Integrated Charge Batching and Casting Width Selection at Baosteel

Lixin Tang, Gongshu Wang
Liaoning Key Laboratory of Manufacturing System and Logistics, The Logistics Institute, Northeastern University, Shenyang, 110819, China
{lixintang@mail.neu.edu.cn, wanggongshu@ise.neu.edu.cn}
Zhi-Long Chen
Robert H. Smith School of Business, University of Maryland, College Park, Maryland 20742, zchen@rhsmith.umd.edu

We study an integrated charge batching and casting width selection problem arising in the continuous casting operation of the steelmaking process at Shanghai, China based Baosteel. This decision-making problem is not unique to Baosteel; it exists in every large iron and steel company in the world. We collaborated with Baosteel on this problem from 2006 to 2008 by developing and implementing a decision support system (DSS) that replaced their manual planning method. The DSS is still in active use at Baosteel. This paper describes the solution algorithms we developed and imbedded in the DSS. For the general problem that is strongly NP-hard, a column generation-based branch-and-price (B&P) solution approach is developed to obtain optimal solutions. By exploiting the problem structure, efficient dynamic programming algorithms are designed to solve the subproblems involved in the column generation procedure. Branching strategies are designed in a way that ensures that after every stage of branching the structure of the subproblems is preserved such that they can still be solved efficiently. We also consider a frequently occurring case of the problem where each steel grade is incompatible with any other grade. For this special case, a two-level polynomial-time algorithm is developed to obtain optimal solutions. Computational tests on a set of real production data as well as on a more diverse set of randomly generated problem instances show that our algorithms outperform the manual planning method that Baosteel used to use by a significant margin both in terms of tundish utilization for almost every case, and in terms of total cost for most cases. Consequently, by replacing their manual method with our DSS, the estimated benefits to Baosteel include an annual cost saving of about US $1.6 million and an annual revenue increase of about US $3.25 million.

Subject classifications: decision support system; continuous casting; tundish; charge batching; branch and price; column generation; dynamic programming.

Area of review: OR Practice.

History: Received March 2013; revisions received August 2013, January 2014; accepted March 2014. Published online in Articles in Advance May 9, 2014.

1. Introduction

Steel is by far the most widely used metallic material and continues to be vitally important to our society. Since the 1990s, China’s steel industry has undergone a substantial transformation and has made remarkable achievements. China’s annual steel output has exceeded 100 million tons since 1996 and has been ranked the first in the world every year over the past 15 years (Worldsteel Association 2012). Baosteel Group Corporation, headquartered in Shanghai, China, is the largest and most advanced iron and steel enterprise in China. With a total steel output of 44.27 million tons in 2011, Baosteel was ranked third among the steel enterprises in the world. Faced with an increasingly competitive market environment, Baosteel has adopted a lean management philosophy by using advanced information systems and optimization methods to enhance its core competence. We have collaborated with Baosteel on many production operations management problems since 1996 by developing and implementing a number of production planning computerized systems. This paper is motivated by our recent collaboration with Baosteel on solving a particular problem, called integrated charge batching and casting width selection problem, in the continuous casting operation of their steelmaking process. This is a commonly faced problem, not only at Baosteel, but also at every large iron and steel company in the world.

Production Process

The overall production process in any integrated steel company can roughly be divided into four stages: ironmaking, steelmaking, hot rolling, and cold rolling. In this article, we focus on the steelmaking stage, which is often regarded as a bottleneck in the whole production process. Figure 1 illustrates the steelmaking stage, consisting of melting, refining, and continuous casting steps. First, hot metal from blast furnaces is melted into crude steel through the addition of scrap and slag flux in the steelmaking furnace. A full furnace or ladle load of molten steel (about 250 to 300 tons)
Continuous casting plays a very important role in the modern steel industry. The rate of adoption of continuous casting is one of the key indicators of the comprehensive level of a nation’s steel industry. This rate has exceeded 96% in China since 2005. Unlike the traditional ingot casting, which is based on a discrete production mode, continuous casting adopts a serial batch production mode so that its production is more efficient with less energy consumption, higher productivity, and improved product quality. During continuous casting, molten steel flows from the ladle, via a tundish into water-cooled copper molds that simultaneously contain it, cool it, and move it forward. The steel strand is completely solidified at the bottom of the continuous caster and is immediately cut into slabs with required lengths. It takes about 40 to 70 minutes for a ladle of molten steel to go through the continuous caster. The tundish acts as a buffer between the ladle and the continuous caster, so that an empty ladle can be removed and a new, full one can be positioned without interruption of the continuous casting process. Once the heat-resistant material coated at the lining of the tundish is burnt out by high temperature molten steel, the tundish reaches its lifespan and has to be replaced by a new tundish. The lifespan of a tundish varies from 300 to 400 minutes depending on the coating material. Hence, at most five to eight ladles of molten steel can be cast in a given tundish before it reaches its lifespan. When a tundish has to be replaced, the caster needs to be shut down and cleaned, which incurs both a set-up time (varying from one to two hours) resulting in productivity loss and a set-up cost (about US $4,000) to repair the used tundish. Hence, it is critical for any steel company to increase the tundish utilization, represented by the average number of charges cast in each tundish, so that the productivity is improved and production cost is reduced. This can be achieved through effective production planning and scheduling.

Production Planning and Scheduling

Because of the complexity of the production process in the steelmaking stage, like most other large steel companies, Baosteel follows a hierarchical approach for its production planning and scheduling in this stage, which consists of the following three steps (shown in Figure 2). The first step is order batching, which is to consolidate the customers’ orders taken from an order pool into charges subject to technical constraints in the melting process. The second step is charge batching, which is to group and sequence charges to form casts to meet the batch production mode in the continuous casting process. The last step is production scheduling, which is to assign and schedule the batches generated in the first two steps over various production processes from melting to continuous casting. In this article, we focus on the charge batching problem in the second step, which involves both batching charges to form casts and selecting casting widths for the selected charges to maximize the tundish utilization and minimize related costs. An order corresponds to a number of slabs that satisfies specific customer requirements; a charge is a set of orders in the planning level that corresponds to a ladle of molten steel in the production process; and a cast is a set of charges in the planning level that is represented by multiple ladles of molten steel consecutively processed in a tundish. We also note that during production, no strand interruption is allowed in the continuous casting process such that all the ladles of molten steel for a cast must be available without interruption. This is a constraint that is considered when making scheduling decisions and is strictly enforced in practice by all necessary means such as slowing down the casting of a ladle to wait for the next ladle to arrive. The charge batching problem we consider does not deal with these issues.

There are two steelmaking shops (1, 2) at Baosteel where there are three and two continuous casters, respectively. They are managed separately in the production planning and scheduling level. Baosteel makes order batching and charge batching decisions once per weekday and makes daily scheduling decisions in a real-time fashion for each steelmaking shop separately. From Tuesday through Friday, each day’s order batching and charge batching decisions are made on the afternoon of the previous day. The decisions for Saturday, Sunday, and Monday are all made on Friday.
When grouping and sequencing charges to form a cast, a specific steel grade, and each charge consists of several orders with an identical grade. Because casting is a continuous process, a change in steel grade from one charge to the next results in mixing molten steel with different steel grades in a tundish. In such a situation, at least two slabs containing hybrid grades will be produced. If the metallurgical compositions of the two grades are sufficiently close, the resulting hybrid-grade slabs can still be used to fill an order; otherwise, the hybrid-grade slabs could be regarded as waste. In the earlier case, the two steel grades are said to be compatible. Because of high production cost per slab (about US $8,000), it is required that the steel grades of every two adjacent charges within a tundish must be compatible. Within a tundish, whenever a charge with one grade is followed by another charge with a different but compatible grade, there is a grade switch cost estimated as the profit loss caused by using a hybrid-grade slab to fill an order. Compatibility relationships of steel grades and grade switch costs are carefully configured by Baosteel’s expert planners from both engineering and market viewpoints. The grade switch cost usually varies from US $100 to US $400, depending on dissimilarity of their metallurgical compositions.

- The width of the nozzle located at the bottom of a tundish, which determines the casting width of the slabs in the charges currently in the tundish, is adjustable to one of about a dozen widths (e.g., 900 mm, 950 mm, ..., 1,450 mm). However, operational policies impose three restrictions on the nozzle adjustment to avoid possible risks: (i) nozzle adjustment can only occur in the end of a charge because each charge can only have one casting width; (ii) the nozzle width can only be adjusted at most once during the casting process; (iii) if the nozzle width is adjusted, it can only go from the current width to its adjacent narrower width (e.g., from 1,000 mm to 950 mm) and doing so incurs a width switch cost because of the associated profit loss. Frequent adjustment to the nozzle may cause molten steel leakage. Also, adjusting nozzle width too much or adjusting nozzle width from narrow to wide is not allowed because doing so could cause serious quality problems including crusting the slab surface.

