MODELS FOR THE INCLUSION OF WORKERS WITH DISABILITIES IN FLOW SHOP SCHEDULING PROBLEMS

Germano Caumo Carniel, Alexander J. Benavides, Marcus Ritt
Instituto de Informática – Universidade Federal do Rio Grande do Sul
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil
{gccarniel,ajbenavides,marcus.ritt}@inf.ufrgs.br

ABSTRACT

Persons with disabilities have severe problems participating in the job market and their unemployment rate is usually much higher than the average of the population. This motivates the research of new modes of production which allow the inclusion of these persons at a low overhead.

In this paper we study the inclusion of persons with disabilities into flow shops with the objective of minimizing the makespan. Since flow shops usually have only a few machines, we focus on the inclusion of a single worker with disabilities into such a production line. We define the problem, propose several mathematical models for solving it, as well as realistic test instances. Computational tests show that the best model is able to optimally solve instances of moderate size. This allows us to study the potential benefits of this approach. In particular, we observe that the increase in makespan is lower than expected and does not depend on severe disabilities of the workers.

KEYWORDS. Flow shop scheduling. Workers with Disabilities. Integer Programming.

Combinatorial optimization

RESUMO

Portadores de deficiências encontram enormes dificuldades em participar ativamente do mercado de trabalho e sua taxa de desemprego normalmente é muito maior que a média da população. Isso motiva a pesquisa por novos modos de produção que permitam a inclusão dessas pessoas com poucas perdas.

Neste artigo nós estudamos a inclusão de portadores de deficiências em flow shops com o objetivo de minimizar o makespan. Como flow shops normalmente possuem poucas máquinas, nós focamos na inclusão de somente um trabalhador com deficiências. O problema é definido e são propostos modelos matemáticos para resolvê-lo, assim como instâncias de teste realistas. Testes computacionais mostram que o melhor modelo consegue resolver otimamente instâncias médias, permitindo-nos estudar os possíveis benefícios dessa estratégia. Em particular, nós observamos que o aumento do makespan é menor do que o esperado, não dependendo tanto das deficiências dos trabalhadores.

PALAVRAS CHAVE. Escalonamento de tarefas, Trabalhadores com deficiências, Programação Inteira.

Otimização combinatória

1. Introduction

In 2004, the World Health Survey and the "Global Burden of Disease" project estimated the population of persons with a disability of 15 years and older around 785 (15.6%) to 975 (19.4%) million. According to the World Health Organization and the International Labour Organization, unemployment rates are much higher for persons with disabilities than for persons without disabilities in both developed and developing countries. However, almost all tasks can be performed by persons with disabilities, since they often have the necessary skills, and most of them can be productive in an appropriate environment (WHO, 2011).

Governments of many countries adopt strategies to incorporate persons with disabilities into the labour force, for example in Sheltered Work Centers for Disabled (SWDs) or by laws that oblige companies to contract a minimum percentage of workers with disabilities. These models of socio-labor integration try to overcome the stereotype that considers people with disabilities as unable to develop continuous professional work (Miralles et al., 2010).

Studies have shown that the vast majority of available job positions for workers with disabilities are in the production area (Costa et al., 2009). Motivated by similar studies that have demonstrated that such workers can be integrated successfully in assembly lines (e.g. Miralles et al. (2007)), we study in this paper the integration of workers with disabilities into flow shops with the objective of minimizing the makespan. Since flow shops have relatively few machines, and legislation usually foresees an integration of about 2% to 5% of workers with disabilities, we focus on the case of the integration of a single worker into a flow shop.

The rest of this paper is organized as follows: the next section reviews the literature on related problems and in Section 3 we define the problem formally and give an example. In Section 4 we propose three integer programming (IP) models for the integration of a single worker with disabilities into a traditional flow shop. In Section 5 we define a set of instances modelling realistic conditions, present computational experiments done with the three models using a commercial IP solver, and analyze the result. Finally, in Section 6 we discuss the results, and offer some conclusions.

