1. **Modelling**

Wi là thời gian hàng tại điểm i sẵn sàng

Drone giúp xe tải lấy hàng rồi đưa tới điểm hẹn cho xe tải ( điểm hẹn = customers’ locations) - Hàng có release date (at the depot)

Var:

N: set of customers {1..n}

D: subset of customers that can be visited by the drone

0: the depot

H: set of trucks

K: set of drones

Xhij = 1 if node j is visited immediately after node i by the truck h #truck route

rk ij = 1 if node j is visited (not necessarily immediately) after node i by the drone k i = {0..n} j={1..n+1} \ {i}

#nếu drone k đến thăm j thì rk0j = 1

#define tất cả các quãng đường giữa các điểm mà drone k đến thăm

ukj = 1, if the drone k flies to node j to resupply the truck with new orders

#discriminate: flies from the depot or the node i

#define các quãng đường mà trực tiếp từ depot tới các điểm của drone k (trong những cái rkij = 1, những cái có ukj = 0, uki = 1 thì quãng đường drone k đi sẽ là ij)

Yhij = 1 if the order of the customer j is loaded onto the truck at node i

#check the orders that are on the truck h

Thi: time when the truck departs node i

ski: time when the drone is launched for node i

epxilon(h) i: waiting time of the truck h at node i

Parameters:

Wi: Release date of the order of the customer i

tij : the truck travel time associated with the edge (i;j)

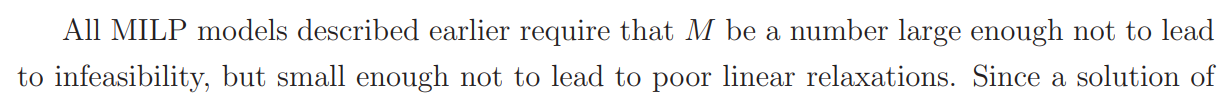
dj : Drone flying time between the depot and the node j

sigma: time for receiving and unloading orders from the drone

ai: Drone load capacity units required to transport the order of customer i

A: drone maximum load capacity

M: A sufficiently large constant



# Text

Description automatically generated: upper bound, wait all orders ready and TSP optimal value

Constrains:

Capacity and time limit of drone

Speed of drone = 1.5 | 2 speed of truck

Truck routing constraints (xij)

+)Mỗi xe truck h xuất phát 1 lần: tổng sigma xh0j = 1 với mọi h thuộc H

+) Mỗi truck về depot 1 lần

+) chỉ có 1 truck đến thăm mỗi node 1 lần: ( them tổng sigma các truck)

+) chỉ có 1 truck rời 1 node 1 lần(trừ depot): them tổng sigma

Drone routing constraints: rkij ukj #uk0 = số node drone k đến thăm

+) tổng rk0j = uk0

+)

Objective: Minimize(max Thn+1 , h thuộc H) ( time truck h phục vụ xong khách hàng cuối cùng và quay về depot)

1. **Method Approach**

Decomposition approach

+) We consider the hypothetical situation where there exist as many drones as customers -> the problem is transformed into a TSP with Time Window(TSP-TW)

(Time Window of customer = Release Date of that order + Flying time to that node)

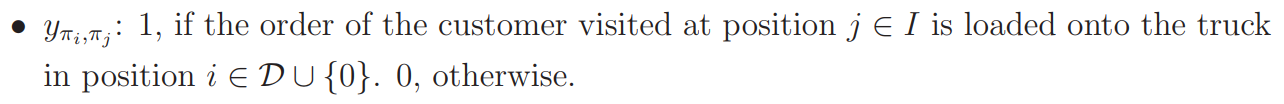
+) After truck-routing -> drone-resupply decisions

Truck route : π = (π0, π1, . . . , π|N|) where πi ∈ N, π0 = 0 - depot

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, Word

Description automatically generated 

Constraints:

Objective: 

1. Init solution

Init solution:

#chia các góc hợp lý:

Mỗi 1 điểm có 2 feature: khoảng cách với depot, release date

* Kết hợp 2 thuộc tính này (sau khi chuẩn hóa dữ liệu)

#chia góc sao cho tổng điểm trong mỗi góc là gần bằng nhau

#Trong mỗi góc: (Xe tải và tập các điểm cần phải thăm)

Xe tải đi theo lộ trình TSP-TW

Earliest Time = Release Date + Flying time

Truck có ràng buộc thời gian sớm nhất để có thể bắt đầu khởi hành ở node i,Ti >=

+) 2-opt giải TSP-TW ?

