TALENT HOLD COST MINIMIZATION IN FILM PRODUCTION

TAI CHIU EDWIN CHENG
Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University
Hung Hom, Kowloon, Hong Kong

BERTRAND MIAO-TSONG LIN∗ AND HSIAO-LAN HUANG
Institute of Information Management, National Chiao Tung University
Hsinchu 300, Taiwan

(Communicated by Ada Che)

Abstract. This paper investigates the talent scheduling problem in film production, which is known as rehearsal scheduling in music and dance performances. The first lower bound on the minimization of talent hold cost is based upon the outside-in branching strategy. We introduce two approaches to add extra terms for tightening the lower bound. The first approach is to formulate a maximum weighted matching problem. The second approach is to retrieve structural information and solve a maximum weighted 3-grouping problem. We make two contributions: First, our results can fathom the matrix of a given partial schedule. Second, our second approach is free from the requirement to schedule some shooting days in advance for providing anchoring information as in the other approaches, i.e., a lower bound can be computed once the input instance is given. The lower bound can fit different branching strategies. Moreover, the second contribution provides a state-of-the-art research result for this problem. Computational experiments confirm that the new bounds are much tighter than the original one.

1. Introduction. Production of motion pictures is a complex process. In this paper we investigate the talent scheduling problem, which is one of the crucial decisions in the budget planning of film production [11]. The first study on talent scheduling by Cheng et al. [3] was inspired by a film company in Calgary, Canada. The talent scheduling problem can be cast as rehearsal scheduling with applications in music [12] and dance [14]. The mathematical origin of the problem can be traced to the serialization problem [1 [4] [6] [7]. Other interesting applications include archaeological analysis [8], DNA analysis [5], and others [1 [2].

We adopt the notation in Cheng et al. [3]. A film production plan has n shooting days d1, d2, . . . , dn that require m actors a1, a2, . . . , am. A 0-1 day-out-of-days matrix (DODM) T = (tij)mxn specifies that tij = 1 if actor ai is required on shooting day dj; 0, otherwise. The talent hold cost column vector C = (c1, c2, . . . , cm)T
specifies the daily hold cost $c_i$ of actor $a_i$. In the context of concert rehearsal scheduling, each day is referred to as a piece of music and is additionally associated with a duration specifying the length of this scene. The results presented in this paper can be easily generalized to the setting with durations.

The planning of shooting days consists of two types of talent cost, namely base cost and hold cost. The base cost of actor $a_i$ is equal to the number of shooting days he needs to be present times his daily pay cost, i.e., $c_i \times \sum_{j=1}^{n} t_{i,j}$. The base cost is fixed regardless of how the shooting days are arranged. The hold cost is the extra cost for retaining an actor on location although the actor is not required. To illustrate the problem definition, consider the instance shown in Figure 1. If the default shooting sequence $(d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10})$ is adopted, then hold days will occur to the actors. For example, actor $a_3$ has to be present from the third day to the eighth day, which incurs three hold days: $d_4, d_6,$ and $d_7$. The base cost of actor $a_3$ is $3 \times 1,100$ and the extra cost under the shooting sequence is $3 \times 1,100$.

In the given instance with the default shooting sequence, the hold days for all the actors are indicated by the underlined zeros.

Through film-editing in post-production, the various components of the film are not necessarily filmed in the same order as they appear in the final version of the film. Let $\sigma$ be a permutation of the shooting days. With respect to sequence $\sigma$, let $e_i(\sigma)$ and $l_i(\sigma)$ be the first and last working day of actor $a_i$, respectively. Actor $a_i$ must be present from the beginning of day $e_i(\sigma)$ to the end of day $l_i(\sigma)$. Any day between $e_i(\sigma)$ and $l_i(\sigma)$ that does not require actor $a_i$ incurs a hold day to him. The number of hold days of actor $a_i$ is given by $h_i(\sigma) = (l_i(\sigma) - e_i(\sigma) + 1) - \sum_{j=e_i(\sigma)}^{l_i(\sigma)} t_{i,j}$. Therefore, the total hold cost of sequence $\sigma$ for the film is the sum of the hold costs of all the actors, i.e., $H(\sigma) = \sum_{i=1}^{m} c_i \times h_i(\sigma)$. Let $\Pi$ be the set of all the permutations of the shooting days. The problem is to find $\sigma^* \in \Pi$ such that $H(\sigma^*) = \min_{\sigma \in \Pi} \{H(\sigma)\}$.

| actors $a_i$ | $d_1$ | $d_2$ | $d_3$ | $d_4$ | $d_5$ | $d_6$ | $d_7$ | $d_8$ | $d_9$ | $d_{10}$ | hold cost $c_i$ |
|--------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|----------------|
| $a_1$        | 1     | 1     | 1     | 0     | 0     | 1     | 0     | 0     | 0     | 1     | 800            |
| $a_2$        | 1     | 1     | 0     | 1     | 0     | 0     | 0     | 0     | 1     | 0     | 700            |
| $a_3$        | 0     | 0     | 1     | 0     | 0     | 0     | 1     | 0     | 0     | 1     | 1,100          |
| $a_4$        | 1     | 1     | 0     | 1     | 1     | 1     | 0     | 0     | 1     | 1     | 550            |
| $a_5$        | 1     | 0     | 0     | 1     | 0     | 0     | 0     | 1     | 0     | 0     | 450            |
| $a_6$        | 0     | 1     | 0     | 0     | 1     | 0     | 0     | 1     | 0     | 1     | 750            |
| $a_7$        | 0     | 0     | 1     | 1     | 0     | 1     | 1     | 1     | 0     | 1     | 610            |
| $a_8$        | 0     | 0     | 0     | 1     | 1     | 0     | 0     | 0     | 0     | 1     | 500            |
| $a_9$        | 1     | 0     | 0     | 0     | 1     | 1     | 1     | 0     | 0     | 0     | 900            |
| $a_{10}$     | 0     | 0     | 1     | 0     | 1     | 0     | 0     | 0     | 1     | 0     | 850            |
| $a_{11}$     | 1     | 0     | 0     | 0     | 1     | 0     | 0     | 1     | 0     | 0     | 800            |
| $a_{12}$     | 1     | 0     | 0     | 1     | 1     | 0     | 0     | 0     | 1     | 0     | 700            |

