Chapter 5
Decomposition Techniques as Metaheuristic Frameworks

Marco Boschetti, Vittorio Maniezzo, and Matteo Roffilli

Abstract Decomposition techniques are well-known as a means for obtaining tight lower bounds for combinatorial optimization problems, and thus as a component for solution methods. Moreover a long-established research literature uses them for defining problem-specific heuristics. More recently it has been observed that they can be the basis also for designing metaheuristics. This tutorial elaborates this last point, showing how the three main decomposition techniques, namely Dantzig-Wolfe, Lagrangean and Benders decompositions, can be turned into model-based, dual-aware metaheuristics. A well known combinatorial optimization problem, the Single Source Capacitated Facility Location Problem, is then chosen for validation, and the implemented codes of the proposed algorithms are benchmarked on standard instances from literature.

5.1 Introduction

Traditionally, heuristic methods, and metaheuristics in particular, have been primal-only methods. They are usually quite effective in solving the given problem instances, and they terminate providing the best feasible solution found during the allotted computation time. However, disregarding dual information implies some obvious drawbacks, first of all not knowing the quality of the proposed solution, but also having possibly found an optimal solution at the beginning of the search and having wasted CPU time ever since, having

Marco Boschetti
Department of Mathematics, University of Bologna, Bologna, Italy
e-mail: `marco.boschetti@unibo.it`

Vittorio Maniezzo · Matteo Roffilli
Department of Computer Science, University of Bologna, Bologna, Italy
e-mail: `vittorio.maniezzo@unibo.it,roffilli@csr.unibo.it`

V. Maniezzo et al., (eds.), *Matheuristics*, Annals of Information Systems 10, DOI 10.1007/978-1-4419-1306-7_5,

searched a big search space that could have been much reduced, or having disregarded important information that could have been very effective for constructing good solutions.

Dual information is also tightly connected with the possibility of obtaining good lower bounds (making reference, here and forward, to minimization problems), another element which is not a structural part of current metaheuristics. On the contrary, most mathematical programming literature dedicated to exact methods is strongly based on these elements for achieving the obtained results. There is nothing, though, that limits the effectiveness of dual/bounding procedures to exact methods. There are in fact wide research possibilities both in determining how to convert originally exact methods into efficient heuristics and in designing new, intrinsically heuristic techniques, which include dual information.

In this tutorial we examine a possibility from the second alternative. There are many ways in which bounds can be derived, one of the most effective of these is the use of decomposition techniques [6]. These are techniques primarily meant to exploit the possibility of identifying a subproblem in the problem to solve and to decompose the whole problem in a *master problem* and a *subproblem*, which communicate via dual or dual-related information. The popularity of these techniques derives both from their effectiveness in providing efficient bounds and from the observation that many real-world problems lead themselves to a decomposition.

Unfortunately, despite their prolonged presence in the optimization literature, there is as yet no clear-cut recipe for determining which problems should be solved with decompositions and which are better solved by other means. Clearly, decomposition techniques are foremost candidates for problems which are inherently structured as a master and different subproblems, but it is at times possible to effectively decompose the formulation of a problem which does not show such structure and enjoy advantages. Examples from the literature of effective usage of decomposition techniques (mainly Lagrangean) on single-structure problems include, e.g., set covering [13, 14], set partitioning [3, 32, 12] and crew scheduling [11, 18, 19, 24].

In a previous paper [9] we observed that the general structure of decomposition techniques can be extended from bound computation to include feasible solution construction. According to this, decompositions such as Dantzig-Wolfe, Benders or Lagrangean provide a rich framework for designing metaheuristics. In this work we elaborate this point, showing how the three mentioned approaches can be practically applied to a well-known combinatorial optimization problem, namely the Single Source Capacitated Facility Location Problem.

The structure of the chapter is as follows. In Section 5.2 we introduce the three basic decomposition techniques: Lagrangean relaxation, Dantzig-Wolfe decomposition, and Benders decomposition. Section 5.3 shows, for each of the three methods, how to derive a possible metaheuristic. Section 5.4 introduces the Single Source Capacitated Facility Location Problem, which will be used for benchmarking the algorithms. Finally, Section 5.5 shows the computational results obtained with our implementation of the proposed metaheuristics.

5.2 Decomposition Methods

This section briefly overviews the three decomposition techniques we will use as a basis for metaheuristics design. These decompositions can be applied to continuous, mixed-integer and pure integer linear programming problems. Since decomposition is a basic operations research topic, which can be found in any mathematical programming textbook, we only present here the basic formulae in the general case of a mixed integer problem. The discussion is for a minimization problem, being trivial to apply it to a maximization one.

The problem to solve, called P, has the following structure:

$$z_P = \min\ \mathbf{c}_1\mathbf{x} + \mathbf{c}_2\mathbf{y} \tag{5.1}$$
$$s.t.\ \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \geq \mathbf{b} \tag{5.2}$$
$$\mathbf{D}\mathbf{y} \geq \mathbf{d} \tag{5.3}$$
$$\mathbf{x} \geq \mathbf{0} \tag{5.4}$$
$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{5.5}$$

We assume, for ease of presentation, that the feasibility region is non-null and bounded.

5.2.1 Lagrangean Relaxation

Lagrangean relaxation permits to obtain a lower bound to problem P by removing some difficult constraints and by dualizing them into the objective function by means of *Lagrangean penalties*. For example, if in problem P we relax constraints (5.2) using the non-negative Lagrangean penalty vector $\boldsymbol{\lambda}$, we obtain the following formulation LR:

$$z_{LR}(\boldsymbol{\lambda}) = \min\ \mathbf{c}_1\mathbf{x} + \mathbf{c}_2\mathbf{y} + \boldsymbol{\lambda}(\mathbf{b} - \mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{y}) \tag{5.6}$$
$$s.t.\ \mathbf{D}\mathbf{y} \geq \mathbf{d} \tag{5.7}$$
$$\mathbf{x} \geq \mathbf{0} \tag{5.8}$$
$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{5.9}$$

$z_{LR}(\boldsymbol{\lambda})$ is a valid lower bound to the optimal value of P, i.e., $z_{LR}(\boldsymbol{\lambda}) \leq z_P$, for every $\boldsymbol{\lambda} \geq \mathbf{0}$. To identify the penalty vector $\boldsymbol{\lambda}$ that maximizes the lower bound $z_{LR}(\boldsymbol{\lambda})$, we solve the so-called *Lagrangean dual*, which can be formulated as follows:

$$z_{LR} = \max \{z_{LR}(\boldsymbol{\lambda}) : \boldsymbol{\lambda} \geq \mathbf{0}\} \tag{5.10}$$

For solving the Lagrangean dual, an internal *subproblem* LR must be solved for each penalty vector λ. LR is as follows:

$$z_{LR}(\boldsymbol{\lambda}) = \min\ (\mathbf{c}_1 - \boldsymbol{\lambda}\mathbf{A})\mathbf{x} + (\mathbf{c}_2 - \boldsymbol{\lambda}\mathbf{B})\mathbf{y} + \boldsymbol{\lambda}\mathbf{b} \tag{5.11}$$

$$s.t.\ \mathbf{D}\mathbf{y} \geq \mathbf{d} \tag{5.12}$$

$$\mathbf{x} \geq \mathbf{0} \tag{5.13}$$

$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{5.14}$$

If the subproblem is solved to integrality, it is possible that the lower bound provided by z_{LR} is tighter than the linear relaxation of problem P.