2. Literature review

The integration of workers with disabilities into flow shops has, to the best of our knowledge, not been studied in the literature so far. However, several researchers have been studying the problem of integrating persons with disabilities in the area of production. For assembly lines, Miralles et al. (2007) introduced the Assembly Line Worker Assignment and Balancing Problem (ALWABP). In this problem the task execution time depends on the worker and tasks as well as workers must be assigned simultaneously to a fixed number of stations such that the production rate of the assembly line is maximized. This problem is NP-hard, since it generalizes the NP-hard Simple Assembly Assembly Line Balancing Problem, and researches have focused mainly on heuristic methods for solving it (see for example Miralles et al. (2008, 2010); Blum and Miralles (2011); Araújo et al. (2012); Mutlu et al. (2013); Moreira et al. (2012)).

The ALWABP considers the heterogeneity of the workers when there are only workers with disabilities available. Costa et al. (2009) studied the effect of including only one disabled worker in an assembly line. This models the case of conventional factories, where disabled workers must be employed among regular workers. They conclude that it is pos-

	Machine			
Job	M_1	M_2	M_3	M_4
J_1	1	2	2	1
J_2	1	1	2	2
J_3	2	1	1	2
J_4	1	3	2	1

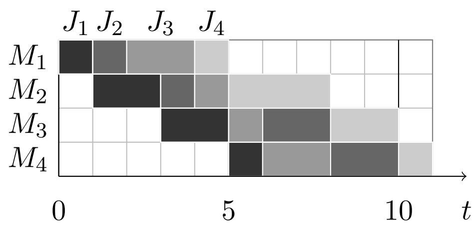

Figure 1. Example of a flow shop instance. Left: Processing times. Right: Gantt chart of the optimal schedule.

sible to integrate disabled people transparently with only little impact on the efficiency of the assembly line.

The flow shop scheduling problem (FSSP) has been extensively studied in the literature. Most research focuses on the permutation flow shop problem (PFSSP), where all the jobs have to be processed in the same order on all machines, although Potts et al. (1991) showed that there are instances for which the makespan of an optimal solution for the PFSSP is worse than the optimal solution for the non-permutation FSSP by more than a factor of $\sqrt{m}/2$. The PFSSP can be solved in polynomial time for two machines (Johnson, 1954) but is NP-hard for three or more machines (Garey and Johnson, 1979). The existence of a constant factor polynomial-time approximation algorithm for flow shop scheduling is open (although there exists a polynomial-time approximation scheme for a fixed number of machines (Hall, 1998)). We will discuss mathematical models for the FSSP and the PFSSP in Section 4. For more details on approaches to solve the (P)FSSP, we refer to the excellent surveys of Gupta and Stafford (2006) and Potts and Strusevich (2009).

3. Flow shop scheduling including workers with disabilities

In a FSSP, we have to schedule a set of jobs $J_1, \ldots, J_n$ on machines $M_1, \ldots, M_m$. Each job J_i must be processed on machine M_r in time p_{ri} without preemption. The jobs cannot be processed in parallel, and the machines can process only one job at a time. In the PFSSP the solutions are restricted to permutation schedules, where the jobs have to be processed on all machines in the same order.

The most common objective function for the FSSP and the PFSSP is to minimize the makespan $C_{\max} = \max C_i$, i.e. the maximum over all completion times C_i, $i \in [n]$ of the jobs[1]. Examples of other objective functions include the total flowtime $\sum C_i$, or, for given deadlines, the total lateness or tardiness, or weighted version thereof. In this paper, we focus on the minimization of the makespan of the FSSP and PFSSP.

An example of an instance of the FSSP is shown in Figure 1. In the table on the left we are given four jobs to process and their processing times on four machines. The optimal schedule shown on the right has a makespan of 11. Note that this is no permutation schedule, since jobs two and three exchange their processing order on machine three. In a PFSSP instance, this schedule is not allowed.

The situation encountered when ordinary companies need to integrate disabled workers in their workforce can be seen in the left part of Figure 2. In addition to the times that regular workers take to perform the operations, we have times for a worker with disabilities, which normally exceed the time of the regular workers. In the example, the times of the worker with disabilities were chosen randomly in the interval $[p, 2p]$, for a pro-

[1] We use the notation $[n] = \{1, 2, \ldots, n\}$.

