

Dynamic Vehicle Routing Systems (DVRP): Eine Übersicht

Basierend auf der Doktorarbeit „The Dynamic Vehicle Routing Problem“ von Allen Larsen

Agenda

- ▶ Motivation
- ▶ statisches & dynamisches VRP
 - ▶ Eigenschaften, Unterschiede, technische Anforderungen, Beispiele
- ▶ Methoden und Probleme
 - ▶ A-priori- und Echtzeitoptimierungsmethoden
- ▶ Grad der Dynamik
 - ▶ Maß zur Bestimmung der Dynamik (mit und ohne Zeitfenster)
 - ▶ Framework zur Klassifikation von Problemen
- ▶ partiell dynamisches VRP
- ▶ kapazitives VRP
- ▶ a-priori dynamisches VRP
- ▶ Erweiterungen / Ausblick

Motivation

- ▶ für Planer immer größere, komplexere Probleme (zeitl. Restriktionen)
- ▶ durch Telekommunikations- und Informationstechnik liegt der Fokus auf Geschwindigkeit und Pünktlichkeit (Just-in-Time)
- ▶ schnelles Reagieren auf Veränderungen
- ▶ verstärktes Interesse an dynamischen Transportmodellen, in denen die Daten als zeitabhängig betrachtet werden können
- ▶ Verfügbarkeit und Umgang mit einer Menge an Echtzeitinformationen parallel zum Herausfinden der Route
- ▶ DVRP ist ein Musterbeispiel, in dem das intelligente Benutzen der Echtzeitinformationen eine Firma von einer anderen unterscheidet
- ▶ kosteneffizientes Routen
- ▶ 10-15% des BIP sind Distributionskosten → auch kleine Verbesserungen wichtig

statisches & dynamisches VRP

Statisches VRP

- ▶ Musterbeispiel:TSP
- ▶ VRP ist Verallgemeinerung vom TSP
- ▶ Ziel: Minimierung der Kosten
- ▶ alle planungsrelevanten Informationen vorher bekannt
- ▶ relevante Informationen ändern sich nicht, nachdem die Route erstellt wurde

statisches & dynamisches VRP

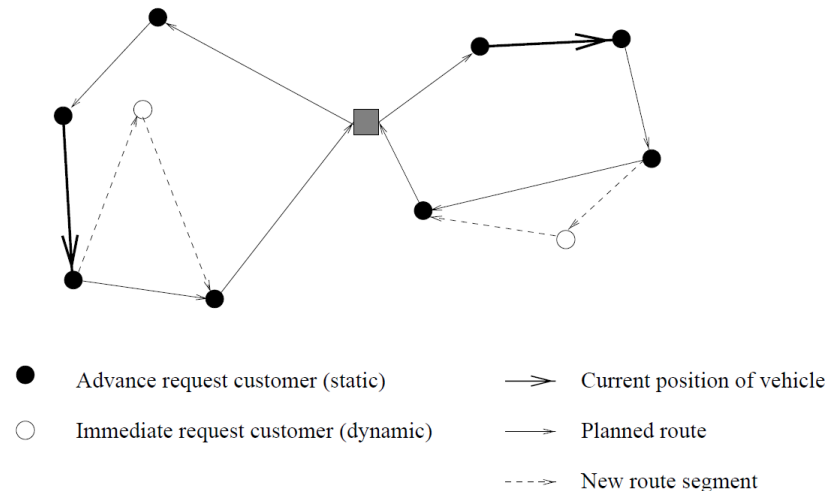
Dynamisches VRP

- ▶ nicht alle relevanten Informationen zum Planungszeitpunkt bekannt
- ▶ relevante Informationen können sich ändern, nachdem die Routen erstellt wurden
- ▶ Informationen: geografische Lage, vor Ort Servicezeit, Kapazitäten, Reisezeiten, ...
- ▶ Zeit ist immer wichtig
- ▶ Zeitpunkt, wann wichtige Informationen bekannt gemacht werden, unterscheidet dynamische von statischen VRPs

statisches & dynamisches VRP

Dynamisches VRP

- ▶ ideal (rechts):
 - ▶ dynamische Aufträge in geplante Routen einfügen, ohne dass die Reihenfolge noch nicht besuchter Kunden verändert wird
 - ▶ minimale Verzögerung
- ▶ in Praxis (links):
 - ▶ kompliziertes Einfügen
 - ▶ Re-Planning
- ▶ je begrenzter und komplexer das Routingproblem, desto komplizierter das Einfügen
(Bsp.: Zeitfensterbeschränkungen)



