLSI hardware is used. A radically different approach is used in [29]: the task is transformed into a combinatorial optimization problem (the points of the search space are binary strings), and solved with the Reactive Search Optimization algorithm. To speed up the neighborhood evaluation phase a stochastic sampling of the neighborhood is adopted and a “smart” iterative scheme is used to compute the changes in the performance function caused by changing a single weight. RSO escapes rapidly from local minima, it is applicable to non-differentiable and even discontinuous functions and it is very robust with respect to the choice of the initial configuration. In addition, by fine-tuning the number of bits for each parameter one can decrease the size of the search space, increase the expected generalization and realize cost-effective VLSI.

Reactive Tabu Search in Semi-supervised Classification is proposed in [133]. With a linear kernel their RTS implementation can effectively find optimal global solutions for the primal Mixed Integer Programming Transductive Support Vector Machine (MIP-TSVM) with relatively large problem dimension.

In contrast to the exhaustive design of systems for pattern recognition, control, and vector quantization, an appealing possibility consists of specifying a general architecture, whose parameters are then tuned through Machine Learning (ML). ML becomes a combinatorial task if the parameters assume a discrete set of values: the Reactive Tabu Search (RTS) algorithm permits the training of these systems with low number of bits per weight, low computational accuracy , no local minima "trapping", and limited sensitivity to the initial conditions [17, 16].

Special-purpose VLSI modules have been developed to be used as components of fully autonomous massively-parallel systems for real-time adaptive applications. In contrast to many "emulation" approaches, the developed VLSI completely reflects the combinatorial structure used in the learning algorithms.

Applications considered are in the area of pattern recognition (Optical Character Recognition), event triggering in High Energy Physics [3], control of non-linear systems [28], compression of EEG signals [24]. The first product of a joint project between University of Trento, INFN and IRST was the TOTEM chip [16, 18], and more general special-purpose VLSI realizations are described in [4, 3]. A parallel neurochip for neural networks implementing the Reactive Tabu Search algorithm and application case studies are presented in [52]. An FPGA implementation of the TOTEM chip is presented in [6].

3.3 Continuous optimization

A simple benchmark on a function with many suboptimal local minima is considered in [29], where a straightforward discretization of the domain is used. A novel algorithm for the global optimization of functions (C-RTS) is presented in [30], in which a combinatorial optimization method cooperates with a stochastic local minimizer. The combinatorial optimization component, based on Reactive Search Optimization, locates the most promising boxes, where starting points for the local minimizer are generated. In order to cover a wide spectrum of possible applications with no user intervention, the method is designed with adaptive mechanisms: in addition to the reactive adaptation of the prohibition period , the box size is adapted to the local structure of the function to be optimized (boxes are larger in "flat" regions, smaller in regions with a "rough" structure). An application of intelligent prohibition-based strategies to continuous optimization is presented in [43].

A Reactive Affine Shaker method for continuous optimization is studied in [36, 37]. The work presents an adaptive stochastic search algorithm for the optimization of functions of continuous variables where the only hypothesis is the pointwise computability of the function. The main design criterion consists of the adaptation of a search region by an affine transformation which takes into account the local knowledge derived from trial points generated with uniform probability. Heuristic adaptation of the step size and direction allows the largest possible movement per function evaluation. The experimental results show that the proposed technique is, in spite of its simplicity, a promising building block to consider for the development of more complex optimization algorithms, particularly in those cases where the objective function evaluation is expensive.

A Gregarious Particle Swarm Optimizer (GPSO) is proposed in [106]. The particles (the different local searchers) adopt a reactive determination of the step size, based on feedback from the last iterations. This is in contrast to the basic particle swarm algorithm, in which no feedback is used for the self-tuning of algorithm parameters. The novel scheme presented, when tested on a benchmark for continuous optimization, besides generally improving the average optimal values found, reduces the computation effort.

3.4 Real-world applications

A real-world application in the area of electric power distribution: service restoration in distribution systems is studied in [125], fast optimal setting for voltage control equipment considering interconnection of distributed generators is proposed in [103], Service Restoration in Distribution Systems in [66], distribution load transfer operation in [65].

A continuation of the work in [44] is proposed in [108] to aid in the coordination and synchronization of the production and delivery of multi-product newspapers to bulk delivery locations. The problem is modeled as an open vehicle routing problem with time windows and zoning constraints. The methodology is applied to the newspaper production and distribution problem in a major metropolitan area.

In the field of industrial production planning, [58] studies applications of modern heuristic search methods to pattern sequencing problems. Flexible job-shop scheduling is studied in [42] and [41]. The plant location problem is studied in [53]. The work [59] is dedicated to solving the continuous flow-shop scheduling problem Adaptive self-tuning neurocontrol is considered in [107]: the objective is to construct an adaptive control scheme for unknown time-dependent nonlinear plants.

