OPERATIONS RESEARCH | COMPUTER SCIENCE INTERFACES

Roberto Battiti
Mauro Brunato
Franco Mascia

Reactive Search and Intelligent Optimization

Springer

Reactive Search and Intelligent Optimization

OPERATIONS RESEARCH/COMPUTER SCIENCE INTERFACES

Professor Ramesh Sharda
Oklahoma State University

Prof. Dr. Stefan Voß
Universität Hamburg

Bierwirth / *Adaptive Search and the Management of Logistics Systems*
Laguna & González-Velarde / *Computing Tools for Modeling, Optimization and Simulation*
Stilman / *Linguistic Geometry: From Search to Construction*
Sakawa / *Genetic Algorithms and Fuzzy Multiobjective Optimization*
Ribeiro & Hansen / *Essays and Surveys in Metaheuristics*
Holsapple, Jacob & Rao / *Business Modelling: Multidisciplinary Approaches — Economics, Operational and Information Systems Perspectives*
Sleezer, Wentling & Cude / *Human Resource Development and Information Technology: Making Global Connections*
Voß & Woodruff / *Optimization Software Class Libraries*
Upadhyaya et al / *Mobile Computing: Implementing Pervasive Information and Communications Technologies*
Reeves & Rowe / *Genetic Algorithms — Principles and Perspectives: A Guide to GA Theory*
Bhargava & Ye / *Computational Modeling and Problem Solving In the Networked World: Interfaces in Computer Science & Operations Research*
Woodruff / *Network Interdiction and Stochastic Integer Programming*
Anandalingam & Raghavan / *Telecommunications Network Design and Management*
Laguna & Martí / *Scatter Search: Methodology and Implementations in C*
Gosavi / *Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*
Koutsoukis & Mitra / *Decision Modelling and Information Systems: The Information Value Chain*
Milano / *Constraint and Integer Programming: Toward a Unified Methodology*
Wilson & Nuzzolo / *Schedule-Based Dynamic Transit Modeling: Theory and Applications*
Golden, Raghavan & Wasil / *The Next Wave in Computing, Optimization, and Decision Technologies*
Rego & Alidaee / *Metaheuristics Optimization via Memory and Evolution: Tabu Search and Scatter Search*
Kitamura & Kuwahara / *Simulation Approaches in Transportation Analysis: Recent Advances and Challenges*
Ibaraki, Nonobe & Yagiura / *Metaheuristics: Progress as Real Problem Solvers*
Golumbic & Hartman / *Graph Theory, Combinatorics, and Algorithms: Interdisciplinary Applications*
Raghavan & Anandalingam / *Telecommunications Planning: Innovations in Pricing, Network Design and Management*
Mattfeld / *The Management of Transshipment Terminals: Decision Support for Terminal Operations in Finished Vehicle Supply Chains*
Alba & Martí / *Metaheuristic Procedures for Training Neural Networks*
Alt, Fu & Golden / *Perspectives in Operations Research: Papers in Honor of Saul Gass' 80th Birthday*
Baker et al / *Extending the Horizons: Adv. In Computing, Optimization, and Dec. Technologies*
Zeimpekis et al / *Dynamic Fleet Management: Concepts, Systems, Algorithms & Case Studies*
Doerner et al / *Metaheuristics: Progress in Complex Systems Optimization*
Goel / *Fleet Telematics: Real-time Management & Planning of Commercial Vehicle Operations*
Gondran & Minoux / *Graphs, Dioïds and Semirings: New Models and Algorithms*
Alba & Dorronsoro / *Cellular Genetic Algorithms*
Golden, Raghavan & Wasil / *The Vehicle Routing Problem: Latest Advances and New Challenges*
Raghavan, Golden & Wasil / *Telecommunications Modeling, Policy and Technology*

Roberto Battiti • Mauro Brunato • Franco Mascia

Reactive Search and Intelligent Optimization

Springer

Roberto Battiti
Università Trento
Dip. Ingegneria e Scienza dell'Informazione
Via Sommarive, 14
38100 Trento
Italy
battiti@disi.unitn.it

