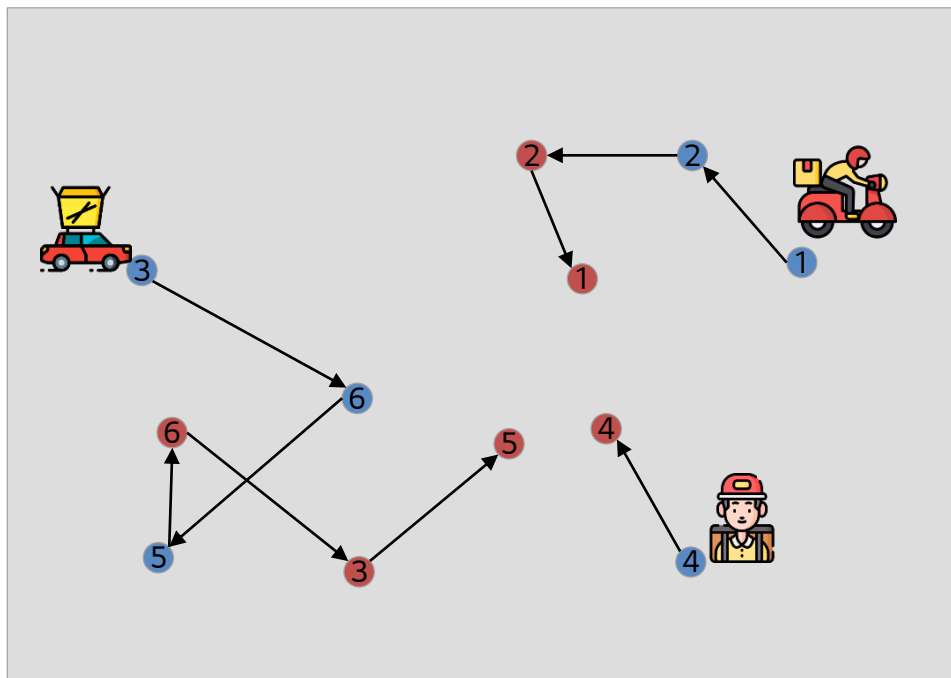


OGC 발표

팀명: 문선균
팀원: 문성수 / 신선종 / 최원균
소속: LG CNS
조직: 스마트물류 & 시티 사업부

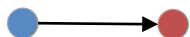
문제 설명

주문 묶음 배송 문제



● : 픽업 위치 ● : 배달 위치

주문 정보



- 주문 시각
- 주문 준비 시간
- 주문 용량
- 배달한계시간

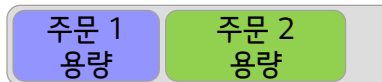
배달원 정보



- 용량
- 이동속도
- 접근시간
- 비용

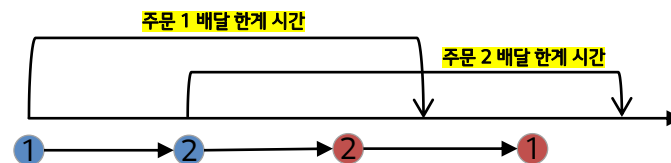
제약조건

• 용량 제약



- 묶음 배송 주문 용량의 합 ≤ 배달원 용량

• 시간 제약 및 방문 순서 제약



- 각 주문의 배달 한계 시간 준수
- 묶음 내 모든 픽업이 수행된 후 배달 시작

• 주문 만족 제약 및 배달원 할당 제약



- 모든 주문 빠짐없이 배달
- 배달원과 묶음 배송 1:1 할당

목표

- 주문 묶는 방법 결정
- 묶음의 배송 순서 결정
- 묶음에 할당할 배달원을 결정

