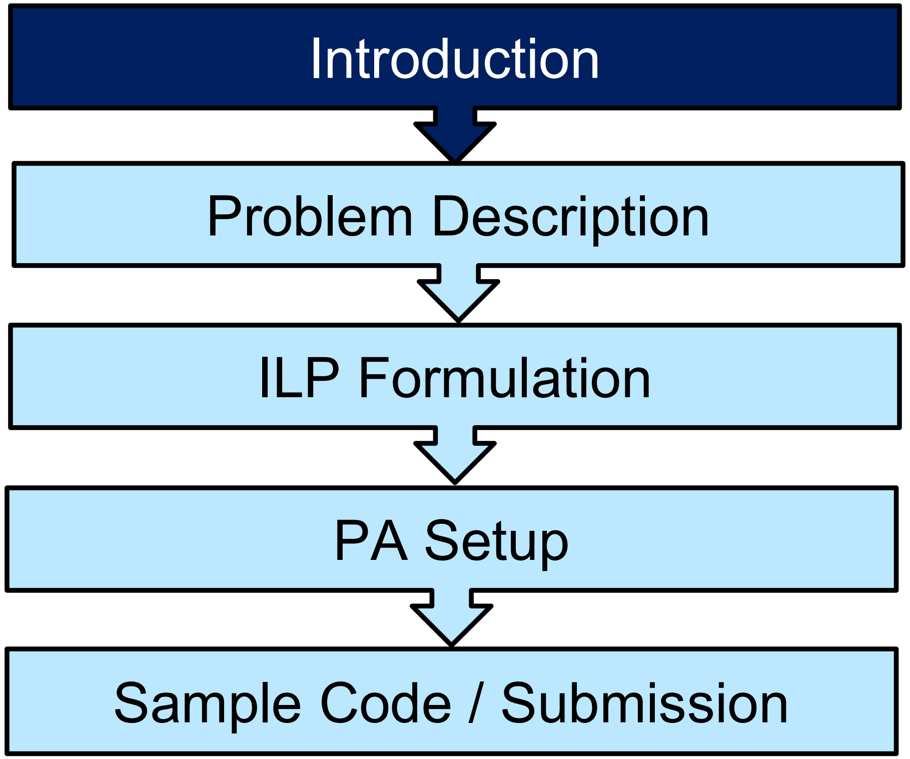
Programming Assignment 2
Fixed-Outline Incremental ILP-Based Floorplanning

TA: Yu-Hsiang Huang (johnnyhuang1007@gmail.com)

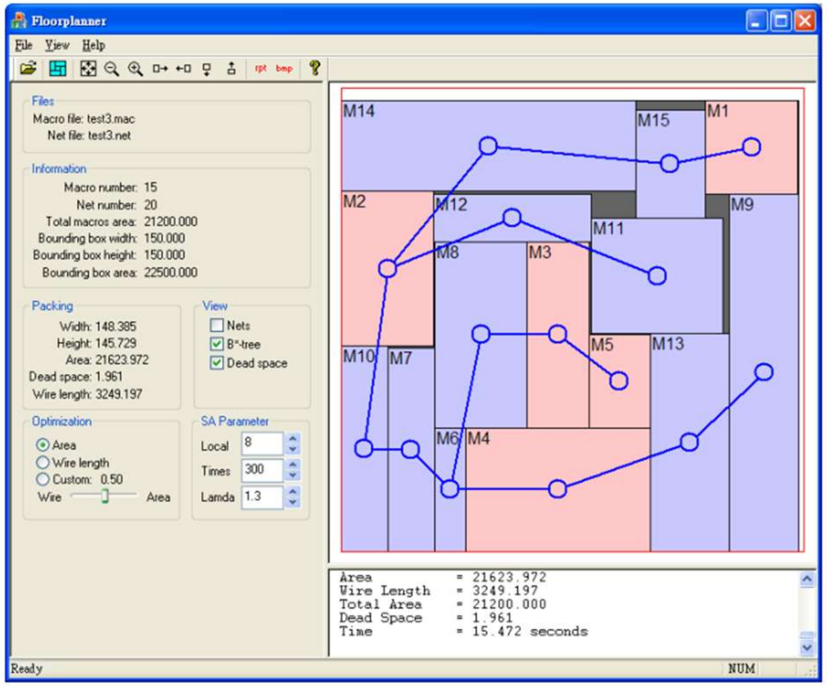
2025/10/15

Outline


Introduction
Problem Description
ILP Formulation
PA Setup
Sample Code / Submission

Introduction – Floorplanning

- Part of Physical Design flow that:
  - Decide the position of target macros
  - Minimize the dataflow between each macros
- Input:
  - A set of modules,
    - hard or soft
    - rectangle or rectilinear
  - Pin location of each modules
  - A set of netlist
- Output:
  - reduce wirelength, minimize area
- Methodology:
  - Heuristic Method, e.g., Simulated Annealing
  - Analytical Method, e.g., Gradient Descend
  - Mathematic Method, e.g., ILP Formulation



Files
Macro file: test3.mac
Net file: test3.net

Information
Macro number: 15
Net number: 20
Total macros area: 21200.000
Bounding box width: 150.000
Bounding box height: 150.000
Bounding box area: 22500.000

Packing
Width: 148.385
Height: 145.729
Area: 21623.972
Dead space: 1.961
Wire length: 3249.197

View
□ Nets
☑ B*-tree
☑ Dead space

Optimization
● Area
○ Wire length
○ Custom: 0.50
Wire — Area

SA Parameter
Local 8
Times 300
Lamda 1.3

```
Area            = 21623.972
Wire Length     = 3249.197
Total Area      = 21200.000
Dead Space      = 1.961
Time            = 15.472 seconds
```

By Tung-Chieh Chen

Introduction – Integer Linear Programming

- An Optimization Problem with standard form is in the form of:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p \end{array}$$

- Where $f_0(x)$ is objective function, $f_i(x)$ is in-equality constraint, $h_i(x)$ is equality constraint
- When the objective and constraint functions are linear functions, and the variables are all integer, then it's an Integer Linear Programming Problem

$$\begin{array}{ll} \text{minimize} & c^T x + d \\ \text{subject to} & Gx \preceq h \\ & Ax = b, \end{array}$$

$u \preceq v \ \ means \ u_i \leq v_i \ for \ i = 1, \ldots, m$

Introduction – Integer Linear Programming

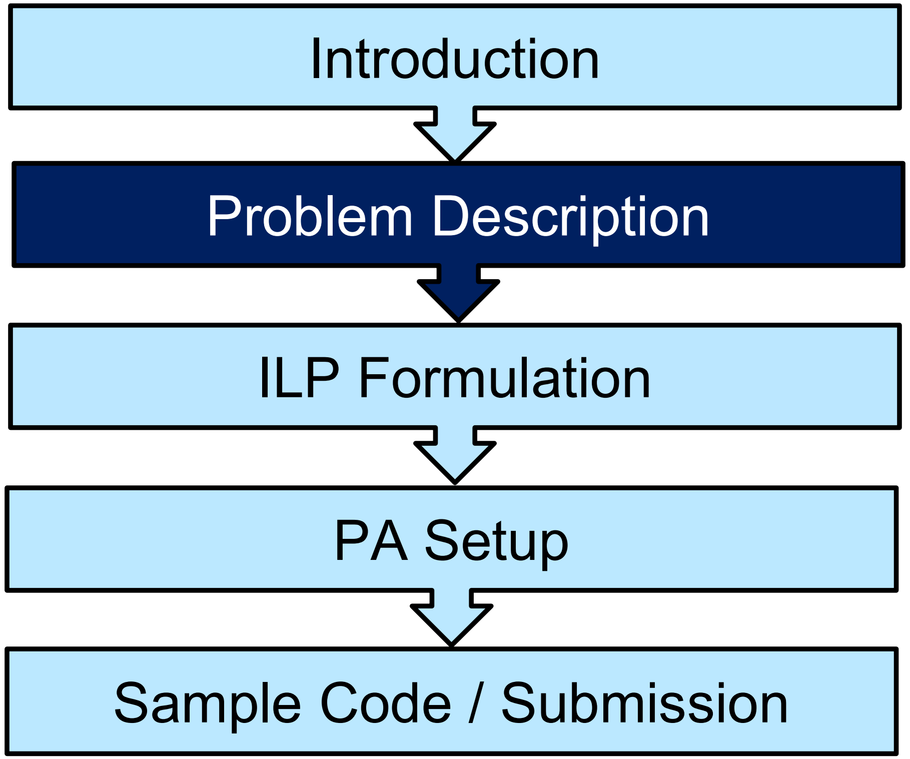
- Objective and constraints are linear functions of the variables.