Quelle: Larsen, Allen: Dynamic Vehicle Routing Problem S. 6

statisches & dynamisches VRP

Dynamisches VRP

- ▶ Untersuchung von DVR-Systemen durch Verwendung von zufällig generierten Daten (besser zu verarbeiten als Daten aus Realität)
- ▶ relevante Punkte für Datengenerierung
 - ▶ Methode zum Generieren der Zeiten, zu denen dynamische Aufträge eintreffen
 - ▶ Verwendung von Verteilungen, die die Bedürfnisse der Kunden und die vor Ort Servicezeiten charakterisieren
 - ▶ geografische Position der Kunden
 - ▶ Reaktionszeiten des Systems
 - ▶ Reisezeiten

statisches & dynamisches VRP

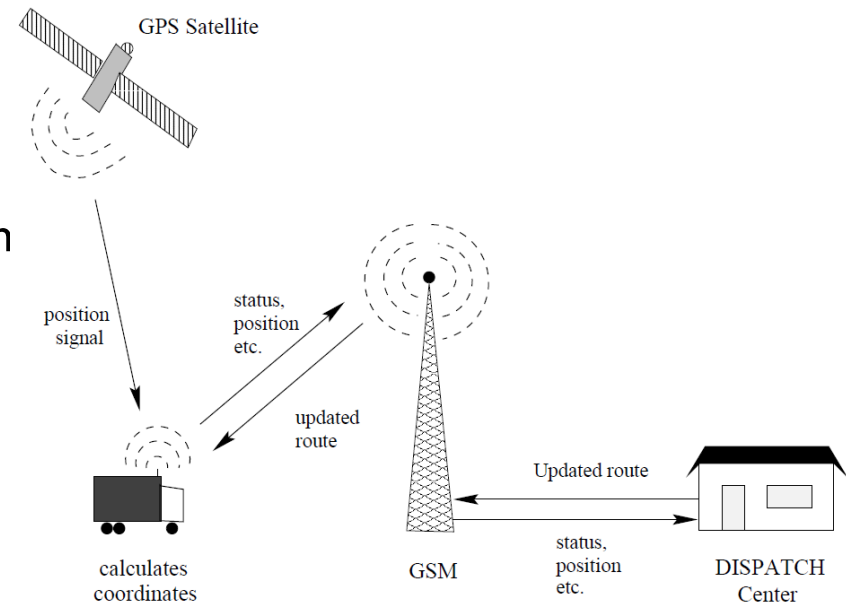
Statisch vs. Dynamisch

- ▶ Zeit ist immer wichtig (wo sind Fahrzeuge zum bestimmten Zeitpunkt)
- ▶ Problem ohne festes Ende (Routen / Pfade beliebig erweiterbar)
- ▶ Informationen über Zukunft dürfen unpräzise oder unbekannt sein
- ▶ kurzfristige Events sind wichtiger
- ▶ Update-Mechanismus für Informationen ist sehr wichtig (bei Veränderungen, in Lösungsmethode integriert)
- ▶ Gewährleistung von Neuordnung und Neuzuweisung zur Verbesserung suboptimaler Routen
- ▶ schnellere Berechnungszeit notwendig, um Lösung so schnell wie möglich zu finden (oftmals Verbesserung durch Heuristiken)
- ▶ Mechanismus für unbegrenzten Aufschub wichtig (bspw. bei grausigen geografischen Gegebenheiten und Zeitfensterbeschränkungen)
- ▶ Zielfunktionen dürfen unterschiedlich sein
 - ▶ statische ZF nicht sinnvoll, da Prozess unbegrenzt laufen kann
 - ▶ keine Infos über Zukunft → Optimierung der bekannten Inputs
- ▶ Zeitbeschränkungen dürfen unterschiedlich sein (Zeitfenster verletzen als Auftrag ablehnen)
- ▶ geringere Flexibilität beim Variieren der Fahrzeugflotte
- ▶ Betrachtung von Warteschlangen wird wichtig, um das System nicht zu überfordern

statisches & dynamisches VRP

technische Anforderungen DVRP

- ▶ Routingssystem mit aktuellen Informationen versorgen
- ▶ Equipment zur
 - ▶ Positionsbestimmung
 - ▶ Übermittlung zu fixen Zeitpunkten und
 - ▶ Interpolationsschema
 - ▶ Kommunikation
 - ▶ Mobilfunknetz
 - Anschaffungskosten ↓
 - Betriebskosten ↑
 - gesunkene Preise
 - ▶ Rundfunknetz



Quelle: Larsen, Allen: Dynamic Vehicle Routing Problem S. 14

statisches & dynamisches VRP

Beispiele aus der Realität

- ▶ Traveling Repairman
- ▶ Kurier Service (Austragen und Einsammeln, Transport von A nach B)
- ▶ Verteilung von Heizöl (80% statisch, 20% dynamisch)
- ▶ dynamische Dial-A-Ride Systeme (ein oder mehrere „Produkte“ auflesen und zu anderem Ort bringen, wo sie „verteilt werden“)
- ▶ Taxi (sehr dynamisch, oftmals bestimmte Wartepositionen)
- ▶ Rettungsdienst (Echtzeit)

Probleme und Methoden

A-priori optimierungsbasierte Methoden

- ▶ wahrscheinliche Informationen über unsichere zukünftige Events
- ▶ Planer bestimmt eine oder mehr Routen basierend auf wahrscheinlichen Informationen über zukünftige Servicerequests, Kundenbedarfen, Reisezeiten oder anderen Parametern
- ▶ Routen werden geplant, bevor Fahrzeuge das Depot verlassen
- ▶ Re-Optimierung der Route jedes Mal durchführen, wenn es neue Infos gibt
 - ▶ sehr zeitaufwändig das Problem jedes Mal neu zu lösen
 - ▶ Verwirrung der Fahrer durch neue Routen
 - ▶ Verschlechterung der Beziehung zum Kunden
- ➔ Lösung: a-priori Strategien als Alternative zur Re-Optimierung
- ▶ Beispielprobleme:
 - ▶ probabilistisches TSP (Wahrscheinlichkeiten an Knoten)
 - ▶ probabilistisches VRP (wahrscheinlichkeitsbasierte Bedarfe)
 - ▶ **stochastisches VRP**