	Regular				With disabilities			
Job	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4
J_1	1	2	2	1	2	4	2	∞
J_2	1	1	2	2	1	1	4	∞
J_3	2	1	1	2	4	2	1	∞
J_4	1	3	2	1	1	4	2	∞

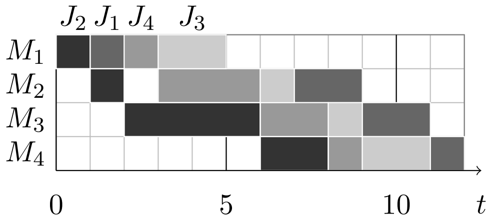

Figure 2. An instance of FSISP, where a single worker with disabilities must operate one machine in a flow shop. Left: Processing times for regular worker and worker with disabilities. Right: Gantt chart of optimal schedule.

Table 1. Notation used in the integer programming models.

A	set of machines which the worker with disabilities can operate
p_{ri}	processing time of job J_i on machine M_r when executed by regular workers
d_{ri}	processing time of job J_i on machine M_r when executed by the worker with disabilities
M	a large constant
C_{ri}	completion time of job J_i on machine M_r
S_{ri}	starting time of job J_i on machine M_r
Y_{rj}	waiting time of job in sequence position j after it finishes processing on machine M_r
T_{ri}	processing time of job J_i on machine M_r after assigning the worker with disabilities
T_{rj}	processing time of job in sequence position j on machine M_r after assigning the worker with disabilities
D_{ik}	1, if job J_i precedes job J_k, 0, otherwise, for $i < k$
D_{ikr}	1, if job J_i precedes job J_k on machine r, 0, otherwise, for $i < k$
Z_{ij}	1, if job J_i is assigned to sequence position j, 0 otherwise
X_r	1, if the worker with disabilities is assigned to machine M_r, 0 otherwise

cessing time of p of a regular worker. The worker with disabilities may also be unable to operate some of the machines. In the example, this is the case for machine M_4 represented by processing times of ∞ on this machine.

The problem of inserting a worker with disabilities into a flow shop (flow shop insertion and scheduling problem, FSISP) is defined as follows: we have to assign the worker with disabilities to a machine he is able to operate, and find a valid schedule of the jobs, such that the makespan is minimized. We call the variant restricted to permutation schedules the permutation flow shop insertion and scheduling problem (PFSISP).

The optimal makespan for FSISP in the above example is 12, obtained when assigning the disabled worker to machine M_3, and can be seen in Figure 2.

4. Mathematical formulation of FSISP and PFSISP

In this section we present a mathematical formulation of the FSISP, and two formulations for the PFSISP that consider the inclusion of a worker with disabilities. When presenting the models we use subscripts $r \in [m]$ for machines, $i \in [n]$ for jobs, and $j \in [n]$ for sequence positions. Table 1 explains the variables used in the models.

4.1. Flow shop scheduling and insertion problem

The model for flow shop scheduling is based on the classical model of Wagner (1959) using dichotomous constraints for ordering the jobs on each machine. We extend this model by modifying the processing times according to the assignment of the worker with disabilities.

$$\begin{aligned}
\textbf{min.} \quad & C_{\max}, && && (1)\\
\textbf{s.t.} \quad & S_{ri} + T_{ri} \le C_{\max}, && \forall r \in [m], i \in [n], && (2)\\
& S_{ri} + T_{ri} \le S_{r+1,i}, && \forall r \in [m-1], i \in [n], && (3)\\
& S_{ri} + T_{ri} \le S_{rk} + P(1 - D_{ikr}), && \forall r \in [m], i, k \in [n], i < k, && (4)\\
& D_{ikr} + D_{kir} = 1, && \forall i, k \in [n], i < k, r \in [m], && (5)\\
& T_{ri} = p_{ri}(1 - X_r) + d_{ri} X_r, && \forall r \in [m], i \in [n], && (6)\\
& \sum_{r \in A} X_r = 1. && && (7)
\end{aligned}$$

In this model constraint (2) defines $C_{\max}$ as the latest completion time. Constraints (3) and (4) set the starting time of all operations according to the precedences. Constraint (5) enforces precedence relations for operations on the same machine. The processing time of an operation depends on the machine the worker with disabilities is assigned to, and is defined in constraint (6). Constraint (7) requires that the disabled worker is assigned to one of the machines he can operate.

