A Logic-Based Benders Approach to Home Healthcare Delivery

Aliza Heching
Compassionate Care Hospice
aliza.heching@cchnet.net

J. N. Hooker
Carnegie Mellon University
jh38@andrew.cmu.edu

Ryo Kimura
Carnegie Mellon University
rkimura@andrew.cmu.edu

March 2018

Abstract

We propose an exact optimization method for home healthcare delivery that relies on logic-based Benders decomposition (LBBD). The objective is to match patients with healthcare aides and schedule multiple home visits over a given time horizon, so as to maximize the number of patients served while taking into account patient requirements, travel time, and scheduling constraints. Unlike classical Benders methods, LBBD allows us to exploit a natural decomposition of the problem into a master problem, solved by mixed integer programming, and a subproblem that decouples into small scheduling problems, solved by constraint programming. We report computational results based on data obtained from a major home hospice care provider. We find that LBBD is far superior to mixed integer programming on all but a few instances with narrow time windows. It solves problems of realistic size to optimality if the aim is to conduct staff planning on a rolling basis, without temporal dependencies between visits. We also find that a version of LBBD known as branch and check usually outperforms standard LBBD on the instances tested.

1 Introduction

Home healthcare is one of the world's most rapidly growing industries, due primarily to cost advantages and aging populations. The number of home healthcare aides in the United States, for example, has doubled in the last decade (Span 2016). Home care is not only less expensive than institutional care but offers other advantages. It allows patients to be treated in the comfortable and familiar surroundings of home, which are less stressful than an institutional environment. It

reduces the risk of acquiring drug-resistant infections that may spread in hospitals and nursing homes. The increasing availability of portable equipment and online consultation makes home care feasible for a growing range of conditions. Hospice care, which provides palliative rather than curative treatment, is particularly suited for the home. It may consist of a variety of services, including assistance in everyday tasks, nursing care, psychological counseling, physical therapy, religious/spiritual support, and bereavement services for the family.

The cost-effectiveness of home healthcare depends critically on the efficient assignment, scheduling and routing of healthcare aides, whom we call *aides* for short. Aides typically start their work shift at home or a central office, travel directly from one patient to the next, and return to home or office at the end of the shift. Aides must be qualified to service patients to whom they are assigned, and the schedule must observe a number of constraints imposed by the availability of aides, patient needs, work rules, and legal and regulatory requirements. These include time windows for each patient visit and each aide's departure from and return to home base.

We propose an exact method for solving the home healthcare problem that relies on *logic-based Benders decomposition* (LBBD). While many heuristic methods have been proposed for the problem, an exact method is particularly useful when it is necessary to determine what level and type of staffing are adequate to meet existing or projected patient needs. By maximizing the number of patients that can be served by a given set of aides, one can determine with certainty whether these aides are adequate, or additional aides must be hired and trained. Maximizing the population served also tends to result in less travel time and idle time for aides. The method can be modified to accommodate other objectives as well.

Logic-based Benders decomposition (Hooker 2000, Hooker and Ottosson 2003) is well suited for this application because the problem naturally decomposes into an assignment task and a scheduling task. The assignment portion of the problem becomes the Benders master problem, leaving the routing and scheduling for the Benders subproblem, which further decouples into a separate problem for each aide. While classical Benders decomposition requires that the subproblem be a linear or nonlinear programming problem (Benders 1962, Geoffrion 1972), LBBD generalizes the classical method to accommodate an arbitrary subproblem, such as the scheduling subproblem posed by home healthcare. A variant of LBBD, *branch and check*, has the same characteristics but solves the master problem once rather than repeatedly as in standard LBBD (Hooker 2000, Thorsteinsson 2001). It can be advantageous when the master problem is much harder to solve than the subproblem. We apply both standard LBBD and branch and check to the home healthcare problem.

A logic-based Benders approach has two additional advantages. The subproblem decouples into small scheduling problems that remain roughly the same size as the overall problem grows in size, allowing the algorithm to scale up to real-world applications. In addition, the master

problem and subproblem can be solved with methods that are best suited for each. We solve the master problem with mixed integer/linear programming (MILP), which is well suited for computing optimal assignments, while we solve the subproblem with the powerful scheduling algorithms in a constraint programming (CP) solver.

Our research was occasioned by a project undertaken for a major home hospice care organization. In this and in many other contexts, a weekly schedule is required, in which each patient is visited a specified number of times each week. The task is to determine which aide serves each patient, on which days of the week, and at what time of day. We therefore formulate a model that schedules patient visits over a given time horizon, with multiple visits per patient if so mandated by the patient care plan. Patients may also require visits from two or more different types of aides. The model can also accommodate a limited number of temporal dependencies between visits, as when patients require two or more aides to be present simultaneously for a joint task. If this kind of teamwork is the norm, however, the scheduling problem no longer decouples, and LBBD may not be an efficient solution method. We tested LBBD on problem instances that do not include such temporal dependencies.

Due to the nature of hospice (where eligible patients have a life expectancy of six months or less if the illness runs its normal course), the patient population is very dynamic. The problem presented to us was to update an existing aide schedule in response to projected changes in the patient population, as this allows the organization to anticipate staffing needs. We therefore focus primarily on the computation of a rolling schedule, a task that arises in many other applications as well. This means that when newly admitted patients replace some existing patients in the population, we find aides and visit times for the new patients while allowing the visit times of patients in service to be rescheduled. To maintain continuity of service – a key contributor to quality of service and patient satisfaction – we require that existing patients be served by the same aides on the same days as before. Other types of continuity constraints are easily incorporated into the model.

LBBD is especially well suited for computation of a rolling schedule, because the structure of the decomposition makes the problem much easier to solve when a subset of patients is replaced, even though the visit times are scheduled for the entire population. However, to test LBBD on a problem with very different characteristics, we also applied it to a Danish home care scheduling problem originally studied by Rasmussen et al. (2012). This application requires solving the problem from scratch rather than on a rolling basis.

We found that LBBD can solve instances of realistic size to optimality, with solution times ranging from a few seconds to a few minutes on nearly all instances, depending on the number of new patients. Branch and check proved to be significantly faster than standard LBBD on the hospice care instances, as might be expected, because the master problem is much harder to solve

than the subproblem. Branch and check is also far superior to MILP except on some instances with narrow time windows. Both forms of LBBD are dramatically faster than MILP on the Rasmussen instances. The master problem is easier to solve than the subproblem in some of these instances, and for these, standard LBBD tends to outperform branch and check.

The paper is organized as follows. After a review of previous work in Section 2 below, we formulate the home healthcare problem in Section 3, along with options for modifying the model. Section 4 provides a brief description of LBBD and its application to the home care problem. Section 5 states the scheduling subproblem and indicates how Benders cuts are generated and strengthened. Section 6 states the master problem and indicates how the Benders model can be altered to accommodate different objective functions. Section 7 indicates how branch and check differs from standard LBBD. A key element in the success of LBBD is the inclusion of a subproblem relaxation in the master problem, and we describe three possible relaxations in Section 8. Computational results are reported in Section 9, which is followed by conclusions and suggestions for future research.

2 Previous Work

Due to the difficulty of solving life-sized home healthcare delivery problems, nearly all existing methods are heuristic algorithms. Recent studies have used tabu search (Hertz and Lahrichi 2009, Rest and Hirsch 2016), pattern or column generation (Allaoua et al. 2013, Cappanera and Scutellà 2015), variable neighborhood search (Trautsamwieser and Hirsch 2011, Mankowska, Meisel, and Bierwirth 2014), variable neighborhood search combined with scatter search and other heuristics (Hiermann et al. 2015), constraint programming combined with heuristics (Nickel, Schröder, and Steeg 2012, Rendl et al. 2012), an inexact Benders method (Ciré and Hooker 2012), and separate solution of the rostering and scheduling components of the problem (Yalçındağ et al. 2014).

