Integer Programming Techniques for Makespan Minimizing Workforce Assignment Models that Recognize Human Learning

Huan Jin¹, Barrett W. Thomas¹, Mike Hewit¹

^a*Department of Management Sciences, Tippie College of Business, University of Iowa,
Iowa City, IA, 52242*

^b*Information Systems and Operations Management, Loyola University Chicago, Chicago,
IL, 60611*

Abstract

We study optimization techniques for makespan minimizing workforce assignment problems wherein human learning is explicitly modeled. The key challenge in solving these problems is that the learning functions that map experience to worker productivity are usually nonlinear. This paper presents a set of techniques that enable the solution of much larger instances of such problems than seen in the literature to date. The first technique is an exact linear reformulation for the general makespan minimizing workforce assignment models with learning. Next, we introduce a computationally efficient means for generating an initial feasible solution (which our computational experiments indicate is often near-optimal). Finally, we present methods for strengthening the formulation with cover inequalities and a lower bound on the objective function value of the optimal solution. With an extensive computational study we demonstrate the value of these techniques and that large instances can be solved much faster than have previously been solved in the literature. To focus the paper on the presented methodology, we solve a makespan minimizing workforce assignment problem that has few complicating constraints. However, the techniques can be adapted to speed up the solution of most any makespan minimizing workforce assignment problem.

*Corresponding author

Keywords: Integer Programming, Learning Curves, Task Assignment,
Makespan, Non-linear programming

Integer Programming Techniques for Makespan Minimizing Workforce Assignment Models that Recognize Human Learning

Abstract

We study optimization techniques for makespan minimizing workforce assignment problems wherein human learning is explicitly modeled. The key challenge in solving these problems is that the learning functions that map experience to worker productivity are usually nonlinear. This paper presents a set of techniques that enable the solution of much larger instances of such problems than seen in the literature to date. The first technique is an exact linear reformulation for the general makespan minimizing workforce assignment models with learning. Next, we introduce a computationally efficient means for generating an initial feasible solution (which our computational experiments indicate is often near-optimal). Finally, we present methods for strengthening the formulation with cover inequalities and a lower bound on the objective function value of the optimal solution. With an extensive computational study we demonstrate the value of these techniques and that large instances can be solved much faster than have previously been solved in the literature. To focus the paper on the presented methodology, we solve a makespan minimizing workforce assignment problem that has few complicating constraints. However, the techniques can be adapted to speed up the solution of most any makespan minimizing workforce assignment problem.

Keywords: Integer Programming, Learning Curves, Task Assignment, Makespan, Non-linear programming

1. Introduction

Workforce planning is the process by which companies deploy their workforce to complete the tasks of the organization. In making these assignments, it is important to recognize that the assignments that they make in the short term have a longer term impact on the organization’s capacity. In particular, workers learn and become more productive the more experience that they gain.

In this research, we assume a set of heterogeneous workers and a set of unrelated jobs that must be completed over the course of a given horizon. As workers work on a particular job, they become more experienced and thus more productive. The increase in productivity is often referred to as “learning.” We assume that each individual worker has a known learning curve that determines that worker’s productivity. Importantly, we account for the fact that some workers learn faster than others. The objective is to minimize the time required to complete all of the jobs (known as minimizing the makespan). We call this problem the makespan minimizing workforce assignment problem with learning.

The key challenge in solving the makespan minimizing workforce assignment problem with learning is that the learning curves are usually nonlinear. As a result, most work in the literature is limited to solving small-sized problems. This paper presents a set of techniques that enable the solution of much larger instances of such problems. In particular, this paper presents three techniques that are contributions to the literature on solving makespan minimizing workforce assignment problems that explicitly model learning. First, we present an exact linear reformulation for the general makespan minimizing workforce assignment model with learning. While the reformulation technique is adapted from the literature, this paper is the first to apply it to the learning function considered in this paper and in the context of a makespan minimizing workforce assignment problem. Then, we introduce a computationally efficient means for generating an initial feasible solution (which our computational experiments indicate is often near-optimal). We also present methods for strengthening the formulation with cover inequalities and a lower bound on the objective func-

tion value of the optimal solution. With an extensive computational study we demonstrate the value of these techniques. To focus the paper on the techniques we present we solve a makespan minimizing workforce assignment problem that has few complicating constraints. However, the techniques we present can be adapted to speed up the solution of most any makespan minimizing workforce assignment problem. Further, the presented techniques do not depend on the particular learning and forgetting function.

This paper is organized as follows. Section 2 presents the literature review on workforce planning and learning curves and introduces more precisely the problem description. Section 3 first presents a nonlinear formulation of the makespan minimizing workforce assignment problem with learning. The section then introduces the linear reformulation of the problem, a way to generate initial feasible solutions, as well as the inequalities and bound that strengthen the formulation. Section 4 presents our datasets and computational experiments demonstrating the value of the reformulation, the initial solution, cover inequalities, and lower bound. Section 5 offers the conclusions and discusses future avenues of research.

2. Literature Review

In this paper, we focus on scheduling jobs to workers with the goal of minimizing the makespan. Our models account for the fact that workers learn and thus increase productivity as they gain experience.

The most closely related work is that of Corominas et al. (2010) who introduce a piecewise linearization of a learning function for a task assignment model. The largest problem solved has five tasks and four workers and they do so with a two-segment piecewise linear approximation to linearize a concave learning function. Olivella et al. (2013) extend Corominas et al. (2010) to consider due dates and cross-training goals and use a similar solution approach to Corominas et al. (2010). The paper solves problems with up to four workers and eight tasks. Heimerl & Kolisch (2010) addresses a problem similar to Olivella et al.

60 (2013) while also including forgetting and company skill levels. Using a non-
 61 linear programming solver that cannot guarantee optimal solutions, Heimerl &
 62 Kolisch (2010) solve problems with six workers and 20 tasks. In contrast, in the
 63 work presented in this paper, we solve exactly problems with up to 20 workers
 64 and 30 tasks. A comprehensive review of other work that incorporates learning
 65 into workforce planning models can be found in Hewitt et al. (2015).