Notice that it is possible to add to the LR formulation constraints that are redundant in the original formulation, but that can help the convergence. Moreover, it is sometimes possible to obtain feasible dual solutions directly from the Lagrangean penalties. Approaches based on this property have been used, e.g., to generate reduced problems which consider only the variables of k-least reduced costs (e.g., [11, 12, 24]).

5.2.2 Dantzig-Wolfe Decomposition

Dantzig-Wolfe decomposition [16] is an iterative procedure which successively approximates the linear relaxation of problem P by decomposing it into a sequence of smaller and/or easier *subproblems*. The subproblems dynamically generate the columns of a *master problem* corresponding to the LP relaxation of P.

In order to use the same decomposition as in Section 5.2.1, let F be the feasible region induced by constraints (5.3)–(5.5), i.e. $F = \{(\mathbf{x}, \mathbf{y}) : \mathbf{D}\mathbf{y} \geq \mathbf{d}, \mathbf{x} \geq \mathbf{0}, \mathbf{y} \geq \mathbf{0}$ and integer$\}$, which we assume finite and non-null, and let $\{(\mathbf{x}^t, \mathbf{y}^t) : t = 1, \ldots, T\}$ be the set of the extreme points of F. Dantzig-Wolfe proceeds by identifying optimal (with respect to the current cost function) extreme points of F, computed as solutions of a *subproblem*, then passing them to the *master problem* in order to check them against the relaxed constraints, i.e., those not F-defining. The master problem is formulated as a constrained linear combination of the proposed extreme points. After having computed the cost of the best combination of the so far proposed extreme points of F, taking into consideration also the relaxed constraints retained in the master, the subproblem costs are updated, and are computed as reduced costs derived from the dual values of the relaxed constraints. The subproblem is then solved again, to see whether a new, less expensive extreme point can be found.

In the case of problem P, a possible master problem, obtained again by relaxing the "difficult" constraints (5.2) is as follows:

$$z_{MDW} = \min \sum_{t=1}^{T} (\mathbf{c}_1\mathbf{x}^t + \mathbf{c}_2\mathbf{y}^t)\mu_t \tag{5.15}$$

$$s.t. \sum_{t=1}^{T} (\mathbf{A}\mathbf{x}^t + \mathbf{B}\mathbf{y}^t)\mu_t \geq \mathbf{b} \tag{5.16}$$

$$\sum_{t=1}^{T} \mu_t = 1 \tag{5.17}$$

$$\mu_t \geq 0, \qquad t = 1, \ldots, T \tag{5.18}$$

The corresponding subproblem is:

$$z_{SDW}(\mathbf{u}, \alpha) = \min\ (\mathbf{c}_1 - \mathbf{u}\mathbf{A})\mathbf{x} + (\mathbf{c}_2 - \mathbf{u}\mathbf{B})\mathbf{y} - \alpha \tag{5.19}$$

$$s.t.\ \mathbf{D}\mathbf{y} \geq \mathbf{d} \tag{5.20}$$

$$\mathbf{x} \geq \mathbf{0} \tag{5.21}$$

$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{5.22}$$

where $\mathbf{u}$ and α are the dual variables corresponding to constraints (5.16) and (5.17) of the master problem, respectively.

If the subproblem optimal solution $(\mathbf{x}^*, \mathbf{y}^*)$ has a value $z_{SDW}(\mathbf{u}, \alpha) < 0$, we can add the corresponding column $(\mathbf{A}\mathbf{x}^* + \mathbf{B}\mathbf{y}^*)$ of cost $(\mathbf{c}_1\mathbf{x}^* + \mathbf{c}_2\mathbf{y}^*)$ into the master problem, otherwise we have reached the optimal solution of MDW. Notice that subproblem SDW is identical to LR if we replace $\mathbf{u}$ and α with $\boldsymbol{\lambda}$ and $-\boldsymbol{\lambda}\mathbf{b}$, respectively.

At each iteration of the procedure, a valid lower bound to the optimal solution value of the original problem is given by $z_{MDW} + z_{SDW}$ (see [6] for further details). This lower bound is not monotonically nondecreasing. Therefore, we need to maintain the best value obtained through the iterations.

5.2.3 Benders Decomposition

Benders decomposition [8] computes a lower bound to the optimal cost of the original problem by solving a *master problem* which fixes some of its variables. Then, to improve the lower bound, it solves a *subproblem* which adds new constraints to the master.

Let $\mathbf{w}$ and $\mathbf{v}$ be the dual variables of problem P associated to constraints (5.2) and (5.3), respectively. The dual of P is as follows:

$$z_D = \max \ \mathbf{wb} + \mathbf{vd} \tag{5.23}$$
$$s.t. \ \mathbf{wA} \leq \mathbf{c_1} \tag{5.24}$$
$$\mathbf{wB} + \mathbf{vD} \leq \mathbf{c_2} \tag{5.25}$$
$$\mathbf{w} \geq \mathbf{0} \tag{5.26}$$
$$\mathbf{v} \geq \mathbf{0} \tag{5.27}$$

The dual D can be also rewritten as $z_D = \max\{z_{SD}(\mathbf{w}) : \mathbf{wA} \leq \mathbf{c_1}, \mathbf{w} \geq \mathbf{0}\}$, where

$$z_{SD}(\mathbf{w}) = \max \ \mathbf{wb} + \mathbf{vd} \tag{5.28}$$
$$s.t. \ \mathbf{vD} \leq \mathbf{c_2} - \mathbf{wB} \tag{5.29}$$
$$\mathbf{v} \geq \mathbf{0} \tag{5.30}$$

Let $\mathbf{y}$ be the dual variables associated to constraints (5.29). The dual of SD becomes:

$$z_{SP}(\mathbf{w}) = \min \ (\mathbf{c}_2 - \mathbf{wB})\mathbf{y} + \mathbf{wb} \tag{5.31}$$
$$s.t. \ \mathbf{Dy} \geq \mathbf{d} \tag{5.32}$$
$$\mathbf{y} \geq \mathbf{0} \text{ and integer} \tag{5.33}$$

Upon denoting $W = \{\mathbf{w} : \mathbf{wA} \leq \mathbf{c_1}, \mathbf{w} \geq \mathbf{0}\}$ and $Y = \{\mathbf{y} : \mathbf{Dy} \geq \mathbf{d}, \mathbf{y} \geq \mathbf{0}$ and integer$\}$, we can rewrite problem D as:

$$z_D = \max_{w \in W} \min_{\mathbf{y} \in Y} (\mathbf{c}_2 - \mathbf{wB})\mathbf{y} + \mathbf{wb}. \tag{5.34}$$

Let $\{\mathbf{w}^t, t = 1, \ldots, T\}$ be the set of the extreme points of W. Since we have assumed the feasible region to be finite and non-null, we have that $z_D = \min_{\mathbf{y} \in \mathbf{Y}} \max_{t=1,\ldots,T} (\mathbf{c}_2 - \mathbf{w}^t\mathbf{B})\mathbf{y} + \mathbf{w}^t\mathbf{b}$, which is equivalent to the following formulation MB:

$$z_{MB} = \min \ z \tag{5.35}$$
$$s.t. \ z \geq (\mathbf{c}_2 - \mathbf{w}^t\mathbf{B})\mathbf{y} + \mathbf{w}^t\mathbf{b}, \quad t = 1, \ldots, T \tag{5.36}$$
$$\mathbf{y} \in \mathbf{Y} \tag{5.37}$$