There are relatively few exact methods. Redjem et al. (2012) formulated the home healthcare problem with an MILP model but solved only small instances (15 patients). Chahed et al. (2009) used a specialized branch-and-bound algorithm to schedule home chemotherapy, but again tested it only on a very small instance (8 patients).

Rasmussen et al. (2012) scaled up to problem instances of realistic size by solving an MILP model of the problem with column generation and a specialized branching scheme. However, only some of the smaller instances (20–80 patients) were solved to optimality within an hour. The remainder were solved after grouping visits into clusters, so as to reduce the number of visits in the model (clustering can, of course, be used in LBBD if desired). This sacrifices optimality, but the authors report that the solutions are optimal or close to optimal on smaller instances that could also be solved optimally.

The results of Rasmussen et al. show that a column generation method is a viable approach to exact solution of the home care problem, at least for smaller real-world instances. A direct comparison with the results we report is difficult, due to differences in the problem solved. They compute a schedule for one day and one visit per patient, while we schedule multiple visits per patient over a time horizon of several days. On the other hand, we reschedule on a rolling basis, and their problem instances include temporal dependencies between visits.

Benders decomposition was introduced by Benders (1962) and extended to accommodate nonlinear programming subproblems by Geoffrion (1972). Logic-based Benders decomposition was developed by Hooker (1995, 2000) and Hooker and Ottosson (2003). The computational advantages of LBBD have since been demonstrated in a wide range of applications, partially surveyed by Hooker (2012) and Ciré, Çoban, and Hooker (2015). Guidelines for applying LBBD can be found in these references and Hooker (2007). Codato and Fischetti (2006) developed a method based on *combinatorial Benders cuts* that is closely related to LBBD and applies to the specific case of an MILP subproblem. Rahmaniani et al. (2017) provide an excellent survey of recent developments in Benders decomposition, including LBBD.

Branch and check was introduced by Hooker (2000) and first applied by Thorsteinsson (2001), who coined the term "branch and check." The method has received much less attention than standard LBBD, but Sadykov (2004, 2008) uses it to minimize the weighted number of late jobs on a single machine, and Lam and Van Hentenryck (2016) use it for vehicle routing on a congested network. Beck (2010) compares performance with standard LBBD on several different problems, as well as presenting a variation of branch and check that avoids solving the subproblems under certain circumstances.

This paper is based on methodology presented in a conference paper by Heching and Hooker (2016) but goes significantly beyond it. It modifies the decomposition to allow the option of requiring that a patient's visits occur at the same time each day. It further develops the time window relaxation described in the earlier paper, experiments with assignment and multicommodity flow relaxations, and compares standard LBBD with branch and check. It is also based on a new implementation that uses SCIP and Gecode as MILP and CP solvers, respectively, rather than commercial solvers.

3 The Model

We define the home healthcare problem over d days. Each patient j must be visited on v_j days during this time period by an assigned aide having a set Q_j of qualifications. Each visit has a duration of p_j time units and must take place within a time window $[r_j, d_j]$.

We will let binary variable $y_{ijk} = 1$ if aide i is assigned to visit patient j on day k. If there

are restrictions on which days patients can be visited, we indicate this with the generic constraint $y \in K$. For example, in a weekly schedule ($d = 7$), patient j may require two visits that must be scheduled on Tuesday and Thursday or Tuesday and Friday. There may be separation constraints to ensure that the visits are spaced evenly throughout the time horizon, particularly when the schedule is cyclic. For example, in a schedule with $v_j = 2$, we may require that the visits be separated by at least two days in the cycle, so that visits on Monday and Saturday would be infeasible.

Each aide i has a set Q'_i of qualifications. On any given day k, aide i begins at a starting location b_i, travels to the home of each assigned patient, and returns to the terminal location b'_i (normally $b_i = b'_i$). The travel time between aide/patient locations j and j' is $t_{jj'}$ time units, based on an optimal route that is calculated in advance. Aide i must leave location b_i during the time window $[r_{b_i}, d_{b_i}]$ and return to location b'_i during $[r_{b'_i}, d_{b'_i}]$. In addition, aide i cannot be on duty more than U_i time units during the scheduling horizon. We will suppose that the aide "clocks in" on arrival at the first patient of the day and "clocks out" on departure from the last patient, but this can be altered if desired.

The remaining variables of the model are as follows. We let binary variable $\delta_j = 1$ if patient j is assigned an aide, and binary variable $x_{ij} = 1$ if aide i is assigned to patient j. We also let integer variable $\pi_{ik\nu}$ denote the νth patient visited by aide i on day k, and real variable s_j denote the time that the visit to patient j starts on each day it occurs. We are therefore supposing that visits to patient j occur at the same time, because this simplifies notation and reflects the real-world situation we modeled. This assumption can easily be relaxed by adding a day index k to the variables s_j and modifying the model in the obvious way.

The problem can be stated as follows:

$$\max \sum_j \delta_j \tag{1}$$

$$\sum_i x_{ij} = \delta_j, \ \sum_{i,k} y_{ijk} = v_j \delta_j, \ \ \forall j \tag{2}$$

$$y_{ijk} \le x_{ij}, \ \ \forall i, j, k \tag{3}$$

$$x_{ij} = 0, \ \ \forall i, j \text{ with } Q_j \not\subseteq Q'_i \tag{4}$$

$$y_{ib_ik} = y_{ib'_ik} = 1, \ \ \forall i, k \tag{5}$$

$$y \in K \tag{6}$$

$$\delta_j, x_{ij}, y_{ijk} \in \{0, 1\}, \ \ \forall i, j, k \tag{7}$$

$$n_{ik} = \sum_j y_{ijk}, \ \ \text{all-different}\{\pi_{ik\nu} \mid \nu = 1, \ldots, n_{ik}\}, \ \forall i, k \tag{8}$$

$$\pi_{ik\nu} \in \{j \mid y_{ijk} = 1\}, \ \ \forall i, k, \text{ and } \nu = 1, \ldots, n_{ik} \tag{9}$$

$$\pi_{ik1} = b_i, \ \pi_{ikn_{ik}} = b'_i \ \ \forall i, k \tag{10}$$

$$r_j \le s_j \le d_j - p_j, \quad \forall i, j \tag{11}$$

$$s_{\pi_{ik\nu}} + p_{\pi_{ik\nu}} + t_{\pi_{ik\nu}\pi_{ik,\nu+1}} \le s_{\pi_{ik,\nu+1}}, \quad \forall i, k, \text{ and } \nu = 1, \ldots, n_{ik} - 1 \tag{12}$$

$$\sum_k \left(s_{\pi_{ik,n_{ik}-1}} + p_{\pi_{ik,n_{ik}-1}} - s_{\pi_{ik2}} \right) \le U_i, \quad \forall i \tag{13}$$

$$s_j \in R, \text{ for all } j; \quad \pi_{ikk'} \in Z, \ \forall i, k, k' \tag{14}$$

The objective (1) is to maximize the number of patients served. Other objectives are possible, as discussed in Section 6. Constraint (2) defines δ_j and ensures that patients are visited the required number of times by their assigned aide. Constraint (3) says that patients are only visited by aides who are assigned to them. Constraint (4) prevents patients from being served by aides without the proper qualifications. Constraint (5) ensures that aides visit their starting and ending locations. Constraint (6) enforces restrictions on which days visits may be scheduled.

The remainder of the model schedules the aides. This part of the model will appear in the subproblem, which will be solved by constraint programming (CP). Several of the constraints have a form that is peculiar to CP models, which typically contain "global constraints," or high-level constraints that convey information to the solver about special structure in the problem. Constraint (8) defines n_{ik}, the number of patients visited by aide i on day k. It also uses the *all-different* global constraint to require the variables $\pi_{ik\nu}$ to take on distinct values. Constraint (9) ensures that the patients who are sequenced for a given aide on a given day are in fact assigned to that aide. Constraint (10) requires aides to visit their starting location first and ending location last. Note that the variable π has another variable n_{ik} as one of its indices, a standard feature of CP models. Constraint (11) ensures visits occur within the required time windows. Constraint (12) ensures there is enough time to travel between locations, and constraint (13) enforces the maximum work time for aides. These two constraints likewise contain variable indices.