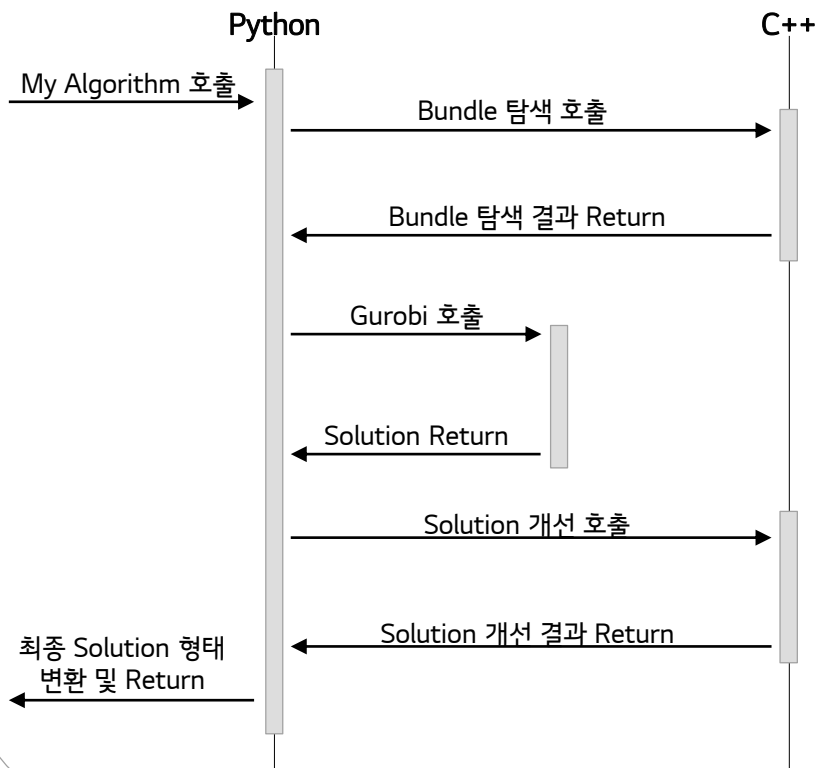


평균 배송비용 최소화

알고리즘 구현 Overview

- Python과 C++을 이용하여 알고리즘을 구현함.

Python과 C++의 호출 관계



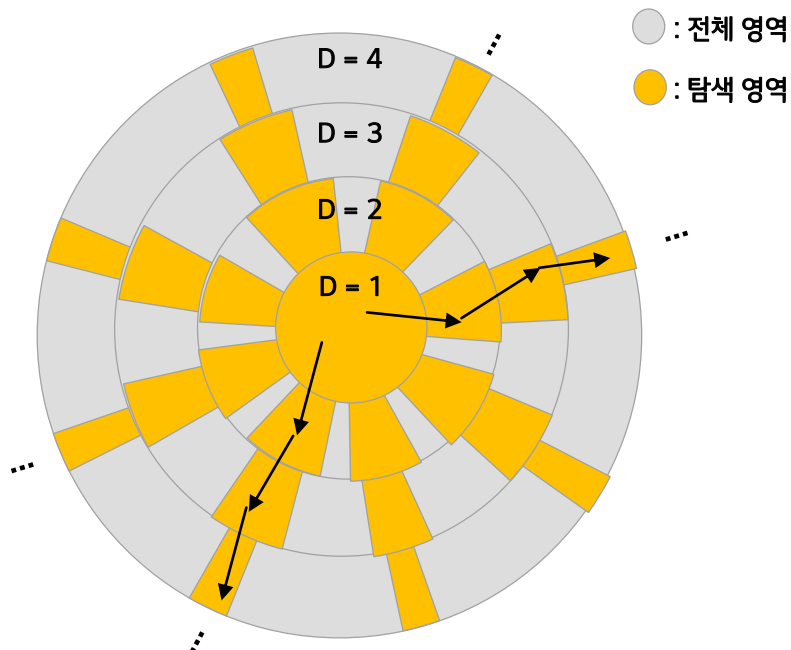
구현 특징

- Bundle 탐색 시간 및 전체적인 알고리즘 속도 향상을 위해 C++ 사용
- Python에서 C++ 호출 및 C++ 에서 Python Class를 동일하게 사용하기 위해 Pybind11 사용
- 다양한 형 Type이 입력될 수 있도록 C++ Variant를 이용하여 Heap 구현

알고리즘 로직 Overview

Bundle 탐색

- BFS + 탐색 순서 조정 + pruning



Bundle 조합 구성

- 평균 cost 낮은 상위 N개 bundle 선정
- 수리 모형 통한 최적해 도출

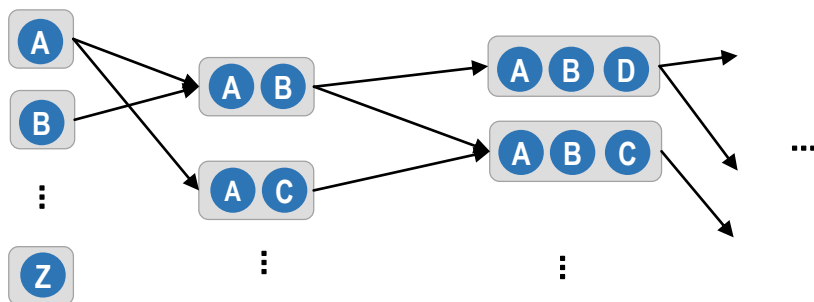
수리모형

- 목적식: bundle 비용 합계 최소화
- 제약식
 - 각 주문은 정확히 1개의 bundle에 포함됨
 - 배달 수단의 상한선 내에서 bundle 선정