Probleme und Methoden

Stochastisches VRP

- ▶ einige Elemente des Problems sind stochastisch
 - ▶ Bsp.: unbekannte Kunden, ungewisse Reisezeiten
- ▶ Verallgemeinerung vom Probabilistischen VRP
- ▶ zwei Phasen
 - ▶ 1): a-priori Planung
 - ▶ 2): „Recourse“, um mit Problemen, wie erschöpfte Kapazitäten, umzugehen, was in diesem Fall die Kosten des Umweges zum Depot darstellt
- ▶ Recourse ist ein Verfahren / eine Policy für den Umgang mit Problemen
- ▶ Alternative: Besuch des Depots, wenn man in der Nähe ist und die Restkapazität unter einem bestimmten Grenzwert liegt

Probleme und Methoden

Stochastisches VRP

- ▶ Einteilung in sechs Kategorien
 - ▶ TSP mit stochastischen Kunden (probabilistisches TSP)
 - ▶ TSP mit stochastischen Reisezeiten (Kantenkosten variieren, Ziel: Maximierung der Wahrscheinlichkeit die Tour in einer gegebenen Deadline zu beenden)
 - ▶ m-TSP mit stochastischen Reisezeiten (für m Fahrzeuge, Start und Ende am Depot, Ziel: Minimierung #Fahrzeuge)
 - ▶ VRP mit stochastischen Bedarfen (Bedarfe sind zufällig)
 - ▶ VRP mit stochastischen Kunden (wie TSP mit stochastischen Kunden, nur mit deterministischen Bedarfen)
 - ▶ VRP mit stochastischen Kunden und Bedarfen (probabilistisches VRP)

Probleme und Methoden

Echtzeit Optimierungsmethoden

- ▶ alle zuvor präsentierten Methoden benutzen stochastische oder wahrscheinlichkeitsbasierte Informationen
- ▶ Routen wurden geplant, bevor ein Fahrzeug das Depot verlässt
- ▶ jetzt: Konstruktion der Routen, wenn sich die Fahrzeuge auf ihren Routen befinden
- ▶ Beispielprobleme:
 - ▶ Dynamisches TSP
 - ▶ **Dynamisches TRP**

Probleme und Methoden

Dynamisches TRP

- ▶ Sammeln von Bedarfen, Erschaffen von Touren und Verteilen von Fahrzeugen
- ▶ Wartezeit ist wichtiger als Reisekosten
- ▶ Wartezeiten: Zeit von Eintreffen von Bedarf i bis Start des Service
- ▶ Techniker, dynamische Bedarfe, gleichverteilte Einsatzorte
- ▶ gleiche vor Ort Servicezeit für alle Bedarfe
- ▶ Systemzeit: Zeit zw. Eintreffen von Bedarf i und die Zeit, nach der der Service beendet wurde
- ▶ Ziel: Designen einer Routing Policy, die die Systemzeit minimiert
- ▶ durch Analysen von diversen Policies konnten untere Grenzen für die minimale Systemzeit ermittelt werden
 - ▶ mit zunehmendem Verkehr steigt die Grenze

Probleme und Methoden

Dynamisches TRP - einige Policies

- ▶ First Come First Serve (FCFS)
- ▶ FCFS Stochastic Queue Median (FCFS-SQM)
 - ▶ direkt vom Median der Service Region zum Kunden
 - ▶ nach Beenden des Service Rückkehr zum Median und Warten auf neue Serviceanfrage
- ▶ Nearest Neighbor (NN)
 - ▶ nach Beenden des Service Reise zum nächsten Nachbarn mit Bedarf
- ▶ Traveling Salesman Problem (TSP)
 - ▶ Bedarfe zu einer Menge der Größe n gesammelt
 - ▶ wenn Menge voll, wird optimale TSP-Tour berechnet und Bedarfe in der entsprechenden Reihenfolge befriedigt
 - ▶ #Mengen $t > 1 \rightarrow$ FCFS
- ▶ Space Filling Curve (SFC)
- ▶ Partitioning Policy (PART)
 - ▶ Service Region in m^2 Subregionen aufteilen \rightarrow Abarbeitung mit FCFS
 - ▶ sind alle Bedarfe einer Subregion befriedigt \rightarrow Reise zur nächsten