4.2. Permutation flow shop insertion and scheduling problem

Tseng and Stafford (2007) and Stafford et al. (2004) present an extensive comparison of models for the PFSSP. The model the authors identified as performing best, called TS3, however, is harder to adapt for the PFSIP, since it multiplies the processing times with binary decision variables Z_{ij}, which define the permutation of the jobs. For this reason, we study the extension of two models to the PFSISP: the best model using dichotomous constraints, called LYeq, as well as a linearization of the TS3 model.

Adapting the LYeq model to the PFSISP

The LYeq model of Liao and You (1992) for the job shop scheduling problem was applied by Pan (1997) to the PFSSP. It can be modified in the same manner as the Wagner model to include an additional worker with disabilities. The model uses the additional surplus variables Q_{rik} for the time between the completion of job J_k and the start of job J_i, if k precedes i. The constant P can be set to $\sum_{j \in [n]} \max_{r \in [m]} d_{ri}$.

$$\begin{aligned}
\textbf{min.} \quad & C_{\max}, && && (8)\\
\textbf{s.t.} \quad & C_{mi} \le C_{\max}, && \forall i \in [n], && (9)\\
& C_{1i} \ge T_{1i}, && \forall i \in [n], && (10)\\
& C_{r+1,i} - C_{ri} \ge T_{r+1,i}, && \forall r \in [m-1], i \in [n], && (11)\\
& P D_{ik} + C_{ri} - C_{rk} - T_{ri} = Q_{rik}, && \forall r \in [m], i, k \in [n], i < k, && (12)\\
& Q_{rik} \le P - T_{ri} - T_{rk}, && \forall r \in [m], i, k \in [n], i < k, && (13)\\
& Q_{rik} \ge 0 && \forall r \in [m], i, k \in [n] && (14)\\
& \text{equations (6) and (7).}
\end{aligned}$$

Constraint (9) defines $C_{\max}$ as the latest completion time on the last machine. Constraint (10) ensures each job can only be completed on machine 1 after it is fully processed on that machine. Constraint (11) states that each job is processed on only one machine at a time. Constraint (12) replaces the dichotomous constraints in the Wagner model, and equation (13) upper bounds the surplus variables Q_{rik}, in case i precedes k.

Adapting the TS3 model to the PFSISP

The TS3 model was proposed by Tseng and Stafford (2007). The underlying principle of the formulation is to define the starting times of job J_i at machine M_r by the sum of its predecessors on the first machine, plus the sum of its processing times on machines $1, \dots, r-1$ and the waiting time on these machines before starting on the next machines. The comparison of these partial sums yields compact constraints for expressing the relative starting times for a given job permutation.

$$\textbf{min.} \quad C_{\max} = \sum_{p \in [n]} T_{1p} + \sum_{q \in [2,m]} T_{qn} + \sum_{q \in [m-1]} Y_{qn}, \tag{15}$$

$$\textbf{s.t.} \quad \sum_{i \in [n]} Z_{ij} = 1, \qquad \forall i \in [m], \tag{16}$$

$$\sum_{j \in [n]} Z_{ij} = 1, \qquad \forall i \in [n], \tag{17}$$

$$T_{1,j-1} - T_{r,j-1} + \sum_{q \in [r-1]} T_{qj} - T_{q,j-1} + \sum_{q \in [r-1]} Y_{qj} - Y_{q,j-1} \geq 0, \qquad \forall r \in [2,m], j \in [2,n], \tag{18}$$

$$T_{rj} = \sum_{i \in [n]} p_{ri}(1 - X_r)Z_{ij} + d_{ri}X_r Z_{ij}, \qquad \forall r \in [m], j \in [n], \tag{19}$$

equation (7).