66 Using a model for assembly-line production first presented in Nembhard &
 67 Norman (2007), Hewitt et al. (2015) introduces a technique for deriving exact
 68 linear reformulations of nonlinear learning functions. The reformulation tech-
 69 nique presented in Hewitt et al. (2015) models nonlinear functions with discrete
 70 domains and ranges as sets of binary and linear variables and constraints. He-
 71 witt et al. (2015) can solve problems up to 20 workers and 40 tasks in less
 72 than an hour. However, Hewitt et al. (2015) considers a different non-linear
 73 model of human learning than we consider in this paper. Thus, while this pa-
 74 per adapts the reformulation technique presented in Hewitt et al. (2015), it
 75 differs in three important ways: (1) We consider a different class of scheduling
 76 problems (makespan minimizing workforce assignment problems), (2) We con-
 77 sider a different quantitative model of how experience translates to proficiency,
 78 and, (3) We provide techniques that significantly reduce the solve time of the
 79 reformulated model.

80 Table 1 summarizes the literature most closely related to that in this pa-
 81 per. The first column cites the paper being summarized and the second notes
 82 from what paper the summarized paper’s model is derived. The third and
 83 fourth columns identify whether or not the model includes cross-training goals
 84 and whether or not the model is nonlinear, respectively. The fifth and sixth
 85 columns (found in the second layer of the table) highlight the solution method
 86 and whether or not the method is a heuristic method, respectively. The final
 87 three columns indicated the largest problem solved in each paper by indicat-
 88 ing the number of workers, tasks and time periods, respectively, of the largest
 89 problem solved.

90 The study of how humans learn has a long history. With the introduction

Table 1 Summary of Related Literature

Publication	Related Models	Cross-Training Goals	Nonlinear Model		
Nembhard & Norman (2007)					
Corominas et al. (2010)	-		✓		
Heimerl & Kolisch (2010)	-	✓	✓		
Olivella et al. (2013)	Corominas et al. (2010)	✓	✓		
Hewitt et al. (2015)	Nembhard & Norman (2007)				
This Paper	Corominas et al. (2010)				

Publication	Solution Approach	Heuristic	Workers	Tasks	Periods
Nembhard & Norman (2007)	Nonlinear Programming		2	4	10
Corominas et al. (2010)	Piecewise Linearization	✓	4	5	20
Heimerl & Kolisch (2010)	Primal-Dual Interior Point	✓	6	4	5
Olivella et al. (2013)	Approximate Convex Piecewise Linearization	✓	4	8	40
Hewitt et al. (2015)	Reformulated Mixed Integer Program		20	40	40
This Paper	Reformulated Mixed Integer Program		20	30	NA

of his “power model,” Wright (1936) is often credited with introducing the first mathematical description of the relationship between experience and productivity. These mathematical descriptions are often called learning curves. To the best of the authors’ knowledge, all of the commonly employed learning curves (Power, exponential, hyperbolic) are nonlinear. Dar-El (2000), Jaber & Sikström (2004), Jaber (2006), and Anzanello & Fogliatto (2011) provide broad and thorough surveys of the subject. Hewitt et al. (2015) provide a review of learning curves in optimization models.

We apply the three-parameter exponential learning curve model first proposed by Bevis et al. (1970). The exact description of the model is given in the next section. However, the methods discussed subsequently in this paper do not require a particular learning and forgetting function, only that experience be measured in discrete units. Examples of other such learning functions include Nembhard & Norman (2007), Sayin & Karabati (2007), Gutjahr et al. (2008), and Heimerl & Kolisch (2010).

106 3. Problem Formulation

107 In this section, we introduce a model for the minimum makespan minimizing
 108 workforce assignment problem with learning. We first present the straightfor-
 109 ward nonlinear formulation and then present the exact linear reformulations. We
 110 also introduce the procedure for generating initial feasible solution, the cover
 111 inequality, and the lower bound to strengthen the reformulation.

112 We first introduce the notation that is used throughout the paper, followed
 113 by a formal problem description and mathematical formulation.

114 Data

- $\mathcal{W}$ Set of workers, $i = 1, \dots, |\mathcal{W}|$.
- $\mathcal{J}$ Set of jobs, $j = 1, \dots, |\mathcal{J}|$.
- $\mathcal{T}$ Set of periods in the horizon, $t = 1, \dots, |\mathcal{T}|$.
- v_j Volume of work required by job j , measured in units of standard work
 time, typically hours (standard work time units) of work when the
 work is carried out by an experienced worker, $j \in \mathcal{J}$
- 115 p_{ij} Initial experience of worker i on task j , in the same measurement of
 the job volume, $i \in \mathcal{W}, j \in \mathcal{J}$.
- r_{ij} Individual learning rate of worker i on task j , $i \in \mathcal{W}, j \in \mathcal{J}$.
- K_{ij} The maximum worker i 's performance on task j when the learning
 process is concluded, given in the number of units produced per time
 period, $i \in \mathcal{W}, j \in \mathcal{J}$.

116 Variables

- T_{max} Completion time of the last task to be finished.
- x_{ijt} Binary variable that indicates whether task j is done by worker i in
 a period of time t , $i \in \mathcal{W}, j \in \mathcal{J}, t \in \mathcal{T}$.
- 117 c_{ijt} The accumulated experience done by worker i for task j at the end
 of period t , measured in units of time periods performed by i on j by
 time t , $i \in \mathcal{W}, j \in \mathcal{J}, t \in \mathcal{T}$.
- φ_{ijt} Productivity of job j that worker i is expected to do in period t , $i \in$
 $\mathcal{W}, j \in \mathcal{J}, t \in \mathcal{T}$.

118 We assume that a set $\mathcal{W}$ of workers works the jobs in $\mathcal{J}$ in a finite planning
 119 horizon $\mathcal{T}$. The completion of each job j in $\mathcal{J}$ requires a volume of work v_j . As
 120 in Corominas et al. (2010), we assume that the volume of work is measured in
 121 units of standard work time, typically hours (standard work time units) of work
 122 when the work is carried out by an experienced worker.

123 Each worker i in $\mathcal{W}$ has a learning function on each job j in $\mathcal{J}$. We let c_{ijt}
 124 be the accumulated experience of worker i on job j before period t . It is defined
 125 as the sum of the work time units done by worker i for job j by time t (see
 126 Equation (6)). The initial accumulated experience c_{ij1} is 0. Then, the amount
 127 of work completed in a unit of time by worker i on job j during time t will be
 128 a function of c_{ijt} :

$$\varphi_{ijt} = K_{ij} \left[1 - e^{-(c_{ijt} + p_{ij})/r_{ij}} \right], \quad (1)$$

129 where K_{ij} is an asymptotic parameter specifying the maximum productivity
 130 that can be achieved by worker i on job j , learning rate r_{ij} is the time needed
 131 to reach 63% of the asymptote's production rate. The larger the value of r is, the
 132 slower is the rate of improvement in performance. (see Figure 1). Parameter p_{ij}
 133 corresponds to a worker's prior experience evaluated in units of standard work
 134 time units-typically hours of standard work time. The objective is to minimize
 135 the "makespan", or the completion of the last job, denoted as T_{max} .

