# Multi-robot task allocation algorithm

Gegeben:

* Ungerichteter Graph G mit n+1 Knoten, einer davon der Startpunkt und x Kanten
* Kapazität von 3 Objekten

Algorithmus-Umsetzung nCAR:

## Initialisierung

* Nodes = Knoten
* Vcount, Anzahl an Robotern, unnötig
* Ctotal, Gesamtkosten von allen Pfaden, = 0
* Rlist, Pfadliste, = 0

## While-Schleife

* Jede Iteration berechnet für einen Roboter eine mögliche (optimale) Route
* Läuft solange bis keine Knoten mehr vorhanden sind und alle Knoten als Sets auf die einzelnen Roboter verteilt sind
* Aufruf feasibleRoute(N, C), Bestimmung von möglichen anzufahrenden Knoten *T* für einen Roboter basierend auf seiner Kapazität und
* T ist ein Set von Knoten, *T* ein Set von Knoten, die eine mögliche Route bilden können und ein Set von nicht zugewiesenen Knoten
* Aufruf TSP(*T*), Berechnung eines hamiltonischen Kreises R aus *T* und der aufzubringenden Kosten c
* Für TSP wird der Annäherungsalgorithmus nach Christofides benutzt: „Worst-case analysis of a new heuristic fort he traveling salesman problem“
* R ist ein hamiltonischer Kreis, in dem jeder Knoten einmal angefahren wird und das Ziel gleich dem Start ist
* Hinzufügen von R zur Liste von allen R Rlist für alle Roboter
* Inkrementierung der Anzahl der benötigten Roboter
* Aufsummierung der Gesamtkosten Ctotal mit den aktuellen berechneten Kosten c
* Zuweisung der nichtzugewiesen Knoten zu der Liste der zu untersuchenden Knoten N
* Rückgabe der Liste von allen Routen Rlist, der Anzahl benötigter Roboter Vcount und der Gesamtkosten Ctotal

## feasibleRoute-Funktion

* for-Schleife läuft sooft wie Anzahl an Nodes N
  + Berechnung Kosten von Pfad durch alle Knoten, singleNodeCycle? und setzen als Minimalkosten Pmin
  + wird N(0) und N(i) hinzugefügt, Anfangs- und Endwert
  + wird N ohne N(i) zugewiesen, alle Knoten außer den Aktuellen
  + wird demand(Bedarf) vom aktuellen Knoten N(i).d zugewiesen
  + k wird Zählerwert zugewiesen
  + while-Schleife läuft solange wie kleiner als Kapazität des Roboters C ist
    - nächster Nachbar von i wird bestimmt, unter Berücksichtigung, dass die Kapazität des Roboters C nicht überschritten wird und sein Index wird k zugewiesen
    - wenn kein möglicher Nachbar gefunden wird (k=0), break aus der Schleife
    - sonst um Demand von nächsten Nachbarn erhöhen
    - nächsten Nachbarn hinzufügen
    - Aus nächsten Nachbarn rausstreichen
    - ist Set von nächsten Nachbarknoten, sind nichtzugewiesen Knoten
  + Berechnung Kosten p aus Summe von Eulerkreis von und singleNodeCycle von
  + Wenn neu berechnete Kosten p kleiner als Pmin, dann ist p neues Pmin, neues *T* und neues
* Rückgabe von *T* und , 589 595 596 621 626