Probleme und Methoden

Ähnliche Probleme

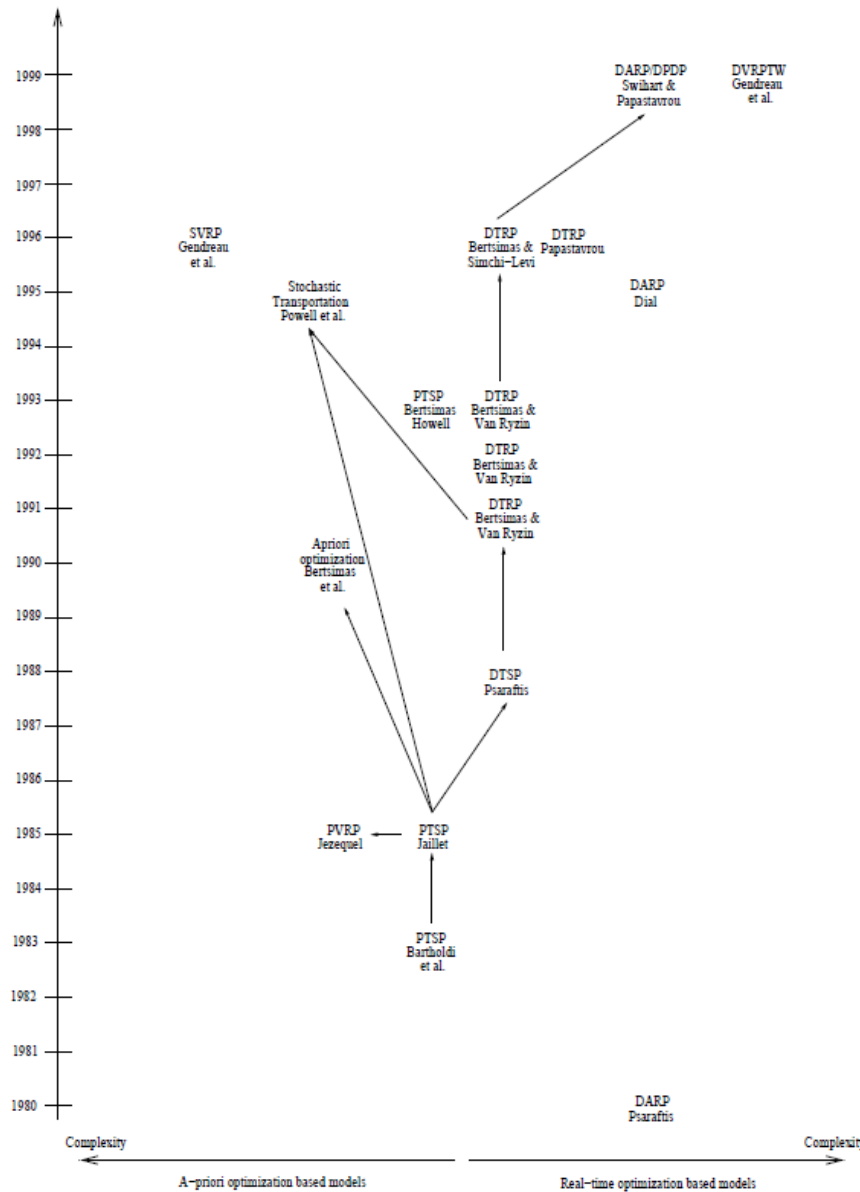
Echtzeit Fahrzeug Routing und (Zeit-)Planung (DVRPTW)

- ▶ Motivation durch Kurierservice
- ▶ Kunden erscheinen in Echtzeit und müssen innerhalb eines soften Zeitfensters bedient werden
- ▶ Heuristiken auf parallele Plattform implementiert

Zeitabhängige TSP und VRP

- ▶ Ziel: Minimierung der Gesamtzeit der Routen
- ▶ Reisezeit abhängig von Weg und Tageszeit (Berufsverkehr, Staus)
- ▶ alle anderen Daten sind statisch / deterministisch
- ▶ verschiedene Heuristiken auf Basis von Nearest Neighbor

Probleme und Methoden - Chronologischer Überblick



horizontal:
 Basismodelle wie
 DTRP oder PTSP
 sind mittig und
 korrespondieren
 mit Komplexität

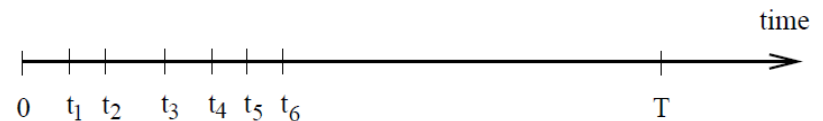
Bspw. ist DVRPTW
 sehr weit rechts,
 was die hohe
 Komplexität des
 Systems zeigen soll

Quelle: Larsen, Allen: Dynamic Vehicle Routing Problem S. 41

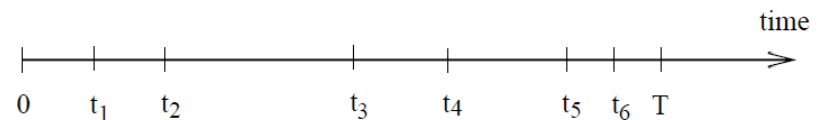
Grad der Dynamik

- ▶ $(\# \text{dyn. Requests}) / (\# \text{total Requests})$
- ▶ in Szenario A kürzere Route als in Szenario B, da frühzeitig alle Informationen bekannt sind
 - darum höhere Planungsqualität
 - optimale TSP-Tour konstruierbar
- ▶ Szenario B vom Planer bevorzugt, da er mehr Zeit hat, um auf dynamische Requests reagieren zu können
- ▶ größte Anzahl in einem Pool wartender Anfragen verbessert Qualität der Lösung im Hinblick auf Minimierung der gesamten Fahrstrecke
- ▶ wenn bereits ältere Requests im Wartepool sind:
 - ▶ Szenario A ist schwieriger zu bewältigen, da mehr Requests in kürzerer Zeit zu bearbeiten sind