Franco Mascia
Università Trento
Dip. Ingegneria e Scienza dell'Informazione
Via Sommarive, 14
38100 Trento
Italy
mascia@disi.unitn.it

Mauro Brunato
Università Trento
Dip. Ingegneria e Scienza dell'Informazione
Via Sommarive, 14
38100 Trento
Italy
brunato@disi.unitn.it

http://reactive-search.org/

ISSN: 1387-666X
ISBN: 978-0-387-09623-0 e-ISBN: 978-0-387-09624-7
DOI: 10.1007/978-0-387-09624-7

Library of Congress Control Number: 2008934910

Printed on acid-free paper

springer.com

Contents

Acknowledgments

Considerate la vostra semenza:
fatti non foste a viver come bruti,
ma per seguir virtute e canoscenza.

Li miei compagni fec'io sì aguti,
con questa orazion picciola, al cammino,
che a pena poscia li avrei ritenuti;

e volta nostra poppa nel mattino,
de' remi facemmo ali al folle volo,
sempre acquistando dal lato mancino.

Consider your origins:
you're not made to live as beasts,
but to follow virtue and knowledge.

My companions I made so eager,
with this little oration, of the voyage,
that I could have hardly then contained them;

that morning we turned our vessel,
our oars we made into wings for the foolish flight,
always gaining ground toward the left.

(Dante, Inferno Canto XXVI, translated by Anthony LaPorta)

A book is always, at least in part, a collective intelligence creation, and it is always difficult to acknowledge the countless comments and contributions obtained by different colleagues, friends, and students, while reading preliminary versions, commenting, and discovering mistakes. We thank them all and of course we keep responsibility for the remaining errors. For example, comments on different chapters have been submitted by Holger Hoos, Gabriela Ochoa, Vittorio Maniezzo, Fred Glover, Matteo Gagliolo, Qingfu Zhang, Mark Jelasity, Frank Hutter, Nenad Mladenović, Lyle McGeoch, Christian Blum, Stefan Voss, Alberto Montresor, Nicolò Cesa-Bianchi, Jean-Paul Watson, Michail Lagoudakis, Christine Solnon, and Francesco Masulli. In particular, we thank Elisa Cilia, Paolo Campigotto, and our young LION team members Roberto Zandonati, Marco Cattani, Stefano Teso, Cristina Di Risio, Davide Mottin, and Sanjay Rawat for their fresh initial look at the book topics and their constructive comments. Finally, some of the figures in this book are courtesy of the artistic ability of Marco Dianti (Figs. 1.1, 1.4, and 2.1), and Themis Palpanas (Fig. 2.2).

Last but not least, we thank our partners (Annamaria, Anna, and Elisa) for their patience during the most intense moments related to writing this book.

We are happy to hear from our readers, for signaling opinions, suggesting changes, impressions, novel developments. Please use one of the following emails: battiti@disi.unitn.it, or brunato or mascia. The reactive search online community at

http://reactive-search.org/
http://reactive-search.com/

and the LION laboratory (machine Learning and Intelligent OptimizatioN) Web site at

http://intelligent-optimization.org/

are good places to look for updated information, software, and teaching material related to this book.

Roberto, Mauro, and Franco

Chapter 1
Introduction: Machine Learning for Intelligent Optimization

Errando discitur.
Chi fa falla; e fallando s'impara.
We learn from our mistakes.
(Popular wisdom)

You win only if you aren't afraid to lose.
(Rocky Aoki)

Mistakes are the portals of discovery.
(James Joyce)

This book is about learning for problem solving. If you are not already an expert in the area, let us start with some motivation, just to make sure that we talk about issues that are not far from everybody's human experience.

