Instruction

This tutorial is designed to introduce users to the structure of our test instances and result files, as well as provide a guidance on how to run our precompiled procedures.

1. Precursory Statement

Our procedures depend on the Gurobi solver (Version 10.0) to solve MILP models. Therefore, before running the procedure, please ensure that Gurobi 10.0 is installed and properly configured on your platform. Please note that other versions of Gurobi are not be supported and may cause compatibility issues. We highly recommend running our procedures on a Linux platform. For Windows users, please ensure that your system is running Windows 10 and that Visual Studio 2019 (or other versions) is installed. This is because our Windows executables are compiled using the Microsoft compiler (cl.exe). Failure to meet these requirements may lead to errors during execution.

2. Download and Unzip

To run our precompiled procedures, please download the following folders and archives:

- ✧ **bin**: This folder contains the executable procedures for the developed MILP models and heuristics.
- ✧ **instances.zip**: This archive includes all the test datasets used in our computational experiments. Please extract these files to access the data necessary for testing and validation.
- ✧ **Preresults.zip**: This archive contains the results from our preliminary experiments, which can be used for comparison with your own findings.

Before running the procedure, please unzip the "`instances.zip`" archive into a directory of your choice. In addition, please unzip the "`Preresults.zip`" archive as a folder named "`Preresults`", ensuring that this folder is placed parallel to the "`bin`" folder. This step is essential because the computation of the Gap for heuristics relies on the MIP bounds, which are retrieved from the precomputed result files stored in the "`Preresults`" folder

3. Overview and Examples

In the following, we provide a comprehensive overview of the structure of each folder and detailed instructions for executing the program via the command line.

(1) instances

This folder contains all instance datasets utilized in our computational experiments. It includes 8 sub-folders, each corresponding to a distinct configuration examined in our study (please refer to *Section 5.1.1 Test instances* for further details). These sub-folders are named using the format "R_1_{M}_{S}", where "M" and "S" denote the proportions of medium- and short-span tasks, respectively. For example, in the sub-folder "~/instances/R_1_1_2/", the ratio among long-, medium-, and short-span tasks is consistently 1:1:2 for all instances. Each sub-folder contains two types of text files, named "{N}.txt" and "layout_{N}.txt", where "N" is the number of tasks, ranging from 5 to 140.

The "{N}.txt" file provides the necessary parameters for executing procedures, specifically the time required for an empty crane to move between any two stacks. These parameters are formatted as a "2N × 2N" matrix, where the element in the i-th row and j-th column indicates the time required for an empty crane to move from stack i to stack j. Without loss of generality, we designate the original stack and the target stack of task i as stack i and stack $\mathrm{N}+i$, $1 \leq i \leq \mathrm{N}$, respectively. For example, the file "~/instances/R_1_1_2/5.txt" represents an instance comprising 5 tasks, involving a total of 10 stacks. Consequently, the crane moving times are formatted as a 10×10 matrix. The element in the 4th row and 10th column is 7, indicating that it takes 7 minutes for an empty crane to move from the original stack of task 4 to the target stack of task 5.

```
# ~/instances/R_1_1_2/5.txt
0 3 1 1 2 1 2 2 4 6
3 0 4 4 3 2 2 1 2 3
1 4 0 1 2 2 2 3 5 7
1 4 1 0 1 2 2 3 5 7
2 3 2 1 0 2 1 2 4 6
1 2 2 2 2 0 2 2 3 5
2 2 2 2 1 2 0 1 3 5
2 1 3 3 2 2 1 0 2 4
4 2 5 5 4 3 3 2 0 2
6 3 7 7 6 5 5 4 2 0
```

The "layout_{N}.txt" file is not essential for running the procedure but just serves as an auxiliary resource to facilitate the identification of specific stack locations. For each instance, "layout_{N}.txt" records the precise coordinates of the original and target stacks for the N tasks involved. For example, the first line in "~/instances/R_1_1_2/layout_5.txt" reads "1 7 3 6 3", indicating that, in this instance, the original stack for task 1 is located at the 7th column and 3th row, while the target stack for task 1 is at the 6th column and 3th row.

```
# ~/instances/R_1_1_2/layout_5.txt
1 7 3 6 3
2 4 1 6 1
3 8 3 5 1
4 8 2 3 3
5 7 1 1 2
```