Problem MB is the *Benders master problem* and constraints (5.36) are the so-called *Benders cuts*. The number of Benders cuts T is usually huge; therefore, the master problem is initially solved considering only a small number T' of Benders cuts, i.e., $T' \ll T$. In order to ascertain whether the solution is already optimal or an additional cut should be added to the master, we need to solve a *subproblem* SB. Since problem D defined in (5.34) is equivalent to:

$$z_D = \min_{\mathbf{y} \in Y} \left(\mathbf{c}_2\mathbf{y} + \max_{w \in W} \mathbf{w}(\mathbf{b} - \mathbf{By}) \right) \tag{5.38}$$

the subproblem SB is:

$$z_{SB}(\mathbf{y}) = \max\ \mathbf{w}(\mathbf{b} - \mathbf{By}) \tag{5.39}$$
$$s.t.\ \mathbf{wA} \leq \mathbf{c_1} \tag{5.40}$$
$$\mathbf{w} \geq \mathbf{0} \tag{5.41}$$

Notice that a primal solution $(\mathbf{x}, \mathbf{y})$ of problem P, useful for the metaheuristics discussed in Section 5.3, can be obtained by the dual of SB defined as:

$$z_{SP}(\mathbf{y}) = \min\ \mathbf{c}_1\mathbf{x} \tag{5.42}$$
$$s.t.\ \mathbf{Ax} \leq \mathbf{b} - \mathbf{By} \tag{5.43}$$
$$\mathbf{x} \geq \mathbf{0} \tag{5.44}$$

Also for problems MB and SB, it is possible to add constraints that are redundant in the original formulation, but can help convergence.

It is interesting to show that also for Benders decomposition we can have a subproblem equivalent to the ones of Lagrangean relaxation and Dantzig-Wolfe decomposition. In fact, if the dual D, (5.23)–(5.27), is rewritten as:

$$z_D = \max \{z_{SD'}(\mathbf{w}) : \mathbf{w} \geq \mathbf{0}\} \tag{5.45}$$

where

$$z_{SD'}(\mathbf{w}) = \max\ \mathbf{wb} + \mathbf{vd} \tag{5.46}$$
$$s.t.\ \mathbf{0} \leq \mathbf{c_1} - \mathbf{wA} \tag{5.47}$$
$$\mathbf{vD} \leq \mathbf{c_2} - \mathbf{wB} \tag{5.48}$$
$$\mathbf{v} \geq \mathbf{0} \tag{5.49}$$

the dual of SD′ is identical to subproblem LR, defined by (5.11)–(5.14), after replacing the penalty vector $\boldsymbol{\lambda}$ with $\mathbf{w}$.

5.3 Metaheuristics Derived from Decompositions

In this section we show how metaheuristic frameworks can be directly derived from the three decomposition methods previously described. Notice that the proposed algorithms are not the only ones that could be derived from the used decompositions, but they represent reasonable frameworks, which we have already used with success on different problems. We hope that this chapter may serve as a means to foster research on different or more general metaheuristic frameworks, including other approaches deriving from decomposition techniques.

5.3.1 A Lagrangean Metaheuristic

The literature is rich with heuristics based on the Lagrangean decomposition structure outlined above. An excellent introduction to the whole topic of Lagrangean relaxation, and of related heuristics, can be found in [7]. A general structure of a Lagrangean heuristic, common to most applications, is given in Algorithm 1.

Algorithm 1: LagrHeuristic

1 identify an "easy" subproblem LR($\boldsymbol{\lambda}$)
2 **repeat**
3   solve subproblem LR($\boldsymbol{\lambda}$) obtaining solution $\mathbf{x}$
4   check for unsatisfied constraints
5   update penalties $\boldsymbol{\lambda}$
6   construct problem solution using $\mathbf{x}$ and $\boldsymbol{\lambda}$
7 **until** *(end_condition)* ;

This pseudocode is obviously underspecified for a direct application, being at an abstraction level where metaheuristics are usually presented. However, notice that this structure already shows the essential ingredients of a metaheuristic, i.e., it is "an iterative master process that guides and modifies the operations of a subordinate heuristic" at Step 6.

Steps 1 and 3 are problem-dependent, such as neighborhood definition or crossover implementation in other contexts. Step 4 is trivial, while Step 5 can be implemented by means of any state-of-the-art technique, usually subgradient optimization or bundle methods. Moreover, some of these techniques have been proved to converge not only to the optimal $\boldsymbol{\lambda}$, but also to the optimal $\mathbf{x}$ of the linear relaxation (see Sherali and Choi [29] and Barahona and Anbil [4]), thereby possibly providing a particularly "intelligent" starting point for Step 6.

5.3.2 A Dantzig-Wolfe Metaheuristic

As for any metaheuristic, also for Dantzig-Wolfe we can propose a general structure that will have to be detailed in some of its steps in order to apply it to specific problems. Here, we propose one possible effective structure, but again, alternative ones are possible.

The master problem MDW should be defined to be easy to solve to optimality, while the subproblem SDW can be difficult and it could be needed to solve it heuristically. The proposed pseudocode for algorithm DWHeuristic tries to generate feasible solutions making use of the dual solutions $(\mathbf{u}, \alpha)$ provided by MDW and of the primal solution $(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{T} (\mathbf{x}^t, \mathbf{y}^t)\mu_t$

Algorithm 2: DWHEURISTIC

1 identify a master MDW and an "easy" subproblem SDW($\mathbf{u}$,α), set T=0
2 **repeat**
3   solve master problem MDW
4   given the solution $\boldsymbol{\mu}$ of MDW define $(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{T} (\mathbf{x}^t, \mathbf{y}^t)\mu_t$
5   solve problem SDW($\mathbf{u}$,α), where ($\mathbf{u}$,α) is the dual solution of MDW
6   construct feasible solutions using $(\mathbf{x}, \mathbf{y})$ and/or ($\mathbf{u}$,α), generated by MDW,
7   and/or $(\mathbf{x}', \mathbf{y}')$, generated by SDW($\mathbf{u}$,$\alpha$)
8   **if** *(no more columns can be added)* **then**
9   STOP
10   **else**
11   set $T = T + 1$
12   add the column $(\mathbf{x}', \mathbf{y}')$ generated at step 5
13 **until** *(end_condition)* ;

and $(\mathbf{x}', \mathbf{y}')$ generated by solving MDW and SDW($\mathbf{u}$,α), respectively. However, it is possible to include other local search algorithms, based on different neighborhoods. For example, we can generate a feasible solution using the solutions $(\mathbf{x}^t, \mathbf{y}^t)$ associated to the columns of MDW with $\mu_t > 0$ in its current solution.

5.3.3 A Benders Metaheuristic

The identification of a common structure for Benders based heuristics is more difficult than for Lagrangean or Dantzig-Wolfe ones, since the proposals in the literature vary much, and usually Benders decomposition is used in a very problem-dependent fashion. We propose here one possible structure, which already proved effective, but again, alternative ones are possible.

The structure can be applied both to MIP problems, as sketched in Section 5.2, and to pure IP problems. The subproblem SP (see Equation (5.42)) could be defined over integer or binary variables, in both cases it is necessary to use its linear relaxation in order to obtain its dual SB (Equation (5.39)).

