# A Heuristic Approach for Big Bucket Multi-Level Production Planning Problems

Kerem Akartunalı* Andrew J. Miller†

6 November 2007

**Abstract**

Multi-level production planning problems in which multiple items compete for the same resources frequently occur in practice, yet remain daunting in their difficulty to solve. In this paper we propose a heuristic framework that can generate high quality feasible solutions quickly for various kinds of lot-sizing problems. In addition, unlike many other heuristics, it generates high quality lower bounds using strong formulations, and its simple scheme allows it to be easily implemented in the Xpress-Mosel modeling language. Extensive computational results from widely used test sets that include a variety of problems demonstrate the efficiency of the heuristic, particularly for challenging problems.

**Keywords:** Integer Programming, Production Planning, Heuristics, Relax-and-Fix, Strong Formulations

## 1 Introduction

Production planning problems occur often in manufacturing environments and have therefore interested researchers and practitioners for decades. The early MRP (Materials Requirement Planning) approach, while accounting for bill-of-material structures, follows decision rules that are too simple to achieve feasible plans (let alone high quality plans) consistently, particularly due to not taking capacity constraints into account. The more advanced systems of MRP-II (Manufacturing resource planning) and ERP (Enterprise Resource Planning) combine different methodologies such as MRP and aggregate planning, but they fail to account

*Corresponding author. Work carried out at University of Wisconsin-Madison. Current address: Dept. of Mathematics and Statistics, University of Melbourne, Parkville, VIC 3010, Australia. e-mail: *K.Akartunali@ms.unimelb.edu.au*, Fax: +61-3-8344-4599

†Industrial and Systems Engineering, University of Wisconsin, 1513 University Avenue, 53706 Madison, WI, USA. e-mail: *amiller@engr.wisc.edu*

accurately for capacity. In this paper, we investigate multi-level capacitated production planning problems, for which MRP-II and ERP systems aim to find feasible production plans. Pochet and Wolsey [2006] give extensive details on the production planning, and Hoppe and Spearman [2000] explain MRP and ERP models in detail.

Since the seminal paper of Wagner and Whitin [1958] in 1958 on the uncapacitated lot-sizing problem, various forms of production planning problems have been investigated. Even though some special cases of lot-sizing problems can be solved in polynomial time (e.g. Florian and Klein [1971], Federgruen and Tzur [1991], Aggarwal and Park [1993] and van Hoesel and Wagelmans [1996]), real-world problems are much more complicated and computationally challenging, therefore more efficient methods are needed to tackle these difficulties. Moreover, the capacitated version of even the single-item lot-sizing problem is $\mathcal{NP}$-hard (Florian et al. [1980]), so these problems are also theoretically hard to solve. Given deterministic demand and other parameters such as inventory holding, production, and setup costs, lot-sizing problems seek a minimum-cost production plan that satisfies all the constraints of the finite horizon under consideration.

The solution approaches proposed so far for these problems vary from exact approaches based on mathematical programming to heuristic methods. Mathematical programming attempts to improve the formulations so that these can be solved faster and more efficiently, and in principle, these methods provide exact solutions with lower bounds. On the other hand, heuristics are practical methods to quickly discover good solutions, not necessarily the best, and these methods usually do not provide lower bounds that would guarantee solution quality.

## 1.1 Mathematical Programming Approaches

Mathematical programming provides tools for exact solution approaches for MIP problems, and we will give an overview of such approaches for production planning models. For production planning problems, polyhedral analysis has been useful in attempts to solve challenging models and to provide strong lower bounds. The first group of these exact methods strengthens the original formulation by adding valid inequalities. Barany et al. [1984a] propose valid inequalities that define the polytope of the single-item uncapacitated lot-sizing problem, which are also used for the strong formulation obtained in our framework. The capacitated problem with start-up costs is studied by Constantino [1996]. There are insightful polyhedral studies about some special cases of the single-item problem, see e.g. Pochet and Wolsey [1988], Pochet and Wolsey [1994] and Loparic et al. [2001]. Atamtürk and Muñoz [2004] investigate the bottleneck cover structure of single-item capacitated problems in their recent polyhedral study.

The second group of exact methods use extended reformulations of the model, which basically adds new variables to the problem to strengthen it. Krarup and Bilde [1977] provide a facility location reformulation of the problem, Eppen and Martin [1987] propose the shortest path reformulation which is equivalent to the facility location reformulation but

is smaller in size. Rardin and Wolsey [1993] define multicommodity reformulations for fixed charge network flow problems, which give stronger reformulations than the facility location reformulation for the multi-level lot-sizing problem. See also Belvaux and Wolsey [2001] and Wolsey [2002] for recent studies considering reformulation and modeling issues.

Although Pochet and Wolsey [1991] and Belvaux and Wolsey [2000] extend some of the single-item problem results to big-bucket problems, the current literature on strong formulations for big-bucket problems is limited. An exception is the study of multi-item problems in Miller et al. [2000] and Miller et al. [2003].

## 1.2 Production Planning Heuristics

Problem-specific heuristic algorithms have been proposed and used for complex, realistic lot-sizing problems, such as multi-level, capacitated problems. For a detailed study with comparisons of earlier lot-sizing heuristics, see Maes and van Wassenhove [1986]. Many heuristic approaches use decomposition ideas, and can be grouped as follows: 1) Lagrangian-based decomposition: Trigeiro et al. [1989] and Tempelmeier and Derstroff [1996] are primary examples for this group of heuristics. 2) Coefficient modification: Katok et al. [1998] propose a two-stage heuristic, where coefficient modification is used for finding an initial solution, and restricted LP relaxations of the second stage try to improve the initial solution. Van Vyve and Pochet [2004] propose a coefficient modification based heuristic algorithm to be used within branch-and-cut. 3) Forward scheme and Relax-and-fix: Afentakis and Gavish [1986] work on complicated BOM (Bills of Material) structures and use a forward scheme to obtain solutions while using Lagrangian relaxation for lower bounds. Belvaux and Wolsey [2000] solve practical lot-sizing problems using a special branch-and-cut system that employs relax-and-fix heuristics. Stadtler [2003] and Federgruen et al. [2007] primarily use the idea of "time windows" in their frameworks. Note that our approach also uses relax-and-fix idea. 4) Local search: A good example is the neighborhood search heuristic of Simpson and Erenguc [2005].

## 1.3 General MIP Heuristics

Real-world MIP problems are often computationally very difficult, hence researchers have often used heuristic approaches to tackle them. Heuristics can be classified as "Improvement Heuristics" (i.e. start with a solution and try to improve it) and "Constructive Heuristics" (i.e. start with no solution and try to find one).

Even though there are many problem-specific heuristics in the MIP literature, there are comparatively few general MIP heuristics. Recent general MIP heuristics include "Local Branching" by Fischetti and Lodi [2003], which uses the idea of branching on the neighborhoods of the current MIP solution, and "Relaxation Induced Neighborhood Search" (RINS) by Danna et al. [2005], which searches the neighborhood between the LP relaxation solution and the current MIP solution. The "Feasibility Pump" of Fischetti et al. [2005] provides a

framework designed for finding initial feasible solutions for very hard problems, and Balas et al. [2001] enumerate facets to find solutions for pure 0-1 problems. For a general review of MIP heuristics, such as LP-and-fix and relax-and-fix, refer to Pochet and Wolsey [2006], pp. 107-113.

### 1.4 Organization of the Paper

The approach we propose in this paper is a decomposition-based framework that incorporates strengthened formulations and ideas used in general MIP heuristics. Our approach is simple and flexible enough to apply to a variety of production planning problems.

In Section 2, we give the basic formulation of the problem, and we also discuss some generalizations of the problem since our goal is to define a heuristic flexible enough to handle many kinds of problems. In Section 3, we discuss strengthened formulations for big bucket problems. Section 4 covers the general MIP heuristic methods that we use in our framework. Section 5 is devoted to the explanation of the framework of the proposed heuristic and any options that may be used. In Section 6, we present extensive computational results obtained using a number of published data sets. Finally, we conclude with future directions in Section 7.

## 2 Problem Formulation

We consider the general multi-level lot-sizing problem, where the objective is to minimize the total cost. In order to achieve the optimal plan, production and inventory quantities, as well as setup decisions, have to be determined. While doing so, capacities should not be exceeded and demand should be satisfied. Before providing the basic formulation of the problem, we will present the notation.