136 3.1. Nonlinear Formulation

137 Given the above assumptions and definitions, we present the following Makespan
 138 Minimizing Workforce Assignment with Learning (MwL) model. A solution to
 139 this model prescribes an assignment for each worker i in $\mathcal{W}$ to a job j in $\mathcal{J}$ in
 140 period t in $\mathcal{T}$. We denote these assignments by the binary variables x_{ijt} . The
 141 MwL is:

Figure 1 Three-parameter Exponential Learning Curve by Learning Rate

$$\text{Min } T_{max} \quad (2)$$

$$\text{S.t. } tx_{ijt} \leq T_{max} \quad i \in W, j \in J, t \in \mathcal{T} \quad (3)$$

$$\sum_{j \in J} x_{ijt} \leq 1, \quad i \in W, t \in \mathcal{T} \quad (4)$$

$$\sum_{i \in W} x_{ijt} \leq 1, \quad j \in J, t \in \mathcal{T} \quad (5)$$

$$\text{MwL} \quad c_{ijt} = \sum_{k=1}^{t-1} x_{ijk}, \quad i \in W, j \in J, t = 2, \dots, T \quad (6)$$

$$\varphi_{ijt} = K_{ij} \left(1 - e^{-(c_{ijt} + p_{ij})/r_{ij}} \right), i \in W, j \in J, t \in \mathcal{T} \quad (7)$$

$$\sum_{i \in W} \sum_{t \in \mathcal{T}} x_{ijt} \varphi_{ijt} \geq v_j, \quad j \in J \quad (8)$$

$$x_{ijt} \in \{0, 1\} \quad i \in W, j \in J, t \in \mathcal{T} \quad (9)$$

Equation (2) is the objective minimizing the makespan T_{max} . Constraints (3) define the makespan T_{max} to be the completion time of the last job. Constraints (4) enforce that a worker cannot work more than one job in a period. Constraints (5) limit only one worker can work on a job in any period. Constraints (6) define c_{ijt} as the accumulated experience of worker i for job j before t . Then, Constraints (7) uses this accumulated experience value to compute productivity during period t . Constraints (8) enforce that the volume of work as-

sociated with job j , v_j , be performed by the end of the horizon. Constraints (9) define the domain of the variables x_{ijt} .

We focus on a makespan minimizing workforce assignment model with few complicating constraints in this paper to highlight the general applicability of the solution techniques we present next. The MwL can be modified to handle many constraints such as adding ready times, due dates, and precedence constraints to jobs. However, the solution techniques we discuss next can still be used for each of these modified models.

3.2. Exact Linear Reformulation of MwL

In preliminary work, we ran experiments in which we sought to solve the nonlinear model MwL using the nonlinear solver Couenne. Couenne reached its node limit and could not return an integer solution even for a problem with two workers and two jobs with a time limit of 1800 seconds. As a result, we turn to other means of solving the instances of the proposed problem.

This section presents a reformulation that linearizes MwL, leaving a model that is easier to solve. We call this reformulation the Makespan Minimizing Workforce Assignment with Reformulated Learning (MwRL) model. Hewitt et al. (2015) introduce an analogous technique for a serial production line problem with human learning and prove that the reformulation is exact for the problem studied in their paper. We introduce a proof for the reformulation in the context of makespan minimizing workforce assignment. The reformulation of MwL relies on the fact that job assignments, and thus the accumulated experience, are measured in discrete units. As a result, the nonlinear learning function can be linearized by enumerating its possible range values and associating a binary variable with each one. The result is that an optimal solution to the MwRL model is also an optimal solution to the MwL model.

We let $\hat{\varphi}_{ijt}^l$ be the productivity when worker i has l accumulated experience doing job j up to time t . Note that, given i, j, t , and l , we can compute $\hat{\varphi}_{ijt}^l$ *a priori* as $\hat{\varphi}_{ijt}^l = K_{ij} (1 - e^{-(l+p_{ij})/r_{ij}})$ (see Equation (7)) and thus treat it as a parameter to an optimization model. Recalling the definition of c_{ijt} in

179 Constraints (6), a worker can gain at most $t - 1$ units of accumulated experience on a job by t . Thus, there are at most $t - 1$ possible values for each
 180 worker and each job for period t . We define binary variables z_{ijt}^l associated
 181 with each parameter $\hat{\varphi}_{ijt}^l$, where l is the accumulated experience of worker i up
 182 to time t on job j . With these new parameters and variables, the nonlinear
 183 Constraints (7) can be replaced by (10)-(12). Constraints (10) assign the productivity φ_{ijt} the appropriate pre-defined parameter value $\hat{\varphi}_{ijt}^l$ when $z_{ijt}^l = 1$,
 184 while Constraints (11) ensure that the l in the assigned pre-defined $\hat{\varphi}_{ijt}^l$ is equal
 185 to the accumulated experience c_{ijt} . Constraints (12) ensure that each individual
 186 is assigned only one productivity in each period. Given the redefinition
 187 of φ_{ijt} in Constraints (10), nonlinear Constraints (8) can be replaced by linear
 188 Constraints (13)-(14), where M_{ijt} is a big number. Specifically, we use
 189 $M_{ijt} = \hat{\varphi}_{ijt}^{t-1} = \max\{\hat{\varphi}_{ijt}^l, l = 0, 1, \dots, t-1\}$; the maximum production rate
 190 worker i can achieve on job j in period t . We choose this value for M_{ijt} as it
 191 does not render any feasible solutions infeasible.

