6-9-2022

Learning to generalize Dispatching rules on the Job Shop Scheduling

Zangir Iklassov
Mohamed bin Zayed University of Artificial Intelligence

Dmitrii Medvedev
Mohamed bin Zayed University of Artificial Intelligence

Ruben Solozabal
Mohamed bin Zayed University of Artificial Intelligence

Martin Takac
Mohamed Bin Zayed University of Artificial Intelligence

Learning to generalize Dispatching rules on the Job Shop Scheduling

Zangir Iklassov*
Zangir.Iklassov@mbzuai.ac.ae

Dmitrii Medvedev*
Dmitrii.Medvedev@mbzuai.ac.ae

Ruben Solozabal*
Ruben.Solozabal@mbzuai.ac.ae

Martin Takáč
Takac.MT@gmail.com

Mohamed bin Zayed University of Artificial Intelligence (MBZUAI)
Masdar City, Abu Dhabi, UAE

Abstract

This paper introduces a Reinforcement Learning approach to better generalize heuristic dispatching rules on the Job-shop Scheduling Problem (JSP). Current models on the JSP do not focus on generalization, although, as we show in this work, this is key to learning better heuristics on the problem. A well-known technique to improve generalization is to learn on increasingly complex instances using Curriculum Learning (CL). However, as many works in the literature indicate, this technique might suffer from catastrophic forgetting when transferring the learned skills between different problem sizes. To address this issue, we introduce a novel Adversarial Curriculum Learning (ACL) strategy, which dynamically adjusts the difficulty level during the learning process to revisit the worst-performing instances. This work also presents a deep learning model to solve the JSP, which is equivariant w.r.t. the job definition and size-agnostic. Conducted experiments on Taillard's and Demirkol's instances show that the presented approach significantly improves the current state-of-the-art models on the JSP. It reduces the average optimality gap from 19.35% to 10.46% on Taillard's instances and from 38.43% to 18.85% on Demirkol's instances. Our implementation is available online [2].

1 Introduction

The Job Shop Scheduling Problem (JSP) is a combinatorial problem with vast implications for scheduling optimization of real-world tasks. It is formulated as a set of jobs, each consisting of a set of operations, to be processed on a set of heterogeneous machines in the shortest possible time. Furthermore, each operation is described with two features: the specific machine each operation shall be assigned to and the operational time it takes to complete the operation. The above formulation applies to any economic task concerned with the optimal assignment of capital goods versus means of production, i.e., education, research, manufacturing, storage, transportation, sales, etc. However, JSP is an NP-hard combinatorial problem, which usually means that solving them optimally is impractical. Thus, there are two basic approaches to finding an approximate solution for this class of problems. The first approach consists of various optimization methods, including integer programming, constraint programming, or meta-heuristic algorithms. However, those are computationally expensive methods. The second approach is to employ hand-engineered heuristics that provide flexibility to react to unexpected events in the scheduling plan.

*Equal contribution

[2] https://github.com/Optimization-and-Machine-Learning-Lab/Job-Shop/tree/main_nips

Preprint. Under review.

Designing such heuristic rules is a daunting task that requires specialized knowledge of the problem. For example, Priority Dispatch Rules (PDRs) are a family of scheduling heuristics that are fast and easy to implement but computationally expensive. They are also sensitive to the initialization of the problem's instances. Therefore, researchers started to gravitate toward Reinforcement Learning (RL) to discover well-behaved domain-specific heuristics automatically. Unlike PDRs, RL uses instance information to particularize a solution strategy and has already demonstrated promising results [27, 17], though the problem of generalization is yet to be addressed.

In this paper we provide a nostrum for generalization on the JSP using deep RL. To this end, we design a problem-specific architecture and novel training approach that stress this direction. Our main contributions are summarized below:

- This work presents a **deep learning model** to solve the JSP. Particularly, we address the JSP as a Markov Decision Process (MDP), in which the model iteratively constructs a solution based on the operations that are yet to be scheduled, as well as on the state on the problem resolution. The model is equivariant w.r.t. the job information and size-agnostic, enabling us to train the model on different problem sizes. It is key for the generalization study we perform in this paper.
- We also present **Adversarial Curriculum Learning (ACL)**, a novel curriculum strategy to improve generalization based on adversarial instances. Curriculum learning (CL) used to suffer from catastrophic forgetting when transferring the learned skills between different problem sizes. To address this issue, we propose ACL, a learning strategy that dynamically reinforces the model on worst-performing instances. Conducted experiments show this strategy presents an improvement in the difficulty selection when compared to previous CL approaches.

The aforesaid ideas deliver an improvement in performance and generalization when compared to state-of-the-art works on JSP [27, 23]. Notably, we reduce the optimality gap from 19.35% to 10.46% on Taillard's instances and from 38.43% to 18.85% on Demirkol's instances.

2 Related work

First attempts of using RL to address scheduling problems date back to the 90s [12, 13, 29]. On specific relevance is Zhang and Dietterich's paper [29] on allocating resources for NASA shuttle missions. As it was reflected in the literature, one of the advantages of using RL for such a purpose is that it can be seamlessly used for static, dynamic [5, 1] or stochastic variants of the problem. However, due to the complexity of the problems addressed, it could only be applied to small instances of the problem, limiting its applicability significantly.

The rise of deep learning allowed to extrapolate this technique to more realistic problem instances. Prior works particularized deep neural networks, e.g., Pointer Networks [22], to learn in combinatorial spaces. In [2], deep RL was implemented for the first time to learn *end-to-end* solutions to combinatorial problems. The authors used the Pointer Network in an actor-critic architecture to address the Travelling Salesman Problem. Further studies implemented Transformer networks [4, 20, 7]. However, all these works share in common that they are based on sequence-to-sequence models, where the complete solution is output at once. Thereby, heavy sampling and searching techniques were required at interference to improve the solution. Our approach is in line with [15], where the Vehicle Routing Problem is described as a Markov Decision Process, and the solution is iteratively constructed based on sequential decisions.

In the particular case of the JSP, several techniques have been applied to learn on it. Imitation learning was used in [6] to learn from optimal solutions on training instances that were labeled using a Mixed-Integer Programming (MIP) solver. RL has also been applied to the problem, e.g., to select pre-defined candidate PDRs according to the scheduling conditions [1, 9]. Other works have addressed the problem from a multi-agent perspective, e.g., in cooperative manufacturing [5, 25] where each agent controls a production line. Moreover, numerous examples of scheduling in many application domains, including manufacturing [9, 23], distributed computing [14, 28, 18] or supply chains. Despite the effort, many of these approaches do not beat heuristics. In addition, a major limitation in many of these works is that the state representation is hard-bounded by some factors

(e.g., size of jobs or number of operations to consider), not enabling to scale the solution to arbitrary problem sizes.

Zhang et al. [27] presented a size-agnostic model that allows generalizing on different instance sizes. They formulate the JSP as a disjunctive graph and use a high discriminative Graph Isomorphism Network (GIN) to embed the states in the resolution procedure. They prove to capture raw features from small problem instances and manage to successfully extrapolate to much larger instances. Even though graph neural networks have shown unprecedented success embedding not-euclidean spaces, this embedding does not capture the most relevant information to design constructive heuristics on scheduling problems and the features of the remaining operations.

One of the key features that improve the generalization capabilities of a model is the use of CL; this aspect has been pointed out several times in the literature. E.g. [30] employs transfer learning to reconstruct the trained policies on problems of different sizes. However, policy transfer is still relatively costly and inconvenient. Also, [11] uses lifelong learning, where an agent will not only learn to optimize one specific problem instance but reuse what it has learned from previous instances. They also proposed a parallel training method that combines asynchronous updates with a deep deterministic policy gradient to speed up model training. In [10], the authors use an improved CL strategy with an adaptive staircase mechanism, where, at each iteration, the model can change the difficulty level, i.e., go the previous level, stay at the same level, or advance to the next one. In our work, we build on top of this idea, tracking the model's behavior at each stage, allowing it to jump to the worst-performing levels.

3 Method

Problem formulation. In the deterministic JSP, a finite set of n jobs $\{J_i\}_{i=1}^{n}$ are to be processed on a finite set of m machines $\{M_j\}_{j=0}^{m}$. Each job J_i consists of a chain of m operations $O_{i,1} \rightarrow ... \rightarrow O_{i,m}$, that have to be processed in a predetermined order. For each operation $O_{i,j}$, the machine assigned $M_{i,j}$ and the duration time $D_{i,j}$ are given, and constitute the definition of the problem instance. This problem aims to define the scheduling order of the operations such that the total execution period (makespan) is minimized.

Constraints. Several constraints have to be taken into account. The no-overlap constraint determines that each machine can process one operation at a time; once an operation initiates processing on a given machine, it must be completed without interruption. In addition, the precedence constraints establish that the order of operations inside a job J_i, an operation $O_{i,j}$ cannot be scheduled until the previous operation in the job $O_{i,j-1}$ finishes.

3.1 Learning the policy

The complete schedule consists of $n \cdot m$ dispatches that must be assigned sequentially. To this end, we build an autoregressive model that, given an instance of the problem x, iteratively constructs the solution, and the operations are dispatched one at a time according to the scheduling policy. This is, at a decision step t, the model $\pi_\theta(a|s_t, x_b)$ takes as an input the instance definition and the state s_t on the resolution process (state of the machines, or dynamic input of the model shown on 1 and outputs the probability distribution for taking action a_t at that time in the resolution process. The selected operation is scheduled, and the next state s_{t+1} is obtained. The process repeats until all pending operations are scheduled. At that point, the solution y is defined as the sequence of selected actions $a_1, a_2, ..., a_{n \cdot m}$.

We define the makespan as the maximum total execution time denoted by $\mathcal{T}^\pi(x)$. The reward function $R_t(s_t, a_t)$ is computed as the difference between execution times in two consecutive states s_t and s_{t+1}. That is, the sum of all collected rewards is equal to the makespan. Furthermore, we resort to policy gradient methods of Reinforce Algorithm in [26] to infer the model $\pi_\theta(a|s_t, x)$, such that $\min_\pi \mathbb{E}[\mathcal{T}^\pi(x)]$. We train the model updating the gradients as follows,

$$\nabla_\theta \hat{J}^\pi(\theta) \approx -\frac{1}{B} \sum_{b=1}^{B} \sum_{t=1}^{n \cdot m} \Big((G(s_t, x_b) - v_\phi(s_t, x_b)) \cdot \nabla_\theta \log \pi_\theta(a_t|s_t, x_b) \Big).$$

Additionally, in order to reduce the variance of the gradients, and therefore, to speed up the convergence, we also introduce a critic to estimate the baseline for the problem instances v_ϕ, which is parameterized using ϕ, and it is trained to minimize

$$L(\phi) = \frac{1}{B}\sum_{b=1}^{B}\sum_{t=1}^{n\cdot m}||v_\phi(s_t, x_b) - G(s_t, x_b)||^2,$$

where $v_\phi(s_t, x_b)$ is the estimate of the value function at state s_t, and

$$G(s_t, x_b) = \sum_{t}^{n\cdot m} R_t$$

is the actual accumulated expected reward starting at step t until all $n \cdot m$ operations are dispatched.

3.2 Inference strategies

So far, we have defined the policy function π, giving the distribution of possible actions at each step, as well as the learning algorithm. However, as discussed in the literature, many works rely on decoding strategies at inference to improve the greedy algorithm's results. In this paper, we consider the following strategies:

Sampling. In sampling strategy actions are randomly chosen from the policy distribution, i.e. $a_t \sim \pi_\theta(\cdot|s_t, x_b)$. This way, the proposed actions for the same state and instance may differ. In order for the *Sampling* strategy to outperform *Greedy*, it requires fine-tuning of the number of training iterations.

POMO. Policy Optimization with Multiple Optima (POMO) incorporates both ideas of *Greedy* and *Sampling*. At the initial state s_0, it rolls out several actions $\{a_0^{(1)}, a_0^{(2)}, ...\}$ creating several possible trajectories $\{s_1^{(1)}, s_1^{(2)}, ...\}$, and then developing these trajectories *greedily*. In [8], authors demonstrate that, for some combinatorial problems, this approach delivers better results when compared to *Sampling*.

Beam search. This strategy greedily chooses k actions $\{a_0^{(1)}, ..., a_0^{(k)}\}$ at the initial state s_0. This leads to forming k trajectories $\{s_1^{(1)}, ..., s_1^{(k)}\}$. Then, at each trajectory, new set of k actions is rolled out, and the total number of actions becomes $k \times k$. At this point, the likelihood of each action is calculated, and the strategy greedily chooses the next k actions. This way, *Beam* explores the k most-likely trajectories at-each-step.

4 Architecture

Top view of architecture. The architecture we propose in this paper is depicted in Fig. 1(a). This model is size-agnostic with regard to the input of a problem, demonstrates close-to-optimal performance, and generalizes well. Our model combines a deep learning preprocessing of input with an actor-critic network. The input consists of two branches: dynamic $s_t^{(2)}$ and static $O_{i,j}^{(2)}$. Dynamic input carries the information regarding the current load of the machines and their remaining times, while the static input propagates information operation-wise and job-wise within an instance of JSP. To make the model size agnostic, we pass both inputs through a fully-connected layer to get their multidimensional embeddings. Then static embedding is independently preprocessed in a recurrent manner and concatenated with dynamic embedding, followed by the pass into the RL network.

Job encoding. Considering that scheduling an operation at a time t depends only on the successive operations, we employ reverse LSTM shown in Fig. 1(b), which propagates information starting from the last operation in this job till the current operation. Here $e_{(i,j)}$ is the embedding of the operation $O_{i,j}$ that is currently to be scheduled, and $e_{(i,j+1)}$, $e_{(i,j+2)}$ are the embeddings of the following operations in the same job. The output of the reverse LSTM $e_{(i,j)}$ is lastly passed to *set2set* neural network.

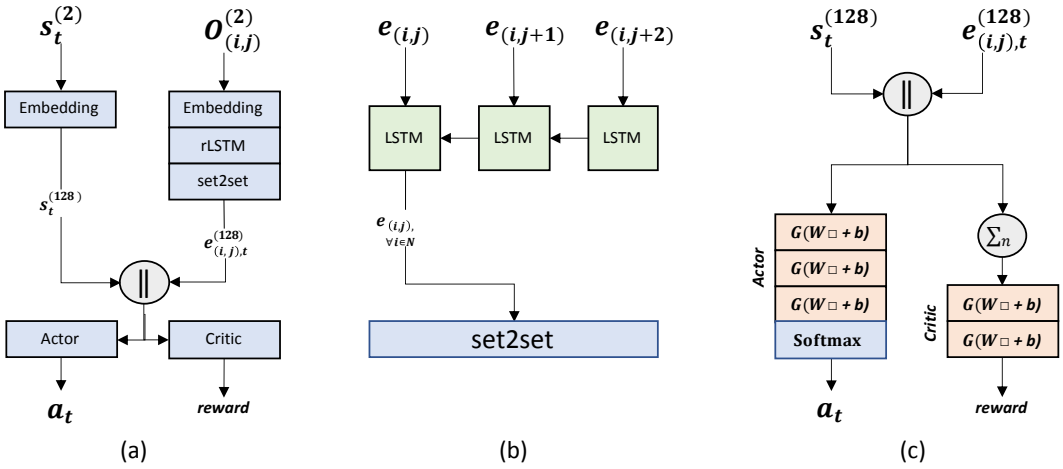

Figure 1: Deep learning model for the JSP. (a) Represents the top view of the model showing all the processes involved in the model from getting static and dynamic inputs of the instance till getting action and expected reward. (b) Represents the reverse LSTM block used to combine intra-job information by reversing operation embeddings. (c) Represents the concatenation of the static and dynamic embeddings and Actor-Critic block.