Various applications of RSO focussed on problems arising in the design and management of telecommunication networks. RSO for traffic grooming in optical WDM networks is considered in [12]. Optimal conformance test selection is studied in [51]. Conformance testing is used to increase the reliability of telecommunication applications. Locating Hidden Groups in Communication Networks Using Hidden Markov Models is addressed in [93]. A communication network is a collection of social groups that communicate via an underlying communication medium. In such a network, a hidden group may try to camouflage its communications amongst the typical communications of the network. The task of increasing internet capacity is considered in [62]. The multiple-choice multi-dimensional knapsack problem (MMKP) with applications to service level agreements and multimedia distribution is studied in [76] and [77]. They consider the model of allocation resources and the dynamic adaptation of system of resources for multimedia multi-sessions. High quality solutions, reaching the optimal/best for several instances are obtained through a reactive local search scheme. In the area of wireless and cellular communication networks, the work in [13] considers the Optimal Wireless Access Point Placement for Location-Dependent Services, and the work in [61] proposes a tabu search heuristic for the dimensioning of 3G multi-service networks.

A heuristic approach based on a hybrid operation of reactive tabu search (RTS) and adaptive memory programming (AMP) is proposed in [91] to solve the vehicle routing problem with backhauls (VRPB). One is given a set of customers, some of which are linehauls (delivery points) and some are backhauls (collection points), a set of homogeneous vehicles and a depot. A distinguishing feature of this model is that all backhaul customers must be visited after all linehaul customers are served on each route. RTS is used with an escape mechanism which manipulates different neighborhood schemes in order to continuously balance intensification and diversification during the search process. The adaptive memory strategy takes the search back to the unexplored regions of the search space by maintaining a set of elite solutions and using them strategically with the RTS. The authors of [101] address the pickup and delivery problem with time windows using reactive tabu search.

Real-time dispatch of trams in storage yards is studied in [131].

In the military sector, simple versions of Reactive Tabu Search are considered in [87] in a comparison of techniques dedicated to designing an unmanned aerial vehicle (UAV) routing system. Hierarchical Tabu Programming is used in [7] for Finding the Underwater Vehicle Trajectories. Aerial reconnaissance simulations is the topic of [110]. The authors of [9] use an adaptive tabu search approach for solving the aerial fleet refueling problem.

In the automotive sector, RSO is used in [71] for improving vehicle safety. In detail, a mixed reactive tabu search method is used to optimize the design of a vehicle B-pillar subjected to roof crush.

Reactive Tabu Search and sensor selection in active structural acoustic control problems is considered in [85].

Visual representation of data through clustering is considered in [47].

The solution of the engineering roof truss design problem is discussed in [70] An application of reactive tabu search for designing barrelled cylinders and domes of generalized elliptical profile is studied in [32]. The cylinders and domes are optimized for their buckling resistance when loaded by static external pressure by using a structural analysis tool.

A Reactive Stochastic Local Search Algorithms is used to solve the Genomic Median Problem in [90]. The genomic median problem is an optimization problem inspired by a biological issue: it aims at finding the chromosome organization of the common ancestor to multiple living species. It is formulated as the search for a genome that minimizes a rearrangement distance measure among given genomes. An adaptive bin framework search method for a beta-sheet protein homopolymer model is used in [120], protein folding is studied in [118, 119]. Additional applications in bio-informatics include for example [120], which proposes an adaptive bin framework search method for a beta-sheet protein homopolymer model. A novel approach is studied based on the use of a bin framework for adaptively storing and retrieving promising locally optimal solutions. Each bin holds a set of conformations within a certain energy range and an one uses an adaptive strategy for restarting a given search process with a conformation retrieved from these bins when the search stagnates. An adaptive mechanisms chooses which conformations should be stored, based on the set of conformations already stored in memory, and biases choices when retrieving conformations from memory in order to overcome search stagnation. The energy and diversity thresholds for each bin are dynamically modified during the search process.

An adaptive meta-search method that alternates between two distinct modes of the search process at different levels is proposed in [119] for protein folding. The high level process ensures that unexplored promising parts of the search landscape are visited and the low-level search provides the thorough exploration of local neighborhoods. Multiple search processes are used in an intelligent way.