*Figure 1. Day-out-of-days matrix.*
Off-the-shelf software, such as Sunfrog Film Scheduling and Gorilla, is available for managing film production. Still, analytic approaches are required to be embedded in the decision engines to compose effective schedules in an automatic manner. Cheng et al. presented an outside-in approach for exploring the enumeration tree in the development of a branch-and-bound algorithm. They proposed a lower bound to prune the unnecessary branching. Nordström and Tufekçi proposed a genetic algorithm for producing approximate solutions. We notice that the studied model is related to the serialization problem, for which Adelson et al. designed an $O(mn2^n)$ dynamic programming algorithm to minimize the number of enclosed zero’s. Based upon the lower bound of Cheng et al., Lin gave an idea to improve the lower bound. de la Banda et al. proposed several dominance properties that help the dynamic program and the branch-and-bound algorithm to curtail the unnecessary recursion and conducted extensive experiments to test the performance of the new properties. The main contribution of this study is to provide a systematic development of lower bounds for the talent scheduling problem.

We organize the remaining part of the paper as follows: In Section 2 we review the lower bound provided by Cheng et al. In Section 3 we introduce an approach to enhance the lower bound. In Section 4 we present another approach to enhance the lower bound. In Section 5 we present and discuss computational results on evaluating the merits of the proposed lower bounds. We conclude the paper and suggest topics for future research in Section 6.

2. First lower bound. One of the potential uses of upper or lower bounds in dealing with optimization problems is to facilitate the design of efficient branch-and-bound algorithms. To construct a search tree for solving a sequencing problem, the most commonly adopted approach is to iteratively assign items to positions of the sequence starting from the front end. For some problems, like scheduling to minimize the total tardiness, sequences can be constructed from the rear end so as to achieve a tighter lower bound. To facilitate the design of lower bounds for use in branch-and-bound algorithms to solve the talent scheduling problem, Cheng et al. proposed the outside-in strategy that assigns days to the front end and the rear end simultaneously.

Denote $P$ as a partial sequence, in which the first $k_1$ days and the last $k_2$ days of the shooting plan have been determined. If actor $a_i$ has to show up on at least one of the first $k_1$ days, then let $\varepsilon_i(P) = 1$; otherwise, $\varepsilon_i(P) = 0$. If actor $a_i$ has to be present on at least one of the last $k_2$ days, then $\lambda_i(P) = 1$; otherwise, $\lambda_i(P) = 0$. For $\varepsilon_i(P) = 1$, denote $e_{i,1}(P)$ (respectively, $e_{i,2}(P)$) as the start day (respectively, end day) of actor $a_i$ among the first $k_1$ days. Similarly, for $\lambda_i(P) = 1$, denote $l_{i,1}(P)$ (respectively, $l_{i,2}(P)$) as the start day (respectively, end day) of actor $a_i$ among the last $k_2$ days. In the following equations, $P$ denotes the set of the remaining unscheduled days. Figure gives a visual depiction of the zones in which hold days might or might not be included. Note that the hold days within the purple shaded zones are calculated for they are inevitable subject to the partial schedule $P$. The potential hold days among the remaining days of $P$ are not included because $P$ could be optimally scheduled without any hold days. Depending on the scenarios of each actor in the partial schedule $P$, the number of hold days of actor $a_i$ is calculated as follows:

---

2Sunfrog Tech: sunfrog-tech.com
3Jungle Software: www.junglesoftware.com/products/gorilla_features.php
Case 1: $\varepsilon_i(P) = 1$ and $\lambda_i(P) = 1$:

$$h_i(P) = \sum_{j=\varepsilon_i(P)+1}^{k_1} (1 - t_{i,j}) + \sum_{j=n-k_2+1}^{l_{i,2}(P)-1} (1 - t_{i,j}) + \sum_{d_j \in \mathcal{P}} (1 - t_{i,j}). \quad (1)$$

Case 2: $\varepsilon_i(P) = 1$ and $\lambda_i(P) = 0$:

$$h_i(P) = \begin{cases} 
\sum_{j=\varepsilon_i(P)+1}^{k_1} (1 - t_{i,j}), & \text{if } \sum_{d_j \in \mathcal{P}} t_{i,j} \geq 1; \\
\sum_{j=\varepsilon_i(P)+1}^{l_{i,2}(P)-1} (1 - t_{i,j}), & \text{otherwise.}
\end{cases} \quad (2)$$

Case 3: $\varepsilon_i(P) = 0$ and $\lambda_i(P) = 1$:

$$h_i(P) = \begin{cases} 
\sum_{j=n-k_2+1}^{l_{i,2}(P)-1} (1 - t_{i,j}), & \text{if } \sum_{d_j \in \mathcal{P}} t_{i,j} \geq 1; \\
\sum_{j=\varepsilon_i(P)+1}^{l_{i,1}(P)+1} (1 - t_{i,j}), & \text{otherwise.}
\end{cases} \quad (3)$$

Case 4: $\varepsilon_i(P) = 0$ and $\lambda_i(P) = 0$:

In this case, no hold days are incurred if $\tilde{P}$ is optimally scheduled. Incidentally, our new developments will improve this situation.