+)

#Drone: (sau khi đã có truck-routing)

+) Trường hợp hàng ở xa ngoài giới hạn bay của drone

(cộng thêm thời gian chờ của xe tải ở node trước)

Mã hóa:

+ chia làm 2 mảng A[i], B[i]

A[i] = {1,..,n} #integer #vị trí customers

KGM: hoán vị + chia H-1 vách ngăn

#thứ tự các điểm trong ngăn h sẽ là thứ tự điểm đi của xe h

B[i] = [0,1] #real number #

1. Evolutionary Algorithm

Genetic Algorithm (tối ưu hóa rời rạc)

Different Evolutionary Algorithm (

Evolution Strategies (tối ưu tham số thực)

Evolution Programming (học máy, tối ưu số)

Genetic Programming (học máy)

Multifactorial Evolutionary Algorithm

Tabu search

Mã hóa

Khởi tạo

Các phương pháp tạo ra neighborhood: relocation move, exchange move, 2-opt move. (mỗi bước lặp sẽ chọn ngẫu nhiên 1 phương pháp tạo ra neighborhood từ lời giải tốt nhất hiên đang có theo tham số xác suất các pp được cố định từ đầu)

Tạo ra toàn bộ neighborhood xong chỉnh sửa lại lời giải để nó khả thi xong sẽ tìm ra lời giải tốt nhất cho bước lặp tiếp theo

Sau mỗi lần tạo ra neighborhood ví dụ swap (1,3) thì sẽ thêm nó vào tabu list – 1 danh sách cấm sử dụng lại cái này trong 1 số vòng lặp tiếp theo ( số vòng lặp sẽ dựa trên số điểm data để nó k bị swap lại lời giải ban đầu)

1. Genetic Algorithm (GA)

Thường được áp dụng với tối ưu hóa rời rạc.

+ Không quá nhanh, sử dụng các heuristic để mang lại kết quả tốt nhất

+ Lai ghép từ các cá thể cha mẹ tốt, có chọn lọc (nhiều mô hình lai ghép, chọn lọc khác nhau)

Mã hóa lời giải -> Khởi tạo quần thể -> LOOP (Đánh giá độ thích nghi -> Sinh quần thể mới (Lai ghép, Đột biến) -> Chọn lọc -> Kiểm tra điều kiện dừng) -> END

1. Các phương pháp mã hóa lời giải

Mã hóa nhị phân

Mã hóa đa giá trị

Mã hóa hoán vị

Mã hóa cây (cạnh, Prufer, mã hóa đỉnh cha, … )

Prufer:

NetKeys:

Rời rạc hóa

1. Các phương pháp lai ghép

Theo phương pháp mã hóa

Mã hóa nhị phân, đa giá trị:

Lai ghép theo điểm cắt

Lai ghép đồng bộ

Mã hóa hoán vị:

Lai ghép tương hợp bộ phận (PMX)

Lai ghép thứ tự (OX)

Lai chu trình (CX)

Mã hóa số thực:

Lai ghép chéo hóa nhị phân (SBX)

Mã hóa cây:

Lai ghép trộn cạnh

1. Các phương pháp đột biến

Đảo bit (mã hóa nhị phân)

Đổi chỗ (nhị phân, hoán vị)

Đổi giá trị (số nguyên, số thực)

Đảo đoạn (nhị phân, hoán vị, số nguyên)

Đột biến cây

Đột biến đa thức (số thực)

1. Các phương pháp chọn lọc

Chọn lọc ngẫu nhiên

Chọn lọc theo vòng quay roulette

Tính tổng độ thích nghi của quần thể -> xác suất chọn phần tử cá thể thứ i là fitness(i)/sum -> xác suất quỹ tích của cá thể i -> lấy ngẫu nhiên một số thực r [0,1] để xem cá thể i được lựa chọn không.

Chọn lọc theo xếp hạng

Ranking các cá thể (theo fitness hoặc một hàm đặc biệt)

Chọn lọc theo thể thức giao đấu

Chọn ra k cá thể, ghép ngẫu nhiên để giao đấu (so sánh) rồi chọn 2 cá thể tốt nhất làm cha mẹ

1. Các phương pháp đấu tranh sinh tồn

Nạp lại hoàn toàn

Nạp lại ngẫu nhiên

Nạp lại các cá thể ưu tú

1. Genetic Programming

A special Genetic Algorithm

GA biểu diễn các cá thể (NST) dưới dạng chuỗi các alen

GP biểu diễn các cá thể dưới dạng cây các hàm số ( programmings)

* Mục tiêu GP là tìm một chương trình tối ưu trong tập không gian các chương trình (Eg: tối ưu kiến trúc mạng neural)

Đảm bảo ngữ pháp:

Terminal set: variables, constant, module

Functional set: +, -, \*, /, exp, and, or, if else, …

1. Biểu diễn cá thể:

Các nút lá của cây ( Terminal set)