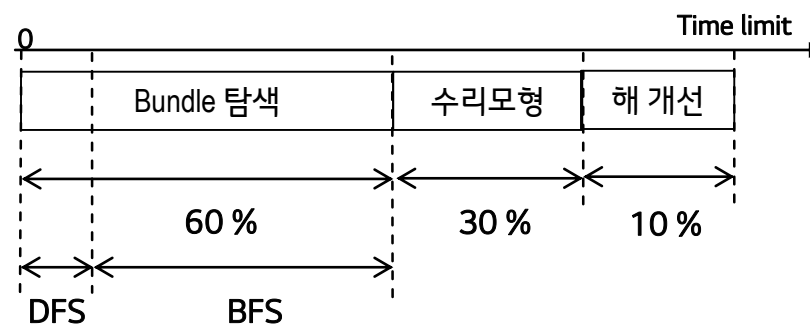
Bundle 해 개선

- 수리모형에서 선정된 Bundle 대상 완전 탐색
 - 음식점/고객 방문 순서 확정

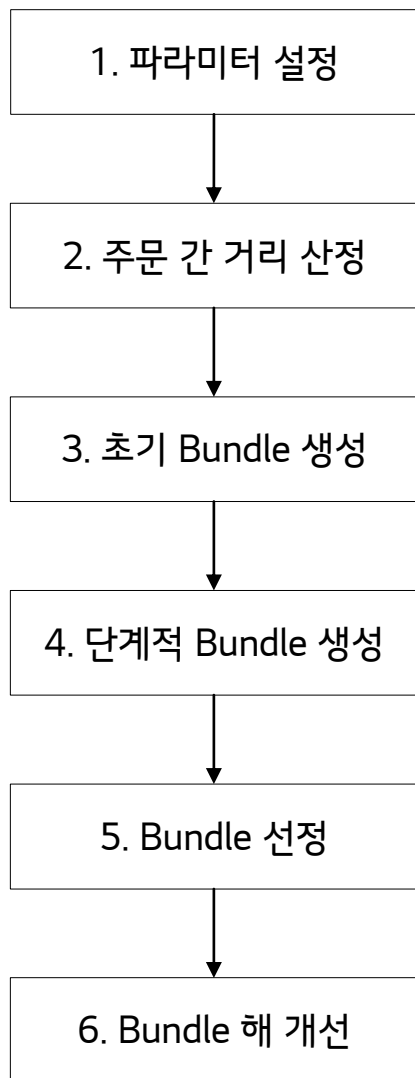
D = 1 → D = 2 → D = 3



Time limit에 따른 시간 배분



주요 알고리즘 흐름도



- K, time limit 기반 파라미터 설정
- 탐색 시간 설정 (초기 Bundle, 단계적 Bundle, Gurobi 탐색 시간 설정)
- Pruning 설정 (Bundle 최대 분기 수, depth 별 Bundle 수 상한, 생성 실패 최대 횟수 등)

- Pair 단위 거리 산정
- '준비 시간' / '수령 시간' 고려한 거리 보정
- Triple 이상의 Bundle 생성 시에도 Pair 단위 거리 활용함

- DFS 기반으로 각 주문에서 가까운 순으로 greedy하게 순차적으로 Bundle 병합
- 일정 횟수 이상 병합 실패 시 중지

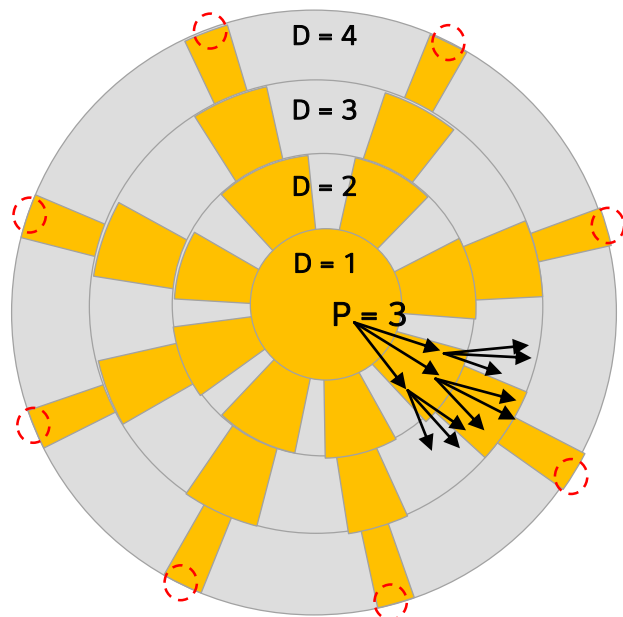
- BFS 기반 단계별 탐색 (Depth에 따라 단계 구분)
- 각 단계 완료 시점 아래 항목을 수행함
 - 공간적 여유가 있는 상위 M개 Bundle 선정
 - 탐색 상태 기반 pruning 관련 동적 파라미터 조정

- 평균 Cost가 작은 상위 N개 Bundle + 단일 Bundle 선정
- Gurobi 호출 및 Bundle 최종 선정

- 선정된 Bundle에 대해 음식점/고객 방문 순서 기준 Permutation 완전 탐색

1. 파라미터 설정

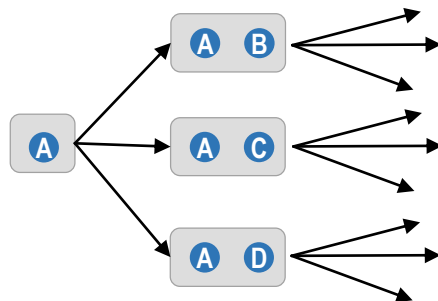
- T (timelimit), K (주문 수) 값에 따라 유동적으로 탐색 영역을 제한하기 위한 파라미터 설정



○: 타겟 영역

- 타겟 영역: 더 좋은 Bundle을 생성하기 위한 탐색 영역

- P (분기 수)