Combining the above analysis, the first lower bound for any complete schedule derived from the partial schedule $P$ is given by

$$LB_1(P) = \sum_{i=1}^{m} c_i h_i(P).$$

As depicted in Figure color-matrix, the first three cases correspond to the purple shaded areas in segments (a), (b), and (c). The lower bound $LB_1$ comprises the inevitable hold cost in the purple shaded area. Consider the partial schedule $P$, in which $\sigma_1 = d_2, \sigma_2 = d_3, \sigma_9 = d_6$, and $\sigma_{10} = d_7$, given in Figure 3 as an example. The lower bound is calculated as $LB_1(P) = (4 \times 800 + 3 \times 550 + 3 \times 610) + (1 \times 700 + 1 \times 750) = 8,130$.

3. **Maximum weighted matching.** Attempting to identify more inevitable hold days from among the unscheduled days, we focus on the green shaded areas in segments (b) and (c) in Figure 2. With regard to segment (b) in Figure 2, define $\Phi(P) = \{a_i | \varepsilon_i(P) = 1 \text{ and } \lambda_i(P) = 0\}$, i.e., $\Phi(P)$ contains actors who are required by at least one of the first $k_1$ days, but not by any of the last $k_2$ days. Similarly, define set $\Lambda(P) = \{a_i | \varepsilon_i(P) = 0 \text{ and } \lambda_i(P) = 1\}$ corresponding to segment (c).

Using the instance in Figure 1, we assume that days $d_2, d_3, d_6$, and $d_7$ are fixed in the partial schedule $P$. Grouping the actors according to the colour segments, we have the partial matrix shown in Figure 3 with $\Phi(P) = \{a_2, a_3, a_6, a_{10}\}$ and $\Lambda(P) = \{a_9\}$. We discuss the hold cost that will be inevitably incurred by the actors of $\Phi(P)$.
For actors $a_{i_1}, a_{i_2} \in \Phi(P)$, let $\alpha_{i_1,i_2}(P)$ denote the set of unscheduled days that require actor $a_{i_1}$ but not actor $a_{i_2}$, i.e., $\alpha_{i_1,i_2}(P) = \{d_j| t_{i_1,j} = 1, t_{i_2,j} = 0\}$. Define set $\alpha_{i_1,i_2}(P) = \{d_j| t_{i_1,j} = 0, t_{i_2,j} = 1\}$ similarly. For the actors of $\Phi(P) = \{a_2, a_3, a_6, a_{10}\}$, we have the subsets:

- $\alpha_{2,3}(P) = \{d_1, d_4, d_9\}$, $\alpha_{2,3}(P) = \{d_5, d_8\}$;
- $\alpha_{2,6}(P) = \{d_1, d_4, d_9\}$, $\alpha_{2,6}(P) = \{d_5, d_8, d_{10}\}$;
- $\alpha_{2,10}(P) = \{d_1, d_4\}$, $\alpha_{2,10}(P) = \{d_5\}$;
- $\alpha_{3,6}(P) = \emptyset$, $\alpha_{3,6}(P) = \{d_{10}\}$;
- $\alpha_{3,10}(P) = \{d_8\}$, $\alpha_{3,10}(P) = \{d_9\}$;
- $\alpha_{6,10}(P) = \{d_8, d_{10}\}$, $\alpha_{6,10}(P) = \{d_9\}$.

Through these subsets, we have the following property.
Lemma 3.1. Given partial schedule $P$, actors $a_{i_1}, a_{i_2} \in \Phi(P)$ together incur a hold cost of at least
\[ w_{i_1, i_2} = \min \left\{ c_{i_1 | \alpha_{i_1, i_2}^- (P) |}, c_{i_2 | \alpha_{i_1, i_2}^- (P) |} \right\} \]
for the unscheduled days of $\alpha_{i_1, i_2}^- (P) \cup \alpha_{i_1, i_2}^- (P)$ in any complete schedule derived from $P$.

Proof. Consider the illustrative arrangement of the unscheduled days in Figure 4 for actors $a_i$ and $a_j$. If in some optimal schedule derived from the partial schedule $P$, the last day of $\alpha_{i_1, i_2}^- (P) \cup \alpha_{i_1, i_2}^- (P)$ is from $\alpha_{i_1, i_2}^- (P)$, then actor $a_{i_2}$ is idle for at least $|\alpha_{i_1, i_2}^- (P)|$ days, resulting in a cost of $c_{i_1 | \alpha_{i_1, i_2}^- (P) |}$. The counterpart case leading to a cost not less than $c_{i_1 | \alpha_{i_1, i_2}^- (P) |}$ of actor $a_{i_1}$ can be similarly analyzed. The lemma thus follows.

\[
\begin{array}{c|c|c|c}
\alpha_{i_1, i_2}^- \cup \alpha_{i_1, i_2}^- & a_1 & 1000 & 1111 \\
& a_2 & 1111 & 0000 \\
\end{array}
\]

