1 Parallel Hybrid Metaheuristics

C. COTTA, E-G. TALBI, and E. ALBA

1.1 INTRODUCTION

Finding optimal solutions is in general computationally intractable for many combinatorial optimization problems, e.g., those known as NP-hard [54]. The classical approach for dealing with this fact was the use of approximation algorithms, i.e., relaxing the goal from finding the optimal solution to obtaining solutions within some bounded distance from the former [61]. Unfortunately, it turns out that attainable bounds in practice (that is, at a tenable computational cost) are in general too far from the optimum to be useful in many problems. The days in which researchers struggled to slightly tighten worst-case bounds that were anyway far from practical, or in which finding a PTAS (let alone a FPTAS) for a certain problem was considered a whole success are thus swiftly coming to an end. Indeed, two new alternative lines of attack are being currently used to treat these difficulties. On one hand, a new corpus of theory is being built around the notion of fixed-parameter tractability that emanates from the field of parameterized complexity [40][41]. On the other hand, metaheuristics approaches are being increasingly used nowadays. Quoting [27], the philosophy of these latter techniques is *"try to obtain probably optimal solutions to your problem, for provably good solutions are overwhelmingly hard to obtain"*. See also [27][97] for some prospects on the intersection of both fields (parameterized complexity and metaheuristics).

Focusing on the latter techniques, metaheuristics approaches can be broadly categorized into two major classes: single-solution search algorithms (also known as trajectory-based or local-search based algorithms), and multiple-solution search algorithms (also-known as population-based or –arguably stretching the term– evolutionary algorithms). Examples of the former class are descent local search (LS) [109], greedy heuristics (GH) [81], simulated annealing (SA) [72], or tabu search (TS) [56]. Among the latter class, one can cite genetic algorithms (GA) [63], evolution strategies (ES) [115], genetic programming (GP) [74], ant colony optimization (ACO) [22], scatter search (SS) [55], estimation of distribution algorithms (EDAs) [79], and others. We refer the reader to [23][58][108][116] for good overviews of metaheuristics.

Over the years, interest in metaheuristics has risen considerably among researchers in combinatorial optimization. The flexibility of these techniques makes them prime candidates for tackling both new problems and variants of exiting problems. This fact,

together with their remarkable effectiveness (not to mention the algorithmic beauty of the paradigm), acted as a major attractor of the research community attention. In this expanding scenario, the communal confidence on the potential of these techniques had to endure a crucial test in the middle of the 1990s when some important theoretical results were released. To be precise, the formulation of the so-called *No-Free-Lunch Theorem* (NFL) by Wolpert and Macready [139] made it definitely clear that a search algorithm strictly performs in accordance with the amount and quality of the problem knowledge they incorporate. However, far from undermining the confidence on the usefulness of metaheuristics, this result contributed to render clear the need for adopting problem-dependent strategies within these techniques. Following the terminology of L. Davis [38], who together with other researchers pioneered this line of work long before these results became public, we use the term *hybrid* metaheuristics to denote to these techniques. Here, the term "hybrid" refers to the fact that metaheuristics are typically endowed with problem-dependent knowledge by combining them with other techniques (not necessarily metaheuristic).

The best results found for many practical or academic optimization problems are obtained by hybrid algorithms. Combinations of algorithms such as descent local search, simulated annealing, tabu search, and evolutionary algorithms have provided very powerful search algorithms. The introduction of parallelism plays a major role in these hybrid algorithms. The reason is twofold: (i) the use of parallel programming techniques grants access to utilizing highly-powerful computational platforms (multiprocessor systems, or distributed networks of computers), and (ii) parallelism opens a plethora of possibilities for attaining new hybrid algorithms, of increased search capability.

We will here present an overview of these parallel hybrid metaheuristics, focusing both on algorithmic and computational aspects. Previously, and for the sake of self-containedness, sequential hybrid metaheuristics will be briefly surveyed.

1.2 HISTORICAL NOTES ON HYBRID METAHEURISTICS

The hybridization of metaheuristics in sequential scenarios dates back to the origins of the paradigm itself, although it was initially considered as a minor issue (at least within the majority of the evolutionary computing community). Before the popularization of the NFL Theorem, general-purpose (evolutionary) metaheuristics were believed to be quasi-optimal searchers, globally better than other optimization techniques. All theoretical nuances that our current understanding of the issue have brought (see e.g. [42, 138]) notwithstanding, these results stress the fact that hybridizing (in its broad sense of problem-augmentation) is determinant for achieving top-performance. With hindsight, the group of researchers that advocated for the centrality of this hybridization philosophy were on the right track. Let us start by providing some historical background on the development of the topic.

One of the keystones in the development of hybrid metaheuristics was the conception of *memetic algorithms* [106] (MAs). This term was given birth in the late 1980s to denote a family of metaheuristics that tried to blend several concepts from

tightly separated –at that time– families such as EAs and SA. The adjective *memetic* comes from the term *meme*, coined by R. Dawkins [39] to denote an analogous to the *gene* in the context of cultural evolution. One of the first algorithms to which the MA label was assigned dates from 1988 [106], and was regarded by many as a hybrid of traditional genetic algorithms (GAs) and simulated annealing (SA). Part of the initial motivation was to find a way out of the limitations of both techniques on a well-studied combinatorial optimization problem the MIN EUCLIDEAN TRAVELING SALESMAN (ETSP) problem. Less than a year later, in 1989, Moscato and Norman identified several authors who were also pioneering the introduction of heuristics to improve the solutions before recombining them [59][100] (see other references and the discussion in [95] and [97]). Particularly coming from the GA field, several authors were introducing problem-domain knowledge in a variety of ways. In [95] the denomination of memetic algorithms was introduced for the first time. It was also suggested that *cultural evolution* can be a better working metaphor for these metaheuristics to avoid the biologically constrained thinking that was restricting progress at that time. See also [96][97][98].

L. Davis was other of the champions of this hybrid approach to optimization. He advocated for the inclusion of problem-dependent knowledge in genetic algorithms by means of *ad hoc* representations, or by embedding specialized algorithms within the metaheuristic. He followed this latter approach in [94], where backpropagation [120] was used as a local-searcher within a genetic algorithm aimed at optimizing the weights of a neural network. This algorithm qualifies as a memetic algorithm, and constitutes an approach that has been used by many other researchers in this context. We refer to [25] for a general survey of these previous works, and an empirical comparison of GA, ES, and EDAs hybridized with backpropagation. Davis also studied the combination of GAs with the conspicuous k-means algorithm for classification purposes [69][70]. Regarding *ad hoc* representations, he provided strong empirical evidence on the usefulness of utilizing problem-specific non-binary representations within GAs, e.g., real-coded representations. These guidelines are well-illustrated in [38].

It must be noted that an important part of the metaheuristic community — traditionally associated to the field of operations research (OR)— grew in these years in relative isolation from the EC community. They were thus largely unconstrained by disputable theoretical disquisitions, or by the dominant orthodoxy. This allowed a more pragmatic view of the optimization process, free of biologically-oriented thinking. One of the most distinctive and powerful fruits of this line of research is scatter search. The foundations of this population-based metaheuristic can be traced back to the 1970s in the context of combining decision rules and problem constraints [55]. Despite some methodological differences with other population-based metaheuristics (e.g., SS relies more on deterministic strategies rather than on randomization), SS shares some crucial elements with MAs: both techniques are explicitly concerned with using all problem-knowledge available. This typically results in the use of problem-dependent combination procedures and local-improvement strategies, see [57][78].

