최적화 그랜드 챌린지 2025

Roll-on Roll-off 선박의 선적 계획 수립을 위한 휴리스틱 알고리듬

Team 12HUB2
나용수 이세영 구상모

September 19, 2025

Overview

1. 선적 알고리듬

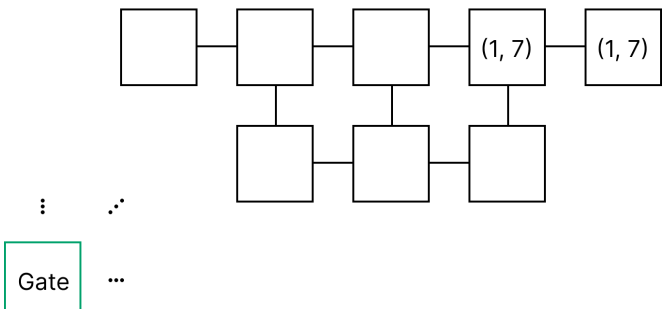
2. 하역 알고리듬

3. 무작위성을 활용한 개선 전략

4. 알고리듬 특장점 및 개선 방향

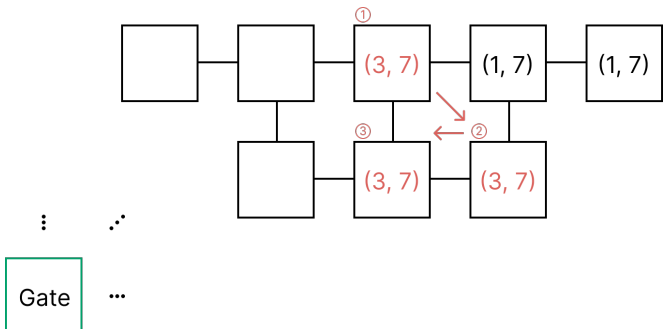
선적 알고리듬 아이디어

- 화물을 **(선적 항구, 하역 항구)**로 표기
- 항구 3에서 목적지 항구가 7인 화물을 적재하는 상황을 가정



선적 알고리듬 아이디어

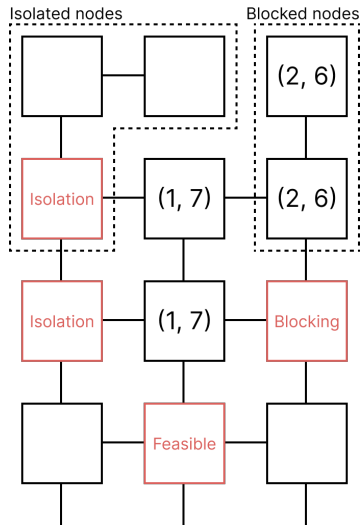
- 패턴 1: **동일 목적지 화물들은 인접하게 배치**하여 군집을 이루도록 함
- 패턴 2: 다른 화물의 배치를 염두에 두어 **최대한 "구석"에 배치**하도록 함



군집 확장 배치 휴리스틱

- 동일 목적지 화물에 인접하도록 배치
- 발생하는 상황을 시뮬레이션해 가능한 상황에만 배치를 진행

1. **Feasible**: 바로 배치
2. **Isolation**: 배치해야 하는 화물의 수 고립된 노드 수보다 많으면 해당 노드들을 포함 모두 배치
3. **Blocking**: Block된 모든 화물들의 목적지가 현재 배치 중인 화물의 목적지와 같거나 더 먼 경우 배치



군집 확장 배치 휴리스틱

군집 확장 배치

1. $d-$*화물* 리스트가 입력된다.
2. Deck에서 $d-$*화물*이 배치되어 있는 노드 집합 A를 찾는다.
3. A 중 하나 이상의 노드와 인접하면서 비어 있는 노드 집합 B를 찾는다.
4. B 중 centrality가 가장 낮은 노드 C를 고르고 B에서 제거한다(tie breaking: |shortest path| $\rightarrow$ index). $|B| = 0$인 경우 종료한다.
5. C에서의 배치 시뮬레이션을 수행하고 결과에 따라 배치를 진행하거나 하지 않는다.:
   (a) Feasible한 경우 즉시 배치한다.
   (b) Isolation이 발생하면서 배치 가능한 경우 즉시 배치한다.
   (c) Blocking이 발생하면서 배치 가능한 경우 즉시 배치한다.
6. 배치해야 하는 화물이 남아 있는 경우 절차 **4.**로 돌아가고, 없는 경우 종료한다.