Figure 4. Illustration of Lemma 3.1

Lemma 3.1 is valid for any pair of actors of $\Phi(P)$. Taking the sum of the pairing costs corresponding to disjoint pairs provides an extra value not counted in $LB_1(P)$. In order to obtain a tighter lower bound on the hold cost of the actors of $\Phi(P)$ with respect to the unscheduled days of $P$, we can find an optimal pairing of the actors of $\Phi(P)$ by constructing a weighted graph $G(\Phi(P)) = (V(\Phi(P)), E(\Phi(P)))$ as follows: Each actor $a_i \in \Phi(P)$ is represented as a node $v_i \in V(\Phi(P))$. Every two nodes $v_{i_1}$ and $v_{i_2}$ of $V(\Phi(P))$ are connected by an edge $(v_{i_1}, v_{i_2}) \in E(\Phi(P))$ with weight $w_{i_1, i_2} = \min \{c_{i_1 | \alpha_{i_1, i_2}^- (P) |}, c_{i_2 | \alpha_{i_1, i_2}^- (P) |}\}$. So the maximum weighted matching of $G(\Phi(P)) = (V(\Phi(P)), E(\Phi(P)))$ yields a lower bound $Max-w\text{-}matching(\Phi(P))$ for the actors $\Phi(P)$.

Applying the same line of reasoning to the actors of subset $\Lambda(P)$, we obtain the following theorem.

Theorem 3.2. Given partial schedule $P$,
\[ LB_2(P) = LB_1(P) + Max-w\text{-}matching(\Phi(P)) + Max-w\text{-}matching(\Lambda(P)) \]
is a lower bound on the optimal solution derived from $P$. □

Applying Lemma 3.1 to the partial schedule $P$ given in Figure 3, we have:
\[
\begin{align*}
w_{a_2, a_3} &= \min \{700 \times 2, 1100 \times 3\} = 1,400, \\
w_{a_2, a_6} &= \min \{700 \times 3, 750 \times 3\} = 2,100, \\
w_{a_2, a_10} &= \min \{700 \times 1, 850 \times 2\} = 700, \\
w_{a_3, a_6} &= \min \{1100 \times 1, 750 \times 0\} = 0, \\
w_{a_3, a_10} &= \min \{1100 \times 1, 850 \times 1\} = 850, \\
w_{a_6, a_10} &= \min \{750 \times 1, 850 \times 2\} = 750.
\end{align*}
\]
The resulting weighted graph is shown in Figure 5. $Max-matching(\Phi(P))$ is calculated as $2,100+850=2,950$. Since $\Lambda(P)$ contains only $a_9$, $Max-matching(\Lambda(P)) = 0$. $LB_2(P) = LB_1(P) + 2,950 = 11,080$. 

\[ \]
It is noted that preparing the weighted graph takes $O(m^2 n)$ time and solving the maximum weighted matching problem takes $O(m^3)$ time, so the overall running time to compute $LB_2(P)$ is $O(m^2 \max\{m, n\})$.

Another advantage of the lower bound $LB_2$ is its applicability in branch-and-bound algorithms that adopt sequential branching. Specifically, fixing some positions at the front end, we can deploy Theorem 3.2 to obtain a lower bound.

4. Maximum weighted 3-grouping. In the development of lower bounds $LB_1$ and $LB_2$, some shooting days are already scheduled at the front part and/or at the rear part as sentinels, which provides anchoring information on zeros enclosed between two or more ones on a row. In the following we improve the lower bound $LB_2$ by considering the yellow shaded area (d) in Figure 2. The contribution of the new development is not only an enhancement of the lower bound $LB_2$, but also the establishment of a new lower bound itself. The new development provides a viable way to calculate a lower bound based on the input instance only, i.e., we can derive a lower bound at the root node without the need to fix any days in specific positions.

For example, consider three actors $a_{i_1}$, $a_{i_2}$, and $a_{i_3}$ who are not required by the first $k_1$ days nor the last $k_2$ days. The unscheduled days can be categorized into the following eight disjoint subsets:

$$
\begin{align*}
\beta_{i_1, i_2, i_3} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (0, 0, 0)^T\}; \\
\beta_{\tilde{i_1, i_2, i_3}} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (0, 1, 1)^T\}; \\
\beta_{i_1, \tilde{i_2, i_3}} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (1, 1, 0)^T\}; \\
\beta_{\tilde{i_1, i_2, i_3}} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (0, 0, 1)^T\}; \\
\beta_{\tilde{i_1, \tilde{i_2, i_3}}}} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (0, 1, 0)^T\}; \\
\beta_{i_1, \tilde{i_2, i_3}} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (1, 0, 0)^T\}; \\
\beta_{\tilde{i_1, \tilde{i_2, i_3}}} & = \{d_j \in P| (t_{i_1,j}, t_{i_2,j}, t_{i_3,j})^T = (1, 1, 1)^T\}.
\end{align*}
$$

Take subset $\beta_{i_1, i_2, i_3}$ for example. It contains the unscheduled days that require actor $a_{i_2}$ and actor $a_{i_3}$, but not actor $a_{i_1}$. The following lemma gives a lower bound on the hold cost when actors $a_{i_1}, a_{i_2}$, and $a_{i_3}$ are considered simultaneously.