- Since hot strip machines in the hot rolling stage are capable of squeezing a slab into a coil within a range of different widths, the width of the slabs in a charge (which is determined by the casting width for the charge) does not have to be exactly the same as the required width of the coils specified by the orders. As a result, each charge is associated with multiple possible casting widths (usually two to five possible widths). For each cast, as discussed earlier, at most two possible casting widths can be selected. The casting width(s) selected for the cast must be feasible for all the charges in the cast.
Finally, as discussed earlier, each tundish has a limited lifespan. Consequently, the total casting time of the charges that are processed within a tundish must not exceed the tundish lifespan.

In summary, the problem we study in this paper is to batch charges into casts, one for each of a given number of tundishes, and selecting one or two casting widths for each cast such that tundish utilization is maximized and the total grade switch and width switch cost of these casts is minimized. Since in this problem charge batching and casting width selection decisions are considered jointly, we call this problem integrated charge batching and casting width selection problem. As discussed above, this is a very important problem commonly faced in the steel industry. However, this problem has received little attention in the academic literature (reviewed in §2). No existing papers have considered charge batching and casting width selection decisions in an integrated manner. Furthermore, all existing papers use heuristic solution approaches.

Baosteel used to rely on a manual method based on a greedy heuristic to make charge batching and casting width selection decisions. Their manual method is described in §5. We collaborated with Baosteel on this problem from 2006 to 2008 with the main goal of replacing their manual method by an optimization-based decision support system (DSS) and improving their tundish utilization and reducing related costs. We considered the charge batching and casting width selection decisions jointly as one integrated problem, as described above. We made the following technical and practical contributions. We showed that the problem is strongly NP-hard. We exploited the problem structure and developed some optimality properties, which enabled us to develop a branch-and-price (B&P) solution approach to obtain optimal solutions for the integrated problem. The linear programming (LP) relaxation of each B&P node was solved by a column generation approach where subproblems were solved by using efficient dynamic programming algorithms. In addition, valid inequalities were developed to make the LP relaxation tighter. Our algorithm is capable of generating optimal solutions for the daily problems faced by each of the two steelmaking shops at Baosteel within an acceptable computation time. In addition, for a frequently occurring special case of the problem where each steel grade is incompatible with any other grade, we developed a two-level polynomial-time algorithm to solve it to optimality. Using test instances from their real production data, as well as randomly generated test instances, it is shown that our algorithms outperform their manual method by a significant margin both in terms of tundish utilization for almost every test instance and in terms of total grade and width switch cost for most test instances. Consequently, by replacing their manual method with our DSS, the estimated benefits to Baosteel include an annual cost saving of about US $1.6 million and an annual revenue increase of about US $3.25 million.

The rest of the paper is organized as follows. In §2, we briefly review related literature. Section 3 defines the problem and discusses its complexity. The solution algorithms are presented in §4. Section 5 reports computational tests, system implementation, and financial benefits to Baosteel. Finally, conclusions are drawn in §6.

2. Literature Review

The steel industry often uses batch production mode in various production processes to improve productivity and facility utilization and reduce energy consumption. Consequently, steel companies are often faced with many batching problems in their production planning and scheduling. Batching problems in the steel industry have been extensively studied in the literature since the early 1980s. Batching in the steel industry can generally be classified into two categories: parallel batching and serial batching. In parallel batching, items in the same batch are processed together, whereas in serial batching, items in the same batch are sequentially processed one by one.

Parallel batching problems are mainly concerned with grouping decisions and consider capacity and dissimilarities between items as main constraints. Examples of parallel batching problems in the steel production include order batching in the melting process, assignment of orders to slabs in the end of the steelmaking stage, slab batching in the reheating process, and coil batching in the batch annealing process. Representative recent papers on parallel batching include Kalagnanam et al. (2000), Naphaoe et al. (2001), Balakrishnan and Geunes (2003), Dawande et al. (2004), and Tang et al. (2011). Since our paper is on a serial batching problem, we do not review these papers in detail.

Serial batching problems are concerned with both grouping and sequencing decisions and consider capacity and changeover issues between adjacent items as main constraints. Serial batching problems are generally more complex than parallel batching problems because sequencing of items within a batch is also a decision. Examples of serial batching problems include charge batching in the continuous casting process, slab batching in the hot rolling process, and coil batching in finishing processes (e.g., color coating). Serial batching problems studied in most existing papers arise from the hot rolling and cold rolling stages. The problem we consider is the only serial batching problem in the steelmaking stage. Few existing papers have considered this problem. Box and Herbe (1988) discuss various practical factors related to charge batching at LTV Cleveland Steel Works and demonstrate the benefits of using a computer system they developed. However, no mathematical model or solution approach is given in their paper. Chang et al. (2000) consider a substantially simplified version of our problem. In their problem, casting width for each charge is given and not a decision, and there is no grade switch cost, and hence sequencing of charges within a batch is not a decision. They formulate their problem as a mixed integer program and solve it using a heuristic that rounds the fractional solution generated by a column generation...
approach for the LP relaxation to an integer solution. Tang and Wang (2008) consider an order batching problem and a charge batching problem. Their charge batching problem is also a simplified version of our problem where casting width for each charge is given and not a decision. They develop a tabu search heuristic for their charge batching problem. To the best of our knowledge, our study is the first attempt to consider charge batching and casting width selection jointly in an integrated manner, and our branch-and-price algorithm is the first exact solution algorithm for this problem.

Representative papers dealing with serial batching problems in hot rolling and cold rolling stages include Lopez et al. (1998), Cowling (2003), Tang et al. (2008), and Höhn et al. (2011). There are significant differences between our problem and theirs. The casting width selection decision, which is a part of our problem, does not exist in their problems. Also, batching constraints involved are very different because of different process requirements in different stages of the steel production.

In addition to batching problems, there are various scheduling problems that arise in many steel production processes. Scheduling problems are very different from batching problems in structure and are mainly concerned with the allocation, sequencing, and timing of jobs on the corresponding machines. Representative recent scheduling papers include Harjunkoski and Grossmann (2001), Pacciarelli and Pranzo (2004), Missbauer et al. (2009), Frasch et al. (2011), and Höhn et al. (2012).

Our problem is somewhat similar to the multiple knapsack problem (Forrest et al. 2006) if each tundish is viewed as a knapsack and each charge is viewed as an item. However, our problem is much more complex because in our problem there are sequence dependent constraints, as well as casting width selection decisions, in addition to batching decisions. Without the casting width selection decisions, our problem also has a somewhat similar structure as the capacitated team orienteering and profitable tour problem (Archetti et al. 2009). However, the presence of casting width selection decisions in our problem makes our problem much more complex. In our problem, there are at least 50 charges involved, and for each charge, exactly one of the two to five allowable casting widths for this charge can be selected. This makes the solution space for the casting width selection decisions alone extremely large. Furthermore, in our problem a large number of charges have the same or very similar steel grades and allowable casting widths with some other charges, which leads to a high level of symmetry in our problem (e.g., any two charges with the same parameters can be interchanged in a solution). All this makes our problem extremely difficult to solve.

3. Problem Definition and Complexity

Let \( N = \{1, 2, \ldots, n\} \) be a given set of \( n \) charges, and \( m \) be the number of available tundishes. Each charge \( i \in N \) is associated with a steel grade \( g_i \). Only the charges with the same steel grade or compatible steel grades can be consecutively cast within a tundish. We call this restriction the grade switch constraint. For the given set of charges \( N \), we let \( G = \{1, 2, \ldots, g\} \) be the set of steel grades involved. For each \( k \in G \), we denote \( N_k = \{i \in N \mid g_i = k\} \) as the set of charges with steel grade \( k \), \( n_k = |N_k| \) as the number of charges with steel grade \( k \), and \( G_k \subseteq G \) as the set of steel grades that are compatible with grade \( k \). For each pair of grade \( (k, j) \), \( k \in G, j \in G_k \), if a charge with grade \( j \) is cast immediately after a charge with grade \( k \), it occurs a grade switch cost \( \alpha_{ij} \) which represents the profit loss due to producing slabs of hybrid grades \( k \) and \( j \) at the boundary of the two charges. When the two adjacent charges have the same grade, then there is no grade switch cost, i.e., \( \alpha_{ij} = 0 \) for \( k \in G \). We note that the compatibility of each pair of grades is independent of that of other pairs. For example, it is possible that grade \( i \) is compatible with grade \( j \), which is compatible with grade \( k \), but grade \( i \) is not compatible with grade \( k \). Moreover, the grade switch costs of compatible pairs may not satisfy the triangle inequality, i.e., if any two of the three grades \( i, j, k \) are compatible, \( \alpha_{ij} + \alpha_{jk} \) is not necessarily greater than \( \alpha_{ik} \).