When a patient requires visits from two or more aides, the model can represent the patient as two or more distinct patients with different requirements. If there are *temporal dependencies* between the visits, they must be enforced by constraints on the start times. For example, if two aides must be present at the same time to perform a task together, we regard the patient as two patients j and j' and add the constraint $s_j = s_{j'}$. If one aide must pick up where the other left off, we can add the constraint $s_j + p_j = s_{j'}$. If the two visits should not overlap, we can give them nonoverlapping time windows, in which case no temporal constraints are necessary. Alternatively, if we want the windows to overlap but not the visits, we can impose a nonoverlapping constraint for the visits – a standard option in CP solvers.

As patients are frequently admitted or discharged from service, it is often desirable to modify the schedule to include the newly admitted patients and remove the discharged patients while rescheduling patients remaining in service as little as possible. The schedule may also be adjusted at the beginning of the day to reflect unavailability of aides or other contingencies. Such updates are

easily accommodated by adding constraints to the above model. Suppose that patient j is currently scheduled to be serviced by aide i at time t on days k for $k \in K_j$. To retain this arrangement, we merely set $y_{ijk} = 1$ for $k \in K_j$ in the model and modify the time window $[r_j, d_j]$ to $[t, t + p_j]$. To allow flexibility in the time of day, we leave the time window unchanged. To fix the aide assignment but not the day of the week, we set $x_{ij} = 1$ in the model and leave y_{ijk} unfixed. We can also require that only certain aides take on new patients (or patients whose aides are unavailable). We need only add the constraint $x_{ij} = 0$ for each of the remaining aides i and all new patients j.

4 Logic-Based Benders Decomposition

Logic-based Benders decomposition (LBBD) applies to optimization problems of the form $\min\{f(x, y) \mid C(x, y), C(x)\}$, where $C(x, y)$ is a constraint set containing variables x and y, and $C(x)$ a constraint set containing only x. Fixing x to a value $\bar{x}$ that satisfies $C(x)$ defines the *subproblem* $\min\{f(\bar{x}, y) \mid C(\bar{x}, y)\}$. In many applications, including the present one, the subproblem decouples into smaller problems that can be solved separately.

The subproblem is solved to obtain an optimal value v^*, which indicates that cost cannot be less than v^* when x is fixed to $\bar{x}$. We therefore have the bound $f(\bar{x}, y) \geq v^*$ for any y. The solution of the subproblem is analyzed to obtain a *Benders cut*, which is a more general bound $f(x, y) \geq \beta_{\bar{x}}(x)$ that applies for any value of x. The Benders cut is added to a *master problem*, which is solved to obtain the next value $\bar{x}$ to which x is fixed. The kth master problem is

$$\min\left\{v \;\middle|\; C(x);\;\; v \geq \beta_{x^i}(x),\; i = 1, \ldots, k-1\right\}$$

where $x^1, \ldots, x^{k-1}$ are the solutions of the first $k-1$ master problems. The optimal value v_k of the master problem is a lower bound on the optimal value of the original problem, and each $\beta_{x^i}(x^i)$ is an upper bound. The algorithm terminates when $v_k = \min\{\beta_{x^i}(x^i) \mid i = 1, \ldots, k-1\}$.

In principle, a Benders cut is found by examining the proof that v^* is optimal in the subproblem. The proof can be regarded as a solution of the *inference dual* of the subproblem. The same proof may yield a useful bound $\beta_{\bar{x}}(x)$ for values of x other than $\bar{x}$. In the special case of a linear programming problem, the inference dual is the linear programming dual. The proof takes the form of dual multipliers, which form the basis for a classical Benders cut. These concepts are discussed further in Hooker (2000, 2007, 2012), Hooker and Ottosson (2003).

In the present application, the objective function depends only on the master problem variables x, so that the problem has the form $\min\{f(x) \mid C(x, y), C(x)\}$. The subproblem becomes a feasibility problem, which may simply be written $C(\bar{x}, y)$. When the subproblem is feasible, the Benders algorithm terminates with an optimal solution. When it is infeasible, the proof of infeasibility may establish infeasibility for values of x other than $\bar{x}$, giving rise to a Benders cut in the form of a constraint $B_{\bar{x}}(x)$ that must be satisfied by any feasible x. One can always use a

simple *nogood cut* $x \neq \bar{x}$, but it is desirable to find stronger cuts. The master problem now has the form $\min\{f(x) \mid C(x);\ B_{x^i}(x),\ i = 1, \ldots, k-1\}$.

When the proof of infeasibility is not directly accessible from the solver, a Benders cut must be inferred in some other manner. One approach is to tease out the nature of the infeasibility proof by checking heuristically if the subproblem remains infeasible when some of the premises $x_j = \bar{x}_j$ are dropped. For example, if the subproblem remains infeasible when x_j is fixed to $\bar{x}_j$ only for $j = 1, \ldots, q$, we have the Benders cut $(x_1, \ldots, x_q) \neq (\bar{x}_1, \ldots, \bar{x}_q)$, which excludes more solutions than $x \neq \bar{x}$. This strategy has proved successful in several contexts and will be used here (Ciré, Çoban, and Hooker 2015, Hooker 2005, 2006, 2007). A second approach is to deduce from the structure of the subproblem an *analytical Benders cut* that strengthens the nogood cut. Analytical cuts have been used in a wide variety of applications, such as Terekhov, Beck, and Brown (2007), Hooker (2007), Fazel-Zarandi and Beck (2009), Peterson and Trick (2009), Çoban and Hooker (2013).

We decompose the home healthcare problem by assigning aides and visit days to patients in the master problem and visit times in the subproblem. Because the objective function depends only on master problem variables, the subproblem becomes a feasibility problem. In previous work, Heching and Hooker (2016) decoupled the subproblem into a separate scheduling problem for each aide and each day. However, it is often useful in practice to require a patient's visits on different days to occur at the same time. This couples the daily scheduling problems for each aide, but we nonetheless obtained better computational results than in the earlier paper.

When there are temporal dependencies between visits, the scheduling problems for the aides performing the visits must be coupled. Since each solution of the master problem may assign different aides to patients, the subproblem may decouple differently in each Benders iteration. LBBD is most effective when relatively few visits are subject to temporal dependencies, because this allows most of the aides to be scheduled separately.

5 Subproblem

The subproblem normally decouples into a separate scheduling problem for each aide. Each scheduling problem checks whether there is a schedule that observes the time windows while taking account of visit durations, travel times, and simultaneity constraints. If not, a Benders cut is generated as described below.

The subproblem formulation consists of the scheduling constraints (8)–(13) after the daily assignment variables y_{ijk} are fixed to the values $\bar{y}_{ijk}$ they receive in the solution of the previous master problem. The scheduling problem S_i for each aide i is

$$\text{all-different}\left\{\pi_{k\nu} \mid \nu = 1, \ldots, \bar{n}_k\right\}, \quad \forall k$$

$$\pi_{k1} = b_i,\ \pi_{k\bar{n}_k} = b'_i,\ \ \forall k$$
$$r_j \le s_j \le d_j - p_j,\ \ \forall j \in \bigcup_k P_{ik}$$
$$s_{\pi_{k\nu}} + p_{\pi_{k\nu}} + t_{\pi_{k\nu}\pi_{k,\nu+1}} \le s_{\pi_{k,\nu+1}},\ \ \forall k \text{ and } \nu = 1, \dots, \bar{n}_k - 1$$
$$\sum_k \left(s_{\pi_{k,\bar{n}_k-1}} + p_{\pi_{k,\bar{n}_k-1}} - s_{\pi_{k2}} \right) \le U_i,\ \ \forall i$$
$$\pi_{k\nu} \in P_{ik},\ \ \nu = 1, \dots, \bar{n}_k$$

where $P_{ik} = \{j \mid \bar{y}_{ijk} = 1\}$ and $\bar{n}_k = |P_{ik}|$. If the scheduling problem for two or more aides must be coupled, the variables $\pi_{k\nu}$ become $\pi_{ik\nu}$ as in the main model, and the above constraints are repeated for each of the aides i that must be coupled. Constraints are added to reflect temporal dependencies as described earlier.