Set2set and dynamic embeddings. We use *set2set* [21] to combine the embeddings into a single global embedding disregarding the previous positioning in the jobs' sequence. Then, we attend to the fact that the problem instance on the input contains static information only. To follow the dynamic changes in the solution, we create two dynamic features for each machine: occupancy status at time t and time to finish the current operation. Both features are combined in $s_t^{(2)}$ dynamic input and pass through a fully-connected layer to get multidimensional embedding. In our model, dynamic embeddings are distributed over operations depending on which machine is the last in use in every job. Finally, both dynamic and static embeddings are concatenated and passed to the Actor-Critic network.

5 Improving generalization using Curriculum Learning

CL is a methodology for training on data of increasing difficulty that mimics the approach of social and educational systems to build a curricula [24]. There are several ways to construct the difficulty staircase and define rules for selecting the difficulty level during the training. In this work, we consider the following strategies:

Incremental Curriculum Learning (ICL). Incremental CL sequentially trains the model on increasing difficulty levels. In the case of the JSP, we learn a separate model for each problem size for a fixed number of learning steps. However, as mentioned in [10], the model is prone to experience catastrophic forgetting during the learning. Moreover, the number of models will equal the number of considered problem sizes (levels).

Uniform Curriculum Learning (UCL). In uniform CL, the model chooses at each iteration the problem level from a uniform distribution over considered sizes. It allows the model to learn policies inherent to different problem levels instead of incremental learning. However, as a rule, learning from tasks of small size is more accessible, so choosing a level from all sizes with uniform distribution at once may not be the most helpful learning strategy.

Adaptive Staircase Curriculum Learning (ASCL). Adaptive staircase is introduced for RL in [10]. This method is based on a supple choice of the next difficulty level. The model starts training from the smallest considered level $l = l_{min}$. Every i^{th} iteration, one checks the model's performance on test data. Depending on the model's performance on the current l data, the method proposes to go upwards to $l = l + 1$, decrease the difficulty to $l = l - 1$, or continue at l. It allows the model to start learning from the simplest policies and then sequentially increase the difficulty level. It

Table 1: Base learning on Taillard's instances. Columns represent different models trained on certain size, rows represent Taillard's data sizes. **Objective** shown as an average total time of schedules for a given size and **Gap** as an average percentage difference from optimal solutions (the less the gap the better result is)

	Instances		**(15×15)**	**(20×15)**	**(20×20)**	**(30×15)**	**(30×20)**
TAILLARD	15×15	Obj.	**1413.0**	1461.8	1452.6	1470.8	1481.5
		Gap	**(14.98%)**	(18.97%)	(18.23%)	(19.7%)	(20.59%)
	20×15	Obj.	**1606.6**	1692.1	1692.1	1696.3	1710.2
		Gap	**(17.69%)**	(23.97%)	(23.99%)	(24.29%)	(25.31%)
	20×20	Obj.	**1898.1**	1964.5	1958.4	1989.8	2029.4
		Gap	**(17.37%)**	(21.49%)	(21.11%)	(23.04%)	(25.47%)
	30×15	Obj.	**2153.0**	2251.8	2256.2	2275.3	2276.2
		Gap	**(20.41%)**	(25.86%)	(26.14%)	(27.19%)	(27.27%)
	30×20	Obj.	**2375.6**	2517.7	2503.8	2516.6	2540.2
		Gap	**(21.92%)**	(29.22%)	(28.52%)	(29.19%)	(30.4%)
	50×15	Obj.	**3207.0**	3356.7	3362.6	3336.4	3346.7
		Gap	**(15.67%)**	(21.07%)	(21.27%)	(20.33%)	(20.69%)
	50×20	Obj.	**3297.4**	3476.0	3473.8	3496.2	3518.0
		Gap	**(15.95%)**	(22.25%)	(22.15%)	(22.96%)	(23.73%)
	100×20	Obj.	**5879.9**	6120.9	6110.1	6153.5	6145.5
		Gap	**(9.58%)**	(14.06%)	(13.86%)	(14.67%)	(14.52%)

solves problems of incremental and unform learning strategies. However, one checks the model's performance at the current level and shifts level only with a step size equal to one. This approach does not estimate the model's generalization performance on other levels.

Adversarial Curriculum Learning (ACL). In this work, we propose ACL as a technique to improve the model's generalization performance. Like in ASCL, the agent is trained on a job with the current difficulty level l starting from the smallest problem size l_{min}. However, decisions to update a level are taken more adaptively. Every i_{th} iteration, one observes the performance (rewards) of the model on all considered problem levels. It calculates normalized probabilities of their percentage deviations from optimal solutions' rewards. Then, it randomly chooses the problem level from the list of levels up to the current one, following this distribution's probabilities. This sampling procedure is repeated on every iteration, and the model is trained on chosen level's instance. After getting a percentage gap on l size data less than a threshold value, the level is raised to $l + 1$. Otherwise, one decreases l to a random smaller size using the distribution of performance gap probabilities. This algorithm generalizes the ASCL method's decision of level backtrack and instance choice for training. Thus, ASCL is a particular case of ACL, where the latter allows the model to focus on the most adversarial sizes.

6 Experimentation

Datasets. We train and evaluate our model on scheduling instances of below sizes generated from Taillard's [19] and Demirkol's (DMU) [3] instances: 6 × 6, 10 × 10, 15 × 15, 20 × 15, 20 × 20, 30 × 20 and 30 × 15. To test a generalization of the model, we use substantially larger instances that are not included in the training set: 50 × 15, 50 × 20, and 100 × 20.

Baselines and base models. For the choice of *baseline models*, we resort to the analysis of priority dispatch rules (PDRs) in [16] and choose the best performing PDRs, as well as the most popular ones in the research community: Shortest Processing Time (SPT), Minimum Ratio of Flow Due Date to Most Work Remaining (FDD/WKR), Most Work Remaining (MWKR), Most Operations Remaining (MOPNR). Among neural combinatorial solvers, we use the latest state-of-the-art results of the Graph Neural Model on public JSP benchmarks presented in [27]. For comparison with the optimal solution, we refer to Google OR-Tools, the exact solver built on constraint programming. We also introduce the notion of *base model* which has the architecture shown at Fig. 1(a) and uses *Sampling* strategy for selection action.

Table 2: Incremental learning on Taillard's instances.

	Instances		(15×15)	(20×15)	(20×20)	(30×15)	(30×20)
TAILLARD	15×15	Obj.	1413.0	**1398.1**	1408.5	1431.1	1435.7
		Gap	(14.98%)	**(13.76%)**	(14.61%)	(16.48%)	(16.83%)
	20×15	Obj.	1606.6	**1588.1**	1597.7	1653.9	1657.1
		Gap	(17.69%)	**(16.34%)**	(17.06%)	(21.17%)	(21.41%)
	20×20	Obj.	1898.1	**1875.8**	1892.3	1929.8	1929.4
		Gap	(17.37%)	**(15.98%)**	(17.01%)	(19.34%)	(19.29%)
	30×15	Obj.	2153.0	**2123.7**	2163.4	2187.7	2229.6
		Gap	(20.41%)	**(18.78%)**	(20.99%)	(22.35%)	(24.66%)
	30×20	Obj.	2375.6	**2356.2**	2388.4	2429.1	2455.0
		Gap	(21.92%)	**(20.94%)**	(22.61%)	(24.68%)	(26.01%)
	50×15	Obj.	3207.0	**3189.0**	3226.0	3264.9	3261.8
		Gap	(15.67%)	**(15.02%)**	(16.35%)	(17.77%)	(17.64%)
	50×20	Obj.	3297.4	**3276.3**	3326.7	3337.8	3384.4
		Gap	(15.95%)	**(15.22%)**	(17.0%)	(17.37%)	(19.01%)
	100×20	Obj.	5879.9	**5851.9**	5949.0	5993.4	6036.8
		Gap	(9.58%)	**(9.05%)**	(10.86%)	(11.68%)	(12.5%)

Figure 2: CL and Base models results on Taillard's instances.

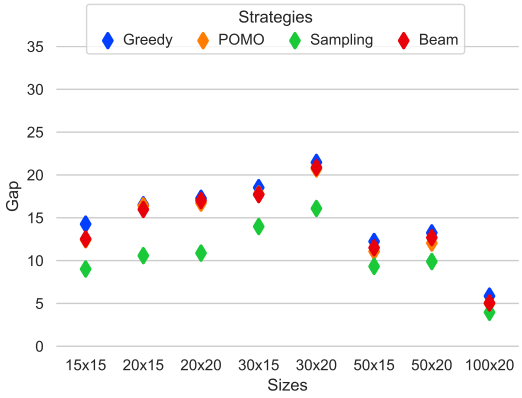

Figure 3: Different sampling strategies results on Taillard's instances.

6.1 Results

The implementation details can be found in Appendix A.

First, we train five instances of the base model on five small-to-medium sizes: 15x15, 20x15, 20x20, 30x15, and 30x20. Each instance of the base model is dedicated to one of five problem sizes. Then, we test all base model instances on an extended set of sizes, adding 50x15, 50x20, and 100x20 to assess generalization. The results in table 1 reflect dominating performance of the base model trained on the smallest size, 15x15. It shows the smallest optimality gap on all testing sizes, from 14.98% for 15x15 to 9.58% for 100x20. However, the model's generalization can be improved further, as the base model trained on a small size can hardly capture all combinatorial space of action policies inherent in large-size problems.

Next, we address generalization by incorporating and comparing different CL methods. We start with ICL, for which we train five instances of the base model in the following manner: the model assigned to the 15x15 size is trained on this size only, the model assigned to 20x15 is trained sequentially on the previous size, which in this case 15x15, and on its assigned 20x15 size. The same logic applies for all sizes up to 30x20, and each learning cycle takes n iterations. Table 2 shows the results of this approach. Here, the 20x15 model, being trained consecutively on 15x15 and 20x15 sizes, exhibits the best performance reducing the average gap to 9.05% for 100x20 problem instances. ICL

Table 3: Results of Baseline and ACL models on Taillard's insances.

	Instances		SPT	FDD/WKR	MWKR	MOPNR	Zhang et al.	Adversarial CL
TAILLARD	15×15	Obj.	1546.1	1808.6	1464.3	1481.3	1547.4	**1339.8**
		Gap	(25.89%)	(47.15%)	(19.15%)	(20.53%)	(25.96%)	**(9.02%)**
	20×15	Obj.	1813.5	2054.0	1683.6	1686.7	1774.7	**1509.3**
		Gap	(32.82%)	(50.57%)	(23.35%)	(23.55%)	(30.03%)	**(10.58%)**
	20×20	Obj.	2067.0	2387.2	1969.8	1968.3	2128.1	**1793.1**
		Gap	(27.75%)	(47.61%)	(21.81%)	(21.71%)	(31.61%)	**(10.87%)**
	30×15	Obj.	2419.3	2590.8	2214.8	2195.8	2378.8	**2038.1**
		Gap	(35.27%)	(45.02%)	(23.91%)	(22.83%)	(33.0%)	**(13.98%)**
	30×20	Obj.	2619.1	3045.0	2439.0	2433.6	2603.9	**2261.5**
		Gap	(34.44%)	(56.3%)	(25.17%)	(24.94%)	(33.62%)	**(16.09%)**
	50×15	Obj.	3441.0	3736.3	3240.0	3254.5	3393.8	**3030.8**
		Gap	(24.11%)	(34.77%)	(16.86%)	(17.37%)	(22.38%)	**(9.32%)**
	50×20	Obj.	3570.8	4022.1	3352.8	3346.9	3593.9	**3125.1**
		Gap	(25.54%)	(41.5%)	(17.95%)	(17.68%)	(26.51%)	**(9.89%)**
	100×20	Obj.	6139.0	6620.7	5812.2	5856.9	6097.6	**5578.9**
		Gap	(14.41%)	(23.39%)	(8.31%)	(9.15%)	(13.61%)	**(3.96%)**

also improves performance for all model instances on their subsequent problem sizes compared to corresponding base models. However, the ICL approach has an obvious drawback. This drawback comes from the fact that the best performance is achieved on a smaller-size model, 20x15, meaning that the larger-size models are not learning the policies.

To address the problem of ICL, we compare it to three other types of CL (Figure 2). One can see that UCL generally shows near-ICL behavior, while both adaptive and adversarial models demonstrate notably better performance. Particularly, ACL shows the smallest optimality gap on all test sizes, except for 15x15, where ASCL slightly outperforms ACL by 0.24%. The adversarial approach displays the smallest optimality gap from 10.58% for 20x15 instances to 3.96% for 100x20 instances.

We also provide a comparison of different inference strategies on ACL as the best-performing learning methodology. Figure 3 presents a comparison of *Greedy, POMO, Sampling and Beam* methods. For POMO and Beam, we use three possible trajectories. Both strategies demonstrate approximately similar results. The Greedy algorithm displays slightly worse performance because it is the special case of Beam search. The sampling strategy shows the best result on all test sizes, and, consequently, we use it for all experiments.

At this point, we incorporate ACL for training strategy and Sampling for selection strategy to compare its overall performance versus aforesaid baselines. Table 3 provides the comparison results on Taillard's dataset. The Adversarial model displays robust behavior on all test sizes, improving previous best results on average by 45.94%. This model shows a gap of 9.02% to 13.98% on all the sizes except for 100x20, where all compared models demonstrate relatively good performance. The reason for such observation may be the fact that Taiilard's dataset has relatively simple instances on 100x20 size.

To further test our model, we refer to DMU dataset consisting of 80 instances from 20x15 to 50x20 sizes. Table 4 shows outperforming behavior of Adversarial model again. On average, it improves previous best results by 50.95%. However, absolute values of the optimality gap are higher for this data and in the range of 14.66% to 25.42%. It may be the result of DMU benchmark consisting of more complex instances compared to Taillard. Results of other models also confirm this fact on all test sizes.

The complete experimentation on the Taillard and DMU instances is deferred to Appendix B.

7 Conclusions and Future Work

In this work, we present a deep-RL model to automatically learn heuristic dispatching rules on the JSP. To this end, we formulate the resolution process as a Markov Decision Process, in which solutions are iteratively constructed based on intermediate states of the resolution process. In this work, we present a model that is equivariant w.r.t. the job information and size-agnostic, i.e., it

Table 4: Results of Baseline and ACL models on DMU instances.

