# Using CP for Compound Dispensing between Microplates

**Ramiz Gindullin**[1], María Andreína Francisco Rodríguez[1], Brinton Seashore-Ludlow[2], Ola Spjuth[1,3]

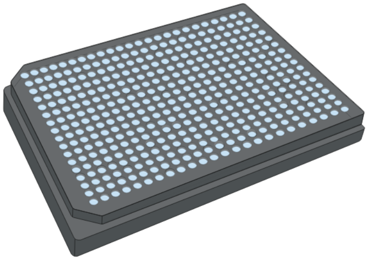
[1]Uppsala Universitetet, Uppsala, Sweden,
[2]Karolinska Institutet, Stockholm, Sweden,
[3]Phenaros Pharmaceuticals AB, Uppsala, Sweden

August 20, 2025

# Contents

# Contents

# Microplates



# Dispensing solutions

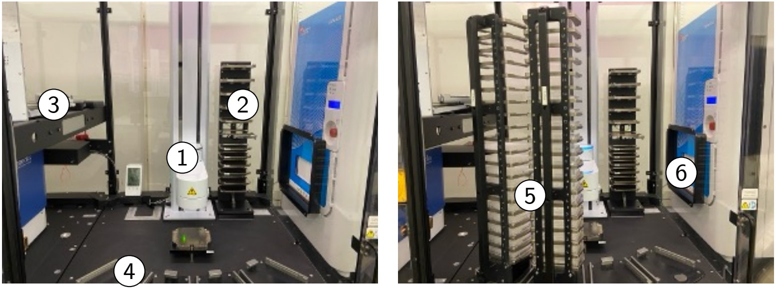
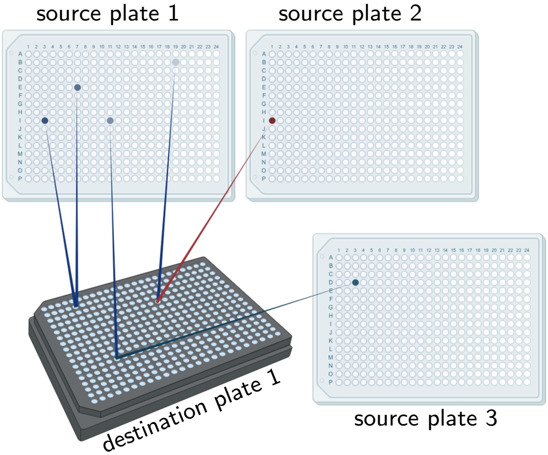
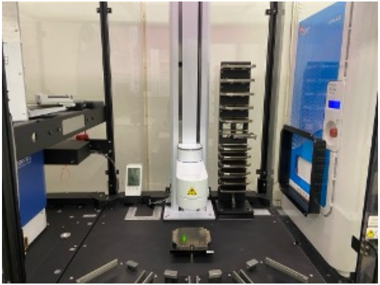


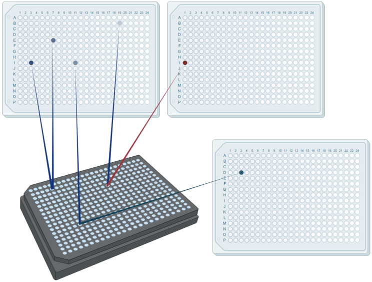

Figure: 1) the robotic arm, 2) the plate deliddler, 3) the sealer, 4) base for plate hotels, 5) plate hotels, 6) the dispenser.

# Testing drug combinations



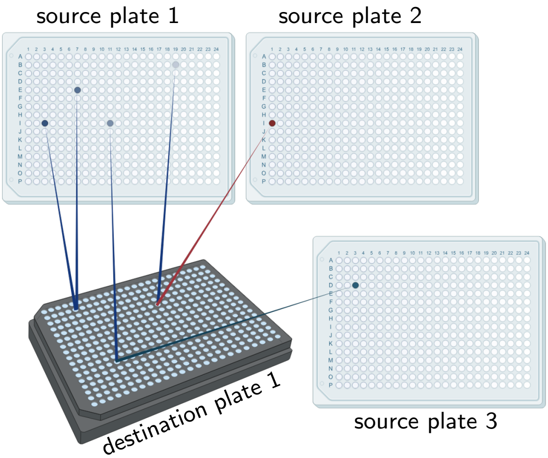
# Plate swaps





# Contents

# Full and simplified problems



# Input data

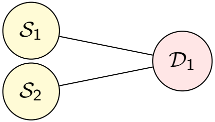
| Source plates | Destination plates, $i \in [1, c]$ | | | | | | |
|---|---|---|---|---|---|---|---|
| $j = 1$ | 1 | 1 | 1 | 1 | 1 | 1 | ... |
| $j = 2$ | 0 | 0 | 1 | 0 | 1 | 0 | ... |
| $j = 3$ | 0 | 1 | 0 | 0 | 0 | 0 | ... |
| $j = 4$ | 0 | 0 | 0 | 1 | 0 | 1 | ... |

