Summer 2016

Data-centric solution methodologies for vehicle routing problems

Fahrettin Cakir
University of Iowa

DATA-CENTRIC SOLUTION METHODOLOGIES FOR VEHICLE ROUTING PROBLEMS

by

Fahrettin Cakir

A thesis submitted in partial fulfillment
of the requirements for the Doctor of Philosophy
degree in Business Administration in the
Graduate College of
The University of Iowa

August 2016

Thesis Supervisors: Professor W. Nick Street
Associate Professor Barrett W. Thomas

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

_____

PH.D. THESIS

_____

This is to certify that the Ph.D. thesis of

Fahrettin Cakir

has been approved by the Examining Committee for
the thesis requirement for the Doctor of Philosophy degree
in Business Administration at the August 2016 graduation.

Thesis Committee: _____
W. Nick Street, Thesis Supervisor

_____
Barrett W. Thomas, Thesis Supervisor

_____
Samuel Burer

_____
Gautam Pant

_____
Xun Zhou

To my family.

ACKNOWLEDGEMENTS

I am indebted to my advisors, Nick Street and Barrett Thomas. Research questions tend to spawn many other questions and it becomes a challenge in itself to stay focused and on track. Their helpful direction has made it possible for me to discipline my research habits. Regular meetings with them have significantly broadened and improved my perspective when tackling scientific problems that can be very challenging and equally diverse. I would like to thank both of them not only for their time and effort, but for the insights they have provided which has made me a better researcher. I would like to thank Samuel Burer for his help on the mathematical formulation of assignment problems and their solution approaches. I would like to thank Xun Zhou for bringing to my attention the literature on spatial interpolation. Finally, I am thankful to Gautam Pant for his general support and guidance in my other responsibilities as a graduate student.

I would like to especially thank Renea Jay for her patience and assistance in my very long journey in graduate school.

ABSTRACT

Data-driven decision making has become more popular in today's businesses including logistics and vehicle routing. Leveraging historical data, companies can achieve goals such as customer satisfaction management, scalable and efficient operation, and higher overall revenue.

In the management of customer satisfaction, logistics companies use consistent assignment of their drivers to customers over time. Creating this consistency takes time and depends on the history experienced between the company and the customer. While pursuing this goal, companies trade off the cost of capacity with consistency because demand is unknown on a daily basis. We propose concepts and methods that enable a parcel delivery company to balance the trade-off between cost and customer satisfaction. We use clustering methods that use cumulative historical service data to generate better consistency using the information entropy measure.

Parcel delivery companies route many vehicles to serve customer requests on a daily basis. While clustering was important to the development of early routing algorithms, modern solution methods rely on metaheuristics, which are not easily deployable and often do not have open source code bases. We propose a two-stage, shape-based clustering approach that efficiently obtains a clustering of delivery request locations. Our solution technique is based on creating clusters that form certain shapes with respect to the depot. We obtain a routing solution by ordering all locations in every cluster separately. Our results are competitive with a state-of-the-art vehicle routing solver in terms of quality. Moreover, the results show that the algorithm is more scalable and is robust to problem parameters in terms of runtime.

Fish trawling can be considered as a vehicle routing problem where the main objective is to maximize the amount of fish (revenue) facing uncertainty on catch. This uncertainty creates an embedded prediction problem before deciding where to harvest. Using previous catch data to

train prediction models, we solve the routing problem a fish trawler faces using dynamically updated routing decisions allowing for spatiotemporal correlation in the random catch. We investigate the relationship between the quality of predictions and the quality of revenue generated as a result.

PUBLIC ABSTRACT

Data-driven decision making has become more popular in today's businesses including logistics and vehicle routing. Leveraging historical data, companies can achieve goals such as customer satisfaction management, scalable and efficient operation, and higher overall revenue.

In the management of customer satisfaction, logistics companies use consistent assignment of their drivers to customers over time. While doing so, companies trade off the cost of capacity with consistency because demand is unknown on a daily basis. We propose concepts and methods that enable a parcel delivery company to balance the trade-off between cost and customer satisfaction.

Parcel delivery companies route many vehicles to serve customer requests on a daily basis. Research has focused mainly on tailored heuristic approaches to generate routes. These approaches are not easily deployable and often do not have open source code bases. We propose a scalable clustering algorithm that efficiently obtains a clustering of delivery request locations. We show that our algorithm compares favorably to a state-of-the-art vehicle routing solver in terms of scalability and robustness.

Fish trawling can be considered as a vehicle routing problem where the main objective is to maximize the amount of fish (revenue) facing uncertainty on catch. This uncertainty creates an embedded prediction problem before deciding where to harvest. Using previous catch data to train prediction models, we solve the routing problem a fish trawler faces using dynamically updated routing decisions. We investigate the relationship between the quality of predictions and the quality of revenue generated as a result.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recent years has seen a huge surge in collecting information and data while business operations are carried out. This data-rich operating environment calls for data-centric approaches while conducting business operations. In this thesis, we propose data-driven solution methodologies for routing problems that utilize historical data to make better business decisions in the areas of parcel delivery and fish harvesting.

In the last decade, logistics companies have begun to consider the management of customer satisfaction in addition to cost-related considerations such as fuel, time, capacity, and fleet. This new focus is motivated in part by the success of the United Parcel Service (UPS) company. Some attribute this success to UPS drivers who "... form a real bond with customers and take tremendous ownership of their customers and routes" (UPS, 2007). Drivers are the first hand service provider for logistics companies such as UPS and FedEx. Forming desired customer relationships involves drivers consistently serving the same set of customers. Meeting these goals take time and depends on the history experienced between the logistic company and the customer. While doing so, companies trade off the cost of capacity with consistency because demand is unknown on a daily basis. The trade-off of consistency versus routing efficiency has mainly been studied from a static perspective. Effective policies for the management of customer satisfaction and traditional resource costs should be responsive to fluctuations in service requirements in a dynamic fashion as opposed to current approaches.

In order to increase customer retention, companies should consider developing a stable relationship between customers and drivers who are first-hand service providers to customers. Researchers have shown that companies that create satisfied, loyal customers have more repeat business (Smith, 2002, 2004). Customer relationship management is a way to express having the ability to organize and maintain a connection with clients, customers, and service agents (Clinton, 2011; UPS, 2007). Loyal customers develop expectations based on how they have received service in the past and what they should expect in the future (Bowersox et al., 2002). These expectations can create a habitual preference of the unique service provided by their drivers. This habitual preference is a function of the unique history experienced by each customer.

Customers that do not receive service that they expect to receive can be left dissatisfied. See

Figures 1.0.1, 1.0.2 and 1.0.3 for three complaints posted to www.consumeraffairs.com[1] describing the kind of service or lack thereof, some customers have received. From these complaints and many like them, we learn that it is not service time that influences customer satisfaction, but rather very peculiar things that a driver needs to know about each and every customer.

I've expected packages to be delivered, but I am working during the day, so when the driver left me a notice on the door, I did sign on the back and left an apartment number of my neighbors, so the packages can be delivered there. But when the driver came again, he did not leave packages with neighbors but left me notice again. I called UPS, spoke to 6 different people and I was told that they don't care about what is written on the back of a yellow UPS notice. It's up to the driver only if he wants to leave packages with neighbors or not and I was told that I should go to UPS location and get my packages myself. I do not drive and there is 4 packages for me to pick up. UPS provides the worst services ever.

Figure 1.0.1: UPS complaint 1.

I had an absolutely wonderful experience with UPS a few days ago. It was 4:30 on a Friday afternoon, and a considerate account executive with UPS happened to notice that something we were having shipped, due to the fact that we don't have an account with that specific company, was going to cost over $6000 for an item that only cost $200! Instead of just ignoring this, figuring that it wasn't her business, she actually came to our office to let us know. And since it just so happened that no one with any knowledge of this shipment was there at the time, she left her contact information, and was happy to help once contacted. Once it all was said and done, we were thankfully able to rectify the situation. Not sure what we would have done had she not gone out of her way- thanks, T.!!!
We also happen to have the most wonderful delivery driver on the face of this earth. The office that I work in is a relatively small, family-owned business, and many of the shipments that we get are quite time-sensitive (items that need to be kept cold, orders for customers, etc.). John ALWAYS goes out of his way to ensure that we get our orders as early as possible, is extremely friendly, and always considerate. Good job, UPS!!

Figure 1.0.2: UPS complaint 2.

Customers see the face of the driver hired by a courier service company when they demand service. This makes the driver-customer relationship important (UPS, 2007). Drivers learn customer preferences by way of this unique relationship. Companies encourage their drivers to meet these

[1] `www.consumeraffairs.com/delivery/united/parcel/service.htm` accessed on 10.12.14 at 11:20 pm.

My address is very clear on my box straight across from my house. Last Christmas they delivered my package 4 houses down. The neighbors brought me my package then I watched the driver drive past my house and took my package to someone else's porch which is far from my mailbox and the numbers aren't even close. Friday they were to deliver again and said on the tracking that it was left on my porch but no package. Lucky me that they delivered it to a porch that the person brought it and put it in my mailbox. Now another package to be delivered on the following Monday said on the tracking that they gave it to the woman customer on the property. Must of been a ghost because me and my husband was working at 2:16pm when it supposedly gave to me. I called them and I am sitting here waiting for a before 10am phone call from UPS and no call yet. From now on I will steer away from orders that will be delivered from UPS.

Figure 1.0.3: UPS complaint 3.

individual preferences for customer management purposes. For example, customers tend to have preferences regarding contingencies such as missed delivery. These preferences can range from leaving a package at a special location like the door or the porch, or even in the building's office or with a neighbor. As another example, packages might require the customer to be present at the time of delivery. Drivers can learn customer availability over time. If a missed delivery occurs, this means that the driver must revisit the customer again tomorrow, which consumes vehicle capacity for the next day and multiple trips have to be made for successful completion of service.