Each charge is for a set of slabs to be produced for a given set of customer orders. The corresponding orders of a charge collectively specify the width of the final product (coils) to be produced. The allowed width of the slabs for each charge \( i \in N \) can vary within a known interval \([w_{i1}, w_{i2}]\). The length of the width interval, i.e., \( w_{i2} - w_{i1} \), is determined by the maximum allowance for width squeeze at the hot rolling machine, which depends on the steel grade. Thus, the lengths of width intervals for different charges with the same steel grade are identical, i.e., \( w_{i1} - w_{i2} = w_{j1} - w_{j2} \) if \( g_i = g_j \).

The width of the nozzle located at the bottom of the caster determines the width of the slabs. According to the machine parameters and operational policies of the caster, the nozzle may be set to a finite number of equally spaced widths, denoted as \( W = \{w_1, w_2, \ldots, w_r\} \), where \( r \) is the number of widths that can be set, \( w_1 < w_2 < \cdots < w_r \), and \( w_r - w_1 = w_2 - w_1 = \cdots = w_r - w_{r-1} = \Delta \) for some \( \Delta > 0 \).

Thus, the feasible casting widths for a given charge \( i \in N \) are the subset of \( W \) associated with the nozzle that falls within the width interval of the slabs for this charge \([w_{i1}, w_{i2}]\). For notational convenience, we use index \( j \) to represent width \( w_j \) and hence the set of the allowable widths of the nozzle \( W \) can be represented as \( R = \{1, 2, \ldots, r\} \). The set of allowable casting widths for charge \( i \in N \) is denoted as \( R_i = \{l_1, l_1 + 1, \ldots, u_i\} \) where the minimum allowable width \( l_i = \min\{j \in R \mid w_j \geq w_{i1}\} \) and the maximum allowable width \( u_i = \max\{j \in R \mid w_j \leq w_{i2}\} \). For any two charges \( i \) and \( j \) with the same steel grade, because the lengths of their width intervals are equal, i.e., \( w_{i1} - w_{i2} = w_{j1} - w_{j2} \), we have \( -1 \leq |R_i| - |R_j| \leq 1 \). For each charge \( i \in N \), we need to choose a casting width to use from \( R_i \). We call this decision the casting width selection decision.

Because of technical limitations and operational policies, we can only change the casting width of the nozzle at most
once during the casting process, and if the width change does occur, it can only be changed from the current width, say \( w_j \), to its adjacent smaller width, \( w_{j-1} \). We call this restriction the width switch constraint. When the change of the casting width occurs in a tundish, a trapeziform slab with top width \( w_j \) and bottom width \( w_{j-1} \) is produced, which results in profit loss because the bevel edges of the trapeziform slab have to be cut off. Because \( w_{j-1} - w_j = w_j - w_1 = \cdots = w_j - w_{j-1} = \Delta \), whenever a width switch occurs, the change in width (i.e., \( w_j - w_{j-1} = \Delta \)) is always the same regardless of what the current width \( w_j \) is. This means that the area of the trapeziform slab that has to be cut off is always the same, which is \( \Delta \) times the length of the trapeziform slab. Thus, we can assume that the profit loss due to a width switch is independent of the specific widths involved, and let \( \beta \) denote the width switch cost due to this profit loss.

The casting time of a charge depends on several factors but is differentiated by the casting width of the charge only. The casting time decreases with the casting width. Let \( t_j \) be the casting time of a charge with casting width \( j \in R \). Then \( t_1 > t_2 > \cdots > t_j \). Let \( T \) denote the lifespans of a tundish. The total casting time of the charges that are consecutively cast within a tundish must not exceed the tundish lifespan \( T \). This requirement is called the tundish lifespan constraint.

A feasible cast is a sequence of charges satisfying the above-described grade switch constraint, width switch constraint, and tundish lifespan constraint. Based on the width switch constraint, all feasible casts can be partitioned into one-width casts where a single casting width is used for all the charges, and two-width casts where there is a width switch. We denote a feasible one-width cast as \( \omega = (j: i_1, i_2, \ldots, i_q) \) in which \( q \) charges \( i_1, i_2, \ldots, i_q \in N \) are consecutively cast in this sequence in a tundish with the casting width \( j \), which has to satisfy \( l_h \leq j \leq u_h, \forall h \in \{1, 2, \ldots, q\} \). Clearly, the tundish lifespan constraint implies that the number of charges covered in a one-width cast with width \( j \) has to be no greater than \( \delta_j = \lceil T/t_j \rceil \), where notation \( \lceil z \rceil \) is the integer largest less than or equal to \( z \). Similarly, we denote a feasible two-width cast as \( \omega = (j + 1: i_1, i_2, \ldots, i_q; j: i_{p+1}, i_{p+2}, \ldots, i_q) \) in which \( q \) charges, \( i_1, i_2, \ldots, i_q \in N \), are consecutively cast in this sequence in a tundish, a casting width \( j + 1 \) satisfying \( l_h \leq j + 1 \leq u_h, \forall h \in \{1, 2, \ldots, p\} \), is used for the first \( p \) charges, and a narrower casting width \( j \) satisfying \( l_h \leq j \leq u_h, \forall h \in \{p + 1, p + 2, \ldots, q\} \), is used for the other \( q - p \) charges. Clearly, a two-width cast has to satisfy \( p t_{j+1} + (q - p) t_j \leq T \) following the tundish lifespan constraint. We use \( c_{\omega} \) to denote the cost of a feasible cast \( \omega \) including total grade and width switch cost. The cost of a one-width cast \( \omega = (j: i_1, i_2, \ldots, i_q) \) is \( c_{\omega} = \sum_{h=1}^{q-1} \alpha_{i_h,i_{h+1}} \) and that of a two-width cast \( \omega = (j + 1: i_1, i_2, \ldots, i_q; j: i_{p+1}, i_{p+2}, \ldots, i_q) \) is \( c_{\omega} = \sum_{h=1}^{q-1} \alpha_{i_h,i_{h+1}} + \beta \).

The problem is to find \( m \) feasible casts, one for each given tundish, such that the total number of charges packed into the \( m \) casts is maximized and the total grade and width switch cost of these casts is minimized. Obviously, the two objectives are conflicting. However, in practice, the first objective, measuring the tundish utilization, is the foremost important because a tundish is much more expensive compared to a grade switch or width switch cost. It can cost about US $4,000 to coat a tundish refractory lining, whereas the grade or width switch cost is about US $50 to $400. Thus, we maximize the number of charges packed first, and after that we minimize the total grade and width switch cost. Mathematically, this can be done by combining these two objectives into a single one as follows. We assign a sufficiently large reward \( \rho \) (e.g., US $10,000) to each charge packed, and define the net reward of a cast as \( C_{\omega} = q\rho - c_{\omega} \), where \( q \) is number of charges covered in this cast. Then the single objective of the problem is to maximize the total net reward of these \( m \) casts.

The problem can be proved to be strongly NP-hard. The NP-hardness proof is given in Online Appendix 1 (available as supplemental material at http://dx.doi.org/10.1287/ opre.2014.1278). The problem can be formulated as a mixed integer program (MIP), which is given in Online Appendix 2. Similarities of many charges in their characteristics (as discussed in the end of §2) lead to a high level of symmetry in the MIP formulation. In addition, a majority of the constraints in the MIP formulation are conditional constraints used to represent the logical relationship of the finishing times between each pair of adjacent charges within a cast. Consequently, the LP relaxation of the MIP formulation is very weak and a direct commercial MIP solver such as IBM ILOG CPLEX would fail to solve it to optimality within several hours. Online Appendix 2 provides some computational results.

4. Solution Algorithms

We first present in §4.1 some optimality properties of the integrated charge batching and casting width selection problem. In §4.2, we develop a column generation based branch-and-price exact solution algorithm for the general problem. In §4.3, we consider a frequently occurring special case of the problem where the steel grades of the charges are all incomparable. An optimal polynomial-time algorithm for solving this case of the problem is proposed.

4.1. Optimality Properties

**Lemma 1.** There exists an optimal solution that satisfies the following:

(i) In any one-width cast \( \omega = (j: i_1, i_2, \ldots, i_q) \) contained in this solution, the casting width \( j \) used is equal to \( \min \{u_i | h = 1, 2, \ldots, q\} \).