The same pragmatic view that gives rise to SS can be found in the origins of asynchronous teams (A-Teams), back in the 1980s [131]. These techniques generalized the notion of population from a set of solutions to a set of any relevant structures for the problem at hand (e.g., solutions and partial solutions may co-exist). This set of structures acts as a shared memory, much like it is done in blackboard systems [104]. A set of agents operate on this shared memory following a predefined data-flow in order to produce new structures. These agents are assimilable to the operators of population-based metaheuristics, and as in MAs and EAs may comprise local-improvers, constructive heuristics, and selection and replacement procedures. See [130] for more details.

1.3 CLASSIFYING HYBRID METAHEURISTICS

It is possible to fit the vast majority of hybrid metaheuristics into some of the major hybrid templates described in the previous section. Nevertheless, it is worth trying to provide a more systematic characterization of hybrid metaheuristics, so that the structure of the algorithm can be easily grasped. Several authors have attempted such classification schemes. We will here address two of these.

E.-G. Talbi [128] proposed a mixed hierarchical-flat classification scheme. The hierarchical component captures the structure of the hybrid, whereas the flat component specifies the features of the algorithms involved in the hybrid. More precisely, The structure of the hierarchical portion of the taxonomy is shown in the upper part of Figure 1.1. At the first level, we may distinguish between low-level and high-level hybridizations. The low-level hybridization addresses the functional composition of a single optimization method. In this hybrid class, a given function of a metaheuristic is replaced by another metaheuristic. On the contrary, in high-level hybrid algorithms, the different metaheuristics are self-contained. We have no direct relationship to the internal workings of a metaheuristic. As to relay hybridization, a set of metaheuristics is applied one after another, each using the output of the previous as its input, acting in a pipeline fashion. On the other hand, teamwork hybridization represents cooperative optimization models, in which we have many parallel cooperating agents, where each agent carries out a search in a solution space.

Four classes are thus derived from this hierarchical taxonomy:

- LRH (Low-level Relay Hybrid): typical examples belonging to this class are MAs or SS where local improvement is performed by some non-general purpose mechanism. For example, in [2], the TSP is tackled using a MA endowed with 2-opt optimization. Also for this problem, [91] defines a LRH hybrid combining simulated annealing with local search. In both cases, the main idea is to embed deterministic local search techniques into the metaheuristic so that the latter explores only local optima.

- LTH (Low-level Teamwork Hybrid): this class typically comprises combinations of metaheuristics with strong exploring capability (e.g., most EAs) with exploitation-oriented metaheuristics (e.g., most single-solution metaheuris-

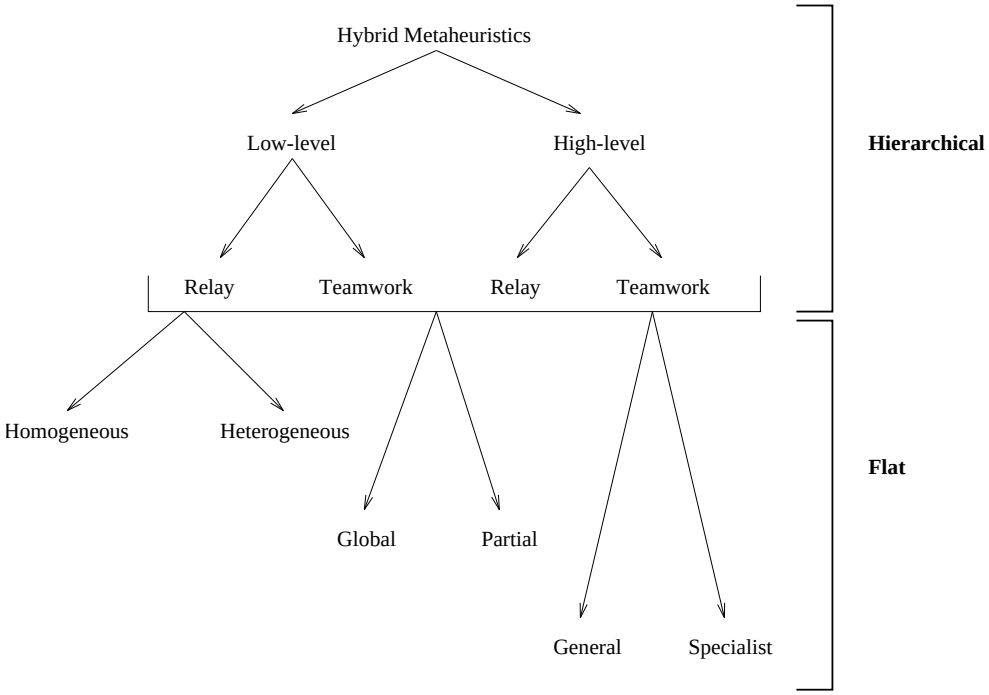

Fig. 1.1 Talbi's classification of hybrid metaheuristics.

tics). In such a combination, the former algorithms will try to optimize locally, while the population-based algorithms will try to optimize globally. For example, when a GA is used as a global optimizer, it can be augmented with HC or SA to perform local search (a typical local-search-based MA). This can be done in a variety of ways. First of all, it is possible to use the local search algorithm as a mutation operator. There are numerous examples of this strategy, using HC [125][134] [68], TS [51][50][71][133], or SA [16][19][137]. This kind of operators is usually qualified as *Lamarckian*, referring to the fact that individuals are replaced by the local optima found after applying local search (contrary to the Baldwin model where the local optima is just used to evaluate the individual). Other possibility is using local-search within crossover, e.g., [86][107]. A similar strategy has been used in non-crossover-based metaheuristics such as ACO [127][124], where local search has been introduced to intensify the search.

- HRH (High-level Relay Hybrid): in a HRH hybrid, self-contained metaheuristics are executed in a sequence. For example, it is well known that EAs are not well suited for fine-tuning structures which are very close to optimal solutions. Instead, the strength of EAs is in quickly locating the high performance re-

gions of vast and complex search spaces. Once those regions are located, it may be useful to apply local search heuristics to the high performance structures evolved by the EA. Many authors have used the idea of HRH hybridization for EAs. In [87][129], the authors introduce respectively SA and TS to improve the population obtained by a GA. In [105], the author introduces HC to improve the results obtained by an ES. In [85], the algorithm proposed starts from simulated annealing and uses GAs to enrich the solutions found. Experiments performed on the graph partitioning problem using the tabu search algorithm exploiting the result found by a GA give better results than a search performed either by the GA, or the tabu search alone [129].

- HTH (High-level Teamwork Hybrid): the HTH scheme involves several self-contained algorithms performing a search in parallel, and cooperating to find an optimum. Ideally HTH would perform at least as well as one algorithm alone, although there might be undesired detrimental interactions (i.e., one algorithm could mislead another one, or let it fall within a fitness trap). The HTH hybrid model has been applied to SA [44], GP [73], ES [135], ACO [89], SS [37] and TS [45] among others.

As to the flat classification, several dichotomies are defined:

- Homogeneous versus heterogeneous: In homogeneous hybrids, all the combined algorithms use the same metaheuristic. Hybrid algorithms such as the island model for GAs [111], belong to this class of hybrids. Arguably, the term 'hybrid' is somewhat forced with homogeneous algorithms, unless different parameters are used in each of these. For example, in HTH based on tabu search, the algorithms may be initialized with different initial solutions, tabu list sizes, etc [136].