*Indices and Sets:*

| | |
|---|---|
| $NT$ | Number of periods |
| $NI$ | Number of items |
| $NK$ | Number of machines |
| $endp$ | Set of all the end-items (items with external demand) |
| $\delta(i)$ | Set of immediate successors of item $i$ |

*Variables:*

| | |
|---|---|
| $x_t^i$ | Production variable for item $i$ in period $t$ |
| $y_t^i$ | Setup variable for item $i$ in period $t$ |
| $s_t^i$ | Inventory holding variable for item $i$ in period $t$ |

*Parameters:*

| | |
|---|---|
| $f^i$ | Setup cost for item $i$ (per setup) |
| $h^i$ | Inventory holding cost for item $i$ (per unit, per period) |
| $d_t^i$ | Demand for end-product $i$ in period $t$ |
| $d_{t,t'}^i$ | Total demand for end-product $i$ from period $t$ to $t'$, i.e., $d_{t,t'}^i = \sum_{\bar{t}=t}^{t'} d_{\bar{t}}$ |
| $r^{ij}$ | Number of items required of $i$ to produce one unit of $j$ |
| $a_k^i$ | Variable time necessary to produce one unit of $i$ on machine $k$ |
| $ST_k^i$ | Setup time for item $i$ on machine $k$ |
| $C^k$ | Capacity of machine $k$ |

Then, the formulation of the problem follows:

$$\min \sum_{t=1}^{NT}\sum_{i=1}^{NI} f^i y_t^i + \sum_{t=1}^{NT}\sum_{i=1}^{NI} h^i s_t^i \tag{1}$$

$$\text{s.t. } x_t^i + s_{t-1}^i - s_t^i = d_t^i \qquad t \in [1, NT], i \in endp \tag{2}$$

$$x_t^i + s_{t-1}^i - s_t^i = \sum_{j \in \delta(i)} r^{ij} x_t^j \qquad t \in [1, NT], i \in [1, NI] \backslash endp \tag{3}$$

$$\sum_{i=1}^{NI} (a_k^i x_t^i + ST_k^i y_t^i) \leq C^k \qquad t \in [1, NT], k \in [1, NK] \tag{4}$$

$$x_t^i \leq M_t^i y_t^i \qquad t \in [1, NT], i \in [1, NI] \tag{5}$$

$$y \in \{0,1\}^{NT \text{x} NI} \tag{6}$$

$$x \geq 0^{NT \text{x} NI} \tag{7}$$

$$s \geq 0^{NT \text{x} NI} \tag{8}$$

Here $M_t^i$ is the maximum amount of item $i$ that can be produced in $t$, and can be defined formally as follows:

$$M_t^i = \min(d_{t,NT}^i, \frac{C^k - ST_k^i}{a_k^i}) \qquad i \in endp$$

$$M_t^i = \min(\sum_{j \in endp} r^{ij} d_{t,NT}^j, \frac{C^k - ST_k^i}{a_k^i}) \qquad i \in [1, NI] \backslash endp$$

Constraints (2) and (3) ensure the production balance and demand satisfaction for end-items and other items respectively, (4) are the capacity constraints, (5) ensure that the setup variables are set to be 1 if there is positive production, and finally (6), (7) and (8) provide the integrality and nonnegativity requirements.

For an alternative formulation of the problem, we define echelon demand parameters $D_t^i$ and echelon stock variables $E_t^i$ as follows:

$$D_t^i = d_t^i + \sum_{j \in \delta(i)} r^{ij} D_t^j \qquad t \in [1, NT], i \in [1, NI] \tag{9}$$

$$E_t^i = s_t^i + \sum_{j \in \delta(i)} r^{ij} E_t^j \qquad t \in [1, NT], i \in [1, NI] \tag{10}$$

Note that echelon and original demands and stocks are equal for the end-items. Substituting (10) into (2) and (3) for $s_t^i$, and using the definition (9), we obtain the following equation, which can replace (2) and (3) in the original formulation:

$$x_t^i + E_{t-1}^i - E_t^i = D_t^i \qquad t \in [1, NT], i \in [1, NI] \tag{11}$$

To satisfy (8), we can define the following constraint:

$$E_t^i \geq \sum_{j \in \delta(i)} r^{ij} E_t^j \qquad t \in [1, NT], i \in [1, NI] \tag{12}$$

Finally, to eliminate the original inventory variables $s$, we define echelon inventory holding costs as $h^{*j} = h^j - \sum_{i:j \in \delta(i)} r^{ij} h^i$, and replace the objective function (1) with the following function:

$$\sum_{t=1}^{NT} \sum_{i=1}^{NI} f^i y_t^i + \sum_{t=1}^{NT} \sum_{i=1}^{NI} h^{*i} E_t^i \tag{13}$$

Hence, we can define the feasible region as $X = \{(x, y, E) | (4) - (7), (11), (12), E \geq 0\}$, which is the echelon stock reformulation of the original formulation and will be used in the remainder of the paper as the "basic formulation". The problem can be then defined as: $\min\{(13) | (x, y, E) \in X\}$.

We could easily include the possibility of overtime in the problem statement by updating the capacity constraint (4) and adding overtime cost to the objective function. On the other hand, the possibility of backlogging can be incorporated into the model by defining backlogging variables $b_t^i$. Aside from the objective function, the only change in the initial formulation would be the replacement of (2) with

$$x_t^i + s_{t-1}^i - s_t^i + b_t^i - b_{t-1}^i = d_t^i \qquad t \in [1, NT], i \in endp$$

Similarly, we can include backlogging into the echelon stock reformulation by replacing (11) with the following set of constraints:

$$x_t^i + E_{t-1}^i - E_t^i + \sum_{j \in endp} r^{ij} (b_t^j - b_{t-1}^j) = d_t^i \qquad t \in [1, NT], i \in [1, NI]$$

Note also that the first term in the definition of big-M quantity used in (5) should be updated to $D^i_{1,NT}$. Other additional characteristics could be incorporated as well, and the interested reader should refer to Pochet and Wolsey [2006] for more detail on lot-sizing models. Some of the test problems we consider in Section 6 incorporate some of these factors.

# 3 Strengthening the Formulation

Next, we consider several ways to strengthen formulations, in order to provide high quality lower bounds. First, we consider the $(\ell, S)$ inequalities proposed by Barany et al. [1984a] and generalized by Pochet and Wolsey [1991] to multi-level problems using echelon stocks, given as follows:

$$\sum_{t \in S} x^i_t \leq \sum_{t \in S} D^i_{t,\ell} y^i_t + E^i_\ell \qquad \ell \in [1, NT], i \in [1, NI], S \subseteq [1, \ell] \tag{14}$$

Note that because of multiple levels, (14) uses echelon demand and echelon stock variables defined in the last section. For each item, although there are an exponential number of $(\ell, S)$ inequalities, a simple polynomial separation algorithm exists (Barany et al. [1984a]). Note that $(\ell, S)$ inequalities define the convex hull of the uncapacitated single-item lot-sizing problem, see Barany et al. [1984b].

Next we discuss the facility location reformulation of Krarup and Bilde [1977] originally proposed for single-item problems. The reformulation uses new variables $u^i_{t,t'}$. Each of these variables indicates the amount of item $i$ that is produced in period $t$ to satisfy demand in period $t'$. This reformulation can be applied to the problem by adding the following three constraints into the basic formulation:

$$u^i_{t,t'} \leq D^i_{t'} y^i_t \qquad t \in [1, NT], t' \in [t, NT], i \in [1, NI] \tag{15}$$

$$\sum_{t=1}^{t'} u^i_{t,t'} = D^i_{t'} \qquad t' \in [1, NT], i \in [1, NI] \tag{16}$$

$$x^i_{t'} \geq \sum_{t=t'}^{NT} u^i_{t',t} \qquad t' \in [1, NT], i \in [1, NI] \tag{17}$$

Note that the projection of the facility location reformulation onto the space of original variables gives the convex hull of the uncapacitated single-item lot-sizing problem, see Krarup and Bilde [1977].

Let $X_{LS} = \{(x, y, E) | (4), (5), (7), (11), (12), (14), E \geq 0, 0 \leq y \leq 1\}$ and $X_{FL} = \{(x, y, E, u) | \ (4), (5), (7), (11), (12), (15) - (17), E \geq 0, 0 \leq y \leq 1\}$, i.e., the LP relaxations of the multi-level problem with all $(\ell, S)$ inequalities and the echelon stock facility location reformulation, respectively.

**Proposition 1** $\min\{(13) | (x, y, E) \in X_{LS}\} = \min\{(13) | (x, y, E, u) \in X_{FL}\}$

In words, adding $(\ell, S)$ inequalities to the original formulation and using the echelon stock facility location reformulation provide the same lower bound for the multi-level lot-sizing problem. For the proof of the proposition, please refer to Akartunalı [2007]. The corollary below follows immediately.