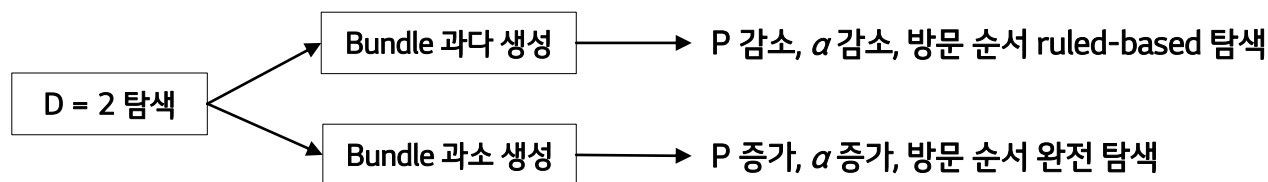
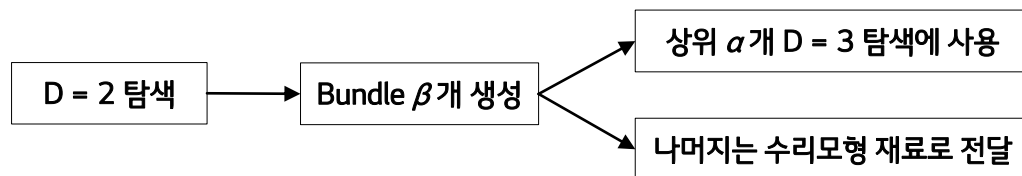
M = 큰 수, 실험을 통해 산정

$$p = \left\lceil \frac{M \cdot T}{K^2 \cdot \alpha_\tau} \right\rceil$$

α의 타겟 value

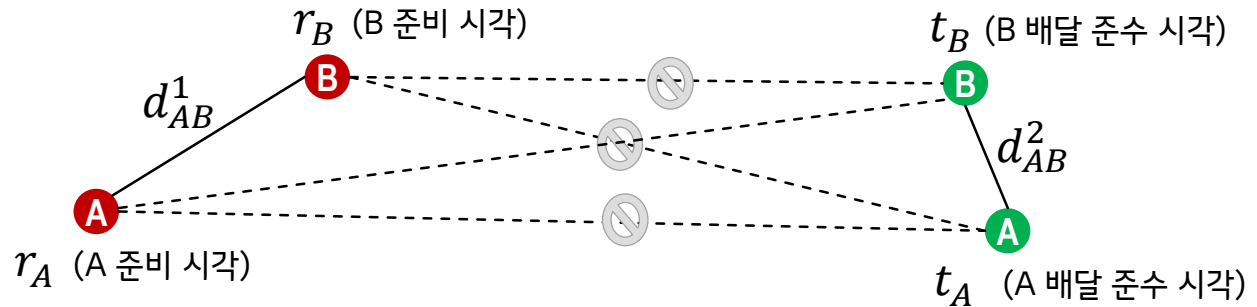
- α (Depth 별 Bundle 수 상한)

$$\alpha = \text{round}\left(\frac{M \cdot T}{K^2 \cdot p}\right)$$



2. 주문 간 거리 산정

주문 - 주문 간 거리



$$Dist_{AB} = d_{AB}^1 + d_{AB}^2 + w_1 \cdot |r_A - r_B| + w_2 \cdot |t_A - t_B|$$

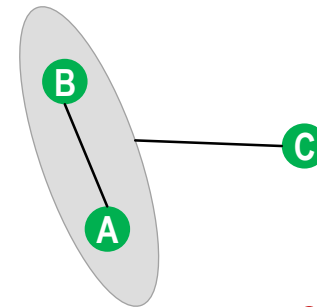
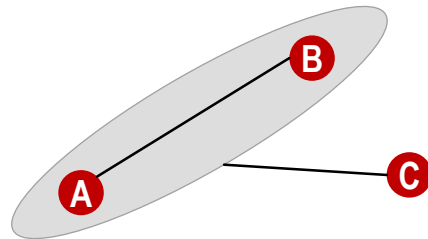
픽업 위치 간 거리