  In heterogeneous algorithms, different metaheuristics are used. A heterogeneous HTH algorithm based on genetic algorithms and tabu search has been proposed in [34] to solve a network design problem. The population of the GA is asynchronously updated by multiple tabu search algorithms. The best solutions found by tabu search algorithms build an elite population for the GA.

  The GRASP method (Greedy Randomized Adaptive Search Procedure) may be seen as an iterated heterogenous HRH hybrid, in which local search is repeated from a number of initial solutions generated by randomized greedy heuristic [48][47]. The method is called adaptive because the greedy heuristic takes into account the decisions of the precedent iterations [46].

- Global versus partial: in global hybrids, all the algorithms search in the whole research space. The goal is here to explore the space more thoroughly. All the above mentioned hybrids are *global* hybrids, in the sense that all the algorithms solve the whole optimization problem. A global HTH algorithm based on tabu search has been proposed in [32], where each tabu search task performs a given number of iterations, then broadcasts the best solution. The best of all solutions becomes the initial solution for the next phase.

In partial hybrids, the problem to be solved is decomposed into sub-problems, each one having its own search space. Then, each algorithm is dedicated to the search in one of these sub-spaces. Generally speaking, the sub-problems are all linked with each others, thus involving constraints between optima found by each algorithm. Hence, the algorithms communicate in order to respect these constraints and build a global viable solution to the problem. This approach has been applied for GAs [66], and for SA with TS algorithms [126].

- Specialist versus general: all the above mentioned hybrids are *general* hybrids, in the sense that all the algorithms solve the same target optimization problem. *Specialist* hybrids combine algorithms which solve different problems. An example of such a HTH approach has been developed in [10] to solve the quadratic assignment problem (QAP). A parallel TS is used to solve the QAP, while a GA makes a diversification task, which is formulated as another optimization problem. The frequency memory stores information relative to all the solutions visited by TS. The GA refers to the frequency memory to generate solutions being in unexplored regions.

  Another approach of specialist hybrid HRH heuristics is to use a heuristic to optimize another heuristic, i.e. find the optimal values of the parameters of the heuristic. This approach has been used to optimize SA and noisy methods (NM) by GA [77], ACO by GA [3], and a GA by a GA [121].

C. Cotta [24] proposed another taxonomy for hybrid metaheuristics. This taxonomy has the dichotomy *strong* vs. *weak* as its root. First of all, strong hybridization refers to the placement of problem-knowledge into the core of the algorithm, affecting its internal components. Specifically, a representation of problem solutions allowing the extraction of their most important features, and reproductive operators working on that representation are required. These two aspects are closely related and must be carefully selected to obtain adequate performance. The term 'strong' reflects the tight coupling that exists between the basic model and the included knowledge. Examples of strong hybrid algorithms include metaheuristics using *ad hoc* operators, tailored to deal with the particulars of the problem, e.g., the Edge Assembly Crossover operator defined in [102] for the TSP (involving a greedy repair algorithm), or the different recombination operators defined in [29] for flowshop scheduling (involving the manipulation of non-trivial features of solutions), or in [28] for Bayesian network inference (involving the manipulation of phenotypic information). Strong hybridization also comprises algorithms with non-trivial genotype-to-phenotype mappings. For example, the use of *decoders* [93] in order to produce *non-homogeneous* representations [24][119] (representations that do not cover uniformly the search space, but give preference to promising regions), as in e.g. [20] for the multi-constraint knapsack problem and the set covering problem. Problem-space search [30][123] –the use of metaheuristic with an embedded construction heuristic that is guided through problem-space– also fall within this class.

On the other hand, it is possible to combine algorithms performing different searches, thus resulting in a weak hybrid algorithm. This term tries to capture the fact

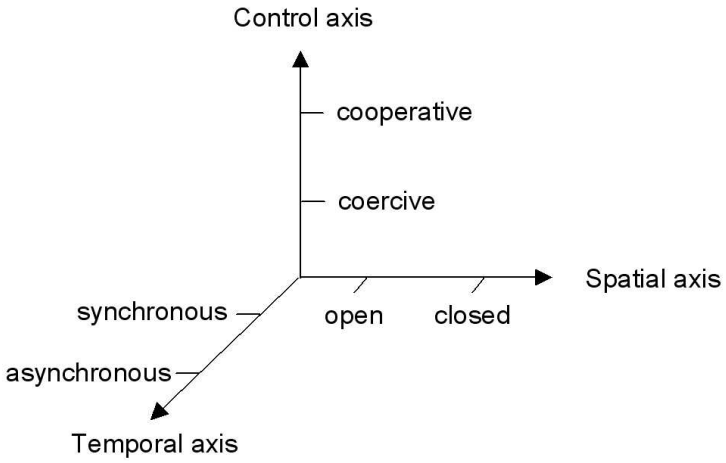

Fig. 1.2 Taxonomy of weak hybrid algorithms.

of hybridization taking place in a higher level and through a well-defined interface, therefore allowing algorithms to retain their identities. This terminology is consistent with the classification of problem-solving strategies in artificial intelligence as *strong* and *weak* methods [92]. It also relates closely to the low-level vs. high-level dichotomy in Talbi's taxonomy, and as in the former it allows further refinement. To be precise, a three-dimensional basis is defined (see Figure 1.2), each of whose axes supports a certain dichotomy, as shown below:

- Control axis: this axis determines whether two algorithms interact at the same level, or there exists a hierarchical relationship between them. The first case corresponds to *cooperative* systems, in which each algorithm performs its own search, with eventual information exchanges. The second case describe *coercive* models, in which one of the algorithms assumes a role of master, imposing the way the second algorithm has to perform its search (by means of either an external control of the parameterization, or by setting exploration constraints).

  Usually, cooperative models are associated to techniques with similar computational requirements, so a useful information exchange can be performed. For example, the homogeneous hybrids mentioned before (e.g., [111][136]), and in general *go-with-the-winners* algorithms [8] [110] are comprised here. On the other hand, coercive models are usually associated to embedded metaheuristics, i.e., high-level relay hybrids. Besides the HRH mentioned before, one can cite the hybridizations of exact and heuristic techniques defined in [31], where branch-and-bound (BnB) is used as a crossover operator.

- Temporal axis: this axis captures those aspects related to when the interaction between algorithms takes place, and what the behavior of the algorithm between interactions is. In principle, there are two major possibilities, *synchronism* and *asynchronism*. The first case comprises those models in which one of the algorithm waits at a synchronization point. Example of such synchronous algorithms are relay hybrids as described before, as well as teamwork hybrids with synchronous interaction (e.g., synchronous migration-based EAs [7]). The second case comprises asynchronous teamwork hybrids, as well as some mixtures of teamwork and relay hybrids (e.g., in [26] a coercive HRH model is asynchronously parallelized, resulting in the simultaneous execution of a GA master, and several BnB slaves; recall that the sequential version of this hybrid would by a synchronous coercive model).

- Spatial axis: this axis allows classifying weak hybrids from the point of view of the search space explored by each of the involved algorithms. To be precise, *open* and *closed* models can be distinguished. In the former, there exist no constraint on the search space considered by each of the algorithms, i.e., any point in the search space is potentially reachable. In closed models exploration is restricted to a certain subspace. This can owe to a external coercion by one of the algorithms, or to an internal feature of the algorithm behavior (e.g., think of decomposition approaches, in which each algorithms is assigned a portion of the search space).