Human problem solving is strongly connected to learning. Learning takes place when the problem at hand is not well known at the beginning, and its structure becomes clearer and clearer when more experience with the problem is available. For concreteness, let us consider skiing. What distinguishes an expert skier from a novice is that the novice knows some instructions but needs a lot of experience to *fine tune* the techniques (with some falling down into local minima and restarts), while the real expert jumps *seamlessly* from sensorial input to action, without effort and reasoning. The knowledge accumulated from the previous experience has been *compiled into parameters of a neural system* working at very high speed. Think about yourself driving a car, and try to explain in detail how you move your feet when driving: After so many years the knowledge is so hardwired into your neural system that you hardly need any high-level thinking. Of course, this kind of fine tuning of strategies and knowledge compilation into parameters of a dynamical system (our nervous system) is quite natural for us, while more primitive creatures are more rigid in their behavior. Consider a fly getting burnt by an incandescent light bulb. It is deceived by the novel situation because no light bulb was present during the genetic evolution of its species, apart from a very distant one called "sun." You know the rest of the story: The fly will get burnt again and again and again (Fig. 1.1). Lack of lifelong learning and rapid self-tuning can have disastrous consequences. In fact, the human ability of learning and quick reaction to life-threatening dangers and novel contexts has been precious for the survival of our species when our ancestors were living in unfriendly environments such as forests, hunted by animals with higher physical strength. In addition to learning, search by trial-and-error, generation, and test, repeated modifications of solutions by small local changes are also part of human life.

R. Battiti et al., *Reactive Search and Intelligent Optimization*, Operations Research/Computer Science Interfaces Series 45, DOI 10.1007/978-0-387-09624-7_1,

Fig. 1.1 Attracted by the lightbulb and repeatedly burnt: The fly is not intelligent enough to learn from experience and develop an escape strategy. (Image courtesy of Marco Dianti)

What is critical for humans is also critical for many human-developed problem-solving strategies. It is not surprising that many methods for solving problems in artificial intelligence, operations research, and related areas follow the *search* scheme, for example, *searching for an optimal configuration on a tree of possibilities* by adding one solution component at a time, and backtracking if a dead-end is encountered, or *searching by generating a trajectory of candidate solutions* on a landscape defined by the corresponding solution value.

For most of the relevant and difficult problems (look for "computational complexity" and "NP-hardness" in your favorite search engine), researchers now believe that *the optimal solution cannot be found exactly in acceptable computing times*, which grow as a low-order polynomial of the input size. In some cases, similar negative results exist about the possibility of approximating a solution with performance guarantees (*hardness of approximation*). These are well-known negative results established in the last decades of theoretical computer science. Hardness of approximation, in addition to NP-hardness, is a kind of "affirmative action" for heuristics.

Heuristics used to suffer from a bad reputation, citing from Papadimitriou and Steiglitz [205]:

> *6. Heuristics* Any of the <five> approaches above without a formal guarantee of performance can be considered a "heuristic." However *unsatisfactory mathematically*, such approaches are certainly valid in practical situations.

Unfortunately, because of the above-mentioned theoretical results, we are condemned to live with heuristics for very long times, maybe forever, and some effort is required to make them more satisfactory, both from a theoretical and from a practical point of view. This implies that an important part of computer science is becoming experimental: some of its theories cannot be demonstrated by deduction as in mathematics, but have to be subjected to careful experimentation. Terms such

as *experimental algorithmics* and *experimental computer science* have been used to underline this transformation. To make an analogy: There is no mathematical proof of Newton's law of gravitation, it simply describes in a *unified* manner many related physical phenomena, apples falling from a tree and planets orbiting around other planets, and it has been refined and subjected to careful experimentation.

In addition, there is no mathematical proof that the speed of light is independent of the observer, in fact a careful experiment was needed to demonstrate that this is indeed the case. Similarly, serious experimental work can be done in computer science, actually *it must be done* given the mentioned theoretical difficulties in proving results, without being accused of lack of mathematical proofs.

A particular delicate issue in many heuristics is their detailed tuning. Some schemes are not rigid and demand the specification of choices in the detailed algorithm design, or values of internal parameters. Think about our novice skier, its detailed implementation ranging from world champion to ... the writers of this book: The difference in parameter tuning is evident.