- The Problem is NP-hard
  - Runtime grows exponentially as the number of variable increases

- Popular solvers: Gurobi, CPLEX, OR-Tools

- For this assignment, we expect students to solve a floorplanning problem using ILP method

Outline



Introduction

Problem Description

ILP Formulation

PA Setup

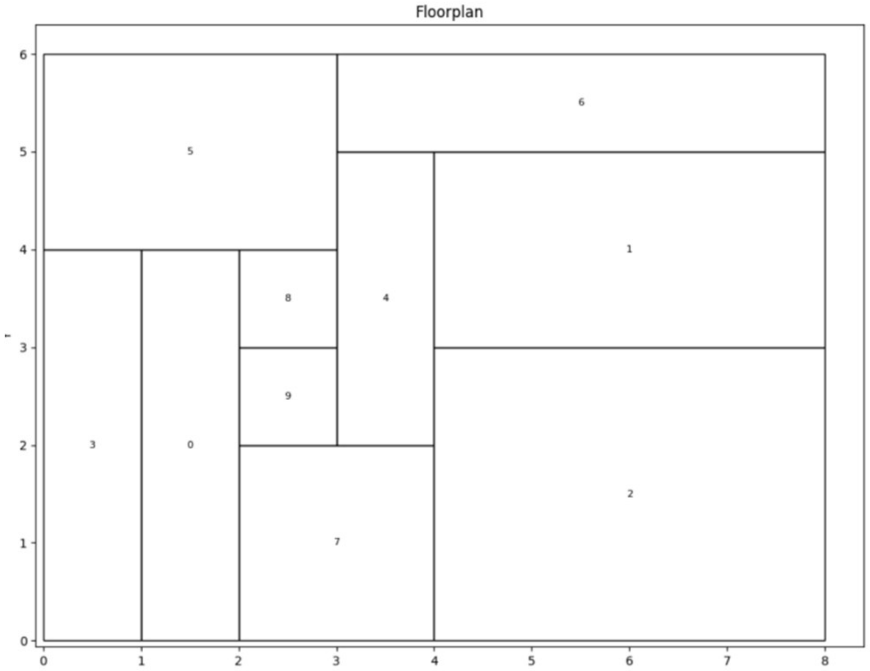
Sample Code / Submission

Problem Statement

- Input:
  - Set of rotatable hard modules, fixed-outline [W,H], problem category
- Objective:
  - minimize the bounding area of a floorplan, Bounding area: $A = \left(\max_{i\in\mathcal{M}} (x_i + w_i')\right) \cdot \left(\max_{i\in\mathcal{M}} (y_i + h_i')\right).$
- Constraint:
  - Every modules should be placed inside given outline
  - No overlaps between each modules
- Two problem category:
  - 0: Find an optimal result (minimum bounding area) of the target module set
  - 1: Find a feasible floorplan that minimizes the bounding area as much as possible
- Outputs:
  - coordinates and rotation for all modules

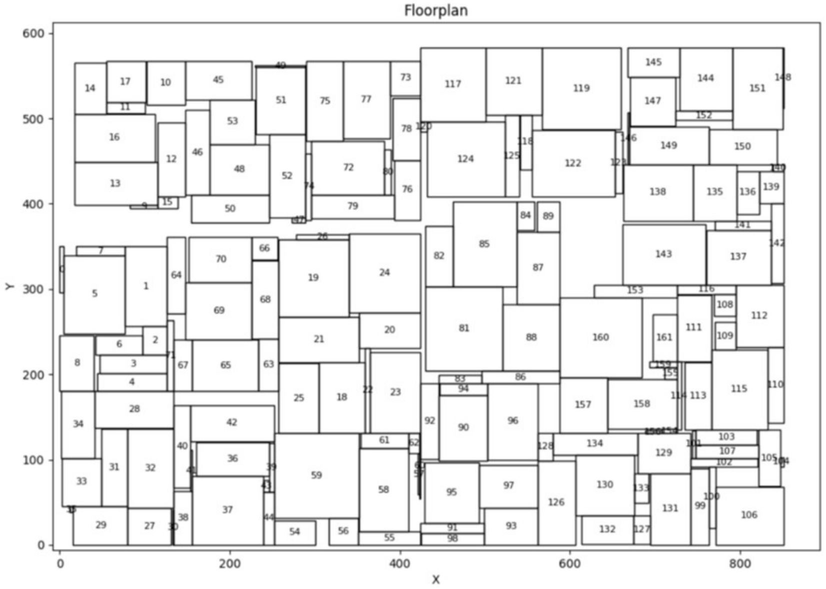
Problem Category 0

- The covered area of a given outline is equal to the total area of modules
  - W * H = Area(modules)
- In other word, every floorplanning result should achieve 100% utilization
- Let M be the set of input modules: $4 \leq |M| \leq 11$
- Runtime Limit: 1 hour
- Example result:



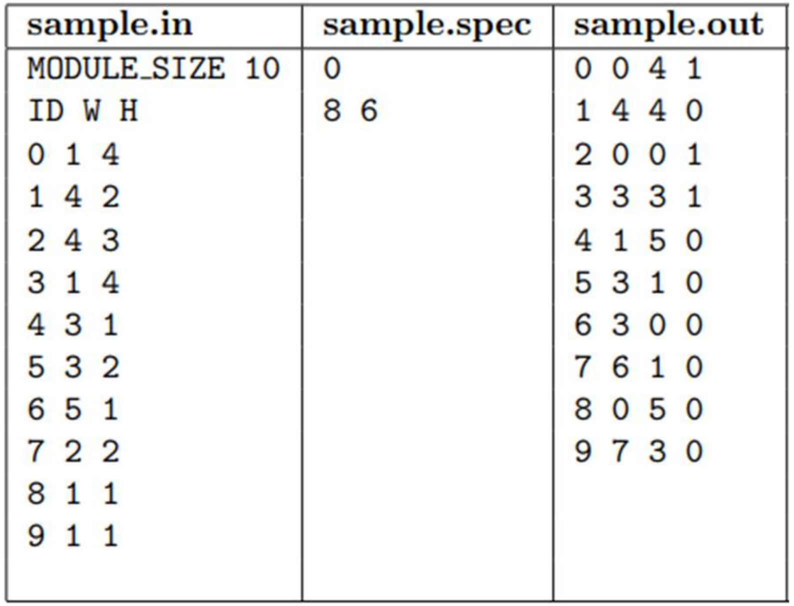
Problem Category 1

- The given outline is larger than the optimal solution
  - W * H > Area(bounding area)
- Let M be the set of input modules: $20 \leq |M| \leq 500$
- Runtime Limit: 1 hour
  - It is impossible to solve this problem by only by a single ILP formulation
- Example result:



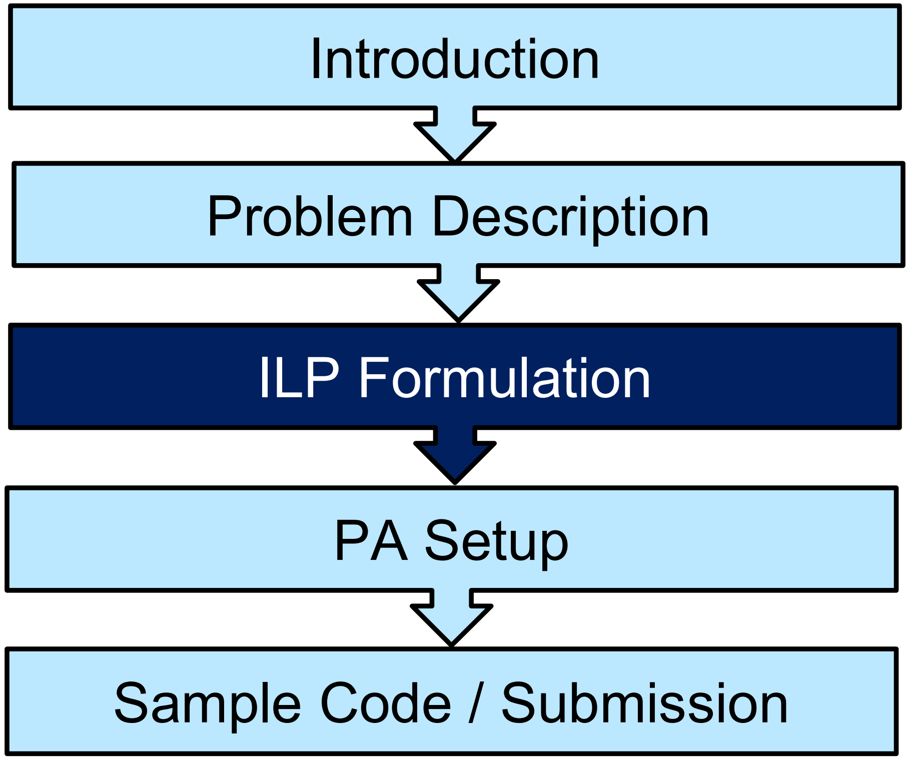
Input/Output Format

- .in: List of all modules
  - First line: MODULE_SIZE [num]
  - Second line: column headers
  - Following Lines: module information
    - ID always starts from 0 and increases by 1
    - W and H are always integers
- .spec:
  - First line: problem category (0 or 1)
  - Second line: given outline size W, H
- .out: placement result (ID pos.x pos.y rotatation)
  - (pos.x, pos.y): coordinates of the lower-left corner
  - rotatation = 0 → no rotation, rotatation = 1 → rotated 90° counterclockwise
- Execution command example:
  - bin/fp ./input/sample.in ./spec/sample.spec ./sample.out



sample.in	sample.spec	sample.out
MODULE_SIZE 10	0	0 0 4 1
ID W H	8 6	1 4 4 0
0 1 4		2 0 0 1
1 4 2		3 3 3 1
2 4 3		4 1 5 0
3 1 4		5 3 1 0
4 3 1		6 3 0 0
5 3 2		7 6 1 0
6 5 1		8 0 5 0
7 2 2		9 7 3 0
8 1 1		
9 1 1		

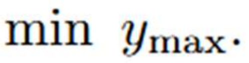
Outline


Introduction
Problem Description
ILP Formulation
PA Setup
Sample Code / Submission