$$\text{Min } T_{max}$$

$$\text{S.t } (3) - (6), (9)$$

$$\text{MwRL} \quad \varphi_{ijt} = \sum_{l=0}^{t-1} \hat{\varphi}_{ijt}^l z_{ijt}^l, \quad \forall i \in W, j \in J, t \in \mathcal{T}; \quad (10)$$

$$\sum_{l=0}^{t-1} l z_{ijt}^l \leq c_{ijt}, \quad \forall i \in W, j \in J, \quad t \in \mathcal{T}, \quad (11)$$

$$\sum_{l=0}^t z_{ijt}^l \leq 1, \quad \forall i \in W, j \in J, \quad t \in \mathcal{T}, \quad (12)$$

$$\varphi_{ijt} \leq M_{ijt} x_{ijt}, \quad \forall i \in W, \quad j \in J, t \in \mathcal{T} \quad (13)$$

$$\sum_{i=1}^W \sum_{t=1}^T \varphi_{ijt} \geq v_j, \quad \forall j \in J, \quad (14)$$

$$z_{ijt}^l \in \{0, 1\} \quad i \in W, j \in J, t \in \mathcal{T}, l = 0, \dots, t-1. \quad (15)$$

194 **Proposition 3.1.** *MwRL is an exact linear reformulation of MwL*

195 *Proof.* First, note that the MwRL and MwL share the same objective function

and Constraints (3)-(6) and (9). Further, Constraints (8) are equivalent to Constraints (13)-(14) that guarantee the volume of all jobs are finished. Second, Constraints (7) in MwL are equivalent to Constraints (10)-(12) in MwRL. To see this, note that the variable φ_{ijt} is a function of c_{ijt} . The variable c_{ijt} has at most $t - 1$ possible values given the definition of c_{ijt} in Constraints (6). Thus, we can precompute $\hat{\varphi}_{ijt}^l, l = 0, \dots, t - 1$ for accumulated experience values of $0, \dots, t - 1$. For each i and j , Constraints (10) associate the appropriate $\hat{\varphi}_{ijt}^l$ with the accumulated experience c_{ijt} (by Constraints (11)). Constraints (12) enforce that only one productivity is assigned to a worker in one period.

Thus, because the MwRL has the same objective function as the MwL and either shares the same constraints or replaces constraints in the MwL with equivalent linearized constraints, the MwRL is an exact linear reformulation of MwL. $\square$

By Proposition (3.1), the MwRL is an exact reformulation of the MwL, but lacks the nonlinearities. While this linearization does make it possible to solve larger instances than are possible with the nonlinear formulation, the reformulation does require additional variables and constraints. When reforming MwL to MwRL, we add $|\mathcal{W}||\mathcal{J}| \sum_{t=1}^T t$ extra binary variables z_{ijt}^l , and $|\mathcal{W}||\mathcal{J}| \sum_{t=1}^T t$ extra parameters $\hat{\varphi}_{ijt}^t$. We have also added $3|\mathcal{W}||\mathcal{J}||\mathcal{T}|$ extra linear constraints. These extra variables and constraints mean that large instances of the MwRL are still challenging for commercial branch and bound solvers. In the remainder of this section, we discuss ways in which the computational burden of the MwRL can be reduced. Section 3.3 demonstrates how we generate initial feasible solutions to MwRL, while Section 3.4 introduces valid inequalities and a lower bound on the objective function value of MwRL.

3.3. Initial Feasible Solution

We note that MwRL requires definition of the set $\mathcal{T}$, and the instantiation of $\mathcal{T}$ requires knowing an upper bound on T_{max} . We generate this bound through a high quality initial feasible solution (IFS). We also use this IFS to improve

our branch-and-bound approach. Knowledge of the objective function value of a high quality IFS to an integer program at the start of the solution process can reduce solution time in multiple ways. First, this upper bound on the optimal value of the optimization problem can be used during preprocessing routines to further reduce the domains of variables. Second, during branch and bound, this upper bound can be used to fathom nodes for which the lower bound is too great. Finally, with an IFS, the solver does not need to spend time finding its own initial feasible solution.

To generate the IFS, we solve a makespan minimizing workforce assignment problem that recognizes human learning (like the MwRL) but does not allow jobs to be split. Specifically, a job can only be assigned to one worker, and once a worker starts working a job, he or she will complete it. We call the resulting problem the Makespan Minimizing Workforce Assignment Problem with Learning and No Splitting (MwLNS). Note that the MwLNS is a restriction of the MwRL; all the assignments that are feasible for the MwLNS are feasible for the MwRL, but the converse is not true. As such, solutions to the MwLNS can be mapped to solutions to the MwRL.

The MwLNS includes the binary decision variables x'_{ij} that indicate whether job j is assigned to worker i . Given the restriction that an individual complete a job once they begin it, we can calculate *a priori* the minimum number of time units the worker would need to complete the job, which we denote by pt_{ij} . Specifically, we calculate pt_{ij} by solving the optimization problem given in equation (16). We can solve this optimization problem through iteration, increasing the accumulated experience c_{ij} (starting at 0) until the productivity reaches or exceeds the volume of job j . Note that the fact that an individual learns is embedded in this optimization problem.

$$pt_{ij} = \min_l \left\{ \sum_{c_{ij}=0}^l K_{ij} \left(1 - e^{-(l+p_{ij})/r_{ij}} \right) \geq v_j, l \in Z^+ \right\} \quad (16)$$

Using these pt_{ij} values, we formulate and solve the MwLNS to find an initial solution to the MwRL and a value for T_{max} :

$$Min \quad T_{max}^{MwLNS}$$

$$S.t \quad \sum_{i \in W} x'_{ij} = 1, \quad j \in J \quad (17)$$

$$(MwLNS) \quad \sum_{j \in J} x'_{ij} pt_{ij} \leq T_{max}, \quad i \in W \quad (18)$$

$$x'_{ij} \in \{0, 1\} \quad i \in W, j \in J, \quad (19)$$

Commercial solvers can often solve instances of the MwLNS model almost instantaneously. This solution can then be transformed into a solution to the MwRL with a procedure we next describe with an example. Consider a case in which there are two workers and four jobs. Table 2 presents the processing time pt_{ij} for each worker i on each job i , accounting for the learning that takes place as worker i performs job j . Table 2 also present the assignments from an optimal solution of this instance of the MwLNS.

The solution shows that worker 1 is assigned to job 2 with a processing time of 2 periods and job 3 with a processing time of 3 periods. Worker 2 performs job 1, which takes 1 period and job 4, which takes 4 periods. The objective value (makespan) of this example is 5 periods.