SCENARIO A:



SCENARIO B:



Quelle: Larsen, Allen: Dynamic Vehicle Routing Problem S. 57

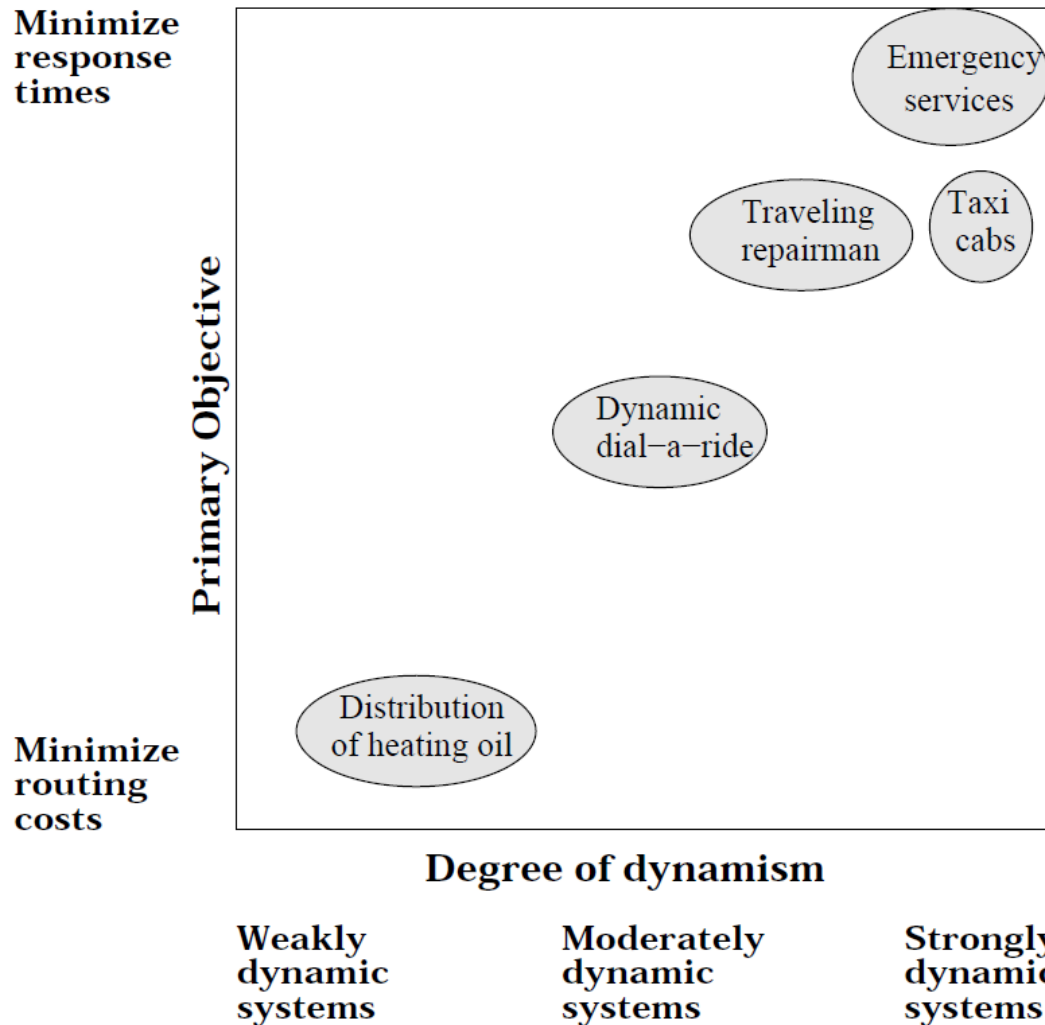
Grad der Dynamik

- ▶ aber: keine Betrachtung von Ankunftszeiten der dynamischen Requests
- ▶ somit ist ein System, das dynamische Requests zu Beginn der Planung hat, gleich einem System, das dynamische Requests erst spät während des Tages bekommt (in Realität nicht)

Erweiterung zum effektiven Grad der Dynamik

- ▶ Erweiterung um Betrachtung der Zeiten, wann dynamische Requests eintreffen
- ▶ beschreibt wie spät durchschnittlich ein Request empfangen wird im Vergleich zum spät möglichsten Zeitpunkt an den der Request empfangen werden könnte
 - ▶ $edod = 0 \rightarrow$ rein dynamisch
 - ▶ $edod = 1 \rightarrow$ rein statisch

Grad der Dynamik - Klassifikation



Quelle: Larsen, Allen: Dynamic Vehicle Routing Problem S. 63

Grad der Dynamik - Klassifikation

Schwach dynamische Systeme

- ▶ Bsp.: Verteilung Heizöl, Flüssiggas → mehr als 80% der Kunden bekannt
- ▶ dynamische Requests je nach verfügbaren Ressourcen bearbeitet
- ▶ Problemlösung durch Anpassung statischer Prozeduren
- ▶ bei jedem neuen Input wird ein solches Problem gelöst

Stark dynamische Systeme

- ▶ Bsp.: Rettungsdienste, Taxi
- ▶ schnelle Veränderung der Daten und alle Requests sind wichtig
- ▶ nur geringe Qualität von a-priori Informationen bezüglich Ort, Bedarfen,...
- ▶ wenn jedoch a-priori Informationen oder Erwartungen vorhanden sind, sollten diese auch verwendet werden
- ▶ Queueing bei heavy Traffic