ILP Formulation

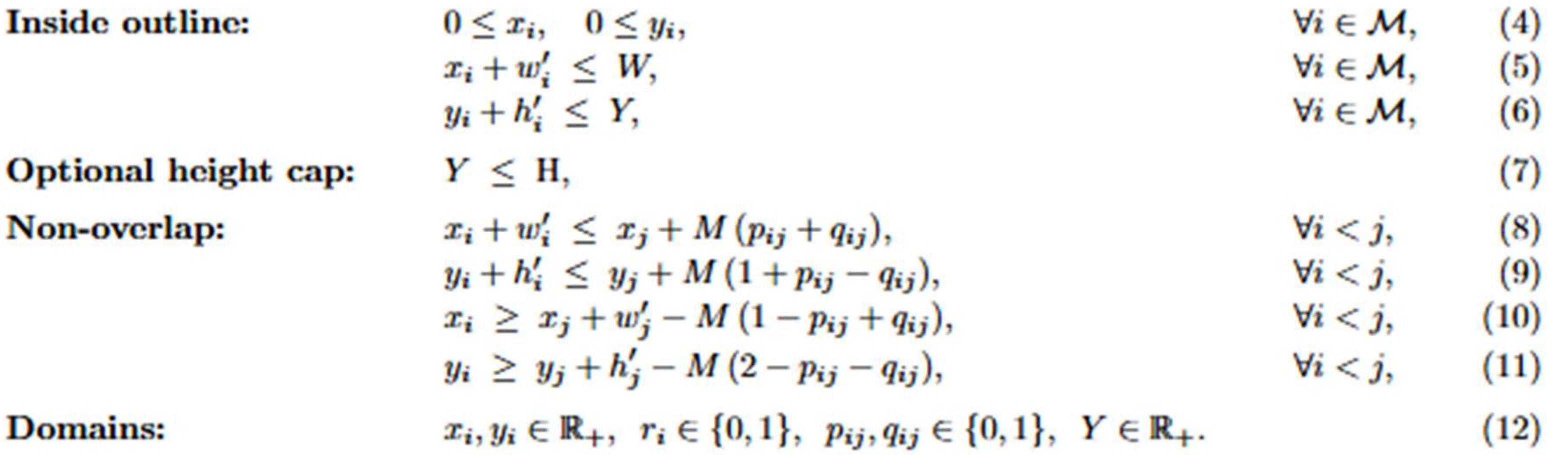
- Area objective is non-linear
  - Alternative way: fix the floorplan’s width and minimize floorplan’s height instead

Objective



$$\min \; y_{\max}.$$

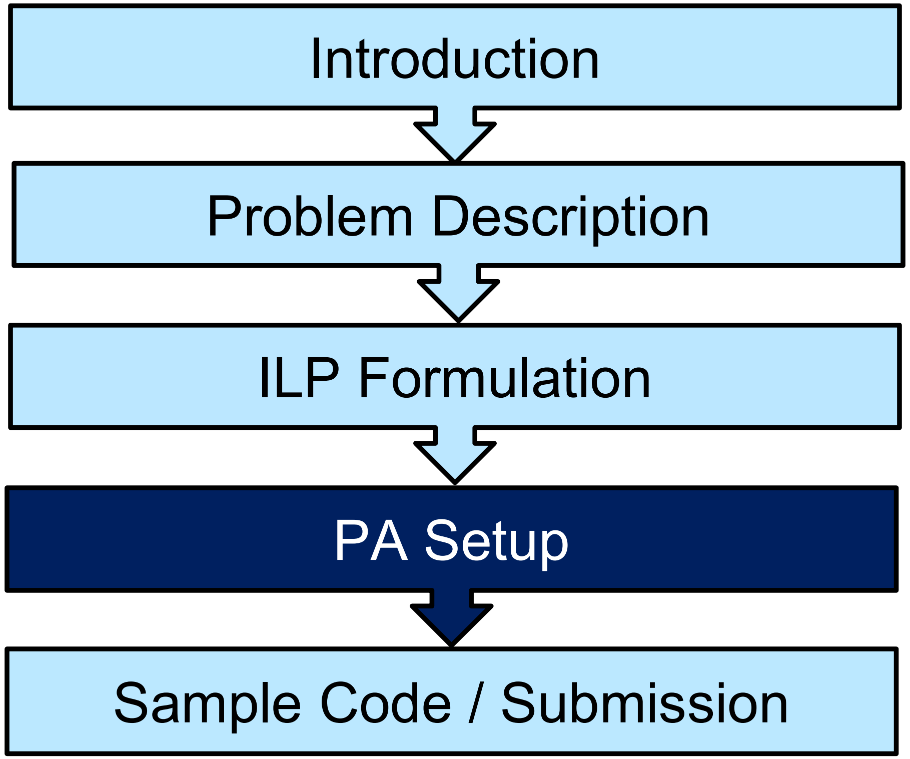
Constraints



Inside outline:	$0 \le x_i, \quad 0 \le y_i,$	$\forall i \in \mathcal{M},$	(4)
	$x_i + w_i' \le W,$	$\forall i \in \mathcal{M},$	(5)
	$y_i + h_i' \le Y,$	$\forall i \in \mathcal{M},$	(6)
Optional height cap:	$Y \le \mathrm{H},$		(7)
Non-overlap:	$x_i + w_i' \le x_j + M\,(p_{ij} + q_{ij}),$	$\forall i < j,$	(8)
	$y_i + h_i' \le y_j + M\,(1 + p_{ij} - q_{ij}),$	$\forall i < j,$	(9)
	$x_i \ge x_j + w_j' - M\,(1 - p_{ij} + q_{ij}),$	$\forall i < j,$	(10)
	$y_i \ge y_j + h_j' - M\,(2 - p_{ij} - q_{ij}),$	$\forall i < j,$	(11)
Domains:	$x_i, y_i \in \mathbb{R}_+, \;\; r_i \in \{0,1\}, \;\; p_{ij}, q_{ij} \in \{0,1\}, \;\; Y \in \mathbb{R}_+.$		(12)