  Local-search-based MAs are typical examples of open models since the local-searcher can potentially visit any point of the search space (obviously, the features of the search landscape are determinant for this purpose; anyway, this does not obey to internal algorithmic reasons, but depends on the input to the algorithm). Another open (synchronous coercive) model can be found in [53], where a GA is used to explore the queue of open problems in a BnB algorithm. On the other hand, the globally optimal forma completion approach defined in [114] is a closed model (during recombination, some feature of solution are fixed, and the subspace of solutions with those features is exhaustively explored to determine the best solution). A related closed model is the dynastically optimal recombination defined in [31], where the subspace comprising all solutions that can be built using the information comprised in a set of parents is exhaustively explored during recombination.

The taxonomies described in this section are aimed at specifying the algorithmic and functional details of a hybrid metaheuristics. Next section will be devoted to discuss some computational details affecting the implementation of these hybrids.

1.4 IMPLEMENTING PARALLEL HYBRID METAHEURISTICS

Parallelism can be brought into hybrid metaheuristics in many ways. A taxonomy for this purpose is actually proposed in [128]. We here concentrate on the parallelization of hybrid metaheuristics on general-purpose computers, since this is the most

widespread computational platform. Parallel hybrids may then be firstly classified according to the different characteristics of the target parallel architecture:

- SIMD versus MIMD: In SIMD (Single Instruction stream, Multiple Data stream) parallel machines, the processors are restricted to execute the same program. They are very efficient in executing synchronized parallel algorithms that contain regular computations and regular data transfers. So, SIMD machines have been used for some parallel hybrid algorithms such a HTH based on tabu search arranged in a 2-dimentional cyclic mesh on a Maspar MPP-1 [43].

  When the computations or the data transfers become irregular or asynchronous, the SIMD machines become much less efficient. In parallel MIMD (Multiple Instruction stream, Multiple data stream), the processors are allowed to perform different types of instructions on different data. HTH hybrids based respectively on tabu search [49][45], simulated annealing, and genetic algorithms [101] have been implemented on networks of transputers.

- Shared-memory versus Distributed-memory: The advantages of parallel hybrids implemented on shared-memory parallel architectures are their simplicity. However, parallel distributed-memory architectures offer a more flexible and fault-tolerant programming platform.

- Homogeneous versus Heterogeneous: Most massively parallel machines (MPP) and cluster of workstations (COW) such as IBM SP/2, Cray T3D, and DEC Alpha-farms are composed of homogeneous processors. The proliferation of powerful workstations and fast communication networks have shown the emergence of heterogeneous network of workstations (NOW) as platforms for high-performance computing (see Figure 1.3).

Parallel hybrid metaheuristics can also be classified according to whether the number and/or the location of work (tasks, data) depend or not on the load state of the target parallel machine:

- Static: this category represents parallel heuristics in which both the number of tasks of the application and the location of work (tasks or data) are generated at compilation time (static scheduling). The allocation of processors to tasks (or data) remains unchanged during the execution of the application regardless of the current state of the parallel machine. Most of the proposed parallel heuristics belong to this class.

  An example of such an approach for TS is presented in [112]. The neighborhood is partitioned in equal size partitions depending on the number of workers, which is equal to the number of processors of the parallel machine. In [18], the number of tasks generated depends on the size of the problem and is equal to n^2, where n is the problem size. In [11], a parallel GA is proposed, where the number of tasks generated is equal to the population size which is fixed at compilation time.

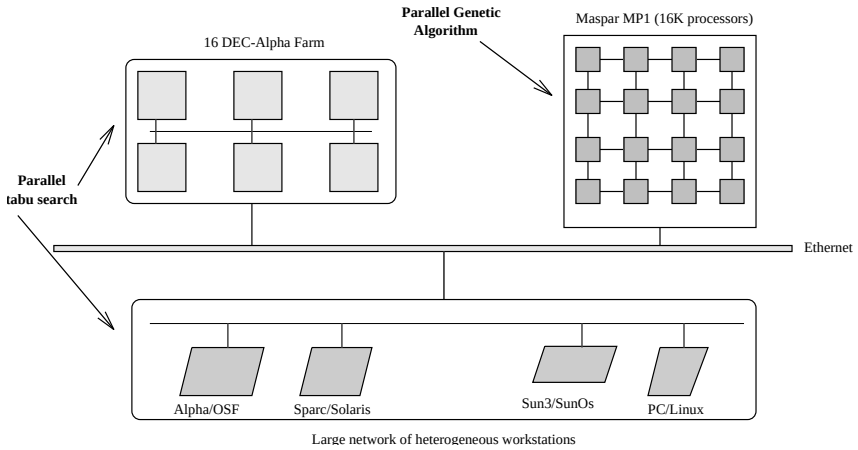

Fig. 1.3 Parallel implementation of heterogeneous HTH algorithms.

When there are noticeable load or power differences between processors, the search time of the static approach presented is derived by the maximum execution time over all processors (presumably on the most highly loaded processor or the least powerful processor). A significant number of tasks are often idle waiting for other tasks to complete their work.

- Dynamic: to improve the performance of parallel static heuristics, dynamic load balancing must be introduced [112][12]. This class represents heuristics for which the number of tasks is fixed at compilation time, but the location of work (tasks, data) is determined and/or changed at run-time. Load balancing requirements are met in [112] by a dynamic redistribution of work between processors. During the search, each time a task finishes its work, it proceeds to a work-demand.

  However, the degree of parallelism in this class of algorithms is not related to load variation in the target machine: when the number of tasks exceeds the number of idle nodes, multiple tasks are assigned to the same node. Moreover, when there are more idle nodes than tasks, some of them will not be used.

- Adaptive: *parallel adaptive programs* are parallel computations with a dynamically changing set of tasks. Tasks may be created or killed as a function of the load state of the parallel machine. A task is created automatically when a node becomes idle. When a node becomes busy, the task is killed. In [10], a parallel adaptive implementation has been proposed for HTH specialist hybrid combining tabu search and genetic algorithms.

1.5 APPLICATIONS OF PARALLEL HYBRID METAHEURISTICS

This section will provide an overview of the numerous applications of parallel metaheuristics. Of course, this overview is far from exhaustive since new applications are being developed continuously. However, it is intended to be illustrative of the practical impact of these optimization techniques. Table 1.1 shows such an illustrative sample of applications. It must be note that we have focused on those applications involving a truly parallel implementation of a hybrid metaheuristic.

For further information about applications of parallel hybrid metaheuristics we suggest querying bibliographical databases or web browsers for the keywords *"parallel hybrid metaheuristic"* or *"parallel hybrid evolutionary algorithm"*.

1.6 CONCLUSIONS

This work has presented a general overview of hybrid metaheuristics, with special emphasis on their utilization on parallel environments. For this purpose, we have surveyed different taxonomies of these techniques, covering both algorithmic and computational aspects of parallel metaheuristics.

As it was mentioned before, pure (i.e., general-purpose, not augmented with problem knowledge) population-based heuristics such as GAs, GP, ES, and ACO are not well suited in general to search in highly-dimensional combinatorial spaces. Hence the need for hybridizing with other techniques in order to achieve practical results. A very common strategy to do so is hybridizing population-based metaheuristics with local search heuristics (open synchronous coercive models – HRH), as it is done in MAs/SS. Most times, this is done in sequential settings, although the authors often indicate in their future work the parallelization of the algorithms. This is an indication of the growing interest in developing parallel hybrid algorithms.