Parameter tuning is a crucial issue both in the scientific development and in the practical use of heuristics. In some cases the detailed tuning is executed by a researcher or by a final user. As parameter tuning is user dependent, the reproducibility of the heuristics results is difficult as is comparing different parametric algorithms. Algorithm A can be better than algorithm B if tuned by Roberto, while it can be worse if tuned by Mauro.

In this book we consider some machine learning methods that can be profitably used in order to *automate the tuning* process and make it an integral and *fully documented* part of the algorithm. In particular, the focus is on learning schemes where the accumulated knowledge is compiled into the parameters of the method, or the parameters regulating a dynamical system to search for a solution. These schemes are called *subsymbolic* to differentiate them from high-level reasoning schemes popular in artificial intelligence. In many cases subsymbolic learning schemes work without giving a high-level symbolic explanation. Think about neural networks, and about the champion skier who cannot explain how he allocates forces to the different muscles during a slalom competition.

If learning acts *online*, i.e., while the algorithm is solving an instance of a problem, *task-dependent local properties* can be used by the algorithm to determine the appropriate balance between *diversification* and *intensification*. Deciding whether it is better to look for gold where the other miners are excavating (*intensification/exploitation*) or to go and explore other valleys and uncharted territories (*diversification/exploration*) is an issue that excruciated forty-niners, which we will meet over and over again in the following chapters. Citing for example from [217], "diversification drives the search to examine new regions, and intensification focuses more intently on regions previously found to be good. Intensification typically operates by re-starting from high quality solutions, or by modifying choice rules to favor the inclusion of attributes of these solutions."