M is a sufficiently large number

Outline



Introduction

Problem Description

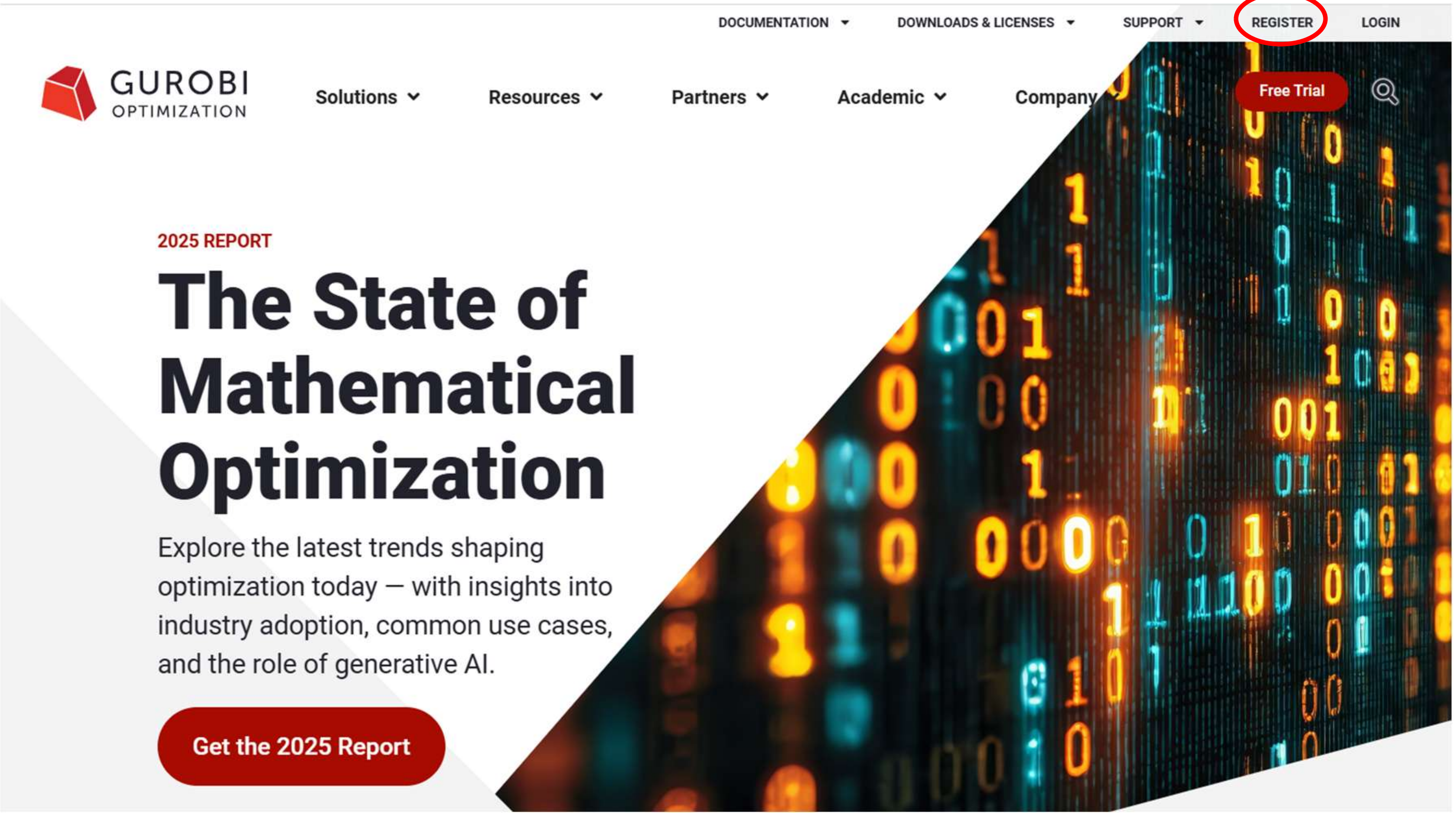
ILP Formulation

PA Setup

Sample Code / Submission

Gurobi Account Registration

- Register your account: website



DOCUMENTATION ▾ DOWNLOADS & LICENSES ▾ SUPPORT ▾ REGISTER LOGIN

GUROBI OPTIMIZATION

Solutions ⌄ Resources ⌄ Partners ⌄ Academic ⌄ Company ⌄ Free Trial

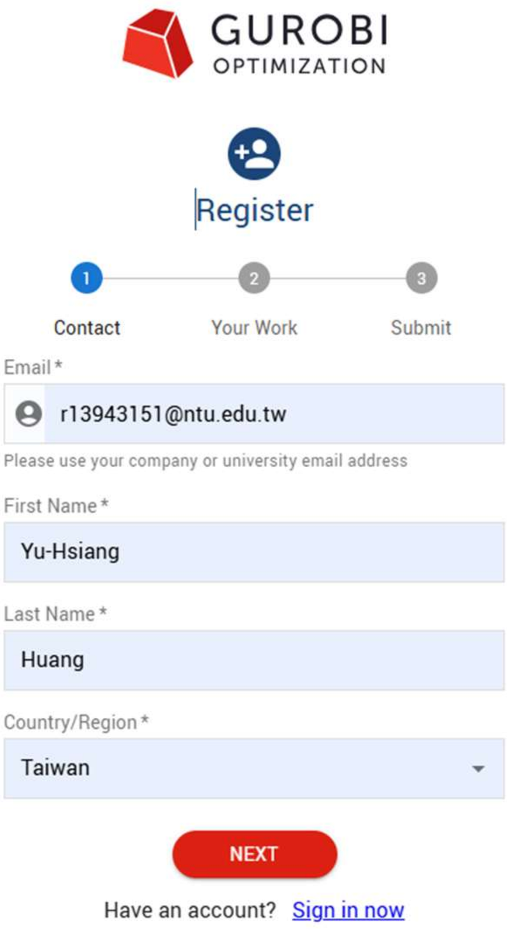
2025 REPORT

The State of Mathematical Optimization

Explore the latest trends shaping optimization today — with insights into industry adoption, common use cases, and the role of generative AI.

Get the 2025 Report

Gurobi Account Registration



Email *

r13943151@ntu.edu.tw

Please use your company or university email address

First Name *

Yu-Hsiang

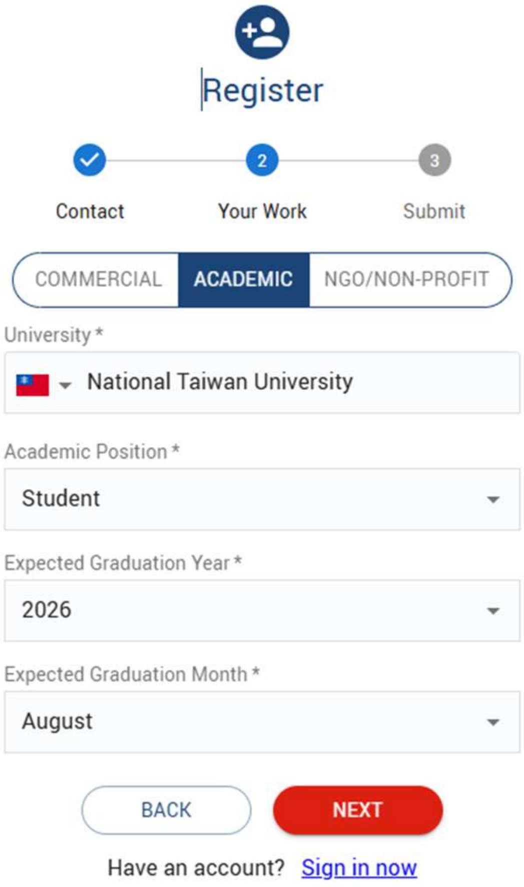
Last Name *

Huang

Country/Region *

Taiwan

Have an account? Sign in now



University *

National Taiwan University

Academic Position *

Student

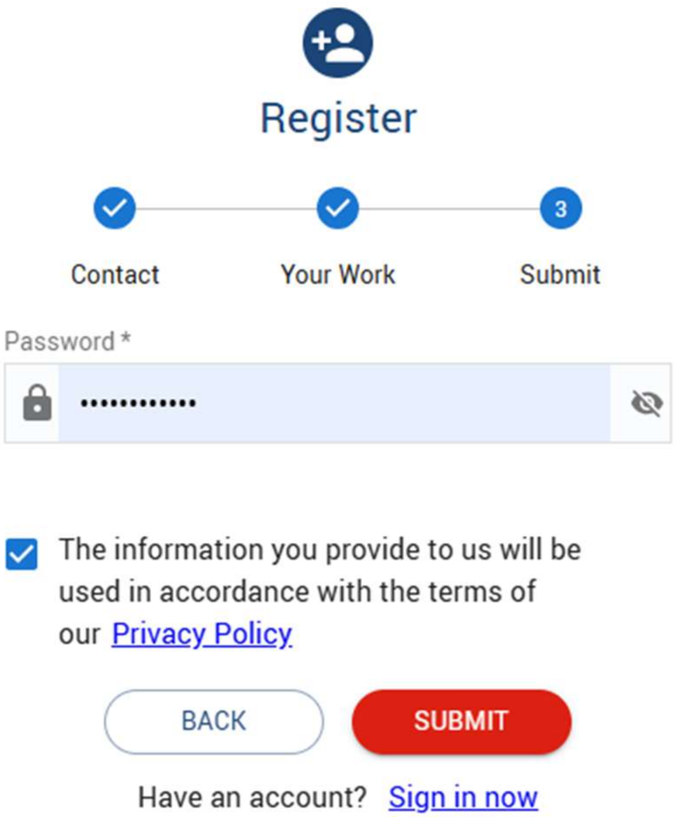
Expected Graduation Year *

2026

Expected Graduation Month *

August

Have an account? Sign in now



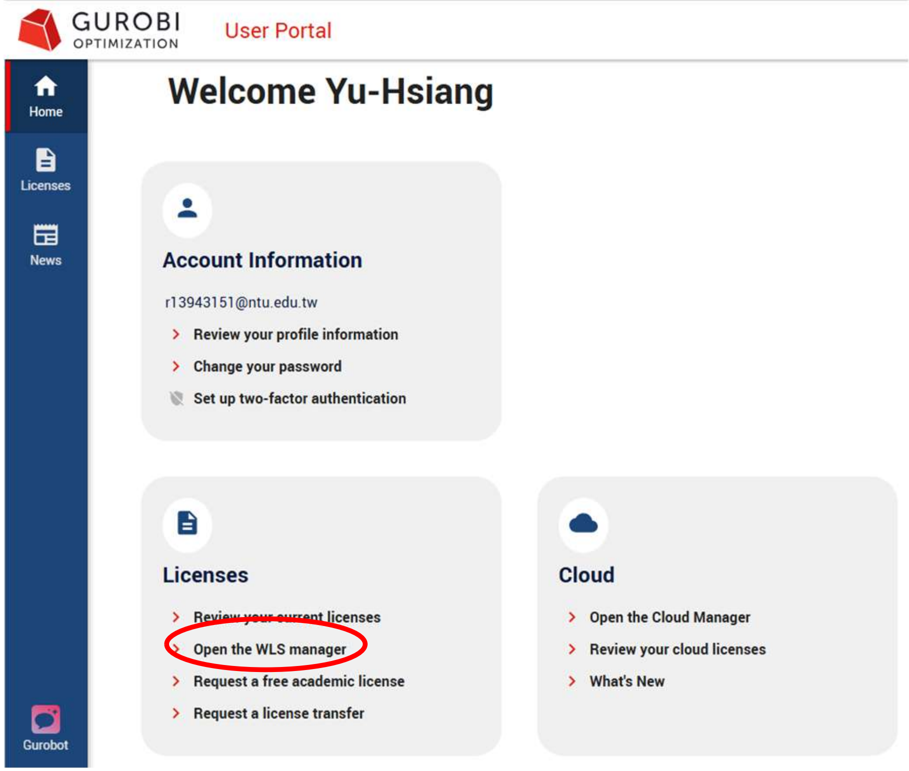
Password *

•••••••••••••

☑ The information you provide to us will be used in accordance with the terms of our Privacy Policy

Have an account? Sign in now

License Request



GUROBI OPTIMIZATION User Portal

Welcome Yu-Hsiang

Account Information

r13943151@ntu.edu.tw

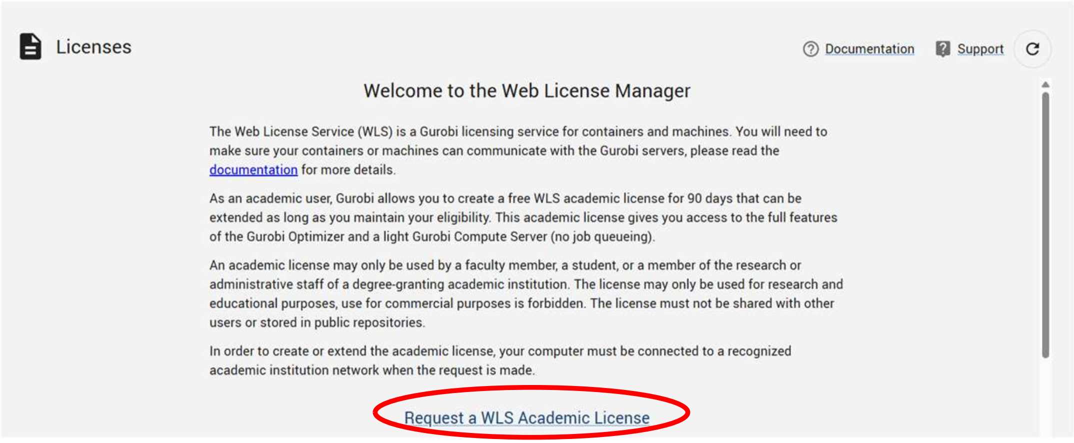
- Review your profile information
- Change your password
- Set up two-factor authentication

Licenses

- ~~Review your current~~ licenses
- Open the WLS manager
- Request a free academic license
- Request a license transfer

Cloud

- Open the Cloud Manager
- Review your cloud licenses
- What's New



Licenses

Welcome to the Web License Manager

The Web License Service (WLS) is a Gurobi licensing service for containers and machines. You will need to make sure your containers or machines can communicate with the Gurobi servers, please read the documentation for more details.