	Instances		SPT	FDD/WKR	MWKR	MOPNR	Zhang et al.	Adversarial CL
DMU	20×15	Obj.	4951.5	4666.3	4909.9	4513.2	4215.3	**3610.0**
		Gap	(64.13%)	(53.57%)	(62.15%)	(49.16%)	(38.95%)	**(19.36%)**
	20×20	Obj.	5690.5	5298.2	5489.0	5052.3	4804.5	**4028.9**
		Gap	(64.57%)	(52.52%)	(58.16%)	(45.17%)	(37.74%)	**(15.98%)**
	30×15	Obj.	6306.2	6016.5	6252.9	5742.8	5557.9	**4522.0**
		Gap	(62.57%)	(54.12%)	(60.95%)	(47.14%)	(41.86%)	**(16.35%)**
	30×20	Obj.	7036.0	6827.3	6925.0	6491.9	5967.4	**5106.0**
		Gap	(65.91%)	(60.09%)	(63.16%)	(51.97%)	(39.48%)	**(20.0%)**
	40×15	Obj.	7601.2	7420.0	7484.2	7105.5	6663.9	**5731.9**
		Gap	(55.88%)	(51.42%)	(52.87%)	(44.72%)	(35.38%)	**(17.49%)**
	40×20	Obj.	8538.1	8210.9	8460.9	7870.7	7375.8	**6584.1**
		Gap	(63.0%)	(55.52%)	(61.11%)	(49.22%)	(39.38%)	**(25.42%)**
	50×15	Obj.	8975.4	9150.2	8906.0	8436.5	8179.4	**7242.1**
		Gap	(50.37%)	(52.53%)	(48.93%)	(40.79%)	(36.2%)	**(21.54%)**
	50×20	Obj.	10132.8	9899.6	9807.0	9408.0	8751.6	**7176.9**
		Gap	(62.2%)	(57.26%)	(56.4%)	(49.61%)	(38.86%)	**(14.66%)**

enables us to train the model on instances of the JSP of different sizes. In order to improve the generalization of the model, this work uses Curriculum Learning. In this direction, we present a novel ACL strategy, which dynamically adjusts the difficulty of the learning instances according to the model's performance during the learning process. Experiments on Taillard's and Demirkol's instances show that our model improves the optimality gap w.r.t. the current state-of-the-art model by 45.94% and 50.95%, respectively. Our results corroborate that the contributions of this paper, the architecture and the ACL, improve the generalization on large sizes. There are several future directions for this work. For example, we would like to remark the potential of this technology for addressing stochastic or partially observable problems, domains where traditional approaches have shown limited capabilities.

References

[1] M Emin Aydin and Ercan Öztemel. Dynamic job-shop scheduling using reinforcement learning agents. *Robotics and Autonomous Systems*, 33(2-3):169–178, 2000.

[2] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016.

[3] Ebru Demirkol, Sanjay Mehta, and Reha Uzsoy. Benchmarks for shop scheduling problems. *European Journal of Operational Research*, 109(1):137–141, August 1998.

[4] Michel Deudon, Pierre Cournut, Alexandre Lacoste, Yossiri Adulyasak, and Louis-Martin Rousseau. Learning heuristics for the tsp by policy gradient. In *International conference on the integration of constraint programming, artificial intelligence, and operations research*, pages 170–181. Springer, 2018.

[5] Thomas Gabel and Martin Riedmiller. Adaptive reactive job-shop scheduling with reinforcement learning agents. *International Journal of Information Technology and Intelligent Computing*, 24(4):14–18, 2008.

[6] Helga Ingimundardottir and Thomas Philip Runarsson. Discovering dispatching rules from data using imitation learning: A case study for the job-shop problem. *Journal of Scheduling*, 21(4):413–428, 2018.

[7] Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475*, 2018.

[8] Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. Pomo: Policy optimization with multiple optima for reinforcement learning. *Advances in Neural Information Processing Systems*, 33:21188–21198, 2020.

[9] Chun-Cheng Lin, Der-Jiunn Deng, Yen-Ling Chih, and Hsin-Ting Chiu. Smart manufacturing scheduling with edge computing using multiclass deep q network. *IEEE Transactions on Industrial Informatics*, 15(7):4276–4284, 2019.

[10] Michal Lisicki, Arash Afkanpour, and Graham W Taylor. Evaluating curriculum learning strategies in neural combinatorial optimization. *arXiv preprint arXiv:2011.06188*, 2020.

[11] Chien-Liang Liu, Chuan-Chin Chang, and Chun-Jan Tseng. Actor-critic deep reinforcement learning for solving job shop scheduling problems. *Ieee Access*, 8:71752–71762, 2020.

[12] Sridhar Mahadevan, Nicholas Marchalleck, Tapas K Das, and Abhijit Gosavi. Self-improving factory simulation using continuous-time average-reward reinforcement learning. In *Machine Learning Interantional Workshop*, pages 202–210, 1997.

[13] Sridhar Mahadevan and Georgios Theocharous. Optimizing production manufacturing using reinforcement learning. In *FLAIRS conference*, volume 372, page 377, 1998.

[14] Hongzi Mao, Malte Schwarzkopf, Shaileshh Bojja Venkatakrishnan, Zili Meng, and Mohammad Alizadeh. Learning scheduling algorithms for data processing clusters. In *Proceedings of the ACM special interest group on data communication*, pages 270–288. 2019.

[15] Mohammadreza Nazari, Afshin Oroojlooy, Lawrence V Snyder, and Martin Takáč. Reinforcement learning for solving the vehicle routing problem. In *Conference on Neural Information Processing Systems, NeurIPS 2018*, 2018.

[16] Veronique Sels, Nele Gheysen, and Mario Vanhoucke. A comparison of priority rules for the job shop scheduling problem under different flow time-and tardiness-related objective functions. *International Journal of Production Research*, 50(15):4255–4270, 2012.

[17] Ruben Solozabal, Josu Ceberio, and Martin Takáč. Constrained combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:2006.11984*, 2020.

[18] Penghao Sun, Zehua Guo, Junchao Wang, Junfei Li, Julong Lan, and Yuxiang Hu. Deepweave: Accelerating job completion time with deep reinforcement learning-based coflow scheduling. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 3314–3320, 2021.

[19] E. Taillard. Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64(2):278–285, January 1993.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[21] Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

[22] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *Advances in neural information processing systems*, 28, 2015.

[23] Libing Wang, Xin Hu, Yin Wang, Sujie Xu, Shijun Ma, Kexin Yang, Zhijun Liu, and Weidong Wang. Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning. *Computer Networks*, 190:107969, 2021.

[24] Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[25] Bernd Waschneck, André Reichstaller, Lenz Belzner, Thomas Altenmüller, Thomas Bauernhansl, Alexander Knapp, and Andreas Kyek. Optimization of global production scheduling with deep reinforcement learning. *Procedia Cirp*, 72:1264–1269, 2018.

[26] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[27] Cong Zhang, Wen Song, Zhiguang Cao, Jie Zhang, Puay Siew Tan, and Xu Chi. Learning to dispatch for job shop scheduling via deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1621–1632, 2020.

[28] Di Zhang, Dong Dai, Youbiao He, Forrest Sheng Bao, and Bing Xie. Rlscheduler: an automated hpc batch job scheduler using reinforcement learning. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2020.

[29] Wei Zhang and Thomas G Dietterich. A reinforcement learning approach to job-shop scheduling. In *IJCAI*, volume 95, pages 1114–1120. Citeseer, 1995.

[30] Shuai Zheng, Chetan Gupta, and Susumu Serita. Manufacturing dispatching using reinforcement and transfer learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 655–671. Springer, 2019.

A Appendix

Models and configurations. In this paper, we train networks with hyperparameters being fixed for each problem size and validate each model on 1,000 randomly generated JSP instances which have the same sizes as in Taillard's dataset and remain unchanged throughout the training. For training of both Actor and Critic networks, we use Adam Optimizer with constant 10^{-4} learning rate and batch size of 128.

Our Base, UCL and ICL models are trained for 45000 iterations and then validated. Afterwards, the best performing model is saved. For ACL and ASCL training, every 100_{th} iteration the model is tested on a test data. If during the training, the model is not increasing a difficulty level for consecutive 3000 iterations, ACL algorithm decreases it.

Three random seeds are used during the inference and results are averaged to for final outcome. For Beam and POMO selection strategies, tree-width of 3 is used. For Sampling strategy, the sample size of 128 is used. Embedding size of 128x1 is used for both static and dynamic inputs.

The repository with Python code, results, and descriptions is presented in zip archive. Detailed instructions to reproduce training, evaluation, and potting results are provided in README.md file. The model is built in Python 3.9 using PyTorch library. JSP environment is created using the gym library. Full list of all employed packages and their versions is available in Requirements.txt file stored in the repository. Training is done on NVIDIA A100 SXM 40GB GPU with 2x AMD EPYC 7742 CPUs (128 cores total with 256 threads), and 256GB RAM.

Technical limitations of this work relate to high demand of GPU memory when training on large size JSP instances. This potential limitation may require to decrease significantly the batch size.

Ethical limitations. In this work, we use synthetic data from two know publicly available datasets, Taillard's and Demirkol's scheduling instances of JSP. Furthermore, our framework is built on principals of societal responsibility and respect towards fellow researchers.

Datasets. One can refer to the following publicly available sources:

- Taillard's dataset: http://optimizizer.com/TA.php
- Demirkol's dataset: http://optimizizer.com/DMU.php

B Appendix

Table 5: CL strategies on Taillard's instances. Columns represent models trained using different Curriculum strategies as well as baseline Zhang's, ICL and base models, rows represent Taillard's data sizes. **Objective** shown as an average total time of schedules for a given size and **Gap** as an average percentage difference from optimal solutions (the less the gap the better result is)

	Instances		Zhang et al.	ICL	Base	UCL	ASCL	ACL
TAILLARD	15×15	Obj.	1547.4	1413.0	1413.0	1406.1	**1336.6**	1339.8
		Gap	(25.96%)	(14.98%)	(14.98%)	(14.45%)	**(8.78%)**	(9.02%)
	20×15	Obj.	1774.7	1588.1	1692.1	1627.7	1546.2	**1509.3**
		Gap	(30.03%)	(16.34%)	(23.97%)	(19.27%)	(13.27%)	**(10.58%)**
	20×20	Obj.	2128.1	1892.3	1958.4	1892.1	1809.8	**1793.1**
		Gap	(31.61%)	(17.01%)	(21.11%)	(17.01%)	(11.91%)	**(10.87%)**
	30×15	Obj.	2378.8	2187.7	2275.3	2179.3	2068.1	**2038.1**
		Gap	(33.0%)	(22.35%)	(27.19%)	(21.87%)	(15.66%)	**(13.98%)**
	30×20	Obj.	2603.9	2455.0	2540.2	2419.5	2281.0	**2261.5**
		Gap	(33.62%)	(26.01%)	(30.4%)	(24.2%)	(17.09%)	**(16.09%)**
	50×15	Obj.	3393.8	3261.8	3346.7	3187.5	3088.1	**3030.8**
		Gap	(22.38%)	(17.64%)	(20.69%)	(14.95%)	(11.37%)	**(9.32%)**
	50×20	Obj.	3593.9	3384.4	3518.0	3348.1	3159.4	**3125.1**
		Gap	(26.51%)	(19.01%)	(23.73%)	(17.75%)	(11.1%)	**(9.89%)**
	100×20	Obj.	6097.6	6036.8	6145.5	5843.1	5682.1	**5578.9**
		Gap	(13.61%)	(12.5%)	(14.52%)	(8.89%)	(5.89%)	**(3.96%)**

Table 6: Selection strategies on Taillard's instances.

	Instances		Greedy	POMO	Sampling	Beam
TAILLARD	15×15	Obj.	1404.2	1381.9	**1339.8**	1382.9
		Gap	(14.26%)	(12.43%)	**(9.02%)**	(12.52%)
	20×15	Obj.	1590.3	1588.6	**1509.3**	1582.9
		Gap	(16.52%)	(16.4%)	**(10.58%)**	(15.97%)
	20×20	Obj.	1896.5	1887.5	**1793.1**	1892.1
		Gap	(17.27%)	(16.71%)	**(10.87%)**	(17.01%)
	30×15	Obj.	2118.9	2105.0	**2038.1**	2104.8
		Gap	(18.52%)	(17.78%)	**(13.98%)**	(17.7%)
	30×20	Obj.	2365.7	2350.8	**2261.5**	2354.0
		Gap	(21.47%)	(20.7%)	**(16.09%)**	(20.87%)
	50×15	Obj.	3111.6	3079.1	**3030.8**	3090.9
		Gap	(12.23%)	(11.08%)	**(9.32%)**	(11.5%)
	50×20	Obj.	3219.6	3185.9	**3125.1**	3203.4
		Gap	(13.24%)	(12.05%)	**(9.89%)**	(12.68%)
	100×20	Obj.	5680.9	5633.3	**5578.9**	5637.5
		Gap	(5.86%)	(4.97%)	**(3.96%)**	(5.05%)

Table 7: Selection strategies on Taillard's instances.