Constraint (15) defines $C_{\max}$ as the sum of three components: (i) the processing time of all jobs on the first machine, (ii) the processing time of last job on all remaining machines, and (iii) the waiting times of the last job on all machines, except the last one. Constraints (16) and (17) model the assignment of jobs to positions: each job is assigned to only one sequence position and each sequence position has only one job assigned to it. Constraint (18) relates the starting times of the jobs at sequence positions $j-1$ and j according to the principle explained above. Constraint (19) defines the processing times of the jobs at each sequence position according to the assignment of the worker with disabilities to one of the machines he is able to operate. Note that these constraints are non-linear, but can be easily linearized using standard methods, by introducing n^2m auxiliary binary variables and n^2m additional constraints.

5. Computational Experiments

We conducted a series of computational experiments to compare the performance of the different models, as well as to study the benefit of inserting workers with disabilities into traditional flow shops. We first propose test instances in Section 5.1 and then present the numerical results in Section 5.2.

Table 2. Results for Carlier instances.

		FS			LYeq			TS3		
Var.	Inc.	Gap	Time	Opt	Gap	Time	Opt	Gap	Time	Opt
2	0	14.25	2168.7	4	5.31	1261.9	6	0.00	26.8	8
2	10	14.08	1971.8	4	3.86	1122.3	6	0.00	17.8	8
2	20	13.92	2162.5	4	4.32	1191.8	6	0.00	14.6	8
5	0	25.04	2267.8	3	14.79	1440.9	5	0.00	55.7	8
5	10	17.19	2276.8	3	10.91	1453.0	5	0.00	46.7	8
5	20	12.06	2279.5	3	8.54	1423.8	5	0.00	11.3	8

5.1. Test instances

For the computational experiments we created instances for the inclusion problem based on the well-known flow shop instances proposed by Carlier (1978) and Taillard (1993). We assume that the processing times p_{ri} of a flow shop instance are those of a regular worker. To model a disabled worker, we modify these processing times in two ways. First, a fixed percentage of incompatibilities is introduced. An incompatibility models the case of a worker who is unable to operate some machine, as, for example, machine 4 for the worker with disabilities in the instance given in Figure 2. Second, the processing times are increased to reflect that a disabled worker usually needs more time to execute a job. Based on experiences made with workers in SWDs, we opted to produce instances with no incompatibilities, as well as 10% and 20% of incompatibilities per worker. The processing time p of a regular worker for some job on some machine is increased by choosing uniformly at random a processing time in the interval $[p, 2p]$ or $[p, 5p]$.

We limited our tests to the eight Carlier instances, and the first group of ten Taillard instances with 20 jobs and 5 machines, since the larger instances are hard to solve exactly even in the regular case. With three levels of incompatibilities and two levels of task time variation, we obtain a total of 108 test instances. These instances are available from the authors on demand.

5.2. Numerical results

We solved all instances described above using the three models presented in Section 4. The mathematical models have been solved using the commercial solver CPLEX 12.5 running with a single thread and a time limit of one hour. All computational tests were executed on a PC with an Intel Core i7 processor running at 2.8 GHz, and with 12 GB of main memory.

The results are presented in Table 2 for the Carlier instances and in Table 3 for the Taillard instances. Each table compares the overall averages of the three models for each of the two time variations, and each level of incompatibilities. In the tables model (2)–(7) is denoted by FS. For each combination we report the average percentage relative deviation between the lower and upper bounds found by CPLEX ($\overline{Gap\%}$), the average running time taken by CPLEX (Time), and the number of optimum solutions found for each group of instances (Opt).

The FS model had the worst performance, solving about half of the Carlier instances in an average of 36 minutes and none of the Taillard instances. The LYeq model is clearly better, solving two more Carlier instances in about 60% of the time of the FS model, but was still unable to solve any of the Taillard instances. These relative performance of the two models was expected, since both use dichotomous constraints for defining the job order, but the FS model solves a harder problem. The TS3 model performed much better than

Table 3. Results for Taillard instances.