As an academic user, Gurobi allows you to create a free WLS academic license for 90 days that can be extended as long as you maintain your eligibility. This academic license gives you access to the full features of the Gurobi Optimizer and a light Gurobi Compute Server (no job queueing).

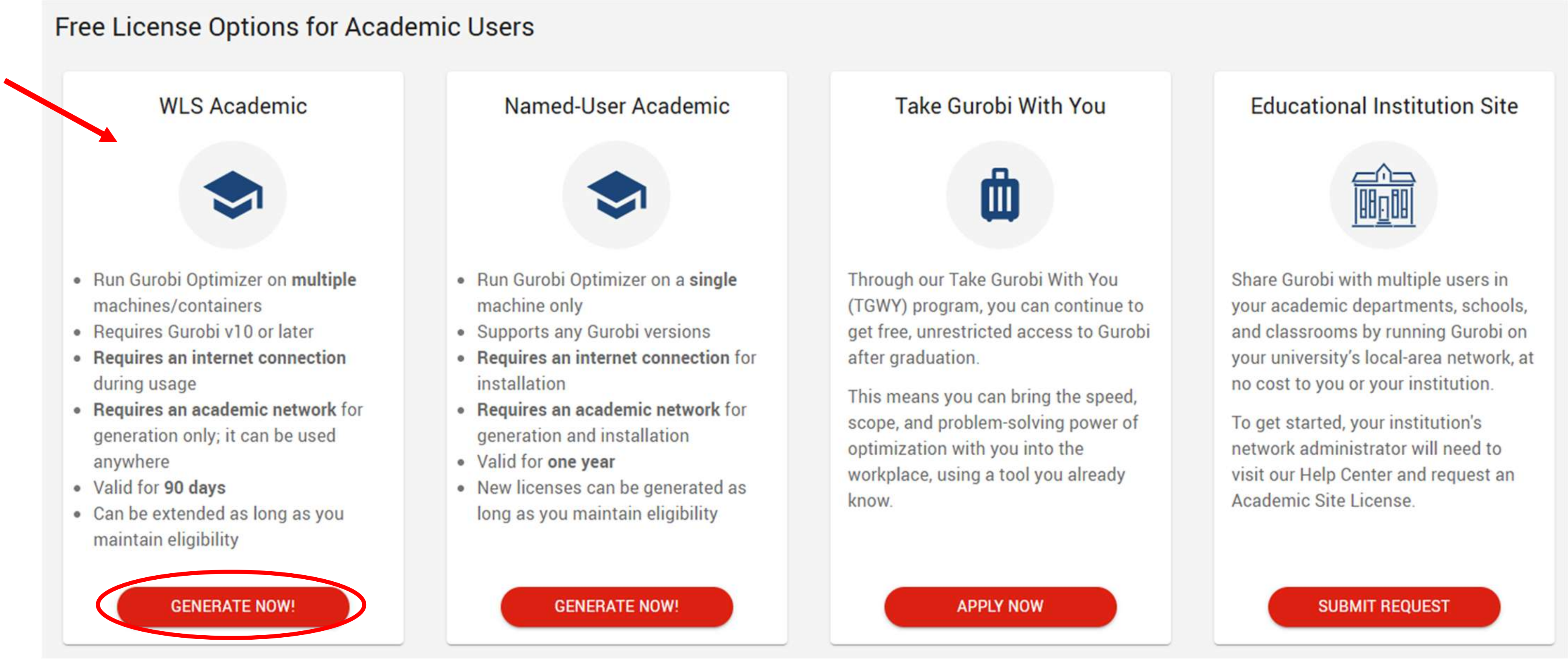
An academic license may only be used by a faculty member, a student, or a member of the research or administrative staff of a degree-granting academic institution. The license may only be used for research and educational purposes, use for commercial purposes is forbidden. The license must not be shared with other users or stored in public repositories.

In order to create or extend the academic license, your computer must be connected to a recognized academic institution network when the request is made.

Request a WLS Academic License

License Request

Choose WLS Academic License instead of Named-User Academic License



Free License Options for Academic Users

WLS Academic

- Run Gurobi Optimizer on **multiple** machines/containers
- Requires Gurobi v10 or later
- **Requires an internet connection** during usage
- **Requires an academic network** for generation only; it can be used anywhere
- Valid for **90 days**
- Can be extended as long as you maintain eligibility

GENERATE NOW!

Named-User Academic

- Run Gurobi Optimizer on a **single** machine only
- Supports any Gurobi versions
- **Requires an internet connection** for installation
- **Requires an academic network** for generation and installation
- Valid for **one year**
- New licenses can be generated as long as you maintain eligibility

GENERATE NOW!

Take Gurobi With You

Through our Take Gurobi With You (TGWY) program, you can continue to get free, unrestricted access to Gurobi after graduation.

This means you can bring the speed, scope, and problem-solving power of optimization with you into the workplace, using a tool you already know.

APPLY NOW

Educational Institution Site

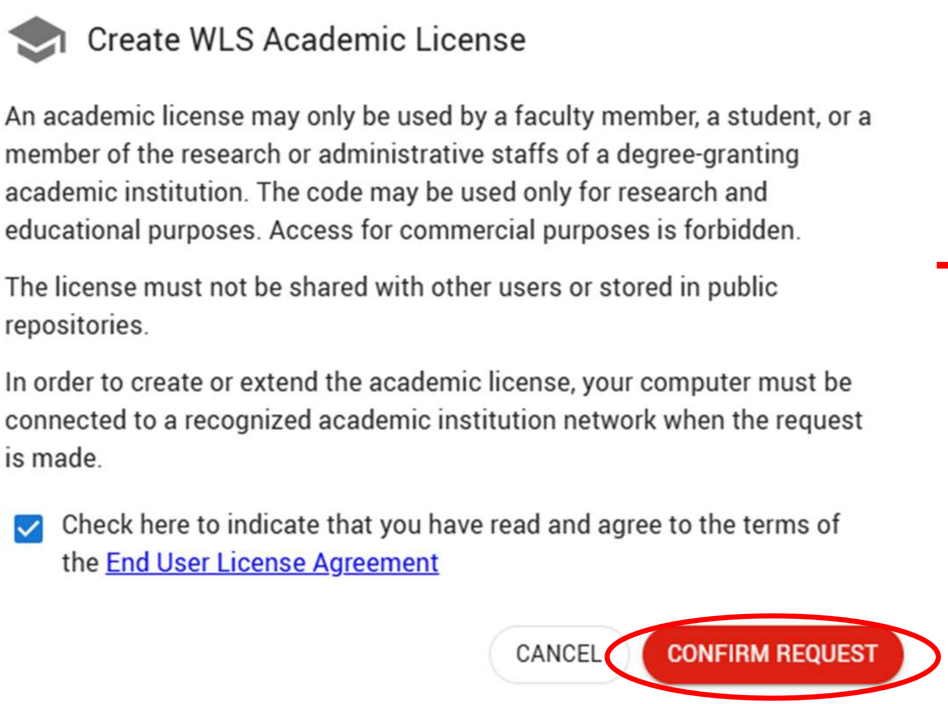
Share Gurobi with multiple users in your academic departments, schools, and classrooms by running Gurobi on your university's local-area network, at no cost to you or your institution.

To get started, your institution's network administrator will need to visit our Help Center and request an Academic Site License.

SUBMIT REQUEST

License Request

- Connect to NTU VPN before confirming request



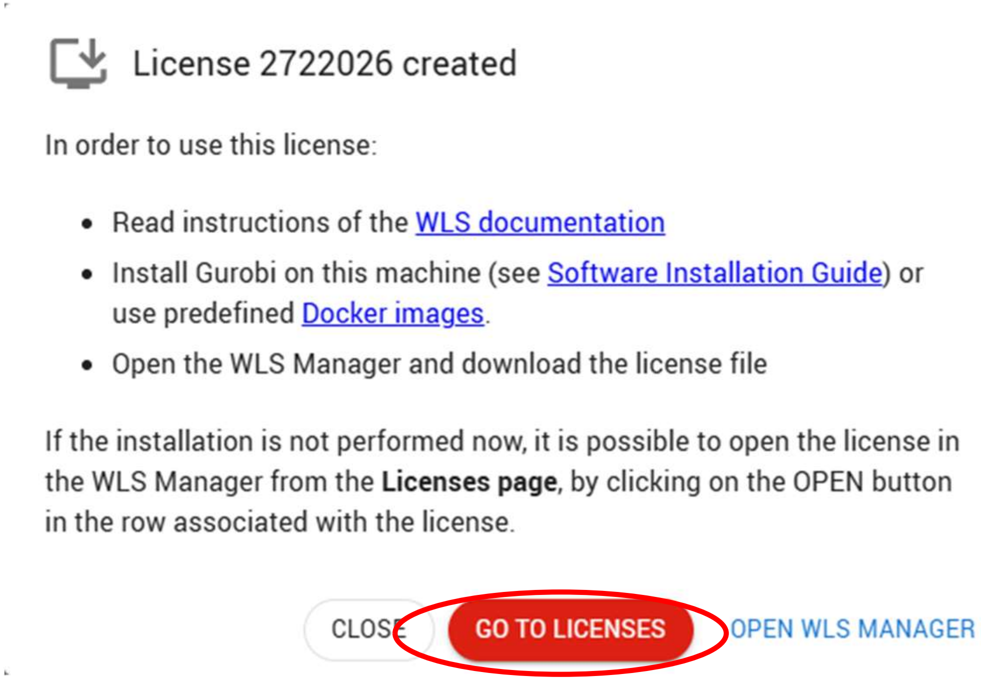
Create WLS Academic License

An academic license may only be used by a faculty member, a student, or a member of the research or administrative staffs of a degree-granting academic institution. The code may be used only for research and educational purposes. Access for commercial purposes is forbidden.