References

1. Abramson, D., Dang, H., Krisnamoorthy, M.: Simulated annealing cooling schedules for the school timetabling problem. Asia-Pacific Journal of Operational Research **16**, 1–22 (1999). URL `citeseer.ist.psu.edu/article/abramson97simulated.html`
2. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: Data Structures and Algorithms. Addison-Wesley (1983)
3. Anzellotti, G., Battiti, R., Lazzizzera, I., Lee, P., Sartori, A., Soncini, G., Tecchiolli, G., Zorat, A.: Totem: a highly parallel chip for triggering applications with inductive learning based on the reactive tabu search. In: AIHENP95. Pisa, IT (1995)
4. Anzellotti, G., Battiti, R., Lazzizzera, I., Soncini, G., Zorat, A., Sartori, A., Tecchiolli, G., Lee, P.: Totem: a highly parallel chip for triggering applications with inductive learning based on the reactive tabu search. International Journal of Modern Physics C **6**(4), 555–560 (1995)
5. Arntzen, H., Hvattum, L.M., Lokketangen, A.: Adaptive memory search for multidemand multidimensional knapsack problems. Comput. Oper. Res. **33**(9), 2508–2525 (2006). DOI http://dx.doi.org/10.1016/j.cor.2005.07.007
6. Avogadro, M., Bera, M., Danese, G., Leporati, F., Spelgatti, A.: The Totem neurochip: an FPGA implementation. In: Signal Processing and Information Technology, 2004. Proceedings of the Fourth IEEE International Symposium on, pp. 461–464 (2004)
7. Balicki, J.: Hierarchical Tabu Programming for Finding the Underwater Vehicle Trajectory. IJCSNS **7**(11), 32 (2007)
8. Baluja, S., Barto, A., Boyan, K.B.J., Buntine, W., Carson, T., Caruana, R., Cook, D., Davies, S., Dean, T., et al.: Statistical Machine Learning for Large-Scale Optimization. Neural Computing Surveys **3**, 1–58 (2000)
9. Barnes, J., Wiley, V., Moore, J., Ryer, D.: Solving the aerial fleet refueling problem using group theoretic tabu search. Mathematical and Computer Modeling **39**, 617–640 (2004)
10. Battiti, R., Bertossi, A., Cappelletti, A.: Multilevel Reactive Tabu Search for Graph Partitioning. Preprint UTM **554** (1999)
11. Battiti, R., Bertossi, A.A.: Greedy, prohibition, and reactive heuristics for graph partitioning. IEEE Transactions on Computers **48**(4), 361–385 (1999)
12. Battiti, R., Brunato, M.: Reactive search for traffic grooming in WDM networks. In: S. Palazzo (ed.) Evolutionary Trends of the Internet, IWDC2001, Taormina, Lecture Notes in Computer Science LNCS 2170, pp. 56–66. Springer-Verlag (2001)
13. Battiti, R., Brunato, M., Delai, A.: Optimal wireless access point placement for location-dependent services. Tech. rep., Universit di Trento DIT-03-052 (2003)
14. Battiti, R., Brunato, M., Mascia, F.: Reactive Search and Intelligent Optimization, *Operations research/Computer Science Interfaces*, vol. 45. Springer Verlag (2008)