Table: Matrix $\mathcal{C}^1$ to denote which source plates are required for any given experiment, where 0 denotes `false` and 1 denotes `true`.

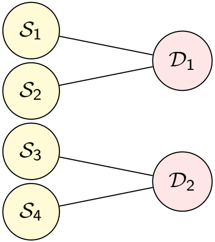
# Two steps process

- Distribute combinations b/w plates (SPP)
- Calculate the minimal possible number of swaps

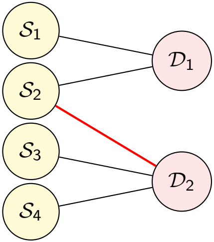
# Counting swaps: basics



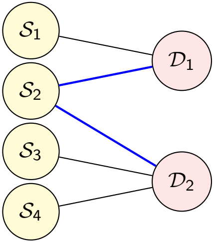
# Counting swaps: basics



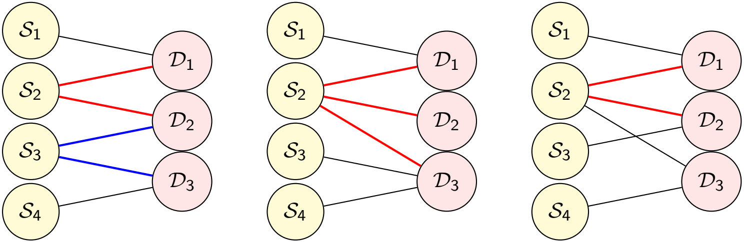
# Counting swaps: valid transitions - 1



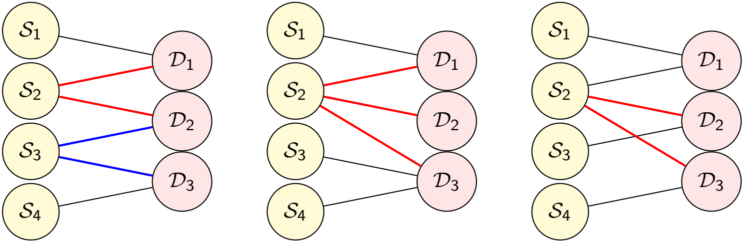
# Counting swaps: valid transitions - 1



# Counting swaps: valid transitions - 2



# Counting swaps: valid transitions - 2



# Contents

# Three groups of variables

- *Destination plates placement* variables, $P$
- *Bipartite network* variables, $\mathcal{A}$
- *Valid transitions* variables, $\mathcal{T}$

# Variable representations

- *Destination plates placement* variables, $P$:
  - Boolean matrix ($P^{I}$)
  - integer array ($P^{II}$)
  - array of sets of integers ($P^{III}$)
- *Bipartite network* variables, $\mathcal{A}$
- *Valid transitions* variables, $\mathcal{T}$

# Variable representations

- *Destination plates placement* variables, $P$:
  - Boolean matrix ($P^{I}$)
  - integer array ($P^{II}$)
  - array of sets of integers ($P^{III}$)
- *Bipartite network* variables, $\mathcal{A}$:
  - Boolean adjacency matrix ($\mathcal{A}^{A}$)
  - array of sets of integers (adjacency list) ($\mathcal{A}^{B}$)
- *Valid transitions* variables, $\mathcal{T}$

# Variable representations

- *Destination plates placement* variables, $P$:
  - Boolean matrix ($P^{I}$)
  - integer array ($P^{II}$)
  - array of sets of integers ($P^{III}$)
- *Bipartite network* variables, $\mathcal{A}$:
  - Boolean adjacency matrix ($\mathcal{A}^{A}$)
  - array of sets of integers (adjacency list) ($\mathcal{A}^{B}$)
- *Valid transitions* variables, $\mathcal{T}$:
  - array of integers