	Instance	Greedy	POMO	Sampling	Beam	UB
15×15	TA1	1445 (17.38%)	1445 (17.38%)	**1379 (12.02%)**	1421 (15.43%)	1231*
	TA2	1405 (12.94%)	1405 (12.94%)	**1325 (6.51%)**	1390 (11.74%)	1244*
	TA3	1469 (20.61%)	1344 (10.34%)	**1338 (9.85%)**	1405 (15.35%)	1218*
	TA4	1304 (10.98%)	1304 (10.98%)	**1275 (8.51%)**	1304 (10.98%)	1175*
	TA5	1345 (9.89%)	1345 (9.89%)	**1303 (6.45%)**	1326 (8.33%)	1224*
	TA6	1360 (9.85%)	1329 (7.35%)	**1325 (7.03%)**	1326 (7.11%)	1238*
	TA7	1426 (16.22%)	1426 (16.22%)	**1332 (8.56%)**	1400 (14.1%)	1227*
	TA8	1422 (16.84%)	1355 (11.34%)	**1325 (8.87%)**	1391 (14.3%)	1217*
	TA9	1460 (14.6%)	1460 (14.6%)	**1412 (10.83%)**	1460 (14.6%)	1274*
	TA10	1406 (13.3%)	1406 (13.3%)	**1384 (11.52%)**	1406 (13.3%)	1241*
20×15	TA11	1623 (19.6%)	1623 (19.6%)	**1506 (10.98%)**	1627 (19.9%)	1357*
	TA12	1587 (16.09%)	1587 (16.09%)	**1514 (10.75%)**	1541 (12.73%)	1367*
	TA13	1617 (20.4%)	1617 (20.4%)	**1510 (12.43%)**	1614 (20.18%)	1343*
	TA14	1548 (15.09%)	1548 (15.09%)	**1455 (8.18%)**	1548 (15.09%)	1345*
	TA15	1546 (15.46%)	1546 (15.46%)	**1516 (13.22%)**	1517 (13.29%)	1339*
	TA16	1584 (16.47%)	1584 (16.47%)	**1485 (9.19%)**	1584 (16.47%)	1360*
	TA17	1698 (16.14%)	1698 (16.14%)	**1614 (10.4%)**	1698 (16.14%)	1462*
	TA18	1614 (15.62%)	1597 (14.4%)	**1560 (11.75%)**	1614 (15.62%)	1396*
	TA19	1561 (17.19%)	1561 (17.19%)	**1451 (8.93%)**	1561 (17.19%)	1332*
	TA20	1525 (13.13%)	1525 (13.13%)	**1482 (9.94%)**	1525 (13.13%)	1348*
20×20	TA21	1913 (16.5%)	1913 (16.5%)	**1838 (11.94%)**	1913 (16.5%)	1642*
	TA22	1853 (15.81%)	1853 (15.81%)	**1752 (9.5%)**	1811 (13.19%)	1600*
	TA23	1854 (19.08%)	1837 (17.98%)	**1748 (12.27%)**	1898 (21.9%)	1557*
	TA24	1907 (16.0%)	1904 (15.82%)	**1807 (9.91%)**	1905 (15.88%)	1644*
	TA25	1903 (19.31%)	1903 (19.31%)	**1772 (11.1%)**	1922 (20.5%)	1595*
	TA26	1944 (18.32%)	1898 (15.52%)	**1802 (9.68%)**	1944 (18.32%)	1643*
	TA27	2001 (19.11%)	2001 (19.11%)	**1895 (12.8%)**	1998 (18.93%)	1680*
	TA28	1826 (13.91%)	1826 (13.91%)	**1788 (11.54%)**	1810 (12.91%)	1603*
	TA29	1839 (13.17%)	1839 (13.17%)	**1753 (7.88%)**	1802 (10.89%)	1625*
	TA30	1925 (21.53%)	1901 (20.01%)	**1776 (12.12%)**	1918 (21.09%)	1584*
30×15	TA31	2105 (19.33%)	2105 (19.33%)	**2028 (14.97%)**	2092 (18.59%)	1764*
	TA32	2163 (21.24%)	2163 (21.24%)	**2092 (17.26%)**	2183 (22.37%)	1784*
	TA33	2225 (24.23%)	2175 (21.44%)	**2114 (18.03%)**	2203 (23.0%)	1791*
	TA34	2108 (15.32%)	2108 (15.32%)	**2077 (13.62%)**	2108 (15.32%)	1828*
	TA35	2172 (8.22%)	2115 (5.38%)	**2103 (4.78%)**	2172 (8.22%)	2007*
	TA36	2124 (16.77%)	2092 (15.01%)	**2046 (12.48%)**	2124 (16.77%)	1819*
	TA37	2208 (24.68%)	2208 (24.68%)	**2080 (17.45%)**	2138 (20.72%)	1771*
	TA38	1943 (16.14%)	1943 (16.14%)	**1882 (12.49%)**	1928 (15.24%)	1673*
	TA39	2106 (17.33%)	2106 (17.33%)	**2029 (13.04%)**	2104 (17.21%)	1795*
	TA40	2035 (21.93%)	2035 (21.93%)	**1930 (15.64%)**	1996 (19.59%)	1669*
30×20	TA41	2440 (21.7%)	2440 (21.7%)	**2356 (17.51%)**	2428 (21.1%)	2005*
	TA42	2375 (22.61%)	2375 (22.61%)	**2260 (16.68%)**	2363 (21.99%)	1937*
	TA43	2352 (27.41%)	2312 (25.24%)	**2163 (17.17%)**	2366 (28.17%)	1846*
	TA44	2464 (24.51%)	2393 (20.92%)	**2293 (15.87%)**	2440 (23.29%)	1979*
	TA45	**2248 (12.4%)**	2248 (12.4%)	2250 (12.5%)	2248 (12.4%)	2000*
	TA46	2410 (20.14%)	2410 (20.14%)	**2314 (15.35%)**	2391 (19.19%)	2006*
	TA47	2211 (17.05%)	2211 (17.05%)	**2141 (13.34%)**	2211 (17.05%)	1889*
	TA48	2350 (21.32%)	2350 (21.32%)	**2289 (18.17%)**	2327 (20.13%)	1937*
	TA49	2415 (23.15%)	2377 (21.21%)	**2270 (15.76%)**	2415 (23.15%)	1961*
	TA50	2392 (24.39%)	2392 (24.39%)	**2279 (18.51%)**	2351 (22.26%)	1923*
50×15	TA51	3295 (19.38%)	3214 (16.45%)	**3181 (15.25%)**	3279 (18.8%)	2760*
	TA52	3082 (11.83%)	3082 (11.83%)	**3029 (9.91%)**	3029 (9.91%)	2756*
	TA53	2944 (8.35%)	2935 (8.02%)	**2921 (7.51%)**	2945 (8.39%)	2717*
	TA54	2924 (2.99%)	2924 (2.99%)	2914 (2.64%)	**2904 (2.29%)**	2839*
	TA55	3108 (16.01%)	3108 (16.01%)	**3050 (13.85%)**	3184 (18.85%)	2679*
	TA56	3201 (15.1%)	3104 (11.61%)	**3012 (8.31%)**	3176 (14.2%)	2781*
	TA57	3315 (12.64%)	3224 (9.55%)	**3180 (8.05%)**	3247 (10.33%)	2943*
	TA58	3143 (8.94%)	3143 (8.94%)	**3103 (7.56%)**	3141 (8.87%)	2885*
	TA59	3100 (16.76%)	3091 (16.42%)	**3009 (13.33%)**	3049 (14.84%)	2655*
	TA60	3004 (10.32%)	2966 (8.92%)	**2909 (6.83%)**	2955 (8.52%)	2723*
50×20	TA61	3242 (13.04%)	**3179 (10.84%)**	3184 (11.02%)	3240 (12.97%)	2868*
	TA62	3309 (15.34%)	3292 (14.74%)	**3229 (12.55%)**	3286 (14.53%)	2869*
	TA63	3069 (11.4%)	3059 (11.03%)	**3004 (9.04%)**	3117 (13.14%)	2755*
	TA64	3044 (12.66%)	3010 (11.4%)	**2916 (7.92%)**	3020 (11.77%)	2702*
	TA65	3256 (19.49%)	3141 (15.27%)	**3068 (12.59%)**	3236 (18.75%)	2725*
	TA66	3249 (14.2%)	3224 (13.32%)	**3144 (10.51%)**	3243 (13.99%)	2845*
	TA67	3194 (13.06%)	3194 (13.06%)	**3120 (10.44%)**	3191 (12.96%)	2825*
	TA68	3040 (9.2%)	3030 (8.84%)	**2955 (6.14%)**	2992 (7.47%)	2784*
	TA69	3383 (10.16%)	3320 (8.11%)	**3295 (7.29%)**	3344 (8.89%)	3071*
	TA70	3410 (13.86%)	3410 (13.86%)	**3336 (11.39%)**	3365 (12.35%)	2995*
100×20	TA71	5769 (5.58%)	5694 (4.21%)	**5653 (3.46%)**	5709 (4.48%)	5464*
	TA72	5462 (5.42%)	5366 (3.57%)	**5365 (3.55%)**	5455 (5.29%)	5181*
	TA73	6052 (8.69%)	5982 (7.44%)	**5885 (5.69%)**	6040 (8.48%)	5568*
	TA74	5510 (3.2%)	5472 (2.49%)	5427 (1.65%)	**5352 (0.24%)**	5339*
	TA75	5850 (8.49%)	5833 (8.18%)	**5790 (7.38%)**	5837 (8.25%)	5392*
	TA76	5859 (9.68%)	5858 (9.66%)	**5748 (7.6%)**	5935 (11.1%)	5342*
	TA77	5677 (4.43%)	5657 (4.07%)	**5588 (2.8%)**	5663 (4.18%)	5436*
	TA78	5699 (5.65%)	5604 (3.89%)	5570 (3.26%)	**5541 (2.73%)**	5394*
	TA79	5548 (3.55%)	5484 (2.35%)	**5424 (1.23%)**	5488 (2.43%)	5358*
	TA80	5383 (3.86%)	5383 (3.86%)	**5339 (3.01%)**	5355 (3.32%)	5183*

Table 8: ICL on Taillard's instances.

	Instance	(15×15)	(20×15)	(20×20)	(30×15)	(30×20)	UB
15×15	TA1	1475 (19.82%)	1457 (18.36%)	1458 (18.44%)	**1453 (18.03%)**	1477 (19.98%)	1231*
	TA2	**1401 (12.62%)**	1414 (13.67%)	1410 (13.34%)	1424 (14.47%)	1426 (14.63%)	1244*
	TA3	**1393 (14.37%)**	1397 (14.7%)	1413 (16.01%)	1467 (20.44%)	1447 (18.8%)	1218*
	TA4	1340 (14.04%)	**1338 (13.87%)**	1351 (14.98%)	1412 (20.17%)	1374 (16.94%)	1175*
	TA5	1390 (13.56%)	**1354 (10.62%)**	1404 (14.71%)	1433 (17.08%)	1388 (13.4%)	1224*
	TA6	1388 (12.12%)	**1382 (11.63%)**	1389 (12.2%)	1401 (13.17%)	1414 (14.22%)	1238*
	TA7	1425 (16.14%)	**1370 (11.65%)**	1374 (11.98%)	1408 (14.75%)	1479 (20.54%)	1227*
	TA8	1407 (15.61%)	1388 (14.05%)	**1369 (12.49%)**	1418 (16.52%)	1383 (13.64%)	1217*
	TA9	**1472 (15.54%)**	1478 (16.01%)	1491 (17.03%)	1488 (16.8%)	1502 (17.9%)	1274*
	TA10	1439 (15.95%)	**1403 (13.05%)**	1426 (14.91%)	1407 (13.38%)	1467 (18.21%)	1241*
20×15	TA11	1602 (18.05%)	**1576 (16.14%)**	1605 (18.28%)	1674 (23.36%)	1674 (23.36%)	1357*
	TA12	1604 (17.34%)	1596 (16.75%)	**1582 (15.73%)**	1640 (19.97%)	1612 (17.92%)	1367*
	TA13	1612 (20.03%)	1569 (16.83%)	**1564 (16.46%)**	1659 (23.53%)	1631 (21.44%)	1343*
	TA14	1562 (16.13%)	**1535 (14.13%)**	1545 (14.87%)	1615 (20.07%)	1626 (20.89%)	1345*
	TA15	1559 (16.43%)	1554 (16.06%)	**1549 (15.68%)**	1604 (19.79%)	1602 (19.64%)	1339*
	TA16	1596 (17.35%)	**1582 (16.32%)**	1592 (17.06%)	1610 (18.38%)	1665 (22.43%)	1360*
	TA17	1723 (17.85%)	1705 (16.62%)	**1682 (15.05%)**	1759 (20.31%)	1754 (19.97%)	1462*
	TA18	1709 (22.42%)	**1662 (19.05%)**	1690 (21.06%)	1747 (25.14%)	1757 (25.86%)	1396*
	TA19	**1544 (15.92%)**	1544 (15.92%)	1546 (16.07%)	1607 (20.65%)	1610 (20.87%)	1332*
	TA20	**1555 (15.36%)**	1558 (15.58%)	1622 (20.33%)	1624 (20.47%)	1640 (21.66%)	1348*
20×20	TA21	1947 (18.57%)	**1907 (16.14%)**	1967 (19.79%)	1926 (17.3%)	1946 (18.51%)	1642*
	TA22	1896 (18.5%)	**1862 (16.38%)**	1865 (16.56%)	1879 (17.44%)	1863 (16.44%)	1600*
	TA23	1856 (19.2%)	**1819 (16.83%)**	1870 (20.1%)	1905 (22.35%)	1870 (20.1%)	1557*
	TA24	1928 (17.27%)	**1887 (14.78%)**	1898 (15.45%)	1958 (19.1%)	1943 (18.19%)	1644*
	TA25	1883 (18.06%)	1871 (17.3%)	**1866 (16.99%)**	1943 (21.82%)	1948 (22.13%)	1595*
	TA26	1885 (14.73%)	**1874 (14.06%)**	1928 (17.35%)	1967 (19.72%)	1953 (18.87%)	1643*
	TA27	1987 (18.27%)	1983 (18.04%)	**1980 (17.86%)**	2023 (20.42%)	2044 (21.67%)	1680*
	TA28	1850 (15.41%)	**1821 (13.6%)**	1841 (14.85%)	1884 (17.53%)	1896 (18.28%)	1603*
	TA29	1896 (16.68%)	1886 (16.06%)	**1856 (14.22%)**	1891 (16.37%)	1954 (20.25%)	1625*
	TA30	1853 (16.98%)	**1848 (16.67%)**	1852 (16.92%)	1922 (21.34%)	1877 (18.5%)	1584*
30×15	TA31	2148 (21.77%)	**2101 (19.1%)**	2167 (22.85%)	2149 (21.83%)	2302 (30.5%)	1764*
	TA32	**2164 (21.3%)**	2179 (22.14%)	2172 (21.75%)	2216 (24.22%)	2339 (31.11%)	1784*
	TA33	2260 (26.19%)	**2173 (21.33%)**	2252 (25.74%)	2350 (31.21%)	2310 (28.98%)	1791*
	TA34	2169 (18.65%)	**2155 (17.89%)**	2202 (20.46%)	2166 (18.49%)	2179 (19.2%)	1828*
	TA35	2209 (10.06%)	**2179 (8.57%)**	2253 (12.26%)	2235 (11.36%)	2320 (15.6%)	2007*
	TA36	2190 (20.4%)	**2157 (18.58%)**	2192 (20.51%)	2248 (23.58%)	2272 (24.9%)	1819*
	TA37	2157 (21.8%)	2145 (21.12%)	**2136 (20.61%)**	2251 (27.1%)	2239 (26.43%)	1771*
	TA38	2026 (21.1%)	**2018 (20.62%)**	2042 (22.06%)	2036 (21.7%)	2056 (22.89%)	1673*
	TA39	2190 (22.01%)	**2144 (19.44%)**	2163 (20.5%)	2194 (22.23%)	2230 (24.23%)	1795*
	TA40	2017 (20.85%)	**1986 (18.99%)**	2055 (23.13%)	2032 (21.75%)	2049 (22.77%)	1669*
30×20	TA41	2499 (24.64%)	**2469 (23.14%)**	2492 (24.29%)	2491 (24.24%)	2543 (26.83%)	2005*
	TA42	**2295 (18.48%)**	2321 (19.82%)	2337 (20.65%)	2363 (21.99%)	2399 (23.85%)	1937*
	TA43	2260 (22.43%)	**2232 (20.91%)**	2323 (25.84%)	2312 (25.24%)	2332 (26.33%)	1846*
	TA44	2334 (17.94%)	**2326 (17.53%)**	2420 (22.28%)	2445 (23.55%)	2506 (26.63%)	1979*
	TA45	2432 (21.6%)	2385 (19.25%)	**2381 (19.05%)**	2491 (24.55%)	2470 (23.5%)	2000*
	TA46	2501 (24.68%)	2439 (21.59%)	**2432 (21.24%)**	2517 (25.47%)	2569 (28.07%)	2006*
	TA47	2258 (19.53%)	2262 (19.75%)	**2257 (19.48%)**	2349 (24.35%)	2398 (26.95%)	1889*
	TA48	2389 (23.34%)	2397 (23.75%)	**2378 (22.77%)**	2479 (27.98%)	2471 (27.57%)	1937*
	TA49	2413 (23.05%)	**2376 (21.16%)**	2455 (25.19%)	2419 (23.36%)	2428 (23.81%)	1961*
	TA50	2375 (23.5%)	**2355 (22.46%)**	2409 (25.27%)	2425 (26.11%)	2434 (26.57%)	1923*
50×15	TA51	3409 (23.51%)	3418 (23.84%)	3448 (24.93%)	3518 (27.46%)	**3394 (22.97%)**	2760*
	TA52	3245 (17.74%)	**3184 (15.53%)**	3203 (16.22%)	3313 (20.21%)	3337 (21.08%)	2756*
	TA53	**3078 (13.29%)**	3085 (13.54%)	3083 (13.47%)	3138 (15.5%)	3122 (14.91%)	2717*
	TA54	3131 (10.29%)	**3070 (8.14%)**	3131 (10.29%)	3218 (13.35%)	3163 (11.41%)	2839*
	TA55	3217 (20.08%)	3192 (19.15%)	**3172 (18.4%)**	3221 (20.23%)	3247 (21.2%)	2679*
	TA56	3123 (12.3%)	**3094 (11.25%)**	3207 (15.32%)	3190 (14.71%)	3290 (18.3%)	2781*
	TA57	3332 (13.22%)	**3331 (13.18%)**	3384 (14.98%)	3354 (13.97%)	3437 (16.79%)	2943*
	TA58	3324 (15.22%)	3318 (15.01%)	**3300 (14.38%)**	3347 (16.01%)	3334 (15.56%)	2885*
	TA59	3144 (18.42%)	**3122 (17.59%)**	3179 (19.74%)	3226 (21.51%)	3196 (20.38%)	2655*
	TA60	**3067 (12.63%)**	3076 (12.96%)	3153 (15.79%)	3124 (14.73%)	3098 (13.77%)	2723*
50×20	TA61	3357 (17.05%)	**3324 (15.9%)**	3366 (17.36%)	3406 (18.76%)	3495 (21.86%)	2868*
	TA62	3381 (17.85%)	**3356 (16.97%)**	3383 (17.92%)	3411 (18.89%)	3496 (21.85%)	2869*
	TA63	**3162 (14.77%)**	3162 (14.77%)	3237 (17.5%)	3225 (17.06%)	3232 (17.31%)	2755*
	TA64	3117 (15.36%)	**3038 (12.44%)**	3124 (15.62%)	3097 (14.62%)	3164 (17.1%)	2702*
	TA65	**3301 (21.14%)**	3345 (22.75%)	3343 (22.68%)	3328 (22.13%)	3314 (21.61%)	2725*
	TA66	3272 (15.01%)	**3248 (14.17%)**	3309 (16.31%)	3329 (17.01%)	3415 (20.04%)	2845*
	TA67	3231 (14.37%)	**3209 (13.59%)**	3295 (16.64%)	3378 (19.58%)	3442 (21.84%)	2825*
	TA68	**3101 (11.39%)**	3104 (11.49%)	3153 (13.25%)	3118 (12.0%)	3174 (14.01%)	2784*
	TA69	3538 (15.21%)	**3476 (13.19%)**	3498 (13.9%)	3556 (15.79%)	3559 (15.89%)	3071*
	TA70	3514 (17.33%)	**3501 (16.89%)**	3559 (18.83%)	3530 (17.86%)	3553 (18.63%)	2995*
100×20	TA71	5990 (9.63%)	**5989 (9.61%)**	6055 (10.82%)	6142 (12.41%)	6238 (14.17%)	5464*
	TA72	5679 (9.61%)	**5654 (9.13%)**	5746 (10.91%)	5765 (11.27%)	5771 (11.39%)	5181*
	TA73	**6144 (10.34%)**	6197 (11.3%)	6326 (13.61%)	6325 (13.6%)	6294 (13.04%)	5568*
	TA74	5680 (6.39%)	**5658 (5.97%)**	5829 (9.18%)	5794 (8.52%)	5859 (9.74%)	5339*
	TA75	6125 (13.59%)	**6069 (12.56%)**	6095 (13.04%)	6273 (16.34%)	6294 (16.73%)	5392*
	TA76	6033 (12.94%)	6035 (12.97%)	**6032 (12.92%)**	6077 (13.76%)	6060 (13.44%)	5342*
	TA77	5865 (7.89%)	**5792 (6.55%)**	5945 (9.36%)	5859 (7.78%)	5935 (9.18%)	5436*
	TA78	5818 (7.86%)	**5794 (7.42%)**	5923 (9.81%)	6039 (11.96%)	6153 (14.07%)	5394*
	TA79	5848 (9.15%)	**5794 (8.14%)**	5834 (8.88%)	5955 (11.14%)	5945 (10.96%)	5358*
	TA80	5617 (8.37%)	**5537 (6.83%)**	5705 (10.07%)	5705 (10.07%)	5819 (12.27%)	5183*

