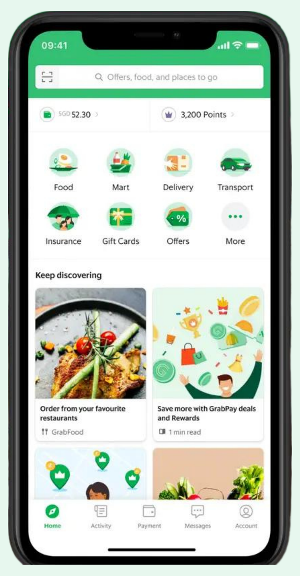
Cravings Can't Wait
Routing food deliveries under time uncertainty

By: Gabriel Afriat & Mariana Suarez
Faculty Advisors: Professor D. Freund & Professor S. Graves
Grab Advisors: Larry Lin & Yibo Ji

Overview

Problem Statement



- **Company:** Southeast Asia superapp
- **Scope:** GrabFood deliveries
- **Current State:** Batching algorithm assigns drivers to routes containing one or more orders based on deterministic time estimates
- **Limitation:** Does not take into account real-world travel time uncertainties

Goal

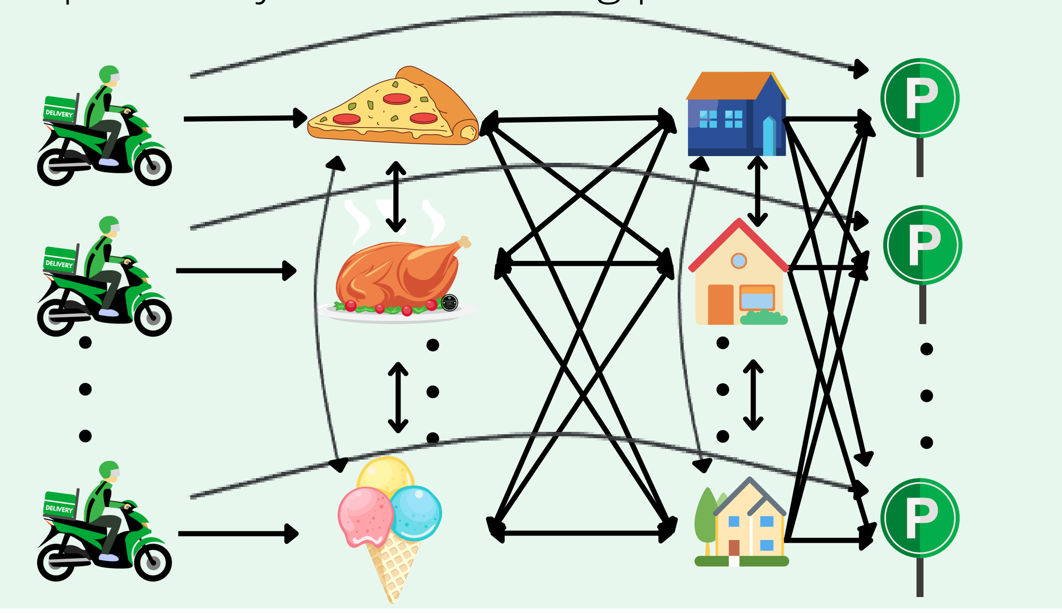
To add **robustness** (the consideration of uncertainty) to Grab's batching algorithm in order to **minimize delays**

Why It Matters?