**Corollary 1** *The shortest path reformulation of Eppen and Martin [1987] is equivalent to the echelon stock facility location formulation, and therefore all three formulations provide the same lower bounds for the multi-level lot-sizing problem.*

Using the fact that both shortest path and facility location reformulations define the convex hull of the single-item uncapacitated problem (see e.g. Pochet and Wolsey [2006]), this corollary can be proven with the same technique used for the proposition. Even though all three methods discussed above provide the same lower bounds, the echelon stock facility location and shortest path reformulations have the disadvantage of making the problem size much larger than the $(\ell, S)$ inequalities, which are dynamically added to the formulation and deleted if inactive. Our experience, particularly from the companion paper Akartunalı and Miller [2007], shows that, for this reason, $(\ell, S)$ inequalities allow for more efficient branch-and-bound and identifying of feasible solutions. Also note that a slightly modified family of $(\ell, S)$ inequalities can be added to problems with backorders, as follows:

$$\sum_{t \in S} x_t^i \leq \sum_{t \in S} (D_{t,l}^i y_t^i + \sum_{j \in endp} r^{ij} b_{t-1}^j) + E_l^i \qquad l \in [1, NT], i \in [1, NI], S \subseteq [1, l]$$

Here, observe that the separation algorithm for these inequalities has the same logic as the one described before. Finally, note that in the case of overtime, the original $(\ell, S)$ inequalities are valid and will be added using the same separation algorithm.

There exist stronger inequalities and reformulations such as the multicommodity reformulation proposed by Rardin and Wolsey [1993], or multi-item single-period submodels used as by Miller et al. [2000], Miller et al. [2003]. However, we have found that using $(\ell, S)$ inequalities alone seems to be the most effective way to strengthen the formulation without making the problem computationally inefficient, as discussed in a companion paper Akartunalı and Miller [2007].

# 4 MIP Heuristics

Here we discuss two techniques that we will use in our proposed framework.

## 4.1 LP-and-Fix

LP-and-fix is a simple technique, closely related to "diving" (see Pochet and Wolsey [2006]) and works as follows: First, we solve the LP relaxation ($LPR$) of an $MIP$. Then, we check

all the integer variables in $LPR$ and fix those having integral values. Finally, the restricted $MIP$ with fixed variables is re-solved, with the hope of finding a solution to the original problem quickly. This scheme is employed in our heuristic framework both to provide an initial solution to be used as a cutoff value, and also throughout the algorithm to generate multiple production plans and to improve the best solution and hence the cutoff value.

## 4.2 Relax-and-Fix

Relax-and-fix is a heuristic method for problems with a special structure, and lot-sizing is well-suited for such a method since decisions made earlier in the horizon are more important then later ones. In this section, we discuss two previously developed heuristic methods that employ the "relax-and-fix" idea.

The first production planning tool that uses relax-and-fix is *bc-prod*, proposed by Belvaux and Wolsey [2000], which is a specialized branch-and-cut system for lot-sizing problems. The system first generates cutting planes, both default Xpress cuts and problem specific cuts, and then the relax-and-fix idea is applied using "time windows", horizons under consideration, for which the length of the window is predefined. The basic description of the "time windows" idea is as follows: Except for the variables in the periods of the predefined time window, relax all the binary variables to be continuous, solve the problem, and using the solution obtained, fix the binary variables in the window. The next window is then processed in the same manner.

Another heuristic algorithm based on relax-and-fix is proposed by Stadtler [2003], which we will refer to as "SH" (Stadtler's Heuristic) from now on. This approach uses time windows that overlap for better quality results. In such an approach, fixing variables in a window will occur only for the periods that do not overlap with the next window. Also, SH allows a period to be relaxed to continuous inside of the window, as well, because of using "bonuses", which are calculated for each different problem. In this framework, each subproblem is formulated and strengthened separately. Figure 1 summarizes the idea of "time windows" in SH. A recent paper of Federgruen et al. [2007] also uses a relax-and-fix approach with time windows.

The main advantage of *bc-prod* is that strong formulations provide good lower bounds for the problem, hence solution quality can be proven without extra computation. On the other hand, *bc-prod* does not have any overlapping windows, therefore it can miss considering the effect of future periods' setup decisions. Also, it can require long computational times because of long windows, large numbers of inequalities added to the formulation and the fact that it solves the subproblem of each window to optimality. Even though SH provides comparatively good results for hard test instances, it should be noted that the implementation is not straightforward because of complex calculations, such as the bonuses. Moreover, as in many other heuristic frameworks, it does not generate lower bounds. In order to determine solution quality, lower bounds need to be generated separately.

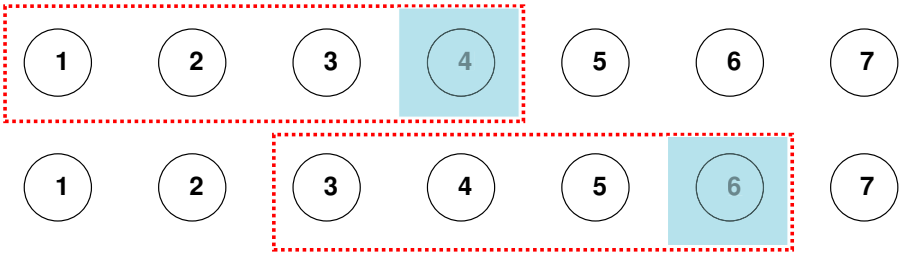


Figure 1: The first 2 "time windows" of SH, with window length of 4 periods, 2 periods overlapping and 1 period relaxed to be continuous

# 5 A Heuristic Framework

As in the last section, we use "time windows", or simply "windows", to refer to time intervals in which binary variables are forced to take binary values. For later periods, binary variables are relaxed to be continuous. The window has a "window length", i.e., length of the time horizon considered in the subproblem. The first part of the window which does not overlap with the next window is called the "fixing interval", since this is where the binary variables will be fixed once the subproblem is solved. Let $\alpha$ and $\beta$ indicate the window length and length of fixing interval, respectively. Note that in our approach the formulation is strengthened for the *entire* horizon, and not just within the window, as in SH.

We also use LP-and-fix idea to find feasible solutions for the original problem throughout the framework in order to provide multiple solutions and upper bounds (cutoff values) for later windows. Our computational experience is that using cutoff values significantly improves the solution process of many windows. Another difference from SH is that we use the objective function defined in Section 2 in each window. This is simpler to implement, and also helps us to gauge the effect of future setup decisions.

The structure of the general framework can be seen in the next page. We can describe the framework in words as follows: After generating violated $(\ell, S)$ inequalities and deleting inactive ones, the LP relaxation solution of the original problem is used to start LP-and-fix to find a first feasible solution for the original problem. Note that it is not guaranteed that LP-and-fix will find a solution, except backlogging is allowed. If a solution is found, it will be used to initialize the cutoff value. After initialization, we start with the first window, i.e., solve the problem defined in Section 2 where all the binary variables are relaxed to continuous for any period beyond the scope of the window. If extra time is available, i.e., relax-and-fix is complete before the pre-set time limit, LP-and-fix is used in order to generate a feasible solution for the original problem. At the end of LP-and-fix, all the binary variables fixed beyond the window will be unfixed before processing the next window. The same procedure will be repeated for future windows.

Here, note that the heuristic provides us a lower bound to the original problem in the

**Input:** Lot-sizing problem
**Output:** Multiple feasible solutions and lower bound for the problem
$(\ell, S)$ separation
Initialize upper bound and cutoff bound using LP-and-fix
**for** i=1 **to** numwin
  Relax the $y$ variables after the window to continuous $(t > (i-1)\beta + \alpha)$
  Solve the sub-problem
  Fix the $y$ variables in the fixing interval $(t \in [(i-1)\beta + 1, i\beta])$
  **if** extra time
    Enforce binary restriction on $y$ variables after the window
    Fix all $y$ variables having an integral value
    Solve the partially fixed MIP
    If solution found, reset cutoff value
    Unfix all $y$ variables after the window
**if** extra time at the end
  Improve the solution

first window since we are solving a partial linear relaxation of the problem in which the formulation is already strengthened. The computational results in the next section show that this lower bound is often competitive with the best known methods.

In the interest of speed, we seek to avoid running a window for a long time to find the optimal solution but obtain a reasonable solution in limited time and move on to the next window. Therefore we set a maximum time for each window. This issue is discussed in more detail in section 6. Also note that, due to these time limits and due to heuristics' nature, neither LP-and-fix nor relax-and-fix is guaranteed to find a solution. However, as the results in the next section indicate, the framework has worked without facing a problem with the control parameters used.

One important question to ask for both relax-and-fix and LP-and-fix is "which variables to fix". One basic concern for relax-and-fix is that the more we fix in a window, the higher probability we have for infeasibility in later windows. Hence, in our early tests, we tried to fix only 0's and only 1's, but no significant difference from fixing both 0's and 1's has been observed. On the other hand, for LP-and-fix, fixing all 0's and 1's appears to create problems occasionally, such as infeasibility and poor-quality solutions, hence only 1's are fixed in that part of the framework.