Current approaches to customer management are characterized by their simplicity. Researchers prefer to artificially impose constraints on assignments. Our research models the consistency problem in a "soft" way rather than imposing constraints (Groer et al., 2009), or exclusively assigning regions to drivers beforehand (Zhong et al., 2007). Dividing a service region at the strategic level necessitates making assumptions about relevant parameters including demand, customer location, and fleet maintenance issues. The strategy of designing service regions at a certain point in time is susceptible to daily variability. This is true for any type of constraints on assignments that in turn make the initial design less effective than predicted. This a priori approach can lead to undesirable situations for making day-to-day decisions in line with its long-term goals. By treating the problem dynamically and without artificial constraints, we are more likely to mitigate the adverse implications of such an a priori approach (imposing artificial constraints) caused by daily parameter fluctuations or unforeseen circumstances.

Collecting data for business purposes has become a recent trend in all industries. A major logistics provider such as UPS has invested heavily in technologies that accumulate information about every aspect of business operations. For example, in the early 2000's, UPS introduced the package flow technology (PFT) that enabled them to streamline vehicle dispatching with less error, higher cost savings and short term predictability of delay in package arrivals to distribution centers. Recently, UPS started using ORION which can be thought of as automated decision making based on data accumulated and provided by the PFT system (ORI, 2014). This new technology, whose future version is planned to be made real-time, creates routes based on optimization procedures. In

our case, to create stable driver-customer assignments in a dynamic fashion, we apply data mining methodologies and concepts from the clustering literature on accumulated service data. We use a dynamic model with no fixed routing or pre-clustering of customers and keep track of necessary information to assist in daily decision making. We quantify the degree of consistency a logistics company provides to their customers. In this light, our research can attract interest from logistics companies.

The main reason for the existence of artificially imposed structures is the scale of the problems facing parcel delivery companies. One method that can address the scale issue is the cluster-first, route-second heuristic in the VRP literature. In this method, the original problem is decomposed into smaller subproblems. Each subproblem turns into a traveling salesman problem (TSP). Existing state-of-the-art TSP solvers can easily manage problems this size: 100-200 nodes. In addition, the original problem turns into a generalized assignment problem (Fisher and Jaikumar, 1981). This original problem can benefit from the use of better mathematical programming techniques (Laporte, 2009). The benefit of such a formulation is that the clustering problem does not contain any subtour elimination constraints that usually complicate the optimization procedure. For day-to-day dispatching, we make use of such a benefit in a specific case, the 1-norm. 1-norm clustering can be done quite efficiently and we can obtain high quality solutions to the clustering problem.

Another focus in this thesis is clustering methods for large scale capacitated vehicle routing problems (VRP). Most effective approaches up-to-date have been to use metaheuristics. However, these heuristics are not easily deployable as they often require tailored implementations that cannot take advantage of or do not have available existing code bases. We propose scalable clustering mechanisms hypothesizing that we may obtain a better trade-off between runtime and solution quality. We propose a system that can be assembled using publicly available solvers.

Improvements in technology has benefited data collection in the business of fish harvesting as well. Catch and effort survey have been regularly conducted to properly assess fish stock abundance in the oceans. Moreover, the abundance of fish tends to exhibit spatial and temporal correlation.

Routing a fishing vessel can be considered as an orienteering problem in which a vehicle traverses a network to maximize its reward or expected reward. However, typically, an optimal solution is too hard to obtain. One solution approach to these types of problems is using approximate dynamic programming (ADP).

Rollout methods are a particular set of techniques used in ADP, which can deliver good but suboptimal solutions. Researchers have shown that it is beneficial to construct a routing solution in a dynamic fashion as information arrives; sticking to a fixed routing solution generated at the beginning of a decision horizon tends to be worse. The correct estimation of the information process, amount of catch in our case, enables the correct evaluation of routing solutions. We propose using spatiotemporal forecasting techniques to help the management of routing strategies for fishing vessels. We shed light on the relationship between the quality of a forecasting method and the quality of the routing solution obtained as a result.

In this thesis, we focus on scalable clustering techniques for vehicle routing, the multi-period

problem facing a courier service company with consistency goals, and the routing problem facing a fish trawler. We address the single period routing problem from a clustering perspective in Chapter 3. The consistency goal, which is essentially an assignment problem, links the routing solutions of different periods together. We model the consistency problem as a dynamic clustering problem in Chapter 4 focusing on the consistency issue where we do not treat the routing of customers. Furthermore, we model the relevant parameters for everyday decision making in a state variable, which we update at every decision epoch. Based on this state information, the courier service company faces a vehicle routing problem for that period. In Chapter 5, we model the problem that a fish trawler faces as a Markov decision problem. The state space in our model captures observations made along a fishing trip. We assume observations can be correlated across regions; therefore, keeping track of observations makes sense for rerouting strategies.

Our research makes several contributions to the literature:

- We propose a novel measure for customer relationship management that captures important features regarding frequency with which drivers visit customers in a lexicographic fashion.
- Our approach does not make restrictive assumptions on the solution space under uncertainty. As a result, this approach promises to be more effective.
- We propose a mathematical programming-based heuristic for the per-period routing decision that is conceptually simpler to implement than metaheuristics, which is the main approach used for practically-sized problems in the literature. Moreover, metaheuristics generally need to be fine tuned to accommodate unforeseen data realizations or make good quality solutions even better. For this reason, a mathematical programming approach can in general be more robust.
- We combine forecasting methods with several approximate dynamic programming heuristics in a rollout framework for routing fish trawlers. We take into account the possible correlation of catch across fishing regions and we validate our approach using real data.

In Chapter 3, we focus on the routing aspect of the problem at every decision epoch. We propose three different cluster-first, route-second algorithms and discuss strength and weakness of each depending on customer distribution.

In Chapter 4, we focus on the policies that can be used to create stable customer-driver relationships. We explore the effectiveness of two methods. In the first, we model consistency as a measure based on normalized mutual information from the data-mining literature. This measure is used for comparing clusterings and their similarity. Second, we use a more refined approach to consistency and model the consistency for every customer separately using information entropy. In these analyses, we do not conduct any routing but only group customers into clusters.

In Chapter 5, we detail a Markov decision model for the fish trawling problem and discuss our solution approach to our model.

Chapter 2

Literature Review

Our research is comprised of several components. First, this research belongs to the dynamic routing literature because information is revealed over time. Second, our approach uses a cluster-first, route-second approach where we take into account customer locations. Such approaches have a rich history in the literature. It is related to the partitioning or clustering techniques that are utilized in areas ranging from data mining, bioinformatics, mathematical programming, and heuristics. Third, routing (orienteering) and predictive modeling in the case of fish trawling.

The VRP literature has been extremely fruitful in terms of the number of published papers over 50 years. Many different variants for the VRP have been considered that consider specific characteristics of the problem such as: time windows, pickup and delivery services, heterogeneous fleet, uncertainty and so on. I will review the VRP related literature in Section 2.1. First, I will review one of the major themes of this research which is customer relationship management from a consistency point of view in subsection 2.1.1. Consistency involves decisions over time, so we review the dynamic vehicle routing literature in subsection 2.1.2. I will focus on large scale VRP in subsection 2.1.3.

Clustering is a very popular form of unsupervised data analysis. It can be considered as an overarching field that has applications in various fields of science and it is frequently used as a data analysis tool in research fields such as machine learning, data mining, artificial intelligence, and bioinformatics. I will review the clustering related literature in section 2.2. Consistency considerations and clustering can be considered as genres in a bigger picture of incorporating knowledge discovery into operations research.

2.1 Vehicle Routing Literature

2.1.1 Consistent Service to Customers

We summarize how the literature quantifies consistency in subsection 2.1.1.1. We compare and contrast our model of consistency with how it is modeled in previous work. Then, we move on to discuss related papers that have considered the consistency issue in subsection 2.1.1.2.

2.1.1.1 Quantifying Consistency

Consistency can serve the purposes of driver learning and customer satisfaction. Zhong et al. (2007) and Smilowitz et al. (2013) form their study focusing on driver learning which essentially affects the traditional objective of travel time. Specifically, Zhong et al. (2007) model how much consistent service decreases delivery completion time for vehicle drivers. This relationship exhibits diminishing marginal returns for every additional visit made by a driver to the same location. This type of a relationship is known as a learning curve (Hancock and Bayha, 1992). In a similar fashion, they present a forgetting curve that can be thought of as the symmetric counterpart of learning.

Smilowitz et al. (2013) use a different approach and one similar to ours. They evaluate the effect of various workforce management measures on traditional routing costs. These measures are collectively termed *workforce metrics.* They use two forms of such metrics: *customer familiarity* and *regional familiarity*. The functional properties they capture with these measures are based on driver learning curves as in Zhong et al. (2007). The general property of these measures is that visiting frequency increases driver learning at a diminishing rate.

We briefly describe their choice of modeling for creating consistency. Let T be the time horizon for planning. Define customer access cost to be the time between arriving at a customer location and successful completion of a delivery at that location. This cost is modeled as a convex function of the frequency with which a customer is visited. Let α^n denote the per visit customer access cost when a driver visits this customer n times within the time horizon, T. They assume that

$$\alpha^n > \alpha^{n+1} \quad \forall n \in T.$$

Moreover, they assume the condition

$$n\alpha^n \leq (n+1)\,\alpha^{n+1} \quad \forall n \in T.$$

Under these assumptions, solutions with fewer drivers visiting a customer are preferred. As a result, this leads to consistency.