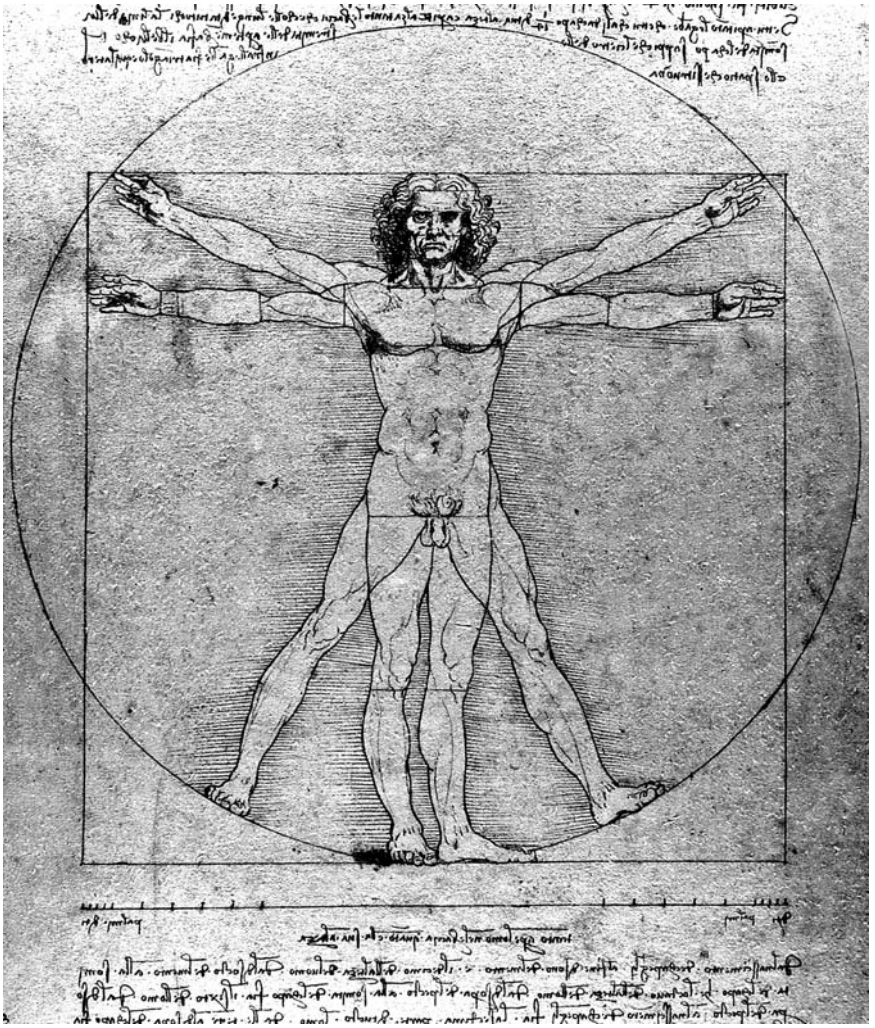

Fig. 1.2 Who invented reactive search? Man is the best example of an effective online learning machine (Leonardo Da Vinci, *Vitruvian Man*, Gallerie dell'Accademia, Venice)

1.1 Parameter Tuning and Intelligent Optimization

As we mentioned, many state-of-the-art heuristics are characterized by a certain number of *choices and free parameters*, whose appropriate setting is a subject that raises issues of research methodology [15, 135, 182]. In some cases the parameters are tuned through a feedback loop that includes *the user as a crucial learning component*: Different options are developed and tested until acceptable results are obtained. The quality of results is not automatically transferred to different instances

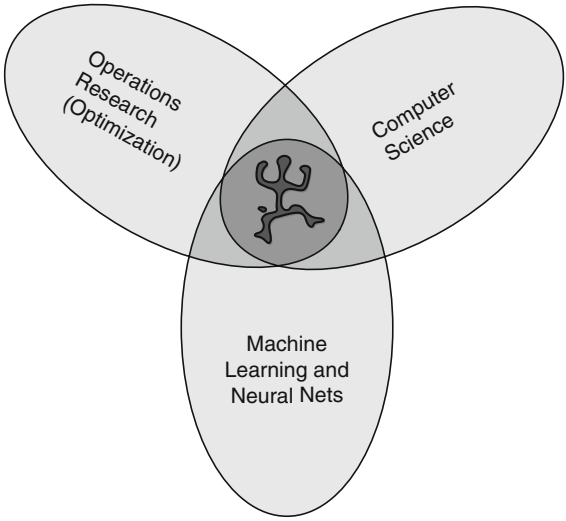

Fig. 1.3 Machine learning and intelligent optimization as the intersection of three disciplines. The logo derived from cave art is related to the first attempts of men to describe themselves in action

and the feedback loop can require a slow "trial and error" process when the algorithm has to be tuned for a specific application. The machine learning community, with significant influx from statistics, developed in the last decades a rich variety of "design principles" that can be used to develop machine learning methods that can be profitably used in the area of parameter tuning for heuristics. In this way one *eliminates the human intervention*. This does not imply higher unemployment rates for researchers. On the contrary, one is now loaded with a heavier task: The algorithm developer must transfer his intelligent expertise into the algorithm itself, a task that requires the exhaustive description of the tuning phase *in the algorithm*. The algorithm complication will increase, but the price is worth paying if the two following objectives are reached:

- *Complete and unambiguous documentation*. The algorithm becomes self-contained and its quality can be judged independently from the designer or specific user. This requirement is particularly important from the scientific point of view, where objective evaluations are crucial. The recent introduction of software archives further simplifies the test and *simple reuse* of heuristic algorithms.
- *Automation*. The time-consuming handmade tuning phase is now substituted by an automated process. Let us note that only the final user will typically benefit from an automated tuning process. On the contrary, the algorithm designer faces a longer and harder development phase, with a possible preliminary phase of exploratory tests, followed by the above described exhaustive documentation of the tuning process when the algorithm is presented to the scientific community.

Reactive search advocates the integration of subsymbolic machine learning techniques into search heuristics for solving complex optimization problems. The word

Fig. 1.4 If algorithms have self-tuning capabilities, complex problem solving does not require technical expertise but is available to a much wider community of final users. (Image courtesy of Marco Dianti)

reactive hints at a ready response to events during the search through an internal *feedback loop for online self-tuning and dynamic adaptation*. In reactive search the past history of the search and the knowledge accumulated while moving in the configuration space is used for self-adaptation in an autonomic manner: The algorithm maintains the internal flexibility needed to address different situations during the search, but the adaptation is automated, and executed while the algorithm runs on a single instance and reflects on its past experience (Fig. 1.2).