(ii) In any two-width cast \( \omega = (j + 1: i_1, i_2, \ldots, i_q; j: i_{p+1}, i_{p+2}, \ldots, i_q) \) contained in this solution, the narrower casting width used \( j \) is equal to \( \min \{u_i | h = p + 1, p + 2, \ldots, q\} \).

**Proof:** See Online Appendix 3.

Let \( U = \{u_i | i \in N\} \) be the set of maximum allowable casting widths of the charges, and \( u_{\max} = \max \{u_i | i \in N\} \) be the maximum width in \( U \). Lemma 1 means that it is always sufficient to consider feasible one-width casts \( \omega = (j: i_1, i_2, \ldots, i_q) \) with a casting width \( j \in U \), and feasible
two-width casts \(\omega = (j+1; i_1, i_2, \ldots, i_p; j; i_{p+1}, i_{p+2}, \ldots, i_q)\), with a narrower casting width \(j \in U \setminus \{u_{\text{max}}\}\). Since \(U \subseteq R\), using this result reduces the solution space of the problem. Therefore, without loss of generality, we assume that any feasible solution we consider in the remainder of this paper satisfies the properties in this lemma.

4.2. An Exact Algorithm for the General Problem

Column generation, imbedded in the branch-and-bound (B&B) framework, is often referred as branch-and-price (B&P). In the literature, exact B&P solution algorithms have been developed to solve successfully practical instances of many combinatorial optimization problems, including vehicle routing problems (Desrochers et al. 1992, Desaulniers 2010), machine scheduling problems (Chen and Powell 1999, Van den Akker et al. 2000), airline crew scheduling problems (Vance et al. 1997, Sandhu and Klabjan 2007), and cutting and packing problems (Valério de Carvalho (2005), Ben Amor et al. 2006, Ceselli and Righini 2008).

In this section, we propose an exact B&P solution algorithm for the general problem. In §4.2.1, the problem is formulated as a set packing type of problem and a column generation procedure is proposed to solve the corresponding linear relaxation problem. In §4.2.2, solution algorithms for solving the pricing problem resulted from the column generation approach are presented. In §4.2.3, branching strategies are given.

4.2.1. Column Generation Approach. Clearly, to find an optimal solution for the general problem, we only need to consider casts satisfying properties (i) and (ii) of Lemma 1. We define \(\Omega\) to be the set of all such casts. For each cast \(\omega \in \Omega\), we define \(a_{\omega i}\) to be 1 if charge \(i\) is covered in this cast, and 0 otherwise. We define a binary decision variable \(\lambda_{\omega i}\) for each \(\omega \in \Omega\) to be 1 if this cast is adopted in the optimal solution and 0 otherwise. The problem can be formulated as the following set packing type of problem:

\[
\begin{align*}
\text{(SP)} & \quad \text{Maximize} & & \sum_{\omega \in \Omega} C_{\omega} \lambda_{\omega} \quad (1) \\
& \text{Subject to} & & \sum_{\omega \in \Omega} a_{\omega i} \lambda_{\omega} \leq 1 \quad i \in N, \quad (2) \\
& & & \sum_{\omega \in \Omega} \lambda_{\omega} = m, \quad (3) \\
& & & \lambda_{\omega} \in \{0, 1\} \quad \omega \in \Omega. \quad (4)
\end{align*}
\]

In this formulation, the objective (1) is to maximize the total net reward of the selected casts. Constraint (2) ensures that each charge is covered at most once. Constraint (3) guarantees that exactly \(m\) casts are selected, one for each tundish. We note that the complications involved in the problem (including decisions on casting width, and constraints on casting width switch, grade switch, and tundish lifespan) are not explicitly shown in (SP). Instead, all such complications are handled in a pricing subproblem, which is described later.

We propose to solve (SP) using a column generation based B&P exact solution approach. Because the effectiveness of this method depends largely on the tightness of the LP relaxation, we exploit the problem structure to strengthen this formulation.

For any two charges \(i, j \in N\) with the same grade (i.e., \(g_i = g_j\) and satisfying one of the following relations: (1) \(l_i = l_j\) and \(u_i > u_j\); (2) \(u_i = u_j\) and \(l_i < l_j\); (3) \(l_i = l_j\) and \(u_i = u_j\) and \(i < j\), we say that charge \(i\) dominates charge \(j\).

Define \(D_i\) to be the set of charges that are dominated by charge \(i\). Then, we can claim that there exists an optimal solution such that if any charge in the set \(D_i\) is covered in this solution, then charge \(i\) must also be covered in this solution. This claim is true because otherwise we can use charge \(i\) to replace any charge that is covered in the solution and dominated by \(i\) without affecting the optimality of the solution. By this claim, we can add the following valid inequality to the formulation (SP):

\[
\sum_{\omega \in \Omega} a_{\omega i} \lambda_{\omega} - \sum_{\omega \in \Omega} a_{\omega j} \lambda_{\omega} \leq 0, \quad i \in N, \quad j \in D_i.
\]

Since the formulation (SP) usually contains an exponential number of cast selection variables, it is impractical to solve (SP) directly. If the integrality requirement on \(\lambda\)-variables is relaxed, i.e., setting \(\lambda_{\omega i} \in [0, 1]\), for \(\omega \in \Omega\), we obtain the LP relaxation of the (SP), denoted as (LSP). A lower bound for (SP) is generated by solving (LSP) using column generation.

The column generation procedure for (LSP) starts with a limited number of columns in (LSP) and generates necessary columns iteratively. In each iteration, we first solve a restricted master problem (which is a restricted version of (LSP) with the columns generated so far). Then we generate columns with most positive reduced costs by solving a pricing problem. Newly generated columns are then added to the restricted master problem, which is then updated. When no new columns with a positive reduced cost can be found in solving the pricing problem, the procedure terminates and (LSP) is solved to optimality.

In our B&P algorithm, every time after we solve the LP relaxation of a B&B node, we try to tighten this LP relaxation formulation by possibly adding a valid inequality as follows. Given the LP relaxation solution, \(\lambda_{\omega i}\) for \(\omega \in \Omega\), where \(\Omega\) is the set of columns in the LP relaxation problem, if the total number of charges covered in the solution, defined as \(\eta = \sum_{\omega \in \Omega} \sum_{i \in N} a_{\omega i} \lambda_{\omega i}\), is fractional, then we add the following valid inequality to the LP relaxation problem and re-solve the problem using the column generation approach:

\[
\sum_{\omega \in \Omega} \sum_{i \in N} a_{\omega i} \lambda_{\omega i} \leq \lceil \eta \rceil.
\]

In (LSP), let dual variable value be \(\pi_i\) for index \(i\) in (2), \(\sigma\) for (3), \(\tau_{ij}\) for index \((i, j)\) in (5), and \(\theta\) for (6). The reduced cost of column \(\omega \in \Omega\), denoted as \(\gamma_{\omega i}\), is given by

\[
\gamma_{\omega i} = C_{\omega} - \sum_{i \in N} \pi_i a_{\omega i} - \sum_{i \in N} \sum_{j \in D_i} (a_{\omega i} - a_{\omega j}) \tau_{ij} - \sigma \sum_{i \in N} a_{\omega i} = \sum_{i \in N} a_{\omega i} \left( \rho - \pi_i + \sum_{j \in D_i} \tau_{ij} - \sum_{j \in B_i} \tau_{ji} - \theta \right) - c_{\omega} - \sigma = \sum_{i \in N} a_{\omega i} \rho_i - c_{\omega} - \sigma = \sum_{i \in \Omega} \rho_i - c_{\omega} - \sigma,
\]
where for each \( i \in N \), \( B_i = \{ j \in N \mid i \in D_j \} \) is the set of charges that dominate charge \( i \), and \( \rho_i = \rho - \tau_i + \sum_{j \in B_i} \tau_j - \sum_{j \in B_i} \theta_j \). The pricing problem is to find a cast \( \omega \in \Omega \) with maximum \( \gamma_\omega \) defined by (6). The constant \( \sigma \) in (6) can be ignored. We redefine the reward for packing a charge \( i \) as \( \rho'_i \) (which can be negative). Thus, the pricing problem is equivalent to finding a cast \( \omega \in \Omega \) with the maximum total net redefined reward, \( \sum_{i \in \omega} \rho'_i - \epsilon_\omega \).

4.2.2. Solving the Pricing Problem. By Lemma 1, we only need to consider one-width casts with a casting width belonging to \( U \) and two-width casts with the pair of casting widths belonging to \( \{(j + 1, j) \mid j \in U \setminus \{u_{\max}\} \} \). Thus, the pricing problem can be decomposed into two subproblems, one for finding a one-width cast with the maximum total net redefined reward, and the other for finding a two-width cast with the maximum total net redefined reward.

