Combining Metaheuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification

Jakob Puchinger and Günther R. Raidl

Institute of Computer Graphics and Algorithms
Vienna University of Technology, Vienna, Austria
{puchinger|raidl}@ads.tuwien.ac.at

Abstract. In this survey we discuss different state-of-the-art approaches of combining exact algorithms and metaheuristics to solve combinatorial optimization problems. Some of these hybrids mainly aim at providing optimal solutions in shorter time, while others primarily focus on getting better heuristic solutions. The two main categories in which we divide the approaches are collaborative versus integrative combinations. We further classify the different techniques in a hierarchical way. Altogether, the surveyed work on combinations of exact algorithms and metaheuristics documents the usefulness and strong potential of this research direction.

1 Introduction

Hard combinatorial optimization problems (COPs) appear in a multitude of real-world applications, such as routing, assignment, scheduling, cutting and packing, network design, protein alignment, and many other fields of utmost economic, industrial and scientific importance. The available techniques for COPs can roughly be classified into two main categories: *exact* and *heuristic* methods. Exact algorithms are guaranteed to find an optimal solution and to prove its optimality for every instance of a COP. The run-time, however, often increases dramatically with the instance size, and often only small or moderately-sized instances can be practically solved to provable optimality. In this case, the only possibility for larger instances is to trade optimality for run-time, yielding heuristic algorithms. In other words, the guarantee of finding optimal solutions is sacrificed for the sake of getting good solutions in a limited time.

Two independent heterogeneous streams, coming from very different scientific communities, had significant success in solving COPs:

- *Integer Programming (IP)* as an exact approach, coming from the operations research community and based on the concepts of linear programming [11].
- Local search with various extensions and independently developed variants, in the following called *metaheuristics*, as a heuristic approach.

Among the exact methods are branch-and-bound (B&B), dynamic programming, Lagrangian relaxation based methods, and linear and integer programming based methods, such as branch-and-cut, branch-and-price, and branch-and-cut-and-price [30].

Metaheuristics include, among others, simulated annealing [21], tabu search [18], iterated local search [26], variable neighborhood search [20], and various population-based models such as evolutionary algorithms [3], scatter search [19], memetic algorithms [28], and various estimation of distribution algorithms [24].

This work is supported by the Austrian Science Fund (FWF) under grant P16263-N04.

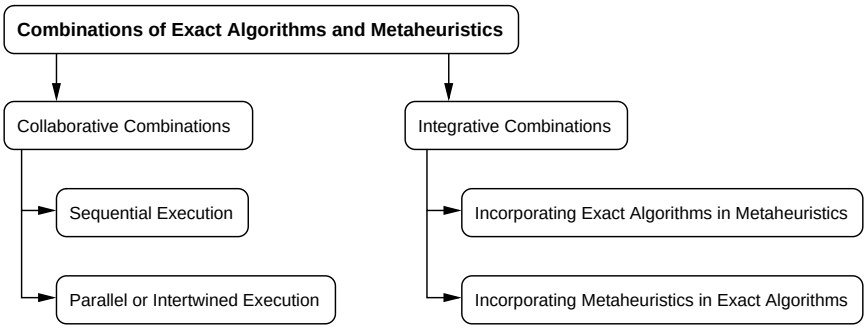

Fig. 1. Major classification of exact/metaheuristic combinations.

Recently there have been very different attempts to combine ideas and methods from these two scientific streams. Dumitrescu and Stützle [13] describe existing combinations, focusing on local search approaches that are strengthened by the use of exact algorithms. In their survey they concentrate on integration and exclude obvious combinations such as preprocessing.

Here, we present a more general classification of existing approaches combining exact and metaheuristic algorithms for combinatorial optimization. We distinguish the following two main categories:

- *Collaborative Combinations*: By collaboration we mean that the algorithms exchange information, but are not part of each other. Exact and heuristic algorithms may be executed sequentially, intertwined or in parallel.
- *Integrative Combinations*: By integration we mean that one technique is a subordinate embedded component of another technique. Thus, there is a distinguished master algorithm, which can be either an exact or a metaheuristic algorithm, and at least one integrated slave.