Parallel schemes ideally provide novel ways to parallelize hybrid algorithms by providing new algorithmic models. Furthermore, parallel computing systems offer a way to provide the large computational power needed for tackling the increasing complexity of hard combinatorial optimization problems. Clusters of existing commodity workstations are a low-cost hardware alternative to run parallel metaheuristics. Obviously, issues of heterogeneity and work load appear. Nevertheless, the flexibility of metaheuristics (either parallel or sequential) makes them much less sensitive to these issues than other classical optimization techniques. For this reason, they constitute excellent candidates to approach the hardest optimization tasks in the years to come.

Table 1.1 Some applications of parallel hybrid metaheuristics.

Domain	Problem	References
Combinatorial Optimization	0-1 Integer Linear Programming	[15] [30] [103]
	Boolean Function	[73] [77]
	Capacitated Network Design	[33]
	Graph Coloring	[62]
	Graph Partitioning	[80] [82] [90]
	Independent Set	[60]
	Maximum Cut	[5]
	Multi-commodity Allocation	[35]
	Packing	[75] [76]
	Quadratic Assignment	[16] [18] [64] [65]
	Set Partitioning	[83] [84]
	Traveling Salesman Problem	[9] [26] [77] [88] [90][99] [100][122]
	Vehicle Routing	[13] [117] [126]
Engineering and Electronics	Cell Placement	[14] [17]
	Floorplan Design	[21]
	VLSI Design	[1] [118]
Functional Optimization		[101] [111] [132] [135]
Machine Learning	Neural Networks	[4] [67]
Physics and Chemistry	Molecular Structure	[113] [140]
Scheduling	Resource Allocation	[5]
	Task Allocation	[52]
	Task Scheduling	[112]
Telecommunications	Antenna Placement	[6]
	Error Correcting Codes	[6]
	Frequency Assignment	[5] [36]
	Mapping	[43]

References

1. E. H. L. Aarts, F. M. I. De Bont, J. H. A. Habers, and P. J. M. Van Laarhoven. Parallel implementations of the statistical cooling algorithms. *Integration*, 4:209–238, 1986.

2. Emile H. L. Aarts and Marco G. A. Verhoeven. Genetic local search for the traveling salesman problem. In T. Bäck, D.B. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation*, pages G9.5:1–7. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

3. F. Abbattista, N. Abbattista, and L. Caponetti. An evolutionary and cooperative agent model for optimization. In *IEEE Int. Conf. on Evolutionary Computation ICEC'95*, pages 668–671, Perth, Australia, Dec 1995.

4. P. Adamidis and V. Petridis. Co-operating populations with different evolution behaviours. In *IEEE Int. Conf. on Evolutionary Computation, ICEC'96*, pages 188–191, Nagoya, Japan, May 1996.

5. E. Alba, F. Almeida, M. Blesa, C. Cotta, M. Díaz, I. Dorta, J. Gabarró, J. González, C. León, L. Moreno, J. Petit, J. Roda, A. Rojas, and F. Xhafa. Mallba: A library of skeletons for combinatorial optimisation. In B. Monien and R. Feldman, editors, *Euro-Par 2002 Parallel Processing*, volume 2400 of *Lecture Notes in Computer Science*, pages 927–932. Springer-Verlag, Berlin Heidelberg, 2002.

6. E. Alba, C. Cotta, F. Chicano, and A.J. Nebro. Parallel evolutionary algorithms in telecommunications: Two case studies. In *Proceedings of the CACIC'02*, Buenos Aires, Argentina, 2002.

7. E. Alba and J.M. Troya. Influence of the migration policy in parallel distributed GAs with structured and panmictic populations. *Applied Intelligence*, 12(3):163–181, 2000.

8. D. Aldous and U. Vazirani. "Go with the winners" algorithms. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 492–501, Los Alamitos, CA, 1994. IEEE.

9. T. Asveren and P. Molitor. New crossover methods for sequencing problems. In H-M.Voigt, W. Ebeling, I. Rechenberg, and H-P. Schewefel, editors, *Parallel*

Problem Solving from Nature PPSN4, volume 1141 of *LNCS*, pages 290–299, Dortmund, Germany, Sept 1996. Springer-Verlag.

10. V. Bachelet, Z. Hafidi, P. Preux, and E-G. Talbi. Diversifying tabu search by genetic algorithms. In *INFORMS'98 on Operations Research and Management Sciences meeting*, Montreal, Canada, Apr 1998.

11. V. Bachelet, P. Preux, and E-G. Talbi. Parallel hybrid meta-heuristics : application to the quadratic assignment problem. In *Parallel Optimization Colloquium POC96*, pages 233–242, Versailles, France, Mar 1996.

12. P. Badeau, M. Gendreau, F. Guertin, J-Y. Potvin, and E. Taillard. A parallel tabu search heuristic for the vehicle routing problem with time windows. *RR CRT-95-84, Centre de Recherche sur les Transports, Université de Montréal, Canada*, Dec 1995.

13. P. Badeau, F. Guertin, M. Gendreau, J-Y. Potvin, and E. D. Taillard. A parallel tabu search heuristic for the vehicle routing problem with time windows. *Transportation Research*, 5(2):109–122, 1997.

14. P. Banerjee and M. Jones. A parallel simulated annealing algorithm for standard cell placement on a hypercube computer. In *IEEE Int. Conf. on Computer-Aided Design*, pages 34–37, Santa Clara, California, USA, Nov 1986.

15. R. Bianchini and C. Brown. Parallel genetic algorithms on distributed-memory architectures. Technical Report 436, University of Rochester, Rochester, NY, USA, Aug 1992.

16. D. E. Brown, C. L. Huntley, and A. R. Spillane. A parallel genetic heuristic for the quadratic assignment problem. In *Third Int. Conf. on Genetic Algorithms ICGA'89*, pages 406–415. Morgan Kauffmann, San Mateo, California, USA, July 1989.

17. A. Casotto, F. Romeo, and A. L. Sangiovanni-Vincentelli. A parallel simulated annealing algorithm for the placement of macro-cells. In *IEEE Int. Conf. on Computer-Aided Design*, pages 30–33, Santa Clara, California, USA, Nov 1986.

18. J. Chakrapani and J. Skorin-Kapov. Massively parallel tabu search for the quadratic assignment problem. *Annals of Operations Research*, 41:327–341, 1993.

19. H. Chen and N. S. Flann. Parallel simulated annealing and genetic algorithms: A space of hybrid methods. In Y. Davidor, H-P. Schwefel, and R. Manner, editors, *Third Conf. on Parallel Problem Solving from Nature*, pages 428–436, Jerusalem, Israel, Oct 1994. Springer-Verlag.

20. P. C. Chu. *A genetic algorithm approach for combinatorial optimization problems*. PhD thesis, University of London, London, UK, 1997.

21. J. Cohoon, S. Hedge, W. Martin, and D. Richards. Distributed genetic algorithms for the floorplan design problem. *IEEE Trans. on Computer-Aided Design*, 10(4):483–492, Apr 1991.

22. A. Colorni, M. Dorigo, and V. Maniezzo. Distributed optimization by ant colonies. In *European Conf. on Artificial Life*, pages 134–142. Elsevier Publishing, 1991.

23. D. Corne, M. Dorigo, and F. Glover. *New Ideas in Optimization*. McGraw-Hill, 1999.

24. C. Cotta. A study of hybridisation techniques and their application to the design of evolutionary algorithms. *AI Communications*, 11(3-4):223–224, 1998.