We first define some notation to be used in the algorithms for solving the subproblems:

\[
\Gamma_j = \{ i \in N \mid l_i \leq j \leq u_i \}, \quad j \in R, \text{ is the set of charges for which } j \text{ is an allowable casting width;}
\]

\[
P_{kj} = \{ i \in \Gamma_j \mid g_i = k \}, \quad k \in G \text{ and } j \in R, \text{ is the subset of charges of } \Gamma_j \text{ with steel grade } k;
\]

\[
n_{kj} = |P_{kj}|, \quad k \in G \text{ and } j \in R, \text{ is the number of charges in } P_{kj}.
\]

For ease of presentation, we reindex the charges in \( P_{kj} \) as \( l_{kj1}, l_{kj2}, \ldots, l_{kjn_{kj}} \), which are in the nondecreasing order of the redefined reward \( \rho'_i \), i.e., \( \rho'_{l_{kj1}} \geq \rho'_{l_{kj2}} \geq \cdots \geq \rho'_{l_{kjn_{kj}}} \).

Clearly, each one-width cast with width \( j \) covers a subset of charges in \( \Gamma_j \), and each two-width cast with a pair of widths \( j + 1 \) and \( j \) contains two segments where the first segment covers a subset of charges in \( \Gamma_{j+1} \) and the second segment covers a subset of charges in \( \Gamma_j \). Note that, in our B&P algorithm, branching procedures may update the sets \( \Gamma_j, P_{kj}, n_{kj} \).

In the following, we propose two algorithms—Algorithm 1 for solving the first subproblem, which is to find a one-width cast with a particular width \( j \in U \); and Algorithm 2 for solving the second subproblem, which is to find a two-width cast with a particular pair of widths \( j + 1 \) and \( j \), for \( j \in U \setminus \{u_{\max}\} \).

Algorithm 1 (For first subproblem). Define \( \varphi(k; x_1, x_2, \ldots, x_g) \) to be the maximum total net redefined reward of a one-width cast with width \( j \) containing a subset of charges of \( \Gamma_j \), where (i) \( k \) is the steel grade of the last charge in the cast, (ii) \( x_{h} \), for \( h \in \{1, 2, \ldots, g\} \), is the number of charges with grade \( h \) covered in the cast.

**Initial conditions:** \( \varphi(0; 0, 0, \ldots, 0) = 0 \). Let \( \alpha_{h0} = 0 \), for \( h \in \{1, 2, \ldots, g\} \).

**Recurseve relations:** For \( k = 1, 2, \ldots, g \), \( x_k = 0, 1, \ldots, n_{kj} \) for each \( k = 1, 2, \ldots, g \), such that \( 1 \leq \sum_{k=1}^{g} x_k \leq \delta_j \).

\[
\varphi(k; x_1, x_2, \ldots, x_g) = \max \{ \varphi(h; x_1, x_2, \ldots, x_{g-1}) + \rho'_{l_{kj}} - \alpha_{h}, \quad h \in G_k \}.
\]

The optimal solution is obtained by computing

\[
\max \left\{ \varphi(k; x_1, x_2, \ldots, x_g) \mid k = 1, 2, \ldots, g; \right\}
\]

\[
x_k = 0, 1, \ldots, n_{kj}, \quad \text{for } k = 1, 2, \ldots, g; \quad \sum_{k=1}^{g} x_k \leq \delta_j \right\}.
\]

In the above DP, the recursive relations consider all possible steel grades of the last charge. By the way the indices of the charges in \( P_{kj} \) are defined, in a cast corresponding to \( \varphi(k; x_1, x_2, \ldots, x_g) \), the covered charges for each grade \( h \) must be the first \( x_h \) ones in \( P_{kj} \). Thus the \( x_h \)th charge for grade \( k \) used in the recursive relations must be \( l_{kjx} \). The time complexity of the DP is bounded by \( O(n^5/\delta^{g+2}) \). This is because there are a total of \( O(g \prod_{h=1}^{g} n_{hj}) \) states in the DP, and the time for calculating each state is bounded by \( O(g) \).

Algorithm 2 (For second subproblem). We now consider the second subproblem, which is to find a sequence of charges containing two segments such that (i) the first segment uses width \( j + 1 \) and consists of a subset of charges in \( \Gamma_{j+1} \); (ii) the second segment uses width \( j \) and consists of a subset of charges in \( \Gamma_j \); (iii) there is no overlap between the two segments, i.e., a charge in \( \Gamma_{j+1} \cap \Gamma_j \) cannot be covered by both segments simultaneously; (iv) grade switch and tundish lifespan constraints are satisfied; (v) the total net redefined reward of the cast is maximized.

The second subproblem is more complicated than the first one because of the no-overlap requirement (iii) described above. To overcome this complication, we relax this requirement by allowing a charge \( i \in \Gamma_{j+1} \cap \Gamma_j \) to appear in both segments of a cast. With this relaxation, we first find two separate one-width casts independently, \( \omega_1 = \{ (j + 1; i_1, i_2, \ldots, i_p) \} \) and \( \omega_2 = \{ (j; h_1, h_2, \ldots, h_q) \} \), by applying Algorithm 1. Then we combine \( \omega_1 \) and \( \omega_2 \) to form a two-width cast \( \omega = \{ (j + 1; i_1, i_2, \ldots, i_p; j; h_1, h_2, \ldots, h_q) \} \), where for ease of computation, the sequence of the charges from \( \omega_2 \) is reversed. Some charges from \( \Gamma_{j+1} \cap \Gamma_j \) may appear twice in the resulting \( \omega \). In this process, we need to ensure that the total casting time of \( \omega_1 \) and \( \omega_2 \) does not exceed the tundish lifespan. The total net redefined reward of \( \omega \) is equal to the sum of the total net redefined reward of \( \omega_1 \) and \( \omega_2 \) minus a width switch cost \( \beta \) and a possible grade switch cost from the last charge in \( \omega_1 \) to the last charge in \( \omega_2 \).

Now we describe our overall algorithm for solving the second subproblem where the two particular widths involved are \( j + 1 \) and \( j \), for \( j \in U \setminus \{u_{\max}\} \).

Define \( \phi_i(h, y) \) to be the maximum total net redefined reward of a one-width cast with width \( j + 1 \) where there are \( y \) charges and the steel grade of the last charge is \( h \). Define \( \phi_z(k, z) \) to be the maximum total net redefined reward of a one-width cast with width \( j \) where there are \( z \) charges and the steel grade of the last charge is \( k \). Step 1. Apply Algorithm 1 for generating one-width casts with width \( j + 1 \). This generates the value of \( \phi_i(h; x_1, x_2, \ldots, x_g) \) for each possible state \( (h; x_1, x_2, \ldots, x_g) \).
max{\(\varphi(h; x_1, x_2, \ldots, x_{k})\mid x_k = 0, 1, \ldots, n_{k,j+1}\) for \(k = 1, 2, \ldots, g; \sum_{i=1}^{g} x_i = y\)}.

Step 2. Apply Algorithm 1 for generating one-width casts with width \(j\). This generates the value of \(\varphi(k; x_1, x_2, \ldots, x_{k})\) for each possible state \((k; x_1, x_2, \ldots, x_{k})\). For each \(k = 1, 2, \ldots, g, z = 1, 2, \ldots, \delta_j\), let \(\phi_z(k, z) = \max\{\varphi(k; x_1, x_2, \ldots, x_{k})\mid x_h = 0, 1, \ldots, n_{h,j}\text{ for } h = 1, 2, \ldots, g; \sum_{h=1}^{g} x_h = z\}\).

Step 3. The optimal solution is found by computing

\[
\max\{\phi_1(h, y) + \phi_2(k, z) - \alpha_{s_k} - \beta \mid h \in G, k \in G_h, y = 1, 2, \ldots, \delta_{j+1}, z = 1, 2, \ldots, \delta_j, t_{j+1}y + tz \leq T\}.
\]

Clearly, the time complexity of Step 3 is bounded by \(O(n^2g^2)\). So the overall complexity of the algorithm is bounded by \(O(n^3g^{3/2} + n^2g^2)\).

We note that if \(\Gamma_{j+1} \cap \Gamma_j = \emptyset\), then no charge appears more than once in the two-width cast found by Algorithm 2. If \(\Gamma_{j+1} \cap \Gamma_j \neq \emptyset\), then there may exist charges that appear twice in the two-width cast found by Algorithm 2. Consequently, we need to redefine the parameter \(a_{is}\) used in the formulation (SP) as the number of times that a charge with grade \(l\) is sequenced immediately before a charge with grade \(k\) in cast \(\omega\) if cast \(\omega\) uses one width \(j\), and 0 otherwise;

\(e_{l,j}^{1} = \) the number of times that a charge with grade \(l\) is sequenced immediately before a charge with grade \(k\) in cast \(\omega\) if cast \(\omega\) uses two widths \(j + 1\) and \(j\) and contains charge \(i\) for which width \(j + 1\) is used, and 0 otherwise;

\(e_{l,j}^{2} = \) the number of times that a charge with grade \(l\) is sequenced immediately before a charge with grade \(k\) in cast \(\omega\) if cast \(\omega\) uses two widths \(j + 1\) and \(j\) and contains charge \(i\) for which width \(j\) is used, and 0 otherwise.