Table 9: Base learning on Taillard's instances.

	Instance	(15×15)	(20×15)	(20×20)	(30×15)	(30×20)	UB
15×15	TA1	1475 (19.82%)	1492 (21.2%)	1466 (19.09%)	**1443 (17.22%)**	1477 (19.98%)	1231*
	TA2	**1401 (12.62%)**	1429 (14.87%)	1437 (15.51%)	1463 (17.6%)	1445 (16.16%)	1244*
	TA3	**1393 (14.37%)**	1476 (21.18%)	1470 (20.69%)	1484 (21.84%)	1481 (21.59%)	1218*
	TA4	**1340 (14.04%)**	1437 (22.3%)	1435 (22.13%)	1470 (25.11%)	1513 (28.77%)	1175*
	TA5	**1390 (13.56%)**	1409 (15.11%)	1451 (18.55%)	1464 (19.61%)	1486 (21.41%)	1224*
	TA6	**1388 (12.12%)**	1408 (13.73%)	1413 (14.14%)	1444 (16.64%)	1421 (14.78%)	1238*
	TA7	1425 (16.14%)	1453 (18.42%)	**1398 (13.94%)**	1463 (19.23%)	1482 (20.78%)	1227*
	TA8	**1407 (15.61%)**	1498 (23.09%)	1458 (19.8%)	1441 (18.41%)	1457 (19.72%)	1217*
	TA9	**1472 (15.54%)**	1543 (21.11%)	1503 (17.97%)	1560 (22.45%)	1600 (25.59%)	1274*
	TA10	**1439 (15.95%)**	1473 (18.69%)	1495 (20.47%)	1476 (18.94%)	1453 (17.08%)	1241*
20×15	TA11	**1602 (18.05%)**	1671 (23.14%)	1687 (24.32%)	1731 (27.56%)	1772 (30.58%)	1357*
	TA12	**1604 (17.34%)**	1678 (22.75%)	1653 (20.92%)	1652 (20.85%)	1649 (20.63%)	1367*
	TA13	**1612 (20.03%)**	1691 (25.91%)	1680 (25.09%)	1682 (25.24%)	1736 (29.26%)	1343*
	TA14	**1562 (16.13%)**	1685 (25.28%)	1687 (25.43%)	1705 (26.77%)	1664 (23.72%)	1345*
	TA15	**1559 (16.43%)**	1675 (25.09%)	1693 (26.44%)	1696 (26.66%)	1666 (24.42%)	1339*
	TA16	**1596 (17.35%)**	1689 (24.19%)	1670 (22.79%)	1657 (21.84%)	1695 (24.63%)	1360*
	TA17	**1723 (17.85%)**	1800 (23.12%)	1795 (22.78%)	1824 (24.76%)	1837 (25.65%)	1462*
	TA18	1709 (22.42%)	1785 (27.87%)	1730 (23.93%)	**1701 (21.85%)**	1723 (23.42%)	1396*
	TA19	**1544 (15.92%)**	1615 (21.25%)	1649 (23.8%)	1674 (25.68%)	1701 (27.7%)	1332*
	TA20	**1555 (15.36%)**	1632 (21.07%)	1677 (24.41%)	1641 (21.74%)	1659 (23.07%)	1348*
20×20	TA21	1947 (18.57%)	1949 (18.7%)	**1944 (18.39%)**	1975 (20.28%)	2063 (25.64%)	1642*
	TA22	1896 (18.5%)	**1894 (18.38%)**	1937 (21.06%)	1938 (21.12%)	1950 (21.88%)	1600*
	TA23	**1856 (19.2%)**	1934 (24.21%)	1903 (22.22%)	1935 (24.28%)	1921 (23.38%)	1557*
	TA24	**1928 (17.27%)**	1989 (20.99%)	1958 (19.1%)	1984 (20.68%)	1999 (21.59%)	1644*
	TA25	**1883 (18.06%)**	1989 (24.7%)	2006 (25.77%)	2035 (27.59%)	2094 (31.29%)	1595*
	TA26	**1885 (14.73%)**	2005 (22.03%)	1971 (19.96%)	2051 (24.83%)	2068 (25.87%)	1643*
	TA27	**1987 (18.27%)**	2049 (21.96%)	2066 (22.98%)	2076 (23.57%)	2146 (27.74%)	1680*
	TA28	**1850 (15.41%)**	1928 (20.27%)	1924 (20.02%)	1939 (20.96%)	2000 (24.77%)	1603*
	TA29	**1896 (16.68%)**	1923 (18.34%)	1926 (18.52%)	2009 (23.63%)	2068 (27.26%)	1625*
	TA30	**1853 (16.98%)**	1985 (25.32%)	1949 (23.04%)	1956 (23.48%)	1985 (25.32%)	1584*
30×15	TA31	**2148 (21.77%)**	2227 (26.25%)	2191 (24.21%)	2263 (28.29%)	2295 (30.1%)	1764*
	TA32	**2164 (21.3%)**	2304 (29.15%)	2332 (30.72%)	2301 (28.98%)	2305 (29.2%)	1784*
	TA33	**2260 (26.19%)**	2361 (31.83%)	2394 (33.67%)	2410 (34.56%)	2383 (33.05%)	1791*
	TA34	**2169 (18.65%)**	2278 (24.62%)	2258 (23.52%)	2317 (26.75%)	2299 (25.77%)	1828*
	TA35	**2209 (10.06%)**	2379 (18.54%)	2353 (17.24%)	2389 (19.03%)	2341 (16.64%)	2007*
	TA36	**2190 (20.4%)**	2326 (27.87%)	2318 (27.43%)	2326 (27.87%)	2327 (27.93%)	1819*
	TA37	**2157 (21.8%)**	2265 (27.89%)	2308 (30.32%)	2329 (31.51%)	2349 (32.64%)	1771*
	TA38	**2026 (21.1%)**	2070 (23.73%)	2092 (25.04%)	2108 (26.0%)	2095 (25.22%)	1673*
	TA39	**2190 (22.01%)**	2243 (24.96%)	2227 (24.07%)	2226 (24.01%)	2270 (26.46%)	1795*
	TA40	**2017 (20.85%)**	2065 (23.73%)	2089 (25.16%)	2084 (24.87%)	2098 (25.7%)	1669*
30×20	TA41	**2499 (24.64%)**	2611 (30.22%)	2572 (28.28%)	2595 (29.43%)	2604 (29.88%)	2005*
	TA42	**2295 (18.48%)**	2461 (27.05%)	2499 (29.01%)	2465 (27.26%)	2534 (30.82%)	1937*
	TA43	**2260 (22.43%)**	2408 (30.44%)	2400 (30.01%)	2464 (33.48%)	2445 (32.45%)	1846*
	TA44	**2334 (17.94%)**	2601 (31.43%)	2612 (31.99%)	2616 (32.19%)	2671 (34.97%)	1979*
	TA45	**2432 (21.6%)**	2558 (27.9%)	2524 (26.2%)	2526 (26.3%)	2552 (27.6%)	2000*
	TA46	**2501 (24.68%)**	2635 (31.36%)	2574 (28.32%)	2635 (31.36%)	2615 (30.36%)	2006*
	TA47	**2258 (19.53%)**	2377 (25.83%)	2382 (26.1%)	2434 (28.85%)	2468 (30.65%)	1889*
	TA48	**2389 (23.34%)**	2552 (31.75%)	2522 (30.2%)	2488 (28.45%)	2495 (28.81%)	1937*
	TA49	**2413 (23.05%)**	2466 (25.75%)	2460 (25.45%)	2428 (23.81%)	2496 (27.28%)	1961*
	TA50	**2375 (23.5%)**	2508 (30.42%)	2493 (29.64%)	2515 (30.79%)	2522 (31.15%)	1923*
50×15	TA51	**3409 (23.51%)**	3545 (28.44%)	3617 (31.05%)	3579 (29.67%)	3551 (28.66%)	2760*
	TA52	**3245 (17.74%)**	3475 (26.09%)	3430 (24.46%)	3417 (23.98%)	3428 (24.38%)	2756*
	TA53	**3078 (13.29%)**	3173 (16.78%)	3176 (16.89%)	3162 (16.38%)	3164 (16.45%)	2717*
	TA54	**3131 (10.29%)**	3362 (18.42%)	3334 (17.44%)	3334 (17.44%)	3310 (16.59%)	2839*
	TA55	**3217 (20.08%)**	3266 (21.91%)	3283 (22.55%)	3266 (21.91%)	3259 (21.65%)	2679*
	TA56	**3123 (12.3%)**	3315 (19.2%)	3320 (19.38%)	3284 (18.09%)	3314 (19.17%)	2781*
	TA57	**3332 (13.22%)**	3454 (17.36%)	3488 (18.52%)	3438 (16.82%)	3504 (19.06%)	2943*
	TA58	**3324 (15.22%)**	3438 (19.17%)	3447 (19.48%)	3419 (18.51%)	3433 (18.99%)	2885*
	TA59	**3144 (18.42%)**	3333 (25.54%)	3316 (24.9%)	3303 (24.41%)	3303 (24.41%)	2655*
	TA60	**3067 (12.63%)**	3206 (17.74%)	3215 (18.07%)	3162 (16.12%)	3201 (17.55%)	2723*
50×20	TA61	**3357 (17.05%)**	3492 (21.76%)	3525 (22.91%)	3628 (26.5%)	3622 (26.29%)	2868*
	TA62	**3381 (17.85%)**	3586 (24.99%)	3596 (25.34%)	3643 (26.98%)	3615 (26.0%)	2869*
	TA63	**3162 (14.77%)**	3342 (21.31%)	3269 (18.66%)	3318 (20.44%)	3345 (21.42%)	2755*
	TA64	**3117 (15.36%)**	3285 (21.58%)	3230 (19.54%)	3299 (22.09%)	3332 (23.32%)	2702*
	TA65	**3301 (21.14%)**	3434 (26.02%)	3398 (24.7%)	3399 (24.73%)	3384 (24.18%)	2725*
	TA66	**3272 (15.01%)**	3447 (21.16%)	3468 (21.9%)	3474 (22.11%)	3505 (23.2%)	2845*
	TA67	**3231 (14.37%)**	3597 (27.33%)	3601 (27.47%)	3582 (26.8%)	3640 (28.85%)	2825*
	TA68	**3101 (11.39%)**	3241 (16.42%)	3306 (18.75%)	3314 (19.04%)	3374 (21.19%)	2784*
	TA69	**3538 (15.21%)**	3639 (18.5%)	3621 (17.91%)	3670 (19.51%)	3645 (18.69%)	3071*
	TA70	**3514 (17.33%)**	3697 (23.44%)	3724 (24.34%)	3635 (21.37%)	3718 (24.14%)	2995*
100×20	TA71	**5990 (9.63%)**	6297 (15.25%)	6239 (14.18%)	6372 (16.62%)	6384 (16.84%)	5464*
	TA72	**5679 (9.61%)**	5791 (11.77%)	5817 (12.28%)	5847 (12.85%)	5863 (13.16%)	5181*
	TA73	**6144 (10.34%)**	6397 (14.89%)	6451 (15.86%)	6406 (15.05%)	6417 (15.25%)	5568*
	TA74	**5680 (6.39%)**	5988 (12.16%)	5938 (11.22%)	5977 (11.95%)	6018 (12.72%)	5339*
	TA75	**6125 (13.59%)**	6359 (17.93%)	6458 (19.77%)	6496 (20.47%)	6466 (19.92%)	5392*
	TA76	**6033 (12.94%)**	6140 (14.94%)	6178 (15.65%)	6150 (15.13%)	6112 (14.41%)	5342*
	TA77	**5865 (7.89%)**	5982 (10.04%)	6004 (10.45%)	5980 (10.01%)	5999 (10.36%)	5436*
	TA78	**5818 (7.86%)**	6250 (15.87%)	6146 (13.94%)	6311 (17.0%)	6294 (16.69%)	5394*
	TA79	**5848 (9.15%)**	6101 (13.87%)	6040 (12.73%)	6081 (13.49%)	5995 (11.89%)	5358*
	TA80	**5617 (8.37%)**	5904 (13.91%)	5830 (12.48%)	5915 (14.12%)	5907 (13.97%)	5183*