Taking into account the intrinsic difficulty of both MB and SB, we propose to consider solving them both heuristically. The effect of solving heuristically MB at step 3 is that it is not guaranteed to produce a lower bound to problem P. When a lower bound is needed, MB must be solved to optimality, or approximated from below. Notice, however, that the main purpose of MB is to produce alternative $\mathbf{y}$ sets, of possibly increasing qualities, and this can be effectively accomplished by heuristic solutions. Step 5 provides an upper bound, i.e., a feasible solution, to the whole problem. Step 6 finds a lower bound to the problem obtained by fixing the $\mathbf{y}$ variables.

Algorithm 3: BendHeuristic

```
1  identify a master MB and an "easy" subproblem SB(y), set T = 0
2  repeat
3      solve (heuristically) master problem MB obtaining the solution (z, y)
4      if (x are requested to be integer) then
5          solve (heuristically) master problem MB obtaining the solution (z, y)
6      solve problem SB(y) obtaining the dual solution w
7      if (no more columns can be added) then
8          STOP
9      else
10         set T = T + 1
11         add to MB the Benders cut generated by problem SB(y)
12 until (end_condition) ;
```

The terminating condition at Step 7 depends on whether the master is solved heuristically or to optimality. In this last case, the condition would be "**if** $z^t \geq z_d$", which in fact implies the impossibility of generating new cuts. However, in a heuristic context such as admitted by Steps 3 and 5, new cuts could be further generated, which could prove useful for continuing search.

5.4 Single Source Capacitated Facility Location

The algorithms presented in Section 5.3 are meant as metaheuristics. They are relatively simple, yet effective and robust approaches. To get state-of-the-art results some sophisticated elements are needed, for these as for any other metaheuristic. However, a straightforward application of these pseudocodes already produces results, which are close to the state-of-the-art. In order to show the robustness and the ease to arrive to fully-defined, problem-specific codes, we report in this section on the application of each proposed approach to the Single Source Capacitated Facility Location Problem (SCFLP).

The SCFLP is a well-known problem that arises in many applications, from clustering problems in data mining to networks design. The problem asks to locate a number of facilities (e.g., plants, warehouses or hubs), that must provide a service to a set of customers, minimizing a global cost. The cost includes fixed charges for opening the facilities and service costs for satisfying customer demands.

Let $J = \{1, \ldots, n\}$ be the index sets of customers and $I = \{1, \ldots, m\}$ the index set of possible facility locations. Each customer j has an associated demand, q_j, that must be served by a single facility; a facility located at site i has an overall capacity of Q_i. The costs are composed of a cost c_{ij} for supplying the demand of a customer j from a facility established at location i and of a fixed cost, f_i, for opening a facility at location i. Let x_{ij}, $i = 1, \ldots, m$,

$j = 1, \ldots, n$, be binary variables such that $x_{ij} = 1$ if customer j is assigned to a facility located at i, 0 otherwise, and let y_i, $i = 1, \ldots, m$, be binary variables such that $y_i = 1$ if a facility is located at site i, 0 otherwise.

A mathematical formulation of the SCFLP is as follows:

$$z_{SCFLP} = \min \sum_{i \in I, j \in J} c_{ij} x_{ij} + \sum_{i \in I} f_i y_i \tag{5.50}$$

$$s.t. \sum_{i \in I} x_{ij} = 1, \qquad j \in J \tag{5.51}$$

$$\sum_{j \in J} q_j x_{ij} \leq Q_i y_i, \qquad i \in I \tag{5.52}$$

$$x_{ij} \in \{0, 1\}, \qquad i \in I, j \in J \tag{5.53}$$

$$y_i \in \{0, 1\}, \qquad i \in I \tag{5.54}$$

The objective function (5.50) asks to minimize the sum of fixed and service costs. Assignment constraints (5.51) ensure that all customers are serviced by exactly one facility; knapsack constraints (5.52) are the facility capacity constraints and, finally, (5.53) and (5.54) are the integrality constraints.

SCFLP is an NP-hard problem and often the optimal value of its LP relaxation, obtained by removing the integrality constraints, is much worse than the optimum integer solution value. In order to improve the optimal value of the LP-relaxation, as suggested in [20], we can add the following additional constraints, redundant in SCFLP:

$$x_{ij} - y_i \leq 0, \qquad \text{for each } i \in I \text{ and } j \in J \tag{5.55}$$

Given its simple structure, the SCFLP has often been used for benchmarking new approaches. Some variants of it exist, the most studied one permits a split assignment of customers to location, thus relaxing constraints (5.53) to $x_{ij} \geq 0, i \in I, j \in J$. Most approximation results, such as Chudak and Shmoys's 3-approximation algorithm [15], refer to this problem version. Closely related problems are also the Capacitated p-median and the Generalized Assignment Problems. Several exact approaches have been proposed for the SCFLP, one of the best known being [25], where a branch and bound scheme based on a partitioning formulation is proposed. However, exact methods do not scale up to large instance sizes.

Large instances have been tackled by means of different kinds of heuristics, from very large scale neighborhood (VLSN) search [2] to reactive GRASP and tabu search [17]. Extensive research has been devoted to Lagrangean heuristics for the SCFLP. Most authors start by relaxing assignment constraints, obtaining a Lagrangean subproblem which separates into n knapsack problems, one for each facility, whose combined solutions provide a lower bound to the problem [5, 26, 31, 20, 28]. However, different relaxations have also been used. Klincewicz and Luss [21] relax the capacity constraints (5.52), thereby obtaining as Lagrangean subproblem an uncapacitated facility location problem, which is solved heuristically. Beasley [7] and Agar and Salhi [1] relax both the assignment and the capacity constraints, and obtain a very robust solution approach, which provides good quality solutions to a number of different location problems, including p-median, uncapacitated, capacitated and single source facility location problems.

Having introduced the basic techniques and the problem they have been applied to, we move on describing how the basic pseudocodes for the Lagrangean, Dantzig-Wolfe and Benders metaheuristics can be specialized for the SCFLP.

5.4.1 Solving the SCFLP with a Lagrangean Metaheuristic

We present here a very straightforward application of LagrHeuristic to the SCFLP. The resulting algorithm is not enough to produce edge-level results, but it shows that already by means of such a simple code it is possible to get quite good performance. The steps of LagrHeuristic for the SCFLP can be specified as follows. (Note that the step numbers refer to the lines in the pseudocode of the metaheuristic.)

Step 1: Identify an "easy" subproblem LR. The relaxation of the assignment constraints (5.51) in problem SCFLP yields the following problem.

$$z_{LR}(\boldsymbol{\lambda}) = \min \sum_{i\in I, j\in J} (c_{ij} - \lambda_j)x_{ij} + \sum_{i\in I} f_i y_i + \sum_{j\in J} \lambda_j \tag{5.56}$$

$$s.t. \sum_{j\in J} q_j x_{ij} \le Q_i y_i, \quad i \in I \tag{5.57}$$

$$x_{ij} \in \{0,1\}, \quad i\in I, j\in J \tag{5.58}$$

$$y_i \in \{0,1\}, \quad i \in I \tag{5.59}$$

where λ_j, $j \in J$, are unrestricted penalties.

Step 3: Solve subproblem LR. Problem LR decomposes naturally into $|I|$ knapsack problems, with objective function $\sum_{i\in I}\left(\sum_{j\in J}(c_{ij}-\lambda_j)x_{ij} + f_i y_i\right)$. Thus, for each $i \in I$ for which $\sum_{j\in J}(c_{ij}-\lambda_j)x_{ij} < -f_i$, the corresponding y_i is set to 1, otherwise to 0.