In the following sections this classification is further refined and examples from the literature are presented, reflecting the current state-of-the-art. Figure 1 gives an overview of this classification.

2 Collaborative Combinations

The different algorithms and approaches described in this section have in common that they are top-level combinations of metaheuristics and exact techniques; no algorithm is contained in another. We further distinguish whether the algorithms are executed sequentially or in an intertwined or even parallel way.

2.1 Sequential Execution

Either the exact method is executed as a kind of preprocessing before the metaheuristic, or vice-versa. Sometimes, it is difficult to say if the first technique is used as initialization of the second, or if the second is a postprocessing of the solution(s) generated by the first.

Clements et al. [7] propose a column generation approach in order to solve a production-line scheduling problem. Each feasible solution of the problem consists of a line-schedule for each production line. First, the squeaky wheel optimization (SWO) heuristic is used to generate feasible solutions to the problem. SWO is a heuristic using a greedy algorithm to construct a solution, which is then analyzed

in order to find the problematic elements. Higher priorities, such that these elements are considered earlier by the greedy algorithm, are assigned to them, and the process restarts until a termination condition is reached. SWO is called several times in a randomized way in order to generate a set of diverse solutions. In the second phase, the line-schedules contained in these solutions are used as columns of a set partitioning formulation for the problem, which is solved using MINTO. This process always provides a solution which is at least as good as, but usually better than the best solution devised by SWO. Reported results indicate that SWO performs better than a tabu-search algorithm.

Applegate et al. [2] propose an approach for finding near-optimal solutions to the traveling salesman problem. They derive a set of diverse solutions by multiple runs of an iterated local search algorithm. The edge-sets of these solutions are merged and the traveling salesman problem is finally solved to optimality on this strongly restricted graph. In this way a solution is achieved that is typically superior to the best solution of the iterated local search.

Klau et al. [22] follow a similar idea and combine a memetic algorithm with integer programming to heuristically solve the prize-collecting Steiner tree problem. The proposed algorithmic framework consists of three parts: extensive preprocessing, a memetic algorithm, and an exact branch-and-cut algorithm applied as post-optimization procedure to the merged final solutions of the memetic algorithm.

Plateau et al. [31] combine interior point methods and metaheuristics for solving the multiconstrained knapsack problem. The first part is an interior point method with early termination. By rounding and applying several different ascent heuristics, a population of different feasible candidate solutions is generated. This set of solutions is then used as initial population for a path-relinking (scatter search) algorithm. Extensive computational experiments are performed on standard multiconstrained knapsack benchmark instances. Obtained results show that the presented combination is a promising research direction.

Sometimes, a relaxation of the original problem is solved to optimality and the obtained solution is repaired to act as a promising starting point for a subsequent metaheuristic. Often, the linear programming (LP) relaxation is used for this purpose, and only a simple rounding scheme is needed. For example, Feltl and Raidl [36] solve the generalized assignment problem using a hybrid genetic algorithm (GA). The LP-relaxation of the problem is solved using CPLEX and its solution is used by a randomized rounding procedure to create a population of promising integral solutions. These solutions are, however, often infeasible; therefore, randomized repair and improvement operators are additionally applied, yielding an even more meaningful initial population for the GA. Reported computational experiments suggest that this type of LP-based initialization is effective.

Vasquez and Hao [43] heuristically solve the multiconstrained knapsack problem by reducing and partitioning the search space via additional constraints that fix the total number of items to be packed. The bounds for these constraints are calculated by solving a modified LP-relaxation of the multiconstrained knapsack problem. For each remaining part of the search space, parallel tabu-search is finally performed starting with a solution derived from the LP-relaxation of the partial problem. This hybrid algorithm yields excellent results also for large benchmark instances with up to 2 500 items and 100 constraints.