Table 10: CL strategies on Taillard's instances.

	Instance	Zhang et al.	ICL	Base	UCL	ASCL	ACL	UB
15×15	TA1	1443 (17.22%)	1475 (19.82%)	1475 (19.82%)	1393 (13.16%)	1387 (12.67%)	**1379 (12.02%)**	1231*
	TA2	1544 (24.12%)	1401 (12.62%)	1401 (12.62%)	1391 (11.82%)	1336 (7.4%)	**1325 (6.51%)**	1244*
	TA3	1440 (18.23%)	1393 (14.37%)	1393 (14.37%)	1408 (15.6%)	1341 (10.1%)	**1338 (9.85%)**	1218*
	TA4	1637 (39.32%)	1340 (14.04%)	1340 (14.04%)	1404 (19.49%)	1305 (11.06%)	**1275 (8.51%)**	1175*
	TA5	1619 (32.27%)	1390 (13.56%)	1390 (13.56%)	1407 (14.95%)	1329 (8.58%)	**1303 (6.45%)**	1224*
	TA6	1601 (29.32%)	1388 (12.12%)	1388 (12.12%)	1375 (11.07%)	**1309 (5.74%)**	1325 (7.03%)	1238*
	TA7	1568 (27.79%)	1425 (16.14%)	1425 (16.14%)	1398 (13.94%)	**1321 (7.66%)**	1332 (8.56%)	1227*
	TA8	1468 (20.62%)	1407 (15.61%)	1407 (15.61%)	1409 (15.78%)	**1316 (8.13%)**	1325 (8.87%)	1217*
	TA9	1627 (27.71%)	1472 (15.54%)	1472 (15.54%)	1478 (16.01%)	**1390 (9.11%)**	1412 (10.83%)	1274*
	TA10	1527 (23.05%)	1439 (15.95%)	1439 (15.95%)	1398 (12.65%)	**1332 (7.33%)**	1384 (11.52%)	1241*
20×15	TA11	1794 (32.2%)	1576 (16.14%)	1671 (23.14%)	1652 (21.74%)	1538 (13.34%)	**1506 (10.98%)**	1357*
	TA12	1805 (32.04%)	1596 (16.75%)	1678 (22.75%)	1589 (16.24%)	**1498 (9.58%)**	1514 (10.75%)	1367*
	TA13	1932 (43.86%)	1569 (16.83%)	1691 (25.91%)	1634 (21.67%)	1543 (14.89%)	**1510 (12.43%)**	1343*
	TA14	1664 (23.72%)	1535 (14.13%)	1685 (25.28%)	1564 (16.28%)	1488 (10.63%)	**1455 (8.18%)**	1345*
	TA15	1730 (29.2%)	1554 (16.06%)	1675 (25.09%)	1621 (21.06%)	**1492 (11.43%)**	1516 (13.22%)	1339*
	TA16	1710 (25.74%)	1582 (16.32%)	1689 (24.19%)	1657 (21.84%)	1558 (14.56%)	**1485 (9.19%)**	1360*
	TA17	1897 (29.75%)	1705 (16.62%)	1800 (23.12%)	1732 (18.47%)	1666 (13.95%)	**1614 (10.4%)**	1462*
	TA18	1794 (28.51%)	1662 (19.05%)	1785 (27.87%)	1645 (17.84%)	1634 (17.05%)	**1560 (11.75%)**	1396*
	TA19	1682 (26.28%)	1544 (15.92%)	1615 (21.25%)	1609 (20.8%)	1526 (14.56%)	**1451 (8.93%)**	1332*
	TA20	1739 (29.01%)	1558 (15.58%)	1632 (21.07%)	1574 (16.77%)	1519 (12.69%)	**1482 (9.94%)**	1348*
20×20	TA21	2252 (37.15%)	1967 (19.79%)	1944 (18.39%)	1937 (17.97%)	1861 (13.34%)	**1838 (11.94%)**	1642*
	TA22	2102 (31.37%)	1865 (16.56%)	1937 (21.06%)	1875 (17.19%)	1769 (10.56%)	**1752 (9.5%)**	1600*
	TA23	2085 (33.91%)	1870 (20.1%)	1903 (22.22%)	1853 (19.01%)	**1745 (12.07%)**	1748 (12.27%)	1557*
	TA24	2200 (33.82%)	1898 (15.45%)	1958 (19.1%)	1916 (16.55%)	1818 (10.58%)	**1807 (9.91%)**	1644*
	TA25	2201 (37.99%)	1866 (16.99%)	2006 (25.77%)	1900 (19.12%)	1807 (13.29%)	**1772 (11.1%)**	1595*
	TA26	2176 (32.44%)	1928 (17.35%)	1971 (19.96%)	1943 (18.26%)	1810 (10.16%)	**1802 (9.68%)**	1643*
	TA27	2132 (26.9%)	1980 (17.86%)	2066 (22.98%)	1939 (15.42%)	1900 (13.1%)	**1895 (12.8%)**	1680*
	TA28	2146 (33.87%)	1841 (14.85%)	1924 (20.02%)	1814 (13.16%)	1798 (12.16%)	**1788 (11.54%)**	1603*
	TA29	1952 (20.12%)	1856 (14.22%)	1926 (18.52%)	1864 (14.71%)	1796 (10.52%)	**1753 (7.88%)**	1625*
	TA30	2035 (28.47%)	1852 (16.92%)	1949 (23.04%)	1880 (18.69%)	1794 (13.26%)	**1776 (12.12%)**	1584*
30×15	TA31	2565 (45.41%)	2149 (21.83%)	2263 (28.29%)	2172 (23.13%)	2072 (17.46%)	**2028 (14.97%)**	1764*
	TA32	2388 (33.86%)	2216 (24.22%)	2301 (28.98%)	2260 (26.68%)	**2072 (16.14%)**	2092 (17.26%)	1784*
	TA33	2324 (29.76%)	2350 (31.21%)	2410 (34.56%)	2295 (28.14%)	2185 (22.0%)	**2114 (18.03%)**	1791*
	TA34	2332 (27.57%)	2166 (18.49%)	2317 (26.75%)	2180 (19.26%)	2086 (14.11%)	**2077 (13.62%)**	1828*
	TA35	2505 (24.81%)	2235 (11.36%)	2389 (19.03%)	2257 (12.46%)	2141 (6.68%)	**2103 (4.78%)**	2007*
	TA36	2497 (37.27%)	2248 (23.58%)	2326 (27.87%)	2235 (22.87%)	2072 (13.91%)	**2046 (12.48%)**	1819*
	TA37	2325 (31.28%)	2251 (27.1%)	2329 (31.51%)	2197 (24.05%)	2098 (18.46%)	**2080 (17.45%)**	1771*
	TA38	2302 (37.6%)	2036 (21.7%)	2108 (26.0%)	2071 (23.79%)	1926 (15.12%)	**1882 (12.49%)**	1673*
	TA39	2410 (34.26%)	2194 (22.23%)	2226 (24.01%)	2119 (18.05%)	2056 (14.54%)	**2029 (13.04%)**	1795*
	TA40	2140 (28.22%)	2032 (21.75%)	2084 (24.87%)	2007 (20.25%)	1973 (18.21%)	**1930 (15.64%)**	1669*
30×20	TA41	2667 (33.02%)	2543 (26.83%)	2604 (29.88%)	2482 (23.79%)	2397 (19.55%)	**2356 (17.51%)**	2005*
	TA42	2664 (37.53%)	2399 (23.85%)	2534 (30.82%)	2416 (24.73%)	**2235 (15.38%)**	2260 (16.68%)	1937*
	TA43	2431 (31.69%)	2332 (26.33%)	2445 (32.45%)	2354 (27.52%)	2194 (18.85%)	**2163 (17.17%)**	1846*
	TA44	2714 (37.14%)	2506 (26.63%)	2671 (34.97%)	2518 (27.24%)	2323 (17.38%)	**2293 (15.87%)**	1979*
	TA45	2637 (31.85%)	2470 (23.5%)	2552 (27.6%)	2443 (22.15%)	2289 (14.45%)	**2250 (12.5%)**	2000*
	TA46	2776 (38.38%)	2569 (28.07%)	2615 (30.36%)	2515 (25.37%)	2341 (16.7%)	**2314 (15.35%)**	2006*
	TA47	2476 (31.07%)	2398 (26.95%)	2468 (30.65%)	2323 (22.98%)	2200 (16.46%)	**2141 (13.34%)**	1889*
	TA48	2490 (28.55%)	2471 (27.57%)	2495 (28.81%)	2396 (23.7%)	**2285 (17.97%)**	2289 (18.17%)	1937*
	TA49	2556 (30.34%)	2428 (23.81%)	2496 (27.28%)	2329 (18.77%)	**2266 (15.55%)**	2270 (15.76%)	1961*
	TA50	2628 (36.66%)	2434 (26.57%)	2522 (31.15%)	2419 (25.79%)	2280 (18.56%)	**2279 (18.51%)**	1923*
50×15	TA51	3599 (30.4%)	3394 (22.97%)	3551 (28.66%)	3357 (21.63%)	3303 (19.67%)	**3181 (15.25%)**	2760*
	TA52	3341 (21.23%)	3337 (21.08%)	3428 (24.38%)	3254 (18.07%)	3106 (12.7%)	**3029 (9.91%)**	2756*
	TA53	3186 (17.26%)	3122 (14.91%)	3164 (16.45%)	3035 (11.7%)	2942 (8.28%)	**2921 (7.51%)**	2717*
	TA54	3266 (15.04%)	3163 (11.41%)	3310 (16.59%)	3134 (10.39%)	3012 (6.09%)	**2914 (2.64%)**	2839*
	TA55	3232 (20.64%)	3247 (21.2%)	3259 (21.65%)	3109 (16.05%)	3065 (14.41%)	**3050 (13.85%)**	2679*
	TA56	3378 (21.47%)	3290 (18.3%)	3314 (19.17%)	3150 (13.27%)	3081 (10.79%)	**3012 (8.31%)**	2781*
	TA57	3471 (17.94%)	3437 (16.79%)	3504 (19.06%)	3326 (13.01%)	3249 (10.4%)	**3180 (8.05%)**	2943*
	TA58	3732 (29.36%)	3334 (15.56%)	3433 (18.99%)	3321 (15.11%)	3166 (9.74%)	**3103 (7.56%)**	2885*
	TA59	3381 (27.34%)	3196 (20.38%)	3303 (24.41%)	3152 (18.72%)	3025 (13.94%)	**3009 (13.33%)**	2655*
	TA60	3352 (23.1%)	3098 (13.77%)	3201 (17.55%)	3037 (11.53%)	2932 (7.68%)	**2909 (6.83%)**	2723*
50×20	TA61	3654 (27.41%)	3495 (21.86%)	3622 (26.29%)	3443 (20.05%)	3221 (12.31%)	**3184 (11.02%)**	2868*
	TA62	3722 (29.73%)	3496 (21.85%)	3615 (26.0%)	3428 (19.48%)	3274 (14.12%)	**3229 (12.55%)**	2869*
	TA63	3536 (28.35%)	3232 (17.31%)	3345 (21.42%)	3163 (14.81%)	3040 (10.34%)	**3004 (9.04%)**	2755*
	TA64	3631 (34.38%)	3164 (17.1%)	3332 (23.32%)	3180 (17.69%)	2933 (8.55%)	**2916 (7.92%)**	2702*
	TA65	3359 (23.27%)	3314 (21.61%)	3384 (24.18%)	3250 (19.27%)	3140 (15.23%)	**3068 (12.59%)**	2725*
	TA66	3555 (24.96%)	3415 (20.04%)	3505 (23.2%)	3334 (17.19%)	**3144 (10.51%)**	3144 (10.51%)	2845*
	TA67	3567 (26.27%)	3442 (21.84%)	3640 (28.85%)	3449 (22.09%)	3148 (11.43%)	**3120 (10.44%)**	2825*
	TA68	3680 (32.18%)	3174 (14.01%)	3374 (21.19%)	3193 (14.69%)	2999 (7.72%)	**2955 (6.14%)**	2784*
	TA69	3592 (16.97%)	3559 (15.89%)	3645 (18.69%)	3501 (14.0%)	3338 (8.69%)	**3295 (7.29%)**	3071*
	TA70	3643 (21.64%)	3553 (18.63%)	3718 (24.14%)	3540 (18.2%)	3357 (12.09%)	**3336 (11.39%)**	2995*
100×20	TA71	6452 (18.08%)	6238 (14.17%)	6384 (16.84%)	5985 (9.54%)	5798 (6.11%)	**5653 (3.46%)**	5464*
	TA72	5695 (9.92%)	5771 (11.39%)	5863 (13.16%)	5556 (7.24%)	5439 (4.98%)	**5365 (3.55%)**	5181*
	TA73	6462 (16.06%)	6294 (13.04%)	6417 (15.25%)	6054 (8.73%)	5997 (7.7%)	**5885 (5.69%)**	5568*
	TA74	5885 (10.23%)	5859 (9.74%)	6018 (12.72%)	5727 (7.27%)	5518 (3.35%)	**5427 (1.65%)**	5339*
	TA75	6355 (17.86%)	6294 (16.73%)	6466 (19.92%)	6154 (14.13%)	5918 (9.76%)	**5790 (7.38%)**	5392*
	TA76	6135 (14.84%)	6060 (13.44%)	6112 (14.41%)	5886 (10.18%)	**5714 (6.96%)**	5748 (7.6%)	5342*
	TA77	6056 (11.41%)	5935 (9.18%)	5999 (10.36%)	5716 (5.15%)	5655 (4.03%)	**5588 (2.8%)**	5436*
	TA78	6101 (13.11%)	6153 (14.07%)	6294 (16.69%)	5941 (10.14%)	5735 (6.32%)	**5570 (3.26%)**	5394*
	TA79	5943 (10.92%)	5945 (10.96%)	5995 (11.89%)	5773 (7.75%)	5567 (3.9%)	**5424 (1.23%)**	5358*
	TA80	5892 (13.68%)	5819 (12.27%)	5907 (13.97%)	5639 (8.8%)	5480 (5.73%)	**5339 (3.01%)**	5183*

Table 11: Results on Taillard's benchmark.