- Key to increase **user satisfaction**
- Vital to understand **trade-off** between time **efficiency** and order **delays**

Problem Architecture

Pick-up delivery vehicle routing problem



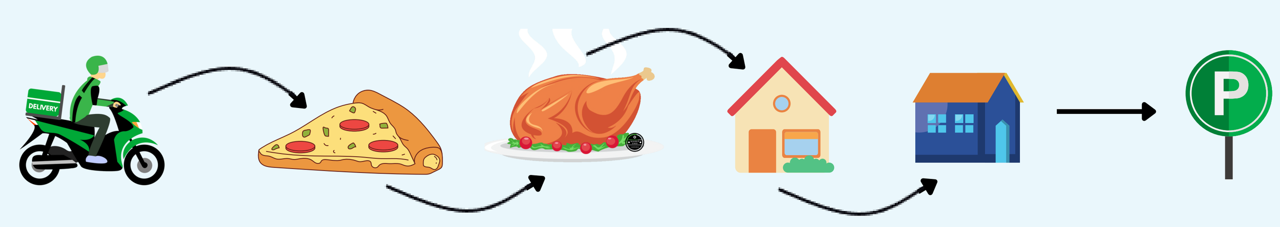
What Do We Need?

Data Input

- **Travel times:** 100 historical times per arc
- **Time windows**: lower and upper time bounds per pick-up and drop-off
- **Vehicle capacity**: number of orders a driver can carry

Model Output

The path that each driver should take



Our Approach

1. Clustering → 2. Graph Construction → 3. Optimization Models → 4. Warm Start → 5. Reduce Dual Gap → 6. Hyperparameter Tuning

Simplification

1. Clustering

Create a **distance metric** between orders to **group** nearby nodes and break down the problem into **smaller problems**

$d^{temp}_{o_1,o_2} =$ } Distance between *order 1* and *order 2*

$t_{P_1P_2} + t_{P_2D_1} +$ } Extent to which *order 2* is on *order 1*'s path

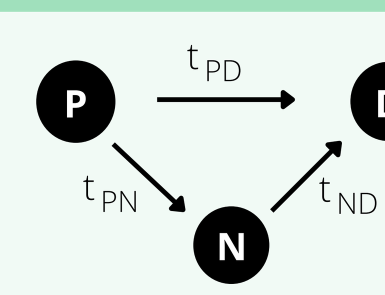
$\min(waiting_{P_1P_2}, waiting_{P_2,D_1}) +$ } Waiting time

$(+\infty)(1_{infeasible\_batching})$ } Infeasibility test

Hierarchical clustering performed on $d_{o_1,o_2} = \min(d^{temp}_{o_1,o_2}, d^{temp}_{o_2,o_1})$

2. Graph Construction

Create a **smaller graph** that includes only **"most likely" arcs** based on cost metric



$cost_{PN} =$ } Cost of the arc between pick-up *P* and node *N*

$t_{PN} + t_{ND} +$ } Extent to which node N is on the path between pick-up *P* and drop-off *D*

$waiting_{PN} +$ } Waiting time after going from *P* to *N*

$(+\infty)1_{infeasible\_arc}$ } Infeasibility test

Arcs (*P,N*) and (*N,D*) are only kept for the **best *k* neighbors** *N*

Similar metrics are defined for arc (*N,D*) and for arc (*D,N*)

Optimization

3. Optimization

Generate **solution path** for drivers by taking into account **time travel uncertainties**

Nominal

Travel times for each arc is assumed to be **known in advance**

$$\min_{\boldsymbol{x},\boldsymbol{S}} \sum_{k\in K} S_{3n+k,k}$$ } Sum of the time spent by each driver in the graph

such that

$$\sum_{k\in K}\sum_{\substack{j\in P\cup D\\(i,j,k)\in l_x}} x_{i,j,k} = 1 \quad \forall\, i\in P$$ } Ensures all pick up nodes are visited

$$\sum_{\substack{j\in V\\(i,j,k)\in l_x}} x_{i,j,k} - \sum_{\substack{j\in V\\(j,n+i,k)\in l_x}} x_{j,n+i,k} = 0 \quad \forall\, k\in K,\ \forall\, i\in P$$ } The driver who picks up the order must drop it off

$$\sum_{\substack{j\in P\cup\{3n+k\}\\(i,j,k)\in l_x}} x_{2n+k,j,k} = 1 \quad \forall\, k\in K$$