Các nút trong của cây ( Functional set)

Nút gốc được chọn ngẫu nhiên trong Funtional set

Eg:

**y := x \* ln(a) + sin(z) / exp(-x) - 3.4**

* Tập kết thúc: x, a, z, 3.4
* Tập hàm: \*, +, - , /, ln, sin, exp

Diagram

Description automatically generated

1. Các phương pháp lai ghép

Chọn ngẫu nhiên một cây con trong mỗi cây cha mẹ rồi tráo đổi chúng với nhau

1. Các phương pháp đột biến

Đột biến nút trong, Đột biến nút kết thúc

(thay thế bằng thành phần khác trong tập tương ứng)

Đột biến đảo

Chọn ngẫu nhiên một nút trong và đảo 2 nút con của nó

Đột biến phát triển cây

Chọn ngẫu nhiên một nút và thay thế toàn bộ cây con của nút đó bằng một cây con ngẫu nhiên khác

Đột biến gauss

Chọn ngẫu nhiên một nút lá có hằng số và thêm giá trị nhiễu Gauss

Đột biến cắt tỉa cây

Chọn ngẫu nhiên một nút và thay thế bằng một giá trị trong tập Terminal set để nó trở thành nút lá

1. Đánh giá độ thích nghi

Các cá thể được đánh giá trên cùng tập mẫu dữ liệu ( Có giá trị đầu vào và đầu ra, tính hiệu suất trung bình trên mẫu – độ lệch giữa đầu ra của cá thể và nhãn)

1. Evolution Programming

Lấy cảm hứng từ mô phỏng các hành vi trong quá trình tiến hóa

Tìm một tập các hành vi tối ưu trong không gian các hành vi quan sát được

Không sử dụng toán tử lai ghép, chỉ sử dụng toán tử đột biến để sinh quần thể mới

1. Evolution Strategies

Cho hộp đen với hàm mục tiêu cần tối ưu f(x)

Không thể tính được đạo hàm, không lồi, f(x) tất định

Gọi là phân phối của các lời giải tốt cho việc tối ưu f(x)

Nếu dạng phân phối là xác định (giả sử gauss) thì

là tham số mang thông tin về lời giải tốt nhất

được cập nhật qua mỗi thế hệ trong EAs

1. Sơ đồ thuật toán

Bước 1: Sinh một quần thể ban đầu P(t) , với N mẫu.

Bước 2: Đánh giá các cá thể trong P(t)

Bước 3: Chọn một tập con cá thể có độ thích nghi tốt nhất trong P(t) và cập nhật lại

Bước 4: t = t+1 và lặp lại bước 1 cho đến khi thỏa mãn ĐK dừng

1. Các loại ES

Dựa theo chiến lược sinh tồn

(: Chọn cá thể tốt nhất từ cá thể con để sinh tồn ở thế hệ tiếp theo

: Chọn cá thể tốt nhất từ tập hợp của cá thể con và cá thể cha trước đó

Các thuật toán phổ biến

1. Simple Gaussian Evolution Strategies

Phân phối là phân phối Gauss n chiều

lưu trữ thông tin của giá trị trung bình và độ lệch chuẩn

Các bước của thuật toán

Bước 1: Khởi tạo

Bước 2: Sinh ngẫu nhiên cá thể từ phân phối

Bước 3: Chọn ngẫu nhiên cá thể tốt nhất trong P(t+1) để cập nhật lại và

Bước 4: Lặp lại bước 2 và 3

* càng cao: Mức độ khám phá của thuật toán càng lớn

Tuy nhiên giá trị khá tương đồng với

Khả năng hội tụ kém khi cao

Hình dạng của phân phối trong SGES là giống nhau ở mọi thởi điểm

1. Covariance Matrix Adaptation Evolution Strategies (CMA – ES)
2. Differential Evolution

Tối ưu dựa trên quần thể, giải các bài toán tối ưu tham số thực, tìm cực trị hàm đa biến, phi tuyến, không khả vi

1. Sơ đồ thuật toán

Khởi tạo -> LOOP (Đột biến -> Lai ghép -> Chọn lọc -> Kiểm tra điều kiện dừng)

Khởi tạo

Giả sử cần tối ưu D tham số

Tham số k trong khoảng [ Lower Bound k, Upper Bound k]

N >= 4

Mỗi cá thể được biểu diễn trong không gian D chiều

Đột biến

Với mỗi cá thể Ik, chọn ra 3 cá thể Ik1, Ik2, Ik3 ngẫu nhiên khác nhau

Vk = Ik3 + F \* ( Ik2 – Ik1 ) #F hằng số scale độ lệch [0,2]