Lin et al. [25] describe an exact algorithm for generating the minimal set of affine functions that describes the value function of the finite horizon partially observed Markov decision process. In the first step a GA is used to generate a set Γ of witness points, which is as large as possible. In the second step a component-wise

http://www.isye.gatech.edu/faculty/Martin_Savelsbergh/software
http://www.ilog.com

domination procedure is performed in order to eliminate redundant points in Γ. The set generated so far does, in general, not fully describe the value function. Therefore, a Mixed Integer Program (MIP) is solved to generate the missing points in the final third step of the algorithm. Reported results indicate that this approach requires less time than some other numerical procedures.

Another kind of sequential combination of B&B and a GA is described by Nagar et al. [29] for a two-machine flowshop scheduling problem in which solution candidates are represented as permutations of jobs. Prior to running the GA B&B is executed down to a predetermined depth k and suitable bounds are calculated and recorded at each node of the explicitly stored B&B tree. During the execution of the GA the partial solutions up to position k are mapped onto the correct tree node. If the bounds indicate that no path below this node can lead to an optimal solution, the permutation is subjected to a mutation operator that has been specifically designed to change the early part of the permutation in a favorable way.

Tamura et al. [40] tackle a job-shop scheduling problem and start from its IP formulation. For each variable, they take the range of possible values and partition it into a set of subranges, which are then indexed. The chromosomes of the GA are defined so that each position represents a variable, and its value corresponds to the index of one of the subranges. The fitness of a chromosome is calculated using Lagrangian relaxation to obtain a bound on the optimal solution subject to the constraints that the values of the variables fall within the correct ranges. When the GA terminates, an exhaustive search of the region identified as the most promising is carried out to produce the final solution.

2.2 Parallel or Intertwined Execution

Instead of a strictly sequential batch approach, exact and heuristic algorithms may also be executed in a parallel or intertwined way. Such peer-to-peer combinations of exact/heuristic techniques are less frequent. An interesting framework for this purpose was proposed by Talukdar et al. [38, 39] with the so-called *asynchronous teams* (A-Teams). An A-Team is a problem solving architecture consisting of a collection of agents and memories connected into a strongly cyclic directed network. Each of these agents is an optimization algorithm and can work on the target problem, on a relaxation—i.e., a superclass—of it, or on a subclass of the problem. The basic idea of A-Teams is having these agents work asynchronously and autonomously on a set of shared memories. These shared memories consist of trial solutions for some problem (the target problem, a superclass, or a subclass as mentioned before), and the action of an agent consists of modifying the memory by adding a solution, deleting a solution, or altering a solution. A-Teams have been successfully utilized in a variety of combinatorial optimization problems, see e.g. [5, 39].

Denzinger and Offerman [12] present a similar multi-agent based approach for achieving cooperation between search-systems with different search paradigms. The TECHS (TEams for Cooperative Heterogenous Search) approach consists of teams of one or more agents using the same search paradigm. The communication between the agents is controlled by so-called send- and receive-referees, in order to filter the exchanged data. Each agent is in a cycle between searching and processing received information. In order to demonstrate the usefulness of TECHS, a GA and a B&B based system for job-shop scheduling is described. The GA and B&B agents exchange only positive information (solutions), whereas the B&B agents can also exchange negative information (closed subtrees). Computational experiments show that the cooperation results in finding better solutions given a fixed time-limit and in finding solutions comparable to the ones of the best individual system alone in less time.

3 Integrative Combinations

In this section we discuss approaches of combining exact algorithms and metaheuristics in an integrative way such that one technique is a subordinate embedded component of another technique.

3.1 Incorporating Exact Algorithms in Metaheuristics

We start by considering techniques where exact algorithms are incorporated into metaheuristics.