The number of variables in the subproblem is limited by the fact that an aide can service only a limited number of patients in a day, say L, regardless of the overall size of the problem instance. Suppose that there are m aides and n patients, and there is no coupling of aides. Then there are at most n variables s_j and at most dL variables $\pi_{k\nu}$ in each of the m scheduling problems, which are solved separately. The number of variables in the subproblem therefore increases linearly with the number of patients.

If scheduling problem S_i is infeasible, we initially generate a nogood cut $\sum_k \sum_{j \in P_{ik}} (1 - y_{ijk}) \ge 1$ that prevents the same set of patients from being assigned to aide i on their corresponding days in subsequent assignments. To strengthen the cut, we re-solve S_i for subsets of P_{ik} using the following heuristic. For each k, we initially set $\bar{P}_{ik} = P_{ik}$, and for each $j \in \bar{P}_{ik}$ we do the following: remove j from $\bar{P}_{ik}$, re-solve S_i, and restore j to $\bar{P}_{ik}$ if the modified S_i is feasible. By replacing P_{ik} with $\bar{P}_{ik}$ in the nogood cut, we obtain a Benders cut that results in significantly better performance.

6 Master Problem

The basic master problem consists of constraints (1)–(7) of the original problem and the Benders cuts generated in all previous iterations, as described above. Because the problem is solved by MILP, the constraints (6) on days assignments must be encoded as linear inequality constraints. This is usually not difficult and will be illustrated in Section 9.

The master problem contains mnd variables y_{ijk}, mn variables x_{ij}, and n variables δ_j. The number of variables therefore increases linearly with the number of patients, as in the subproblem. Furthermore, when a rolling schedule is computed, many of the variables y_{ijk} and x_{ij} effectively drop out of the problem because they are fixed to 0 or 1.

We augment the master problem with a relaxation of the subproblem, because computational experience in Ciré, Çoban, and Hooker (2015) and elsewhere indicates that including such a

relaxation is crucial to obtaining good performance. Normally, the relaxation would contain only master problem variables y_{ijk}, x_{ij} and δ_j rather than variables in the subproblem. This is quite different from a classical relaxation, which contains variables from the problem being relaxed. We present a *time window relaxation*, described in Section 8.1 below, that contains only the variables y_{ijk}. A number of time window relaxations for other types of problems are described in Hooker (2012).

In an effort to find stronger subproblem relaxations, we also experimented with relaxations that contain variables from the subproblem. In particular, we used the classical assignment relaxation and a multicommodity flow relaxation, described in Sections 8.2 and 8.3, respectively. The solution values of the subproblem variables are discarded after the master problem is solved, and new solution values obtained when the subproblem is solved.

The decomposition can be modified to accommodate other objective functions, including those defined in terms of subproblem variables. In the latter case, the subproblem becomes an optimization problem, and the Benders cuts become inequalities as described in Section 4. Cuts of this kind can be constructed in analogy with the nogood cuts used here. Suppose, for example, that the hourly wage for aide i is c_i, and we wish to minimize total wages. We can convert U_i in the subproblem to a variable, and the scheduling problem for aide i now has an objective of minimizing $c_i U_i$. If z_i^* is the minimum cost found for aide i's schedule, the Benders cut consists of the inequalities

$$z_i \geq z_i^* \sum_k \sum_{j \in P_{ik}} (1 - y_{ijk}), \quad \forall i$$

and the master problem has the objective function $\sum_i z_i$, where z_i is the cost of aide i. This cut imposes the lower bound z_i^* on the cost of aide i if the same patients are assigned to the aide on the same days. The cut can be strengthened heuristically by re-solving the scheduling problems. Heuristics and subproblem relaxations for various objective functions are described, for example, in Hooker (2006, 2007, 2012).

7 Branch and Check

Branch and check can be useful when the master problem is much harder to solve than the subproblem. It solves the master problem only once with a branch-and-bound procedure, rather than repeatedly as in standard LBBD. Each time a feasible solution is found at a node of the branching tree, the current values of the master problem variables are sent to the subproblem, and the resulting subproblem is solved. Feasible solutions generated by primal heuristics may also be sent to the subproblem. If the subproblem is infeasible, one or more Benders cuts are generated, and they are enforced throughout the remainder of the branching process.

We will find that the scheduling subproblem generally solves much more rapidly than the master problem in the hospice care instances, which suggests that branch and check may be preferable to standard LBBD for these instances. In fact, branch and check could benefit from relaxations stronger than the time window relaxation, since it may be advantageous to invest more time in solving a master problem that better reflects the original problem. We test these hypotheses in the computational section below.

8 Subproblem Relaxation

We now describe the three subproblem relaxations we investigated for use in the standard LBBD and branch and check algorithms.

8.1 Time Window Relaxation

The time window relaxation generalizes a similar relaxation used in Heching and Hooker (2016). It is based on the idea that the total duration of visits and travel assigned to an aide must be no greater than the length of a time interval into which the visits must fit.

For each aide i and day k, define a set $\{[r_{b_i}, \alpha_{ik\ell}] \mid \ell \in L_{ik}\}$ of *backward intervals* that begin with the start of the aide's shift, and a set $\{[\beta_{ik\ell}, d_{b'_i}] \mid \ell \in L'_{ik}\}$ of *forward intervals* that end with the termination of the shift. The time window relaxation requires that the visits that are assigned to aide i on day k, and whose time windows lie inside a given backward interval, must have a total duration that fits in that interval. There is a similar requirement for forward time intervals. In the case of backward intervals, the minimum travel time from the previous visit is included in the visit duration, and in the case of forward intervals, the minimum travel time to the next visit is included.

To state the relaxation more precisely, let $J[t, t']$ be the set of patients whose time windows lie in the interval $[t, t']$, so that $J[t, t'] = \{j \mid [r_j, d_j] \subseteq [t, t']\}$. Let the *backward augmented duration* p'_{ijk} for a patient j, aide i and day k be the visit duration p_j plus the minimum travel time from the previous visit, which may be the aide's origin base. The *forward augmented duration* p''_{ijk} is p_j plus the minimum travel time to the next visit, which may be the aide's terminal base. So we have

$$p'_{ijk} = p_j + \min\left\{t_{b_i j}, \min_{j' \in J_{ik}} \{t_{j'j}\}\right\}, \quad p''_{ijk} = p_j + \min\left\{\min_{j' \in J_{ik}} \{t_{jj'}\}, t_{jb'_i}\right\}$$

where J_{ik} is the set of patients that are already assigned aide i on day k, or that have not yet been assigned to an aide. Thus the backward augmented duration is a lower bound on the time required to reach and carry out a visit, and similarly for the forward augmented duration.