The heuristic framework depends on many parameters and options. These include the length of a window, whether we have to overlap consecutive windows, and how long the overlap should be. It is obvious that the shorter the length of the window, the easier it is to solve the related subproblem. However, this can deteriorate the solution quality since decisions become more myopic and since the number of windows grows and hence time

allocated to a window decreases as well. It is best to have a window length which neither takes too much time nor finds too poor solutions. When consecutive windows do not overlap, we deal with fewer windows, but this does not facilitate the consideration of setup decisions from later periods and therefore can result in bad solutions. After experimentation, we use the window length of 3, with an overlap of 1 period between windows. In this case, for example for a problem with $NT = 10$, there will be 5 windows to process.

We have found it advisable to allocate more time to earlier windows than to later windows because the problems are bigger in size and hence harder to solve, and also to employ LP-and-fix as often as possible in earlier periods in order to generate good solutions. After assigning a total time for a problem, this allocation process is achieved inside of the algorithm. We divided windows into four sets of the same size.Thus, if we have a total of 12 windows, the first three windows are in the first set, the next three windows are in the second set and so forth. Then we assigne 1.75, 1.25, 0.75 and 0.25 times of the average time per window to those groups respectively. Thus, if we have a total of 180 seconds to process 12 windows, we allocate $\frac{180}{12} * 1.75 = 26.25$ seconds to each of the first three windows, $\frac{180}{12} * 1.25 = 18.75$ seconds to each of the next three windows, and so forth.

We also set a relative gap parameter so that if we obtain a solution with a duality gap less than this preset amount, the heuristic stops trying to complete the time assigned to this window and moves on either to the next window or starts LP-and-fix with the extra time if it is sufficient for that purpose. Our computational experience suggests that choosing gap values that are too small neither guarantees a better solution at the end nor allows LP-and-fix sufficient time to generate good solutions. In addition to these major parameters, there are some minor parameters used in the heuristic framework in order to increase its efficiency, such as minimum and maximum times to employ LP-and-fix. Also, note that we start employing LP-and-fix in the windows not after the first window but after the second window, because computational experience suggests that early LP-and-fix procedures take much more time to find a solution than later LP-and-fix procedures.

# 6 Computational Results

In order to provide diversified results and test the generality of our heuristic approach, we used the following various test sets from the literature for our computations:

- Test instances generated by Tempelmeier and Derstroff [1996] and Stadtler [2003]: These include overtime variables. Sets A+ and B+ involve problems with 10 items, 24 periods and 3 machines, and sets C and D involve problems with 40 items, 16 periods and 6 machines. Sets B+ and D include setup times. We chose the hardest instances of each data set for our computations, i.e., for each data set, we picked 10 assembly and 10 general instances with the highest duality gaps according to the results of Stadtler [2003].

- Multi-level LOTSIZELIB [1999] library instances: These include single-machine problems with big bucket capacities. Backlogging is allowed. The problems vary between 40 item, single end-item problems and 15 item, 3 end-item problems, with both assembly and general BOM structures. All problems have 12 periods.

- MULTI-LSB instances: We have generated 4 sets of test problems based on the problem described in the paper of Simpson and Erenguc [2005], each set having 30 instances with low, medium and high variability of demand. From now on, we will call these sets SET1, SET2, SET3, and SET4. These instances are different from the previous problems in that they take component commonality into consideration and hence consider setup variables for each family so that setup times are defined for each family of items instead of for each item. While keeping the original data such as BOM structures and holding costs, we removed the setup costs and added backlogging variables into the problem to obtain problems with a different nature from the other problems. The backlogging costs are set to the double of inventory holding costs for the first two sets, and 10 times the inventory holding costs for the last two sets. Except the problems in SET2, which considers a horizon of 24 periods, all the instances have 16 periods. The main difference between these three sets is that they have different resource utilization factors, and the rest of the data remain the same. All instances consider 78 items and have an assembly structure, and all instances allow backlogging in the last period. For more details about the instances, see Multi-LSB [2006] and Simpson and Erenguc [2005].

All the test instances were run on a PC with an Intel Pentium 4 2.53 GB processor and 1 GB of RAM. All the formulations and the heuristic algorithm were implemented using only Xpress Mosel, which is a high-level algebraic modeling language that is intuitive to use. In all the computational experiments, the Xpress-MP 2004C package and Mosel version 1.4.1 are used. We have also prepared a website web [2006] for this paper, where all the data sets and sample Mosel files can be accessed by other researchers for their testing purposes.

Different test sets will be discussed separately since we are using different benchmarks for each set, and different characteristics of each set make it more interesting to analyze each separately. We assigned a total time of 180 seconds for each instance of A+, B+, LOTSIZELIB, SET1 and SET2, and 500 seconds for each C, D, SET3 and SET4 instance because of the problem complexity. Also, a 0.5% duality gap in a window is successful for A+, B+, LOTSIZELIB, SET1 and SET2 instances, duality gaps of 2.5% for later windows and 4% for earlier windows for the C, D and SET4 instances, and finally 10% duality gap for SET3 instances. These percentages are set intuitively from the problem difficulties, e.g., if the default problem solved by Xpress augmented with $(\ell, S)$ inequalities results with a 10% gap and if there are 10 windows, 1% will be assigned to each window.

For the test sets of Tempelmeier and Derstroff [1996] and Stadtler [2003], we ran an executable of SH (Stadtler's Heuristic) on our computer to provide fair comparisons. Another benchmark used for these instances is default Xpress augmented with the generation of

Table 1: Summary of results for Tempelmeier and Derstroff [1996] and Stadtler [2003] instances

| Test Set | Best Solution found by | | | Average duality gap | | |
|---|---|---|---|---|---|---|
| | SH | Xpress | Heuristic | SH | Xpress | Heuristic |
| A+ | 4 | - | 16 | 29.28% | 25.28% | 24.47% |
| B+ | 11 | 1 | 7 | 29.27% | 34.21% | 29.86% |
| C | 11 | - | 9 | 30.69% | 35.40% | 28.33% |
| D | 7 | - | 3 | 215.96% | 364.57% | 198.54% |

$(\ell, S)$ inequalities. Inactive inequalities are deleted at the root node. The main results are summarized in Table 1: The first three columns show for how many instances of a particular test set any of these three methods found the best solution, and the last three columns show the average duality gaps of each method at each set. Note that since SH does not generate lower bounds, the $(\ell, S)$ lower bound obtained at the root node is used for that purpose.

Table 2: Pairwise comparisons of the heuristics with benchmarks for Tempelmeier and Derstroff [1996] and Stadtler [2003] instances

| Test Set | SH vs. Heuristic | | | Xpress vs. Heuristic | | | |
|---|---|---|---|---|---|---|---|
| | # S | # H | UB | # X | # H | UB | LB |
| A+ | 4 | 16 | 2.26% | - | 20 | 5.77% | 5.15% |
| B+ | 12 | 7 | -1.63% | 4 | 15 | 5.73% | 4.08% |
| C | 11 | 9 | -0.22% | - | 20 | 6.91% | 1.26% |
| D | 7 | 3 | 2.45% | - | 10 | 74.94% | 1.12% |

Table 2 summarizes pairwise comparisons between our heuristic and the two benchmarks. The first two columns show how many times SH and our heuristic provide a better solution, respectively, and the next column shows the average difference between our heuristic's and SH's upper bounds, respectively, calculated as (SH Bound - Heuristic Bound)/ Heuristic Bound. The last four columns are prepared in the same fashion, for the comparison between our heuristic and the default Xpress, and the last two columns are calculated as (Xpress Bound - Heuristic Bound)/ Heuristic Bound. It is easy to observe that our heuristic generates good solutions compared to SH, with a good lower bound guarantee. On the other hand, as expected, default Xpress generates better lower bounds than our heuristic, but it is also notable that the harder the problem is, the better the lower bounds the heuristic generates compared to those generated by Xpress. Also note that the proposed heuristic method improves the lower bounds obtained at the root node with $(\ell, S)$ inequalities on average by 1.11%. For detailed results for all the instances of these test sets, please refer to Appendix A.

Table 3: Computational Results on LOTSIZELIB [1999] Instances with time limit of 180 seconds

| Instance | Optimal Solution | Xpress | | | Heuristic | | |
|---|---|---|---|---|---|---|---|
| | | UB | LB | Time | UB | LB | Time |
| LLIB B | 3,965 | 3,965 | 3,957 | 180 | 3,973 | 3,915 | 23 |
| LLIB C* | 2,083 | 2,125 | 2,046 | 180 | 2,113 | 2,067 | 73 |
| LLIB D* | 6,482 | 6,852 | 4,723 | 180 | 7,002 | 4,714 | 180 |
| LLIB E* | 2,801 | 3,099 | 2,537 | 180 | 2,952 | 2,416 | 152 |
| LLIB F* | 2,429 | 2,458 | 2,180 | 180 | 2,429 | 2,099 | 99 |

* indicates an instance that could not be solved to optimality by default Xpress in 900 seconds.