Methodologies of interest for reactive search include machine learning and statistics, in particular reinforcement learning and neuro-dynamic programming, active or query learning, and neural networks.

Intelligent optimization, a superset of reactive search, refers to a more extended area of research, including online and offline schemes based on the use of memory, adaptation, incremental development of models, experimental algorithmics applied to optimization, intelligent tuning, and design of heuristics. In some cases the work is at an upper level, where basic methods are properly guided and combined, and the term *metaheuristics* has been proposed in the past. A problem with this term is that the boundary signaled by the "meta" prefix is not always clear: In some cases the intelligence is embedded and the term *intelligent optimization* underlines the holistic

point of view concerned with complete systems rather than with their dissection into parts (the basic components and the meta-level) (Fig. 1.3).

The metaphors for reactive search derive mostly from the individual human experience. Its motto can be "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 reactive search. *Memetic* algorithms share a similar focus on learning, although their concentration is on *cultural evolution*, describing how societies develop over time, more than on the capabilities of a single individual.

Nature and biology-inspired metaphors for optimization algorithms abound in this time frame. It is to some degree surprising that most of them derive from genetics and evolution, or from the emergence of collective behaviors from the interaction of simple living organisms that are mostly hard-wired with little or no learning capabilities. One almost wonders whether this is related to ideological prejudices in the spirit of Jean-Jacques Rousseau, who believed that man was good when in the state of nature but is corrupted by society, or in the spirit of the "evil man against nature." But metaphors lead us astray from our main path: we are strong supporters of a pragmatic approach, an algorithm is effective if it solves a problem in a competitive manner without requiring an expensive tailoring, not because it corresponds to one's favorite elaborate, fanciful, or sexy analogies. Furthermore, at least for a researcher, in most cases an algorithm is of scientific interest if there are ways to analyze its behavior and explain why and when it is effective.

As a final curiosity, let us note that the term *reactive* as "readily responsive to a stimulus" used in our context is not in contrast with proactive as "acting in anticipation of future problems, needs, or changes." In fact, to obtain a reactive algorithm, the designer needs to be proactive by appropriately planning modules into the algorithm so that its capability of autonomous response increases. In other words, reactive algorithms need proactive designers!

1.2 Book Outline

The book does not aim at a complete coverage of the widely expanding research area of heuristics, metaheuristics, stochastic local search, etc. The task would be daunting and bordering on the myth of Sisyphus, condemned by the gods to ceaselessly rolling a rock to the top of a mountain, whence the stone would fall back of its own weight. The rolling stones are in this case caused by the rapid development of new heuristics for many problems, which would render a book obsolete after a short span.

We aim at giving the main principles and at developing some fresh intuition for the approaches. We like mathematics but we also think that hiding the underlying motivations and sources of inspiration takes some color out of the scientific work

("Grau, teurer Freund, ist alle Theorie. Und grün des Lebens goldner Baum" – Gray, my good friend, is all theory, and green is life's own golden tree – Johann Wolfgang von Goethe). On the other hand, pictures and hand-waving can be very dangerous in isolation and we try to avoid these pitfalls by also giving the basic equations when possible, or by at least directing the reader to the bibliographic references for deepening a topic.

The point of view of the book is to look at the zoo of different optimization beasts to underline *opportunities for learning and self-tuning strategies*. A leitmotiv is that seemingly diverse approaches and techniques, when observed from a sufficiently abstract point of view, show the deep interrelationships and *unity*, which is characteristic of science.

The focus is definitely more on methods than on problems. We introduce some problems to make the discussion more concrete or when a specific problem has been widely studied by reactive search and intelligent optimization heuristics.