25. C. Cotta, E. Alba, R. Sagarna, and P. Larrañaga. Adjusting weights in artificial neural networks using evolutionary algorithms. In P. Larrañaga and J.A. Lozano, editors, *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*, pages 350–373. Kluwer Academic Publishers, Boston MA, 2001.

26. C. Cotta, J.F. Aldana, A.J. Nebro, and J.M. Troya. Hybridizing genetic algorithms with branch and bound techniques for the resolution of the tsp. In D.W. Pearson, N.C. Steele, and R.F. Albrecht, editors, *Artificial Neural Nets and Genetic Algorithms 2*, pages 277–280, Wien New York, 1995. Springer-Verlag.

27. C. Cotta and P. Moscato. Evolutionary computation: Challenges and duties. In A. Menon, editor, *Frontiers of Evolutionary Computation*, pages 53–72. Kluwer Academic Publishers, Boston MA, 2004.

28. C. Cotta and J. Muruzábal. Towards a more efficient evolutionary induction of bayesian networks. In J.J. Merelo et al., editors, *Parallel Problem Solving From Nature VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 730–739. Springer-Verlag, Berlin, 2002.

29. C. Cotta and J.M. Troya. Genetic forma recombination in permutation flowshop problems. *Evolutionary Computation*, 6(1):25–44, 1998.

30. C. Cotta and J.M. Troya. A hybrid genetic algorithm for the 0-1 multiple knapsack problem. In G.D. Smith, N.C. Steele, and R.F. Albrecht, editors, *Artificial Neural Nets and Genetic Algorithms 3*, pages 251–255, Wien New York, 1998. Springer-Verlag.

31. C. Cotta and J.M. Troya. Embedding branch and bound within evolutionary algorithms. *Applied Intelligence*, 18(2):137–153, 2003.

32. T. D. Crainic, M. Toulouse, and M. Gendreau. *Towards a taxonomy of parallel tabu search algorithms*. Technical Report CRT-933, Centre de Recherche sur les Transports, Université de Montréal, Montréal, Canada, Sep 1993.

33. T. G. Crainic and M. Gendreau. A cooperative parallel tabu search for capacitated network design. Technical Report CRT-97-27, Centre de recherche sur les transports, Université de Montréal, Montréal, Canada, Canada, 1997.

34. T. G. Crainic, A. T. Nguyen, and M. Gendreau. Cooperative multi-thread parallel tabu search with evolutionary adaptive memory. *2nd Int. Conf. on Metaheuristics, Sophia Antipolis, France*, July 1997.

35. T. G. Crainic, M. Toulouse, and M. Gendreau. Synchronous tabu search parallelization strategies for multi-commodity location-allocation with balancing requirements. *OR Spektrum*, 17:113–123, 1995.

36. W. Crompton, S. Hurley, and N.M. Stephen. A parallel genetic algorithm for frequency assignment problems. In *Proceedings of IMACS SPRANN'94*, pages 81–84, 1994.

37. V-D. Cung, T. Mautor, P. Michelon, and A. Tavares. Recherche dispersée parallèle. In *Deuxième Congrés de la Société Francaise de Recherche Opérationnelle et d'Aide à la décision ROADEF'99*, Autrans, France, Jan 1999.

38. L.D. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York NY, 1991.

39. R. Dawkins. *The Selfish Gene*. Clarendon Press, Oxford, 1976.

40. R. Downey and M. Fellows. *Parameterized Complexity*. Springer-Verlag, 1998.

41. R.G. Downey and M.R. Fellows. Fixed-parameter tractability and completeness I: Basic results. *SIAM Journal on Computing*, 24(4):873–921, August 1995.

42. Stefan Droste, Thomas Jansen, and Ingo Wegener. Perhaps not a free lunch but at least a free appetizer. In Wolfgang Banzhaf, Jason Daida, Agoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark Jakiela, and Robert E. Smith, editors, *Proceedings of theFirstGenetic and Evolutionary Computation Conference(GECCO '99)*, pages 833–839, San Francisco CA, 13–17 1999. Morgan Kaufmann Publishers, Inc.

43. I. De Falco, R. Del Balio, and E. Tarantino. Solving the mapping problem by parallel tabu search. In *IASTED Conf.*, Paris, France, 1994.

44. I. De Falco, R. Del Balio, and E. Tarantino. An analysis of parallel heuristics for task allocation in multicomputers. *Computing*, 3, 59 1995.

45. I. De Falco, R. Del Balio, E. Tarantino, and R. Vaccaro. Improving search by incorporating evolution principles in parallel tabu search. In *Int. Conf. on Machine Learning*, pages 823–828, 1994.

46. T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

47. T. A. Feo, M. G. C. Resende, and S. H. Smith. A greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, 42:860–878, 1994.

48. T. A. Feo, K. Venkatraman, and J.F. Bard. A GRASP for a difficult single machine scheduling problem. *Computers and Operations Research*, 18:635–643, 1991.

49. C-N. Fiechter. A parallel tabu search algorithm for large travelling salesman problems. *Discrete Applied Mathematics*, 1994.

50. C. Fleurant and J. A. Ferland. Genetic and hybrid algorithms for graph coloring. *Annals of Operations Research*, 63, 1996.

51. C. Fleurent and J. A. Ferland. Genetic hybrids for the quadratic assignment problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 16:173–188, 1994.

52. S-M. Foo. An approach of combining simulated annealing and genetic algorithm. Master's thesis, University of Illinois, Urbana-Champaign, 1991.

53. Alan P. French, Andrew C. Robinson, and John M. Wilson. Using a hybrid genetic-algorithm/branch and bound approach to solve feasibility and optimization integer programming problems. *Journal of Heuristics*, 7(6):551–564, 2001.

54. M.R. Garey and D.S Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman and Co., San Francisco CA, 1979.

55. F. Glover. Heuristics for integer programming using surrogate constraints. *Decision Sciences*, 8:156–166, 1977.

56. F. Glover. Tabu search - part I. *ORSA Journal of Computing*, 1(3):190–206, 1989.

57. F. Glover. Scatter search and path relinking. In D. Corne, M. Dorigo, and F. Glover, editors, *New Methods in Optimization*, pages 291–316. McGraw-Hill, London, 1999.

58. F. Glover and G. Kochenberger. *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston MA, 2003.

59. M. Gorges-Schleuter. ASPARAGOS: An asynchronous parallel genetic optimization strategy. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 422–427. Morgan Kaufmann Publishers, 1989.

60. M. Hifi. A genetic algorithm-based heuristic for solving the weighted maximum independent set and some equivalent problems. *Journal of the Operational Research Society*, 48(6):612–622, 1997.

61. D.S. Hochbaum. *Approximation algorithms for NP-hard problems*. International Thomson Publishing, 1996.

62. Tad Hogg and Colin P. Williams. Solving the really hard problems with cooperative search. In *Proc. of AAAI93*, pages 231–236, Menlo Park, CA, 1993. AAAI Press.

63. J. H. Holland. *Adaptation in natural and artificial systems*. Michigan Press University, Ann Arbor, MI, USA, 1975.

64. C. L. Huntley and D. E. Brown. Parallel genetic algorithms with local search. Technical Report IPC-TR-90-006, University of Virginia, Charlottesville, VA, USA, July 1991.

65. C. L. Huntley and D. E. Brown. A parallel heuristic for quadratic assignment problems. *Computers and Operations Research*, 18:275–289, 1991.