On the other hand, we interpret consistency in the manner of Groër et al. (2009). We see consistency as a means with which customer satisfaction is provided. In this light, we propose information-theoretic measures to quantify the degree of stable customer relationship management. These measures can be seen as a 'soft' version and an extension of the hard constraints applied in Groër et al. (2009). These measures can be thought of as measuring how much more frequently a driver or group of drivers have visited a customer over a multi-period time frame. One benefit of these measures is that they capture the effect of the second most frequent driver and third most frequent driver and so on, aside from the most frequent driver. This is akin to a lexicographic ordering of the drivers assigned to customers. Such a lexicographic ordering may be beneficial because it may be necessary for more than one driver to have visited a customer over a multi-period time frame (this is the case at UPS). Allowing multiple drivers to visit a customer is related to paired vehicle recourse strategies. In these strategies, pairs of vehicles (drivers) are assigned to customers.

Our approach implicitly allows for such assignments where as in the literature the relationship between two vehicles is fixed. See section 4.2.1 for details.

2.1.1.2 Literature on Consistency in Vehicle Routing

The consistency feature in logistics problems has been studied by several papers from several pespectives. They can be characterized by whether or not these studies have explicitly modeled demand uncertainty. We first discuss studies with perfect information regarding customer requests. Lastly, we discuss those studies with stochasticity.

There has been work that studies consistency in a deterministic setting. One important study that addresses consistency is Smilowitz et al. (2013). They study consistent service, or "effective managerial decision making," under the setting of the periodic vehicle routing problem (PVRP). They quantify the value of consistency using two different metrics: *customer familiarity* and *region familiarity*. They model the trade-off between the traditional VRP objective of travel cost and consistency-enchanced driver productivity. It is important to note that their model corresponds to perfect information regarding future requests for delivery. In this regard, they use a periodic vehicle routing framework for their analysis. Perfect information is more likely to apply to short term planning whereas customer satisfaction with regard to consistency should be more appropriately interpreted as a long term benefit accrued by the customer. In contrast to their work, our work assumes future demand is unknown, and we focus on the customer's perspective for consistent service. However, similar to their paper, we model the trade-off in the objective, but use information-entropy related measures from the data mining literature.

Another study that deals with consistency in an environment with no uncertainty about customer demand is Groer et al. (2009). The authors incorporate constraints that make routing solutions consistent over driver-customer pairs and with respect to the time of service for each individual customer over multiple visits. They present an algorithm for their mixed-integer formulation which is used to solve large-scale simulated instances. Their algorithm is based on the record-to-record (RTR) travel algorithm proposed by Golden et al. (1998) and improved by Li and Olafsson (2005). Tarantilis et al. (2012) apply a template-based tabu search heuristic for the same problem. In these studies, the authors evaluate consistency based on the time differential between arrival times for service to a customer. Unlike these studies, our study does not incorporate consistency requirements as hard constraints. In this way, we can model the trade-off explicitly and optimize over gross consistency costs and gross travel costs.

The aforementioned studies are generalized by Kovacs et al. (2014). Under the setting of perfect information on future demand, they allow customers to be visited by at most a certain number of drivers instead of only one. They relax time-consistency constraints found in Groer et al. (2009) by modeling customer time windows as only AM/PM windows. They penalize the maximal arrival time difference in the objective to model the trade-off between time-consistency and travel cost. They use a flexible neighborhood search method to solve their model.

Our research differs from previously mentioned work because we consider uncertainty in fu-

ture demand, and we model customer preferences differently in a lexicographic fashion. Customer satisfaction is a long process such that future knowledge of business parameters like demand and customer location is not likely to be known beforehand. Therefore, a different consistency measure is needed. We propose such a measure.

Among the studies that deal with uncertainty while creating consistency in their model is Zhong et al. (2007) who obtain driver-customer consistency by assigning a set of customers exclusively to only one driver (core areas) and having different drivers serve unassigned customers (flex zones) across the planning horizon. See Figure 2.1.1 from their paper for illustrative purposes. Customers in these areas belong to a "zone" that has an exclusive driver serving them. The second goal of achieving routing efficiency is sought by having different drivers serve another set of customers. For this set of customers, the drivers serving them can change over time because it might be worthwile in terms of travel cost. Although this makes the trade-off partially rigid with respect to driver-customer assignments, the authors argue that it is a good balance of driver familiarity and optimization flexibility. Their model explicitly includes an analytical model for driver learning that is incorporated in a multistage stochastic mathematical program. Additionally, their model takes into account workload balancing considerations and models them as constraints. They evaluate how the number of drivers and total service time is correlated while consistently assigning drivers to customers. They compare their model against a method with no use of core areas, i.e. no customer is assigned to a core area.

Zhong et al. (2007) show that the model with core areas is better than their model with no core areas in terms of creating consistency. This result is counterintuitive in that a model with a restricted solution space delivers better trade-off in terms of consistency and travel cost than a model with no such restriction. If we analyze how they set up their experiments, we see that their solution approach is simple for the no-core-area method. First, they compute routes without considering the effect of driver learning, and second, they assign drivers to routes based on driver learning performance. The driver learning effect guides the solution obtained at this second step, which is clearly suboptimal because it was disregarded when constructing routes. Moreover, the initial construction of the routes are carried out using insertion heuristics.

A more effective approach should optimize over both objectives together. Our framework is flexible while keeping track of necessary information for consistency over time and uses this information in day-to-day operational routing decisions. This should deliver a better trade-off between flexibility and consistency requirements. Customer zones emerge implicitly from our methodology, but at the same time, our method is flexible enough to allow the clustered customers to change over time. In our case, these groups of customers are identified using historical data instead. In simplistic terms, we allow for data to speak for itself as it is realized over time, rather than make probabilistic assumptions about it during an initial strategic design phase. We refer the reader to section 2.1.1.1 where we compare and contrast our approach for modeling consistency to other work.

A second study that models consistency, although indirectly, is Carlsson (2011). He considers a territory planning problem for uncapacitated vehicle routing problems. He presents an algorithm

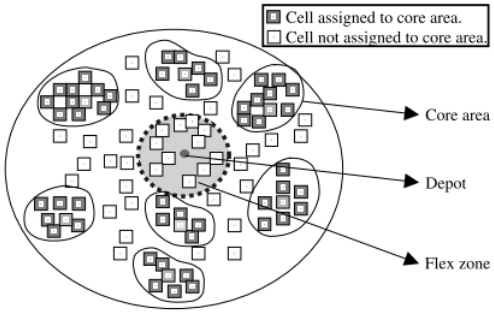


Figure 2.1.1: Core areas in Zhong et al. (2007).

for dividing a region when customer distribution is independently and identically distributed. Additionally, the partitions are devised such that wokload balance is maintained. His analysis is more apt for strategic decision making in logistics companies.

2.1.2 Dynamic Vehicle Routing with Uncertainty

The classical vehicle routing problem has variants that take into account several features of our problem. There is the capacitated VRP (CVRP), where vehicles have finite capacity; the VRP with Time Windows (VRPTW), where requests for delivery need to be met within a specific period of time; the VRP with Pick-up and Delivery (PDP) where customers request delivery and goods to be picked-up, and many other variants. In contrast to the previously mentioned variants for the VRP, uncertainty is an additional important feature for real-life applications of the VRP. Uncertainty is inescapable in today's world with the rapid growth of e-commerce and the demand for courier services in almost real time with same day delivery, and/or second day delivery promised by courier service companies. Research has focused on analyzing the effect of uncertainty in three cases: stochastic customers, stochastic service times, and stochastic demand. In our model, future customers have a bearing on the consistency value of the routing decision we make today, so the dynamic nature of the problem is important in this regard. The literature has addressed the dynamic dimension using several approaches.

There are two main solution approaches to dynamic vehicle routing with uncertainty. Within the a priori solution approach, fixed routing implies every customer will belong to a predetermined route that is constant over the planning horizon. Semi-fixed routes is when vehicles do not visit customers who do not request service, and this leads to a certain level of cost savings. Second, there is the dynamic or variable solution approach where routing decisions are updated based information that has been revealed thus far. Powell et al. (1995) and Thomas (2011) are reviews in this field. We would like to emphasize that our problem is stochastic as well as dynamic.

Fixed routing was first studied by Christofides (1971). A variation to the classic savings algorithm, which relies on the concept of savings gained by joining two routes into one, was applied by Beasley (1984) to the fixed routing problem. As noted in these references, fixed routes can lead to capacity-infeasible solutions or underutilized routes because of customer demand variation over

time. Other methods, like Zhong et al. (2007); Wong and Beasley (1984), construct fixed areas of service rather than fixed routes. Waters (1989) studies uncertainty in the set of customers that need to be visited in a delivery period. It compares the effectiveness of fixed routes, semi-fixed routes, and totally variable routes in terms of the cost savings only. Waters (1989) concludes that the consideration of variability in the routes depends on the number of customers omitted in a single delivery period. While using a fixed-route strategy enables dispatchers to set time of delivery for customers and driver familiarity, it can lead to delivery shortages because one vehicle always serves the same group of customers who can sometimes have higher demand than vehicle capacity. Haughton and Stenger (1998) and Haughton (1998) address this issue, estimate the expected shortage faced with fixed routing and model fixed routing with stochastic customer demand. Haughton (2000) analyzes the benefits of reroute optimization. In Savelsbergh et al. (1993), the authors compared the fixed-routes scheme with a variable-routes scheme and under what conditions the former would be better to use. Their model generates fixed route lengths within 10% of the routes generated by variable routing. This result was true for coefficients of variation smaller than 30% of customer demand. Beasley and Christofides (1997) looks at an an application of fixed routing. A mail order company preferred having fixed routes. The graphical representation of the problem involves sparse customer linkages. The authors analyze the problem using computational geometry concepts to develop an effective heuristic.

Other closely related classes of problems are the Probabilistic Traveling Salesman problem and the Stochastic Vehicle Routing problem which can be viewed as fixed routing strategies under uncertainty. There are survey papers that treat these problems and the related literature (Psaraftis, 1988, 1995; Campbell and Thomas, 2008a).