*목적지 항구가 d인 화물을 $d-$*화물*이라고 표기

군집 시작점

동일 목적지 화물이 존재하지 않는 경우, 적절한 군집 시작점이 필요

- Mode **M**: BFS dead-ends의 일부 중 무작위로 선택된 노드
- Mode **N**: 거리 3 이내에 비어 있는 노드 수가 가장 많은 노드
- Mode **B**: 게이트로부터 가장 멀리 있으면서, 시뮬레이션을 통해 배치 가능하다고 판단되는 노드

각 모드를 순차적으로 실행하여 화물을 모두 배치할 때까지 시도

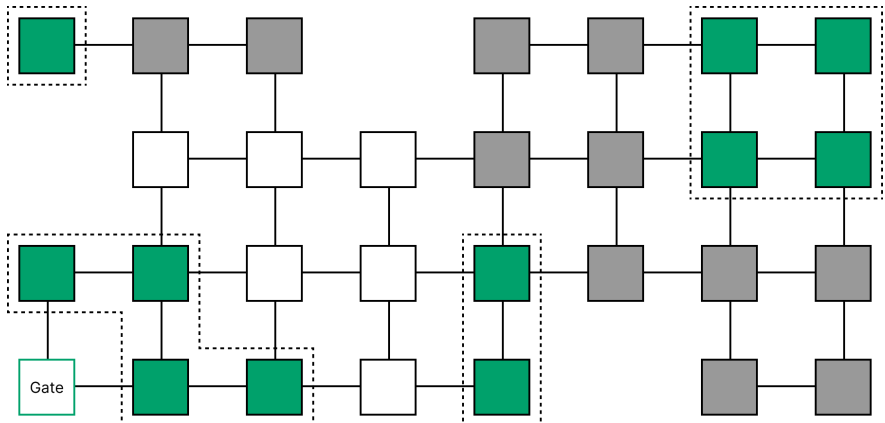
선적 알고리듬

항구 i에서의 선적

1. 배치해야 하는 $(i+1)-$*화물*이 $m > 0$개 존재하는 경우, 게이트로부터 가장 가까운 m개의 노드를 선점해 배치한다.
2. $p \leftarrow P$로 둔다.
3. $p \leftarrow p-1$로 두고 p-*화물* 리스트를 구성한다.
4. 화물 리스트의 길이가 0인 경우, $p > i$이면 절차 **3.**로 돌아가고, $p = i$이면 종료한다.
5. 아래 **(a)**-**(d)** 절차를 순서대로 수행한다. 수행할 때마다 화물 리스트를 업데이트하며, 도중에 $p-$*화물*을 모두 배치한 경우 절차 **3.**으로 돌아간다.
   - **(a)** $p-$*화물*이 deck에 존재하는 경우 *군집 확장 배치*를 수행한다.
   - **(b)** **M** 모드로 화물 배치를 시도한다. 배치에 성공한 경우 *군집 확장 배치*를 수행한다.
   - **(c)** **N** 모드로 화물 배치를 시도한다. 배치에 성공한 경우 *군집 확장 배치*를 수행한다.
   - **(d)** **B** 모드로 화물 배치를 시도한다. 배치에 성공한 경우 *군집 확장 배치*를 수행한다.
6. 배치하지 못한 화물이 남아 있다면 *나머지 화물 리스트*에 추가한다.
7. $p > i$이면 절차 **3.**으로 돌아간다.
8. *나머지 화물 리스트*의 화물들을 베이스라인으로 선적한다.