배달 위치 간 거리

준비 시각 차이에 따른 가중치

배달 준수 시각 차이에 따른 가중치

Bundle - 주문 간 거리



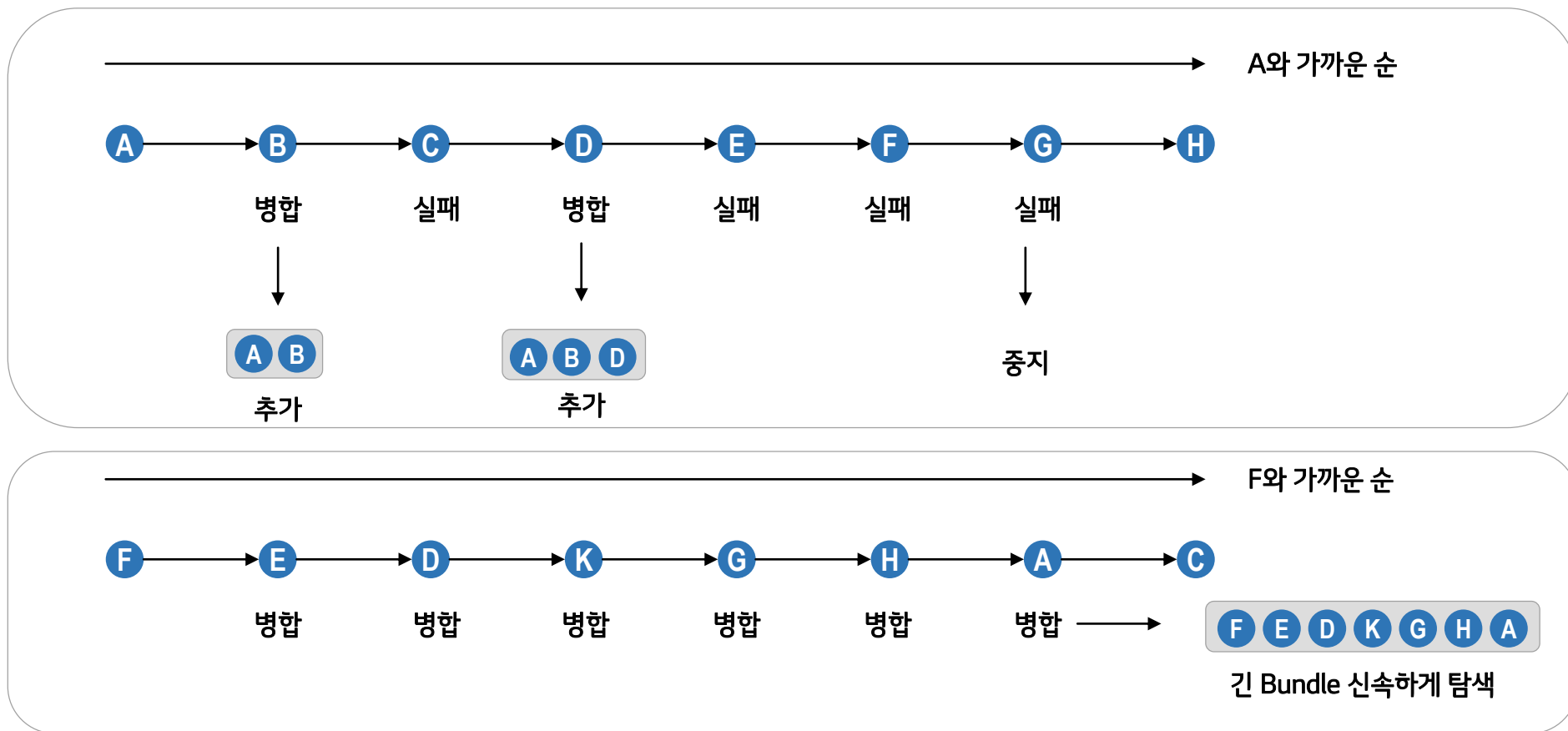
● : 픽업 위치

● : 배달 위치

$$Dist_{XC} = \min_{Y \in X} Dist_{YC} \quad X = \{A, B\}$$

3. 초기 Bundle 생성

- BFS가 긴 Bundle 생성에 불리한 점을 보완하기 위해 초기 Bundle은 DFS + Greedy 기반으로 탐색 공간을 줄여 빠르게 탐색함



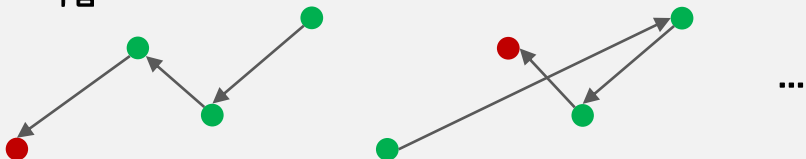
- 각 주문별 / Rider별 Bundle 생성 시도
- 선택된 주문에 대해 가까운 순으로 greedy 하게 순차적 병합 시도 & 일정 횟수 이상 병합 실패 시 중지
- 주문과 가까운 순 정렬 기준은 주문 간 거리 생성 결과 값을 기준으로 함

4. 단계적 Bundle 생성 (1/2)

방문 순서 제한

	픽업		배달		총 가지 수
완전 탐색	$N!$	\times	$N!$	$=$	$(N!)^2$
Rule-based	N	\times	2	$=$	$2N$

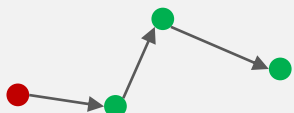
픽업



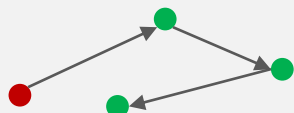
- 시작 위치 각 주문으로 설정 → 총 N가지 경로
- 마지막 주문에 가까운 순으로 Greedy하게 순차적 연결
- 탐색 주문 역순 정렬 → 마지막 픽업 위치의 다양성 확보