		FS			LYeq			TS3		
Var.	Inc.	Gap	Time	Opt	Gap	Time	Opt	Gap	Time	Opt
2	0	60.02	-	0	52.72	-	0	0.86	1861.7	7
2	10	59.84	-	0	52.78	-	0	0.03	2065.4	9
2	20	58.44	-	0	51.63	-	0	0.00	1301.8	10
5	0	67.78	-	0	64.15	-	0	1.64	2496.1	7
5	10	65.12	-	0	63.55	-	0	1.20	2182.8	7
5	20	58.83	-	0	62.34	-	0	1.90	2557.5	5

Table 4. TS3 average of best solution percentage difference compared to the normal problem

		Carlier	Taillard
Var.	Inc.	APRD (%)	APRD (%)
2	0	7.44	14.74
2	10	7.86	17.03
2	20	9.25	18.31
5	0	75.82	106.65
5	10	75.82	108.43
5	20	77.71	119.16

the other models, solving all Carlier instances in a small fraction of the time and being able to solve 75% of Taillard instances. Thus, the TS3 model for the PFSSP continues to be the strongest model for PFSISP, even with the additional overhead from linearizing constraints (19). We can also see that an increased time variation makes the problem in general more difficult to solve, while increasing the incompatibilities makes it easier, since it reduces the number of feasible assignments of the worker with disabilities to machines.

To assess the potential loss or benefit of including a worker with disabilities into a flow shop, we computed the average percentage relative deviation (APRD) between the best solution found by the modified TS3 model and the best solution of the underlying regular PFSSP problem. Table 4 shows the results for the Carlier and Taillard instances. We can first observe that the increase in the makespan when inserting a worker with disabilities depends only weakly on the number of incompatibilities for percentages that can be expected in practice. Further, the makespans increase in all cases. They never exceed 20% for a time variation in $[p, 2p]$, but the makespan almost doubles for a $[p, 5p]$. The insertion of a worker with disabilities cannot be hidden in these instances, and increases more than linearly with the time variation. This is to be expected, since the disabled workers must operate a machine and process all the jobs on it, and the increased processing time usually cannot be hidden in idle times. On the other hand, the makespan increases less than one would expect from the increase of the processing times of a single machine. The Taillard instances, for example, have processing times chosen uniformly from $[1, 99]$, and thus an expected processing time of $\overline{p} \approx 50$ and an expected makespan of $\overline{p}(n + m - 1) \approx 1200$ for the standard instances. (The relative deviation of the true makespan from $\overline{p}(n+m-1)$ for the Taillard instances is in average $0 \pm 4.3\%$.) In this model a processing time variation of $[p, (1 + 2r)p]$ of a single machine would result in a expected increase of $r\overline{p}n$ in the makespan, and we would expect an increase of about 42% and 166% for $[p, 2p]$ and $[p, 5p]$, respectively. This indicates that the additional flexibility of assigning a worker with disabilities to any of the machines may help to hide some of the expected costs by the increase of processing times.

6. Concluding remarks

In this paper we have proposed and studied the insertion of workers with disabilities into flow shops, where one has to find an optimal assignment of the disabled worker to a workstation, as well as an optimal schedule for the operations. We have proposed three mathematical formulations for solving this problem, and a set of realistic test instances.

The comparison of the three models shows that only the permutation flow shop variant can solve instances of moderate size to optimality. We further can conclude that the best model for the regular flow shop, TS3, also performs best for worker insertion.

From a practical point of view, our results show that the inclusion of workers with disabilities can be optimized with standard techniques for moderate problem sizes. For larger flow shop instances the application of heuristics seems to be necessary. The incompatibilities have almost no impact on the resulting makespan, such that workers which cannot operate some machines, but have almost regular processing times for the remaining machines can be integrated into a flow shop with virtually no overhead. For larger processing time variations, the makespan will increase, but our results suggest that an optimized assignment may reduce that increase to less than the expected value. This suggests further that companies can contribute to integrate people with disabilities in their production systems with only moderate losses in productivity. We hope this can lower the prejudice and help to increase the participation of disabled people in the market and in society.

Acknowledgements

This research has been supported by FAPERGS, project no. 12/2046-6, by CAPES-Brazil and MEC-Spain (coordinated project CAPES-DGU258-12/PHB-0012-PC). Alexander J. Benavides has received support from Petrobras. This support is gratefully acknowledged.