Exactly Solving Relaxed Problems The usefulness of solutions to relaxations of an original problem has already been mentioned in Section 2.1. Besides exploiting them to derive promising initial solutions for a subsequent algorithm, they can be of great benefit for heuristically guiding neighborhood search, recombination, mutation, repair and/or local improvement. Examples where the solution of the LP-relaxation and its dual were exploited in such ways are the hybrid genetic algorithms for the multiconstrained knapsack problem from Chu and Beasley [6] and Raidl [35].

Exactly Searching Large Neighborhoods A common approach is to search neighborhoods in local search based metaheuristics by means of exact algorithms. If the neighborhoods are chosen appropriately, they can be relatively large and nevertheless an efficient search for the best neighbor is still reasonable. Such techniques are known as Very Large-Scale Neighborhood (VLSN) search [1].

Burke et al. [4] present an effective local and variable neighborhood search heuristic for the asymmetric traveling salesman problem in which they have embedded an exact algorithm in the local search part, called HyperOpt, in order to exhaustively search relatively large promising regions of the solution space. Moreover, they propose a hybrid of HyperOpt and 3-opt which allows to benefit from the advantages of both approaches and gain better tours overall. Using this hybrid within the variable neighborhood search metaheuristic framework also allows to overcome local optima and to create tours of high quality.

Dynasearch [8] is another example where exponentially large neighborhoods are explored. The neighborhood where the search is performed consists of all possible combinations of mutually independent simple search steps and one Dynasearch move consists of a set of independent moves that are executed in parallel in a single local search iteration. Independence in the context of Dynasearch means that the individual moves do not interfere with each other; in this case, dynamic programming can be used to find the best combination of independent moves. Dynasearch is restricted to problems where the single search steps are independent, and it has so far only been applied to problems, where solutions are represented as permutations.

For the class of partitioning problems, Thompson et al. [41, 42] defined the concept of a cyclic exchange neighborhood, which is the transfer of single elements between several subsets in a cyclic manner; for example, a 2–exchange move can be seen as a cyclic exchange of length two. Thompson et al. showed that for any current solution to a partitioning problem a new, edge-weighted graph can be constructed, where the set of nodes is split into subsets according to a partition induced by the current solution of the partitioning problem. A cyclic exchange for the original problem corresponds to a cycle in this new graph that uses at most one node of each subset. Exact and heuristic methods that solve the problem of finding the most negative-cost subset-disjoint cycle (which corresponds to the best improving neighbor of the current solution) have been developed.

Puchinger et al. [34] describe a combined GA/B&B approach for solving a real-world glass cutting problem. The GA uses an order-based representation, which is decoded using a greedy heuristic. The B&B algorithm is applied with a certain probability enhancing the decoding phase by generating locally optimal subpatterns. Reported results indicate that the approach of occasionally solving subpatterns to optimality may increase the overall solution quality.

The work of Klau et al. [22] has already been mentioned in Section 2.1 in the context of collaborative sequential combinations. When looking at the memetic algorithm we encounter another kind of exact/heuristic algorithm combination. An exact subroutine for the price-collecting Steiner tree problem on trees is used to locally improve candidate solutions.

Merging Solutions Subspaces defined by the merged attributes of two or more solutions can, like the neighborhoods of single solutions, also be searched by exact techniques. The algorithms by Clements et al. [7], Applegate et al. [2], and Klau et al. [22], which were already discussed in Section 2.1, also follow this idea, but are of sequential collaborative nature. Here, we consider approaches where merging is iteratively applied within a metaheuristic.

Cotta and Troya [9] present a framework for hybridizing B&B with evolutionary algorithms. B&B is used as an operator embedded in the evolutionary algorithm. The authors recall the necessary theoretical concepts on forma analysis (formae are generalized schemata), such as the dynastic potential of two chromosomes x and y, which is the set of individuals that only carry information contained in x and y. Based on these concepts the idea of dynastically optimal recombination is developed. This results in an operator exploring the potential of the recombined solutions using B&B, providing the best possible combination of the ancestors' features that can be attained without introducing implicit mutation. Extensive computational experiments on different benchmark sets comparing different crossover operators with the new hybrid one show the usefulness of the presented approach.