배달

1. 가까운 순



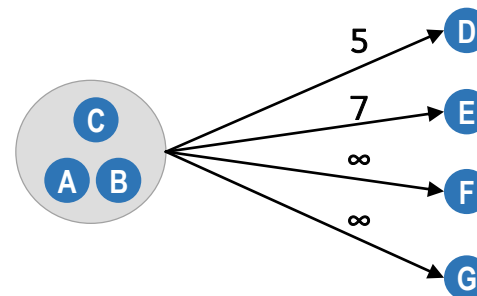
2. 배달 준수 시각 빠른 순



● : 마지막 픽업 위치

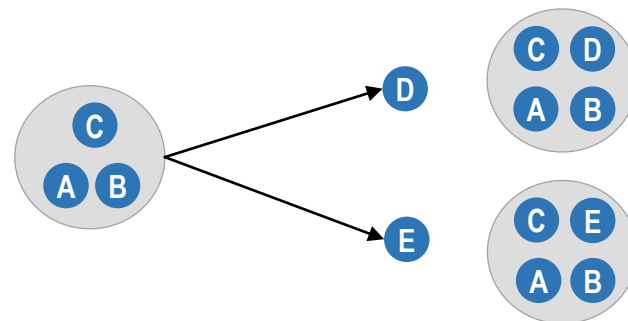
Bundle 최대 분기 수 / 실패 횟수 제약

1. Bundle과 주문 간 거리 산정



- Bundle 내 각 주문과 가까운 M개 주문 한정 탐색
- 소요 시간: $O(K \log K) \rightarrow O(DM + M \log M)$

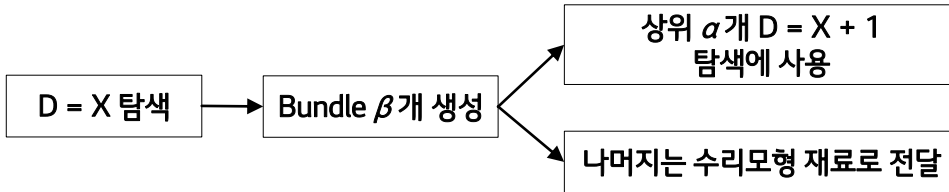
2. Bundle과 거리 가까운 순 병합 시도 및 중지



- 다음 2가지 조건으로 특정 Bundle 병합 종료
- A. Bundle 최대 분기 수 도달
- B. 연속 병합 실패 횟수 상한 도달

4. 단계적 Bundle 생성 (2/2)