Intelligent optimization, the application of machine learning strategies in heuristics is a very wide area, and the space in this book dedicated to *reactive search* techniques (online learning techniques applied to search methods) is wider because of personal interest. This book is mainly dedicated to local search and variations, although similar reactive principles can be applied also for other search techniques (for example, tree search).

The structure of most of the following chapters is as follows: (i) the basic issues and algorithms are introduced, (ii) the parameters critical for the success of the different methods are identified, and (iii) opportunities and schemes for the automated tuning of these parameters are presented and discussed.

Let us hit the road.

Chapter 2
Reacting on the Neighborhood

How many shoe-shops should one person visit before making a choice?
There is not a single answer, please specify whether male or female!
(An author of this book who prefers anonymity)

2.1 Local Search Based on Perturbations

A basic problem-solving strategy consists of starting from an initial tentative solution and trying to improve it through repeated small changes. At each repetition the current configuration is slightly modified (*perturbed*), the function to be optimized is tested, the change is kept if the new solution is better, otherwise another change is tried. The function $f(X)$ to be optimized is called with more poetic names in some communities: *fitness* function, *goodness* function, *objective* function.

Figure 2.1 shows an example in the history of bike design. Do not expect historical fidelity here; this book is about algorithms! The first model is a starting solution with a single wheel; it works but it is not optimal yet. The second model is a randomized attempt to add some pieces to the original designs, the situation is worse. One could revert back to the initial model and start other changes. But let us note that, if one insists and proceeds with a second addition, one may end up with the third model, clearly superior from a usability and safety point of view! This real-life story has a lesson: Local search by small perturbations is a tasty ingredient but additional spices are in certain cases needed to obtain superior results. Let us note in passing that everybody's life is an example of an optimization algorithm in action: most of the changes are localized; dramatic changes do happen, but not so frequently. The punctilious reader may notice that the goodness function of our life is not so clearly defined. To this we answer that this book is not about philosophy, let us stop here with far-fetched analogies and go down to the nitty-gritty of the algorithms.

Local search based on perturbing a candidate solution is a first paradigmatic case where simple learning strategies can be applied. Let us define the notation. $\mathscr{X}$ is the search space, $X^{(t)}$ is 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 $\mu_0, \mu_1, ..., \mu_M$ to the current configuration:

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

R. Battiti et al., *Reactive Search and Intelligent Optimization*, Operations Research/Computer Science Interfaces Series 45, DOI 10.1007/978-0-387-09624-7_2,

Fig. 2.1 A local search example: How to build a better bike, from the initial model (*left*) to a worse variation (*middle*), to the final and better configuration (*right*). (Image courtesy of Marco Dianti)

If the search space is given by binary strings with a given length L: $\mathscr{X} = \{0,1\}^L$, the moves can be those changing (or complementing or *flipping*) the individual bits, and therefore M is equal to the string length L.

Local search starts from an admissible configuration $X^{(0)}$ and builds a *search trajectory* $X^{(0)}, ..., X^{(t+1)}$. 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. Let us note that maximization of a function f is the same problem as minimization of $-f$. Like all symmetric situation, this one can create some confusion with the terminology. For example, steepest descent assumes a minimizing point of view, while hill climbing assumes the opposite point of view. In most of the book we will base the discussion on minimization, and *local minima* will be the points that cannot be improved by moving to one of their neighbors. *Local optimum* is a term that can be used both for maximization and minimization.

$$Y \leftarrow \text{IMPROVING-NEIGHBOR}(N(X^{(t)})) \tag{2.1}$$

$$X^{(t+1)} = \begin{cases} Y & \text{if } f(Y) < f(X^{(t)}) \\ X^{(t)} & \text{otherwise (search stops)} \end{cases} \tag{2.2}$$

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.

Local search is surprisingly effective because most combinatorial optimization problems have a very *rich internal structure* relating the configuration X and the f value. The analogy when the input domain is given by real numbers in $\mathbb{R}^n$ is that of a continuously differentiable function $f(x)$ – continuous with continuous derivatives. If one stays in the neighborhood, the change is limited by the maximum value of the derivative multiplied by the displacement. In more dimensions,