For the LOTSIZELIB instances, on the other hand, default Xpress with $(\ell, S)$ inequalities has been the primary benchmark. The reason that we did not test SH on these problems is that it is not designed for problems with backlogging and significant modification would be necessary. Also note that these instances already have known optimal solutions, hence the solution quality is more transparent. Table 3 shows results of the LOTSIZELIB instances: The first column indicates the optimal solution, then the next three columns show respectively upper and lower bounds obtained by default Xpress and in how many seconds, and finally the last three columns present results of our heuristic. As expected, default Xpress generates better lower bounds generally, however, the heuristic provides comparably good solutions, better solutions in 3 out of 5 instances, and in less time.

Finally, we analyze the 4 sets of MULTI-LSB. These instances vary from easy to very hard problems, as can be seen in detailed results in Appendix A. Here, we summarize the results in Table 4 separately for each different set and also separately for different problem difficulties, as follows: Problems with a duality gap less than 10% after the default time with Xpress augmented with $(\ell, S)$ inequalities are called "easy", problems with a duality gap more than 50% are called "hard", and the rest are called "moderate".

Table 4 is organized as follows: For "Upper Bounds" and "Lower Bounds", "#X" and "#H" indicate how many times in that set Xpress and the proposed heuristic found a better bound, respectively. Note that cases in which both methods generate exactly the same upper bound are not included in these numbers. Finally, the last two columns indicate the average difference between the bounds in percentage, and these are calculated by the expression (Xpress Bound - Heuristic Bound)/ Heuristic Bound. As the summary indicates, the heuristic finds generally better solutions for the problem than Xpress does. Similar to previous results, Xpress default lower bounds are often better than the heuristic lower bounds, although it is interesting to observe that in SET3, i.e., the hardest test instances of all our computations, and in the hard instances of SET4, the heuristic generates lower bounds that are competitive with those of Xpress. This seems to be due to the fact that, for the hardest problems, the problems' difficulty prevents Xpress from making more than minuscule improvements to the lower bound during the branch-and-bound process.

Table 4: Summary of results for MULTI-LSB instances

| Test Set | Class | Upper Bounds | | Lower Bounds | | Ave. Difference | |
|---|---|---|---|---|---|---|---|
| | | # X | # H | # X | # H | UB | LB |
| SET1 | Easy | 1 | 10 | 11 | 1 | 0.67% | 2.24% |
| | Moderate | 0 | 18 | 5 | 13 | 7.22% | -0.11% |
| SET2 | Easy | 0 | 17 | 13 | 3 | 1.24% | 0.49% |
| | Moderate | 0 | 13 | 2 | 10 | 4.66% | -0.50% |
| SET3 | Hard | 0 | 30 | 15 | 15 | 28.48% | -0.10% |
| SET4 | Easy | 2 | 1 | 3 | 0 | -1.02% | 6.34% |
| | Moderate | 1 | 9 | 9 | 1 | 2.78% | 5.84% |
| | Hard | 0 | 17 | 4 | 13 | 9.35% | -3.88% |

*"Easy" class of SET1 and "hard" class of SET3 has each one instance where both methods found the exactly same solution, and one easy and one moderate instance of SET2 had equal lower bounds for both methods, which are not included in the numbers above.

To summarize this section, the proposed heuristic generates good solutions for different kinds of lot-sizing problems when compared to the default Xpress solver augmented with $(\ell, S)$ inequalities; Xpress generated a better solution than the heuristic only in 10 of 194 instances, and these were close to the heuristic's solutions. The comparative efficiency of the heuristic is especially noticeable for harder problems, such as set D, SET3 and SET4, where the heuristic improves default Xpress solutions significantly. On the other hand, the proposed heuristic generates results competitive with those generated by SH for the test sets A+, B+, C and D. The advantages of the proposed heuristic over SH are its flexibility in being applicable to problems with different characteristics, its ability to generate lower bounds, its ease of implementation, and the fact that it generates multiple solutions.

# 7 Conclusion

We have presented a heuristic framework that is designed for easy implementation and is flexible enough to handle a variety of production planning problems. The heuristic finds both good solutions and competitive lower bounds. Moreover, computational results indicate that the heuristic is particularly effective on the most difficult problems, where effectiveness is measured by comparison with our benchmarks.

We have also provided further evidence that using $(\ell, S)$ inequalities is a good method to strengthen the formulation of big-bucket problems. Computational experience suggests that they should be used within any default solver. Moreover, when combined with an effective heuristic framework, they can help to find good solutions as well.

Even though mathematical programming provides tools for exact solutions, extended reformulations generally result in big problems that cannot be solved efficiently. A recent

study of Van Vyve and Wolsey [2005] suggests that partial reformulations can be applied in order to have high quality lower bounds while preventing the problem size from growing too much. Similar approaches may be effective for hard lot-sizing problems as well, and this is left for future research.

While the proposed framework for finding good solutions to lot-sizing problems is comparably efficient, many of the instances discussed in the computational results remain challenging. As we discuss in detail in our companion paper Akartunalı and Miller [2007], big bucket lot-sizing problems still need a deeper analysis and better understanding of the polyhedral structure to improve lower bounds and provide better solutions; polyhedral analysis focused on their particular structure is an important future research area.

Another interesting question to be answered is whether this simple approach can be applied to other challenging MIP problems or not. We know that production planning problems have a special structure in which early decisions affect later decisions, but similar structures may be applicable to other problems as well, e.g. scheduling and facility location problems.

**Acknowledgement.** The research carried out was supported by the National Science Foundation grant No. DMI 0323299. The authors are grateful to Hartmut Stadtler and Christopher Sürie for providing the executable file of Stadtlers Heuristic (SH). Thanks are also due to anonymous referees for valuable comments that improved the presentation of the paper.

# References

`http://ms.unimelb.edu.au/~kerema/research/heuristic/`, 2006.

P. Afentakis and B. Gavish. Optimal lot-sizing algorithms for complex product structures. *Operations Research*, 34(2):237–249, 1986.

A. Aggarwal and J.K. Park. Improved algorithms for economic lot size problems. *Operations Research*, 41(3):549–571, 1993.

K. Akartunalı. *Computational Methods for Big Bucket Production Planning Problems: Feasible Solutions and Strong Formulations*. PhD thesis, Industrial Engineering, University of Wisconsin-Madison, 2007.

K. Akartunalı and A.J. Miller. A computational analysis of lower bounds for big bucket production planning problems. Available at *Optimization Online*, `http://www.optimization-online.org/DB_HTML/2007/05/1668.html`, 2007.

A. Atamtürk and J.C. Muñoz. A study of the lot-sizing polytope. *Mathematical Programming*, 98:443–465, 2004.

E. Balas, S. Ceria, M. Dawande, F. Margot, and G. Pataki. OCTANE: A new heuristic for pure 0-1 programs. *Operations Research*, 49(2):207–225, 2001.

I. Barany, T.J. Van Roy, and L.A. Wolsey. Strong formulations for multi-item capacitated lot-sizing. *Management Science*, 30(10):1255–1261, 1984a.

I. Barany, T.J. Van Roy, and L.A. Wolsey. Uncapacitated lot sizing: The convex hull of solutions. *Mathematical Programming Study*, 22:32–43, 1984b.

G. Belvaux and L.A. Wolsey. bc-prod: A specialized branch-and-cut system for lot-sizing problems. *Management Science*, 46(5):724–738, 2000.

G. Belvaux and L.A. Wolsey. Modelling practical lot-sizing problems as mixed-integer programs. *Management Science*, 47(7):993–1007, 2001.

M. Constantino. A cutting plane approach to capacitated lot-sizing with start-up costs. *Mathematical Programming*, 75:353–376, 1996.

E. Danna, E. Rothberg, and C. Le Pape. Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming*, 102:71–90, 2005.

G.D. Eppen and R.K. Martin. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Operations Research*, 35(6):832–848, 1987.

A. Federgruen and M. Tzur. A simple forward algorithm to solve general dynamic lot sizing models with $n$ periods in O($nlogn$) or O($n$) time. *Management Science*, 37(8):909–925, 1991.

A. Federgruen, J. Meissner, and M. Tzur. Progressive interval heuristics for multi-item capacitated lot sizing problem. *Operations Research*, 55(3):490–502, 2007.

M. Fischetti and A. Lodi. Local branching. *Mathematical Programming*, 98(1):23–47, 2003.

M. Fischetti, F. Glover, and A. Lodi. The feasibility pump. *Mathematical Programming*, 104 (1):91–104, 2005.

M. Florian and M. Klein. Deterministic production planning with concave costs and capacity constraints. *Management Science*, 18(1):12–20, 1971.

M. Florian, J.K. Lenstra, and H.G. Rinnooy Kan. Deterministic production planning: Algorithms and complexity. *Management Science*, 26(7):669–679, 1980.

W. J. Hoppe and M. L. Spearman. *Factory Physics*. McGraw-Hill, 2000.

E. Katok, H.S. Lewis, and T.P. Harrison. Lot sizing in general assembly systems with setup costs, setup times, and multiple constrained resources. *Management Science*, 44(6): 859–877, 1998.