**Lemma 4.1.** In any complete schedule derived from the partial schedule $P$, actors $a_{i_1}, a_{i_2}$, and $a_{i_3}$ together incur a hold cost of at least $z_{i_1, i_2, i_3} = x_{i_1, i_2, i_3} + y_{i_1, i_2, i_3}$.
where \( x_{i_1,i_2,i_3} = \min \{ c_{i_1} | \beta_{1,i_1,i_2,i_3} |, c_{i_2} | \beta_{1,i_1,i_2,i_3} |, c_{i_3} | \beta_{1,i_1,i_2,i_3} | \}, \) and

\[
y_{i_1,i_2,i_3} = \begin{cases} 
0, & \text{if } \beta_{1,i_1,i_2,i_3} = \emptyset; \\
\min \left\{ \min \{ c_{i_2}, c_{i_3} \} | \beta_{1,i_1,i_2,i_3} |, \min \{ c_{i_1}, c_{i_2} \} | \beta_{1,i_1,i_2,i_3} |, \min \{ c_{i_1}, c_{i_3} \} | \beta_{1,i_1,i_2,i_3} | \right\}, & \text{otherwise.} 
\end{cases}
\]

Proof. There are eight disjoint subsets of the unscheduled jobs. To develop the first term \( x_{i_1,i_2,i_3} \), we consider three subsets \( \beta_{1,i_1,i_2,i_3} \), \( \beta_{1,i_1,i_2,i_3} \), and \( \beta_{1,i_1,i_2,i_3} \) of the unscheduled jobs. In the following discussion, when we call the first (last) job, we refer to the first (last) job among the jobs of \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) in an optimal schedule derived from the partial schedule \( P \). Table 1 shows different combinations of the first job and the last job. We discuss x-1 and x-4 of the six possible cases. Other cases can be similarly analyzed.

Case x-1: The first and the last jobs are both from \( \beta_{1,i_1,i_2,i_3} \) (Figure 6(a)). Actor \( a_{i_2} \) has at least \( | \beta_{1,i_1,i_2,i_3} | \) days idle and actor \( a_{i_3} \) has at least \( | \beta_{1,i_1,i_2,i_3} | \) days idle. The hold cost of this case is no less than \( c_{i_2} | \beta_{1,i_1,i_2,i_3} | + c_{i_3} | \beta_{1,i_1,i_2,i_3} | \).

Case x-4: The first job is from \( \beta_{1,i_1,i_2,i_3} \) and the last job is from \( \beta_{1,i_1,i_2,i_3} \) (Figure 6(b)). Actor \( a_{i_3} \) has at least \( | \beta_{1,i_1,i_2,i_3} | \) days idle, resulting in a hold cost of \( c_{i_3} | \beta_{1,i_1,i_2,i_3} | \).

\[
\begin{align*}
\beta_{1,i_1,i_2,i_3} & \cup \beta_{1,i_1,i_2,i_3} \\
\beta_{1,i_1,i_2,i_3} & \cup \beta_{1,i_1,i_2,i_3} \\
\beta_{1,i_1,i_2,i_3} & \cup \beta_{1,i_1,i_2,i_3}
\end{align*}
\]

Figure 6. Analysis of \( x_{i_1,i_2,i_3} \) in Lemma 4.1

From the above analysis of the six cases shown in Table 1, we see that for the unscheduled days of \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \), actors \( a_{i_1}, a_{i_2}, \) and \( a_{i_3} \) together incur a hold cost of at least \( x_{i_1,i_2,i_3} = \min \{ c_{i_1} | \beta_{1,i_1,i_2,i_3} |, c_{i_2} | \beta_{1,i_1,i_2,i_3} |, c_{i_3} | \beta_{1,i_1,i_2,i_3} | \} \).

| Table 1. Development of \( x_{i_1,i_2,i_3} \) in Lemma 4.1 |
|---|---|
| Arrangement | Minimum hold cost |
| x-1: \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) | \( c_{i_1} | \beta_{1,i_1,i_2,i_3} | + c_{i_2} | \beta_{1,i_1,i_2,i_3} | \) |
| x-2: \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) | \( c_{i_2} | \beta_{1,i_1,i_2,i_3} | + c_{i_3} | \beta_{1,i_1,i_2,i_3} | \) |
| x-3: \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) | \( c_{i_3} | \beta_{1,i_1,i_2,i_3} | + c_{i_2} | \beta_{1,i_1,i_2,i_3} | \) |
| x-4: \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) | \( c_{i_3} | \beta_{1,i_1,i_2,i_3} | \) |
| x-5: \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) | \( c_{i_2} | \beta_{1,i_1,i_2,i_3} | \) |
| x-6: \( \beta_{1,i_1,i_2,i_3} \cup \beta_{1,i_1,i_2,i_3} \) | \( c_{i_1} | \beta_{1,i_1,i_2,i_3} | \) |

\( x_{i_1,i_2,i_3} = \min \{ c_{i_1} | \beta_{1,i_1,i_2,i_3} |, c_{i_2} | \beta_{1,i_1,i_2,i_3} |, c_{i_3} | \beta_{1,i_1,i_2,i_3} | \} \)
Next, we proceed to analyze the second term \( y_{1,2,i_3} \). The unscheduled shooting days of the four disjoint subsets \( \beta_{1,1,2,i_3}, \beta_{1,2,1,i_3}, \beta_{1,1,2,i_3}, \beta_{1,2,1,i_3} \), and \( \beta_{1,1,2,i_3} \) are considered. We analyze the cases corresponding to the combinations of the first day and the last day. Referring to Table 2, we discuss cases y-1, y-2, y-5, and y-8.