The license must not be shared with other users or stored in public repositories.

In order to create or extend the academic license, your computer must be connected to a recognized academic institution network when the request is made.

☑ Check here to indicate that you have read and agree to the terms of the End User License Agreement



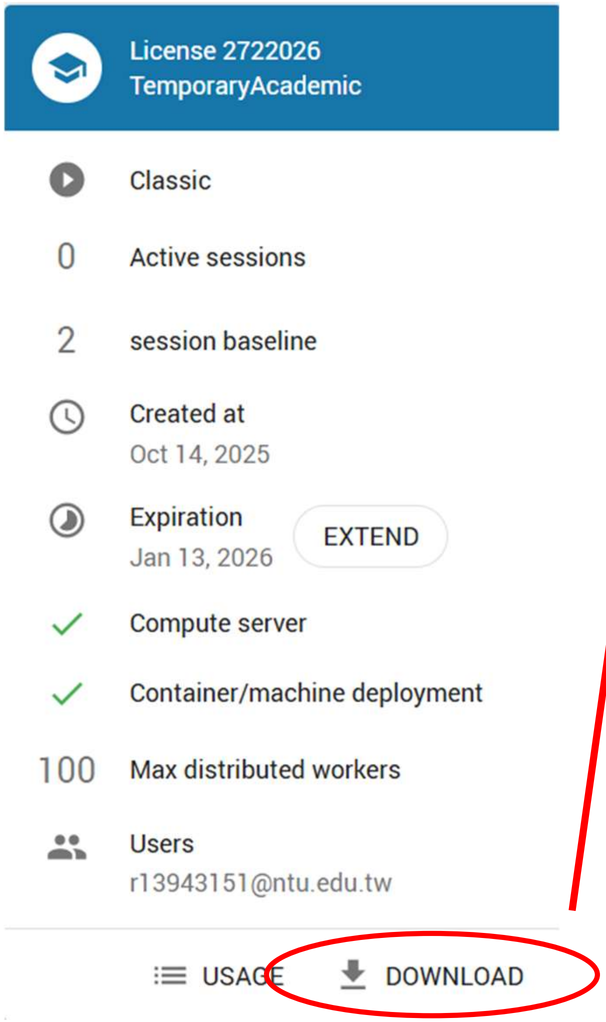
License 2722026 created

In order to use this license:

- Read instructions of the WLS documentation
- Install Gurobi on this machine (see Software Installation Guide) or use predefined Docker images.
- Open the WLS Manager and download the license file

If the installation is not performed now, it is possible to open the license in the WLS Manager from the **Licenses page**, by clicking on the OPEN button in the row associated with the license.

License Request



License 2722026
TemporaryAcademic

Classic

0 Active sessions

2 session baseline

Created at
Oct 14, 2025

Expiration
Jan 13, 2026

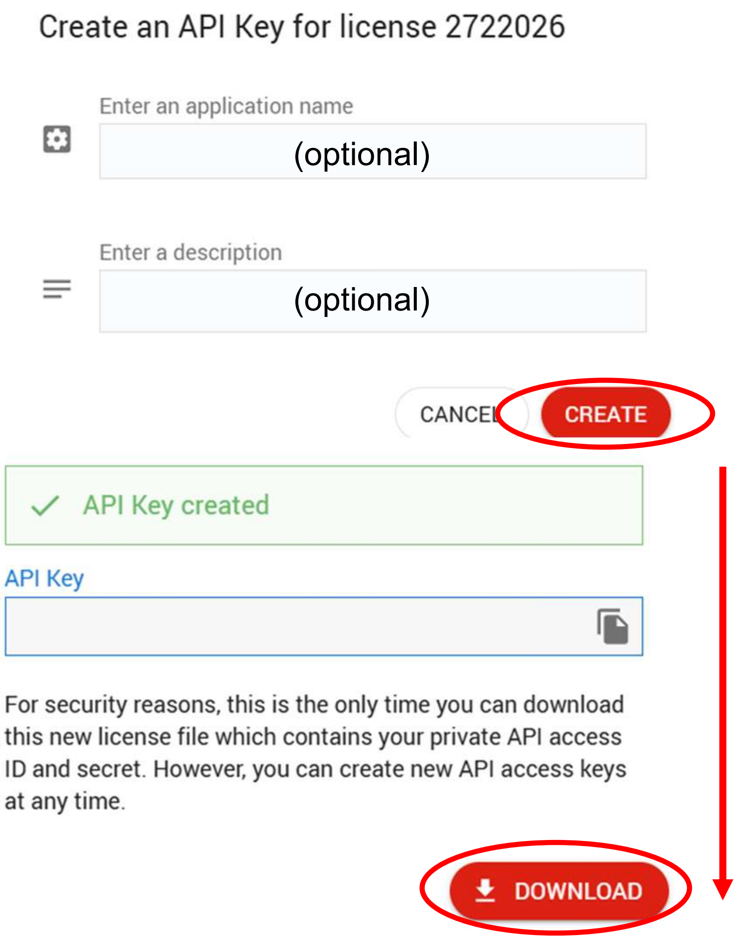
EXTEND

✓ Compute server

✓ Container/machine deployment

100 Max distributed workers

Users
r13943151@ntu.edu.tw



Create an API Key for license 2722026

Enter an application name
(optional)

Enter a description
(optional)

✓ API Key created

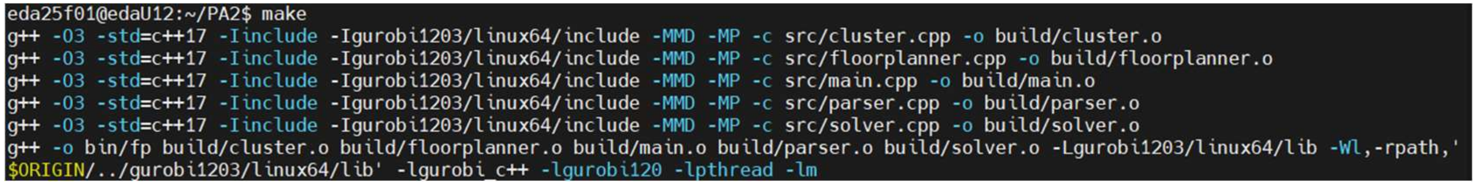
API Key

For security reasons, this is the only time you can download this new license file which contains your private API access ID and secret. However, you can create new API access keys at any time.

At this point, you should find the file “gurobi.lic” in your Downloads folder

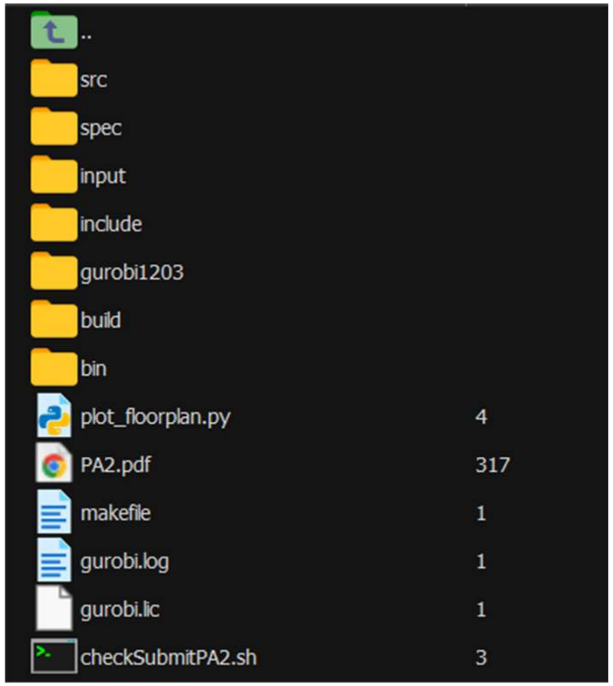
Programming Assignment Setup

- Download “PA2.tar” from NTU COOL
- Upload the file to the EDA union server and unzip it:
  - eda25fXX@edaU12:~$ tar –xvf ./PA2.tar
- Upload “gurobi.lic” to the PA2 folder
- Check if you can compile the sample code:
  - eda25fXX@edaU12:~/PA2$ make



```
eda25f01@edaU12:~/PA2$ make
g++ -O3 -std=c++17 -Iinclude -Igurobi1203/linux64/include -MMD -MP -c src/cluster.cpp -o build/cluster.o
g++ -O3 -std=c++17 -Iinclude -Igurobi1203/linux64/include -MMD -MP -c src/floorplanner.cpp -o build/floorplanner.o
g++ -O3 -std=c++17 -Iinclude -Igurobi1203/linux64/include -MMD -MP -c src/main.cpp -o build/main.o
g++ -O3 -std=c++17 -Iinclude -Igurobi1203/linux64/include -MMD -MP -c src/parser.cpp -o build/parser.o
g++ -O3 -std=c++17 -Iinclude -Igurobi1203/linux64/include -MMD -MP -c src/solver.cpp -o build/solver.o
g++ -o bin/fp build/cluster.o build/floorplanner.o build/main.o build/parser.o build/solver.o -Lgurobi1203/linux64/lib -Wl,-rpath,'
$ORIGIN/../gurobi1203/linux64/lib' -lgurobi_c++ -lgurobi120 -lpthread -lm
```

Though some of the function is not completed,
the sample code is still compliable