15. Battiti, R., Campigotto, P.: Reinforcement learning and reactive search: an adaptive max-sat solver. In: N.F. M. Ghallab C.D. Spyropoulos, N. Avouris (eds.) Proceedings ECAI 08: 18th European Conference on Artificial Intelligence, Patras, Greece, Jul 21-25, 2008. IOS Press, Amsterdam (2008)
16. Battiti, R., Lee, P., Sartori, A., Tecchiolli, G.: Combinatorial optimization for neural nets: Rts algorithm and silicon. Tech. rep., Dept. of Mathematics, University of Trento, IT (1994). Preprint UTM 435
17. Battiti, R., Lee, P., Sartori, A., Tecchiolli, G.: Totem: A digital processor for neural networks and reactive tabu search. In: Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems, MICRONEURO 94, pp. 17–25. IEEE Computer Society Press, Torino, IT (1994). Preprint UTM 436-June 1994, Università di Trento, IT
18. Battiti, R., Lee, P., Sartori, A., Tecchiolli, G.: Special-purpose parallel architectures for high-performance machine learning. In: High Performance Computing and Networking. Milano, IT (1995). Preprint UTM 445, December 1994, Università di Trento, IT
19. Battiti, R., Protasi, M.: Reactive local search for maximum clique. In: G.F. Italiano, S. Orlando (eds.) Proceedings of the Workshop on Algorithm Engineering (WAE'97), Ca' Dolfin, Venice, Italy, pp. 74–82 (1997)
20. Battiti, R., Protasi, M.: Reactive search, a history-sensitive heuristic for MAX-SAT. ACM Journal of Experimental Algorithmics **2**(ARTICLE 2) (1997). Http://www.jea.acm.org/
21. Battiti, R., Protasi, M.: Solving MAX-SAT with non-oblivious functions and history-based heuristics. In: D. Du, J. Gu, P.M. Pardalos (eds.) Satisfiability Problem: Theory and Applications, no. 35 in DIMACS: Series in Discrete Mathematics and Theoretical Computer Science, pp. 649–667. American Mathematical Society, Association for Computing Machinery (1997)
22. Battiti, R., Protasi, M.: Reactive local search techniques for the maximum k-conjunctive constraint satisfaction problem (MAX-k-CCSP). Discrete Applied Mathematics **96**, 3–27 (1999)
23. Battiti, R., Protasi, M.: Reactive local search for the maximum clique problem. Algorithmica **29**(4), 610–637 (2001)
24. Battiti, R., Sartori, A., Tecchiolli, G., Tonella, Zorat, A.: Neural compression: an integrated approach to eeg signals. In: J. Alspector, R. Goodman, T.X. Brown (eds.) International Workshop on Applications of Neural Networks to Telecommunications (IWANNT*95), pp. 210–217. Stockholm, Sweden (1995)
25. Battiti, R., Tecchiolli, G.: Learning with first, second, and no derivatives: a case study in high energy physics. Neurocomputing **6**, 181–206 (1994)
26. Battiti, R., Tecchiolli, G.: The reactive tabu search. ORSA Journal on Computing **6**(2), 126–140 (1994)
27. Battiti, R., Tecchiolli, G.: Simulated annealing and tabu search in the long run: a comparison on QAP tasks. Computer and Mathematics with Applications **28**(6), 1–8 (1994)
28. Battiti, R., Tecchiolli, G.: Local search with memory: Benchmarking rts. Operations Research Spektrum **17**(2/3), 67–86 (1995)
29. Battiti, R., Tecchiolli, G.: Training neural nets with the reactive tabu search. IEEE Transactions on Neural Networks **6**(5), 1185–1200 (1995)
30. Battiti, R., Tecchiolli, G.: The continuous reactive tabu search: blending combinatorial optimization and stochastic search for global optimization. Annals of Operations Research – Metaheuristics in Combinatorial Optimization **63**, 153–188 (1996)
31. Baxter, J.: Local optima avoidance in depot location. The Journal of the Operational Research Society **32**(9), 815–819 (1981)
32. Błachut, J.: Tabu search optimization of externally pressurized barrels and domes. Engineering Optimization **39**(8), 899–918 (2007)
33. Boyan, J., Moore, A.: Learning evaluation functions to improve optimization by local search. The Journal of Machine Learning Research **1**, 77–112 (2001)
34. Boyan, J.A., Moore, A.W.: Learning evaluation functions for global optimization and boolean satisfability. In: A. Press (ed.) In Proc. of 15th National Conf. on Artificial Intelligence (AAAI), pp. 3–10 (1998)