Variable/Dynamic routing involves rescheduling as information is revealed. One important early paper in the stochastic dynamic routing literature is Bertsimas and van Ryzin (1991) where the authors analyze a variation of the VRP called the dynamic traveling repairman problem. Their policies minimize the average time a requested demand spends waiting for service. They approach the problem from a queuing theory point of view. In Bertsimas and Van Ryzin (1993), they extend and improve their initial study to consider multiple vehicles with and without capacity restrictions. Again, from a queuing theory perspective, their results show that system time is reduced by a factor of the square of the number of vehicles in the system.

Powell (1996) is one of the first papers that treated a dynamic vehicle routing problem with stochastic elements where the future impacts today's decisions. Recent work in dynamic routing with stochastic demand include Secomandi (2000) who investigates an approximate policy iteration procedure that estimates the cost-to-go in period t using a parametric function. Secomandi (2000, 2001) proposes a single fixed-route policy and repeatedly uses it to approximate the cost-to-go for a given action by computing the expected cost of cyclically following this fixed route. A more recent study by Goodson et al. (2012) use a cyclic-order neighborhood structure to discover high quality a priori solutions. Their paper contains an up-to-date extensive review of the related literature for VRP problems with stochastic demand. Another study where demand is stochastic is Meisel

et al. (2009), where the authors construct linear value function approximations for a dynamic and stochastic routing problem with a single, uncapacitated vehicle. Bent and Van Hentenryck (2004) employ a multiple scenario approach to solve a general dynamic VRP with stochastic requests. Hvattum et al. (2007) present a branch-and-regret heuristic for problems with stochastic customers, stochastic demand, or both. Hvattum et al. (2006) solve sample-based problems heuristically whose solutions are combined to form a solution to an overall problem with stochastic demand and stochastic customers. Mendoza and Villegas (2013) sample the solution space to find high quality solutions. Sörensen and Sevaux (2009) develop a sampling-based approach that delivers robust and flexible solutions to stochastic variants of the capacitated vehicle routing problem. Ichoua et al. (2006) present a threshold-based tabu search heuristic that accounts for future customer requests. Gendreau et al. (2006) propose neighborhood search heuristics where new requests for pickup and delivery occur real-time. A recent survey of dynamic vehicle routing problems can be found in Pillac et al. (2013). Another recent survey by Berbeglia et al. (2010) focuses on studies about dynamic pickup and delivery problems.

Research focusing on real-life applications and other variants of the vehicle routing problem wih uncertainy have also been considered. Mendoza et al. (2011) develop constructive heuristics for a multicompartment vehicle routing problem with stochastic demand. Mendoza et al. (2010) model the same problem as a stochastic optimization problem with recourse and use a memetic algorithm to obtain solutions. Tirado et al. (2013) address a dynamic stochastic maritime problem where they propose three different heuristics to test the hypothesis that the inclusion of stochastic information within a dynamic planning process is beneficial. Agra et al. (2013) address the robust vehicle routing problem with time windows. Essentially, they optimize a stochastic objective assuming the worst-case scenario for stochastic time windows. Hvattum et al. (2008) develop heuristics based on scenario trees for the stochastic inventory routing problem. Azi et al. (2012) propose adaptive large neighborhood search for a dynamic vehicle routing problem where each vehicle performs delivery over multiple routes in a week. Tagmouti et al. (2011) describe a dynamic arc routing problem motivated by winter gritting applications where service time intervals are updated after weather report updates.

Online algorithms are optimization routines whose data are revealed during algorithm execution. For instance, data can be new customer requests. Van Hentenryck et al. (2010) propose online stochastic algorithms that incorporate knowledge of future requests. It does so by sampling future requests from uncertainty models dependent on realized current data. Van Hentenryck et al. (2009) adapt their framework to online stochastic reservation systems, in particular the online stochastic multi-knapsack problem. Lorini et al. (2011) extend Potvin et al. (2006) to account for communication between drivers and depot when new customer requests are made.

Policies (solutions) that reflect the possibility of a customer request are said to *anticipate* a request from a customer. Thomas (2007) develops waiting strategies for anticipating future customer requests while maximizing the expected number of customers served. He derives analytical results on the structure of the optimal policy given a priori route selection. Thomas and White (2004)

present optimal policy results in a single pickup and delivery vehicle routing problem where the likelihood that each of the vehicle's potential customers will make a pickup request is known. They show the superiority of an optimal anticipatory policy relative to a reactive route design approach. Ghiani et al. (2011) compare an anticipatory solution approach to sample-scenario planning, which is a sampling based routing technique that is computationally expensive relative to the anticipatory approach they propose.

Benson (2001) explores innovations enabled by package tracking technology. He developes a dynamic load selection problem associated with the uncertainty involved in incoming packages to a local UPS facility and the optimal way to allocate packages onto delivery trucks. He describes a prototype decision support system developed for UPS that identifies improvements made possible with the tracking technology. Our methodology addresses customer management concerns of companies like UPS, so it can be part of such a decision support system.

2.1.3 Very Large Scale Vehicle Routing

In our model, the dispatcher faces daily large-scale routing decisions. The parameters for this daily problem are assumed to be fully known to the dispatcher. However, there is no straightforward solution approach. The VRP is known to be a NP-hard problem as shown in Lenstra and Kan (1981). This fact makes it difficult to solve the problem to optimality. Exact methods that find optimal solutions can only tackle problems with at most 100 nodes, or customers. A recent survey of exact methods can be found in Baldacci et al. (2010).

Given the large size problems of interest in this paper, practical methods have to be developed that deliver efficient routes within a reasonable amount of time. Researchers have mainly focused on using metaheuristics for this purpose. This approach can be categorized based on the type of method used. These range from evolutionary/memetic algorithms to tabu search to various local search and neighborhood search algorithms. Our work does not fall into the category of metaheuristics. We review some of the important studies in this area for completeness.

Evolutionary algorithms are a class of nature-inspired intelligence used for hard decision problems. Within this domain, genetic algorithms have become the most popular. These type of algorithms can be roughly broken down into an initialization phase, a selection phase, a crossover phase, a mutation phase and a replacement phase. These phases represent operations on candidate solutions to a problem inspired by their biological counterparts. If you consider combinatorial problems like the VRP which is NP-hard, the solution space needs to be explored well enough in order to obtain as effective a solution as possible. Memetic algorithms are designed for this purpose and combine a local seach phase within a genetic algorithm to be more effective. One important contribution in this field is Nagata and Braysy (2009). They present a memetic algorithm that includes a new local search operator, the edge assembly crossover technique. Their experiments at the time obtained best-known solutions for 28 benchmark problems and new best-known solutions to 12 instances. Mester and Bräysy (2005) present an active-guided evolutionary algorithm where they combine the strengths of guided local search with evolutionary strategies into an iterative two-stage metaheuris-

tic. On a benchmark set comprised of 302 problems, they obtained best-known solutions for 86% of the problem instances at the time within reasonable computation times. Marinakis and Marinaki (2010) have obtained good results using a hybrid genetic-particle swarm optimization technique. There are numerous ways in which researchers have designed genetic algorithms combining various strategies used in other papers. Less recent work are Berger and Barkaoui (2003); Prins (2004); Baker and Ayechew (2003). For a more detailed list of work, see Golden et al. (2008).

There are other techniques that search the neighborhood of a solution to a problem. Marinakis (2012) propose a new multiple phase neighborhood search GRASP algorithm for the CVRP. GRASP stands for Greedy Randomized Adaptive Search Procedure and it has been shown to be quite efficient for the Traveling Salesman Problem. Chen et al. (2010) propose a hybrid heuristic with variable neighborhood descent based optimization. They report competitive results compared to state-of-the-art heuristics. Improvement heuristics are interative procedures that enhance a feasible solution. Kytöjoki et al. (2007) use variable neighborhood search to guide a collection of improvement heuristics which are iterative methods that enhance a feasible solution.

Tabu search is one of the most effective and well known tools to attack large-sized VRP. Cordeau and Maischberger (2012) use a version of the tabu search heuristic for the VRPTW, SDVRP and MDVRP. Their methodology gets very close to best known solutions in the literature. There has been work on parallel algorithms that operate in a simultaneous fashion to find better routes. One such work is Jin et al. (2012) where the authors present a parallel tabu search algorithm that utilizes different neighborhood structures in tabu search threads which cooperate through a solution pool. Their work is competitive with other studies that have obtained best solutions with an average deviation of 0.22% from the literature's best results. Relatively older studies in this area are Gendreau et al. (1994); Xu and Kelly (1996); Barbarosoglu and Ozgur (1999).

The literature has focused on other heuristics such as simulated annealing and ant colony optimization. For a review of methods before 2000, see surveys by Laporte et al. (2000a) and Cordeau et al. (2002).

Our research approaches the difficulty caused by problem scale by using a clustering approach. The clustering approach is a natural way to think of the consistency problem, which essentially depends on which customer gets assigned to which driver. Additionally, this approach is conceptually simpler to implement than metaheuristics in an industrial setting. Most of the metaheuristics are developed for generic problems whereas our problem contains a multi-objective function that calls for design decisions. In order to better optimize over the consistency decision, we model it as an integer programming problem as opposed to using metaheuristical procedures to balance the trade-off between traditional business goals and that of consistency. Moreover, metaheuristics generally need to be fine tuned to accommodate unforeseen data realizations or make good quality solutions even better.

2.1.4 Stochastic Orienteering and Fish Trawling

Orienteering problems are class of problems closely related to vehicle routing problems. Roughly speaking, routing is carried out to maximize total reward in orienteering as opposed to minimizing cost in vehicle routing problems.