Step 4: Check for unsatisfied constraints. The solution of LR can have customers assigned to multiple or to no location. This can be determined by direct inspection.

Step 5: Update penalties $\boldsymbol{\lambda}$. We used a standard subgradient algorithm [26] for updating penalties.

Step 6: construct problem solution using $\mathbf{x}$ *and* $\boldsymbol{\lambda}$. Let $\bar{I}$ be the set of locations chosen in the solution obtained at Step 3. The SCFLP becomes a Generalized Assignment Problem (GAP) as follows:

$$z_{GAP} = \min \sum_{i \in \bar{I}, j \in J} c_{ij} x_{ij} \tag{5.60}$$

$$s.t. \sum_{i \in \bar{I}} x_{ij} = 1, \qquad j \in J \tag{5.61}$$

$$\sum_{j \in J} q_j x_{ij} \leq Q_i, \qquad i \in \bar{I} \tag{5.62}$$

$$x_{ij} \in \{0, 1\}, \qquad i \in \bar{I}, j \in J \tag{5.63}$$

This is still an NP-hard problem, but efficient codes exist to solve it, which we did once per Lagrangean iteration (see the subsequent computational results section for further details).

We formulate the GAP using the original costs $\{c_{ij}\}$ instead of the penalized costs $\{c_{ij} - \lambda_j\}$, which could seem to be an obvious bonus granted by using the Lagrangean relaxation in a heuristic context. This is because in this case, having fixed the set of chosen locations $\bar{I}$, solving the GAP to optimality generates the best possible solution. However, in other circumstances, we can take advantage of using penalized (thus dual-related) costs instead of the original ones (e.g., the fully distributed Lagrangean metaheuristic for a P2P Overlay Network Design Problem described in [10, 22]) obtaining a considerable computational advantage.

Notice that for some iterations, Step 3 may provide a set of locations $\bar{I}$ for which the GAP is unfeasible. In this case no feasible SCFLP solution is generated and LagrHeuristic simply goes on.

5.4.2 *Solving the SCFLP with a Dantzig-Wolfe Metaheuristic*

We have a number of possibilities to decompose our model for the SCFLP. Among them we chose to decompose the problem in such a way as to have a subproblem equivalent to LR, defined for the Lagrangean relaxation described in the previous subsection. The specific steps of DWHeuristic for the SCFLP result as follows.

Step 1: Identify a master MDW and an "easy" subproblem SDW. A possible Dantzig-Wolfe decomposition of the SCFLP maintains the assignment constraints in the master problem:

$$z_{MDW} = \min \sum_{k=1}^{t} \left(\sum_{i \in I, j \in J} c_{ij} x_{ij}^k + \sum_{i \in I} f_i y_i^k \right) \lambda_k \tag{5.64}$$

$$s.t. \sum_{k=1}^{t} \left(\sum_{i \in I} x_{ij}^k \right) \lambda_k = 1, \qquad j \in J \tag{5.65}$$

$$\sum_{k=1}^{t} \lambda_k = 1 \tag{5.66}$$

$$\lambda_k \geq 0, \qquad k = 1, \ldots, t \tag{5.67}$$

and the subproblem is:

$$z_{SDW}(\mathbf{u}, \alpha) = \min \sum_{i \in I, j \in J} (c_{ij} - u_j) x_{ij} + \sum_{i \in I} f_i y_i - \alpha \tag{5.68}$$

$$s.t. \sum_{j \in J} q_j x_{ij} \leq Q_i y_i, \qquad i \in I \tag{5.69}$$

$$x_{ij} \in \{0, 1\}, \qquad i \in I, j \in J \tag{5.70}$$

$$y_i \in \{0, 1\}, \qquad i \in I \tag{5.71}$$

where u_j and α are the dual variables of the master problem corresponding to constraints (5.65) and (5.66), respectively.

Step 3: Solve master problem MDW. As the master problem is relatively easy to solve, we solve it to optimality at each iteration.

Step 5: Solve subproblem SDW. The x_{ij} and y_i are required to be integer, but subproblem SDW is equivalent to LR, (5.56)–(5.59), and it can be decomposed into $|I|$ knapsack problems, with objective function $\sum_{i \in I} \left(\sum_{j \in J} (c_{ij} - u_j) x_{ij} + f_i y_i \right)$. Thus, for each $i \in I$ for which $\sum_{j \in J} (c_{ij} - u_j) x_{ij} < -f_i$, the corresponding y_i is set to 1, otherwise to 0.

Step 6: Construct a feasible solution. We generate a feasible SCFLP solution using the same procedure used for the Lagrangean metaheuristic by solving problem GAP, (5.60)–(5.63), defined according to the SDW solution.

Step 8: Stop condition. If $z_{SDW}(\mathbf{u}, \alpha) \geq 0$ we stop because we have reached the optimal solution of the master problem MDW. Otherwise, we add the new column generated by SDW to the master problem MDW and we go on.

5.4.3 Solving the SCFLP with a Benders Metaheuristic

Also a Benders metaheuristic approach offers a number of possibilities to decompose our model and to generate feasible solutions for the SCFLP. The implementation of BendHeuristic that we have chosen is as follows.

Step 1: Identify a master MB and an "easy" subproblem SP. A possible Benders decomposition of SCFLP involves keeping in the master the decision of which facilities to open, and assigning clients to open facilities as a subproblem. The subproblem is therefore a GAP again.

More in detail, the master problem is:

$$z_{MB} = min \sum_{i \in I} f_i y_i + z_{SP}(\mathbf{y}) \tag{5.72}$$

$$s.t.\ y_i \in \{0,1\}, \qquad i \in I \tag{5.73}$$

and the subproblem becomes:

$$z_{SP}(\mathbf{y}) = min \sum_{i \in I, j \in J} c_{ij} x_{ij} \tag{5.74}$$

$$s.t. \sum_{i \in I} x_{ij} = 1, \qquad j \in J \tag{5.75}$$

$$\sum_{j \in J} q_j x_{ij} \leq Q_i y_i, \qquad i \in I \tag{5.76}$$

$$x_{ij} \leq y_i, \qquad i \in I, j \in J \tag{5.77}$$

$$x_{ij} \in \{0,1\}, \qquad i \in I, j \in J \tag{5.78}$$

where constraints (5.77) are considered only if the integrality constraints (5.73) are relaxed.

Step 3: Solve master problem MB. As the master problem, even though NP-hard after the addition of Bender's cuts, was relatively easy to solve for the benchmark instances from the literature, we solved it to optimality at each iteration.

Step 6: Solve subproblem SP. The x_{ij} are required to be integer, but the subproblem is the same GAP we met in Subsection 5.4.1. The same considerations apply.