$$\sum_{\substack{i\in D\cup\{2n+k\}\\(i,j,k)\in l_x}} x_{i,3n+k,k} = 1 \quad \forall\, k\in K$$

$$\sum_{\substack{i\in V\\(i,j,k)\in l_x}} x_{i,j,k} - \sum_{\substack{i\in V\\(j,i,k)\in l_x}} x_{j,i,k} = 0 \quad \forall\, k\in K,\ \forall\, j\in N$$ } Flow constraints

$$S_{i,k} + t_{i,j} - M^{(S)}_{i,j}(1-x_{i,j,k}) \le S_{j,k} \quad \forall\,(i,j,k)\in l_x$$

$$a_i \le S_{i,k} \le b_i \quad \forall\, k\in K,\ \forall\, i\in V$$

$$S_{i,k} \le S_{n+i,k} \quad \forall\, k\in K,\ \forall\, i\in P$$ } Time window constraints

$$L_{i,k} + 1 - M^{(L)}(1-x_{i,j,k}) \le L_{j,k} \quad \forall\,(i,j,k)\in l_x \text{ with } j\in P$$

$$L_{i,k} - 1 - M^{(L)}(1-x_{i,j,k}) \le L_{j,k} \quad \forall\,(i,j,k)\in l_x \text{ with } j\in D$$

$$L_{i,k} \le C \quad \forall\, k\in K,\ \forall\, i\in V$$

$$L_{2n+k,k} = L_{3n+k,k} = 0 \quad \forall\, k\in K$$ } Capacity constraints

$$x_{i,j,k}\in\{0,1\} \quad \forall\,(i,j,k)\in l_x$$

$$S_{i,k}\ge 0 \quad \forall\, k\in K,\ \forall\, i\in V$$

$$L_{i,k}\ge 0 \quad \forall\, k\in K,\ \forall\, i\in V$$ } Binary and positivity constraints for variable definition

Robust

Time constraint **verified for any time in a given uncertainty set**

$$S_{i,k} + t^{(u)}_{i,j} - M^{(S,RC)}_{i,j}(1-x_{i,j,k}) \le S_{j,k}$$

$$\forall\, k\in K,\ \forall\,(i,j)\in A,\ \forall\, t^{(u)}_{i,j}\in\mathcal{T}_{i,j}(\gamma)$$

where hyperparameter γ measures the size of the uncertainty set and therefore the level of robustness

Distributionally Robust

Time constraints **verified in expectation** for **any probability distribution** in a given ambiguity set

$$\sum_{l\in I_{training}} p^{(l)}_{i,j}\left(S_{i,k} + t^{(l)}_{i,j} - M^{DRC}_{i,j}(1-x_{i,j,k}) - S_{j,k}\right) \le 0,$$

$$\forall\, k\in K,\ \forall\,(i,j)\in A,\ \forall (p_{i,j})_{l\in I_{training}} \in \mathcal{P}_{i,j}(\rho)$$

where hyperparameter ρ measures the size of the ambiguity set and therefore the level of robustness

4. Warm Start

Improve solution **quality** by providing the solver with an initial solution (here the **unbatched solution)**

5. Reducing the Dual Gap

Increase **confidence** in solution quality and facilitate the search for a better solution during **branch and bound** by experimenting with **5 formulations**

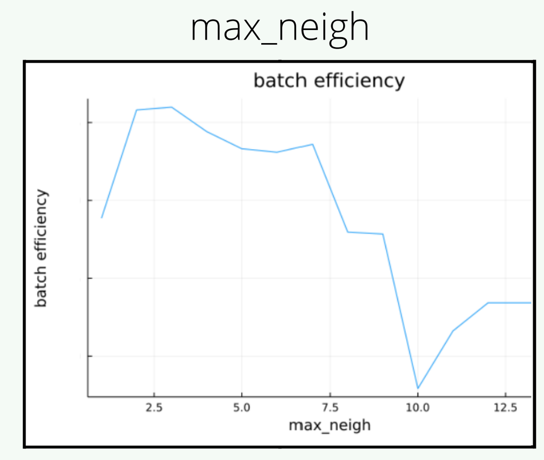
6. Fine-tuning

Identify **best combination** of **hyperparameters** for the **clustering**, **graph construction**, and **optimization** to improve performance across models

A direct fine-tuning would be **very long**.