We now observe that the sum of the backward augmented durations of visits in $J[r_{b_i}, \alpha_{ik\ell}]$ must

be at most the width of the backward interval $[r_{b_i}, \alpha_{ik\ell}]$, and similar for any forward interval:

$$\sum_{j \in J[r_{b_i}, \alpha_{ik\ell}]} p'_{ijk} y_{ijk} \leq \alpha_{ik\ell} - r_{b_i}, \quad \ell \in L_{ik}; \qquad \sum_{j \in J[\beta_{ik\ell}, d_{b'_i}]} p''_{ijk} y_{ijk} \leq d_{b'_i} - \beta_{ik\ell}, \quad \ell \in L'_{ik} \tag{15}$$

This is because the visits and travel to each visit must fit between the beginning of the aide's shift and the end of the backward interval, and similarly for a forward interval. Inequalities (15), collected over all aides i and days k, comprise a time window relaxation.

To obtain tighter inequalities (15), the backward and forward intervals should be chosen to have a large *density*. That is, the visits that can take place within them should have a large total duration relative to the width of the interval. To accomplish this, we need only consider the backward intervals $[r_{b_i}, d_j]$ and the forward intervals $[r_j, d_{b'_i}]$ for all patients j. The corresponding densities are

$$\rho_j = \frac{1}{d_j - r_{b_i}} \sum_{j' \in J[r_{b_i}, d_j]} p'_{ij'k}, \qquad \rho'_j = \frac{1}{d_{b'_i} - r_j} \sum_{j' \in J[r_j, d_{b'_i}]} p''_{ij'k}$$

respectively. We now let the set L_{ik} of backward intervals contain those intervals $[r_{b_i}, d_j]$ for which ρ_j is sufficiently large, and similarly for the set L'_{ik} of forward intervals.

The inequalities (15) are still fairly weak when scheduling all patients from scratch, because the shortest travel time from the last (or next) visit is a weak bound on the actual travel time. However, they are more effective when scheduling on a rolling basis, because the shortest travel time is computed only over patients who are already assigned aide i on day k or are unassigned.

Additionally, we add the following relaxation of the "no overtime" constraint, which states that the combined duration of all patients visited by an aide cannot exceed the aide's work limit:

$$\sum_j v_j p_j x_{ij} \leq U_i, \quad \forall i \tag{16}$$

8.2 Assignment Relaxation

The second relaxation we considered is based on the assignment relaxation of the traveling salesman problem. We define a new binary variable $w_{ijj'k} = 1$ if aide i visits patient j immediately prior to patient j' on day k. We add the constraints

$$w_{ijb'_ik} + \sum_{j' \neq j} w_{ijj'k} = w_{ib_ijk} + \sum_{j' \neq j} w_{ij'jk} = y_{ijk}, \quad \forall i, j, k \tag{17}$$

$$w_{ib_ijk} + \sum_{j' \neq j} w_{ij'jk} = w_{ijb'_ik} + \sum_{j' \neq j} w_{ijj'k}, \quad \forall i, j, k \tag{18}$$

plus similar constraints in which j and/or j' is a home base. In addition, we include a simple feasibility constraint based on the total hours spent traveling/working in a day

$$\sum_j \left(t_{b_ij} w_{ib_ijk} + t_{jb'_i} w_{ijb'_ik} + p_j y_{ijk} + \sum_{j' \neq j} t_{jj'} w_{ijj'k} \right) \leq d_{b'_i} - r_{b_i}, \quad \forall i, k \tag{19}$$

a strengthened version of the inequalities (15) from the time window relaxation

$$\begin{aligned}
&\sum_{j\in J_{i\ell}} \Big(t_{b_i,j} w_{ib_ijk} + p_j y_{ijk} + \sum_{\substack{j'\in J_{i\ell}\\ j'\neq j}} t_{j'j} w_{ij'jk} \Big) \leq \alpha_{i\ell} - r_{b_i}, \quad \ell \in L_i \\
&\sum_{j\in J'_{i\ell}} \Big(p_j y_{ijk} + t_{jb'_i} w_{ijb'_ik} + \sum_{\substack{j'\in J'_{i\ell}\\ j'\neq j}} t_{jj'} w_{ijj'k} + \Big) \leq d_{b'_i} - \beta_{i\ell}, \quad \ell \in L'_i
\end{aligned} \tag{20}$$

as well as a strengthened version of the inequalities (16)

$$\sum_{j,k} p_j y_{ijk} + \sum_{j'\neq j} t_{jj'} w_{ijj'k} \leq U_i, \quad \forall i \tag{21}$$

8.3 Multicommodity flow relaxation

The third relaxation is based on the well-known multicommodity flow model for the vehicle routing problem with time windows (Cordeau and Laporte 2006). In addition to including all variables and constraints of the assignment relaxation, we include the variables s_{ijk} and the constraints

$$s_{ij'k} \geq s_{ijk} + p_j + t_{jj'} - M_{jj'}(1 - w_{ijj'k}), \quad \forall i, j, j', k \tag{22}$$

$$r_j \leq s_{ijk} \leq d_j - p_j, \quad \forall i, j, k \tag{23}$$

plus similar constraints in which j and/or j' is a home base. Here $M_{jj'} = \max\{0, d_j + p_j + t_{jj'} - r_{j'}\}$. The inequalities (19) and (20) are also strengthened slightly, by replacing r_{b_i} with s_{ib_ik} and $d_{b'_i}$ with $s_{ib'_ik}$.

If we constrain the variables $w_{ijj'k}$ to be binary (rather than continuous), the multicommodity flow relaxation becomes an MILP formulation of the original problem.

9 Computational Results

We tested standard LBBD (S-LBBD) and branch and check (B&Ch) on three datasets. One consists of real-world data provided by a major hospice care organization, one is obtained by modifying these data, and one consists of Rasmussen instances. We solved instances in the first two datasets on a rolling basis, and the instances in the third from scratch.

9.1 Hospice Care Instances

The task presented to us was to update an existing schedule so as to accommodate projected changes in the patient population. In particular, management wished to determine whether a given staff was adequate to serve the new population while meeting all requirements. We therefore maximized the number of patients that can be served by a specified work force with specified qualifications.

The aide assignments and scheduled days of the week were fixed for patients in service, but the time of day could be rescheduled.

The organization maintains a weekly schedule for each region it serves; some regions provide service seven days per week while others offer aide visits on weekdays only. In our dataset, visits are scheduled on weekdays only. The patients required multiple visits per week, in accordance with their plan of care; per patient request, these were all scheduled at exactly the same time of day. When there are two visits per week, consecutive visits should be separated by at least two days, and when there are three visits per week, these visits should be separated by at least one day. This was enforced in the master problem by replacing the generic constraint (6) with

$$y_{ijk} + y_{ij,k+\tau} \leq 1, \ \ \forall i, j \text{ with } v_j \in \{2, 3\}, \ \forall \tau, k \text{ with } 1 \leq \tau \leq 4 - v_j, \ 1 \leq k \leq 5$$

When formulating the time window relaxation, we noted that almost all the time windows either span most of the morning or most of the afternoon. It was therefore natural to use one backward interval ending at noon, and one forward interval beginning at noon, for each aide i. Thus we set $L_i = L'_i = \{1\}$ and $\alpha_{i1} = \beta_{i1} =$ noon for each i. This choice of α and β turns out to yield the highest-density non-trivial backward and forward interval for almost every aide i, where density is defined as in Section 8.1, and a nontrivial interval is one that includes and excludes at least one visit.

To obtain an initial schedule, we ran a greedy heuristic on an 80-patient population using 20 aides. Since the heuristic could only schedule 48 patients, we ran the standard LBBD algorithm on 60 of these patients, including 40 pre-scheduled by the greedy heuristic and 20 treated as new patients. They collectively required 270 visits, since the patients required between 2 to 5 visits per week. LBBD scheduled all of the new patients using 18 aides. The resulting 60-patient schedule was used as a starting point for computational tests. It is better than a heuristic schedule but worse than an optimal one, as one might expect when scheduling on a rolling basis.