Marino et al. [27] present an approach where a GA is combined with an exact method for the Linear Assignment Problem (LAP) to solve the graph coloring problem. The LAP algorithm is incorporated into the crossover operator and generates the optimal permutation of colors within a cluster of nodes, hereby preventing the offspring to be less fit than its parents. The algorithm does not outperform other approaches, but provides comparable results. The main conclusion is that solving the LAP in the crossover operator strongly improves the performance of the GA compared to the GA using crossover without LAP.

Exact Algorithms as Decoders In evolutionary algorithms, candidate solutions are sometimes only incompletely represented in the chromosome, and an exact algorithm is used as decoder for determining the missing parts in an optimal way.

Staggemeier et al. [37], for example, present a hybrid genetic algorithm to solve a lot-sizing and scheduling problem minimizing inventory and backlog costs of multiple products on parallel machines. Solutions are represented as product subsets for each machine at each period. Corresponding optimal lot sizes are determined when the solution is decoded by solving a linear program. The approach outperforms a MIP formulation of the problem solved using CPLEX.

3.2 Incorporating Metaheuristics in Exact Algorithms

We now turn to techniques where metaheuristics are embedded within exact algorithms.

Metaheuristics for Obtaining Incumbent Solutions and Bounds In general, heuristics and metaheuristics are often used to determine bounds and incumbent solutions in B&B approaches. For example, Woodruff [44] describes a chunking-based selection strategy to decide at each node of the B&B tree whether or not reactive tabu search is called in order to eventually find a better incumbent solution. The chunking-based strategy measures a distance between the current node and nodes already explored by the metaheuristic in order to bias the selection toward distant points. Reported computational results indicate that adding the metaheuristic improves the B&B performance.

Metaheuristics for Column and Cut Generation In branch-and-cut and branch-and-price algorithms, the dynamic separation of cutting-planes and the pricing of columns, respectively, is sometimes done by means of heuristics including metaheuristics in order to speed up the whole optimization process.

Filho and Lorena [14] apply a heuristic column generation approach to graph coloring. They describe the principles of their constructive genetic algorithm and give a column generation formulation of the problem. The GA is used to generate the initial columns and to solve the slave problem (the weighted maximum independent set problem) at every iteration. Column generation is performed as long as the GA finds columns with negative reduced costs. The master problem is solved using CPLEX. Some encouraging results are presented.

Puchinger and Raidl [32, 33] propose new integer linear programming formulations for the three-stage two-dimensional bin packing problem. Based on these formulations, a branch-and-price algorithm was developed in which fast column generation is performed by applying a hierarchy of four methods: (a) a greedy heuristic, (b) an evolutionary algorithm, (c) solving a restricted form of the pricing problem using CPLEX, and finally (d) solving the complete pricing problem using CPLEX. Computational experiments on standard benchmark instances document the benefits of the new approach. The combination of all four pricing algorithms in the proposed branch-and-price framework yields the best results in terms of the average objective value, the average run-time, and the number of instances solved to proven optimality.

Metaheuristics for Strategic Guidance of Exact Search French et al. [16] present a GA/B&B hybrid to solve feasibility and optimization IP problems. Their hybrid algorithm combines the generic B&B of the MIP-solver XPRESS-MP with a steady-state GA. It starts by traversing the B&B tree. During this phase, information from nodes is collected in order to suggest chromosomes to be added to the originally randomly initialized GA-population. When a certain criterion is fulfilled, the GA is started using the augmented initial population. When the GA terminates, its fittest solution is passed back and grafted onto the B&B tree. Full control is given back to the B&B-engine, after the newly added nodes were examined to a certain degree. Reported results on MAX-SAT instances show that this hybrid approach yields better solutions than B&B or the GA alone.