We create a solution to the MwRL by ordering the jobs performed by worker i in ascending order of processing time pt_{ij} (see Table 3). This solution also yields an objective value of 5 periods and does not violate any constraints of the MwRL. We can then feed this transformed solution to an optimization solver as the starting point for its search for an optimal solution to the MwRL

3.4. Strengthening the Formulation

While overcoming the nonlinearities of the learning curve improves the tractability of the model, we can also strengthen the formulation using valid inequalities and a lower bound on the objective function value of the optimal solution. By strengthening the formulation, a mixed integer programming solver is likely to require much less time to solve instances of the problem.

pt_{ij}	Job 1	Job 2	Job 3	Job 4
Worker 1	2	2	3	5
Worker 2	1	3	4	4
x'_{ij}	Job 1	Job 2	Job 3	Job 4
Worker 1	0	1	1	0
Worker 2	1	0	0	1

Table 2 Parameters and Solutions of MwLNS for an Example of two workers and four jobs

x_{ijt}		Period 1	Period 2	Period 3	Period 4	Period 5
Worker 1	Job 2	1	1	0	0	0
	Job 3	0	0	1	1	1
Worker 2	Job 1	1	0	0	0	0
	Job 4	0	1	1	1	1

Table 3 Transform into Solutions x_{ijt} of MwRL

3.4.1. Valid Inequality

First, we present a cover inequality that is added to MwRL to strengthen the bound produced by its linear relaxation. Consider Constraint (8) and derive valid inequality for MwRL. We can weaken the constraint $\sum_{i \in W} \sum_{t \in \mathcal{T}} x_{ijt} \varphi_{ijt} \geq v_j$ by replacing the variable φ_{ijt} with the value $\bar{\varphi}_j = \max_i \hat{\varphi}_{ijT}^{T-1}$, where $\hat{\varphi}_{ijT}^{T-1} = \max\{\hat{\varphi}_{ijT}^l, l = 0, 1, \dots, T-1\}$ is the maximum production rate of a worker i on a job j over the entire time horizon. The resulting constraint is:

$$\sum_{i \in W} \sum_{t \in \mathcal{T}} \bar{\varphi}_j x_{ijt} \geq v_j. \quad (20)$$

Dividing both sides by $\bar{\varphi}_j$ and round the right hand side to the ceiling integer since the sum of the binary variables is integral. As such, we have the cover inequality:

$$\sum_{i \in W} \sum_{t \in \mathcal{T}} x_{ijt} \geq \left\lceil \frac{v_j}{\bar{\varphi}_j} \right\rceil, \quad \forall j \in J. \quad (21)$$

285 3.4.2. Lower Bound

286 One can also strengthen a formulation by adding a lower bound on the opti-
 287 mal objective value. To derive such a bound, we solve an integer program that is
 288 similar in form to MwL, but much simpler to solve. Assume all the workers have
 289 the maximum possible productivity on all jobs at all time periods $\hat{\varphi}_{ijt}^{t-1}$. We re-
 290 move all the nonlinearities in MwL and call the following program the Makespan
 291 Minimizing Workforce Assignment with Maximum Learning (MwML) model.

$$\begin{aligned}
 & \text{Min} \quad T_{max}^{MwML} \\
 & \text{S.t} \quad (3) - (5), (9) \\
 & (MwML) \quad \sum_{i \in W} \sum_{t \in \mathcal{T}} x_{ijt} \hat{\varphi}_{ijt}^{t-1} \geq v_j, \quad \forall j \in J
 \end{aligned} \tag{22}$$

292 The MwML is a relaxation of MwL (some constraints are removed, some are
 293 weakened by replacing variables with their maximum values). We can solve the
 294 MwML prior to solving the MwRL to provide a global bound on the optimal
 295 objective function value. We add the constraint:

$$T_{max}^{MwML} \leq T_{max} \tag{23}$$

296 to the MwRL. We can generate additional bounds during the execution of the
 297 branch and bound by solving the MwML at different nodes in the branch-and-
 298 bound tree with some variables in the MwML fixed to values determined by
 299 branching decisions.

300 4. Computational Experiments and Results

301 In this section, we present the results of computational experiments that we
 302 performed to test the effectiveness of our formulations and techniques.

303 4.1. Experimental setting

304 We generate 11 classes of instances, each class differentiated by the number
 305 of workers and tasks. Our goal in generating the sets is to test to limits of the

proposed solution methodology rather than to recreate instances reflecting any particular environment. The classes range from 5 workers, 10 tasks up to 20 workers, 30 tasks. To give perspective on the sizes of these instances, recall that Corominas et al. (2010) solve a close variant of the MwRL in which instances of only five tasks and four workers were solved via an approximation (a piecewise linearization) scheme. We detail the instance sizes, with respect to the number of workers, $|\mathcal{W}|$, and the number of jobs, $|\mathcal{J}|$ in Table 4.

For each class, we generate nine cases for a total of 99 instances. Each case is defined by the learning traits of the workers (asymptote parameter K , prior experience p , and learning rate r) and the job volumes. We sample the learning traits data based on the empirical data found in Mazur & Hastie (1978).

$ \mathcal{W} $	$ \mathcal{J} $
5	10, 15
10	10, 15, 20
15	15, 20, 25
20	20, 25, 30

Table 4 Instance sizes

Mazur & Hastie (1978) collect performance-related results from various experimental settings, including students writing letters and factory workers making cigars. In our dataset, the asymptote parameter is generated from the interval $[8, 10]$, and the prior experience is assumed to be 0.5, which is the minimum experience for a worker to start on a job. The learning rate is classified as “fast”, “medium”, or “slow,” with the sampling intervals for each classification given in Table 5. Figure 2 illustrates these curves at the midpoint of their respective range. The job volumes also define the problem and a job volume is classified as “short”, “medium”, or “long” job with the integer values in intervals in Table 5.

For each combination of $|\mathcal{W}|, |\mathcal{J}|$, nine cases are generated, with the cases varying according to worker learning rates and job volumes in Table 6. For example, all workers in case 1 are fast learners whose learning rates are randomly sampled from $[0.5, 1]$, while the job volumes are randomly drawn from the integers in $[|\mathcal{W}|/|\mathcal{J}|, 5|\mathcal{W}|/|\mathcal{J}|]$. Other cases are generated similarly. Overall, these datasets are chosen to test the computational effectiveness of the reformulations and other model enhancements and not to necessarily reflect the

	Notation	Settings		
Asymptote	K	[8, 10]		
Prior experience	p	0.5		
		Fast	Medium	Slow
Learning rate	r	[0.5, 1]	[2, 4]	[5, 8]
		Short	Medium	Long
Volume of work	v	$[\mathcal{W} / \mathcal{J} , 5 \mathcal{W} / \mathcal{J}]$	$[6 \mathcal{W} / \mathcal{J} , 10 \mathcal{W} / \mathcal{J}]$	$[11 \mathcal{W} / \mathcal{J} , 15 \mathcal{W} / \mathcal{J}]$

Table 5 Instance design

337 settings found in any particular practical setting. The datasets are available
338 from http://ir.uiowa.edu/tippie_pubs/66/.