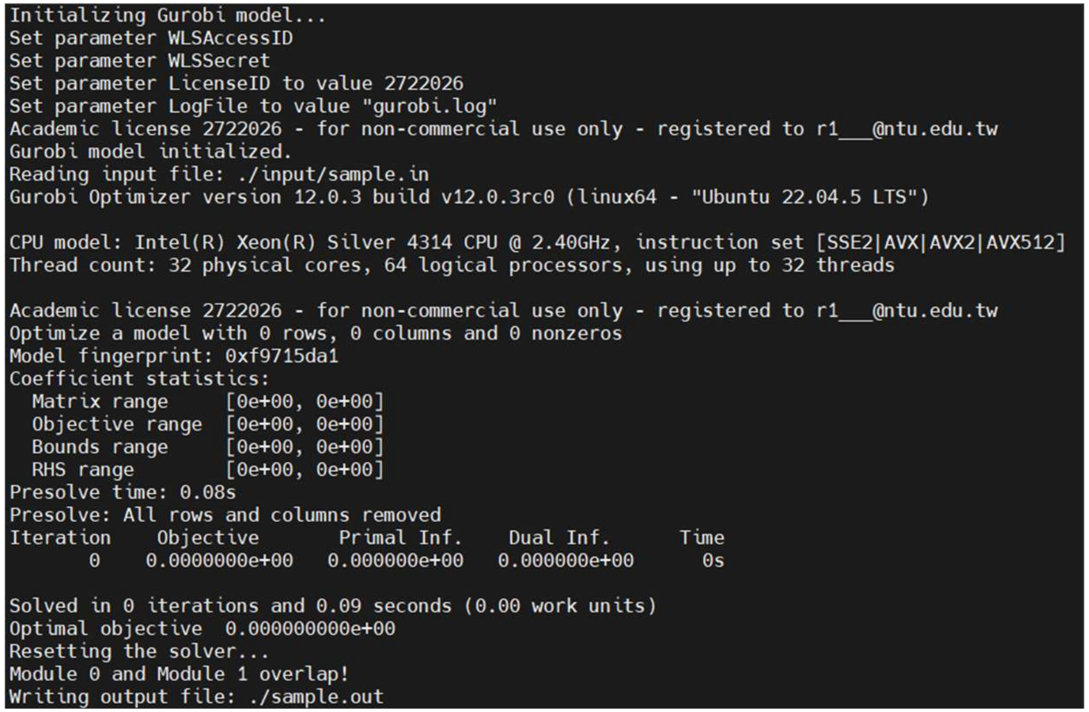


..	
src	
spec	
input	
include	
gurobi1203	
build	
bin	
plot_floorplan.py	4
PA2.pdf	317
makefile	1
gurobi.log	1
gurobi.lic	1
checkSubmitPA2.sh	3

After “make”, you should find the files above under your PA2 folder

Programming Assignment Setup

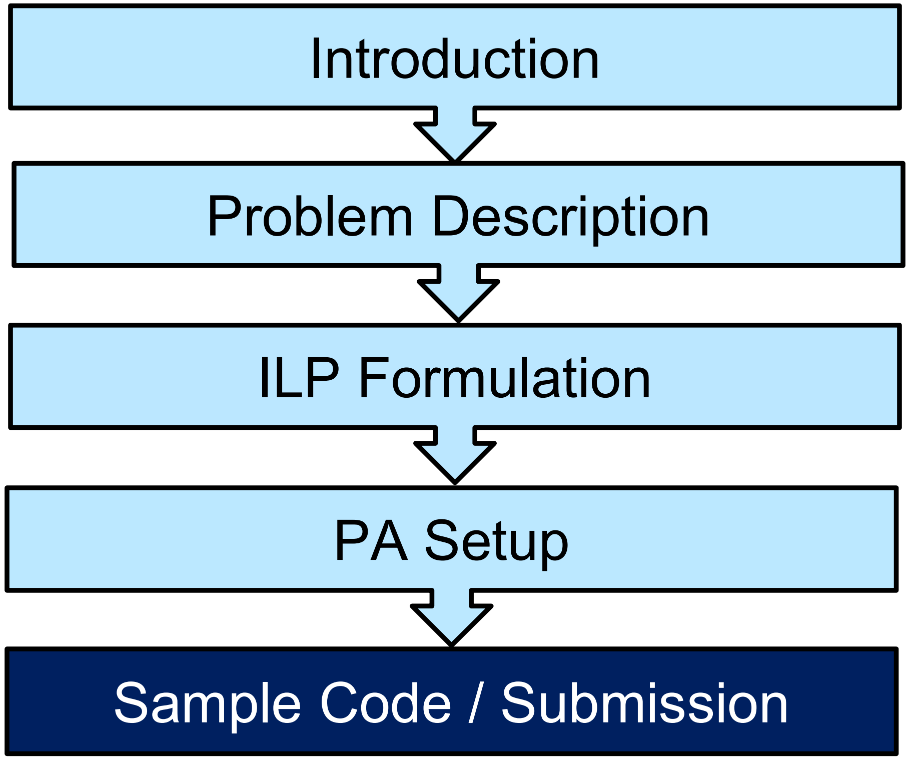
- Type the following command and check if the terminal outputs the following lines:
  - eda25fXX@edaU12:~/PA2$ ./bin/fp ./input/sample.in ./spec/sample.spec ./sample.out



```
Initializing Gurobi model...
Set parameter WLSAccessID
Set parameter WLSSecret
Set parameter LicenseID to value 2722026
Set parameter LogFile to value "gurobi.log"
Academic license 2722026 - for non-commercial use only - registered to r1___@ntu.edu.tw
Gurobi model initialized.
Reading input file: ./input/sample.in
Gurobi Optimizer version 12.0.3 build v12.0.3rc0 (linux64 - "Ubuntu 22.04.5 LTS")

CPU model: Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz, instruction set [SSE2|AVX|AVX2|AVX512]
Thread count: 32 physical cores, 64 logical processors, using up to 32 threads

Academic license 2722026 - for non-commercial use only - registered to r1___@ntu.edu.tw
Optimize a model with 0 rows, 0 columns and 0 nonzeros
Model fingerprint: 0xf9715da1
Coefficient statistics:
  Matrix range     [0e+00, 0e+00]
  Objective range  [0e+00, 0e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [0e+00, 0e+00]
Presolve time: 0.08s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.09 seconds (0.00 work units)
Optimal objective  0.000000000e+00
Resetting the solver...
Module 0 and Module 1 overlap!
Writing output file: ./sample.out
```

If you see the same output, you're ready to start your assignment.
If you get a different result, please briefly describe your setup process and email the TA (johnnyhuang1007@gmail.com)

Outline



Introduction

Problem Description

ILP Formulation

PA Setup

Sample Code / Submission

Sample Code

- For this assignment, we have prepared a sample code which contain:
  - Input / Output Parser
  - Result Plotter
  - Basic Data structure (floorplanner/solver/module/etc…)
- There are three parts of function you need to complete (marked as TODO):
  - bool Floorplanner:: solveCluster(Cluster*, float, float)
    - a function for you to complete ILP formulation and result write back
  - float Floorplanner:: category1Opt()
    - a function for you to develop your framework for Problem Category 1
  - void Cluster::rotate() //This function is optional but strongly recommend to complete
    - A function to rotate a super module (Cluster)

- You are free to adjust any part of the code as long as it produce a valid result

Class Cluster

- An inherent class of a module
  - It extends Module and represents a group of sub-modules
- Submodules member: std::vector<Module*> leaf
  - You can actually use the member to save an “cluster” member (an inherent class of module)
  - Accessor: std::vector<Module*>& getSubModules()
  - Constructor will initialize the leaf member: Cluster(std::vector<Module*>)
- To check if the item in leaf is module/cluster use:
  - bool isCluster(Module* m)
- The rotate() function of cluster is not written (leave as TODO)

bool Floorplanner:: solveCluster(Cluster*, float, float)

- Determine positions and rotations for the target cluster’s sub-modules so they fit within the W×H outline

- Main function for you to complete ILP framework:
  - Add VAR: solver_.addVariable(“X", 0.0, 2.0, GRB_INTEGER);
  - Add Constraint: solver_.addConstraint("c1", {{"Y", 1.0}, {"X", -1.0}}, '<', 100.0); //Y-X < 100
  - Add Objective: solver_.setObjective({{"X", 1.0}, {"Y", 1.0}}, 'M’); //min X+Y

- Also need to complete a write back feature to update positions/rotations of your modules