Kotsikas and Fragakis [23] determine improved node selection strategies within B&B for solving MIPs by using genetic programming (GP). After running B&B for a certain amount of time, information is collected from the B&B tree and used as training set for GP, which is performed to find a node selection strategy more appropriate for the specific problem at hand. The following second B&B phase then uses this new node selection strategy. Reported results show that this approach has potential, but needs to be enhanced in order to be able to compete with today's state-of-the-art node selection strategies.

---

http://www.dashoptimization.com/

Applying the Spirit of Metaheuristics Last but not least, there are a few approaches where it is tried to bring the spirit of local search based techniques into B&B. The main idea is to first search some neighborhood of incumbent solutions more intensively before turning to a classical node selection strategy. However, there is no explicit metaheuristic, but B&B itself is used for doing the local search. The metaheuristic may also be seen to be executed in a "virtual" way.

Fischetti and Lodi [15] introduced local branching, an exact approach combining the spirit of local search metaheuristics with a generic MIP-solver (CPLEX). They consider general MIPs with 0-1 variables. The idea is to iteratively solve a local subproblem corresponding to a classical k-OPT neighborhood using the MIP-solver. This is achieved by introducing a local branching constraint based on an incumbent solution $\overline{x}$, which partitions the search space into the k-OPT neighborhood and the rest: $\Delta(x, \overline{x}) \leq k$ and $\Delta(x, \overline{x}) \geq k + 1$, respectively, with Δ being the Hamming distance of the 0-1 variables. The first subproblem is solved, and if an improved solution could be found, a new subproblem is devised and solved; this is repeated as long as an improved solution is found. If the process stops, the rest of the problem is solved in a standard way. This basic mechanism is extended by introducing time limits, automatically modifying the neighborhood size k and adding diversification strategies in order to improve the performance. Reported results are promising.

Danna et al. [10] present an approach called Relaxation Induced Neighborhood Search (RINS) in order to explore the neighborhoods of promising MIP solutions more intensively. The main idea is to occasionally devise a sub-MIP at a node of the B&B tree that corresponds to a certain neighborhood of an incumbent solution: First, variables having the same values in the incumbent and in the current solution of the LP-relaxation are fixed. Second, an objective cutoff based on the objective value of the incumbent is set. Third, a sub-MIP is solved on the remaining variables. The time for solving this sub-MIP is limited. If a better incumbent could be found during this process, it is passed to the global MIP-search which is resumed after the sub-MIP termination. CPLEX is used as MIP-solver. The authors experimentally compare RINS to standard CPLEX, local branching, combinations of RINS and local branching, and guided dives. Results indicate that RINS often performs best.

4 Conclusions

We gave a survey on very different, existing approaches for combining exact algorithms and metaheuristics. The two main categories in which we divided these techniques are collaborative and integrative combinations. Some of the combinations are dedicated to very specific combinatorial optimization problems, whereas others are designed to be more generally useful. Altogether, the existing work documents that both, exact optimization techniques and metaheuristics, have specific advantages which complement each other. Suitable combinations of exact algorithms and metaheuristics can benefit much from synergy and often exhibit significantly higher performance with respect to solution quality and time. Some of the presented techniques are mature, whereas others are still in their infancy and need substantial further research in order to make them fully developed. Future work on such hybrid systems is highly promising.

References

1. R. K. Ahuja, Ö. Ergun, J. B. Orlin, and A. P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1-3):75–102, 2002.
2. D. Applegate, R. Bixby, V. Chvátal, and W. Cook. On the solution of the traveling salesman problem. *Documenta Mathematica*, Extra Volume ICM III:645–656, 1998.