66. P. Husbands, F. Mill, and S. Warrington. Genetic algorithms, production plan optimisation and scheduling. In H-P. Schewefel and R. Manner, editors, *Parallel Problem Solving From Nature*, volume 496 of *LNCS*, pages 80–84, Dortmund, Germany, Oct 1990. Springer-Verlag.

67. T. Ichimura and Y. Kuriyama. Learning of neural networks with parallel hybrid ga using a royal road function. In *1998 IEEE International Joint Conference on Neural Networks*, volume 2, pages 1131–1136, New York, NY, 1998. IEEE.

68. P. Jog, J. Y. Suh, and D. Van Gucht. The effects of population size, heuristic crossover and local improvement on a genetic algorithm for the traveling salesman problem. In *3rd Int. Conf. Genetic Algorithms*. Morgan Kaufmann, USA, 1989.

69. J.D. Kelly and L. Davis. A hybrid genetic algorithm for classification. In *Proceedings of the Twelveth International Joint Conference on Artificial Intelligence*, pages 645–650, Sidney, Australia, 1991. Morgan Kaufmann.

70. J.D. Kelly and L. Davis. Hybridizing the genetic algorithm and the k–nearest neighbors classification algorithm. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 377–383, San Diego, CA, 1991. Morgan Kaufmann.

71. H. Kim, Y. Hayashi, and K. Nara. The performance of hybridized algorithm of genetic algorithm simulated annealing and tabu search for thermal unit maintenance scheduling. In *2nd IEEE Conf. on Evolutionary Computation ICEC'95*, pages 114–119, Perth, Australia, Dec 1995.

72. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

73. J. Koza and D. Andre. Parallel genetic programming on a network of transputers. Technical Report CS-TR-95-1542, Stanford University, 1995.

74. J. R. Koza. *Genetic programming*. MIT Press, Cambridge, USA, 1992.

75. B. Kroger, P. Schwenderling, and O. Vornberger. Parallel genetic packing of rectangles. In H-P. Schwefel and R. Manner, editors, *Parallel Problem solving from nature*, volume 496 of *LNCS*, pages 160–164, Dortmund, Germany, Oct 1990. Springer-Verlag.

76. B. Kroger, P. Schwenderling, and O. Vornberger. Genetic packing of rectangles on transputers. In P. Welch et al., editor, *Transputing 91*. IOS Press, 1991.

77. M. Krueger. *Méthodes d'analyse d'algorithmes d'optimisation stochastiques à l'aide d'algorithmes génétiques*. PhD thesis, Ecole Nationale Supèrieure des Télécommunications, Paris, France, Dec 1993.

78. M. Laguna and R. Martí. *Scatter Search. Methodology and Implementations in C*. Kluwer Academic Publishers, Boston MA, 2003.

79. P. Larrañaga and J.A. Lozano. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation*. Kluwer Academic Publishers, Boston MA, 2001.

80. G. Von Laszewski and H. Muhlenbein. Partitioning a graph with parallel genetic algorithm. In H-P. Schwefel and R. Manner, editors, *Parallel Problem solving from nature*, volume 496 of *LNCS*, pages 165–169, Dortmund, Germany, Oct 1990. Springer-Verlag.

81. E. L. Lawler. *Combinatorial optimization: Networks and matroids*. Holt, Rinehart and Winston, USA, 1976.

82. K-G. Lee and S-Y. Lee. Efficient parallelization of simulated annealing using multiple markov chains: An application to graph partitioning. In T. N. Mudge, editor, *Int. Conf. on Parallel Processing*, pages 177–180. CRC Press, 1992.

83. D. Levine. *A parallel genetic algorithm for the set partitioning problem*. PhD thesis, Argonne National Laboratory, Illinois Institute of Technology, Argonne, USA, May 1994.

84. D. Levine. A parallel genetic algorithm for the set partitioning problem. In I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics: Theory & Applications*, pages 23–35. Kluwer Academic Publishers, 1996.

85. F. T. Lin, C. Y. Kao, and C. C. Hsu. Incorporating genetic algorithms into simulated annealing. *Proc. of the Fourth Int. Symp. on AI*, pages 290–297, 1991.

86. M. Lozano, F. Herrera, N. Krasnogor, and D. Molina. Real-coded memetic algorithms with crossover hill-climbing. *Evolutionary Computation*, 12(3), 2004.

87. S. W. Mahfoud and D. E. Goldberg. Parallel recombinative simulated annealing: A genetic algorithm. *Parallel computing*, 21:1–28, 1995.

88. M. Malek, M. Guruswamy, M. Pandya, and H. Owens. Serial and parallel simulated annealing and tabu search algorithms for the traveling salesman problem. *Annals of Operations Research*, 21:59–84, 1989.

89. C. E. Mariano and E. Morales. A multiple objective ant-q algorithm for the design of water distribution irrigation networks. In *First International Workshop on Ant Colony Optimization ANTS'98*, Bruxelles, Belgique, Oct 1998.

90. O. C. Martin and S. W. Otto. Combining simulated annealing with local search heuristics. *Annals of Operations Research*, 63:57–75, 1996.

91. O. C. Martin, S. W. Otto, and E. W. Felten. Large-step markov chains for the TSP: Incorporating local search heuristics. *Operation Research Letters*, 11:219–224, 1992.

92. Z. Michalewicz. A hierarchy of evolution programs: An experimental study. *Evolutionary Computation*, 1(1):51–76, 1993.

93. Z. Michalewicz. Decoders. In T. Bäck, D.B. Fogel, and Z. Michalewicz, editors, *Handbook of Evolutionary Computation*, pages C5.3:1–3. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997.

94. D. Montana and L. Davis. Training feedforward neural networks using genetic algorithms. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 762–767, San Mateo, CA, 1989. Morgan Kaufmann.

95. P. Moscato. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Technical Report Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA, 1989.

96. P. Moscato. Memetic algorithms: A short introduction. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 219–234. McGraw-Hill, 1999.

97. P. Moscato and C. Cotta. A gentle introduction to memetic algorithms. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 105–144. Kluwer Academic Publishers, Boston MA, 2003.

98. P. Moscato, C. Cotta, and A. Mendes. Memetic algorithms. In G.C. Onwubolu and B.V. Babu, editors, *New Optimization Techniques in Engineering*, pages 53–85. Springer-Verlag, Berlin Heidelberg, 2004.

99. P. Moscato and M.G. Norman. A memetic approach for the traveling salesman problem: Implementation of a computational ecology for combinatorial optimization on message-passing systems. In M. Valero, E. Onate, M. Jane, J.L.

Larriba, and B. Suarez, editors, *Parallel Computing and Transputer Applications*, pages 187–194. IOS Press, Amsterdam, 1992.

100. H. Mühlenbein, M. Gorges-Schleuter, and O. Krämer. Evolution Algorithms in Combinatorial Optimization. *Parallel Computing*, 7:65–88, 1988.

101. H. Muhlenbein, M. Schomisch, and J. Born. The parallel genetic algorithm as function optimizer. *Parallel Computing*, 17:619–632, 1991.

102. Y. Nagata and Sh. Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. In Th. Bäck, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms, East Lansing, EUA*, pages 450–457, San Mateo, CA, 1997. Morgan Kaufmann.

103. S. Niar and A. Freville. A parallel tabu search algorithm for the 0-1 multidimensional knapsack problem. In *Int. Parallel Processing Symposium*, Geneva, Switzerland, 1997. IEEE Society.