339

Figure 2 Learning curves with different learning rates

Table 6 Instance cases

341 In the experiments, we seek to understand the computational performance
342 of the integer programs and the proposed speed-up techniques. First, we seek
343 to understand to what degree the solvability of MwRL scales with respect to
344 the number of workers and jobs. Next, we seek to understand whether the solve
345 time for an instance of the MwRL is related to the case used to create it. Then,
346 we seek to understand whether the speed-up techniques have an impact.

347 All the experiments presented in the next section were performed on a com-

puter cluster where each node has 32-64 cores with AMD Opteron 2.2 GHz or
 AMD Interlagos 2.6 bulldozer processors and 128-256 GB of memory. All in-
 stances were solved using Gurobi version 5.2, and all Gurobi parameters were
 left at their default values, other than the time limit, which was set to 600
 seconds.

4.2. Computational Results

We first focus on the computational tractability of the MwRL. All times
 reported are in seconds. Because the makespan values for these instances are
 often relatively small (less than 20), we report absolute gaps as opposed to
 relative gaps. As such, we report in Table 7 the average gap at termination
 for each instance of the MwRL that Gurobi tries to solve. An entry of “0.00”
 indicates that Gurobi was able to solve the instance. Similarly, we report in
 Table 8 the time Gurobi requires to reach termination, either because it solved
 the instance or it reached the 600 seconds time limit.

First, we observe that Gurobi solves 41.41% of the instances (41 out of 99).
 We next observe that it is not the size of the instance that impacts Gurobi’s
 ability to solve it, but the case used to generate it. This can also be seen in
 the following figures. Figure 3 presents the average time Gurobi requires for
 all instances of a given size and Figure 4 the average time for all instances of a
 given case. We see that runtimes are fairly constant across instance sizes, but
 differ greatly by case. Gurobi can (relatively) easily solve instances from Cases
 1, 2, 3, 4, and 7. Examining the results further, we see that Gurobi can quickly
 solve instances that either have workers that possess a fast learning rate or jobs
 that have a short volumes.

$ \mathcal{W} - \mathcal{J} $	Case								
	1	2	3	4	5	6	7	8	9
5-10	0.00	0.00	1.00	0.00	2.00	2.00	0.00	2.00	5.00
5-15	0.00	0.00	0.00	0.00	4.00	3.00	1.00	5.00	0.00
10-10	0.00	0.00	0.00	0.00	1.00	1.00	0.00	1.00	2.00
10-15	0.00	0.00	0.00	0.00	1.00	3.00	0.00	2.00	0.00
10-20	0.00	0.00	1.00	0.00	3.00	5.00	1.00	7.00	0.00
15-15	0.00	0.00	0.00	0.00	1.00	5.00	0.00	4.00	0.00
15-20	0.00	0.00	0.00	0.00	1.00	3.00	1.00	4.00	0.00
15-25	0.00	0.00	2.00	1.00	5.00	7.00	1.00	5.00	0.00
20-20	0.00	0.00	0.00	0.00	2.00	0.00	1.00	7.00	0.00
20-25	0.00	1.00	1.00	0.00	1.00	6.00	1.00	5.00	0.00
20-30	0.00	0.00	1.00	0.00	2.00	6.00	1.00	0.00	0.00
Average	0.00	0.09	0.55	0.09	2.09	3.73	0.64	3.82	0.64

Table 7 Abs. gaps when solving MwRL

$ \mathcal{W} - \mathcal{J} $	Case								
	1	2	3	4	5	6	7	8	9
5-10	0.29	8.11	600.00	4.12	600.00	600.00	8.45	600.00	600.00
5-15	6.30	6.76	7.72	0.97	600.00	600.00	600.00	600.00	600.00
10-10	0.38	16.35	28.61	18.70	600.00	600.00	274.72	600.00	600.00
10-15	9.01	23.06	111.25	15.62	600.00	600.00	463.55	600.00	600.00
10-20	14.01	18.67	600.00	17.38	600.00	600.00	600.00	600.00	600.00
15-15	1.97	20.60	80.16	11.00	600.00	600.00	331.51	600.00	600.00
15-20	6.73	14.23	56.78	9.81	600.00	600.00	600.00	600.00	600.00
15-25	11.75	29.55	600.00	600.00	600.00	600.00	600.00	600.00	600.00
20-20	4.65	40.01	129.53	39.26	600.00	600.00	600.00	600.00	600.00
20-25	7.14	600.00	600.00	21.97	600.00	600.00	600.00	600.00	600.00
20-30	11.91	63.32	600.00	27.17	600.00	600.00	600.00	600.00	600.00
Average	6.74	76.42	310.37	69.64	600.00	600.00	479.84	600.00	600.00

Table 8 Times when solving MwRL

Figure 3 Average run-time for instances by $|\mathcal{W}| - |\mathcal{J}|$

Figure 4 Average run-time for instances of a given case

We next turn our attention to the impact of strengthening the formulation (cover inequalities and lower bound) and the IFS proposed earlier. Table 9 indicates for each method name which techniques were used. We first display in Figure 5 the average runtimes when the cover inequalities are and are not used, by case. We see that using the cover inequalities is quite effective as they decrease the runtime in every case. On average, the runtime with the cover inequalities is 74.90% of the runtime than without. Similarly, while 41.41% of the instances were solved within the 600 second time limit when the cover inequalities are not used, 56.57% of instances are solved when it is. On the other hand, Figure 6 indicates that using the lower bound as well as the cover inequalities does not reduce run-times.