We ran the tests for different rates of patient turnover in the 60-patient population. One instance was generated for each number $n = 8, \ldots, 25$ of new patients, where the new patients are assumed to be the last n patients in the list of 60. We designated 8 of the 18 aides as available to cover the new patients (along with their pre-assigned patients), because a minimum of 9 aides were required in nearly every instance. This allowed us to test computational performance near the phase transition for the problem. We set a maximum time limit of one hour.

9.2 Implementation

We implemented the algorithms using SCIP version 3.2.1 (Achterberg 2009) and the CP solver Gecode version 4.4.0 (Gecode Team 2016). The master problem was solved by SCIP, and the scheduling subproblems by Gecode. The SCIP presolver removes variables in the master problem

that are fixed to 0 or 1 by preassignments. The scheduling problems were formulated with a combination of Hamiltonian path constraints, unary resource constraints, and element constraints. The problems were solved with branch-and-bound search, first branching on sequence variables and then on start-time variables. We implemented S-LBBD via a custom dialog (for SCIP), which made calls to SCIP and Gecode to solve the master problem and solve the subproblem/generate Benders cuts, respectively.

We implemented B&Ch by incorporating into the master problem an additional constraint that enforced feasibility of the subproblems. We implemented a custom constraint handler for this constraint, which made calls to Gecode to determine feasibility of the subproblem and generated Benders cuts accordingly. We generated cuts for feasible solutions found by primal heuristics, as well as for those obtained in the branching process, because this proved to accelerate solution significantly. The solvers were run in Arch Linux on a laptop with an Intel Core i5 processor and 7.75 GB RAM.

We formulated an MILP model for the problem by modifying the well-known multicommodity flow model for the vehicle routing problem with time windows (Desrochers et al. 1988, Desrochers and Laporte 1991, Cordeau et al. 2007). The model consists of (1)–(7), (17)–(18), and (22)–(23). Because there are mn^2d variables $w_{ijj'k}$, the number of variables increases quadratically with the number of patients. This remains the case for a rolling schedule, because all patient visits can be resequenced even when many staff and day assignments are fixed. Thus while preassignments in a rolling schedule make the Benders master problem significantly smaller, they have relatively little effect on the size of the MILP model, in which most of the variables are sequencing variables $w_{ijj'k}$.

SCIP uses reliability branching and pseudocosts by default (Achterberg 2009). However, we turned off reliability branching for the B&Ch master problem, because in this context, SCIP 3.2.1 occasionally attempts to branch on a variable whose value has become fixed, throwing an error. This is likely due to the pseudocost calculations becoming invalid over time as B&Ch generates global cuts. The removal of reliability branching for B&Ch appears otherwise to be of little consequence, because the results are very similar with and without reliability branching on instances where no error is generated.

9.3 Results for Hospice Care Instances

Computational results for the hospice care instances appear in Table 1. The table shows the number of new patients in each problem instance, as well as the number of patients covered in the optimal solution. Computation times are indicated for all three methods, as well as the number of Benders iterations for S-LBBD. Although we tested the Benders methods using all three relaxations, Table 1 shows results for the time-window relaxation only, because it proved to be by far the most effective. Computation times for MILP are those obtained by SCIP.

Table 1: Solution times for 60-patient hospice care instances requiring 270 visits

New patients	New visits	Patients covered	MILP Time (s)	S-LBBD Iters	S-LBBD Time (s)	B&Ch Time (s)
8	40	60	43.6	7	3.17	**0.63**
9	45	59	41.0	13	5.99	**0.71**
10	50	59	46.6	7	3.27	**0.74**
11	55	59	53.3	11	5.63	**0.70**
12	60	59	53.2	12	6.49	**1.30**
13	65	59	63.0	21	12.3	**1.11**
14	70	58	113	84	72.3	**9.28**
15	75	58	223	86	77.0	**9.78**
16	80	58	844	91	98.5	**43.5**
17	85	59	1591	93	106	**31.1**
18	90	58	3017	116	202	**62.0**
19	95	58	1189	119	388	**90.0**
20	100	57	1016	124	1251	**600**
21	105	58	923	168	1272	**380**
22	110	58	*	217	951	**523**
23	115	58		264	*	**2092**
24	120	*				

*Computation time exceeded one hour.

Both S-LBBD and B&Ch are faster than MILP on nearly every instance, and B&Ch is consistently superior to S-LBBD. In fact, B&Ch is almost always between one and two orders of magnitude faster than MILP. These results indicate that a Benders method can scale up to problem instances of realistic size. Patient records indicate that a 5–8% turnover per week is typical in practice. Therefore B&Ch allows staff planning as much as 6 weeks in advance for 60 patients collectively requiring 270 visits, with computation times ranging from less than a second to ten minutes, depending on the number of new patients. This is adequate for many if not most hospice care situations.

As indicated by Table 2, solution of the subproblem consumes less that one percent of the S-LBBD solution time for the hospice care instances. The superior performance of B&Ch over S-LBBD is therefore consistent with our hypothesis that B&Ch may benefit from fast solution of the subproblem.

We also hypothesized that B&Ch may benefit from including tighter relaxations in the master problem, such as the assignment and multicommodity flow relaxations discussed earlier. Table 3 compares the performance of these two relaxations with the time window relaxation. The tighter relaxations lead to much worse performance, refuting the hypothesis. This might be explained

Table 2: Percent of solution time devoted to subproblem

	S-LBBD		B&Ch	
Instances	Avg	Max	Avg	Max
Original 60-patient instances	0.1	0.2	1.4	3.9
Narrow time windows	0.1	0.1	2.8	6.0
Fewer visits per patient	0.0	0.1	1.7	3.5
Rasmussen, weighted objective	0.4	0.8	6.3	13.6
Rasmussen, covering objective	1.2	1.5	85.6	99.7

by the fact that the tighter relaxations result in many fewer Benders cuts, and in fact none at all for the multicommodity flow relaxation. Since a cut is generated at each feasible node of the search tree at which the subproblem is infeasible, this indicates that the search discovers fewer feasible nodes when the relaxation is tighter. This is presumably because the tighter bound allows backtracking at a higher level in the tree. Because there are fewer cuts, less information is obtained from the subproblem, and in fact no information in the case of the multicommodity flow relaxation. Evidently, the reduced information flow results in poorer performance.

Given these results, one might question whether even the time window relaxation is helpful, especially when patient time windows span half a day as in the test instances. Table 4 reveals that the time window relaxation yields a significant, if not dramatic, reduction in computation time. It should therefore be included in the master problem. Table 4 also shows the advantage of generating cuts from feasible solutions obtained by primal heuristics, rather than solely from those obtained in the branching process.

9.4 Modified Hospice Care Instances

To clarify further the effect of problem structure on the performance of S-LBBD and B&Ch, we solved two modifications of the hospice care problem.

The first modification uses much narrower patient time windows. We replaced the original time windows with time windows centered around each patient's visit as scheduled in the initial heuristic solution. We then set the length of the time window to be twice that of the visit duration.

The second modification is inspired by the fact that over 80% of patients require five visits per week in the original dataset. This was the situation as presented by the company, but one might ask how performance differs when there are fewer visits per week. We therefore changed the number of required visits per week for each patient to a uniformly drawn random number from 1 to 5, with the duration of each visit is kept the same. We ran the greedy heuristic to produce a new initial schedule.

Table 3: Performance of branch and check with three types of relaxation

New	Time window relaxation			Assignment relaxation			Multicommodity flow relax.		
patients	# nodes	# cuts	Time (s)	# nodes	# cuts	Time (s)	# nodes	# cuts	Time (s)
8	1	5	0.63	45	3	35.8	30	0	56.6
9	2	14	0.71	35	0	30.9	16	0	61.0
10	2	16	0.74	18	0	29.4	37	0	64.3
11	1	19	0.70	38	2	30.3	125	0	159
12	120	57	1.30	788	9	36.2	244	0	208
13	38	50	1.11	304	3	37.1	292	0	459
14	7691	202	9.28	19103	36	181	*	*	*
15	7976	221	9.78	16866	52	254			
16	44197	290	43.5	70486	52	1394			
17	20316	332	31.1	59337	51	3073			
18	51890	467	62.0	*	*	*			
19	65085	520	89.6						
20	481199	789	600						
21	217671	745	380						
22	348012	860	523						
23	1010641	1386	2092						
24	*	*	*						

*Computation time exceeded one hour.