The closest study to the problem presented in Chapter 5 is Ilhan et al. (2008). The authors consider the problem of finding a tour that visits a subset of nodes on a graph within a prespecified time limit and maximizes the probability of collecting more than a certain threshold level of profit. Each node is associated with a random profit (reward) that is normally distributed. The authors use a bi-objective genetic algorithm to solve their model. In addition to the fact that our problem does not make normality assumptions on the rewards, we make dynamic decision on a graph with correlated rewards between nodes.

There are also other studies that consider orienteering problems with stochastic features. Campbell et al. (2011) introduce a problem with stochastic travel and service times. The reward function in their work is related to whether or not a node can be visited by a deadline. As a result, stochastic travel and service times lead to uncertainty in the total reward that can be accrued. They present computational results and characterize optimal solutions under certain assumptions. Zhang et al. (2014b) present a problem with stochastic wait times that applies to the pharmaceutical industry. They employ an a priori planning strategy with certain recourse actions and derive the expected total expected reward analytically. With this analytical evaluation of the objective possible, they use a variable neighborhood search heuristic to solve their problem. A more recent work by the same authors focuses on dynamic orienteering on a network of queues (Zhang et al., 2014a). Evers et al. (2014) approach a similar problem with stochastic weights on the arcs between nodes on a graph using a two-stage stochastic model with recourse. In these studies, correlated rewards are not explicitly considered and they employ a priori solution approaches under independence assumptions on the random variables.

Next, we focus on studies that develop decision models for the fishing industry specifically. These studies use dynamic programming and Markov decision problems similar to our modeling technique.

Various studies have proposed dynamic models of decision making. Babcock and Pikitch (2000) present a model where bottom trawlers target different species under management-imposed limits on landings for each species. The paper focuses on the effect of trip limits on the choice of fishing strategy in terms of targeting bottom rockfish or deepwater Dover sole. Dorn (2001) focuses on the sequential decision making process for a fleet of vessels based in Seattle, WA, operating in the Pacific hake fishery. By modeling the dynamics for the Pacific hake fishery, the author develops a dynamic sequential decision making model in terms of a Markov Decision Process. The author models the sequence of decisions which include what threshold prey density should be used to decide to leave an area, how observations can be combined into the decision to harvest an area, and how much time should be spent searching an area before deciding to leave or not. It is assumed that areas exhibit independent population dynamics unlike a correlated reward structure such as

ours. Moreover, the author assumes the vessel moves only to adjacent areas if it decides to fish in a different area. We do not have such restriction. In order to model the decision to move to a new area, the author updates the mean catch rate of all areas according to a Kalman filter update process from the observed catch rates. The Kalman filter is an update process that combines previous observations with new ones. Since areas exhibit independent dynamics, the Kalman filter is applied to each area independently, hence there is no correlation assumed to exist between mean catch rates between areas. The author uses threshold based decision rules to move to a new area: if the abundance, measured in mean catch rate, is lower than a threshold, the vessel leaves for the adjacent fishing area. The same rule applies to start fishing in an area after searching produces an estimate for an area. These decision rules are justified by way of the marginal value theorem in optimal foraging theory. Dorn (2001) carries out numerical search over parameters that govern the decision rules previously discussed to find the combination of parameters that maximize daily revenue using simulation. His results suggest that the reward surface is flat in the parameter neighborhood of the maximum reward. When new observations have a relatively lower weight than previous observations in updating the state, then a high catch rate threshold is a poor strategy because the vessel is always seeking high density areas, but never believes information (observations) that indicate a high mean catch rate. When new observations have a relatively higher weight than previous observations in updating the state, then a low catch rate threshold does not substantially reduce the average reward because the vessel does not act upon information from fishing and searching until it indicates that fish density is very low.

2.2 Knowledge Discovery and Clustering

There has been growing interest in incorporating data mining (knowledge discovery) techniques into the Management Science/Operations Research field. Every operations research problem has an underlying set of information that is valuable for secondary analysis and business decision support for future planning (Meisel and Mattfeld, 2010). Each time information is updated, the decision making process can account for it and update decisions if necessary. Furthermore, it is valuable to analyze data for pattern extraction to support this process. Data mining has a collection of methods to do just this. The application fields are very diverse, ranging from manufacturing to finance to logistics to marketing. A good general approach is discussed in Meisel and Mattfeld (2010). In our case, in order to be able to provide consistent service to customers, we use concepts for computing similarity between clusterings based on the historical values of past assignments.

The concepts and methods of knowledge discovery have been applied in numerous fields. In manufacturing, Li and Olafsson (2005) use decision tree induction, which is a supervised learning algorithm, to create schedules for single machine production environments. Based on mined knowledge, they derive rules for dispatching jobs. In Shao et al. (2006), the authors use an association rule algorithm to determine alternatives for bicycle production given customer attributes. In Tseng et al. (2006) and Lin (2009), the authors apply data mining to supply chain supplier selection based on

supplier attributes. An application to fleet management maintenance was carried out in Sawicki and Zak (2009) where they derive the minimum number of attributes to be able to assess the condition of vehicles in a fleet. Cooper and Giuffrida (2000) study a decision problem on the amounts of stock to hold for various products in a retail store setting. In Greco et al. (2002), decision rules for company financing are developed. They suggest using a dominance-based rough set approach for rule determination. In the area of health care management, Delesie and Croes (2000) apply data visualization and present a similarity measure of insurance reimbursements for medical procedures. The general idea in this line of work is to find relationships in the data that are not normally assumed or are unknown, and make use of it in the business process.

2.2.1 Clustering in Data Mining

From the data mining perspective, clustering is a grouping of data objects based on a similarity measure. Data objects in the same group are deemed to be more similar than to other objects in different groups. Clustering can group data objects in various ways. The most common categorization of a clustering method is whether the clustering method is hierarchical or partitional. In hierarchical clustering, clusters are nested within each other. However, in partitional clustering, clusters divide the set of objects into non-overlapping mutually exclusive subsets. Our focus in our research is to use partitional clustering techniques.

Clustering aims to find useful groups (clusters) of objects, where usefulness is defined by the goals of the data analysis (Tan et al., 2005). Usefulness of a clustering depends on the specific objective or goal for the data analysis. A researcher may be interested in finding prototype-based clusters where each cluster is represented by a center. A researcher may be interested in graph-based clusters for data represented as a graph or a researcher may look for densely populated regions of objects in the dataset. In a vehicle routing setting, the type of clusterings to look for are more likely to be prototype-based clusters akin to the clustering techniques in the VRP literature. We employ such a perspective. There are other categories of clustering algorithms including density-based clustering (Ester et al., 1996; Ankerst et al., 1999) and grid-based clustering (Wang et al., 1997).

2.2.1.1 Protoype-based Clustering

In prototype-based clustering, each cluster is represented by a prototype and all members of a cluster are closer to their respective prototype than they are to prototypes of other clusters. Prototype-based clustering contains center-based clustering techniques such as the k-means algorithm and its variants. Other prototype-based clustering techniques include mixture model clustering and kernel clustering. We review each type for completeness and talk about their relation to our research.

The most conventional prototype-based clustering technique is k-means clustering (MacQueen, 1967). The basic k-means algorithm alternates between assigning each point to its nearest center and updating centers for each cluster. The key part of this method is the distance measure used to compute the distances between a point and a center. Researchers have used various distance

functions such as the Manhattan (1-norm), Euclidean, cosine, and Bregman distance functions. Each of these functions produce a cluster with a different shape. From an optimization perspective, this problem falls in the domain of mixed nonlinear integer programming. The basic k-means algorithm can be considered as an iterative improvement scheme for this mixed nonlinear integer program. We apply a similar iterative solution approach for our clustering problems in Chapter 3.

The type of clusters the k-means algorithm delivers, which would be a group of customers in a VRP setting, tend to be biased towards having a convex shape. See Figure 2.2.1. In addition, the natural number of clusters that exist in the data may or may not be equal to the parameter, k, that is set by the user. Some important properties of k-means are: (i) it is efficient in clustering large data sets, (ii) its performance depends on the initialization of centers, since it can converge to bad local optima, and (iii) it works on numerical data. It is not suitable for categorical data. For such data, there is a variant of k-means called the k-mode algorithm that can be used, but categorical data does not apply to our study. The variants of the k-means algorithm such as the Sort-means algorithm, k-modes algorithm, k-medians, and k-harmonic means algorithm are discussed in Gan et al. (2007).

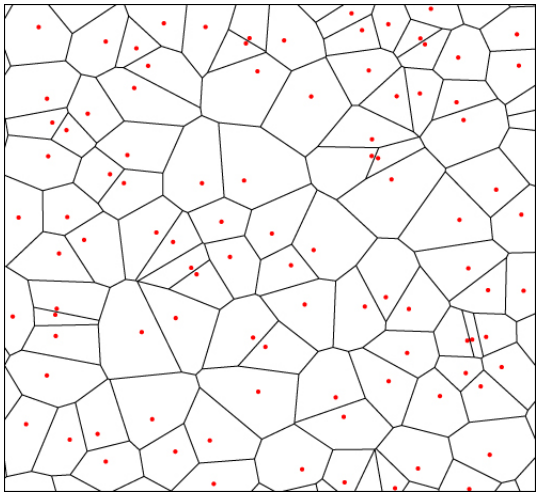


Figure 2.2.1: Convex partitions: red dots as prototypes.

Prototypes of a cluster can be statistical distributions. This type of a clustering approach is called mixture model clustering. Researchers use a dataset to estimate parameters of the statistical distributions they try to fit the data into. For example, Gaussian mixture models generate data from finitely many normal distributions that capture ellipsoidal clusters in the data. General introductions to mixture model clustering can be found in Fraley and Raftery (1998) and Mitchell (1997).