35. Braysy, O.: A reactive variable neighborhood search for the vehicle-routing problem with time windows. INFORMS JOURNAL ON COMPUTING **15**(4), 347–368 (2003)
36. Brunato, M., Battiti, R.: RASH: A self-adaptive random search method. In: C. Cotta, M. Sevaux, K. Sörensen (eds.) Adaptive and Multilevel Metaheuristics, *Studies in Computational Intelligence*, vol. 136. Springer (2008)
37. Brunato, M., Battiti, R., Pasupuleti, S.: A memory-based rash optimizer. In: A.F.R.H.H. Geffner (ed.) Proceedings of AAAI-06 workshop on Heuristic Search, Memory Based Heuristics and Their applications, pp. 45–51. Boston, Mass. (2006). ISBN 978-1-57735-290-7
38. Brunato, M., Hoos, H., Battiti, R.: On effectively finding maximal quasi-cliques in graphs. In: V. Maniezzo, R. Battiti, J.P. Watson (eds.) Proc. 2nd Learning and Intelligent Optimization Workshop, LION 2, Trento, Italy, December 2007, *LNCS*, vol. 5313. Springer Verlag (2008)
39. Cerulli, R., Fink, A., Gentili, M., Voss, S.: Metaheuristics comparison for the minimum labelling spanning tree problem. The Next Wave on Computing, Optimization, and Decision Technologies, Springer-Verlag, New York pp. 93–106 (2005)
40. Cerulli, R., Fink, A., Gentili, M., Voß, S.: Extensions of the minimum labelling spanning tree problem. Journal of Telecommunications and Information Technology **4**, 39–45 (2006)
41. Chambers, J., Barnes, J.: New tabu search results for the job shop scheduling problem. The University of Texas, Austin, TX, Technical Report Series ORP96-06, Graduate Program in Operations Research and Industrial Engineering (1996)
42. Chambers, J., Barnes, J.: Reactive search for flexible job shop scheduling. Graduate program in Operations Research and Industrial Engineering, The University of Texas at Austin, Technical Report Series, ORP98-04 (1998)
43. Chelouah, R., Siarry, P.: Tabu search applied to global optimization. European Journal of Operational Research **123**, 256–270 (2000)
44. Chiang, W., Russell, R.: A reactive tabu search metaheuristic for the vehicle routing problem with time windows. INFORMS Journal on Computing **9**, 417–430 (1997)
45. Codenotti, B., Manzini, G., Margara, L., Resta, G.: Perturbation: An efficient technique for the solution of very large instances of the euclidean tsp. INFORMS JOURNAL ON COMPUTING **8**(2), 125–133 (1996)
46. Connolly, D.: An improved annealing scheme for the QAP. European Journal of Operational Research **46**(1), 93–100 (1990)
47. Consoli, S., Darby-Dowman, K., Geleijnse, G., Korst, J., Pauws, S.: Metaheuristic approaches for the quartet method of hierarchical clustering. Tech. rep., Brunel University, West London (2008)
48. Corana, A., Marchesi, M., Martini, C., Ridella, S.: Minimizing multimodal functions of continuous variables with the simulated annealing algorithm. ACM Trans. Math. Softw. **13**(3), 262–280 (1987). DOI http://doi.acm.org/10.1145/29380.29864
49. Cox, B.J.: Object Oriented Programming, an Evolutionary Approach. Addison-Wesley (1990)
50. Crispim, J., Brandao, J.: Reactive tabu search and variable neighborhood descent applied to the vehicle routing problem with backhauls. In: Proceedings of the 4th Metaheuristics International Conference, Porto, MIC, pp. 631–636 (2001)
51. Csöndes, T., Kotnyek, B., Zoltán Szabó, J.: Application of heuristic methods for conformance test selection. European Journal of Operational Research **142**(1), 203–218 (2002)
52. Danese, G., De Lotto, I., Leporati, F., Quaglini, A., Ramat, S., Tecchiolli, G.: A Parallel Neurochip for Neural Networks Implementing the Reactive Tabu Search Algorithm: Application Case Studies. In: Parallel and Distributed Processing, 2001. Proceedings. Ninth Euromicro Workshop on, pp. 273–280 (2001)
53. Delmaire, H., Dıaz, J., Fernandez, E., Ortega, M.: Reactive GRASP and Tabu Search based heuristics for the single source capacitated plant location problem. INFOR **37**, 194–225 (1999)
54. Devarenne, I., Mabed, H., Caminada, A.: Adaptive Tabu Tenure Computation in Local Search. In: Proceedings 8th European Conference on Evolutionary Computation in Combinatorial Optimisation, Napoli, March 2008, *LECTURE NOTES IN COMPUTER SCIENCE*, vol. 4972, p. 1. Springer Verlag (2008)