(2) Preresults

This folder contains our preliminary results, specifically the detailed experimental outcomes reported in the paper. It is important to note that the output results of the heuristics depend on the linear relaxation bounds derived from the MILP model. These bounds are stored within the "~/Preresults" directory. Therefore, before executing any heuristics, please ensure that "Preresults.zip" has been successfully extracted into the folder "~/Preresults". Failure to do so may result in inaccurate information being recorded in the result files.

(3) bin

This folder contains all the executable procedures for our research, including the developed MILP models and heuristics (the T-ALNS and the matheuristic). Below, we provide separate instructions on how to execute the MILP models and heuristics.

- Running MILP models

The executable files for MILP models are named MILP.a (for Linux) and MILP.exe (for Windows). To execute a MILP model, you should navigate to the "~/bin" directory and enter the command line, starting with the name of the executable file followed by **7** key arguments. The specific meanings and options of these arguments are outlined in Table 1.

Table 1. Input rules for command-line parameters (MILP models).

order	meaning	options
[1]	The proportion of medium-span tasks.	0, 1
[2]	The proportion of short-span tasks.	1, 2, 3, 4
[3]	The punishment multiplier when the crane's load is 1.	1.2, 1.4, 1.6, 1.8
[4]	The punishment multiplier when the crane's load is 2.	1.4, 1.6, 1.8, 2.0
[5]	The MILP model used.	OBM, TBM, B1, B2
[6]	The number of slabs.	from 5 to 140
[7]	The storage path of instances	user defined

*The 5th argument offers options of "B1" and "B2". "B1" signifies that the crane is restricted to performing single-load operations, while "B2" allows the crane to perform constrained double-load operations. The input value of the 6th argument should align with the number of tasks specified.

Here, we give an example to demonstrate how to solve the instance "~/instances/R_1_1_2/30.txt" using the [TBM] model. On the Linux platform, the execution relies on the "~/bin/MILP.a" file. Initially, it is necessary to specify two parameters to account for variations in the crane's travel time (refer to the definitions of m_1 and m_2 in *Section 5.1.1 Test instances* for further details). For example, you might set $m_1 = 1.4$ and $m_2 = 2.0$, which implies $\tau'_{\alpha\beta} = 1.4\,\tau_{\alpha\beta}$ and $\tau'_{\alpha\beta} = 2.0\,\tau_{\alpha\beta}$. Subsequently, enter these instance-specific parameters sequentially in the command line according to the aforementioned rules, and press "Enter" to run it. On the Windows platform, it works in a similar way but just simply replace the ".a" file with the ".exe" file in the command line, as shown below.

```
# Example (for Linux)
~/bin> ./MILP.a 1 2 1.4 2.0 TBM 30 ../instances


# Example (for Windows)
~/bin> MILP.exe 1 2 1.4 2.0 TBM 30 ../instances
```

- Running heuristics

Recognizing that different problem sizes may require distinct algorithmic parameter settings for optimal performance, we have compiled the algorithms into three separate executable files. Each executable is specifically configured with parameter combinations suited to one of the three problem scales. Solving instances of different sizes on different platforms necessitates the use of specific executable files, as outlined

in Table 2.

Table 2. Reference for selecting executable files

scale	method	Linux	Windows
Small	T-ALNS	ALNS_Small.a	ALNS_Small.exe
	matheuristic	MATH_Small.a	MATH_Small.exe
Medium	T-ALNS	ALNS_Medium.a	ALNS_Medium.exe
	matheuristic	MATH_Medium.a	MATH_Medium.exe
Large	T-ALNS	ALNS_Large.a	ALNS_Large.exe
	matheuristic	MATH_Large.a	MATH_Large.exe

To run the heuristics (either the T-ALNS algorithm or the matheuristic), you should navigate to the "~/bin" directory and enter the name of the specific executable file. For example, if you want to solve a medium-sized instance using the T-ALNS algorithm, please first input "ALNS_Medium.a" in the command line. Subsequently, proceed to enter **6** key arguments, as described in Table 3.