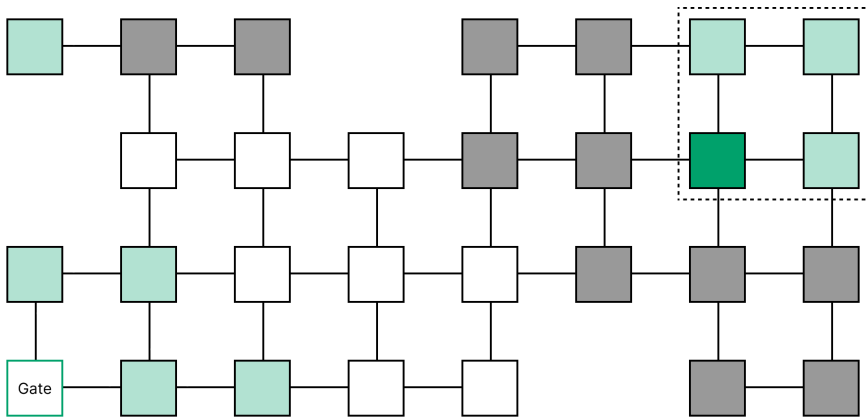
하역 알고리듬

- 하역 대상 화물 중에서 **서로 인접한 노드들끼리의 군집**을 식별
- BFS + Union-Find



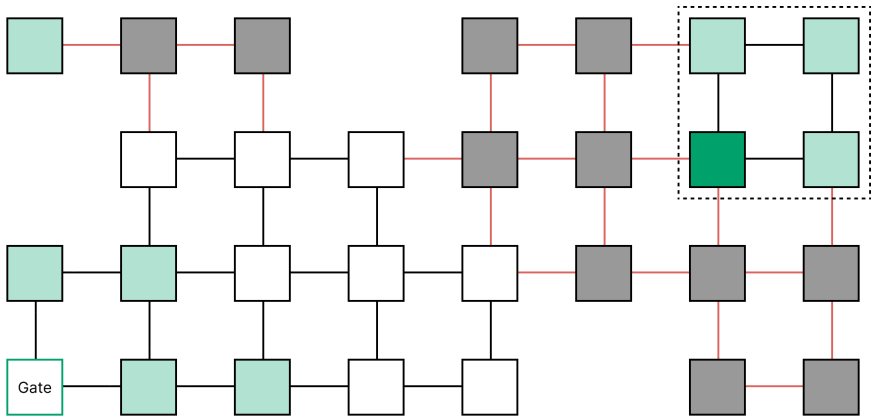
하역 알고리듬

- 하나의 군집을 선택해, 그중 게이트와 가장 가까운 **하나의 노드만을 고려**



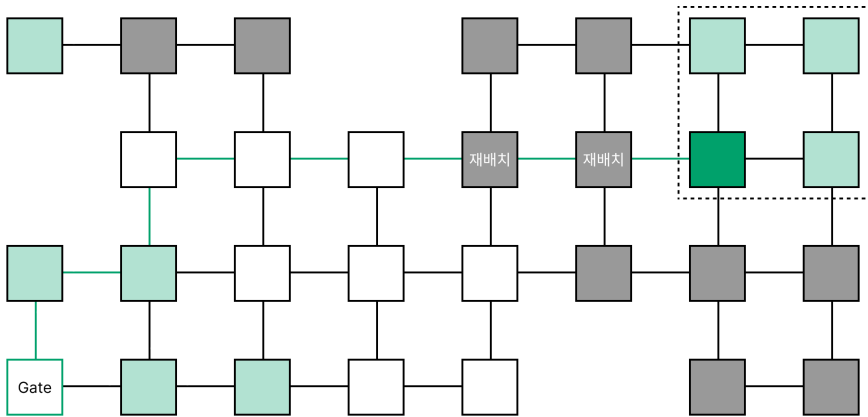
하역 알고리듬

- N =전체 노드 수, F =고정 비용, G =군집 크기
- 검정색 엣지 비용 $= G$ & 붉은색 엣지 비용 $= 2F + 0.1N + G$
- Dijkstra's algorithm을 통해 최소 재배치 경로 계산



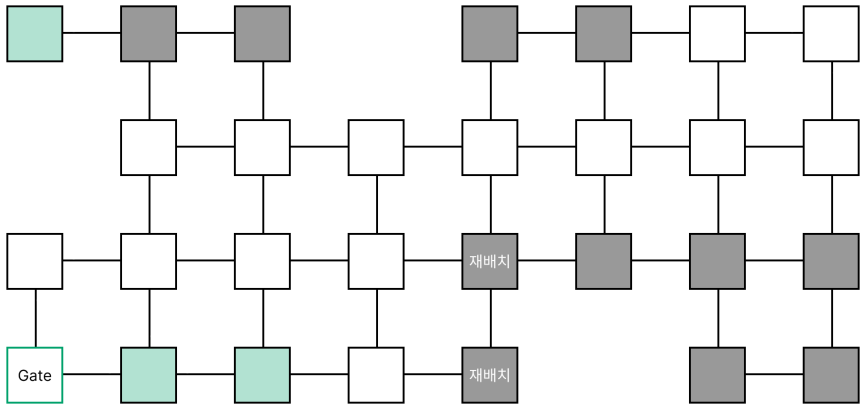
하역 알고리듬

- 재배치 화물은 선적 알고리듬과 동일한 방법으로 재배치를 시도(하역 경로 제외)
- 실패할 경우 임시하역 후 선적 수행



하역 알고리듬

- 동일한 과정을 각 노드 군집마다 반복하여 하역을 완료



무작위성을 활용한 개선 전략

- 현재 알고리듬은 빠른 시간 안에 가능해를 제공하지만, 해의 품질이 우수하지 못함
- 배치 과정의 일부를 무작위로 진행하여 다양한 해를 탐색 시도
- 과거 배치에 크게 의존하는 선적 알고리듬의 특징을 이용/개선

무작위 1. 첫 항구에서 각 군집의 시작점을 무작위로 선택한다.

무작위 2. 각 항구에서 $p-$*화물*의 리스트가 주어졌을 때, 30% 확률로 기존에 배치되어 있던 $p-$*화물* 군집에 확장 배치하지 않고 무작위로 선택된 새로운 노드로부터 확장 배치를 시작한다.

$\rightarrow$timelimit 동안 무작위 탐색을 반복하여 best solution을 최종 해로 도출

알고리듬 구현

- 프로그래밍 언어: C++
- OpenMP를 활용하여 4코어 멀티 스레딩 수행

알고리듬 특장점 및 개선 방향

특장점

- 적당한 가능해를 빠른 시간 안에 제공
- 적은 수의 반복으로 해 품질을 크게 향상
- 사람의 직관과 일치하여 비교적 실현하기 쉬운 선적 계획 수립 가능

개선 방향

- 첫 항구에서의 무작위 반복을 더 빠르게 수렴할 수 있도록 개선
- 각 항구의 선적 단계에서 이후의 수요를 고려하도록 개선

The End