Branching is performed based on the values of the following parameters:

\(k_j^l = \sum_{w \in \Omega} z_{l,j}^{w} \lambda_{w}\), the number of one-width casts with width \(j\) in the solution;

\(k_j^2 = \sum_{w \in \Omega} z_{l,j}^{w} \lambda_{w}\), the number of two-width casts with widths \(j + 1\) and \(j\) in the solution;

\(\nu_j = \sum_{w \in \Omega} a_{isw} \lambda_{w}\), the number of times charge \(i\) appears in the solution;

\(\mu_j^1 = \sum_{w \in \Omega} b_{ij}^{1w} \lambda_{w}\), the number of times charge \(i\) appears in a one-width cast with width \(j\);

\(\mu_j^2 = \sum_{w \in \Omega} b_{ij}^{2w} \lambda_{w}\), the number of times charge \(i\) appears in a two-width cast with widths \(j + 1\) and \(j\) where charge \(i\) uses width \(j\);

\(\mu_j^3 = \sum_{w \in \Omega} b_{ij}^{3w} \lambda_{w}\), the number of times charge \(i\) appears in a two-width cast with widths \(j + 1\) and \(j\) where charge \(i\) uses width \(j + 1\);

\(\sigma_j^1 = \sum_{w \in \Omega} e_{l,j}^{1w} \lambda_{w}\), the number of times that a charge with grade \(l\) is sequenced immediately before a charge with grade \(k\) in a one-width cast with width \(j\);

\(\sigma_j^2 = \sum_{w \in \Omega} e_{l,j}^{2w} \lambda_{w}\), the number of times that a charge with grade \(l\) is sequenced immediately before a charge with grade \(k\) in a two-width cast;

\(\sigma_j^3 = \sum_{w \in \Omega} e_{l,j}^{3w} \lambda_{w}\), the number of times that a charge with grade \(l\) is sequenced immediately before a charge with grade \(k\) in a two-width cast.

If the LP relaxation solution of the B&B node is fractional and the integer part of its objective value is greater than that of the best integer solution generated so far, then a branching procedure is performed. There are four possible cases to consider when we branch down this node:

Case 1. If some of the \(k_j^s\) (\(s = 1, 2\)) parameters are fractional, then we choose a parameter \(k_j^s\), for some \(s \in \{1, 2\}, j \in U\), whose fractional part is closest to 0.5, and create two child nodes by adding one of the following constraints to the corresponding master problem (LSP), respectively:

\[\sum_{w \in \Omega} z_{l,j}^{w} \lambda_{w} \leq \lfloor k_j^s \rfloor\text{ or }\sum_{w \in \Omega} z_{l,j}^{w} \lambda_{w} \geq \lceil k_j^s \rceil.\]

The two subproblems are modified accordingly by taking into account the dual variable value of this constraint.
The two subproblems are modified accordingly by taking into account the dual variable value of this constraint.

If all the \( \kappa_i^r \), \( v_l \), \( \mu_{ij}^r \), \( s_{ik}^r \) parameters are integer but the LP relaxation solution of the current B&B node is fractional, we can always convert the fractional solution into an integral solution with the same objective value by solving several small-scale integer programs.

4.3. A Frequently Occurring Special Case

Baosteel frequently (about four to five days per month) receives a large number of orders for special types of steel products such as shipbuilding heavy plates. Because the shipbuilding industry has a very high-quality requirement for the steel products, it imposes strict ingredients control in the steelmaking stage and requires charges with different grades to be cast in different tundishes. Consequently, any steel grade for shipbuilding is effectively incompatible with any other steel grade. Steelmaking shop 1 is responsible for casting the corresponding charges of such special steel products. In their daily planning, whenever the number of charges for steel products from the shipbuilding industry exceeds one day’s estimated capacity of three casters at shop 1, Baosteel decides to batch and cast these charges in the next day.

In this section, we consider the integrated charge batching and casting width selection problem for such a day when all the charges are for steel products for the shipbuilding industry and hence their grades are mutually incompatible. In this case, each steel grade \( k \in G \) is incompatible with any other grade in \( G \), and all the charges in a cast must have the same steel grade, and hence no grade switch cost is involved. Therefore, the charge batching and casting width selection decisions for the charges with one steel grade are independent of those for the charges with another steel grade, except that all the charges compete for the given \( m \) casts.

We decompose the problem into \( g \) subproblems, one for the charges of each grade \( k \in G \), and solve it through a two-level approach as follows.

At the higher level, the decision we need to make is how many casts to form using the charges of each grade. Define decision variable \( m_k \in \{1, 2, \ldots, m\} \) to be the number of casts to use for the charges of grade \( k \in G \). Denote the net reward of having \( m_k \) casts of charges with grade \( k \) by \( f_k(m_k) \) (which is obtained by solving the subproblem of grade \( k \) at the lower level). Then the higher-level problem can be formulated as the following integer program:

\[
\text{Maximize } \sum_{k \in G} f_k(m_k) \\
\text{subject to } \sum_{k \in G} m_k = m, \quad m_k \geq 0, \quad \text{integer}, \quad \forall \ k \in G.
\]
the net reward, denoted as $f_k(m_k)$, of having $m_k$ casts of charges with grade $k$, for $m_k \in \{1, 2, \ldots, m\}$. The net reward values are used as input of the problem at the higher level, as described earlier. Online Appendix 4 provides some optimality properties of the subproblem of a specific grade $k$ and uses these properties to develop an $O(mn^4)$ time dynamic programming algorithm for the subproblem.

5. System Implementation and Financial Impact

In this section, we describe the implementation of a decision support system (DSS) (§5.1), describe the manual solution method Baosteel was using before the DSS was implemented (§5.2), evaluate the performance of our optimization-based algorithms given in §§4.2 and 4.3 by comparing them to Baosteel’s manual method (§5.3), and discuss estimated financial benefits to Baosteel due to the use of the DSS (§5.4).

5.1. System Implementation

We started working on this project with Baosteel in 2006. At that time, Baosteel was already equipped with a sophisticated computer-based information system, which they called the production and sales integrated information system (PSIIS). This systems streamlined Baosteel’s data entry, data sharing, and data dissemination processes, and guaranteed data accuracy across the whole organization. However, PSIIS was a transactional IT system that lacked optimization-based decision-making capability for generating production plans and schedules for daily operations management. Human-machine interactive editors were used on top of PSIIS in which Baosteel’s expert planners carried out planning and scheduling decisions manually and added them to PSIIS.

A major goal of our collaboration with Baosteel was to streamline Baosteel’s decision-making processes and replace their manual planning and scheduling procedures by optimization-based decision support systems. One of the DSSs that we developed was for the charge batching and casting width selection problem studied in this paper. Our DSS was developed using Microsoft Visual C++ 6.0 integrated development environment. The algorithms developed in §4 were coded in C++ with all the LP problems solved by the LP solver of IBM ILOG CPLEX. These algorithms were compiled into a dynamic linked library that acts as an automatic solver and can be conveniently called from the DSS or other platforms.

To protect system security of PSIIS, as well as make our DSS portable, our DSS neither replaces PSIIS nor is imbedded into PSIIS. Instead, the DSS runs on the top of PSIIS through some data interface as follows. In the beginning of DSS installation, some nonfrequently-updated data (e.g., grades, widths, grade switch costs, casting time for each width, tundish lifespan, etc.) was imported into our DSS through retrieving related data from PSIIS or being configured by the planners. Such data rarely changes but can be dynamically updated whenever needed. When using the DSS for daily planning, the production data (i.e., the data about the charges to be batched) is downloaded from PSIIS and uploaded to DSS every day through a data interface. The planners then configure the downloaded data and start up the automatic solver embedded in DSS. After the solver is finished running, the solution is obtained and displayed on the screen as shown in Figure 3. The planners can evaluate and modify the solution according to their personal experience and preferences. Finally, the solution data is converted into PSIIS defined format and uploaded to PSIIS by a data interface. Since our DSS and PSIIS are kept separate and linked through a data interface, our DSS is highly portable and can be easily linked with existing information systems of other steel companies.