The results appear in Table 5. For the instances with narrow time windows, the pure MILP formulation actually outscales S-LBBD and B&Ch. This is perhaps not surprising, because the time windows are so narrow that their position already determines the schedule to a great extent, and because scheduling is the more difficult task for MILP.

The instances with fewer visits per week are less constrained and therefore further from the phase transition. To correct for this, we reduced the number of available aides to 6, which is enough to cover all but one patient in the optimal solutions. B&Ch is far superior to MILP on these instances, and it remains faster than S-LBBD as well.

These results suggest that the advantage of Benders methods is robust, except when time windows become narrow enough to severely constrain the schedule.

9.5 Rasmussen Instances

Rasmussen et al. (2012) provided us four real-world instances obtained from two Danish municipalities. The task is to assign a given set of crews, each containing members with specific skills, to a population of patients who require certain skills. The municipalities did not provide temporal

Table 4: Effect of time window relaxation and primal heuristic cuts (PHC) on computation time (seconds)

New	S-LBBD		B&Ch		
patients	No relax	Relax	No relax	No PHC	Relax & PHC
8	19.4	3.17	0.95	0.61	0.63
9	11.4	5.99	1.00	0.90	0.71
10	24.2	3.27	1.03	1.19	0.74
11	20.8	5.63	1.29	1.44	0.70
12	16.8	6.49	1.64	1.18	1.30
13	41.1	12.3	3.01	5.50	1.11
14	132	72.3	26.1	28.2	9.28
15	161	77.0	24.3	50.2	9.78
16	232	98.5	52.2	119	43.5
17	128	106	30.5	153	31.1
18	604	202	265	517	62.0
19	957	388	165	632	90.0
20	1185	1251	2378	2998	600
21	4200	1272	3522	1938	380
22	*	951	2033	1190	523
23		*	3045	*	2092
24			*		*

*Computation time exceeded one hour.

dependencies with the instances, as they were handled on an ad-hoc basis, but Rasmussen et al. added a set of dependencies to test their algorithm adequately. We did not include them because our method is designed for problems without temporal dependencies. Rasmussen et al. solved the problem over a time horizon of a single day, while we solved an equivalent 5-day problem that requires that each patient be visited every day at the same time.

The original objective of the Rasmussen instances is to minimize a weighted sum of travel cost, matching costs, and number of uncovered patients. The weights are adjusted so that as many patients as possible are covered, after which matching costs are minimized, followed by travel costs. The matching costs are indicated by giving each crew-patient pair a cost (positive for an undesirable match and negative for a desirable one). We solved each instance twice: once while minimizing a weighted sum of the matching cost and number of uncovered patients, and again while maximizing the number of patients covered.

For the time window relaxation, we explicitly found the best forward interval and best backward interval for each problem instance during the pre-processing phase via an exhaustive search over all reasonable breakpoints (i.e., the ends and starts of task/patient time windows).

Table 5: Solution time (s) for modified hospice care instances

New	Patients	Narrow time windows			New	Patients	Fewer visits per week		
patients	covered	MILP	S-LBBD	B&Ch	patients	covered	MILP	S-LBBD	B&Ch
8	60	53.1	10.1	**1.37**	12	58	58.9	17.0	**0.91**
9	59	40.0	11.3	**1.13**	13	58	58.6	20.5	**1.05**
10	59	40.5	17.9	**1.44**	14	58	71.3	30.6	**1.33**
11	59	41.8	22.6	**1.75**	15	58	123	80.0	**1.32**
12	59	43.4	28.8	**1.11**	16	58	167	259	**1.94**
13	59	42.3	28.3	**1.41**	17	58	253	521	**1.79**
14	59	45.2	62.8	**2.97**	18	58	3357	472	**3.45**
15	59	47.0	69.0	**5.25**	19	58	2364	710	**3.23**
16	59	59.5	96.5	**3.67**	20	59	1518	519	**5.36**
17	59	106	233	**11.0**	21	59	1811	759	**4.52**
18	58	127	349	**69.1**	22	59	3636	717	**5.29**
19	58	**137**	425	164	23	60	*	767	**3.87**
20	57	**153**	557	160	24	60		1990	**3.48**
21	57	**171**	993	437	25	60		2040	**91.6**
22	57	**254**	997	1818	26	60		2577	**4.57**
23	58	**524**	*	*	27	60		2693	**376**
24	58	**903**			28	60		4200	**3834**
25	58	**2369**			29	60		*	*
26	*	*							

*Computation time exceeded one hour.

The number of patients in the four instances hh, ll1, ll2, and ll3 are 150, 107, 60, and 61, respectively. We found that MILP could not come close to solving any of these, and the MILP model for hh was in fact too large to load into the solver. To allow a meaningful comparison with MILP, we reduced the number of patients to 30 in each instance, keeping the number of crews the same. At this point, MILP could solve two of the instances with the weighted objective within an hour, albeit none with the covering objective. As Table 6 indicates, the Benders methods are dramatically superior to MILP, easily solving all 8 instance-objective combinations. Interestingly, standard LBBD is usually faster than B&Ch when the covering objective is used. This is again consistent with the hypothesis that B&Ch is preferable to S-LBBD only when the subproblem solves rapidly, because in these instances, the subproblem consumes a much larger fraction of solution time than in other instances (Table 2).

Table 6: Solution time (s) for modified Rasmussen instances

			Weighted objective			Covering objective		
Instance	Patients	Crews	MILP	LBBD	B&Ch	MILP	LBBD	B&Ch
hh	30	15	*	3.16	**1.41**	*	**23.3**	441
ll1	30	8	*	1.74	**0.43**	*	108	**1.41**
ll2	30	7	2868	1.56	**0.32**	*	**1.38**	6.45
ll3	30	6	1398	2.16	**0.30**	*	**3.07**	5.98

*Computation time exceeded one hour.

10 Conclusions and Future Work

We developed an exact solution method for the home healthcare problem using logic-based Benders decomposition (LBBD). We formulated the problem so as to maximize the number of patients served by a given staff with given qualifications, while taking into account patient requirements, travel time, and scheduling constraints. We create a schedule spanning several days during which patients may receive multiple visits from the same healthcare aide, or visits from multiple aides. Unlike most competing methods developed for this problem, LBBD computes an optimal schedule and therefore allows planners to determine with certainty whether a given work force can serve a given patient population. We tested both standard LBBD and a variant of LBBD (branch and check) that solves the master problem only once.

Based on computational tests in a real-world setting, we conclude that LBBD, and in particular branch and check, can solve a problem of realistic size when scheduling on a rolling basis and there are no temporal dependencies between visits. By contrast, mixed integer/linear programming does not scale up, due to growth in the size of the model, except when time windows are so narrow that their position largely determines the schedule. LBBD has the advantage that the scheduling component of the problem breaks down into small scheduling problems that remain roughly constant in size as the overall problem size increases. In addition, LBBD is particularly well suited to computing a rolling schedule, because the structure of the decomposition makes the problem much easier to solve when only a subset of the patient population is replaced. In one real-world context, however, LBBD easily solved instances from scratch that were intractable for MILP.

Branch and check is faster than standard LBBD on most of the instances tested. However, when the subproblem is harder to solve than the master problem, standard LBBD tends to be faster. Branch and check also benefits from Benders cuts generated from feasible solutions found by primal heuristics. An issue for future research is whether such cuts could accelerate standard LBBD.