Step 11: Add to MB the Benders cut generated by problem SB. To get the subproblem's dual we relaxed constraints (5.78) into $x_{ij} \geq 0$, $i \in I, j \in J$. After associating dual variables w'_j, $j \in J$, to constraints (5.75), w''_i, $i \in I$, to constraints (5.76) and w'''_{ij}, $i \in I$, $j \in J$, to constraints (5.77), problem SB becomes:

$$z_{SB}(\mathbf{y}) = max \sum_{j \in J} w'_j + \sum_{i \in I} Q_i y_i w''_i + \sum_{i \in I} \sum_{j \in J} y_i w'''_{ij} \tag{5.79}$$

$$s.t.\ w'_j + q_j w''_i + w'''_{ij} \leq c_{ij}, \qquad i \in I, j \in J \tag{5.80}$$

$$w''_i \leq 0, \qquad i \in I \tag{5.81}$$

$$w'''_{ij} \leq 0, \qquad i \in I, j \in J \tag{5.82}$$

The master formulation, which includes the added cut, is as follows:

$$z_{MB} = min\ z$$

$$s.t.\ z \geq \sum_{i \in I} \left(f_i + Q_i w''_i + \sum_{j \in J} w'''_{ij} \right) y_i + \sum_{j \in J} w'_j \tag{5.83}$$

$$y_i \in \{0, 1\}, \qquad i \in I \tag{5.84}$$

5.5 Computational Results

We implemented the above described algorithms in C# and Fortran, this last was used by linking algorithms MT1R for solving knapsack problems and MTHG for getting a heuristic solution of GAP problems [23] (codes can be freely downloaded from the page `http://www.or.deis.unibo.it/knapsack.html`). The code was run on a 1.7 GHz laptop with 1Gb of RAM and .NET framework 2.0. Ilog CPLEX 11.1 was used as LP and MIP solver where required.

The benchmark instances are those used by Holmberg et al. [20]; they consist of 71 instances whose size ranges from 50 to 200 customers and from 10 to 30 candidate facility locations. The instances are divided into four subsets. Set 1 has customers and locations with coordinates randomly generated in the interval $[10, 200]$, problems p1 to p12 have 50 customers and 10 possible locations, problems p13 to p24 have 50 customers and 20 possible locations. Set 2 has locations generated in the interval $[10, 300]$. The assignment costs are based on a vehicle routing problem cost distribution (see [20] for details). Set 3 is based on vehicle routing test problems used by Solomon [30], while set 4 is generated as set 1 but the number of potential locations is 30 and the number of customers is 200.

In this section we present computational results for the three proposed metaheuristic procedures, namely, LagrHeuristic, DWHeuristic and BendHeuristic, and compare them with those obtained by the "*dfs*" variant of the VLSN heuristic proposed by [2], which is the best performing metaheuristic algorithm known for the SCFLP. The CPU times reported for *dfs* have been obtained on a PC with an Athlon/1200Mhz processor and 512 Mb RAM, under RedHat Linux 7.1.

Table 5.1 Computational results obtained with procedure LagrHeuristic.

Problem	**Lagrangean Metaheuristic**					**dfs**	
Sets	G_{LP}	G_H	G_L	T_{Best}	T_{Tot}	G_{dfs}	T_{dfs}
p1-p24 avg	0.79	0.01	0.12	0.16	4.38	0.00	0.54
p1-p24 max	2.19	0.06	0.66	0.85	14.80	0.00	1.85
p25-p40 avg	0.77	0.55	0.69	1.51	51.04	0.13	12.67
p25-p40 max	2.02	2.95	1.86	10.77	107.21	0.79	34.08
p41-p55 avg	0.84	0.30	0.31	0.91	10.46	0.03	1.62
p41-p55 max	2.00	2.02	1.86	3.68	31.55	0.18	5.47
p56-p71 avg	0.57	0.21	0.57	27.27	475.15	0.02	15.97
p56-p71 max	2.28	1.06	1.95	201.74	1731.80	0.14	46.60

Let z_{MIP} and z_{LP} be the optimal solutions of problem SCFLP, (5.50)–(5.54), and of its LP relaxation, respectively. Let z_{UB} and z_{LB} be the upper and the lower bounds provided by the proposed procedures, respectively. In Tables 5.1, 5.2 and 5.3, for each set of test instances, we report the following average and maximum values:

G_{LP}: the percentage gap between the optimal MIP solution and the optimal LP solution, i.e., $G_{LP} = \frac{z_{MIP}-z_{LP}}{z_{MIP}} \times 100$;
G_H: the percentage gap between the heuristic solution provided by the proposed procedure and the optimal MIP solution, i.e., $G_H = \frac{z_{UB}-z_{MIP}}{z_{MIP}} \times 100$;
G_L: the percentage gap between the lower bound provided by the proposed procedure and the optimal MIP solution, i.e., $G_L = \frac{z_{MIP}-z_{LB}}{z_{MIP}} \times 100$;
T_{Best}: the computing time in seconds required by the proposed procedure to reach the best heuristic solution found;
T_{Tot}: the total computing time in seconds required by the proposed procedure;
G_{dfs}: the percentage distance from optimality of *dfs*;
T_{dfs}: the CPU time in seconds taken by *dfs*.

5.5.1 Lagrangean Metaheuristic

Procedure LagrHeuristic was used with the α subgradient step control parameter (see [27]) initially set to 0.5, and multiplied by 0.9 when five consecutive non-improving iterations were detected. LagrHeuristic terminated either when an optimal solution was found, i.e., when $z_{LB} = z_{UB}$, when 5000 subgradient iterations were made, or when a time limit of 3600 seconds was reached.

The computational results for procedure LagrHeuristic are reported in Table 5.1. Figure 5.1 shows the evolution of the upper bound z_{UB} and of the lower bounds z_{LB} when LagrHeuristic is applied to instance *p11*. Proce-

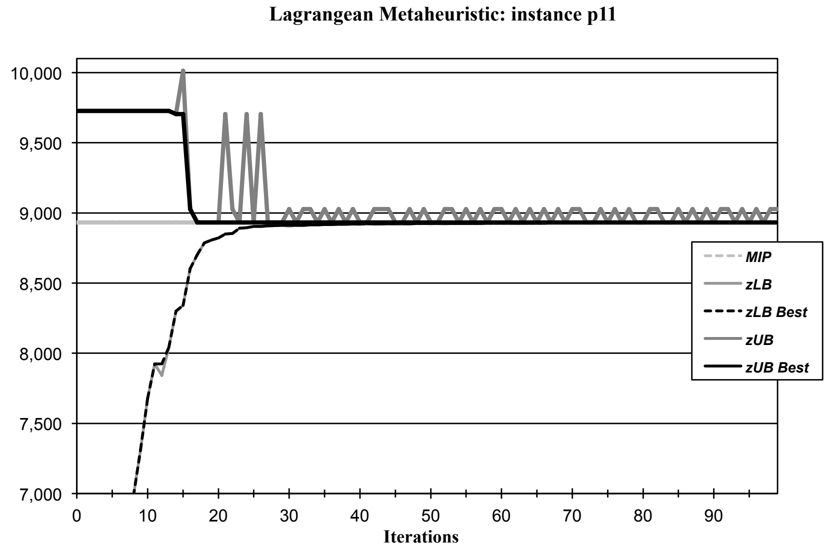

Fig. 5.1 Upper and lower bounds evolution of LagrHeuristic for the instance *p11*.

dure LagrHeuristic shows on all test problems a performance qualitatively similar to the one reported in Figure 5.1.

As repeatedly pointed out, the results we report here are not for showing that we have the best heuristic in the literature, but for showing that even a straightforward implementation of algorithm LagrHeuristic can get close to the state-of-the-art. This is apparent on Table 5.1 where there are not big differences with respect to *dfs* and where the existing gap is mainly due to few instances. It would be rather easy to close that gap by means of some trick on the subgradient algorithm, such as an α-restart or an adaptive anneal (not to mention a local search on the upper bound), but again, this would obfuscate our point.