J. Krarup and O. Bilde. *Plant location, set covering and economic lotsizes: an O(mn) algorithm for structured problems*, pages 155–180. Optimierung bei Graphentheoretischen und Ganzzahligen Probleme. Birkhauser Verlag, 1977.

M. Loparic, Y. Pochet, and L.A. Wolsey. The uncapacitated lot-sizing problem with sales and safety stocks. *Mathematical Programming*, 89:487–504, 2001.

LOTSIZELIB. Lot-sizing problems: A library of models and matrices. `http://www.core.ucl.ac.be/wolsey/lotsizel.htm`, 1999.

J. Maes and L. van Wassenhove. Multi item single level capacitated dynamic lotsizing heuristics: A computational comparison (part i: Static case; part ii: Rolling horizon). *IIE Transactions*, 18:114–129, 1986.

A.J. Miller, G.L. Nemhauser, and M.W.P. Savelsbergh. Solving the multi-item capacitated lot-sizing problem with setup times by branch-and-cut. CORE Discussion Paper 2000/39, *CORE, UCL, Belgium*, 2000.

A.J. Miller, G.L. Nemhauser, and M.W.P. Savelsbergh. On the polyhedral structure of a multi-item production planning model with setup times. *Mathematical Programming*, 94: 375–405, 2003.

Multi-LSB. Multi-item lot-sizing problems with backlogging: A library of test instances. Available at `http://ms.unimelb.edu.au/~kerema/research/multi-lsb/`, 2006.

Y. Pochet and L.A. Wolsey. *Production Planning by Mixed Integer Programming*. Springer, 2006.

Y. Pochet and L.A. Wolsey. Lot-size models with backlogging: Strong reformulations and cutting planes. *Mathematical Programming*, 40:317–335, 1988.

Y. Pochet and L.A. Wolsey. Solving multi-item lot-sizing problems using strong cutting planes. *Management Science*, 37(1):53–67, 1991.

Y. Pochet and L.A. Wolsey. Polyhedra for lot-sizing with Wagner-Whitin costs. *Mathematical Programming*, 67:297–323, 1994.

R.L. Rardin and L.A. Wolsey. Valid inequalities and projecting the multicommodity extended formulation for uncapacitated fixed charge network flow problems. *European Journal of Operational Research*, 71:95–109, 1993.

N.C. Simpson and S.S. Erenguc. Modeling multiple stage manufacturing systems with generalized costs and capacity issues. *Naval Research Logistics*, 52:560–570, 2005.

H. Stadtler. Multilevel lot sizing with setup times and multiple constrained resources: Internally rolling schedules with lot-sizing windows. *Operations Research*, 51:487–502, 2003.

H. Tempelmeier and M. Derstroff. A lagrangean-based heuristic for dynamic multilevel multiitem constrained lotsizing with setup times. *Management Science*, 42(5):738–757, 1996.

W.W. Trigeiro, L.J. Thomas, and J.O. McClain. Capacitated lot sizing with setup times. *Management Science*, 35:353–366, 1989.

C.P.M. van Hoesel and A.P.M. Wagelmans. An O($T^3$) algorithm for the economic lot-sizing problem with constant capacities. *Management Science*, 42(1):142–150, 1996.

M. Van Vyve and Y. Pochet. A general heuristic for production planning problems. *INFORMS Journal of Computing*, 16(3):316–327, 2004.

M. Van Vyve and L. Wolsey. Approximate extended formulations. *Mathematical Programming*, 105:501–522, 2005.

H.M. Wagner and T.M. Whitin. Dynamic version of the economic lot size model. *Management Science*, 5:89–96, 1958.

L.A. Wolsey. Solving multi-item lot-sizing problems with an MIP solver using classification and reformulation. *Management Science*, 48(12):1587–1602, 2002.

# Appendix A: Detailed Results

| Instance | SH UB | LB from $l, S$ | Xpress UB | Xpress LB | Heuristic UB | Heuristic LB |
|---|---|---|---|---|---|---|
| AG501130 | 153,418 | 116,183 | 157,433 | 133,715 | 154,515 | 119,146 |
| AG501131 | 146,500 | 107,829 | 157,450 | 116,004 | 145,225 | 109,714 |
| AG501132 | 156,822 | 118,677 | 160,678 | 127,827 | 154,191 | 121,740 |
| AG501141 | 175,619 | 133,424 | 186,362 | 140,206 | 171,895 | 134,421 |
| AG501142 | 195,110 | 145,508 | 197,582 | 154,895 | 192,582 | 148,911 |
| AG502130 | 168,707 | 122,353 | 170,059 | 146,383 | 167,927 | 128,101 |
| AG502131 | 145,322 | 109,085 | 157,623 | 115,833 | 146,361 | 111,001 |
| AG502141 | 179,371 | 134,971 | 188,933 | 141,121 | 173,640 | 136,353 |
| AG502232 | 124,015 | 97,032 | 123,304 | 103,715 | 121,108 | 97,632 |
| AG502531 | 129,080 | 102,340 | 140,294 | 108,749 | 129,640 | 103,506 |
| AK501131 | 128,095 | 96,968 | 132,020 | 102,978 | 123,366 | 99,020 |
| AK501132 | 124,499 | 101,699 | 126,919 | 106,109 | 123,473 | 103,077 |
| AK501141 | 180,695 | 134,805 | 186,662 | 139,754 | 170,897 | 136,428 |
| AK501142 | 174,701 | 134,880 | 176,504 | 138,587 | 161,262 | 135,875 |
| AK501432 | 110,851 | 92,533 | 114,088 | 99,194 | 109,249 | 93,546 |
| AK502130 | 129,307 | 102,222 | 133,429 | 110,583 | 127,889 | 103,949 |
| AK502131 | 126,394 | 93,369 | 122,273 | 98,627 | 115,819 | 94,969 |
| AK502132 | 119,175 | 96,312 | 121,712 | 100,679 | 118,319 | 97,233 |
| AK502142 | 146,616 | 127,792 | 164,357 | 131,234 | 147,729 | 129,034 |
| AK502432 | 110,094 | 88,980 | 110,896 | 94,459 | 105,415 | 89,609 |
| BG511132 | 140,155 | 108,772 | 166,747 | 115,116 | 137,637 | 110,466 |
| BG511142 | 159,769 | 133,158 | 183,318 | 138,130 | 167,262 | 133,880 |
| BG512131 | 139,839 | 104,054 | 151,552 | 112,311 | 138,752 | 105,804 |
| BG512142 | 199,051* | 142,917 | 219,790 | 150,058 | 227,996 | 143,848 |
| BG521132 | 138,133 | 108,324 | 145,975 | 115,055 | 146,709 | 110,024 |
| BG521142 | 156,694 | 131,363 | 179,427 | 136,920 | 157,802 | 132,604 |
| BG522130 | 154,581 | 113,540 | 164,913 | 131,120 | 156,075 | 121,578 |
| BG522132 | 147,894 | 113,382 | 185,786 | 120,032 | 162,389 | 115,158 |
| BG522142 | 186,268 | 137,126 | 201,118 | 142,692 | 202,851 | 138,077 |
| BK511131 | 123,699 | 92,602 | 128,861 | 97,923 | 120,303 | 94,411 |
| BK511132 | 125,658 | 95,323 | 120,227 | 99,889 | 115,416 | 95,938 |
| BK511141 | 162,629* | 125,307 | 183,541 | 129,896 | 172,762 | 126,769 |

**Note:** * indicates instances that could not be run on our computer with SH's executable, in which case their published results are used.