55. Eiben, A., Horvath, M., Kowalczyk, W., Schut, M.: Reinforcement learning for online control of evolutionary algorithms. In: Brueckner, Hassas, Jelasity, Yamins (eds.) Proceedings of the 4th International Workshop on Engineering Self-Organizing Applications (ESOA'06), LNAI. Springer Verlag (2006). To appear
56. Faigle, U., Kern, W.: Some convergence results for probabilistic tabu search. ORSA Journal on Computing **4**(1), 32–37 (1992)
57. Fescioglu-Unver, N., Kokar, M.: Application of Self Controlling Software Approach to Reactive Tabu Search. In: Self-Adaptive and Self-Organizing Systems, 2008. SASO'08. Second IEEE International Conference on, pp. 297–305 (2008)
58. Fink, A., Voß, S.: Applications of modern heuristic search methods to pattern sequencing problems. Computers and Operations Research **26**(1), 17–34 (1999)
59. Fink, A., Voß, S.: Solving the continuous flow-shop scheduling problem by metaheuristics. European Journal of Operational Research **151**(2), 400–414 (2003)
60. Fleischer, M.A.: Cybernetic optimization by simulated annealing: Accelerating convergence by parallel processing and probabilistic feedback control. Journal of Heuristics **1**(2), 225–246 (1996)
61. Fortin, A., Hail, N., Jaumard, B.: A tabu search heuristic for the dimensioning of 3G multi-service networks. Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE **3** (2003)
62. Fortz, B., , Thorup, M.: Increasing internet capacity using local search. Computational Optimization and Applications **29**(1), 13–48 (2004)
63. Frank, J.: Weighting for godot: Learning heuristics for GSAT. In: PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, vol. 13, pp. 338–343. JOHN WILEY & SONS LTD, USA (1996)
64. Frank, J.: Learning short-term weights for GSAT. In: Proc. INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, vol. 15, pp. 384–391. LAWRENCE ERLBAUM ASSOCIATES LTD, USA (1997)
65. Fukuyama, Y.: Reactive tabu search for distribution load transfer operation. In: Power Engineering Society Winter Meeting, 2000. IEEE, vol. 2 (2000)
66. Genji, T., Oomori, T., Miyazato, K., Hayashi, N., Fukuyama, Y., Co, K.: Service Restoration in Distribution Systems Aiming Higher Utilization Rate of Feeders. In: Proc. of the Fifth Metaheuristics International Conference (MIC2003) (2003)
67. Gent, I., Walsh, T.: Towards an understanding of hill-climbing procedures for sat. In: Proceedings of the Eleventh National Conference on Artificial Intelligence, pp. 28–33. AAAI Press / The MIT Press (1993)
68. Glover, F.: Tabu search - part i. ORSA Journal on Computing **1**(3), 190–260 (1989)
69. Glover, F.: Tabu search - part ii. ORSA Journal on Computing **2**(1), 4–32 (1990)
70. Hamza, K., Mahmoud, H., Saitou, K.: Design optimization of N-shaped roof trusses using reactive taboo search. Applied Soft Computing Journal **3**(3), 221–235 (2003)
71. Hamza, K., Saitou, K., Nassef, A.: Design optimization of a vehicle b-pillar subjected to roof crush using mixed reactive taboo search. pp. 1–9. Chicago, Illinois (2003)
72. Hansen, N.M.P.: Variable neighborhood search. Computers and Operations Research **24**(11), 1097–1100 (1997)
73. Hansen, P., Jaumard, B.: Algorithms for the maximum satisfiability problem. Computing **44**, 279–303 (1990)
74. Hansen, P., Mladenovic, N.: Variable neighborhood search. In: E. Burke, G. Kendall (eds.) Search methodologies: introductory tutorials in optimization and decision support techniques, pp. 211–238. Springer (2005)
75. Hansmann, U.H.E.: Simulated annealing with tsallis weights a numerical comparison. Physica A: Statistical and Theoretical Physics **242**(1-2), 250 – 257 (1997). DOI DOI:10.1016/S0378-4371(97)00203-3
76. Hifi, M., Michrafy, M.: A reactive local search-based algorithm for the disjunctively constrained knapsack problem. Journal of the Operational Research Society **57**(6), 718–726 (2006)

77. Hifi, M., Michrafy, M., Sbihi, A.: A Reactive Local Search-Based Algorithm for the Multiple-Choice Multi-Dimensional Knapsack Problem. Computational Optimization and Applications **33**(2), 271–285 (2006)
78. Hu, B., Raidl, G.R.: Variable neighborhood descent with self-adaptive neighborhood-ordering. In: C. Cotta, A.J. Fernandez, J.E. Gallardo (eds.) Proceedings of the 7th EU/MEeting on Adaptive, Self-Adaptive, and Multi-Level Metaheuristics, malaga, Spain (2006)
79. Hutter, F., Babic, D., Hoos, H., Hu, A.: Boosting Verification by Automatic Tuning of Decision Procedures. Manuscript in preparation (2007)
80. Hutter, F., Hamadi, Y., Hoos, H., Leyton-Brown, K.: Performance prediction and automated tuning of randomized and parametric algorithms. In: Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP 2006). Springer (2006)
81. Hutter, F., Hoos, H., Stutzle, T.: Automatic Algorithm Configuration Based on Local Search. In: PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, vol. 22, p. 1152. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 (2007)
82. Ingber, L.: Very fast simulated re-annealing. Mathl. Comput. Modelling **12**(8), 967–973 (1989)
83. Ishtaiwi, A., Thornton, J.R., A. Anbulagan, S., Pham, D.N.: Adaptive clause weight redistribution. In: Proceedings of the 12th International Conference on the Principles and Practice of Constraint Programming, CP-2006, Nantes, France, pp. 229–243 (2006)
84. Kernighan, B., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell Systems Technical J. **49**, 291–307 (1970)
85. Kincaid, R., Laba, K.: Reactive Tabu Search and Sensor Selection in Active Structural Acoustic Control Problems. Journal of Heuristics **4**(3), 199–220 (1998)
86. Kinney, G., Barnes, J., Colletti, B.: A Reactive Tabu Search algorithm with variable clustering for the Unicost Set Covering Problem. International Journal of Operational Research **2**(2), 156–172 (2007)
87. Kinney Jr, G., Hill, R., Moore, J.: Devising a quick-running heuristic for an unmanned aerial vehicle (UAV) routing system. Journal of the Operational Research Society **56**, 776–786 (2005)
88. Kirkpatrick, S., Jr., C.D.G., Vecchi, M.P.: Optimization by simulated annealing. Science **220**, 671–680 (1983)
89. Laarhoven, P.J.M., Aarts, E.H.L. (eds.): Simulated annealing: theory and applications. Kluwer Academic Publishers, Norwell, MA, USA (1987)
90. Lenne, R., Solnon, C., Stutzle, T., Tannier, E., Birattari, M.: Reactive Stochastic Local Search Algorithms for the Genomic Median Problem. LECTURE NOTES IN COMPUTER SCIENCE **4972**, 266 (2008)
91. Login, A., Areas, S.: Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls. Journal of the Operational Research Society **58**, 1630–1641 (2007)
92. Lourenco, H.: Job-shop scheduling: Computational study of local search and large-step optimization methods. European Journal of Operational Research **83**, 347–364 (1995)
93. Magdon-Ismail, M., Goldberg, M., Wallace, W., Siebecker, D.: Locating hidden groups in communication networks using hidden markov models. LECTURE NOTES IN COMPUTER SCIENCE pp. 126–137 (2003)
94. Martin, O., Otto, S.W., Felten, E.W.: Large-step Markov chains for the traveling salesman problem. Complex Systems **5:3**, 299 (1991)
95. Martin, O., Otto, S.W., Felten, E.W.: Large-step Markov chains for the tsp incorporating local search heuristics. Operation Research Letters **11**, 219–224 (1992)
96. Martin, O.C., Otto, S.W.: Combining simulated annealing with local search heuristics. ANNALS OF OPERATIONS RESEARCH **63**, 57–76 (1996)