We mention here again how the inclusion of dual information into the metaheuristic permits to determine the quality of the best solution found, and possibly its optimality. In our case, out of the 71 instances, 3 could be solved to optimality by the subgradient alone, which evolved weights that lead to the satisfaction also of the relaxed constraints, while 21 other ones were solved to proven optimality since the lower and the upper bound converged to the same cost. In all these cases the computation terminated before the maximum available CPU time, an option which is not available for primal-only heuristics.

Table 5.2 Computational results obtained with procedure DWHEURISTIC.

Problem Name	Dantzig-Wolfe Metaheuristic G_{LP}	G_H	G_L	T_{Best}	T_{Tot}	dfs G_{dfs}	T_{dfs}
p1-p24 avg	0.79	0.04	0.14	11.39	825.22	0.00	0.54
p1-p24 max	2.19	0.76	0.67	32.20	2558.42	0.00	1.85
p25-p40 avg	0.77	0.55	4.83	1096.97	3581.77	0.13	12.67
p25-p40 max	2.02	2.95	12.48	2028.75	3600.72	0.79	34.08
p41-p55 avg	0.84	0.42	0.60	231.95	2793.80	0.03	1.62
p41-p55 max	2.00	2.02	2.48	1246.37	3600.96	0.18	5.47
p56-p71 avg	0.57	9.19	50.19	2875.35	3600.47	0.02	15.97
p56-p71 max	2.28	34.81	100.00	3555.17	3603.10	0.14	46.60

5.5.2 Dantzig-Wolfe Metaheuristic

We initialized the master problem by adding a column corresponding to a dummy facility with a sufficient capacity to serve all customers, but with a fixed cost equal to a known upper bound to the optimal solution cost.

Procedure DWHEURISTIC terminates when no further columns can be added to the master problem. However, since the convergence can be slow, procedure DWHEURISTIC was also stopped when 20000 columns were generated or when a time limit of 3600 seconds was reached.

The computational results reported in Table 5.2 show that the convergence of our basic DWHEURISTIC is slow and is not competitive with the Lagrangean metaheuristic. This behavior is mainly due to the large number of iterations required to obtain a good lower bound and, building on it, good solutions. Figure 5.2 shows a trace in the case of instance *p11*, where about 700 iterations are required to reach a good primal solution and about 1100 iterations to reach a good lower bound. Figure 5.2 confirms the convergence of upper and lower bounds.

DWHEURISTIC finds difficulties in solving set 2 and set 4, where the given time limit is not enough to provide a sufficiently good lower bound. For set 4 the average gap between the primal solution and the lower bound is unsatisfactory.

Clearly this basic schema is not competitive and some modifications are required. For example, our basic implementation of DWHEURISTIC can be improved by adding more columns at each iteration and/or solving heuristically the subproblem SDW, given by (5.68)–(5.71).

5.5.3 Benders Metaheuristic

We initialize the BENDHEURISTIC in the same way as the master problem in the DWHEURISTIC. Procedure BENDHEURISTIC terminates when either no

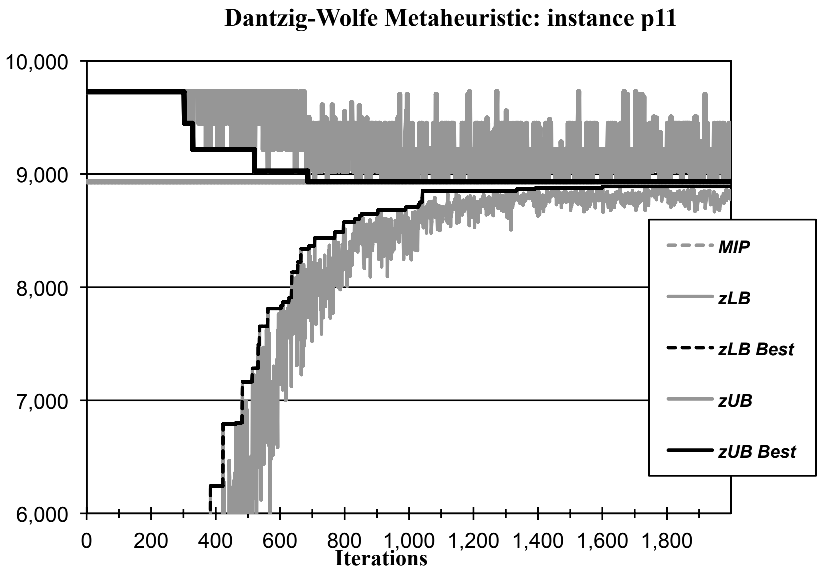

Fig. 5.2 Upper and lower bounds evolution of DWHeuristic for the instance *p11*.

further cuts can be added to the master problem, or when 5000 cuts were added or when a time limit of 3600 seconds was reached.

The computational results reported in Table 5.3 show a lower competitiveness of the basic BendHeuristic, when compared to its Lagrangean counterpart. The basic BendHeuristic outperforms procedure DWHeuristic, requiring less computing time to reach upper and lower bounds of similar quality. Figure 5.3 shows a trace in the case of instance p11, where it is evident that a good primal solution is generated quite quickly but the convergence of the lower bound is slow.

BendHeuristic finds difficulties in solving set 3 and particularly set 4, which has a cost structure that makes the master hard to solve when cuts start to be added. Clearly this basic schema is not competitive on these instances, more sophisticated considerations are required.

However, we believe that, since research on Benders based heuristics counts much less contributions than for instance Lagrange based ones, there is a wide room available for gaining insight on how to improve this basic functioning. For example, in our basic implementation of BendHeuristic we had the master solved to optimality, while it would be worthwhile to solve the master only heuristically.

Table 5.3 Computational results obtained with procedure BendHeuristic.

Problem	Benders Metaheuristic					dfs	
Name	G_{LP}	G_H	G_L	T_{Best}	T_{Tot}	G_{dfs}	T_{dfs}
p1-p24 avg	0.79	0.04	4.26	0.96	1612.28	0.00	0.54
p1-p24 max	2.19	0.76	13.30	6.90	3616.98	0.00	1.85
p25-p40 avg	0.77	0.55	0.34	17.26	2634.50	0.13	12.67
p25-p40 max	2.02	2.95	0.71	48.13	3602.35	0.79	34.08
p41-p55 avg	0.84	0.58	19.80	214.82	1781.79	0.03	1.62
p41-p55 max	2.00	2.02	68.66	1584.98	3611.57	0.18	5.47
p56-p71 avg	0.57	2.89	53.94	277.85	3636.65	0.02	15.97
p56-p71 max	2.28	14.20	80.52	1576.34	3836.41	0.14	46.60

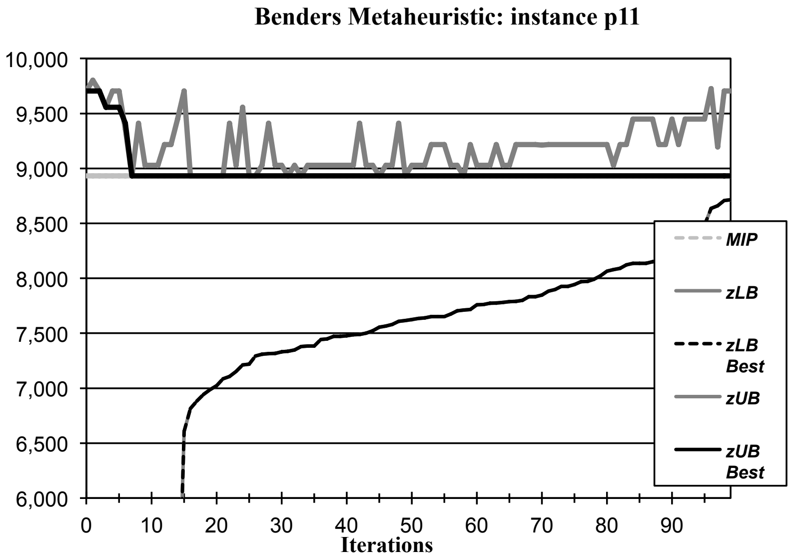

Fig. 5.3 Upper and lower bounds evolution of BendHeuristic for the instance *p11*.