Grad der Dynamik - Klassifikation

Moderat dynamische Systeme

- ▶ Bsp.: Kurierservice
- ▶ wartende / geplante und dynamische Kunden gemischt
- ▶ im Vergleich zu schwach dynamischen Systemen haben dynamische Requests bereits einen beachtlichen Teil der Gesamtanfragen eingenommen
- ▶ zusätzliche stochastische Elemente z.B. unbekannte Kundenbedarfe
- ▶ schwer zu lösen: oftmals Aufschieben bis zum letzt möglichen Zeitpunkt → höhere Entscheidungsqualität durch sinkende Unsicherheit (Idealfall)

- ▶ Alternative: Nutzen der Zeit zwischen Eingabeupdates zur Verbesserung potentieller Routen, was aber detaillierte Informationen über zukünftige Requests benötigt

Partiell Dynamisches TRP

- ▶ Variante des Dynamischen TRPs (alle Kundenaufträge dynamisch)
- ▶ hier: vorhandene und dynamische Kundenaufträge
- ▶ Definition:
 - ▶ konstante Geschwindigkeit
 - ▶ begrenztes Gebiet
 - ▶ Menge an dynamischen Bedarfen (Poisson-verteilt)
 - ▶ Einsatzorte unabhängig voneinander gleich verteilt
 - ▶ jeder Bedarf benötigt identisch verteilte vor Ort Servicezeit
 - ▶ Routenupdate erfolgt an Einsatzorten
 - während der Reise ist Ziel nicht änderbar
 - ▶ Ziel: Minimierung der Routingkosten des Technikers

Partiell Dynamisches TRP

▶ Zwei Unterschiede zum DTRP

- ▶ 1) Menge an bekannten Einsatzorten, Servicezeiten bis zum Ende des Service unbekannt

- ▶ 2) Versuch, verschiedene Ziele zu optimieren

- ▶ DTRP: Minimierung der Gesamtzeit

- ▶ PDTRP: Minimierung der zurückgelegten Strecke

+

Kunden, mit dynamischen Aufträgen sollten in einer priorisierten Reihenfolge bedient werden, so dass Kunden die erst spät anrufen nicht vor Kunden bedient werden, die bereits lange warten (Bsp.: bekannte Aufträge am Morgen, dynamische Aufträge am Nachmittag)

Partiell Dynamisches TRP

Routing Policies

- ▶ FCFS (WZ), FCFS-SQM (WZ), NN (Kosten), PART (Mix)
- ▶ FCFS:
 - ▶ zuerst bekannte Aufträge geplant ohne die Orte zu betrachten
 - ▶ dynamische Aufträge werden Planung so hinzugefügt, wie empfangen
 - ▶ Szenarios mit mehr als einem bekannten Auftrag: Betrachtung des Einsatzortes und Konstruktion einer Route mit Hilfe von NN → NN-FCFS
- ▶ FCFS-SQM:
 - ▶ asymptotisch optimale Systemzeit bei light Traffic
 - ▶ aber: längere Wege als in FCFS, wenn Dreiecksungleichung gilt
- ▶ PART:
 - ▶ bekannte Kunden (in Subregionen) mit Hilfe von NN besuchen
 - ▶ #Subregionen durch Optimierung finden

Partiell Dynamisches TRP

Ergebnisse

- ▶ kurze Routen → viel Wartezeit für Kunden, da nur Distanz optimiert
- ▶ Wartezeit optimierende Policies hängen vom Grad der Dynamik ab
 - ▶ je statischer ein System, desto mehr bekannte Kunden, die vorher bedient werden müssen
 - ▶ lange Wartezeiten für dynamische Kunden
- ▶ \emptyset zurückgelegte Strecke von FCFS und FCFS-SQM fast Dynamik-unabhängig
- ▶ PART hat als Mix aus Wartezeit und Routingkosten als Ergebnis Routenlängen, die zwischen den anderen Policies liegen; weder gut noch schlecht
- ▶ weniger Subregionen → bessere Routen bezüglich Distanz
- ▶ linearer Zusammenhang zw. Grad der Dynamik und Routenlänge
 - ▶ Dynamik steigt → lineare Erhöhung der Routenlänge für alle Policies
- ▶ für verkehrsreiche Systeme bringt NN Policy bessere Ergebnisse

Partiell Dynamisches TRP

Erweiterungen

- ▶ a-priori Infos sind interessant, aber nicht für gesamten Planungszeitraum oder gesamte Serviceregion bekannt. Aber: diese Infos sind für kürzere Zeitspannen und zusammengefasste geografische Regionen bekannt
- ▶ per Algorithmus die Serviceregion in Subregionen und den Tag in Zeitspannen einteilen
- ▶ a-priori Infos können genutzt werden, die Subregionen mit den meisten erwarteten Requests in der nächsten Zeitspanne zu finden und zu bedienen

- ▶ Sammeln von Bedarfen in Mengen → Menge voll: Policy zur Routenminimierung
 - ▶ häufige Updates müssen gemacht werden: NN
 - ▶ wenige Updates müssen gemacht werden: TSP Policy