97. Mastrolilli, M., Gambardella, L.: MAX-2-SAT: How Good Is Tabu Search in the Worst-Case? In: PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE, pp. 173–178. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 (2004)
98. Morris, P.: The breakout method for escaping from local minima. In: Proceedings of the national conference on artificial intelligence, 11, p. 40. John Wiley & sons LTD, USA (1993)
99. Nahar, S., Sahni, S., Shragowitz, E.: Experiments with simulated annealing. In: DAC '85: Proceedings of the 22nd ACM/IEEE conference on Design automation, pp. 748–752. ACM Press, New York, NY, USA (1985). DOI http://doi.acm.org/10.1145/317825.317977
100. Nahar, S., Sahni, S., Shragowitz, E.: Simulated annealing and combinatorial optimization. In: DAC '86: Proceedings of the 23rd ACM/IEEE conference on Design automation, pp. 293–299. IEEE Press, Piscataway, NJ, USA (1986)
101. Nanry, W., Wesley Barnes, J.: Solving the pickup and delivery problem with time windows using reactive tabu search. Transportation Research Part B **34**(2), 107–121 (2000)
102. Nonobe, K., Ibaraki, T.: A tabu search approach for the constraint satisfaction problem as a general problem solver. European Journal of Operational Research (106), 599–623 (1998)
103. Oomori, T., Genji, T., Yura, T., Takayama, S., Watanabe, T., Fukuyama, Y., Center, T., Inc, K., Hyogo, J.: Fast optimal setting for voltage control equipment considering interconnection of distributed generators. In: Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES, vol. 2 (2002)
104. Osman, I., Wassan, N.: A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. Journal of Scheduling **5**(4), 263–285 (2002)
105. Osman, I.H.: Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Ann. Oper. Res. **41**(1-4), 421–451 (1993)
106. Pasupuleti, S., Battiti, R.: The gregarious particle swarm optimizer (G-PSO). In: Proceedings of the 8th annual conference on Genetic and evolutionary computation, pp. 67–74. ACM New York, NY, USA (2006)
107. Potocnik, P., Grabec, I.: Adaptive self-tuning neurocontrol. Mathematics and Computers in Simulation **51**(3-4), 201–207 (2000)
108. Russell, R., Chiang, W., Zepeda, D.: Integrating multi-product production and distribution in newspaper logistics. Computers and Operations Research **35**(5), 1576–1588 (2008)
109. Russell, R., Urban, T.: Vehicle routing with soft time windows and Erlang travel times. Journal of the Operational Research Society (2007)
110. Ryan, J., Bailey, T., Moore, J., Carlton, W.: Reactive tabu search in unmanned aerial reconnaissance simulations. Proceedings of the 30th conference on Winter simulation pp. 873–880 (1998)
111. Sammoud, O., Sorlin, S., Solnon, C., Ghédira, K.: A comparative study of ant colony optimization and reactive search for graph matching problems. In: J. Gottlieb, G.R. Raidl (eds.) Evolutionary Computation in Combinatorial Optimization – EvoCOP 2006, *LNCS*, vol. 3906, pp. 230–242. Springer Verlag, Budapest (2006)
112. Schuurmans, D., Southey, F., Holte, R.: The exponentiated subgradient algorithm for heuristic boolean programming. In: Proceedings of the international joint conference on artificial intelligence, vol. 17, pp. 334–341. Lawrence Erlbaum associates LTD, USA (2001)
113. Selman, B., Kautz, H.: Domain-independent extensions to GSAT: Solving large structured satisfiability problems. In: Proceedings of IJCAI-93, pp. 290–295 (1993)
114. Selman, B., Kautz, H.: An empirical study of greedy local search for satisfiability testing. In: Proceedings of the eleventh national Conference on Artificial Intelligence (AAAI-93). Washington, D. C. (1993)
115. Selman, B., Kautz, H., Cohen, B.: Noise strategies for improving local search. In: Proceedings of the national conference on artificial intelligence, vol. 12. John Wiley & sons LTD, USA (1994)
116. Selman, B., Kautz, H., Cohen, B.: Local search strategies for satisfiability testing. In: M. Trick, D.S. Johson (eds.) Proceedings of the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability, no. 26 in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, pp. 521–531 (1996)