Table 3. Input rules for command-line parameters (MILP models).

order	meaning	options
[1]	The proportion of medium-span tasks.	0, 1
[2]	The proportion of short-span tasks.	1, 2, 3, 4
[3]	The punishment multiplier when the crane's load is 1.	1.2, 1.4, 1.6, 1.8
[4]	The punishment multiplier when the crane's load is 2.	1.4, 1.6, 1.8, 2.0
[5]	The number of slabs.	from 5 to 140
[6]	The storage path of instances	user defined

Continuing with the previous example, suppose you wish to use the T-ALNS algorithm to solve the instance "~/instances/R_1_1_2/30.txt" with settings of $m_1 = 1.4$ and $m_2 = 2.0$. The procedure "ALNS_Medium.a" is the most suitable as $\mathrm{N} = 30$ is a medium-sized instance. Input the command line as follows, and press "Enter" to run it.

```
# Example (for Linux)
~/bin> ./ALNS_Medium.a 1 2 1.4 2.0 30 ../instances


# Example (for Windows)
~/bin> ALNS_Medium.exe 1 2 1.4 2.0 30 ../instances
```

4. Results and Logs

Upon completion of the procedure, a folder named 'Results' will be generated in the main directory to store the summarized outcomes of the computation. Additionally, either an 'ALNSLogFile' or a 'MILPLogFile' folder will be created to document the detailed solution logs for the heuristics or MILP models, respectively.

(4) Results

This folder is designated for storing concise results from the MILP models and heuristics. Take an example to illustrate, if you want to solve the instance "~/instances/R_1_1_2/30.txt" using the TBM model with the settings $m_1 = 1.4$ and $m_2 = 2.0$, simply enter the command "MILP.a 1 2 1.4 2.0 TBM 30 ../instancse" and wait for the procedure to complete execution. Afterwards, two levels of directories and a text file will be automatically generated within the "~/Results" directory, specifically "~/Results/R_1_1_2/F_1.4_2.0/TBM_30.txt". The text file "TBM_30.txt" succinctly reports the results of the execution of TBM model, including the best solution obtained, the best lower bound, the gap, and the computational time. If you execute the T-ALNS algorithm or the matheuristic, the result files "ALNS_{N}.txt" or "MATH_{N}.txt" will be located in the corresponding sub-folder under "~/Results". The information contained in these result files is described below.

```
# ~/Results/R_1_1_2/F_1.4_2.0/TBM_30.txt
178.8   #the best solution obtained
178.8   #the best lower bound
0       #the gap
33.6679 #the computational time (second)
```

```
# ~/Results/R_1_1_2/F_1.4_2.0/ALNS_30.txt
178.8   #the best solution obtained
0       #the gap
14.63   #the computational time (second)
```

```
# ~/Results/R_1_1_2/F_1.4_2.0/MATH_30.txt
178.8   #the best solution obtained
0       #the gap
```

```
4.69    #the total computational time (second)
4.43    #the time overhead for pre-execution of T-ALNS (second)
0.26    #the time overhead for execution of [R-TBM] model (second)
```

(5) MILPLogFile

This directory is designated for storing log files generated during the execution of MILP models. Maintaining these logs is essential for tracking the execution process and facilitating debugging when necessary.

(6) ALNSLogFile

Analogous to the MILPLogFile, this directory is reserved for storing log files produced by heuristic algorithm executions. These logs are vital for monitoring the heuristic processes and addressing any troubleshooting needs.

5. Others

If you have any questions regarding our procedures, please do not hesitate to contact us via the email *Dong_Z3664@mail.nwpu.edu.cn*. We are more than happy to assist you!