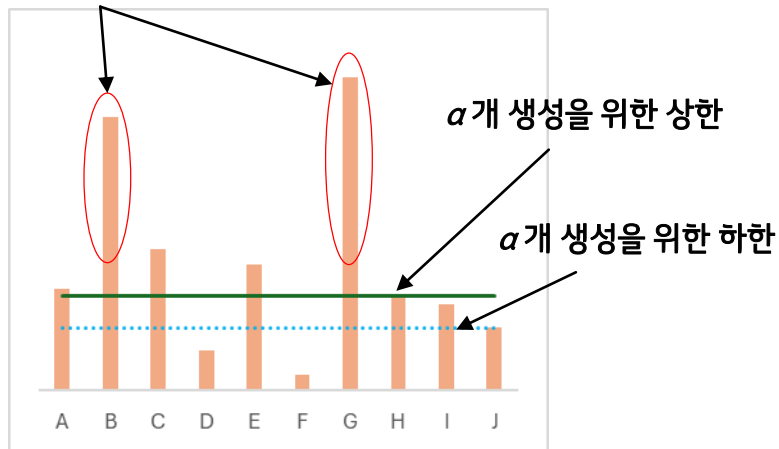
Depth 별 Bundle 수 상한



상위 α 개 선정 방식

- Rider의 잔여 Capa가 큰 순으로 선정
- 오더별 선정 횟수 평활화를 위한 상한선 설정

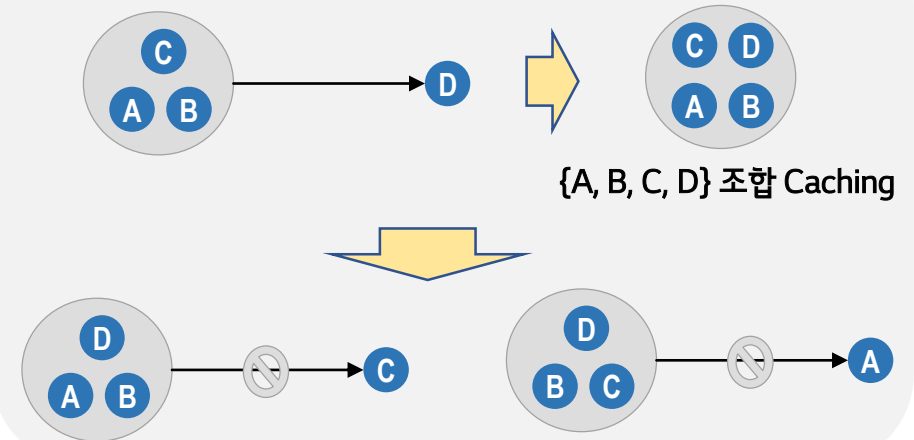
과다 포함 시 탐색 다양성 저하



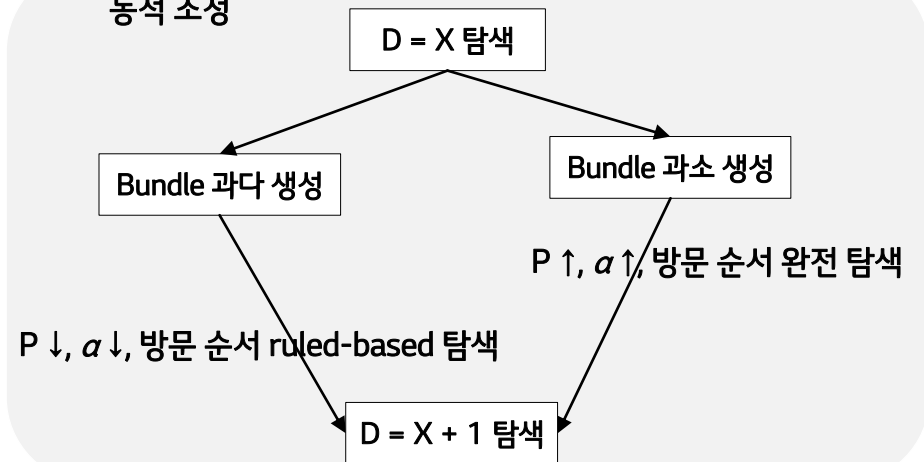
생성된 Bundle β 개 내에서 각 주문을 포함하는 Bundle 개수 분포

Caching / 동적 파라미터 조정

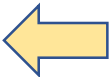
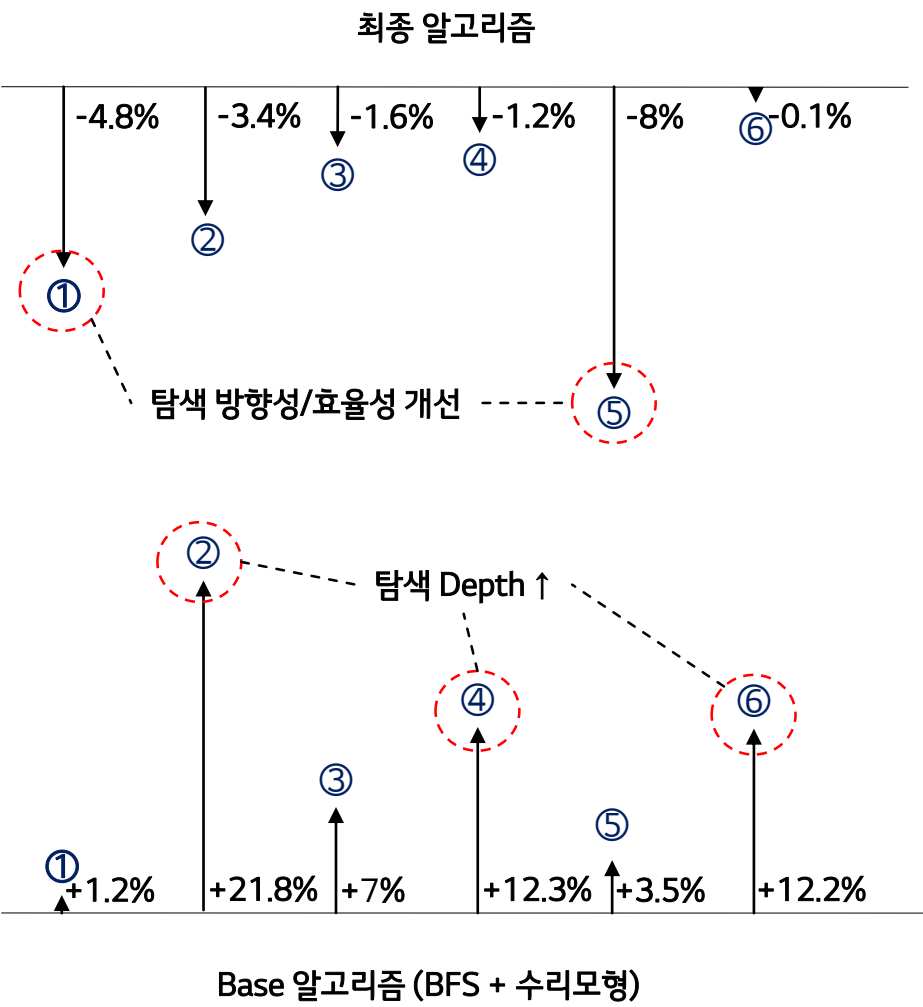
Caching



동적 조정



주요 알고리즘 성능 비교




최종 알고리즘에서 각 항목 제거 시 성능 비교

항목	설명
①	주문 pair 간 산정한 거리 가까운 순 우선 탐색
②	Bundle 최대 분기 수 적용
③	Bundle 분기 시 실패 횟수 상한 설정
④	초기 Bundle 생성
⑤	방문 순서 제한
⑥	Caching / 동적 파라미터 조정