### Table 2. Analysis of \( y_{1,2,i_3} \) for actors \( a_{i_2} \), \( a_{i_3} \), and \( a_{i_5} \).

| Arrangement | Lower bound of costs |
|-------------|----------------------|
| y-1: \( \beta_{1,1,2,i_3} \) | \( c_{i_2}(|\beta_{1,1,2,i_3}| + |\beta_{1,2,1,i_3}|) + c_{i_3}(|\beta_{1,1,2,i_3}| + |\beta_{1,2,1,i_3}|) \) |
| y-2: \( \beta_{1,1,2,i_3} \) | \( c_{i_2}|\beta_{1,1,2,i_3}| \) |
| y-3: \( \beta_{1,2,1,i_3} \) | \( c_{i_3}|\beta_{1,1,2,i_3} \cup \beta_{1,2,1,i_3}| \) |
| y-4: \( \beta_{1,2,1,i_3} \) | \( c_{i_2}|\beta_{1,1,2,i_3} \cup \beta_{1,2,1,i_3}| \) |
| y-5: \( \beta_{1,1,2,i_3} \) | \( c_{i_3}|\beta_{1,1,2,i_3}| \) |
| y-6: \( \beta_{1,1,2,i_3} \) | \( \min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3}| \) |
| y-7: \( \beta_{1,2,1,i_3} \) | \( \min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3} \cup \beta_{1,2,1,i_3}| \) |
| y-8: \( \beta_{1,1,2,i_3} \) | \( \min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3}| \) |
| y-9: \( \beta_{1,2,1,i_3} \) | \( \min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3} \cup \beta_{1,2,1,i_3}| \) |
| y-10: \( \beta_{1,1,2,i_3} \) | \( \min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3}| \) |

\( y_{1,2,i_3} = \min\{\min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3}|, \min(c_{i_1}, c_{i_3})|\beta_{1,1,2,i_3}|, \min(c_{i_2}, c_{i_3})|\beta_{1,1,2,i_3}|\} \)

**Case y-1:** The first job and the last job are both from \( \beta_{1,1,2,i_3} \).
It is clear that all the hold days of all the actors are enclosed by two 1’s. The cost is given by \( c_{i_2}(|\beta_{1,1,2,i_3}| + |\beta_{1,2,1,i_3}|) + c_{i_3}(|\beta_{1,1,2,i_3}| + |\beta_{1,2,1,i_3}|) \).

**Case y-2:** The first job and the last job are both from \( \beta_{1,1,2,i_3} \) (Figure 7(a)).
Actor \( a_{i_3} \) inevitably has a hold cost of \( c_{i_3}|\beta_{1,1,2,i_3} \cup \beta_{1,2,1,i_3}| \). If the days of \( \beta_{1,1,2,i_3} \) are scheduled in the central part, actor \( a_{i_2} \) and actor \( a_{i_3} \) may have no hold day. The minimum cost of this case is thus \( c_{i_3}|\beta_{1,1,2,i_3} \cup \beta_{1,2,1,i_3}| \).

**Case y-5:** The first day is from \( \beta_{1,1,2,i_3} \) and the last day is from \( \beta_{1,1,2,i_3} \) (Figure 7(b)).
The unscheduled shooting days of \( \beta_{1,1,2,i_3} \) and \( \beta_{1,2,1,i_3} \) may become hold days to actors \( a_{i_2} \) and \( a_{i_3} \). The first and the last days, together with any shooting day of \( \beta_{1,1,2,i_3} \) scheduled somewhere in between, will result in actor \( a_{i_2} \) being held for \( r \) days, say, and in actor \( a_{i_3} \) being held for \( |\beta_{1,1,2,i_3}| - r \) days, say. Therefore, the minimum cost of this case is \( \min\{c_{i_2}, c_{i_3}\}|\beta_{1,1,2,i_3}| \). However, the minimum cost of this case will vanish when set \( \beta_{1,1,2,i_3} \) is empty because hold days will not be necessarily incurred by actors \( a_{i_2} \) and \( a_{i_3} \).

**Case y-8:** The first day is from \( \beta_{1,1,2,i_3} \) and the last day is from \( \beta_{1,1,2,i_3} \) (Figure 7(c)).
The idle cost \( c_{i_3}|\beta_{1,1,2,i_3} \cup \beta_{1,1,2,i_3}| \) is inevitable to actor \( a_{i_3} \). Consider actors \( a_{i_1} \) and \( a_{i_2} \), and the unscheduled days of set \( \beta_{1,1,2,i_3} \) and set \( \beta_{1,1,2,i_3} \). By Lemma 3.3. the minimum hold cost of actors \( a_{i_1} \) and \( a_{i_2} \) is \( \min\{c_{i_1}|\beta_{1,1,2,i_3}|, c_{i_2}|\beta_{1,1,2,i_3}| \} \). The minimum cost of this case is thus \( \min\{c_{i_1}|\beta_{1,1,2,i_3}|, c_{i_2}|\beta_{1,1,2,i_3}| + c_{i_3}|\beta_{1,1,2,i_3} \cup \beta_{1,1,2,i_3}| \).
Taking the minimum over the ten cases, we have the value $y_{i_1, i_2, i_3}$ as stated in the lemma.

Since the subsets used for the analysis of the x-type cases and those used for the analysis of the y-type cases are disjoint, the lower bound $z_{i_1, i_2, i_3} = x_{i_1, i_2, i_3} + y_{i_1, i_2, i_3}$ on the hold cost incurred by actors $a_{i_1}, a_{i_2}$, and $a_{i_3}$ is established.

The lower bound given in Lemma 4.1 is calculated for three actors. Similar to the development of the lower bound $LB_3(P)$, we can find an optimal (maximum) 3-grouping of all the actors to obtain a tighter bound. For every three actors $a_{i_1}, a_{i_2}$, and $a_{i_3}$, define a cost $z_{i_1, i_2, i_3}$. The Max-3-Grouping($P$) problem is to partition the actors into groups of three such that the total grouping cost is maximized. The discussion of the third lower bound is summarized in the following theorem.