	Instance	SPT	FDD/WKR	MWKR	MOPNR	Zhang et al.	ACL	UB
15×15	TA1	1462 (18.77%)	1841 (49.55%)	1491 (21.12%)	1438 (16.82%)	1443 (17.22%)	**1379 (12.02%)**	1231*
	TA2	1446 (16.24%)	1895 (52.33%)	1440 (15.76%)	1452 (16.72%)	1544 (24.12%)	**1325 (6.51%)**	1244*
	TA3	1495 (22.74%)	1914 (57.14%)	1426 (17.08%)	1418 (16.42%)	1440 (18.23%)	**1338 (9.85%)**	1218*
	TA4	1708 (45.36%)	1653 (40.68%)	1387 (18.04%)	1457 (24.0%)	1637 (39.32%)	**1275 (8.51%)**	1175*
	TA5	1618 (32.19%)	1787 (46.0%)	1494 (22.06%)	1448 (18.3%)	1619 (32.27%)	**1303 (6.45%)**	1224*
	TA6	1522 (22.94%)	1748 (41.2%)	1369 (10.58%)	1486 (20.03%)	1601 (29.32%)	**1325 (7.03%)**	1238*
	TA7	1434 (16.87%)	1660 (35.29%)	1470 (19.8%)	1456 (18.66%)	1568 (27.79%)	**1332 (8.56%)**	1227*
	TA8	1457 (19.72%)	1803 (48.15%)	1491 (22.51%)	1482 (21.77%)	1468 (20.62%)	**1325 (8.87%)**	1217*
	TA9	1622 (27.32%)	1848 (45.05%)	1541 (20.96%)	1594 (25.12%)	1627 (27.71%)	**1412 (10.83%)**	1274*
	TA10	1697 (36.74%)	1937 (56.08%)	1534 (23.61%)	1582 (27.48%)	1527 (23.05%)	**1384 (11.52%)**	1241*
20×15	TA11	1865 (37.44%)	2101 (54.83%)	1685 (24.17%)	1665 (22.7%)	1794 (32.2%)	**1506 (10.98%)**	1357*
	TA12	1667 (21.95%)	2034 (48.79%)	1707 (24.87%)	1739 (27.21%)	1805 (32.04%)	**1514 (10.75%)**	1367*
	TA13	1802 (34.18%)	2141 (59.42%)	1690 (25.84%)	1642 (22.26%)	1932 (43.86%)	**1510 (12.43%)**	1343*
	TA14	1635 (21.56%)	1841 (36.88%)	1563 (16.21%)	1662 (23.57%)	1664 (23.72%)	**1455 (8.18%)**	1345*
	TA15	1835 (37.04%)	2187 (63.33%)	1696 (26.66%)	1682 (25.62%)	1730 (29.2%)	**1516 (13.22%)**	1339*
	TA16	1965 (44.49%)	1926 (41.62%)	1584 (16.47%)	1638 (20.44%)	1710 (25.74%)	**1485 (9.19%)**	1360*
	TA17	2059 (40.83%)	2093 (43.16%)	1804 (23.39%)	1856 (26.95%)	1897 (29.75%)	**1614 (10.4%)**	1462*
	TA18	1808 (29.51%)	2064 (47.85%)	1751 (25.43%)	1710 (22.49%)	1794 (28.51%)	**1560 (11.75%)**	1396*
	TA19	1789 (34.31%)	1958 (47.0%)	1667 (25.15%)	1651 (23.95%)	1682 (26.28%)	**1451 (8.93%)**	1332*
	TA20	1710 (26.85%)	2195 (62.83%)	1689 (25.3%)	1622 (20.33%)	1739 (29.01%)	**1482 (9.94%)**	1348*
20×20	TA21	2175 (32.46%)	2455 (49.51%)	2044 (24.48%)	1964 (19.61%)	2252 (37.15%)	**1838 (11.94%)**	1642*
	TA22	1965 (22.81%)	2177 (36.06%)	1914 (19.62%)	1905 (19.06%)	2102 (31.37%)	**1752 (9.5%)**	1600*
	TA23	1933 (24.15%)	2514 (61.46%)	1983 (27.36%)	1922 (23.44%)	2085 (33.91%)	**1748 (12.27%)**	1557*
	TA24	2230 (35.64%)	2391 (45.44%)	1982 (20.56%)	1943 (18.19%)	2200 (33.82%)	**1807 (9.91%)**	1644*
	TA25	1950 (22.26%)	2267 (42.13%)	1941 (21.69%)	1957 (22.7%)	2201 (37.99%)	**1772 (11.1%)**	1595*
	TA26	2188 (33.17%)	2644 (60.93%)	1951 (18.75%)	1964 (19.54%)	2176 (32.44%)	**1802 (9.68%)**	1643*
	TA27	2096 (24.76%)	2514 (49.64%)	2091 (24.46%)	2160 (28.57%)	2132 (26.9%)	**1895 (12.8%)**	1680*
	TA28	1968 (22.77%)	2330 (45.35%)	1997 (24.58%)	1952 (21.77%)	2146 (33.87%)	**1788 (11.54%)**	1603*
	TA29	2166 (33.29%)	2232 (37.35%)	1860 (14.46%)	1899 (16.86%)	1952 (20.12%)	**1753 (7.88%)**	1625*
	TA30	1999 (26.2%)	2348 (48.23%)	1935 (22.16%)	2017 (27.34%)	2035 (28.47%)	**1776 (12.12%)**	1584*
30×15	TA31	2335 (32.37%)	2459 (39.4%)	2134 (20.98%)	2143 (21.49%)	2565 (45.41%)	**2028 (14.97%)**	1764*
	TA32	2432 (36.32%)	2672 (49.78%)	2223 (24.61%)	2188 (22.65%)	2388 (33.86%)	**2092 (17.26%)**	1784*
	TA33	2453 (36.96%)	2766 (54.44%)	2349 (31.16%)	2308 (28.87%)	2324 (29.76%)	**2114 (18.03%)**	1791*
	TA34	2434 (33.15%)	2669 (46.01%)	2245 (22.81%)	2193 (19.97%)	2332 (27.57%)	**2077 (13.62%)**	1828*
	TA35	2497 (24.41%)	2525 (25.81%)	2226 (10.91%)	2255 (12.36%)	2505 (24.81%)	**2103 (4.78%)**	2007*
	TA36	2445 (34.41%)	2690 (47.88%)	2365 (30.02%)	2307 (26.83%)	2497 (37.27%)	**2046 (12.48%)**	1819*
	TA37	2664 (50.42%)	2492 (40.71%)	2130 (20.27%)	2190 (23.66%)	2325 (31.28%)	**2080 (17.45%)**	1771*
	TA38	2155 (28.81%)	2425 (44.95%)	2050 (22.53%)	2179 (30.25%)	2302 (37.6%)	**1882 (12.49%)**	1673*
	TA39	2477 (37.99%)	2596 (44.62%)	2221 (23.73%)	2167 (20.72%)	2410 (34.26%)	**2029 (13.04%)**	1795*
	TA40	2301 (37.87%)	2614 (56.62%)	2205 (32.12%)	2028 (21.51%)	2140 (28.22%)	**1930 (15.64%)**	1669*
30×20	TA41	2499 (24.64%)	2991 (49.18%)	2620 (30.67%)	2538 (26.58%)	2667 (33.02%)	**2356 (17.51%)**	2005*
	TA42	2710 (39.91%)	3027 (56.27%)	2416 (24.73%)	2440 (25.97%)	2664 (37.53%)	**2260 (16.68%)**	1937*
	TA43	2434 (31.85%)	2926 (58.5%)	2345 (27.03%)	2432 (31.74%)	2431 (31.69%)	**2163 (17.17%)**	1846*
	TA44	2906 (46.84%)	3462 (74.94%)	2544 (28.55%)	2426 (22.59%)	2714 (37.14%)	**2293 (15.87%)**	1979*
	TA45	2640 (32.0%)	3245 (62.25%)	2524 (26.2%)	2487 (24.35%)	2637 (31.85%)	**2250 (12.5%)**	2000*
	TA46	2667 (32.95%)	3008 (49.95%)	2447 (21.98%)	2490 (24.13%)	2776 (38.38%)	**2314 (15.35%)**	2006*
	TA47	2620 (38.7%)	2940 (55.64%)	2263 (19.8%)	2286 (21.02%)	2476 (31.07%)	**2141 (13.34%)**	1889*
	TA48	2620 (35.26%)	2991 (54.41%)	2356 (21.63%)	2371 (22.41%)	2490 (28.55%)	**2289 (18.17%)**	1937*
	TA49	2666 (35.95%)	2865 (46.1%)	2382 (21.47%)	2397 (22.23%)	2556 (30.34%)	**2270 (15.76%)**	1961*
	TA50	2429 (26.31%)	2995 (55.75%)	2493 (29.64%)	2469 (28.39%)	2628 (36.66%)	**2279 (18.51%)**	1923*
50×15	TA51	3856 (39.71%)	3851 (39.53%)	3435 (24.46%)	3567 (29.24%)	3599 (30.4%)	**3181 (15.25%)**	2760*
	TA52	3266 (18.51%)	3734 (35.49%)	3394 (23.15%)	3303 (19.85%)	3341 (21.23%)	**3029 (9.91%)**	2756*
	TA53	3507 (29.08%)	3394 (24.92%)	3098 (14.02%)	3115 (14.65%)	3186 (17.26%)	**2921 (7.51%)**	2717*
	TA54	3142 (10.67%)	3603 (26.91%)	3272 (15.25%)	3265 (15.01%)	3266 (15.04%)	**2914 (2.64%)**	2839*
	TA55	3225 (20.38%)	3664 (36.77%)	3188 (19.0%)	3279 (22.4%)	3232 (20.64%)	**3050 (13.85%)**	2679*
	TA56	3530 (26.93%)	4016 (44.41%)	3134 (12.69%)	3100 (11.47%)	3378 (21.47%)	**3012 (8.31%)**	2781*
	TA57	3725 (26.57%)	3720 (26.4%)	3261 (10.81%)	3335 (13.32%)	3471 (17.94%)	**3180 (8.05%)**	2943*
	TA58	3365 (16.64%)	3926 (36.08%)	3365 (16.64%)	3420 (18.54%)	3732 (29.36%)	**3103 (7.56%)**	2885*
	TA59	3294 (24.07%)	3672 (38.31%)	3131 (17.93%)	3117 (17.4%)	3381 (27.34%)	**3009 (13.33%)**	2655*
	TA60	3500 (28.53%)	3783 (38.93%)	3122 (14.65%)	3044 (11.79%)	3352 (23.1%)	**2909 (6.83%)**	2723*
50×20	TA61	3606 (25.73%)	4142 (44.42%)	3343 (16.56%)	3376 (17.71%)	3654 (27.41%)	**3184 (11.02%)**	2868*
	TA62	3639 (26.84%)	3897 (35.83%)	3462 (20.67%)	3417 (19.1%)	3722 (29.73%)	**3229 (12.55%)**	2869*
	TA63	3521 (27.8%)	3852 (39.82%)	3233 (17.35%)	3276 (18.91%)	3536 (28.35%)	**3004 (9.04%)**	2755*
	TA64	3447 (27.57%)	4001 (48.08%)	3188 (17.99%)	3057 (13.14%)	3631 (34.38%)	**2916 (7.92%)**	2702*
	TA65	3332 (22.28%)	4062 (49.06%)	3429 (25.83%)	3249 (19.23%)	3359 (23.27%)	**3068 (12.59%)**	2725*
	TA66	3677 (29.24%)	3940 (38.49%)	3287 (15.54%)	3335 (17.22%)	3555 (24.96%)	**3144 (10.51%)**	2845*
	TA67	3487 (23.43%)	3974 (40.67%)	3351 (18.62%)	3392 (20.07%)	3567 (26.27%)	**3120 (10.44%)**	2825*
	TA68	3336 (19.83%)	3857 (38.54%)	3203 (15.05%)	3251 (16.77%)	3680 (32.18%)	**2955 (6.14%)**	2784*
	TA69	3862 (25.76%)	4349 (41.62%)	3550 (15.6%)	3526 (14.82%)	3592 (16.97%)	**3295 (7.29%)**	3071*
	TA70	3801 (26.91%)	4147 (38.46%)	3482 (16.26%)	3590 (19.87%)	3643 (21.64%)	**3336 (11.39%)**	2995*
100×20	TA71	6232 (14.06%)	6818 (24.78%)	6036 (10.47%)	5938 (8.67%)	6452 (18.08%)	**5653 (3.46%)**	5464*
	TA72	5973 (15.29%)	6358 (22.72%)	5583 (7.76%)	5639 (8.84%)	5695 (9.92%)	**5365 (3.55%)**	5181*
	TA73	6482 (16.42%)	6967 (25.13%)	6050 (8.66%)	6128 (10.06%)	6462 (16.06%)	**5885 (5.69%)**	5568*
	TA74	6062 (13.54%)	6381 (19.52%)	5678 (6.35%)	5642 (5.68%)	5885 (10.23%)	**5427 (1.65%)**	5339*
	TA75	6217 (15.3%)	6757 (25.32%)	6029 (11.81%)	6212 (15.21%)	6355 (17.86%)	**5790 (7.38%)**	5392*
	TA76	6370 (19.24%)	6641 (24.32%)	5887 (10.2%)	5936 (11.12%)	6135 (14.84%)	**5748 (7.6%)**	5342*
	TA77	6045 (11.2%)	6540 (20.31%)	5905 (8.63%)	5829 (7.23%)	6056 (11.41%)	**5588 (2.8%)**	5436*
	TA78	6143 (13.89%)	6750 (25.14%)	5700 (5.67%)	5886 (9.12%)	6101 (13.11%)	**5570 (3.26%)**	5394*
	TA79	6018 (12.32%)	6461 (20.59%)	5749 (7.3%)	5652 (5.49%)	5943 (10.92%)	**5424 (1.23%)**	5358*
	TA80	5848 (12.83%)	6534 (26.07%)	5505 (6.21%)	5707 (10.11%)	5892 (13.68%)	**5339 (3.01%)**	5183*

Table 12: Results on DMU benchmark.