Method	Cover ineq.	Lower bound	IFS
MwRL-Cover	Y	N	N
MwRL-Bound	Y	Y	N
MwRL-IFS	Y	N	Y
MwRL-Bound-IFS	Y	Y	Y

Table 9 Method configuration

Figure 5 Average run-time for instances of a given case when using Cover-inequality

Figure 6 Average run-time for instances of a given case when using Cover-inequality and Lower bound

389 We next consider whether there is
 390 an improvement when an IFS is used
 391 along with the cover inequalities. Fig-
 392 ure 7 compares solve times when an
 393 IFS is and is not used along with the
 394 cover inequalities. We see that by
 395 providing an IFS to Gurobi, runtime
 396 is reduced significantly. To be precise,
 397 averaged over all instances, the run
 398 time when an IFS and the cover in-
 399 equalities is used is 42.09% of the run
 400 time when only the cover inequalities
 401 are used. Similarly, while using the
 402 cover inequalities increases the percentage of instances solved to 56.57%, using
 403 an IFS in addition to the cover inequalities increases that percentage to 80.81%.
 404 We also note that the pattern in the average runtime by case is the same when
 405 an IFS is used as when it is not. Both approaches still struggle the most with
 406 instances from Cases 6, 8, and 9. Not surprisingly those are the cases where the
 407 fewest instances are solved for both MwRL-Cover and MwRL-IFS.

Figure 7 Average run-time for instances of a given case when using Cover-inequality and IFS

We next return to considering the use of a lower bound, and report by method and case the average of the absolute gap reported at termination. We note that using the lower bound in conjunction with the IFS has little to no impact on runtime. However, comparing the lines MwRL-IFS and MwRL-Bound-IFS we see that using both the lower bound and the IFS leads to the (provably) highest-quality solution at termination.

Method	Case									Avg.
	1	2	3	4	5	6	7	8	9	
MwRL	0.00	0.09	0.55	0.09	2.09	4.10	0.64	4.20	3.50	1.45
MwRL-Cover	0.00	0.00	0.00	0.00	1.18	2.73	0.27	2.57	2.75	0.85
MwRL-Bound	0.00	0.00	0.00	0.00	0.91	2.27	0.27	2.70	3.25	0.86
MwRL-IFS	0.00	0.00	0.00	0.00	0.18	0.73	0.18	0.55	1.45	0.34
MwRL-Bound-IFS	0.00	0.00	0.00	0.00	0.18	0.64	0.18	0.45	0.82	0.25

Table 10 Average absolute gap by method and case

We next seek to understand why using an IFS is so effective. To that effect, Table 11 reports, averaged over all instances for a given case, the proportion of the number of binary variables that are in the MwRL-Bound-IFS formulation after Gurobi performs preprocessing to the number of binary variables that are in the MwRL formulation after preprocessing. With the IFS, Gurobi is able to preprocess (fix to 0 or 1) many of the binary variables that are in the original MwRL formulation. Not surprisingly, the cases (6, 8, and 9) that we have observed to be the hardest are the ones where Gurobi is able to preprocess the fewest numbers of variables.

	Case								
	1	2	3	4	5	6	7	8	9
% binary var	15.04	34.38	57.95	31.84	58.47	75.74	55.94	76.70	87.96

Table 11 Fraction of MwRL variables in MwRL-Bound-IFS formulation after preprocessing

	Case								
	1	2	3	4	5	6	7	8	9
Abs. gap	0.00	0.00	0.36	0.00	0.18	0.64	0.64	0.45	0.82

Table 12 Average absolute gap between value of initial solution and final lower bound

423 Similarly, Table 12 presents the absolute gap between the objective function
424 value of the IFS and the lower bound reported by Gurobi at termination. We
425 then average these gaps over all instances created with the same case. Not
426 surprisingly, we see a fairly clear relationship between the quality of the IFS
427 and Gurobi’s ability to preprocess variables; the better the quality of the IFS,
428 the more variables Gurobi is able to preprocess.

429 The quality of the IFS is also related to the case used to generate the instance.
430 In particular, for instances where the learning rate is Fast and the job volume
431 is Short or Medium, the IFS is optimal. Similarly, when the learning rate is
432 Medium and the job volume is Short, the IFS is optimal. In a sense, this is not
433 surprising. The IFS is generated by solving a restriction of the MwL in which
434 each job is assigned to exactly one worker. This may be less of a restriction
435 when workers learn very quickly or the job has a Short volume. In both cases,
436 the job can be completed quickly enough that there is no reason for a worker
437 not to finish it.

438 Finally, recalling that the largest instances of models of this sort that have
439 been solved in the literature (and that was with a piecewise-linear approximation
440 scheme) consisted of five workers and four jobs, we repeat Tables 7 and 8 only for
441 when MwRL-Bound-IFS is solved (Tables 13 and 14). Table 14 shows that all
442 instances up to 10 workers and 10 jobs are solved instantaneously, and it is not
443 until instances with 15 workers that the time limit of 600 seconds is consistently
444 hit. However, again it depends on the case. For the cases in which workers learn
445 at a fast rate, instances of all sizes are solved in seconds. Yet, from Table 13, we
446 see that, even when the time limit is hit, the solutions found at that point are
447 typically of high quality as they are only two units away from the best known
448 bound.

449 5. Conclusions and future work

450 In this paper, we study optimization techniques for makespan minimizing
451 workforce assignment problems wherein human learning is explicitly modeled.

$ \mathcal{W} - \mathcal{J} $	Case								
	1	2	3	4	5	6	7	8	9
5-10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5-15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10-10	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
10-15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00
10-20	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
15-15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15-20	0.00	0.00	0.00	0.00	1.00	3.00	1.00	1.00	2.00
15-25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00
20-20	0.00	0.00	0.00	0.00	1.00	2.00	0.00	0.00	0.00
20-25	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	2.00
20-30	0.00	0.00	0.00	0.00	1.00	3.00	1.00	1.00	2.00
Average	0.00	0.00	0.00	0.00	0.27	1.00	0.27	0.45	0.82

Table 13 Abs. gaps when solving MwRL-Bound-IFS

$ \mathcal{W} - \mathcal{J} $	Case								
	1	2	3	4	5	6	7	8	9
5-10	0.06	1.15	0.13	0.51	0.10	0.13	1.87	0.28	0.31
5-15	0.56	0.52	1.21	2.05	0.34	0.45	0.18	0.23	0.29
10-10	1.27	0.11	0.19	1.12	0.18	0.18	0.20	0.19	0.22
10-15	0.23	0.17	6.10	0.29	0.40	600.00	5.96	600.00	600.00
10-20	1.00	1.01	0.25	0.92	0.41	0.69	4.71	1.23	0.61
15-15	0.26	0.55	0.68	0.24	0.26	0.32	0.37	0.52	0.29
15-20	0.12	0.27	6.15	0.85	600.00	600.00	5.20	600.00	600.00
15-25	0.34	0.35	1.47	0.45	0.86	600.00	600.00	600.00	600.00
20-20	0.26	0.34	0.46	0.60	0.69	0.84	1.15	0.36	0.52
20-25	0.25	0.55	6.71	0.63	1.76	600.00	600.00	600.00	600.00
20-30	0.74	0.78	31.61	0.70	600.00	600.00	88.25	600.00	600.00
Average	0.46	0.53	5.00	0.76	109.55	272.96	118.90	272.98	272.93

Table 14 Times when solving MwRL-Bound-IFS

452 In contrast to the commonly-studied problem in which a job is only performed
 453 by one individual and hence learning can be encoded in the parameters of the
 454 problem, we instead study problems wherein learning must be explicitly repre-
 455 sented within the optimization model. To date, the challenge with such models
 456 has come from the fact that quantitative models of how humans learn are non-
 457 linear, leaving optimization problems that are computationally intensive to solve
 458 for all but the smallest instances.