**Theorem 4.2.** Given partial schedule $P$,

$$LB_3(P) = LB_2(P) + \text{Max-3-Grouping}(P)$$

is a lower bound on the optimal solution derived from $P$. □

In combinatorial optimization, there are several NP-hard 3-grouping problems, including 3-Partition and 3-Dimensional Matching. Nevertheless, whether Max-3-Grouping($P$) is NP-hard is unknown. An alternative is to use heuristic algorithms, like greedy selection, to find an approximate solution for 3-Dimensional Matching.

5. **Evaluation of lower bounds.** We present in this section a computational study to assess the merits of the lower bounds. We formulate the Max-3-Grouping problem as an integer program and solve it using the commercial software CPLEX. We use the binary variable $Z_{i,j,k}, i \neq j \neq k$, to indicate if the group of actors $a_i, a_j$, and $a_k$ is accepted or not. That is, if actors $a_i, a_j$, and $a_k$ are grouped to define a
Table 3. Lower bounds subject to outside-in branching scheme.

| Density | lower bound | k=1 | k=3 | k=5 | k=10 |
|---------|-------------|-----|-----|-----|------|
|         | value       | ratio | value       | ratio | value       | ratio | value       | ratio | value       | ratio |
| 0.1     | LB1         | 11,413 | 1.00 | 106,365 | 1.00 | 241,830 | 1.00 | 515,424 | 1.00 |
|         | LB2         | 22,297 | 1.95 | 129,686 | 1.22 | 265,002 | 1.10 | 530,076 | 1.03 |
|         | LB3         | 50,023 | 4.38 | 143,619 | 1.35 | 272,854 | 1.13 | 531,030 | 1.03 |
| 0.2     | LB1         | 43,299 | 1.00 | 282,387 | 1.00 | 462,986 | 1.00 | 723,402 | 1.00 |
|         | LB2         | 81,428 | 1.88 | 324,608 | 1.15 | 496,798 | 1.07 | 727,837 | 1.01 |
|         | LB3         | 112,749 | 2.60 | 333,710 | 1.18 | 498,486 | 1.08 | 727,837 | 1.01 |
| 0.3     | LB1         | 102,531 | 1.00 | 393,930 | 1.00 | 564,109 | 1.00 | 693,117 | 1.00 |
|         | LB2         | 160,744 | 1.57 | 439,086 | 1.11 | 585,516 | 1.04 | 693,716 | 1.00 |
|         | LB3         | 188,908 | 1.84 | 442,229 | 1.12 | 585,516 | 1.04 | 693,716 | 1.00 |

weight $z_{i,j,k}$, as given in Lemma 4.1, then $Z_{i,j,k} = 1$; otherwise $Z_{i,j,k} = 0$.

**IP-3-Grouping:**

Maximize $\sum_{i=1}^{m} \sum_{j=1}^{m} \sum_{k=1}^{m} z_{i,j,k} Z_{i,j,k}$  \hspace{1cm} (4)

subject to

$\sum_{j=1}^{m} \sum_{k=1}^{m} Z_{i,j,k} + \sum_{j=1}^{m} \sum_{k=1}^{m} Z_{j,i,k} + \sum_{j=1}^{m} \sum_{k=1}^{m} Z_{j,k,i} \leq 1$ for all $i$;  \hspace{1cm} (5)

$Z_{i,j,k} \in \{0, 1\}$ for all $i, j, k$.  \hspace{1cm} (6)

The right-hand side of constraints (5) is set to 1 to avoid double or triple counting of each group.

To generate the test instances, we set $m = 50$ actors and $n = 50$ days. We randomly generated the entries of the matrix by a control variable $density$, which was 0.1, 0.2, or 0.3. We then generated a random number for each entry $t_{i,j}$ such that if the random number is greater than $density$, then $t_{i,j} = 0$; otherwise, $t_{i,j} = 1$. We used another control parameter $k \in \{1, 3, 5, 10\}$ to indicate that $k$ days are scheduled in the first $k$ positions and $k$ days are scheduled in the last $k$ positions. This reflects the outside-in branching scheme. In the experiments, for each combination of $k$ and $density$, we generated ten partial schedules in random. We invoked the three lower bounds to compute the lower bound values for the ten instances. The results, averaged over each ten instances, are shown in Table 3. The sub-columns entitled “ratio” contain the ratios of $\frac{LB_2}{LB_1}$ and $\frac{LB_3}{LB_1}$. As evinced from the numerical values, the largest ratios are due to the setting where $k = 1$ and $density = 0.1$, while the lowest ratios are due to the setting where $k = 10$ and $density = 0.3$. This is attributed to the fact that when there are more days scheduled on both sides of the schedule and the actors are more likely to participate in the film, then the hold costs of most actors are fathomed by the first lower bound. Nevertheless, when the information gained from the partial schedules is sparse, i.e., few days are scheduled, the new lower bounds are from 10% to 400% tighter than the first lower bound.