| Instance | SH UB | LB from $l, S$ | Xpress UB | Xpress LB | Heuristic UB | Heuristic LB |
|---|---|---|---|---|---|---|
| BK512131 | 113,996 | 90,733 | 121,357 | 94,459 | 113,536 | 92,058 |
| BK512132 | 115,697 | 90,814 | 118,918 | 95,929 | 112,809 | 91,346 |
| BK521131 | 118,217 | 92,350 | 127,362 | 97,784 | 121,292 | 94,164 |
| BK521132 | 117,423 | 94,257 | 119,690 | 99,359 | 118,464 | 94,957 |
| BK521142 | 153,805* | 124,988 | 157,186 | 128,203 | 154,258 | 125,480 |
| BK522131 | 116,340 | 90,532 | 119,591 | 95,153 | 111,339 | 91,742 |
| BK522142 | 154,286 | 119,559 | 148,471 | 122,836 | 156,557 | 119,625 |
| CG501120 | 1,252,308 | 1,011,260 | 1,383,799 | 1,037,547 | 1,296,913 | 1,027,177 |
| CG501131 | 614,303 | 472,421 | 652,793 | 480,446 | 614,971 | 478,437 |
| CG501141 | 777,831 | 627,035 | 854,196 | 631,554 | 803,432 | 628,114 |
| CG501121 | 1,261,525 | 945,696 | 1,376,403 | 964,910 | 1,247,493 | 959,756 |
| CG502221 | 889,548 | 724,648 | 1,005,102 | 730,536 | 911,616 | 728,105 |
| CG501132 | 842,734 | 561,827 | 851,427 | 619,623 | 849,500 | 606,568 |
| CG501222 | 858,289 | 697,129 | 983,752 | 702,660 | 877,364 | 699,021 |
| CG501142 | 1,146,638 | 754,238 | 1,169,704 | 847,853 | 1,150,718 | 824,887 |
| CG501122 | 1,814,877 | 1,161,383 | 1,832,565 | 1,274,366 | 1,787,833 | 1,281,687 |
| CG502222 | 873,858 | 704,096 | 1,024,136 | 709,326 | 899,970 | 708,597 |
| CK501120 | 176,187 | 141,900 | 187,551 | 145,740 | 179,099 | 143,260 |
| CK501221 | 123,206 | 101,028 | 133,710 | 101,409 | 123,066 | 101,105 |
| CK501121 | 173,083 | 131,993 | 185,519 | 134,662 | 169,804 | 132,840 |
| CK502221 | 123,657 | 101,478 | 133,305 | 102,193 | 122,596 | 101,899 |
| CK501222 | 122,485 | 97,937 | 132,502 | 98,522 | 123,298 | 98,096 |
| CK501422 | 128,558 | 101,864 | 134,032 | 104,802 | 124,315 | 102,150 |
| CK502222 | 119,965 | 98,052 | 134,820 | 98,625 | 122,302 | 98,282 |
| CK501122 | 210,824* | 153,861 | 212,018 | 162,312 | 206,646 | 155,485 |
| CK501132 | 98,363* | 75,257 | 101,602 | 78,690 | 98,248 | 75,782 |
| CK501142 | 120,722* | 90,218 | 122,122 | 92,471 | 115,918 | 90,673 |
| DG512141 | 736,181 | 609,464 | 1,841,593 | 613,816 | 759,136 | 615,992 |
| DG512131 | 581,932 | 465,272 | 2,052,160 | 496,633 | 596,395 | 469,460 |
| DG012132 | 3,625,599* | 554,595 | 3,661,662 | 559,893 | 3,160,347 | 555,689 |
| DG012142 | 3,815,898* | 756,588 | 3,730,720 | 761,229 | 3,121,762 | 756,588 |
| DG012532 | 1,194,003* | 554,167 | 1,558,743 | 558,186 | 1,228,463 | 555,032 |
| DG012542 | 1,413,475* | 756,062 | 1,652,806 | 759,010 | 1,420,613 | 756,062 |
| DG512132 | 2,909,628* | 512,330 | 5,570,581 | 518,927 | 3,010,696 | 514,682 |
| DG512142 | 3,779,026* | 678,733 | 5,655,268 | 688,424 | 3,583,354 | 682,205 |
| DG512532 | 584,491 | 509,567 | 1,031,986 | 516,101 | 597,404 | 512,147 |
| DG512542 | 767,428 | 674,241 | 1,342,964 | 683,455 | 796,831 | 677,189 |

**Note:** * indicates instances that could not be run on our computer with Stadtler's executable, in which case their published results are used.

| Instance | LB from $l, S$ | Xpress UB | Xpress LB | Heuristic UB | Heuristic LB | Improvement in UB |
|---|---|---|---|---|---|---|
| SET1_01 | 17,888 | 25,873 | 18,355 | 22,781 | 18,889 | 13.57% |
| SET1_02 | 23,534 | 30,317 | 23,934 | 28,624 | 24,134 | 5.91% |
| SET1_03 | 21,227 | 26,943 | 21,638 | 26,349 | 21,676 | 2.25% |
| SET1_04 | 22,232 | 30,236 | 22,698 | 26,337 | 23,175 | 14.80% |
| SET1_05 | 21,446 | 27,064 | 21,782 | 25,621 | 21,994 | 5.63% |
| SET1_06 | 22,974 | 29,548 | 23,187 | 26,741 | 23,636 | 10.50% |
| SET1_07 | 20,360 | 27,316 | 20,772 | 24,693 | 21,125 | 10.62% |
| SET1_08 | 25,582 | 32,361 | 25,893 | 29,810 | 26,249 | 8.56% |
| SET1_09 | 16,321 | 24,015 | 16,780 | 21,146 | 17,013 | 13.57% |
| SET1_10 | 17,998 | 23,985 | 18,243 | 22,863 | 18,945 | 4.91% |
| SET1_11 | 11,080 | 13,010 | 12,374 | 12,956 | 11,407 | 0.42% |
| SET1_12 | 24,721 | 27,377 | 25,211 | 26,985 | 25,238 | 1.45% |
| SET1_13 | 20,782 | 24,003 | 21,581 | 23,129 | 21,195 | 3.78% |
| SET1_14 | 22,264 | 25,931 | 22,388 | 25,720 | 22,745 | 0.82% |
| SET1_15 | 12,401 | 15,023 | 13,070 | 14,121 | 12,575 | 6.39% |
| SET1_16 | 15,122 | 17,542 | 16,030 | 17,559 | 15,387 | -0.10% |
| SET1_17 | 20,468 | 24,284 | 20,576 | 23,404 | 20,864 | 3.76% |
| SET1_18 | 11,075 | 13,272 | 11,858 | 12,300 | 11,456 | 7.90% |
| SET1_19 | 13,276 | 18,091 | 14,044 | 17,448 | 13,342 | 3.69% |
| SET1_20 | 14,101 | 19,433 | 14,236 | 17,167 | 14,612 | 13.20% |
| SET1_21 | 10,159 | 12,428 | 11,003 | 12,421 | 10,392 | 0.06% |
| SET1_22 | 38,040 | 40,419 | 38,556 | 40,158 | 38,040 | 0.65% |
| SET1_23 | 29,331 | 30,687 | 29,622 | 30,606 | 29,355 | 0.26% |
| SET1_24 | 28,858 | 32,282 | 30,272 | 32,174 | 29,250 | 0.34% |
| SET1_25 | 51,371 | 53,650 | 51,720 | 53,009 | 51,371 | 1.21% |
| SET1_26 | 39,379 | 42,236 | 39,979 | 41,442 | 39,488 | 1.92% |
| SET1_27 | 40,838 | 43,929 | 41,424 | 43,320 | 40,918 | 1.41% |
| SET1_28 | 39,846 | 40,993 | 40,993 | 40,993 | 40,144 | 0.00% |
| SET1_29 | 23,155 | 25,633 | 23,698 | 25,606 | 23,232 | 0.11% |
| SET1_30 | 68,989 | 71,095 | 69,899 | 70,868 | 68,989 | 0.32% |

**Note:** Only one of these 30 SET1 instances resulted in optimal solution in that allowed time. "Improvement in UB" column indicates (Xpress Upper Bound - Heuristic Upper Bound)/Heuristic Upper Bound.

| Instance | LB from $l, S$ | Xpress UB | Xpress LB | Heuristic UB | Heuristic LB | Improvement in UB |
|---|---|---|---|---|---|---|
| SET2_01 | 46,116 | 57,604 | 46,341 | 55,039 | 46,591 | 4.66% |
| SET2_02 | 47,780 | 59,307 | 47,780 | 57,825 | 48,159 | 2.56% |
| SET2_03 | 40,551 | 52,438 | 40,551 | 49,147 | 40,814 | 6.70% |
| SET2_04 | 36,347 | 47,828 | 36,347 | 44,656 | 36,808 | 7.10% |
| SET2_05 | 45,395 | 57,950 | 45,395 | 55,650 | 45,784 | 4.13% |
| SET2_06 | 45,902 | 57,791 | 45,902 | 54,361 | 45,902 | 6.31% |
| SET2_07 | 52,825 | 63,216 | 52,825 | 61,140 | 53,108 | 3.40% |
| SET2_08 | 48,033 | 60,193 | 48,198 | 56,444 | 48,632 | 6.64% |
| SET2_09 | 37,553 | 46,376 | 37,651 | 44,523 | 37,943 | 4.16% |
| SET2_10 | 38,751 | 50,334 | 38,751 | 49,481 | 39,181 | 1.72% |
| SET2_11 | 65,210 | 70,623 | 65,382 | 69,177 | 65,648 | 2.09% |
| SET2_12 | 62,792 | 68,277 | 62,918 | 66,914 | 62,792 | 2.04% |
| SET2_13 | 34,778 | 41,254 | 34,978 | 40,114 | 34,987 | 2.84% |
| SET2_14 | 62,907 | 68,070 | 63,545 | 67,201 | 62,907 | 1.29% |
| SET2_15 | 59,079 | 62,046 | 59,432 | 61,616 | 59,079 | 0.70% |
| SET2_16 | 75,682 | 81,490 | 75,695 | 79,576 | 75,682 | 2.41% |
| SET2_17 | 36,809 | 43,455 | 37,095 | 41,484 | 36,925 | 4.75% |
| SET2_18 | 77,873 | 84,288 | 77,873 | 83,200 | 78,087 | 1.31% |
| SET2_19 | 54,981 | 59,724 | 55,055 | 59,010 | 55,484 | 1.21% |
| SET2_20 | 119,568 | 124,272 | 119,568 | 122,974 | 119,568 | 1.06% |
| SET2_21 | 22,281 | 24,517 | 22,520 | 24,459 | 22,281 | 0.24% |
| SET2_22 | 51,279 | 54,090 | 51,687 | 53,690 | 51,279 | 0.74% |
| SET2_23 | 29,793 | 35,851 | 29,921 | 33,969 | 29,793 | 5.54% |
| SET2_24 | 65,891 | 69,365 | 66,277 | 68,727 | 65,891 | 0.93% |
| SET2_25 | 75,627 | 79,274 | 76,035 | 78,266 | 75,627 | 1.29% |
| SET2_26 | 60,952 | 63,998 | 61,564 | 63,558 | 60,977 | 0.69% |
| SET2_27 | 53,016 | 56,100 | 53,983 | 54,797 | 53,016 | 2.38% |
| SET2_28 | 44,545 | 46,884 | 45,293 | 46,733 | 44,549 | 0.32% |
| SET2_29 | 93,631 | 97,474 | 93,735 | 96,281 | 93,631 | 1.24% |
| SET2_30 | 68,324 | 72,783 | 68,860 | 71,919 | 68,573 | 1.20% |