459 To overcome the nonlinearity, we use an exact reformulation technique which
 460 replaces the nonlinear function used to model learning with binary variables and
 461 linear constraints. As such we transform the nonlinear optimization problem to
 462 a mixed integer program (MIP), a type of optimization problem that commercial
 463 solvers are capable of solving for many instance sizes. We then develop speed-

up techniques for this MIP. Namely, we derive a procedure for creating an initial feasible solution for the MIP, a valid inequality for the formulation, and a procedure for producing a lower bound on the optimal value of the MIP.

With an extensive computational study, we show that our techniques can solve larger instances in a much faster speed than has been seen in the literature. We also illustrate the effectiveness of the proposed speed-up techniques as well as analyze why they are effective. Finally, we present computational results that show that our new formulation can often be solved in mere seconds. Ultimately, we believe this paper sets the standard, computationally-speaking, for how one solves makespan minimizing workforce assignment problems that explicitly recognize human learning.

There are many human learning activities that are not captured in the problems we study. For example, psychologists have claimed that, by training, workers not only gain experience but can also fundamentally change their learning rates. Also, our problem assumes that jobs are distinct from a learning perspective. In other words, performing one job does not improve the productivity of a worker on another. However, oftentimes there is similarity between jobs (they may share common subjobs). We intend to explore both of these factors in future work. Finally, we intend to incorporate an explicit representation of human learning into more complicated scheduling problems, such as those that have releasing time, due dates, precedence relationship, or have a more complicated structure, such as flow shops.

References

- Anzanello, M. J., & Fogliatto, F. S. (2011). Learning curve modelling of work assignment in mass customized assembly lines. *International Journal of Production Research*, 45, 2919–2938. doi:10.1080/00207540600725010.
- Anzanello, M. J., & Fogliatto, F. S. (2011). Learning curve models and applications: Literature review and research directions. *International Journal of Industrial Ergonomics*, 41, 573–583. doi:10.1016/j.ergon.2011.05.001.

- 493 Corominas, A., Olivella, J., & Pastor, R. (2010). A model for the assign-
494 ment of a set of tasks when work performance depends on experience of all
495 tasks involved. *International Journal of Production Economics*, 126, 335–340.
496 doi:10.1016/j.ijpe.2010.04.012.
- 497 Bevis, F.W., Finniear, C., Towill, D. R. , (1970). Prediction of operator perfor-
498 mance during learning of repetitive tasks. *International Journal of Production*
499 *Research*, 8, 293–305.
- 500 Dar-El, E. M. (2000). *Human Learning: From Learning Curves to Learning*
501 *Organizations* volume 29 of *International Series in Operations Research &*
502 *Management Science*. Boston: Kluwer Academic Publishers.
- 503 Hewitt, M., Chacosky, A., Grasman, S. E., & Thomas, B. W. (2015). Integer
504 Programming Techniques for Solving Non-linear Workforce Planning Models
505 with Learning. *European Journal of Operational Research*, 242, 942–950.
- 506 Gutjahr, W. J., Katzensteiner, S., Reiter, P., Stummer, C., & Denk, M.
507 (2008). Competence-driven project portfolio selection, scheduling and staff
508 assignment. *Central European Journal of Operations Research*, 16, 281–306.
509 doi:10.1007/s10100-008-0057-z.
- 510 Heimerl, C., & Kolisch, R. (2010). Work assignment to and qualification of
511 multi-skilled human resources under knowledge depreciation and company
512 skill level targets. *International Journal of Production Research*, 48, 3759–
513 3781. doi:10.1080/00207540902852785.
- 514 Hewitt, M., Chacosky, A., Grasman, S. E., & Thomas, B. W. (2015). Integer
515 Programming Techniques for Solving Non-linear Workforce Planning Models
516 with Learning. *European Journal of Operational Research*, 242, 942–950.
- 517 Jaber, M. Y. (2006). Learning and Forgetting Models and Their Applications.
518 In A. B. Badiru (Ed.), *Handbook of Industrial and Systems Engineering* In-
519 dustrial Innovation Series chapter 30. (pp. 30–1–30–27). Boca Raton, FL,
520 USA: CRC Press.

- 521 Jaber, M. Y., & Sikström, S. (2004). A numerical comparison of three po-
522 tential learning and forgetting models. *International Journal of Production*
523 *Economics*, 92, 281–294.
- 524 Mazur, J. E., & Hastie, R. (1978). Learning as accumulation: a reexamination
525 of the learning curve. *Psychological bulletin*, 85, 1256–74.
- 526 Nembhard, D. A., & Norman, B. A. (2007). Cross Training in Production
527 Systems with Human Learning and Forgetting. In D. A. Nembhard (Ed.),
528 *Workforce Cross Training* chapter 4. (pp. 111–129). Boca Raton, FL, USA:
529 CRC Press.
- 530 Olivella, J., Corominas, A., & Pastor, R. (2013). Task assignment consider-
531 ing cross-training goals and due dates. *International Journal of Production*
532 *Research*, 51, 37–41.
- 533 Sayin, S., & Karabati, S. (2007). Assigning cross-trained workers to depart-
534 ments: A two-stage optimization model to maximize utility and skill im-
535 provement. *European Journal of Operational Research*, 176, 1643–1658.
536 doi:10.1016/j.ejor.2005.10.045.
- 537 Wright, T. (1936). Factors affecting the cost of airplanes. *Journal of the Aero-*
538 *nautical Sciences*, 3, 122–128.

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

Funding

This material is based upon work supported by the National Science Foun-
dation under Grant No. CMMI-1266010.