117. Selman, B., Levesque, H., Mitchell, D.: A new method for solving hard satisfiability problems. In: Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92), pp. 440–446. San Jose, Ca (1992)
118. Shmygelska, A.: Novel Heuristic Search Methods for Protein Folding and Identification of Folding Pathways. Ph.D. thesis, The University of British Columbia (2006)
119. Shmygelska, A.: An extremal optimization search method for the protein folding problem: the go-model example. In: Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation, pp. 2572–2579. ACM Press New York, NY, USA (2007)
120. Shmygelska, A., Hoos, H.: An adaptive bin framework search method for a beta-sheet protein homopolymer model. BMC Bioinformatics **8**(1), 136 (2007)
121. Steiglitz, K., Weiner, P.: Algorithms for computer solution of the traveling salesman problem. In: Proceedings of the Sixth Allerton Conf. on Circuit and System Theory, Urbana, Illinois, pp. 814–821. IEEE (1968)
122. Taillard, E.: Robust taboo search for the quadratic assignment problem. Parallel Computing **17**, 443–455 (1991)
123. Tompkins, D., Hoos, H.: Warped landscapes and random acts of SAT solving. Proc. of the Eighth Intl Symposium on Artificial Intelligence and Mathematics (ISAIM-04) (2004)
124. Tompkins, F.H.D., Hoos, H.: Scaling and probabilistic smoothing: Efficient dynamic local search for sat. In: Proc. Principles and Practice of Constraint Programming - CP 2002 : 8th International Conference, CP 2002, Ithaca, NY, USA, September 9-13, *LNCS*, vol. 2470, pp. 233–248. Springer Verlag (2002)
125. Toune, S., Fudo, H., Genji, T., Fukuyama, Y., Nakanishi, Y.: Comparative study of modern heuristic algorithms to service restoration in distribution systems. IEEE TRANSACTIONS ON POWER DELIVERY **17**(1), 173–181 (2002)
126. Vossen, T., Verhoeven, M., ten Eikelder, H., Aarts, E.: A quantitative analysis of iterated local search. Computing Science Reports 95/06, Department of Computing Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, Netherlands (1995)
127. Voudouris, C., Tsang, E.: Partial constraint satisfaction problems and guided local search. In: Proceedings of 2nd Int. Conf. on Practical Application of Constraint Technology (PACT 96), London, pp. 337–356 (1996)
128. Voudouris, C., Tsang, E.: Guided local search and its application to the traveling salesman problem. European Journal of Operational Research **113**, 469–499 (1999)
129. Wah, B., Wu, Z.: Penalty Formulations and Trap-Avoidance Strategies for Solving Hard Satisfiability Problems. Journal of Computer Science and Technology **20**(1), 3–17 (2005)
130. White, S.: Concepts of scale in simulated annealing. In: AIP Conference Proceedings, vol. 122, pp. 261–270 (1984)
131. Winter, T., Zimmermann, U.: Real-time dispatch of trams in storage yards. Annals of Operations Research (96), 287–315 (2000). URL `http://citeseer.ist.psu.edu/winter00realtime.html`
132. Youssef, S., Elliman, D.: Reactive prohibition-based ant colony optimization (rpaco): A new parallel architecture for constrained clique sub-graphs. In: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, pp. 63–71. IEEE Computer Society Washington, DC, USA (2004)
133. Zennaki, M., Ech-cherif, A., Lamirel, J.: Using Reactive Tabu Search in Semi-supervised Classification. In: Tools with Artificial Intelligence, 2007. ICTAI 2007. 19th IEEE International Conference on, vol. 2 (2007)