Our DSS was implemented and installed at the Manufacturing Management Department of Baosteel in March of 2008. At the beginning, it was only used by the group of planners managing the production plan of steelmaking shop 2 because our collaboration with Baosteel at that time was focused on shop 2. After one year’s successful use at shop 2, the planners of shop 2 recommended this tool to the planners of steelmaking shop 1, which then started to use the same DSS after a minor system modification. The two shops have some slight differences in production environment and operational management, but share main technical constraints and management objectives. Thus, we made a minor modification to the DSS and implemented it for shop 1.

Later, we found that once in awhile the planners at shop 1 were encountered with a special case of the problem as described in §4.3 where the different steel grades involved in the given set of charges for a day are all mutually incompatible and hence only the charges with the same grade can be batched to form a cast on that day. Our B&P algorithm was capable of solving this special case. However, we found that for this special case, the integrality gap between the LP relaxation solution and the IP solution is much larger than in the general case where not all grades are mutually incompatible. This is because in this special case there are fewer charges that can be batched together to form casts and hence the casts formed contain more unused capacity, leading to a more fractional LP relaxation solution. Therefore, this special case takes our B&P algorithm much longer to solve than the general case of the problem. Fortunately, we were able to exploit the structure of this special case and derive a polynomial-time dynamic programming based algorithm, as described in §4.3, to solve this case. This algorithm was added to our DSS and is run automatically whenever the system is encountered with this special case of the problem.

5.2. Baosteel’s Manual Method

For the charge batching and casting width selection problem, the manual method that Baosteel was using before our DSS was developed is a greedy procedure and can be described as follows.
The planner first views a width-grade matrix generated by PSIIS. Each element of the width-grade matrix records the number of charges \( n_{jk} \) that can be produced at the corresponding combination of width \( j \) and grade \( k \). From the width-grade matrix, the planner can get the number of charges \( \sum_{k \in G} n_{jk} \) and grades \( G_j = \{ k \in G \mid n_{jk} \neq 0 \} \) associated with each width \( j \). For each width \( j \), the planner uses the following steps to generate a cast. All charges associated with width \( j \) are retrieved, listed in a table, and then partitioned into \( |G_j| \) groups according to their grades. Charges belonging to each group are combined together as a lot. From all lots listed in the table, the planner chooses a pair of lots whose grade switch cost is no larger than that of any other pairs, and combines them into a new lot. Then the new lot is continuously combined with another lot until there is no other lot that can be combined because of grade incompatibility or tundish lifespan limit. The new lot is regarded as a cast. If the tundish has some remaining lifespan, the charges that belong to \( \Gamma_{j-1} \) but have not been covered by the cast are considered. They are added to the cast in a greedy way by considering the grade switch constraint and tundish lifespan constraint. After the planner has generated a cast for each possible casting width, the planner chooses the one that covers the most charges, and in case there is a tie, chooses the one with the least total grade and width switch cost. After a cast is chosen, all the charges covered by this cast are removed, and the above procedure is repeated until \( m \) casts have been generated.

Their manual method was straightforward and easy for any planner to follow. However, based on our own experience dealing with various complex combinatorial optimization problems, we believed from the very beginning of this project that such a simple heuristic would not work well for such a complex problem.

5.3. Pilot Tests

We conducted two pilot tests. The first pilot test used the 14 days of real production data, from March 3 to 16, 2006, from steelmaking shop 2. The main goal of this test was to see how much improvement our B&P algorithm described in §4 can achieve over the Baosteel’s manual method described in §5.2. The main characteristics of steelmaking shop 2 and the real data used in this pilot test are summarized as follows.

To accommodate customer’s requirements on different steel products, steelmaking shop 2 can produce about 200 steel grades. These steel grades are classified into about 10 nonoverlapping steel categories (e.g., low-carbon steel, medium-carbon steel, high-carbon steel, and so on) by considering differences between metallurgical compositions. Each steel category contains about 20 to 30 steel grades with similar metallurgical compositions. For each steel grade, the density of grade compatibility (the ratio between the number of compatible grade pairs and the total number of grade pairs) is about 60%–70%. However, steel grades that belong to different steel categories are always incompatible because there is significant difference on their metallurgical compositions. According to the operational philosophy of Baosteel, only the charges belonging to one particular steel category are considered in one day.
Given the daily production capacity of steelmaking shop 2 (which has two continuous casters), the expert planner decided that $m = 8$ casts should be generated every day during the two-week period covered in our pilot study. Based on this, $n = 67$ to 88 charges from a particular steel category with the number of steel grades $g$ varying from 18 to 23 were considered daily. A dozen widths (i.e., $r = 12$), 900 mm, 950 mm, …, 1,450 mm, can be set for both casters in the shop. The expert planner set the width switch cost ($\beta$) to $50$, and set the grade switch cost from $100$ to $400$, depending on specific pair of grades. For each charge, there were two to five allowable casting widths that can be used (e.g., 1,050 mm, 1,100 mm, 1,150 mm). The tundish lifespan ($T$) was set as 350 minutes. The casting time ($t_j$) for each charge $j$ ranged from 40 minutes to 70 minutes, depending on its steel grade.

The main parameters $(n, m, g, r)$ of the 14 days of real production data are shown in Table 1, where each row represents one day’s production.

In our optimization model and B&P algorithm, we set the reward $\rho$ for packing a charge to be $10,000$. We conducted a preliminary computational experiment to see how the value of $\rho$ impacts the performance of the algorithm. The results are reported in Online Appendix 5, which show that (i) as long as $\rho$ is sufficiently large (e.g., greater than the maximum possible total grade switch and width switch cost), it is guaranteed that the number of charges covered is maximized first, and after that, the total cost is minimized; and (ii) the computational time of the algorithm varies little with the value of $\rho$ used. We estimated that there would be no more than 20 grade switches and no more than eight width switches in a solution. Thus, in a solution, the total grade switch and width switch cost would not exceed $20 \times 400 + 8 \times 50 = 8,400$. This is why we set $\rho = 10,000$. Our results in Appendix 5 show that $\rho = 5,000$ is in fact already sufficient.

Table 1 shows the results of the pilot test, including the total number of charges covered and the total grade and width switch cost in the solution obtained by the manual method and that by the B&P algorithm, respectively, and the improvement on tundish utilization and total cost made by the B&P algorithm over the manual method. In addition, it also shows the integrality gap between the optimal solution and the LP relaxation solution obtained at the root B&B node, and the CPU time of the B&P algorithm.

Table 1 demonstrates that our B&P algorithm would improve tundish utilization by an average of 3.50% and reduce total grade and width switch cost by an average of 18.70% if it was used for the two-week period involved in the pilot test. Our algorithm would improve the tundish utilization in almost every day, and reduce total grade and width switch cost in 10 of the 14 days. The test also found that the LP relaxation of the root B&B node in our algorithm provided a very tight upper bound for every test instance. Because of this excellent upper bounding, the branching procedure was often terminated after exploring a small number of B&B nodes. For each test instance, our B&P algorithm was capable of finding optimal solutions within one minute, whereas the human planner often took about 30 minutes to devise a plan by the manual method. Thus, using our DSS also improves human planners’ productivity.

We have also tested performance robustness of our B&P algorithm using randomly generated test problems with more diverse parameter configurations than the real problems involved in the pilot test. In these test problems, $n$ varies from 60 to 150, $m$ from 6 to 12, $g$ from 20 to 30, and $r = 12$. The specifics of this test are given in Online Appendix 6. The results show that our B&P algorithm is capable of generating an optimal solution in a short amount of computational time for every test problem and consistently outperforms the manual method by a significant margin across all the test problems; it improves the tundish utilization in every case.
by 0.62% to 4.82%, and reduces the cost in most cases by 10% to 30%, over the manual method.

To gain some deeper insights about our B&P algorithm, we have also tested the effectiveness of the valid inequalities (5) and (6) using the same test problems as in the first pilot test. We found that (i) although valid inequality (5) does not strengthen the LP upper bound, it reduces the number of B&B nodes explored and hence the computational time by alleviating the problem of symmetry and reducing the solution space; (ii) valid inequality (6) strengthens the LP upper bound at the root node by an average of 0.43%, which leads to a significant reduction of the number of B&B nodes explored and the computational time; (iii) using both valid inequalities (5) and (6) is the most effective and achieves a 69.23% reduction of the number of B&B nodes explored and an 85.89% reduction of the computational time. The detailed test results are given in Online Appendix 7.