References

Araújo, F., Costa, A., and Miralles, C. (2012), Two extensions for the assembly line worker assignment, and balancing problem: parallel stations and collaborative approach, *Int. J. Prod. Econ.*, 140:483–495.

Blum, C. and Miralles, C. (2011), On solving the assembly line worker assignment and balancing problem via beam search, *Comput. Oper. Res.*, 38(1):328–339.

Carlier, J. (1978), Ordonnancements a contraintes disjonctives, *R.A.I.R.O. Recherche operationelle/Operations Research*, 12(4):333–351.

Costa, A. M., Santos, L. M. R., Mayron, C., and Moreira, D. O. (2009), Análise da eficiência de linhas de produção com trabalhadores deficientes, In *Anais do XLI Simpósio Brasileiro de Pesquisa Operacional.*

Garey, M. R. and Johnson, D. S. (1979), *Computers and intractability: A guide to the theory of NP-completeness*, Freeman.

Gupta, J. and Stafford, E. (2006), A comprehensive review and evaluation of permutation flowshop heuristics, *Eur. J. Oper. Res.*, 169(3):699–711.

Hall, L. A. (1998), Approximability of flow shop scheduling, *Math. Prog. B*, 82(1–2):175–190.

Johnson, S. (1954), Optimal two- and three-stage production schedules with setup times included, *Naval research logistics quarterly*, 1(1):61–68.

Liao, C.-J. and You, C.-T. (1992), An improved formulation for the job-shop scheduling problem, *J. Oper. Res. Soc.*, 43(11):1047–1054.

Miralles, C., Garcia-Sabater, J., Andres, C., and Cardos, M. (2007a), Advantages of assembly lines in sheltered work centres for disabled. a case study, *Int. J. Prod. Econ.*, 110(1):187–197.

Miralles, C., García-Sabater, J., Andrés, C., and Cardós, M. (2008), Branch and bound procedures for solving the assembly line worker assignment and balancing problem: Application to sheltered work centres for disabled, *Discrete App. Math.*, 156(3):352–367.

Miralles, C., Garcia-Sabater, J. P., Andrés, C., and Cardos, M. (2007b), Advantages of assembly lines in Sheltered Work Centres for Disabled. A case study, *Int. J. Prod. Res.*, 110(2):187–197.

Miralles, C., Marin-Garcia, J., Ferrus, G., and Costa, A. (2010), Or/ms tools for integrating people with disabilities into employment. a study on valencia's sheltered work centres for disabled, *Int. Trans. Oper. Res.*, 17:457–473.

Moreira, M. C. O., Ritt, M., Costa, A. M., and Chaves, A. A. (2012), Simple heuristics for the assembly line worker assignment and balancing problem, *J. Heuristics*, 18(3):505–524.

Mutlu, O., Polat, O., and Supciller, A. A. (2013), An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-ii, *Comput. Oper. Res.*, 40(1):418 – 426.

Pan, C.-H. (1997), A study of integer programming formulations for scheduling problems, *J. Comput. Syst. Sci.*, 28(1):33–41.

Potts, C. N., Shmoys, D. B., and Williamson, D. P. (1991), Permutation vs. non-permutation flow shop schedules, *Oper. Res. Lett.*, 10(5):281–284.

Potts, C. N. and Strusevich, V. A. (2009), Fifty years of scheduling: a survey of milestones, *J. Oper. Res. Soc.*, 60:S41–S68.

Stafford, E. F., Tseng, F. T., and Gupta, J. N. D. (2004), Comparative evaluation of MILP flowshop models, *J. Oper. Res. Soc.*, 56(1):88–101.

Taillard, E. (1993), Benchmarks for basic scheduling problems, *Eur. J. Oper. Res.*, 64(2):278 – 285.

Tseng, F. T. and Stafford, E. F. (2007), New MILP models for the permutation flowshop problem, *J. Oper. Res. Soc.*, 59(10):1373–1386.

Wagner, H. M. (1959), An integer linear-programming model for machine scheduling, *Nav. Res. Logist.*, 6(2):131–140.

WHO (2011), *World report on disability*, World Health Organization.