- ▶ Betrachtung anderer Kriterien für bspw. zeitorientierte Ziele

Kapazitatives DVRP mit Zeitfenstern

- ▶ dynamische Version VRP mit Zeitfensterbeschränkungen
- ▶ Motivation: Routing Problem von Firmen, die Heizöl an private Haushalte ausliefern
- ▶ zuvor als statisch angesehen, da dem Planer alle Infos bekannt sein sollten
- ▶ aber: 10-20% verbrauchen das gesamte Öl und müssen so schnell wie möglich und bevorzugt bedient werden
- ▶ Bedarfe sind stochastisch, da der Ölverbrauch vom Kunden zu Kunden verschieden ist
- ▶ Fahrzeuge können leer werden und müssen zum Depot zurück

- ▶ Betrachtung von **Batching-Strategien**

Kapazitives DVRP mit Zeitfenstern

Batching-Strategien

- ▶ Idee: Verzögern der Planung von Routen, bis bestimmte #Aufträge eingegangen oder eine gewisse Zeit vergangen ist (seit der letzten Planung) → Auftragsammlung
- ▶ gröÙe- (n Requests) oder zeitgetrieben (fixe Zeitpunkte)
- ▶ durch Aufschieben der Planung sollen zu diesen Zeitpunkten mehr detaillierte und aktuellere Informationen vorhanden sein, die die Qualität der Lösung verbessern (Optimierung)
- ▶ größtes Hindernis in Realität: dynamische Kunden innerhalb eines Zeitfensters bedienen
- ▶ Mix aus Echtzeit-Einfügen und Batching Strategien
 - ▶ Bei Eintreffen eines neuen Requests muss eine Stelle gefunden werden, die das Einfügen ohne Neuplanung der Kunden, die bereits Teil der Lösung sind, erlaubt.
 - ▶ Reoptimierung aller Kunden, die bedient werden sollen, wird durch Batching Strategie durchgeführt

Kapazitives DVRP mit Zeitfenstern

Ergebnisse

- ▶ durchschnittliche Distanz ist höchst abhängig von der Dynamik und der Datenstruktur
- ▶ Verspätung und das Maß, wie viele Kunden gezwungen eingefügt werden müssen, steigen steil für steigende Werte vom Grad der Dynamik
- ▶ Batching Strategien brachten keine Verbesserung
- ▶ in einigen Fällen konnte die Distanz durch Größe-getriebene Strategie verkürzt werden
- ▶ allerdings erfolgen die Verbesserungen auf Kosten der Verspätung
- ▶ viele Testversuche kamen zum selben Ergebnis
- ▶ effektiver Grad der Dynamik mit Zeitfenstern bringt keine Mehrinformationen im Vergleich zu den anderen Maßnahmen
- ▶ Glaube an ein Verbesserung mit Hilfe der Batching Strategien (Verbesserung der Lösung in Idle-Zeiten)

A-priori Dynamisches TSP mit Zeitfenstern

Beschreibung: Dynamisches TSP mit Zeitfenstern

- ▶ Service muss innerhalb eines gegebenen Zeitfensters beginnen
- ▶ softe Zeitfenster, d.h. Zeitfenster dürfen verletzt werden, aber die Ziel-/Kostenfunktion bekommt eine Strafe für Verspätung
- ▶ quadratische Serviceregion mit n gleich großen Subregionen
- ▶ Definition einer Menge von Idle-Points (mögliche Rastpositionen)
- ▶ Idle-Points und Subregionen stehen nicht notwendigerweise in Beziehung
- ▶ vor Ort Servicezeiten unbekannt, bis der Service endet (log-normal)
- ▶ Service kann nicht beginnen, solange das Zeitfenster nicht offen ist, d.h. das Fahrzeug muss ggf. an der aktuellen Position warten

- ▶ Erweiterung um a-priori Informationen bzgl. der Ankunftsintensitäten
- ▶ Verbesserung bzgl. gefahrener Distanz oder der Gesamtzeitverspätung

A-priori Dynamisches TSP mit Zeitfenstern

Routing Policies

- ▶ zwei Situationen nach Beenden des Service
 - ▶ 1: Zeitfenster des nächsten Kunden ist bereits auf oder wird offen sein, wenn das Fahrzeug den Kunden erreicht
 - ▶ 2: Zeitfenster nicht offen oder keine Kunden → Fahrzeug muss warten
- ▶ Warten beim letzten Kunden oder an einem Idle-Point
- ▶ NEAREST: Fahrzeug zum nächsten Idle Point, relativ zur aktuellen Position
- ▶ BUSIEST: Fahrzeug fährt zum Idle Point mit höchster Ankunftsintensität
- ▶ HI-REQ: Fahrzeug zum Idle Point mit höchstem Attraktivscore (Anzahl erwarteter Kunden in einem bestimmten Zeitintervall)

- ▶ NO REPOS: keine Repositionierung (reine Reoptimierung)