**Note:** None of these 30 SET2 instances resulted in optimal solution in that allowed time.

| Instance | LB from $l, S$ | Xpress UB | Xpress LB | Heuristic UB | Heuristic LB | Improvement in UB |
|---|---|---|---|---|---|---|
| SET3_01 | 65,668 | 274,297 | 74,333 | 209,129 | 70,207 | 31.16% |
| SET3_02 | 82,342 | 337,279 | 88,125 | 243,511 | 88,681 | 38.51% |
| SET3_03 | 74,209 | 329,113 | 79,993 | 235,198 | 81,830 | 39.93% |
| SET3_04 | 78,282 | 340,701 | 86,922 | 240,339 | 85,499 | 41.76% |
| SET3_05 | 76,607 | 331,850 | 84,114 | 227,758 | 83,975 | 45.70% |
| SET3_06 | 79,093 | 295,468 | 82,420 | 235,642 | 87,790 | 25.39% |
| SET3_07 | 72,979 | 313,851 | 78,187 | 237,218 | 78,976 | 32.30% |
| SET3_08 | 88,610 | 332,507 | 93,913 | 251,628 | 96,111 | 32.14% |
| SET3_09 | 64,180 | 274,953 | 70,971 | 216,025 | 70,301 | 27.28% |
| SET3_10 | 66,878 | 285,525 | 73,164 | 229,242 | 75,040 | 24.55% |
| SET3_11 | 42,946 | 184,964 | 48,991 | 152,962 | 48,565 | 20.92% |
| SET3_12 | 86,047 | 279,693 | 97,536 | 217,497 | 94,746 | 28.60% |
| SET3_13 | 74,643 | 270,744 | 84,968 | 224,670 | 83,763 | 20.51% |
| SET3_14 | 85,209 | 296,748 | 93,522 | 225,657 | 95,835 | 31.50% |
| SET3_15 | 40,715 | 222,557 | 44,794 | 167,494 | 46,252 | 32.87% |
| SET3_16 | 46,548 | 216,952 | 53,472 | 162,616 | 50,979 | 33.41% |
| SET3_17 | 71,555 | 241,420 | 81,449 | 212,399 | 79,287 | 13.66% |
| SET3_18 | 39,533 | 206,693 | 45,876 | 112,468 | 46,879 | 83.78% |
| SET3_19 | 47,495 | 204,225 | 54,267 | 154,981 | 55,749 | 31.77% |
| SET3_20 | 58,189 | 257,030 | 64,603 | 191,639 | 64,141 | 34.12% |
| SET3_21 | 44,182 | 209,485 | 54,340 | 150,758 | 53,149 | 38.95% |
| SET3_22 | 130,235 | 367,229 | 138,697 | 292,199 | 134,729 | 25.68% |
| SET3_23 | 96,810 | 282,565 | 106,346 | 240,643 | 108,870 | 17.42% |
| SET3_24 | 105,300 | 330,999 | 114,461 | 292,996 | 113,949 | 12.97% |
| SET3_25 | 203,044 | 387,648 | 209,797 | 349,975 | 210,315 | 10.76% |
| SET3_26 | 145,184 | 349,324 | 156,009 | 323,870 | 162,088 | 7.86% |
| SET3_27 | 145,420 | 389,594 | 158,120 | 343,486 | 156,422 | 13.42% |
| SET3_28 | 145,227 | 280,123 | 154,258 | 254,008 | 155,123 | 10.28% |
| SET3_29 | 79,813 | 276,899 | 88,871 | 207,127 | 90,600 | 33.69% |
| SET3_30 | 274,018 | 489,154 | 284,240 | 431,136 | 278,826 | 13.46% |

**Note:** These 30 SET3 instances have an average duality gap of 236% in the allowed time.

| Instance | LB from $l, S$ | Xpress UB | Xpress LB | Heuristic UB | Heuristic LB | Improvement in UB |
|---|---|---|---|---|---|---|
| SET4_01 | 16,353 | 66,394 | 22,119 | 58,720 | 24,809 | 13.07% |
| SET4_02 | 31,541 | 83,818 | 39,072 | 82,496 | 41,655 | 1.60% |
| SET4_03 | 24,864 | 76,954 | 31,191 | 73,740 | 33,577 | 4.36% |
| SET4_04 | 27,786 | 78,073 | 33,786 | 73,651 | 36,553 | 6.00% |
| SET4_05 | 25,450 | 75,596 | 33,836 | 67,874 | 34,995 | 11.38% |
| SET4_06 | 30,632 | 80,132 | 38,733 | 79,781 | 40,893 | 0.44% |
| SET4_07 | 22,650 | 77,171 | 29,248 | 65,736 | 30,514 | 17.40% |
| SET4_08 | 40,532 | 100,638 | 46,045 | 88,388 | 49,106 | 13.86% |
| SET4_09 | 13,490 | 58,504 | 19,061 | 57,070 | 22,860 | 2.51% |
| SET4_10 | 15,542 | 62,611 | 22,729 | 59,319 | 25,077 | 5.55% |
| SET4_11 | 12,802 | 35,384 | 17,589 | 28,989 | 17,446 | 22.06% |
| SET4_12 | 43,341 | 84,540 | 49,504 | 78,062 | 50,961 | 8.30% |
| SET4_13 | 28,152 | 61,670 | 34,355 | 53,833 | 36,046 | 14.56% |
| SET4_14 | 56,174 | 85,443 | 64,907 | 82,406 | 64,257 | 3.69% |
| SET4_15 | 14,628 | 27,634 | 18,706 | 26,980 | 15,797 | 2.42% |
| SET4_16 | 17,171 | 36,115 | 23,792 | 35,280 | 22,606 | 2.37% |
| SET4_17 | 29,001 | 55,101 | 34,116 | 54,515 | 36,832 | 1.07% |
| SET4_18 | 19,184 | 26,279 | 24,679 | 26,596 | 22,950 | -1.19% |
| SET4_19 | 10,724 | 40,237 | 16,512 | 31,974 | 15,037 | 25.84% |
| SET4_20 | 18,718 | 43,396 | 26,400 | 39,983 | 24,035 | 8.54% |
| SET4_21 | 15,812 | 26,322 | 22,061 | 25,899 | 18,149 | 1.63% |
| SET4_22 | 91,715 | 123,711 | 96,535 | 120,166 | 93,603 | 2.95% |
| SET4_23 | 55,058 | 79,912 | 60,799 | 76,857 | 57,446 | 3.97% |
| SET4_24 | 58,919 | 86,332 | 64,856 | 85,119 | 64,305 | 1.43% |
| SET4_25 | 171,987 | 201,717 | 176,087 | 202,501 | 173,205 | -0.39% |
| SET4_26 | 110,570 | 145,290 | 117,138 | 142,090 | 117,379 | 2.25% |
| SET4_27 | 101,114 | 146,827 | 105,713 | 139,874 | 103,814 | 4.97% |
| SET4_28 | 112,892 | 126,027 | 125,250 | 129,789 | 115,176 | -2.90% |
| SET4_29 | 51,149 | 71,618 | 59,369 | 68,320 | 56,930 | 4.83% |
| SET4_30 | 241,678 | 270,746 | 248,344 | 267,976 | 241,712 | 1.03% |

**Note:** These 30 SET4 instances have an average duality gap of 79% in the allowed time.