Estimating parameters of each distribution within a mixture model (estimating parameters of a cluster) is carried out using statistical procedures. Maximum Likelihood Estimation (MLE) is a popular technique to use for this purpose. MLE chooses from a set of parameters those that maximize the probability that the sampled data could have been generated from.

Consider a univariate model where data consists of observations of a scalar variable. So the sample consists of scalar numbers that may have been generated from a mixture of models. Let us hypothesize that the data comes from univariate normal distibutions and we know which point is

generated from which distribution. This means that the parameters of concern are the means μ_i and variances σ_i^2 that belong to a set Θ. Let there be m observations in the sample. Let x_i be the \$i\$th observation of the scalar variable that belongs to a set X. We can write the likelihood expression in general as

$$likelihood\,(\Theta|X) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x_i-\mu_i)^2}{2\sigma_i^2}}. \tag{2.2.1}$$

By applying a logarithmic transformation, we obtain the log-likelihood as

$$log-likelihood\,(\Theta|X) = -\sum_{i=1}^{m} \frac{(x_i-\mu_i)^2}{2\sigma_i^2} - 0.5mlog\,(2\pi) - mlog\,(\sigma) \tag{2.2.2}$$

and we may view this as a function of Θ where the givens are $\{x_i\}_{i=1}^{m}$. By finding the parameters that maximize this expression we find the maximum likelihood estimators.

Since in most cases we do not know which points are generated by which distribution, researchers use the Expectation-Maximization (EM) algorithm. This algorithm is interesting also because it can be regarded as a generalization of the k-means algorithm. See Algorithm 1 for an outline of the method.

Algorithm 1 Expectation-Maximization (EM) algorithm

Guess initial parameters.
while (parameters change)

1. Expectation Step: For each sample point, compute the probability that it belongs to distribution k.
2. Maximization Step: Given probabilities from previous step, update the parameters that maximize the expected likelihood.

end

Mixture model clustering is not suitable for our purpose because points are assumed to belong to more than one cluster. Additionally, in a vehicle routing context, a mixture model would entail information regarding the probability of each customer belonging to a different route based on a vehicle routing objective which is more complicated than relatively simpler objectives like the maximum likelihood function.

It can be the case that the form of the distance function between data objects is not known. In such a case, prototype-based clustering is not straightforward because a cluster prototype cannot be evaluated without a distance function. Kernel and spectral clustering overcome this challenge. In kernel clustering methods, the original input space of the data objects is mapped into a different space which we may call the feature space (Schölkopf et al., 1998; Girolami, 2002). By application of the kernel trick (Girolami, 2002), one can apply the iterative k-means like algorithm in this feature space. In spectral clustering, pairwise distances (the similarity matrix) is known or computed. Based on this similarity matrix, a special kind of matrix is computed. These are graph Laplacian matrices

(Chung, 1997). One computes the first k eigenvectors of such a matrix. Clustering can be carried out on the rows of the eigenvector matrix that induces a partition of the original data objects (Shi and Malik, 2000). Dhillon et al. (2004) show that these two approaches can be unified under a single framework.

Spectral clustering has one major drawback. The computational burden of finding eigenvectors can be significant, especially for large-sized problems (Dhillon et al., 2004). When you consider kernel clustering, the choice of the kernel function is a design decision. Moreover, the transformed problem in the feature space is an assignment problem with a nonlinear objective. This nonlinear objective is the summation of squared-Euclidean norms. It is equivalent to a series of inner norms. Such an objective is hard to optimize compared to say the 1-norm. However, a kernel function necessitates using an inner product norm in the objective and we cannot get around this. So, in our case, optimizing over a clustering objective plus a consistency objective would be extra challenging. The positive side of kernel clustering is that it allows one to look for clusters with various shapes depending on the choice of the kernel.

2.2.1.2 Graph-based Clustering

A general vehicle routing problem can be viewed as a graph-based problem as well. We can view the customers needing service as nodes on a graph and an arc between every pair of customers with arc weight equal to the distance between the pair of customers. This corresponds to a complete graph. The optimal routing solution is one of many feasible routing solutions in this complete graph. Such an optimal routing solution results in grouping the nodes into several clusters, or in graph theory terms, cliques. For this reason, graph-based clustering techniques can be relevant, in particular the popular hierarchical clustering.

The pairwise distances between customers or any other similarity matrix can be used to carry out graph-based clustering. Hierarchical clustering is one of the most widely used graph-based clustering techniques. There are two basic approaches in hierarchical clustering. In the agglomerative approach, in its most basic form, one assumes as a starting point that each individual point represents a cluster and successively merges two clusters until one cluster remains. In the divisive approach, one starts with an all-inclusive cluster and sucessively splits every cluster until only singletons remain. The clusters formed depend on how the similarity (proximity) matrix is defined. Some ways of computing cluster proximity are using the *single link*, *complete link*, and *group average* method. *Single link* computes the proximity between two closest points in different clusters. *Complete link* computes the proximity between two farthest points in different clusters. *Group average* computes the proximity of two clusters as the average pairwise proximities between two clusters. Figure 2.2.2 illustrates these three measures and a nested clustering. While being simple to implement, hierarchical clustering is not a global procedure for clustering. Clusters are formed in a greedy fashion and are thus susceptible to lack of global optimization.

When working with a similarity (proximity) matrix, it can be beneficial to sparsify it. Sparsification refers to setting low-similarity values to zero in a proximity matrix using a specified threshold.

In this way, data size is reduced. Clustering can focus on stronger relationships, thus making the clusters less susceptible to noise.

Another graph-based clustering technique is Minimum Spanning Tree (MST) clustering. It is a divisive hierarchical clustering technique that is used to detect clusters with irregular boundaries. In this technique, first a minimum spanning tree is computed from the similarity matrix. Clusters are formed by breaking links with the largest dissimilarity in this minimum spanning tree and repeating until all clusters are singletons. Wang et al. (2009) propose a divide-and-conquer approach to MST clustering. Their method has a much better performance than standard algorithms with $O\left(n^2\right)$ complexity where n is the number of nodes. Laszlo and Mukherjee (2005) provide an MST partitioning algorithm with group size constraints for microaggregation. Microaggregation is a disclosure limitation technique to protect individual records in datasets.

CHAMELEON is a popular agglomerative form of hierarchical clustering algorithm. It answers several problems with regard to cluster proximity measures. First, the single-link like measures may be misleading in terms of how close two clusters really are. The minimum distance between points in different clusters can lead one to think that two clusters are close when infact they are not so similar considering other points in these clusters are. Second, clusters in a clustering do not necessarily have to have the same characteristics in terms of size, shape, and density (Karypis et al., 1999). Li et al. (2009) develop a hybrid Chameleon clustering algorithm for faster execution and better accuracy.

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is another scalable hierarchical clustering technique for data in an Euclidean space (Zhang et al., 1996). BIRCH was specifically designed for computational efficiency and the authors show that it is superior to CLARANS, which is another clustering method for large-scale data. BIRCH can deliver clusters in a single scan of the dataset and improve the quality of the clustering in additional scans. BIRCH deals with outliers effectively.

CURE (Clustering Using Representatives) is another algorithm that falls in the same group as CHAMELEON. CURE combines protoype-based clustering and hierarchical clustering. Each cluster is represented using multiple representative (prototype) points whose locations are shrunk towards each other by a factor. CURE is designed to be scalable, more robust to outliers, able to identify clusters with non-spherical shape, and clusters with wide variances (Guha et al., 1998).

Hierarchical clustering in the context of grouping customers for routing is not straighforward. It is intuitive to think about customers that are close to a certain customer, i.e., its neighbors. The information that hierarchical clustering can provide for every customer in terms of their neighbors may likely be valuable, but it does not appear to us how such information can be used, especially for large-scale problems where a clustering hierarchy is likely to be deeply nested. Although focusing only on prototype-based clustering does not make use of such information, specialized local search techniques for vehicle routing problems are likely to be more beneficial with regard to neighborhoods of customers.

In our research, we explore suitable clustering methods that can be used to create efficient group

of customers in a cluster-first, route-second heuristic. We refer the reader to Chapter 3.

2.2.2 Clustering in Vehicle Routing

It is important to implement algorithms that obtain efficient solutions in a short period of time especially for recent trends of same-day delivery. Vehicle dispatchers face large scale vehicle routing problems every day. They need to make decisions or test algorithms and receive feedback as quickly as possible. The cluster first, route second heuristic is a well-known general approach for practical-sized vehicle routing problems and variants. We use this approach in our research.

The initial paper that proposed the cluster first, route second approach to vehicle routing is Fisher and Jaikumar (1981). Their heuristic breaks the CVRP problem into two stages. The first stage is a nonlinear assignment problem whereas the second stage consists of separate traveling salesman problems independent of each other once a solution from the initial stage is chosen. Fisher and Jaikumar (1981) present an algorithm that depends on a choice of seed customers that need to be chosen to obtain a solution for the initial stage. They note that the effectiveness of their method is dependent on these initial seed customers. The seed customers can safely be interpreted as centroids as described before. Bramel and Simchi-Levi (1995) use the same approach, but the choices of seed customers are determined through the course of the algorithm. Their heuristic involves solving a capacitated concentrator location problem (CCLP). The CCLP produces an assignment decision in terms of which customer is assigned to which driver. Once the assignments are known, routing customers within each 'cluster' can be carried out separately. The authors solve the CCLP using a Lagrangian relaxation technique by penalizing a set of constraints. Although asymptotically optimal, their heuristic empirically cannot compete with other heuristics Laporte et al. (2000b). Moreover, the Lagrangian relaxation-subgradient method can take a significant amount of time to converge for large instances. The authors extend their approach to vehicle routing problems with time windows in Bramel and Simchi-Levi (1997, 1996).