3. T. Bäck, D. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation*. Oxford University Press, New York NY, 1997.
4. E. K. Burke, P. I. Cowling, and R. Keuthen. Effective local and guided variable neighborhood search methods for the asymmetric travelling salesman problem. In E. Boers et al., editors, *Applications of Evolutionary Computing: EvoWorkshops 2001*, volume 2037 of *LNCS*, pages 203–212. Springer, 2001.
5. S. Chen, S. Talukdar, and N. Sadeh. Job-shop-scheduling by a team of asynchronous agents. In *IJCAI-93 Workshop on Knowledge-Based Production, Scheduling and Control*, Chambery, France, 1993.
6. P. C. Chu and J. E. Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4:63–86, 1998.
7. D. Clements, J. Crawford, D. Joslin, G. Nemhauser, M. Puttlitz, and M. Savelsbergh. Heuristic optimization: A hybrid AI/OR approach. In *Proceedings of the Workshop on Industrial Constraint-Directed Scheduling*, 1997. In conjunction with the Third International Conference on Principles and Practice of Constraint Programming (CP97).
8. R. K. Congram. *Polynomially Searchable Exponential Neighbourhoods for Sequencing Problems in Combinatorial Optimisation*. PhD thesis, University of Southampton, Faculty of Mathematical Studies, UK, 2000.
9. C. Cotta and J. M. Troya. Embedding branch and bound within evolutionary algorithms. *Applied Intelligence*, 18:137–153, 2003.
10. E. Danna, E. Rothberg, and C. Le Pape. Exploring relaxation induced neighbourhoods to improve MIP solutions. Technical report, ILOG, 2003.
11. G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.
12. J. Denzinger and T. Offermann. On cooperation between evolutionary algorithms and other search paradigms. In *Proceedings of the 1999 Congress on Evolutionary Computation (CEC)*. IEEE Press, 1999.
13. I. Dumitrescu and T. Stuetzle. Combinations of local search and exact algorithms. In G. R. Raidl et al., editors, *Applications of Evolutionary Computation*, volume 2611 of *LNCS*, pages 211–223. Springer, 2003.
14. G. R. Filho and L. A. N. Lorena. Constructive genetic algorithm and column generation: an application to graph coloring. In *Proceedings of APORS 2000 - The Fifth Conference of the Association of Asian-Pacific Operations Research Societies within IFORS*, 2000.
15. M. Fischetti and A. Lodi. Local Branching. *Mathematical Programming Series B*, 98:23–47, 2003.
16. A. P. French, A. C. Robinson, and J. M.Wilson. Using a hybrid genetic-algorithm/branch and bound approach to solve feasibility and optimization integer programming problems. *Journal of Heuristics*, 7:551–564, 2001.
17. F. Glover and G. Kochenberger, editors. *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Management Science*. Kluwer Academic Publishers, 2003.
18. F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 1997.
19. F. Glover, M. Laguna, and R. Martí. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 39(3):653–684, 2000.
20. P. Hansen and N. Mladenović. An introduction to variable neighborhood search. In S. Voß, S. Martello, I. Osman, and C. Roucairol, editors, *Meta-heuristics: advances and trends in local search paradigms for optimization*, pages 433–438. Kluwer Academic Publishers, 1999.
21. S. Kirkpatrick, C. Gellat, and M. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.
22. G. Klau, I. Ljubić, A. Moser, P. Mutzel, P. Neuner, U. Pferschy, G. Raidl, and R. Weiskircher. Combining a memetic algorithm with integer programming to solve the prize-collecting Steiner tree problem. In K. Deb et al., editors, *Genetic and Evolutionary Computation – GECCO 2004*, volume 3102 of *LNCS*, pages 1304–1315. Springer, 2004.
23. K. Kostikas and C. Fragakis. Genetic programming applied to mixed integer programming. In M. Keijzer et al., editors, *Genetic Programming - EuroGP 2004*, volume 3003 of *LNCS*, pages 113–124. Springer, 2004.