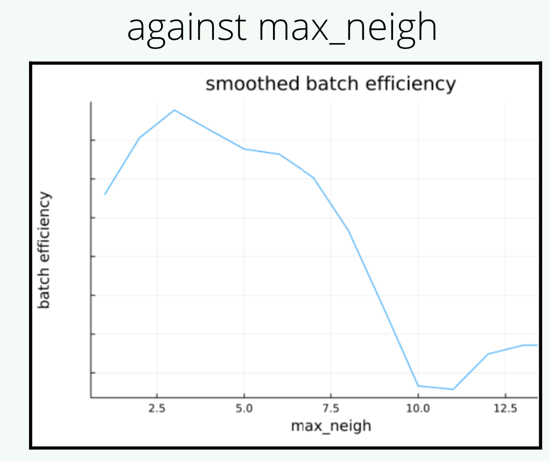
For the **graph construction** parameter "max_neigh", we note the **close to concave** nature of the batch efficency evolution with this parameter.

Batch efficiency against max_neigh



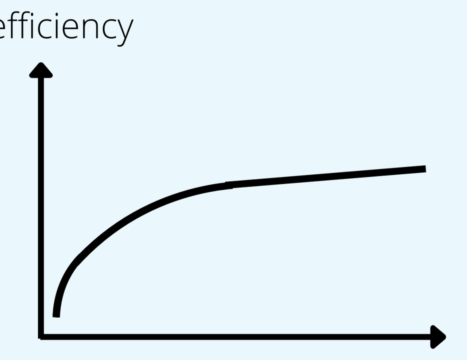
We **smooth** the curve to make it more concave and then we proceed to a **gradient ascent** via **dichotomy**.

Smoothed batch efficiency against max_neigh



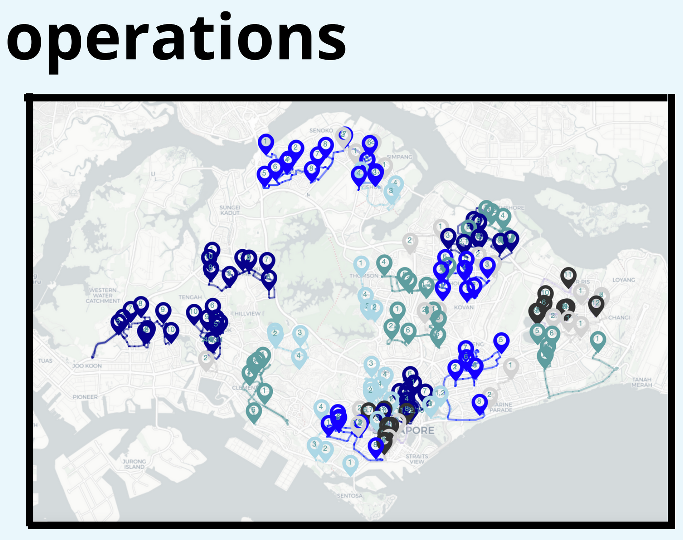
Results

1 Trade-off between time efficiency and order delays



2 Develop robust algorithms that improve Grab's operations

- ↓**16%** Maximum proportion of late orders
- ↓**70%** Maximum delay
- ↓**11%** Batch efficiency



Key Takeaway

By including real-world uncertainty of driver routing times in delivery algorithms, we are able to reduce the number of delayed orders and improve food delivery operations.