	Instance	SPT	FDD/WKR	MWKR	MOPNR	Zhang et al.	ACL	UB
20×15	DMU1	4516 (76.2%)	3535 (37.92%)	3988 (55.6%)	3882 (51.46%)	3323 (29.65%)	**3107 (21.23%)**	2563*
	DMU2	4593 (69.73%)	3847 (42.17%)	4555 (68.33%)	3884 (43.53%)	3630 (34.15%)	**3272 (20.92%)**	2706*
	DMU3	4438 (62.5%)	4063 (48.77%)	4117 (50.75%)	3979 (45.7%)	3660 (34.02%)	**3270 (19.74%)**	2731*
	DMU4	4533 (69.84%)	4160 (55.86%)	3995 (49.68%)	4079 (52.83%)	3816 (42.97%)	**3138 (17.57%)**	2669*
	DMU5	4420 (60.79%)	4238 (54.17%)	4977 (81.05%)	4116 (49.73%)	3897 (41.76%)	**3290 (19.68%)**	2749*
	DMU41	5283 (62.65%)	5187 (59.7%)	5377 (65.55%)	5070 (56.1%)	4316 (32.88%)	**3807 (17.21%)**	3248*
	DMU42	5354 (57.94%)	5583 (64.69%)	6076 (79.23%)	4976 (46.78%)	4858 (43.3%)	**4027 (18.79%)**	3390*
	DMU43	5328 (54.84%)	5086 (47.81%)	4938 (43.5%)	5012 (45.66%)	4887 (42.02%)	**4159 (20.87%)**	3441*
	DMU44	5745 (64.71%)	5550 (59.12%)	5630 (61.41%)	5213 (49.46%)	5151 (47.68%)	**4109 (17.8%)**	3488*
	DMU45	5305 (62.13%)	5414 (65.46%)	5446 (66.44%)	4921 (50.4%)	4615 (41.05%)	**3921 (19.83%)**	3272*
20×20	DMU6	6230 (92.05%)	5258 (62.08%)	5556 (71.27%)	4747 (46.33%)	4358 (34.34%)	**3830 (18.06%)**	3244*
	DMU7	5619 (84.47%)	4789 (57.22%)	4636 (52.2%)	4367 (43.37%)	3671 (20.52%)	**3488 (14.51%)**	3046*
	DMU8	5239 (64.34%)	4817 (51.1%)	5078 (59.28%)	4480 (40.53%)	4048 (26.98%)	**3716 (16.56%)**	3188*
	DMU9	4874 (57.63%)	4675 (51.2%)	4519 (46.15%)	4519 (46.15%)	4482 (44.95%)	**3542 (14.55%)**	3092*
	DMU10	4808 (61.13%)	4149 (39.04%)	4963 (66.32%)	4133 (38.51%)	4021 (34.75%)	**3420 (14.61%)**	2984*
	DMU46	6403 (58.69%)	5778 (43.2%)	6168 (52.86%)	6136 (52.07%)	5876 (45.63%)	**4611 (14.28%)**	4035*
	DMU47	6015 (52.7%)	6058 (53.8%)	6130 (55.62%)	5908 (49.99%)	5771 (46.51%)	**4521 (14.78%)**	3939*
	DMU48	5345 (42.04%)	5887 (56.44%)	5701 (51.5%)	5384 (43.08%)	5034 (33.78%)	**4366 (16.02%)**	3763*
	DMU49	6072 (63.67%)	5807 (56.52%)	6089 (64.12%)	5469 (47.41%)	5470 (47.44%)	**4368 (17.74%)**	3710*
	DMU50	6300 (68.95%)	5764 (54.57%)	6050 (62.24%)	5380 (44.27%)	5314 (42.5%)	**4427 (18.72%)**	3729*
30×15	DMU11	5864 (70.96%)	4798 (39.88%)	4961 (44.64%)	4891 (42.59%)	4435 (29.3%)	**4031 (17.52%)**	3430*
	DMU12	5966 (70.7%)	5595 (60.09%)	5994 (71.5%)	4947 (41.55%)	4864 (39.17%)	**4033 (15.39%)**	3495*
	DMU13	5744 (56.04%)	5324 (44.63%)	6190 (68.16%)	4979 (35.26%)	4918 (33.6%)	**4273 (16.08%)**	3681*
	DMU14	5469 (61.14%)	4830 (42.31%)	5567 (64.02%)	4839 (42.58%)	4130 (21.69%)	**3862 (13.79%)**	3394*
	DMU15	5518 (65.06%)	4928 (47.41%)	5299 (58.51%)	4653 (39.19%)	4392 (31.38%)	**3913 (17.05%)**	3343*
	DMU51	6538 (56.9%)	7002 (68.03%)	6841 (64.17%)	6691 (60.57%)	6241 (49.77%)	**4745 (13.87%)**	4167*
	DMU52	7341 (70.29%)	6650 (54.26%)	6942 (61.03%)	6591 (52.89%)	6714 (55.74%)	**5083 (17.91%)**	4311*
	DMU53	7232 (64.59%)	7170 (63.18%)	7430 (69.09%)	6851 (55.92%)	6724 (53.03%)	**5147 (17.14%)**	4394*
	DMU54	7178 (64.56%)	6767 (55.14%)	6461 (48.12%)	6540 (49.93%)	6522 (49.52%)	**5133 (17.68%)**	4362*
	DMU55	6212 (45.45%)	7101 (66.26%)	6844 (60.24%)	6446 (50.92%)	6639 (55.44%)	**5000 (17.07%)**	4271*
30×20	DMU16	6241 (66.38%)	5948 (58.57%)	5837 (55.61%)	5743 (53.11%)	4953 (32.04%)	**4537 (20.95%)**	3751*
	DMU17	6487 (70.08%)	6035 (58.23%)	6610 (73.31%)	5540 (45.25%)	5379 (41.03%)	**4657 (22.1%)**	3814*
	DMU18	6978 (81.53%)	5863 (52.52%)	6363 (65.53%)	5714 (48.65%)	5100 (32.67%)	**4685 (21.88%)**	3844*
	DMU19	5767 (53.05%)	5424 (43.95%)	6385 (69.45%)	5223 (38.61%)	4889 (29.75%)	**4444 (17.94%)**	3768*
	DMU20	6910 (86.25%)	6444 (73.69%)	6472 (74.45%)	5530 (49.06%)	4859 (30.97%)	**4443 (19.76%)**	3710*
	DMU56	7698 (55.8%)	8248 (66.93%)	7930 (60.49%)	7620 (54.22%)	7328 (48.31%)	**5926 (19.94%)**	4941*
	DMU57	7746 (66.4%)	7694 (65.28%)	7063 (51.73%)	7345 (57.79%)	6704 (44.02%)	**5493 (18.0%)**	4655*
	DMU58	7269 (54.4%)	7601 (61.45%)	7708 (63.72%)	7216 (53.27%)	6721 (42.76%)	**5566 (18.22%)**	4708*
	DMU59	7114 (53.85%)	7490 (61.98%)	7335 (58.63%)	7589 (64.12%)	7109 (53.74%)	**5659 (22.38%)**	4624*
	DMU60	8150 (71.4%)	7526 (58.28%)	7547 (58.72%)	7399 (55.6%)	6632 (39.47%)	**5650 (18.82%)**	4755*
40×15	DMU21	7400 (68.95%)	6416 (46.48%)	6314 (44.16%)	6048 (38.08%)	5317 (21.39%)	**5254 (19.95%)**	4380*
	DMU22	7353 (55.62%)	6645 (40.63%)	6980 (47.72%)	6351 (34.41%)	5534 (17.12%)	**5495 (16.3%)**	4725*
	DMU23	7262 (55.57%)	6781 (45.27%)	6472 (38.65%)	6004 (28.62%)	5620 (20.39%)	**5421 (16.13%)**	4668*
	DMU24	6799 (46.28%)	6582 (41.61%)	7079 (52.3%)	6155 (32.42%)	5753 (23.77%)	**5441 (17.06%)**	4648*
	DMU25	6428 (54.37%)	5756 (38.23%)	6042 (45.1%)	5365 (28.84%)	**4775 (14.67%)**	4872 (17.0%)	4164*
	DMU61	7817 (51.14%)	8757 (69.32%)	8734 (68.87%)	8076 (56.15%)	8203 (58.6%)	**6081 (17.58%)**	5172*
	DMU62	7759 (47.37%)	8082 (53.5%)	8262 (56.92%)	8253 (56.75%)	8091 (53.68%)	**6167 (17.13%)**	5265*
	DMU63	8296 (55.76%)	8384 (57.42%)	8364 (57.04%)	8417 (58.04%)	8031 (50.79%)	**6301 (18.31%)**	5326*
	DMU64	8444 (60.84%)	8490 (61.71%)	8406 (60.11%)	8161 (55.45%)	7738 (47.39%)	**6087 (15.94%)**	5250*
	DMU65	8454 (62.89%)	8307 (60.06%)	8189 (57.78%)	8225 (58.48%)	7577 (45.99%)	**6200 (19.46%)**	5190*
40×20	DMU26	7766 (67.12%)	7240 (55.8%)	7107 (52.94%)	6236 (34.19%)	5946 (27.95%)	**5771 (24.19%)**	4647*
	DMU27	7501 (54.72%)	6965 (43.67%)	7313 (50.85%)	6936 (43.07%)	6418 (32.38%)	**6177 (27.41%)**	4848*
	DMU28	8621 (83.74%)	6516 (38.87%)	8194 (74.64%)	6714 (43.09%)	5986 (27.58%)	**5976 (27.37%)**	4692*
	DMU29	8052 (71.65%)	6971 (48.6%)	7448 (58.77%)	6990 (49.01%)	6051 (28.99%)	**5767 (22.94%)**	4691*
	DMU30	7372 (55.79%)	6910 (46.03%)	7890 (66.74%)	6869 (45.16%)	**5988 (26.54%)**	6015 (27.11%)	4732*
	DMU66	8971 (56.92%)	9606 (68.03%)	8966 (56.83%)	8726 (52.63%)	8475 (48.24%)	**7217 (26.24%)**	5717*
	DMU67	9096 (56.48%)	9103 (56.6%)	9306 (60.09%)	9372 (61.22%)	8832 (51.94%)	**7152 (23.03%)**	5813*
	DMU68	9265 (60.49%)	9431 (63.36%)	9445 (63.61%)	8722 (51.08%)	8693 (50.58%)	**7181 (24.39%)**	5773*
	DMU69	9215 (61.41%)	9951 (74.3%)	9450 (65.53%)	8697 (52.34%)	8634 (51.23%)	**7175 (25.68%)**	5709*
	DMU70	9522 (61.69%)	9416 (59.89%)	9490 (61.15%)	9445 (60.38%)	8735 (48.33%)	**7410 (25.83%)**	5889*
50×15	DMU31	8869 (57.25%)	7899 (40.05%)	8147 (44.45%)	7192 (27.52%)	7156 (26.88%)	**6895 (22.25%)**	5640*
	DMU32	7814 (31.84%)	7316 (23.44%)	8004 (35.04%)	7267 (22.61%)	**6506 (9.77%)**	7231 (22.0%)	5927*
	DMU33	8114 (41.66%)	7262 (26.78%)	7710 (34.6%)	7069 (23.41%)	**6192 (8.1%)**	7076 (23.53%)	5728*
	DMU34	7625 (41.6%)	7725 (43.45%)	7709 (43.16%)	6919 (28.49%)	**6257 (16.19%)**	6598 (22.53%)	5385*
	DMU35	8626 (53.08%)	7099 (25.98%)	7617 (35.17%)	7033 (24.81%)	**6302 (11.84%)**	6906 (22.56%)	5635*
	DMU71	9594 (53.92%)	10889 (74.7%)	9978 (60.08%)	9514 (52.64%)	9797 (57.18%)	**7441 (19.38%)**	6233*
	DMU72	9882 (52.43%)	11602 (78.96%)	10135 (56.33%)	10063 (55.22%)	9926 (53.11%)	**7864 (21.3%)**	6483*
	DMU73	9953 (61.5%)	10212 (65.7%)	9721 (57.73%)	9615 (56.01%)	9933 (61.17%)	**7591 (23.17%)**	6163*
	DMU74	9866 (58.62%)	10659 (71.37%)	10086 (62.15%)	9536 (53.31%)	9833 (58.09%)	**7398 (18.94%)**	6220*
	DMU75	9411 (51.86%)	10839 (74.91%)	9953 (60.61%)	10157 (63.9%)	9892 (59.63%)	**7421 (19.75%)**	6197*
50×20	DMU36	9911 (76.32%)	8084 (43.82%)	8090 (43.92%)	7703 (37.04%)	7470 (32.89%)	**6339 (12.77%)**	5621*
	DMU37	8917 (52.4%)	9433 (61.22%)	9685 (65.53%)	7844 (34.06%)	7296 (24.7%)	**6815 (16.48%)**	5851*
	DMU38	9384 (64.26%)	8428 (47.52%)	8414 (47.28%)	8398 (47.0%)	7410 (29.7%)	**6424 (12.45%)**	5713*
	DMU39	9221 (60.45%)	8177 (42.28%)	9266 (61.23%)	7969 (38.66%)	6827 (18.79%)	**6592 (14.7%)**	5747*
	DMU40	9406 (68.66%)	7773 (39.38%)	8261 (48.13%)	8173 (46.55%)	7325 (31.34%)	**6398 (14.72%)**	5577*
	DMU76	11677 (71.39%)	11576 (69.91%)	10571 (55.16%)	11019 (61.73%)	9698 (42.35%)	**7905 (16.03%)**	6813*
	DMU77	10401 (52.46%)	11910 (74.58%)	11148 (63.41%)	10577 (55.04%)	10693 (56.74%)	**7773 (13.94%)**	6822*
	DMU78	10585 (56.35%)	11464 (69.34%)	10540 (55.69%)	10989 (62.32%)	9986 (47.5%)	**7912 (16.87%)**	6770*
	DMU79	11115 (59.47%)	11035 (58.32%)	11201 (60.7%)	10729 (53.93%)	10936 (56.9%)	**7925 (13.7%)**	6970*
	DMU80	10711 (60.2%)	11116 (66.26%)	10894 (62.94%)	10679 (59.72%)	9875 (47.7%)	**7686 (14.96%)**	6686*