We conducted another pilot test to evaluate the performance of our polynomial-time dynamic programming-based algorithm for the special case of the problem described in §4.3 using five real instances collected from shop 1 in May, 2009. The main characteristics of steelmaking shops 1 and 2 are similar except that the daily production capacity of shop 1 is 1.5 times that of shop 2. Accordingly, the expert planner at shop 1 decided that \( m = 12 \) casts should be generated every day for shop 1. Each instance consisted of 90 to 115 charges with five to six grades involved, each with 10 to 30 charges. The computational results are shown in Table 2. Since all tested instances can be solved within one second by our algorithm, we do not report the CPU time in Table 2. We can observe that our optimal algorithm would improve the tundish utilization by 1.67% but increase the width switch cost by 26.92% on average. We note that it is not surprising that the width switch cost is increased for almost all instances by using our optimal algorithm. The manual method is a greedy heuristic that tries to pack as many charges with the same width as possible in a cast before adding charges with a narrower width if the tundish still has some remaining lifespan, which minimizes the width switch cost, whereas our algorithm considers the tundish utilization first before considering the width switch cost.

It should also be noted that in the case of the general problem, because for a given casting width there are more charges available that can be packed into a cast than in the special case of the problem, the greedy nature of the manual method no longer has any advantage over our B&P algorithm for most test instances with respect to total grade and width switch cost. This was clearly demonstrated in the first pilot test.

### 5.4. Benefits to Baosteel

As demonstrated in §5.3, the advantages of our optimal solution algorithms are improved tundish utilization for almost every case of the test instance and reduced grade and width switch cost for most test instances. By increasing tundish utilization and thereby reducing the number of tundishes needed to cover the given orders, not only the setup cost for repairing the used tundishes, but also the setup time for replacing tundishes and the total amount of scrap steel, are reduced. In the following, we quantify the annual benefits to Baosteel brought by replacing their manual planning approach with our DSS.

The average working time of a caster at shops 1 and 2 in a year is about 330 days. For shop 1, out of 330 working days in a year, about 50 (called special days) are dedicated to charges of products from the shipbuilding industry and the remaining 280 days (called regular days) deal with charges of regular products only. For shop 2, all the 330 days are regular days dealing with charges of regular products only. During a regular day, shop 1 and 2 can process an average of 75 and 50 charges, respectively. During a special day, shop 1 can process an average of 72 charges. Thus, the total yield of the two shops from the regular days in a year is 37,500 (=280 × 75 + 330 × 50) charges, and the yield of shop 1 from the special days in a year is 3,600 (=50 × 72) charges. According to the first pilot test described earlier, during a regular day the average tundish utilization is improved by 3.50% (from 50.79/8 = 6.35 charges per tundish to 52.57/8 = 6.57 charges per tundish) by using our B&P algorithm, which represents an approximate annual saving of 197.7 (=37,500/0.35 – 37,500/0.67) tundishes and setups. Similarly, according to the second pilot test, during a special day, the average tundish utilization is improved by 1.67% (from 72/12 = 6 charges per tundish to 73.2/12 = 6.1 charges per tundish), which implies an annual savings of about 9.8 (=3,600/6 – 3,600/6.1) tundishes and setups. Hence a total of about 207.5 (=197.7 + 9.8) fewer tundishes and setups are used annually.
It costs about US $4,000 to repair a used tundish, and hence a total annual cost savings for repairing tundishes is about US $0.83 million ($4,000 \times 207.5$). Furthermore, the production of each cast generates about 20 tons of scrap steel. The scrap steel is generated for the following reasons. To prevent the slag attached at the metal bath surface from getting in touch with the mold of the caster, some amount of liquid steel should be left in the tundish whenever a cast is completed. Also, a part of the first and last slabs in a cast should be cut because of quality requirements. A saving of 207.5 tundishes translates into saved scrap by about 4,150 ($=20 \times 207.5$) tons. It costs about US $100 to produce one ton of steel product in the steelmaking stage. Therefore, the total annual cost savings due to saved scrap steel is about US $0.41 million ($=4,150 \times 100$).

The set-up time between two casts is about 1.5 hours on average. A savings of 207.5 setups translates into increased production time by about 13 ($=1.5 \times 207.5/24$) days per year. Since about 25 charges can be produced every day by one caster, the increased production time of 13 days translates into increased productivity by about 325 ($=13 \times 25$) charges. Since the continuous casting step is a bottleneck in the overall steel production process, improvement on this step’s productivity increases the throughput of the whole company. The planners estimated that it earns approximately US $10,000 of revenue to produce a charge. Thus, the increased total annual revenue is about US $3.25 million ($=325 \times 10,000$).

Finally, there are also savings on grade and width switch costs. Based on the first pilot test, our algorithm yields a cost saving of about US $244.5 ($=244.5 \times 2\text{,}615 - 126$) per caster per regular day. Thus, the annual grade and width switch cost saving during the regular days by the five casters altogether is about US $0.36 million ($=330 \times 2 \times 244.5 + 280 \times 3 \times 244.5$). On the other hand, based on the second pilot test, during a special day, the width switch cost increases by US $70 ($=330 - 260$) on average. Thus the total cost increase during the 50 special days in a year is US $3,500 ($=50 \times 70$), which is negligible compared to the cost savings calculated earlier.

Based on the above calculations, the estimated annual financial impact to Baosteel by replacing their manual method with our optimization based DSS includes a cost saving of about US $1.6 ($=0.83 + 0.41 + 0.36$) million, and a revenue increase of about US $3.25 million.

We note that the above benefit estimation, which is based on the two pilot tests, can be validated using the aggregate level production data from Baosteel. According to the data provided by Baosteel, in 2006, one year before our DSS was implemented, the average tundish utilization was 6.15 charges per tundish. In 2009 to 2011, after the DSS was installed at both steelmaking shops, the average tundish utilization has increased to 6.41 charges per tundish, an improvement of 4.2% compared to 2006. The planners at Baosteel believe that this improvement is mainly due to the use of DSS. There may also exist other factors, e.g., differences in order compositions in different years, that may have contributed to this improvement to a lesser extent.

In addition to these quantifiable benefits, other benefits to Baosteel include improved quality of the products, less time required for planning, and more time available for planners to do what-if analysis.

6. Conclusions

In this paper, we have described our collaboration with Baosteel on their integrated charge batching and casting width selection problem. We have developed a column generation based branch-and-price exact solution approach for the general problem and a polynomial-time dynamic programming-based algorithm for a special case of the problem. We have implemented a decision support system at Baosteel using these optimization-based algorithms to replace their manual solution method. This has streamlined their daily production planning and brought significant financial benefits to the company.

Beyond Baosteel, we believe that our research can also be applied to other large iron and steel enterprises that have a good information system in place. Our DSS can be easily modified and hooked to an existing information system to replace existing nonoptimization-based approaches and find a better solution to the charge batching and casting width selection problem.

Supplemental Material

Supplemental material to this paper is available at http://dx.doi.org/10.1287/opre.2014.1278.

Acknowledgments

The authors would like to thank the associate editor and three referees for their constructive comments and suggestions, which have been used to improve the readability of the paper. This research is partly supported by the State Key Program of the National Natural Science Foundation of China [Grant 71032004] and the Fund for Innovative Research Groups of the National Natural Science Foundation of China [Grant 71321001].

References


Tang, Wang, and Chen: Integrated Charge Batchting and Casting Width Selection at Baosteel


**Lixin Tang** is Cheung Kong Scholars Chair Professor and director of the Logistics Institute at Northeastern University of China. His research interests include batching and scheduling of the production operations, logistics planning and scheduling, integer programming, and combinatorial optimization. He has implemented a variety of production-logistics planning and scheduling decision support systems for iron and steel companies. His research papers have appeared in academic journals such as *Naval Research Logistics*, *IIE Transactions*, *IEEE Transactions on Evolutionary Computation*, *IEEE Transactions on Power Systems*, *European Journal of Operational Research*, and *Annals of Operations Research*.

**Gongshu Wang** is an associate professor at the Logistics Institute at Northeastern University of China. His major research interests are in the areas of production planning and scheduling in the steel industry and integer and combinatorial optimization methods. He has published several articles in academic journals including *Naval Research Logistics*, *Omega*, and *Computers & Operations Research*.

**Zhi-Long Chen** is a professor of operations management at the Robert H. Smith School of Business, University of Maryland, College Park. He is also a Cheung Kong Scholars Visiting Chair Professor at the Logistics Institute of Northeastern University of China. Dr. Chen does research in the areas of supply chain scheduling and coordination, routing and scheduling of logistics operations, and dynamic pricing. His research has been funded by several grants from the National Science Foundation of the United States. His work has appeared in academic journals such as *Operations Research*, *Management Science*, *Naval Research Logistics*, *IIE Transactions*, *Journal of Scheduling*, *Manufacturing and Service Operations Management*, *Production and Operations Management*, *Operations Research Letters*, and *European Journal of Operational Research*. 