We conducted another set of experiments to examine the effect of adopting the commonly used sequential branching scheme that fills in the front-end positions of a schedule one by one. The computational results are shown in Table 4. The effectiveness of the new lower bounds is more significant for this conventional branching.
Table 4. Lower bounds subject to sequential branching.

| Density | lower bound | \( k = 1 \) | \( k = 3 \) | \( k = 5 \) | \( k = 10 \) |
|---------|-------------|--------------|--------------|--------------|--------------|
|         | value | ratio | value | ratio | value | ratio | value | ratio | value | ratio |
| 0.1     | \( LB_1 \) | 0 | N/A | 7,764 | 1.00 | 22,918 | 1.00 | 87,358 | 1.00 |
|         | \( LB_2 \) | 6,541 | N/A | 26,995 | 3.48 | 51,991 | 2.27 | 127,306 | 1.46 |
|         | \( LB_3 \) | 37,45 | N/A | 49,103 | 6.32 | 67,435 | 2.94 | 132,759 | 1.52 |
| 0.2     | \( LB_1 \) | 0 | N/A | 11,737 | 1.00 | 35,605 | 1.00 | 113,626 | 1.00 |
|         | \( LB_2 \) | 24,446 | N/A | 74,227 | 6.32 | 113,462 | 3.19 | 206,271 | 1.76 |
|         | \( LB_3 \) | 71,501 | N/A | 96,454 | 8.22 | 125,353 | 3.52 | 212,210 | 1.81 |
| 0.3     | \( LB_1 \) | 0 | N/A | 14,769 | 1.00 | 39,563 | 1.00 | 113,626 | 1.00 |
|         | \( LB_2 \) | 44,348 | N/A | 119,872 | 8.12 | 275,254 | 6.96 | 236,553 | 2.08 |
|         | \( LB_3 \) | 95,967 | N/A | 135,891 | 9.20 | 281,540 | 7.12 | 236,553 | 2.08 |

A very interesting phenomenon is that the ratios are higher as density increases. This is especially evident for \( k \in \{1, 3, 5\} \). When there are more 1’s in the scheduled days, more actors will be involved in the calculation of \( LB_2 \), thus increasing the value of \( LB_2 \).

Before closing this section, we note that the new bounds require more computing effort than the previous ones. Therefore, we do not champion the efficiency of constructing the new lower bounds for use in branch-and-bound algorithms but their efficacy in terms of their tightness and applicability in different scenarios.

6. Concluding remarks. We provide three lower bounds on the total hold cost of the talent scheduling problem. The new lower bounds can be deployed to support different branching schemes. The computational study reveals that the new lower bounds are much tighter than the original one. As the zero-one matrix possesses different interpretations and/or applications in other areas, our results have a wide range of applications. For further study, it will be interesting to ascertain the optimality gaps between the proposed lower bounds and optimal solutions. Another direction for further research could be applying the model to treat the problem of optimal linear arrangements of objects that are characterized by several attributes. Objects (days) are individuals or products, and the matrix \( T \) describes if each object possess a certain attribute (actor). A linear arrangement (permutation) describes a spectrum of the concerned objects and its objective value gauges the inappropriateness of the permutation. Extending the model by replacing days with scenes and introducing daily working capacities such that the total length of the scenes included in each single day cannot exceed the capacity, Wang et al. \[13\] incorporated bin packing into the decision. In a similar vein, future research may consider problem settings that closely reflect real-life practice.

Acknowledgments. We are grateful to two anonymous reviewers for their helpful comments on earlier versions of our paper.

REFERENCES

[1] R. M. Adelson, G. Laporte and J. M. Norman, A dynamic programming formulation with diverse applications, Operations Research Quarterly, 27 (1976), 119–121.
[2] M. G. de la Banda, P. J. Stuckey and G. Chu, Solving talent scheduling with dynamic programming INFORMS Journal on Computing, 23 (2011), 120–137.
TALENT HOLD COST MINIMIZATION IN FILM PRODUCTION

[3] T. C. E. Cheng, J. E. Diamond and B. M. T. Lin, Optimal scheduling in film production to minimize talent hold cost, *Journal of Optimization Theory and Applications*, 79 (1993), 479–492.

[4] M. Gendreau, A. Hertz and G. Laporte, A generalized insertion algorithm for the serialization problem, *Mathematical and Computational Modeling*, 19 (1994), 53–59.

[5] R. M. Karp, Mapping the genome: Some combinatorial problems arising in molecular biology, *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing (STOC)*, (1993), 279–285.

[6] G. Laporte, The serialization problem and the travelling salesman problem, *Journal of Computational and Applied Mathematics*, 4 (1978), 259–268.

[7] G. Laporte, Solving a family of permutation problems on 0-1 matrices, *RAIRO (Operations Research)*, 21 (1987), 65–85.

[8] G. Laporte and S. Tailléfer, An efficient interchange procedure for the archaeological serialization problem, *Journal of Archaeological Science*, 14 (1987), 283–289.

[9] B. M. T. Lin, A new branch-and-bound algorithm for the film production problem, *Journal of Ming Chuan College*, 10 (1999), 101–110.

[10] A. L. Nordström and S. Tufekçioğlu, A genetic algorithm for the talent scheduling problem, *Computers and Operations Research*, 21 (1994), 927–940.

[11] R. S. Singleton, *Film Scheduling: Or, How Long Will It Take to Shoot Your Movie?*, Lone Eagle, Los Angeles, U.S.A., 1997.

[12] B. M. Smith, *Constraint Programming in Practice: Scheduling a Rehearsal*, Report APES-67-2003, University of Huddersfield, U.K., 2003.

[13] S. Y. Wang, Y. T. Chuang and B. M. T. Lin, Talent scheduling with daily operating capacities, *Journal of Production and Industrial Engineering*, 33 (2016), 17–31.

[14] M. Wyon, Preparing to perform periodization and dance, *Journal of Dance Medicine and Science*, 14 (2010), 67–72.

Received May 2015; 1st revision August 2015; final revision December 2015.

E-mail address: edwin.cheng@polyu.edu.hk
E-mail address: bmtlin@mail.nctu.edu.tw
E-mail address: lan0903@gmail.com