104. H.P. Nii. Blackboard systems, part one: The blackboard model of problem solving and the evolution of blackboard architectures. *AI Magazine*, 7(2):38–53, 1986.

105. V. Nissen. Solving the quadratic assignment problem with clues from nature. *IEEE Transactions on Neural Networks*, 5(1):66–72, Jan 1994.

106. M.G. Norman and P. Moscato. A competitive and cooperative approach to complex combinatorial search. Technical Report Caltech Concurrent Computation Program, Report. 790, California Institute of Technology, Pasadena, California, USA, 1989. Expanded version published at the Proceedings of the 20th Informatics and Operations Research Meeting, Buenos Aires (20th JAIIO), Aug. 1991, pp. 3.15–3.29.

107. U-M. O'Reilly and F. Oppacher. Hybridized crossover-based techniques for program discovery. In *IEEE Int. Conf. on Evolutionary Computation ICEC'95*, pages 573–578, Perth, Australia, Dec 1995.

108. I. H. Osman and G. Laporte. Metaheuristics: A bibliography. *Annals of Operations Research*, 63:513–628, 1996.

109. C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

110. M. Peinado and T. Lengauer. Parallel "go with the winners algorithms" in the LogP Model. In IEEE Computer Society Press, editor, *Proceedings of the 11th International Parallel Processing Symposium*, pages 656–664. Los Alamitos, California, 1997.

111. C. B. Pettey, M. R. Leuze, and J. J. Grefenstette. A parallel genetic algorithm. *Proc. of the Second Int. Conf. on Genetic algorithms, MIT, Cambridge*, pages 155–161, July 1987.

112. S. C. S. Porto and C. Ribeiro. Parallel tabu search message-passing synchronous strategies for task scheduling under precedence constraints. *Journal of Heuristics*, 1(2):207–223, 1996.

113. W.J. Pullan. Structure prediction of benzene clusters using a genetic algorithm. *Journal of Chemical Information and Computer Sciences*, 37(6):1189–1193, 1997.

114. N.J. Radcliffe and P.D. Surry. Fitness Variance of Formae and Performance Prediction. In L.D. Whitley and M.D. Vose, editors, *Proceedings of the Third Workshop on Foundations of Genetic Algorithms*, pages 51–72, San Francisco, 1994. Morgan Kaufmann.

115. I. Rechenberg. *Evolutionsstrategie : Optimierung technischer systeme nach prizipien der biologischen evolution*. Formann-Holzboog Verlag, Stuttgart, Germany, 1973.

116. C. R. Reeves. *Modern heuristic techniques for combinatorial problems*. Black Scientific Publications, Oxford, UK, 1993.

117. C. Rego and C. Roucairol. A parallel tabu search algorithm for the vehicle routing problem. In I. H. Osman and J. P. Kelly, editors, *Meta-Heuristics: Theory and Applications*, pages 253–295. Kluwer, Norwell, MA, USA, 1996.

118. J. S. Rose, D. R. Blythe, W. M. Snelgrove, and Z. G. Vranecic. Fast, high quality VLSI placement on a MIMD multiprocessor. In *IEEE Int. Conf. on Computer-Aided Design*, pages 42–45, Santa Clara, Nov 1986.

119. F. Rothlauf. *Representations for Genetic and Evolutionary Algorithms*. Studies in Fuzziness and Soft Computing. Physica-Verlag, Heidelberg New York, 2002.

120. D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by backpropagating errors. *Nature*, 323:533–536, 1986.

121. K. Shahookar and P. Mazumder. A genetic approach to standard cell placement using meta-genetic parameter optimization. *IEEE Trans. on Computer-Aided Design*, 9(5):500–511, May 1990.

122. P. M. Sloot, J. A. Kandorp, and A. Schoneveld. Dynamic complex systems: A new approach to parallel computing in computational physics. Technical Report TR-CS-95-08, University of Amsterdam, Netherlands, 1995.

123. R.H. Storer, S.D. Wu, and R. Vaccari. New search spaces for sequencing problems with application to job-shop scheduling. *Management Science*, 38:1495–1509, 1992.

124. T. Stutzle and H. H. Hoos. The MAX-MIN ant system and local search for combinatorial optimization problems: Towards adaptive tools for global optimization. In *2nd Int. Conf. on Metaheuristics*, pages 191–193, Sophia Antipolis, France, July 1997. INRIA.

125. J. Y. Suh and D. Van Gucht. Incorporating heuristic information into genetic search. In *2rd Int. Conf. Genetic Algorithms*, pages 100–107. Lawrence Erlbaum Associates, USA, 1987.

126. E. Taillard. Parallel iterative search methods for vehicle routing problem. *Networks*, 23:661–673, 1993.

127. E. D. Taillard and L. Gambardella. Adaptive memories for the quadratic assignment problem. Technical Report 87-97, IDSIA, Lugano, Switzerland, 1997.

128. E.-G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, 8(5):541–564, 2002.

129. E. G. Talbi, T. Muntean, and I. Samarandache. Hybridation des algorithmes génétiques avec la recherche tabou. In *Evolution Artificielle EA94*, Toulouse, France, Sep 1994.

130. S. Talukdar, L. Baerentzen, A. Gove, and P. de Souza. Asynchronous teams: Cooperation schemes for autonomous agents. *Journal of Heuristics*, 4(4):295–321, 1998.

131. S.N. Talukdar, S.S. Pyo, and T.Giras. Asynchronous procedures for parallel processing. *IEEE Transactions on PAS*, PAS-102(11), 1983.

132. R. Tanese. Parallel genetic algorithms for a hypercube. *Proc. of the Second Int. Conf. on Genetic algorithms, MIT, Cambridge, MA, USA*, pages 177–183, July 1987.

133. J. Thiel and S. Voss. Some experiences on solving multiconstraint zero-one knapsack problems with genetic algorithms. *INFOR*, 32(4):226–242, Nov 1994.

134. N. L. J. Ulder, E. H. L. Aarts, H-J. Bandelt, P. J. M. Van Laarhoven, and E. Pesch. Genetic local search algorithms for the traveling salesman problem. In H-P. Schewefel and R. Manner, editors, *Parallel Problem Solving From Nature*, volume 496 of *LNCS*, pages 109–116, Dortmund, Germany, Oct 1990. Springer-Verlag.

135. H-M. Voigt, J. Born, and I. Santibanez-Koref. Modelling and simulation of distributed evolutionary search processes for function optimization. In H-P. Schwefel and R. Manner, editors, *Parallel Problem solving from nature*, volume 496 of *LNCS*, pages 373–380, Dortmund, Germany, Oct 1990. Springer-Verlag.

136. S. Voss. *Network optimization problems*, chapter Tabu search: Applications and prospects, pages 333–353. World Scientific, USA, 1993.

137. L-H. Wang, C-Y. Kao, M. Ouh-young, and W-C. Chen. Molecular binding: A case study of the population-based annealing genetic algorithms. In *IEEE Int. Conf. on Evolutionary Computation ICEC'95*, pages 50–55, Perth, Australia, Dec 1995.

138. D. Whitley. Functions as permutations: Implications for no free lunch, walsh analysis and summary statistics. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature – PPSN VI*, pages 169–178, Berlin, 2000. Springer.

139. D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

140. C.R. Zacharias, M.R. Lemes, and A.D. Pino. Combining genetic algorithm and simulated annealing: a molecular geometry optimization study. *THEOCHEM-Journal of Molecular Structure*, 430(29–39), 1998.