Our approach is in a similar vein. We are interested in creating solutions with an eye towards customer satisfaction, i.e., consistency. The degree of consistency of a given solution is a matter of the assignment decision. We therefore think a matheuristic is ideal to maximize consistency while making routing as efficiently as possible. A matheuristic is a heuristic that incorporates mathematical optimization models for better optimization (Doerner and Schmid, 2010). Specifically, we combine the consistency objective into a clustering problem that is a quadratic programming problem.

The cluster first, route second heuristic has been applied in other types of problems, too. Guerrero et al. (2014) present a decomposition technique for a two-echelon inventory-location-routing problem. Their heuristic decomposes the problem into a facility location, inventory decision, and routing problem. Federgruen and Zipkin (1984) use the technique in a combined vehicle routing and inventory allocation problem. Campbell and Savelsbergh (2004) present a inventory routing problem solved by creating delivery schedules in the first stage, followed by routing heuristics in the second stage. Prins et al. (2007) propose a cooperative metaheuristic that alternates between

solving a facility location problem (first stage) and routing (second phase). The first stage uses a Lagrangean relaxation technique whereas the second stage uses a granular tabu search heuristic. In a forthcoming paper, Absi et al. (2015) propose a two-phase iterative method for a production routing problem.

The sweep algorithm can be considered as one particular method for the cluster first, route second approach. This algorithm is attributed to Gillett and Miller (1974). The generic version of the sweep algorithm clusters a group of stops into a route based on the polar angle between the stops and the depot. Renaud and Boctor (2002) propose a new sweep-based heuristic for the fleet size and mixed vehicle routing problem. The proposed heuristic generates many candidate solutions and then chooses those that satisfy constraints for the problem using a polynomial set partitioning algorithm. Goodson et al. (2012) extend these work to vehicle routing problems with stochastic demand. Liu and Shen (1999) make use of the sweep heuristic within a two-stage metaheuristic procedure for the vehicle routing problem with time windows. Dondo and Cerdá (2013) formulate a mathematical program in order to choose the starting ray for the sweep heuristic for a VRP with cross-docking. It is also used to generate initial solutions to closely related problems (Imran et al., 2009; Zhong and Cole, 2005; Franceschi et al., 2006; Crevier et al., 2007; Cordeau et al., 1997). We do not use the sweep algorithm because it is not based on optimization. Methods that are superior use it as only a initial solution generator as the sweep algorithm solutions tend to be poor.

The cluster first, route second heuristic can benefit from special mathematical programming techniques for clustering (Laporte, 2009). We consider a clustering problem different from Bramel and Simchi-Levi (1995) in that our seed customers do not necessarily have to be from among the customers with a request. Mathematically, we allow the location of the seed customers (centroids) to be continuous as opposed to being discrete in Bramel and Simchi-Levi (1995). Choosing seeds in such a way makes the clustering problem easier. Additionally, we compute the assignment costs using distance functions like the Manhattan distance. In the 1-norm (Manhattan distance), an effective bilinear programming technique can be used for this purpose. Bradley et al. (1997) propose an effective clustering algorithm with well-founded theory from Bennett and Mangasarian (1993). They use bilinear programming to solve for possible global solutions to 1-norm clustering problems. We show how their algorithm is valid even with capacity restrictions and unit demand. The method they propose involves solving a few linear programs to obtain solutions, from which routing can be easily obtained using TSP solvers, eg. Concorde Applegate et al. (2006).

The formulation proposed by Bradley et al. (1997) corresponds to a nonlinear programming problem with a special structure. There are two sets of constraints: one corresponding to centers and distances of customers to centers, and one involving assigning customers to centers. Since these two sets of constraints are uncoupled (i.e. independent of each other), special bilinear programming methods are applicable, and the clustering problem can be solved by only a few iterations that involve linear programs. We make use of this technique in a slightly different problem with capacity restrictions and show that the technique is still valid in our case.

2.2.3 Approximations for VRP

There has been interest in developing simple vehicle dispatching rules, especially before the '90s when there was not much computational power. One such consideration was the important work in Daganzo (1984), and Newell and Daganzo (1986a,b). This line of study developed analytical formulas of approximations to the optimal shortest path between points on a map and also developed length formulas that apply to points that lie in zones of irregular shape. Under various assumptions regarding density of customer points and capacity of vehicles, the author analyzes the shape a district should take and suggests optimal slenderness factors for these districts in a rather loose way. At the time, this line of thought suggested guidelines for dispatchers to follow while building routes day to day. With large enough capacity, they approximate that the optimal slenderness factor, which is defined to be the ratio of the sides of a rectangle covering a group of customers oriented towards the depot, is a number either less, equal or greater than $\frac{6.7}{capacity}$ depending on how far the group of customers are from the depot. Overall, the main point from this study is that 'well-shaped' districts for a group of customers can vary and this depends on density of customers dispersed on the map, vehicle capacity and total number of customers that the depot will serve. The proposed formula for the average length of a CVRP problem is estimated to be

$$CVRP \approx 2\bar{r}n/C + 0.57\sqrt{nA}$$

where n is the number of customers, $\bar{r}$ is the average distance between the customers and the depot, and C is vehicle capacity, the maximum number of stops a vehicle can make, and A is the area covering the customers and the depot taken to be a rectangle or square.

Daganzo et al. (2012) discuss strong motivation for parsimonious models that can reduce computational complexity with little sacrifice in solution quality. When consider large-scale problems, exact methods are rendered useless by the size of data input, and therefore heuristic methods need to be chosen.

Chien (1992) carried out statistical analysis of the formulas that estimate TSP routes. The analysis was composed of estimating TSP routes on various data with different properties with regard to the rectangular shape of the area to be served and the shape of the circular sectors that can be formed within the area to be served. After fitting statistical parameters into his model, Chien found with mean absolute percentage error 6.9% that a TSP route length had the following statistical relationship on average:

$$TSP = 2.1\bar{r} + 0.67\sqrt{nR},$$

where R is the area of the smallest rectangle covering customers and $\bar{r}$ is same as above.

Kwon et al. (1995) looked at the same problem but also considered neural networks as an approximation to TSP route length. Their results extend Chien's results:

$$TSP \approx [0.83 - 0.0011\,(n+1) + 1.11S/\,(n+1)]\,\sqrt{nA} \qquad R^2 = 0.99 \qquad error = 3.71,$$

$$TSP \approx 0.41\bar{r} + [0.77 - 0.0008\,(n+1) + 0.9S/\,(n+1)]\,\sqrt{nA} \qquad R^2 = 0.99 \qquad error = 3.61.$$

Their main contribution that differs from Chien's work is that accounting for shape of the area improves accuracy. However, they do not find it to have a significant improvement over statistical estimation.

One can consider the length of a VRP solution to be the sum of lengths of TSP routes. Figliozzi (2008) proposes statistical models that estimate CVRP length in a Euclidean setting. Given n customers and m routes, the author studies several models:

$$VRP \approx k_l\sqrt{nA} + 2\bar{r}m,$$

$$VRP \approx k_l\frac{n-m}{m}\sqrt{nA} + 2\bar{r}m,$$

$$VRP \approx k_l\sqrt{nA} + k_m m,$$

$$VRP \approx k_l\frac{n-m}{m}\sqrt{nA} + k_m m,$$

$$VRP \approx k_l\sqrt{nA} + k_b\sqrt{\frac{A}{n}} + k_m m,$$

$$VRP \approx k_l\frac{n-m}{m}\sqrt{nA} + k_b\sqrt{\frac{A}{n}} + k_m m.$$

The parameters involved are k_l, k_m, and k_b, which are estimated using linear regression. One important addition as compared to previous work is the term $\frac{n-m}{m}$ which accounts for the following behavior: first, when $n = m$ the local distance traveled by a vehicle is zero; second, when $n \gg m$ or $m = 1$, the local tour distance tends to behave like as shown by Beardwood et al. (1959). The term $k_b\sqrt{\frac{A}{n}}$ is included into the model to capture the results of Kwon et al. (1995). The term $k_m m$ exists to capture the line-haul distance which increases as m increases or if the depot is farther away from the customers. His results show that models with $\frac{n-m}{m}$ have a superior performance in terms of error and that the last two models were the ones with better performance. However, if model simplicity is taken into account, model 2 is a good and robust model. These estimations are well founded, statistically speaking, but there is no apparent way of approaching it in an optimization framework because of the nonlinearity involved in these estimates which are not mathematically tractable.

Lei et al. (2011) study vehicle routing in the context of a districting problem. A districting problem is essentially designing service regions and assigning drivers to them. They study the problem in a stochastic framework and, hence, use a two-stage approach, where in the first strategic stage they design districts, and in the second stage, after demand requests have been revealed, routing is done at the operational level. In this second operational phase, it is desirable to assign customers to drivers in a consistent manner to improve service quality (Groër et al., 2009). Their objective is not only to obtain districts which deliver low cost routes but they also require that districts be contiguous. For this reason, they have a secondary objective that measures the compactness of a district (Bozkaya et al., 2003). In this measure, the compactness of a region depends on the perimeter of the region and the perimeter of the entire region that a depot is to serve. Specifically:

$$F_{comp}(x) = \frac{\sum_{k=1}^{m} B_k(x) - B}{2Bm}.$$

Ouyang (2007a) proposes algorithms to design vehicle routing zones from analytical approximation guidelines. These guidelines involve the shape and size of the districts to be designed. The design of these zones normally require human intervention, but with the guidelines in this work, large-scale problems can automatically be dealt with. The author discusses the usefulness of this approach in two-stage dynamic VRP settings where in the first stage strategic design decisions are made and in the second phase day-to-day operational routing decisions are made. The basic idea is to design zones that satisfy qualitative size and shape requirements for easy optimization later on during day-to-day optimization. It is also mentioned that this technique is suitable for obtaining solutions for operational decisions as well. Moreover, these clustering-based solutions can be further improved by metaheuristic methods (Daganzo, 1984; Robust et al., 1990).