24. P. Larrañaga and J. Lozano. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publishers, 2001.
25. A. Z.-Z. Lin, J. Bean, and I. C. C. White. A hybrid genetic/optimization algorithm for finite horizon partially observed markov decision processes. *Journal on Computing*, 16(1):27–38, 2004.
26. H. R. Lourenço, O. Martin, and T. Stützle. Iterated local search. In Glover and Kochenberger [17], pages 321–353.
27. A. Marino, A. Prügel-Bennett, and C. A. Glass. Improving graph colouring with linear programming and genetic algorithms. In *Proceedings of EUROGEN 99*, pages 113–118, Jyväskyiä, Finland, 1999.
28. P. Moscato and C. Cotta. A gentle introduction to memetic algorithms. In Glover and Kochenberger [17], pages 105–144.
29. A. Nagar, S. S. Heragu, and J. Haddock. A meta-heuristic algorithm for a bi-criteria scheduling problem. *Annals of Operations Research*, 63:397–414, 1995.
30. G. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization.* John Wiley & Sons, 1988.
31. A. Plateau, D. Tachat, and P. Tolla. A hybrid search combining interior point methods and metaheuristics for 0-1 programming. *International Transactions in Operational Research*, 9:731–746, 2002.
32. J. Puchinger and G. R. Raidl. An evolutionary algorithm for column generation in integer programming: an effective approach for 2D bin packing. In X. Yao et al., editors, *Parallel Problem Solving from Nature – PPSN VIII*, volume 3242 of *LNCS*, pages 642–651. Springer, 2004.
33. J. Puchinger and G. R. Raidl. Models and algorithms for three-stage two-dimensional bin packing. Technical Report TR 186–1–04–04, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 2004. submitted to the European Journal of Operations Research.
34. J. Puchinger, G. R. Raidl, and G. Koller. Solving a real-world glass cutting problem. In J. Gottlieb and G. R. Raidl, editors, *Evolutionary Computation in Combinatorial Optimization – EvoCOP 2004*, volume 3004 of *LNCS*, pages 162–173. Springer, 2004.
35. G. R. Raidl. An improved genetic algorithm for the multiconstrained 0–1 knapsack problem. In D. B. Fogel, editor, *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, pages 207–211. IEEE Press, 1998.
36. G. R. Raidl and H. Feltl. An improved hybrid genetic algorithm for the generalized assignment problem. In H. M. Haddadd et al., editors, *Proceedings of the 2003 ACM Symposium on Applied Computing*, pages 990–995. ACM Press, 2004.
37. A. T. Staggemeier, A. R. Clark, U. Aickelin, and J. Smith. A hybrid genetic algorithm to solve a lot-sizing and scheduling problem. In *Proceedings of the 16th triannual Conference of the International Federation of Operational Research Societies*, Edinburgh, U.K., 2002.
38. S. Talukdar, L. Baeretzen, A. Gove, and P. de Souza. Asynchronous teams: Cooperation schemes for autonomous agents. *Journal of Heuristics*, 4:295–321, 1998.
39. S. Talukdar, S. Murty, and R. Akkiraju. Asynchronous teams. In Glover and Kochenberger [17], pages 537–556.
40. H. Tamura, A. Hirahara, I. Hatono, and M. Umano. An approximate solution method for combinatorial optimisation. *Transactions of the Society of Instrument and Control Engineers*, 130:329–336, 1994.
41. P. Thompson and J. Orlin. The theory of cycle transfers. Technical Report OR-200-89, MIT Operations Research Center, Boston, MA, 1989.
42. P. Thompson and H. Psaraftis. Cycle transfer algorithm for multivehicle routing and scheduling problems. *Operations Research*, 41:935–946, 1993.
43. M. Vasquez and J.-K. Hao. A hybrid approach for the 0–1 multidimensional knapsack problem. In *Proceedings of the International Joint Conference on Artificial Intelligence 2001*, pages 328–333, 2001.
44. D. L. Woodruff. A chunking based selection strategy for integrating meta-heuristics with branch and bound. In S. Voss et al., editors, *Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization*, pages 499–511. Kluwer Academic Publishers, 1999.