A-priori Dynamisches TSP mit Zeitfenstern

Ergebnisse

- ▶ NO REPOS-Policy ist Referenz
- ▶ Serie I: Minimierung der Distanz:
 - ▶ reine Reoptimierung besser, da Repositionierung die Distanz erhöht
- ▶ Gesamtdistanz ist stark abhängig von der Verteilung der Kunden
- ▶ Serie II: Minimierung der Verspätung:
 - ▶ alle Repositionierungspolicies haben für den kleinsten Grenzwert eine größere Anzahl besserer Performances als NO REPOS.
 - ▶ durchschnittliche Verspätung nur sehr wenig verbessert
- ▶ insgesamt betrachtet brachten Repositionierungspolicies schlechte Ergebnisse
- ▶ HI-REQ beste Performanz für beide Serien und alle Szenarios (für Repositionierung)

A-priori Dynamisches TSP mit Zeitfenstern

Ergebnisse

- ▶ min. Distanzen bei 50% Dynamik, ab 80% Dynamik steigt Distanz
- ▶ je größer Dynamik, umso kleiner durchschnittliche Verspätung
- ▶ für stark dynamische Systeme ist Repositionierung besser als Reoptimierung wenn Verspätung minimiert werden soll
- ▶ steigende #Repositionen für steigende Dynamik

- ▶ Ergebnisse zeigen Potential zur Reduzierung der Verspätung wenn a-priori basierte Repositionierungspolicies verwendet werden
aber: auf Kosten einer längeren Distanz
- ▶ HI-REQ hat größte Anzahl von besten Performances bzgl. Minimierung der Verspätung
- ▶ keine eindeutigen Aussagen, welche Policy wann benutzt werden sollte

A-priori Dynamisches TSP mit Zeitfenstern

Weitere Arbeit

- ▶ natürliche Erweiterung: Verallgemeinerung für Verwendung multipler Fahrzeuge, was zu anderen strategischen Entscheidungen führt
- ▶ Untersuchung der potentiellen Einsparungen bei Erlauben von Abweichungen von der ursprünglichen Routen, wenn ein vielversprechender Request in der nahen Umgebung eingegangen ist
- ▶ Erweiterung von HI-REQ um die Betrachtung der Ankunftsintensitäten in der Nachbarschaft bei der Berechnung des Attraktivscores → dadurch können qualitativ bessere Idle Points bestimmt werden

Erweiterungen / Ausblick

Meta-Heuristiken

- ▶ Bsp.: Tabu Suche
- ▶ benötigen viel Rechenzeit für gute Ergebnisse
- ▶ in einigen Anwendungen in der Realität anwendbar
- ▶ Finden einer möglichen Lösung innerhalb weniger Sekunden, was wichtig ist, wenn der Planer prüfen will, ob ein Auftrag abgelehnt werden muss oder ob ein Kunde bedient werden kann
- ▶ initiale Lösung wird nach Eintreffen von dynamischen Requests schrittweise verbessert, wenn es die zeitliche Verteilung der dynamischen Requests erlaubt oder sich das System in einer Idle Phase befindet
- ▶ Schwierigkeiten, wenn Ankunftsintensitäten sehr hoch

- ▶ Annahme, dass Meta-Heuristiken die Performance verbessern, wobei der Erfolg von der zeitlichen Verteilung der dynamischen Requests abhängt

Erweiterungen / Ausblick

Methode des schnellen Einfügens

- ▶ viele Kunden folgen dem gleichen Bedarfsmuster (bspw. Anrufe zu bestimmten Zeiten)
- ▶ Speicherung dieser historischen Informationen
- ▶ Algorithmus, der zuvor Dummy-Kunden als Platzhalter in Route einfügt und bei einem neuen dynamischen Request einen oder mehrere Dummies entfernt
- ▶ entweder passt der neue Request auf einen alten Slot (auf dem vorher der Dummy war) oder es muss ein re-shuffling erfolgen
- ▶ #Dummies aus geschätzten Ankunftsintensitäten, Orte der Dummies aus gespeicherten historischen Informationen
- ▶ auf Kurierservice anwendbar
- ▶ geografischer Ort der Idle Points könnte als Ort für Dummies während Abholung dienen
 - ➔ sehr anwendungsspezifisch

Erweiterungen / Ausblick

Entwicklungen nach 2000

- ▶ Framework von Larsen für DRS basierend auf dem Grad der Dynamik
 - ▶ Schwach: Grad der Dynamik $< 20-30\%$
 - ▶ Moderat: Grad der Dynamik ist zwischen 30 und 80%
 - ▶ Stark: Grad der Dynamik $> 80\%$
- ▶ Echtzeit Methoden zur Reduzierung der Verspätung beim PDTRPTW; Ergebnis: alle Policies verbessern Verspätung signifikant mit nur kleinen Distanzerhöhungen
- ▶ Definition von Warte- und Verlagerungspunkten mittels a-priori Heuristiken
 - ▶ Ausnutzung der Verteilung und Zusammensetzung der Kunden
 - ▶ Fahrzeug kann warten oder den Standort wechseln überall und zu jeder Zeit
 - ▶ Maximierung der bedienten Kunden
- ▶ Analyse von Wartepolicies, die die Wahrscheinlichkeit maximieren, einen zusätzlichen dynamischen Auftrag bearbeiten zu können
- ▶ weitere Forschung in wahrscheinliches Wissen über Ankunft zukünftiger Requests; prognostizierte Requests als Dummy-Kunden in die Routen eingebaut
- ▶ Lösung des DVRP über Ameisen Algorithmen

Vielen Dank
für die Aufmerksamkeit!

Fragen???