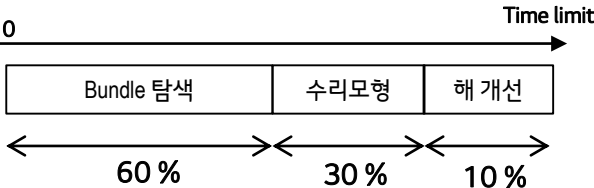
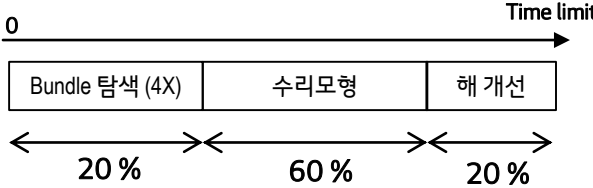


Base 알고리즘에서 각 항목 추가 시 성능 비교

추후 개선 방향 (1/2)

항목	설명
Pair Bundle 생성	<ul style="list-style-type: none"> 기존: 동일 Depth에서 탐색 순서는 아래 순서를 따름 <ul style="list-style-type: none"> - Pair: 주문의 index 순 - Triple 이상: Bundle의 잔여 Capa가 큰 순 → Pair Bundle 생성 시 index에 따른 편향성이 발생함 개선: Pair Bundle 탐색 시 특정 index를 포함하는 Bundle의 상한선 설정 <ul style="list-style-type: none"> → index에 따른 편향성 제거 및 bundle 개수 평활화 → 개선 결과 확인하였으나, 소스 코드 적용 시 누락되었음
Depth별 Bundle 수 상한	<ul style="list-style-type: none"> 기존: 동일 Depth 내 상위 α개 Bundle 산정 시 Rider의 잔여 Capa가 큰 순으로 선정 <ul style="list-style-type: none"> → 잔여 Capa가 큰 Car 위주의 Bundle 선정되며, Bike, Walk에 대한 충분한 Bundle 생성 실패 개선 시도: rider type에 따른 별도의 상한선 설정 필요
Bin packing	<ul style="list-style-type: none"> 기존: 거리가 가까운 순으로 Bundle 생성 시도 → 일정 수준의 거리 내에서 잔여 Capa 줄이는 Bundle 생성 시도 

추후 개선 방향 (2/2)

항목	설명
병렬 프로그래밍	<ul style="list-style-type: none"> Divide and conquer <div>   </div> Portfolio <ul style="list-style-type: none"> - 서로 다른 전략의 4개 알고리즘 병렬 수행 - Shared information 고민 필요
배달 방문 순서	<ul style="list-style-type: none"> 기존: 가까운 순 & 배달 준수 시간 빠른 순 2가지로 적용 개선 시도: N개의 주문에 대해 K개는 시간 빠른 순, (N - K)개는 가까운 순으로 혼합하여 적용
새로운 방식 접근	<ul style="list-style-type: none"> 패턴 기반 Bundle 대량 생성 및 feasibility check Maximum bipartite matching 기반 Bundle 생성 (Pair - Single, Triple - Single)

항목	설명
소회	<p>팀 프로젝트 이행 경험은 많이 있지만 상세설계 없이 이행한 경험은 입사 이후에는 없었던 것 같습니다. 큰 관점에서 컨셉만 잡은 후 개발에 착수하고 설계서 없이 산발적으로 방향성을 결정하여 진행했습니다.</p> <p>결과적으로 소스에 반영되지 못하고 누락된 부분이나, 발표 준비를 하며 떠오른 아이디어가 있어 아쉬웠습니다. 프로젝트 이행처럼 설계서 기반의 개발을 하면 더 좋았을 수도 있으나, 업무와는 별개로 편한 마음으로 진행하려다 보니 이런 아쉬움들이 남게 된 것 같습니다.</p> <p>결선 진출하여 다른 팀의 아이디어를 엿듣고 네트워킹의 기회를 가지는 것을 목표로 진행하여 소기의 목적을 이루었기 때문에 좋은 경험으로 남을 것 같습니다.</p> <p>열정의 촛불이 몇 개 남지 않았는데 다시금 불태워볼 수 있는 기회를 주셔서 감사하며 2, 3회 대회는 더 번창하여 지속적인 대회가 열릴 수 있기를 기원합니다.</p>
발전 사항	<p>솔버를 활용하는 만큼 타임 아웃 제어에 어려움이 있었습니다. 타임 아웃을 2단계로 적용하면 어떨까 생각해 봤습니다.</p> <ul style="list-style-type: none"> - 1단계 초과 시 penalty 부여 - 2단계 초과 시 강제 종료 및 timelimit 표시 <p>ex) 1단계: 60초, 2단계: 120초 $(60 + a)\text{초} \rightarrow \text{score} + \text{penalty} * a$</p> <p>예선전에서 마지막 문제의 최적해가 도출되지 않아 어려움이 있었습니다. (향후 개선함) 각 문제에 대해 가장 좋은 해를 도출한 팀의 값을 기준으로 상대적 점수를 부여받으면 어떨까 싶습니다. 여기에 각 문제별 1등팀은 pareto optimal 달성이므로 추가 benefit 부여도 좋을 것 같습니다.</p> <ul style="list-style-type: none"> - 공동 1등 발생 시 pareto optimal 보장되지 않으며 benefit 부여하지 않음