5.6 Conclusions

This tutorial has shown a possibility to derive metaheuristic frameworks from the three main decomposition techniques from the literature, namely Lagrangean, Benders and Dantzig-Wolfe. This is an example, expanding a proposal first published in [9], of how techniques, originally designed for exact methods, could be included in a purely metaheuristic structure which shows the usual properties of simplicity, robustness and effectiveness.

The main point behind our argument is that research on metaheuristic methods should include elements from the mathematical programming literature in order to get a possibility to overcome the current computational limits, whenever these limits are felt to diminish the practical effectiveness of the available procedures.

We believe that the principal contribution of mathematically elaborate techniques comes from the use of bounds to the cost of optimal solutions and from dual information, two elements that can greatly help in directing search for better-than-current solutions and for determining the quality of the results achieved at any moment during search.

The computational results reported in Section 5.5 show that the heuristic solutions provided by the proposed metaheuristics can be of good quality even when the used dual information corresponds to a lower bound far from the optimal solution. Therefore, we can have good results also when the convergence is slow, with the only disadvantage of failing in reliably evaluating the quality. In the proposed frameworks it is mandatory to solve to optimality the subproblem. This can be a serious limitation when the relaxed problem is still difficult and a valid lower bound is not produced. In this case we can further relax the problem until the resulting problem is computationally tractable. However, this is an interesting issue that deserves further investigations to identify other approaches able to overcome all difficulties.

Research on how metaheuristics should make a strong point of mathematical modules is still at an embryonal level. We hope that this tutorial may help in fostering research along this line, that we believe to be promising.

References

1. M. Agar and S. Salhi. Lagrangean heuristics applied to a variety of large capacitated plant location problems. *Journal of the Operational Research Society*, 49:1072–1084, 1998.
2. R.K. Ahuja, J.B. Orlin, S. Pallottino, M.P. Scaparra, and M.G. Scutellà. A multi-exchange heuristic for the single source capacitated facility location problem. *Management Science*, 50(6):749–760, 2003.
3. A. Atamtürk, G. Nemhauser, and M.W.P. Savelsbergh. A combined Lagrangian, linear programming, and implication heuristic for large-scale set partitioning problems. *Journal of Heuristics*, 1:247–259, 1996.
4. F. Barahona and R. Anbil. The volume algorithm: producing primal solutions with a subgradient method. *Mathematical Programming*, 87:385–399, 2000.
5. J. Barcelo and J. Casanovas. A heuristic Lagrangean algorithm for the capacitated plant location problem. *European Journal of Operational Research*, 15:212–226, 1984.
6. M.S. Bazaraa, J. Jarvis, and H.D. Sherali. *Linear Programming and Network Flows*. John Wiley & Sons, 1990.
7. J.E. Beasley. Lagrangean relaxation. In C.R. Reeves, editor, *Modern heuristic techniques for combinatorial problems*, pages 243–303. Blackwell Scientific Publ., 1993.
8. J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:280–322, 1962.

9. M.A. Boschetti and V. Maniezzo. Benders decomposition, Lagrangean relaxation and metaheuristic design. *Journal of Heuristics*, 15(3):283–312, 2009.
10. M.A. Boschetti, V. Maniezzo, and M. Roffilli. A fully distributed Lagrangean solution for a P2P overlay network design problem. *Submitted for publication*, 2009.
11. M.A. Boschetti, A. Mingozzi, and S. Ricciardelli. An exact algorithm for the simplified multi depot crew scheduling problem. *Annals of Operations Research*, 127:177–201, 2004.
12. M.A. Boschetti, A. Mingozzi, and S. Ricciardelli. A dual ascent procedure for the set partitioning problem. *Discrete Optimization*, 5(4):735–747, 2008.
13. A. Caprara, M. Fischetti, and P. Toth. A heuristic method for the set covering problem. *Operations Research*, 47:730–743, 1999.
14. S. Ceria, P. Nobili, and A. Sassano. A Lagrangian-based heuristic for large-scale set covering problems. *Mathematical Programming*, 81:215–228, 1995.
15. F.A. Chudak and D.B. Shmoys. Improved approximation algorithms for a capacitated facility location problem. In *Proc. 10th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages S875–S876, 1999.
16. G.B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, 8:101–111, 1960.
17. H. Delmaire, J.A. Diaz, E. Fernandez, and M. Ortega. Reactive GRASP and tabu search based heuristics for the single source capacitated plant location problem. *INFOR*, 37:194–225, 1999.
18. R. Freling, D. Huisman, and A.P.M. Wagelmans. Models and algorithms for integration of vehicle and crew scheduling. *Journal of Scheduling*, 6:63–85, 2003.
19. K.L. Hoffman and M. Padberg. Solving airline crew scheduling problems by branch-and-cut. *Management Science*, 39:657–682, 1993.
20. K. Holmberg, M. Ronnqvist, and D. Yuan. An exact algorithm for the capacitated facility location problems with single sourcing. *European Journal of Operational Research*, 113:544–559, 1999.
21. J. Klincewicz and H. Luss. A Lagrangean relaxation heuristic for capacitated facility location with single-source constraints. *Journal of the Operational Research Society*, 37:495–500, 1986.
22. V. Maniezzo, M.A. Boschetti, and M. Jelasity. A fully distributed Lagrangean metaheuristic for a P2P overlay network design problem. In *Proceedings of the 6th Metaheuristics International Conference (MIC 2005)*, Vienna, Austria, 2005.
23. S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley, 1990.
24. A. Mingozzi, M.A. Boschetti, S. Ricciardelli, and L. Bianco. A set partitioning approach to the crew scheduling problem. *Operations Research*, 47:873–888, 1999.
25. A. Neebe and M. Rao. An algorithm for the fixed-charge assigning users to sources problem. *Journal of the Operational Research Society*, 34:1107–1113, 1983.
26. H. Pirkul. Efficient algorithm for the capacitated concentrator location problem. *Computers & Operations Research*, 14:197–208, 1987.
27. B.T. Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9:14–29, 1969.
28. M. Ronnqvist, S. Tragantalerngsak, and J. Holt. A repeated matching heuristic for the single source capacitated facility location problem. *European Journal of Operational Research*, 116:51–68, 1999.
29. H.D. Sherali and G. Choi. Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs. *Operations Research Letters*, 19:105–113, 1996.
30. M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35:254–365, 1987.
31. R. Sridharan. A Lagrangian heuristic for the capacitated plant location problem with single source constraints. *European Journal of Operational Research*, 66:305–312, 1991.

32. M.G.C. Van Krieken, H. Fleuren, and R. Peeters. A Lagrangean relaxation based algorithm for solving set partitioning problems. Technical Report 2004-44, CentER Discussion Paper, 2004.