#Vk: vector đột biến

Lai ghép

Các cá thể con được sinh ra bằng cách lai ghép Ik với Vk

Toán tử lai ghép sử dụng lai ghép nhị thức, sinh ra 1 con

Chọn lọc

Các cá thể con được so sánh với cá thể cha

#Các biến thể khác được sinh ra từ các cách tính vector đột biến

#Hiệu chỉnh các tham số N, CR, F

#N trong khoảng [5D, 10D]

1. Ant Colony Optimization (ACO)

Tối ưu hóa rời rạc

1. Quá trình đi kiếm thức ăn của bầy kiến

Các cá thể kiến đi theo hướng ngẫu nhiên để tìm kiếm thức ăn

Nếu tìm thấy thức ăn, các cá thể kiến mang thức ăn về tổ và để lại một chất hóa học trên đường quay lại tổ của nó (pheromone)

Pheromone trên mỗi đường đi giảm dần theo thời gian

Đường đi nào có pheromone càng cao thì khả năng lựa chọn đi đường đó của các cá thể kiến càng cao

Càng nhiều cá thể kiến tìm thấy thức ăn trên một đường đi thì pheromone của đường đi đó càng cao

1. Sơ đồ thuật toán

Input: Đồ thị G(V,E,w) – đỉnh nguồn s, đỉnh đích t, tham số

Output: Đường đi ngắn nhất giữa s và t

Khởi tạo

Pheromone Tij cho mỗi cạnh (i,j) ; t = 1 ; số lượng kiến N

LOOP ( xây dựng đường đi cho từng cá thể -> cập nhật lời giải tốt nhất -> cập nhật pheromone -> kiểm tra điều kiện dừng )

Xây dựng đường đi cho cá thể kiến k đang ở nút u:

#xác suất đi từ u -> v

là pheromone trên cạnh (i,j)

là mức độ thu hút của cạnh (i,j)

là tập các nút hàng xóm của mà chưa đi qua

và là tham số thuật toán

#= 0: ưu tiên các cạnh được sử dụng nhiều nhất trước đó

# = 0: lựa chọn tham lam theo kinh nghiệm

* Nút v sẽ được lựa chọn theo Roulette

Cập nhật pheromone:

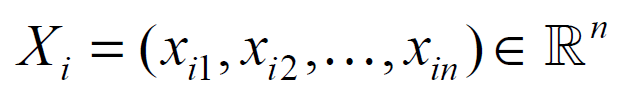
* là tốc độ bay hơi của các pheromone trước đó trên dường đi
* là tổng các pheromone mới mà các cá thể kiến để lại trên đường đi của chúng:

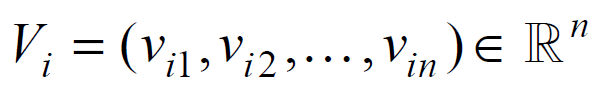
* + là chiều dài của hành trình
  + Q là hằng số kinh nghiệm

1. Particle Swarm Optimization
2. Các thành phần PSO

Swarm: Tập các cá thể (S)

Particle: Các cá thể

Vị trí của các cá thể: 

Vận tốc của các cá thể: 

Vị trí tốt nhất đạt được của cá thể trong quá khứ :

Cá thể tốt nhất của bầy đàn

1. PSO Algorithm

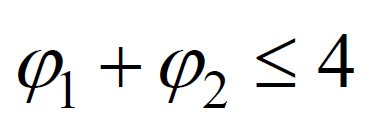
Initial Solution: N particle

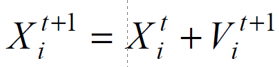
LOOP (Đánh giá độ thích nghi của các cá thể -> Cập nhật vị trí tốt nhất (kinh nghiệm) Pi của các cá thể -> Cập nhật vị trí của cá thể tốt nhất Pg -> Cập nhật vận tốc, vị trí của các cá thể theo Pi, Pg -> kiểm tra điều kiện dừng)

Cập nhật vận tốc: Text

Description automatically generated

#Hệ số gia tốc nhỏ -> hội tụ chậm, lớn -> không hội tụ,

#thông thường 

Cập nhật vị trí: 

1. Các biến thể PSO

Hybrid PSO

GA – PSO

Evolutionary PSO

Hybrid of Differential Evolution and PSO (DEPSO)

Adaptive PSO

PSO in complex environments

1. Multi – Objective Optimization

Quy về đơn mục tiêu

Vector trọng số

Tchebycheff

Penalty – based boundary intersection (PBI)

Pareto Optimal (Use Pareto dominance, Pareto front)

Non – dominated Sorting Genetic Algorithm II (NSGA II)

Strength Pareto Evolutionary Algorithm 2 (SPEA2)

A Multi – Objective Evolutionary Algorithm Based on Decomposition (MOEA/D)