# Why consider various representations?

Representation III: $P_i^{III} = \{j \mid \exists j \in [1, c]\}$

# Why consider various representations?

Representation III: $P_i^{III} = \{j \mid \exists j \in [1, c]\}$:

$$\text{PARTITION\_SET}\left(\left\langle P_1^{III}, P_2^{III}, \ldots, P_d^{III}\right\rangle, \ \langle 1, \ 2, \ \ldots \ , \ c\rangle\right) \quad (1)$$

# Six models

| | $P^{I}$ | $P^{II}$ | $P^{III}$ |
|---|---|---|---|
| $\mathcal{A}^{A}$ | Model I-A | Model II-A | Model III-A |
| $\mathcal{A}^{B}$ | Model I-B | Model II-B | Model III-B |

# Contents

# Benchmark

Synthetic dataset:

- number of destination plates: $d \in \{2, 5, 10, 15\}$,
- number of wells in a single destination plate: $w \in \{2, 4, 6, 8\}$,
- number of source plates: $s \in \{2, 5, 10\}$,
- 10 examples for each combination,
- 480 total instances:
  - 180 small instances,
  - 180 medium instances,
  - 180 large instances

# Computational experiments

- Six models in MiniZinc,
- Solvers compared:
  - Chuffed 0.13.2,
  - CP-SAT 9.11.4210,
  - Gurobi 11.0.3,
- 2024 MacBook Air with 8 cores Apple M3,
- 300-second timeout,
- available at `https://github.com/astra-uu-se/multiplates`

# Results - Chuffed

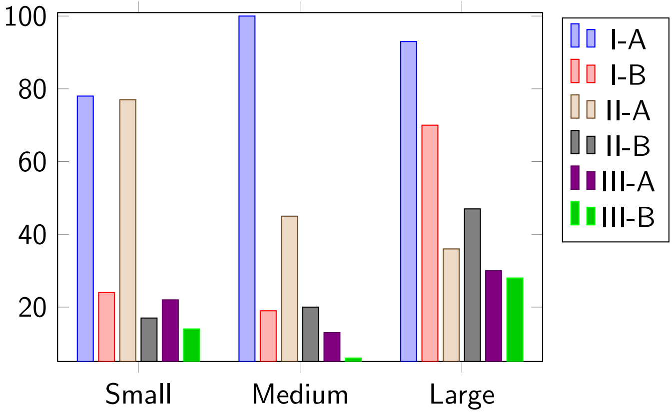



Figure: Number of instances when a model showed best performance

# Results - CP-SAT

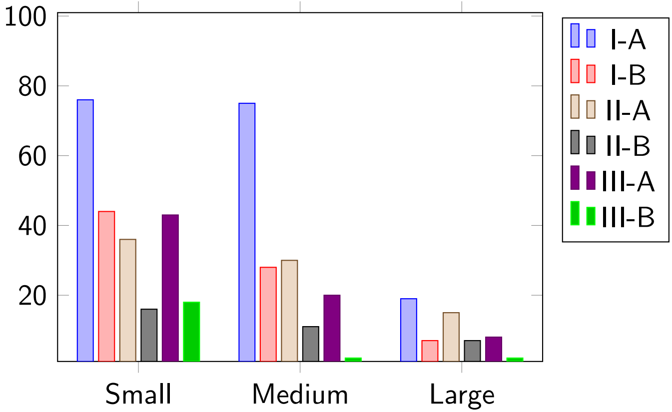



Figure: Number of instances when a model showed best performance

# Results - Gurobi

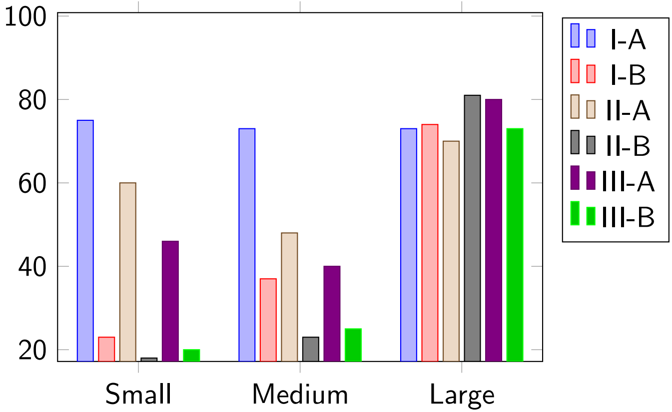



Figure: Number of instances when a model showed best performance

# Contents

# Conclusion

- OR methods and CP are useful for experiment planning,
- The problem is an SPP with a bipartite graph network,
- 6 CP models constructed and evaluated on different solvers