Even though branch and check solves the master problem only once, it does not benefit from adding variables to the master problem to create a tighter relaxation of the subproblem. The reason for this appears to be that an overly tight relaxation results in the creation of fewer Benders cuts during the branching process, and therefore too little information flow from the scheduling subproblem to the master problem. This observation could have implications for future applications of branch and check.

References

Achterberg T, 2009 *SCIP: Solving constraint integer programs. Mathematical Programming Computation* 1:1–41.

Allaoua H, Borne S, Létocart L, Calvo RW, 2013 *A matheuristic approach for solving a home health care problem. Electronic Notes in Discrete Mathematics* 41:471–478.

Beck JC, 2010 *Checking-up on branch and check.* Cohen D, ed., *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, 84–98 (New York: Springer).

Benders JF, 1962 *Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik* 4:238–252.

Cappanera P, Scutellà MG, 2015 *Joint assignment, scheduling and routing models to home care optimization: A pattern-based approach. Transportation Science* 49:830–852.

Çoban E, Hooker JN, 2013 *Single-facility scheduling by logic-based benders decomposition. Annals of Operations Research* 210:245–272.

Chahed S, Marcon E, Sahin E, Feillet D, Dallery Y, 2009 *Exploring new operational research opportunities within the home care context: The chemotherapy at home. Health Care Management Science* 12:179–191.

Ciré A, Çoban E, Hooker JN, 2015 *Logic-based Benders decomposition for planning and scheduling: A computational analysis.* Barták R, Salido M, eds., *COPLAS Proceedings*, 21–29.

Ciré A, Hooker JN, 2012 *A heuristic logic-based Benders method for the home health care problem*, presented at Matheuristics 2012, Angra dos Reis, Brazil.

Codato G, Fischetti M, 2006 *Combinatorial Benders' cuts for mixed-integer linear programming. Operations Research* 54:756–766.

Cordeau JF, Laporte G, 2006 *Modeling and optimization of vehicle routing and arc routing problems.* Appa G, Pitsoulis L, Williams HP, eds., *Handbook on Modelling for Discrete Optimization*, 151–191 (Springer).

Cordeau JF, Laporte G, Savelsbergh M, Vigo D, 2007 *Vehicle routing.* Barnhart C, Laporte G, eds., *Handbook in Operations Research and Management Science*, volume 14, 367–428 (Elsevier).

Desrochers M, Laporte G, 1991 *Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. Operations Research Letters* 10(1):27–36.

Desrochers M, Lenstra JK, Savelsbergh MWP, Soumis F, 1988 *Vehicle routing with time windows: Optimization and approximation.* Golden BL, Assad AA, eds., *Vehicle Routing: Methods and Studies*, 65–84 (Amsterdam: North-Holland).

Fazel-Zarandi MM, Beck JC, 2009 *Solving a location-allocation problem with logic-based Benders decomposition.* Gent IP, ed., *Principles and Practice of Constraint Programming (CP 2009)*, volume 5732 of *Lecture Notes in Computer Science*, 344–351 (New York: Springer).

Gecode Team, 2016 *Gecode: Generic constraint development environment.* Available from `http://www.gecode.org`.

Geoffrion AM, 1972 *Generalized Benders decomposition. Journal of Optimization Theory and Applications* 10:237–260.

Heching A, Hooker JN, 2016 *Scheduling home hospice care by logic-based Benders decomposition.* Quimper CG, ed., *CPAIOR Proceedings*, volume 9676 of *Lecture Notes in Computer Science*, 187–197 (Springer).

Hertz A, Lahrichi N, 2009 *A patient assignment algorithm for home care service. Journal of the Operational Research Society* 60:481–495.

Hiermann G, Prandtstetter M, Rendl A, Puchinger J, Raidl G, 2015 *Metaheuristics for solving a multimodal home-healthcare scheduling problem. Central European Journal of Operations Research* 23:89–113.

Hooker JN, 1995 *Logic-based Benders decomposition. INFORMS National Meeting (INFORMS 1995).*

Hooker JN, 2000 *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction* (New York: Wiley).

Hooker JN, 2005 *A hybrid method for planning and scheduling. Constraints* 10:385–401.

Hooker JN, 2006 *An integrated method for planning and scheduling to minimize tardiness. Constraints* 11:139–157.

Hooker JN, 2007 *Planning and scheduling by logic-based Benders decomposition. Operations Research* 55:588–602.

Hooker JN, 2012 *Integrated Methods for Optimization, 2nd ed.* (Springer).

Hooker JN, Ottosson G, 2003 *Logic-based Benders decomposition. Mathematical Programming* 96:33–60.

Lam E, Van Hentenryck P, 2016 *A branch-and-price-and-check model for the vehicle routing problem with location congestion. Constraints* 21:394–412.

Mankowska DS, Meisel F, Bierwirth C, 2014 *The home health care routing and scheduling problem with interdependent services. Health Care Management Science* 17:15–30.

Nickel S, Schröder M, Steeg J, 2012 *Mid-term and short-term planning support for home health care services. European Journal of Operational Research* 219:574–587.

Peterson B, Trick M, 2009 *A Benders' approach to a transportation network design problem.* van Hoeve WJ, Hooker JN, eds., *CPAIOR Proceedings*, volume 5547 of *Lecture Notes in Computer Science*, 326–327 (New York: Springer).

Rahmaniani R, Crainic TG, Gendreau M, Rei W, 2017 *The Benders decomposition algorithm: A literature review. European Journal of Operational Research* 259:801–817.

Rasmussen MS, Justesen T, Dohn A, Larsen J, 2012 *The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. European Journal of Operational Research* 219:598–610.

Redjem R, Kharraja S, Xie X, Marcon E, 2012 *Routing and scheduling of caregivers in home health care with synchronized visits. 9th International Conference of Modeling, Optimization and Simulation*, 06–08 (Bordeaux, France).

Rendl A, Prandtstetter M, Hiermann G, Puchinger J, Raidl G, 2012 *Hybrid heuristics for multimodal homecare scheduling.* Beldiceanu N, Jussien N, Pinson E, eds., *CPAIOR Proceedings*, volume 7298 of *Lecture Notes in Computer Science*, 339–355 (Springer).

Rest KD, Hirsch P, 2016 *Daily scheduling of home health care services using time-dependent public transport. Flexible Services and Manufacturing Journal* 28:495–525.

Sadykov R, 2004 *A hybrid branch-and-cut algorithm for the one-machine scheduling problem.* Régin JC, Rueher M, eds., *CPAIOR Proceedings*, volume 3011 of *Lecture Notes in Computer Science*, 409–415 (Springer).

Sadykov R, 2008 *A branch-and-check algorithm for minimizing the weighted number of late jobs on a single machine with release dates. European Journal of Operational Research* 189:1284–1304.

Span P, 2016 *Wages for home health care lag as demand grows. New York Times* September 23.

Terekhov D, Beck JC, Brown KN, 2007 *Solving a stochastic queueing design and control problem with constraint programming. Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI 2007)*, volume 1, 261–266 (AAAI Press).

Thorsteinsson E, 2001 *Branch and check: A hybrid framework integrating mixed integer programming and constraint logic programming.* Walsh T, ed., *Principles and Practice of Constraint Programming (CP 2001)*, volume 2239 of *Lecture Notes in Computer Science*, 16–30 (Springer).

Trautsamwieser A, Hirsch P, 2011 *Optimization of daily scheduling for home health care services. Journal of Applied Operational Research* 3:124–136.

Yalçındağ S, Matta A, Şahin E, Shanthikumar JG, 2014 *A two-stage approach for solving assignment and routing problems in home health care services.* Matta A, Li J, Sahin E, Lanzarone E, Fowler J, eds., *Proceedings of the International Conference on Health Care Systems Engineering*, volume 61 of *Proceedings in Mathematics and Statistics*, 47–59 (New York: Springer).