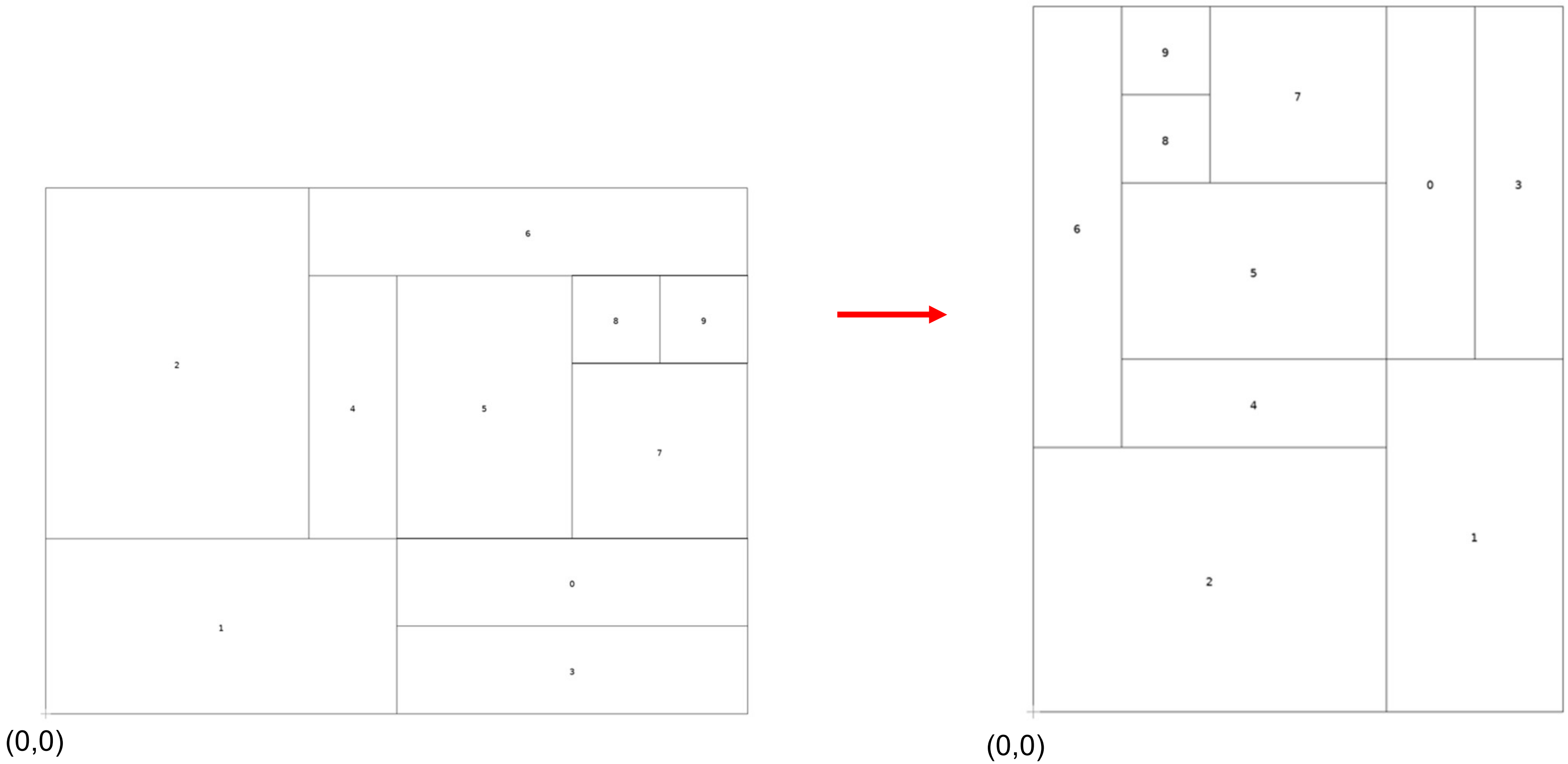
- We do not recommend removing the original solver-state update calls inside solveCluster()

void Cluster::rotate()

- Assume you have solved the sub-modules inside a Cluster and obtained the cluster’s width and height
  - When the cluster is rotated, its lower-left coordinate remains unchanged
  - width/height change accordingly
  - every sub-module update its position and rotation
- “rotated” is the cluster’s rotation flag
  - 0 → 1 rotates the internal modules 90° counterclockwise
  - 1 → 0 rotates them back to 0°
- Hint: Use collectAll() to gather all sub-modules and sub-clusters,
  - Compute each new coordinate independently to avoid recursive calls

void Cluster::rotate() - Example

Rotate 0 → 1

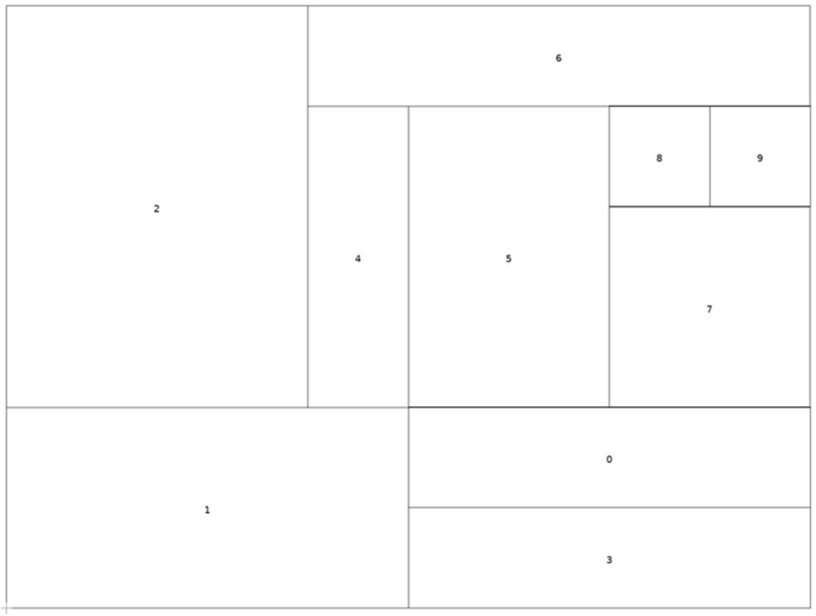


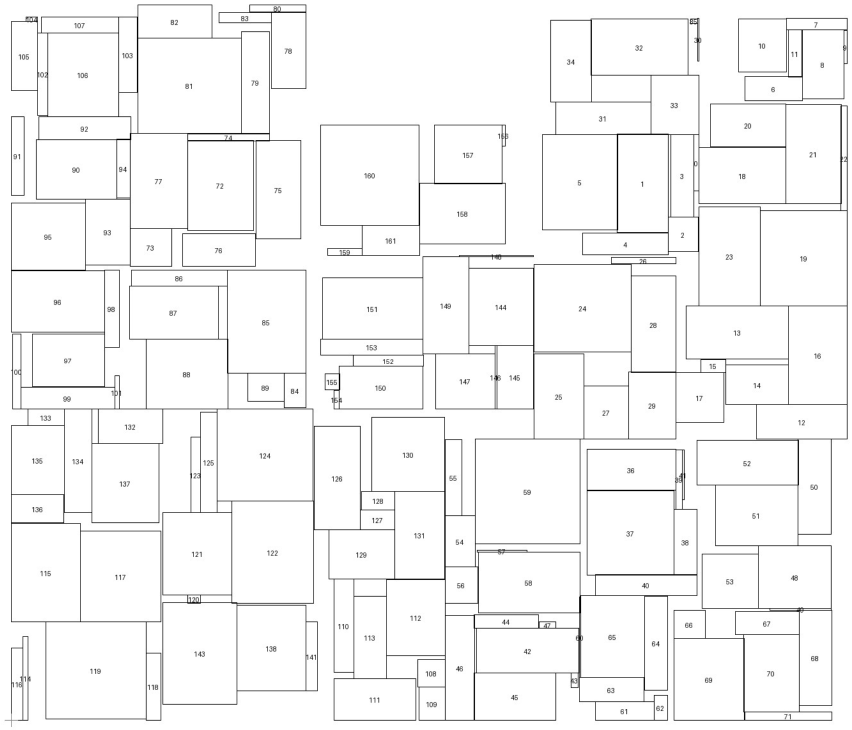
float Floorplanner:: category1Opt()

- We leave this function unimplemented
  - leverage solveCluster() to design an incremental/recursive flow.

Plotter

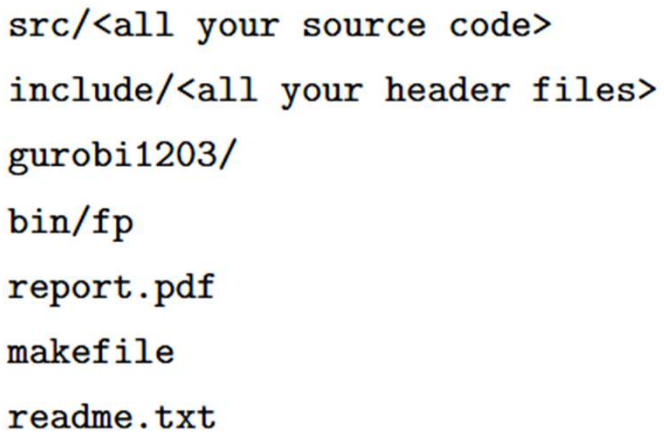
- Usage: python plot_floorplan.py ./input/sample.txt ./sample.out





Submission (I)

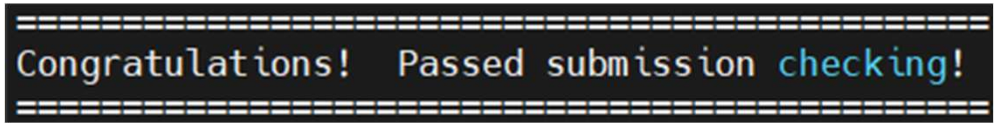
- Rename your working directory: mv PA2 [studentID]_pa2
- Include the following files inside [studentID]_pa2/
  - readme.txt
  - report.pdf
  - If you use Gurobi, include the package (gurobi1203)
- Do not include your gurobi license and input files
- you should have at least the following items in your directory



```
src/<all your source code>
include/<all your header files>
gurobi1203/
bin/fp
report.pdf
makefile
readme.txt
```

Submission (II)

- Compress your directory: tar zcvf [studentID]_pa2.tgz [studentID]_pa2
  - Do not add “./” prefix in the command
- Set execution permission for the checker script: chmod 755 ./checkSubmitPA2.sh
- Run submission checker: ./checkSubmitPA2.sh [studentID]_pa2.tgz
  - Do not add “./” prefix in the command



```
==============================================
Congratulations!  Passed submission checking!
==============================================
```

Submission (III)

- **No Plagiarism**
- Penalty for late submission: 20% per day
- Deadline: November 19, 2025, 23:59

- If you have any question, please contact:
  - Yu-Hsiang Huang (johnnyhuang1007@gmail.com)
  - Office Hours: Thu. 12:30~13:20 @ BL-407, by appointment

Thank you