Clarens and Hurdle (1975) develop an analytical approximation to a similar problem about bus transit design. (Robust et al., 1990) argue that 'idealized' models can be used to obtain easy-to-implement algorithms to large and complex logistics problems. The main idea in this paper is to use previous studies of these authors to implement a cluster-first, route-second-like algorithm with fine tuning using simulated annealing after having obtained an initial VRP solution.

Daganzo and Erera (1999) also address the issue of approximation methods to VRP. They argue that the motivation for approximation is strong when uncertainty is involved in the problems, especially in dynamic settings. Large-scale problems cannot possibly be solved using traditional stochastic programming approaches in an effective way because most of the time statistical assumptions of the underlying distributions can be far from reality. The authors design districts with customers that are known to require delivery a priori and those customers with unknown demand. They use analytical approximation results to design districts. Their techniques are based on Daganzo (1984).

Designing territories as a strategic decision involves uncertainty with regard to demand, customer location and other unforeseen circumstances such as traffic, and road blockage. The works cited before here simplifying assumptions about some of these features to be able to generate ideas

that are mathematically tractable. That is, their results are true on average. On the other hand, if clustering is carried out at every decision epoch as uncertainty is realized and become known, the realized data could guide us towards making the best possible decision for that epoch. Therefore, cumulative decisions would accumulate to the best possible history of decisions.

2.2.4 Fish Harvesting Strategy and Stock Assessment

We focus on the qualitative and quantitative studies of fishermen behavior and strategy. The study of fishermen's behaviour is important for ecological sustainability, as well as economic and fishery management reasons. The nature of fishing activity depends on various factors including vessel type, target species, gear configuration, skipper behavior and experience. We first review studies that analyze fishing activity in various fisheries around the world. Then, we discuss how the uncertainty in catch is modeled in empirical studies.

Prellezo et al. (2009) analyze the selection of fishing areas by Basque trawlers. In a qualitative survey, they ask fishermen about the reason for a particular choice of fishing area on a fishing trip. According to the survey, experience was the most important reason for selecting a particular fishing area. Regulation came up as the second most important response. Expected harvest was the response in 7% of the total responses. Communication with other fishermen accounted for 6% of the responses and fuel consumption 2%. Holland and Sutinen (2000) report based on interviews with fishermen that fishery and location choice is a multi-level process. The decision on what fishery to visit is typically made before beginning a fishing trip. A fishermen may initially have a location in mind. However, he or she may change her mind based on information received en route. This information may come from other vessels or may depend on their observations of potential catch along the way. Teh et al. (2012) investigate the preferences of small-scale fishers in Malaysia based on their mental perceptions, and how this is influenced by imposing spatial regulations.

Bertrand et al. (2007) analyze the strategy employed by fishermen while searching for fish in Peru. They use satellite vessel monitoring systems (VMS) to model the observed trajectories of a large industrial fleet operating on the Peruvian anchovy fishery. Their approach is within the context of a predator-prey relationship. Consequently, they model the movements of fishing vessels as random walks (Turchin, 1998). They use a random walk model, the Levy motion, to describe the motion of vessels. A trajectory is seen as a series of elementary behavioral events. These elementary events are: move lengths, move durations, and move headings (direction). Their data consists of 210,530 trips at sea that correspond to 14,106,533 observations on point-geographical locations. After preprocessing, they use 88,421 fishing trips with 1,600,705 moves. About $80-96\%$ of the trajectories showed no correlation for move lengths and move durations. Moreover, 94% of the fishing trips showed no correlation in move headings. Their statistical analysis agrees with the Levy random walk hypothesis of vessel trajectories.

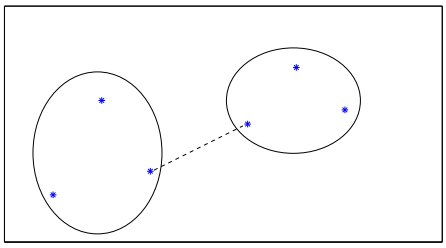
Powell et al. (2003) analyze vessel time allocation for the harvesting of *Illex illecebrosus.* Their data contains position and time information on five commercial fishing trips off the east coast of US for a total of 44 days at sea. They identified seven activities while harvesting: steaming to and

from port, searching, towing, set-up time between tows, steaming overnight and laying-to overnight. Tows tended to average about three hours in duration, whereas searches tended to be about 1.5 hours in duration. The authors suggest that more squid could have been caught had the vessels used less time for searching and more time for towing.

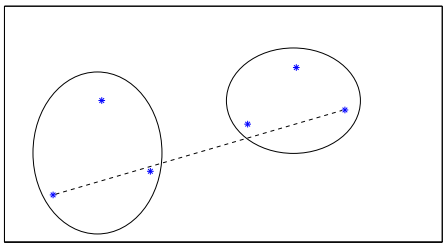
Tuna purse seiners are more traditional kind of vessels used to harvest tuna. Bez et al. (2011) estimate the tuna purse seiners in the Indian Ocean to spend 24%, 49%, and 27% of their time to fishing, tracking, and cruising, respectively. Russo et al. (2011) identify different patterns in trajectories of fishing vessels across different metiers where a metier is defined to be groups of vessels with the same exploitation pattern (e.g. gear used, fishing ground, target species).

Log-normal likelihood is a common assumption in stock assessment models (Punt and Hilborn, 1997). Dorn (2001) models the catch process as a log-normal distribution whose mean is modeled as a first-order autoregressive process with a time resolution of a single tow. Empirical studies also assume a log-normal distribution for monthly catch-per-unit-effort (Winker et al., 2013; Nishida and Chen, 2004; Bishop et al., 2004, 2008). We follow the literature and assume that catch follows a log-normal distribution. Lane (1989) models the catch process as a negative binomial distribution with a time resolution of a single trip.

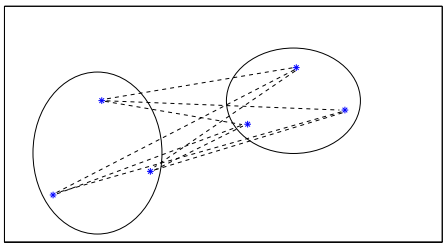
Fishery management use bioeconomical models to assess the regenerative capability of a fishery and its economic viability. A seminal work in bioeconomic models is Schaefer (1957) where the author presents a surplus production model. Surplus production models are used to compute the maximum sustainable yield of a resource. Many contributions have been made over the years. A survey of these models can be found in Knowler (2002).



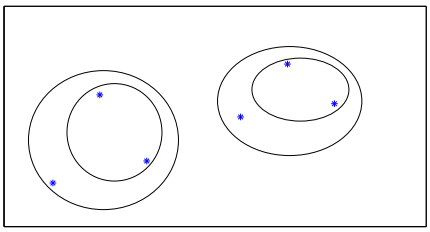
(a) Single link



(b) Complete link



(c) Group average



(d) 2 nested clusters

Figure 2.2.2: Single link, Complete link, and Group average.

Chapter 3

Clustering

3.1 Introduction

In this chapter, our goal is to explore various clustering techniques and apply it to vehicle routing problems. We focus on the daily routing decision facing a vehicle dispatcher situated at a depot. We are mainly interested in analyzing clustering techniques for customer distributions with different properties. In this respect, we focus on problems with only unit demand, especially for large-scale problems. The dispatcher has full knowledge of the customer requests for that day. We propose cluster first, route second algorithms for this purpose.

Exact methods for capacitated vehicle routing problems (CVRP) are able to solve problems with around 100 customers. Yet, there are many applications that require solving problems with hundreds and thousands of customers. As a result, researchers have mainly approached large scale vehicle routing problems using heuristics. This line of work has focused on heuristics such as genetic/memetic algorithms, tabu search, and other neighborhood search algorithms (Nagata and Braysy, 2009; Marinakis, 2012; Cordeau and Maischberger, 2012; Jin, Crainic, and Lø kketangen, 2012; Kytöjoki, Nuortio, Bräysy, and Gendreau, 2007; Mester and Bräysy, 2007).

One method that can address the scale issue is the cluster-first, route-second heuristic. In this method, the original problem is decomposed into smaller subproblems. Each subproblem is a TSP on its own. Existing state-of-the-art TSP solvers can easily manage problems of size up to 100 customers (Applegate et al., 2006). On the other hand, the main problem is a generalized assignment problem (Fisher and Jaikumar, 1981). One important contribution is in Bramel et al. (1992), where the authors modeled the CVRP as a capacitated discrete facility location problem. They use a Lagrangian relaxation technique to tackle moderately-sized problems. The main challenge is estimating the routing cost of adding a customer to a specific cluster by using assignment costs. However, this first stage can benefit from the use of mathematical programming techniques for better optimization (Laporte, 2009). We make use of such a benefit in a specific case, the 1-norm. In the 1-norm, clustering can be done quite efficiently and we can obtain high quality and/or global solutions to the clustering problem. It also makes sense using the cluster-first, route-second approach for large scale problems. In this way, a feasible solution can be obtained efficiently from which

local search heuristics can be used to improve the solution.

A vehicle routing solution consists of different group of customers ordered along a route. Seen from a hierarchical point of view, an optimal solution is comprised of optimal groups of customers and an optimal ordering of those customers within each group or cluster. Depending on the customer distribution, the optimal or near optimal clustering of customers can vary. Knowing the customer distribution can help choose a suitable clustering algorithm that can deliver better routing in terms of a cluster-first, route-second approach. Depending on the clustering tendency of a customer distribution, there can be specific clustering algorithms that stand out in terms of more efficient routing. We study some clustering algorithms from the data mining literature and propose others for this purpose.

In the following sections, we propose three cluster first, route second methods. We first focus on a modified k-medians (1-norm) technique, but use a special solution method. The modified k-medians problem has capacity restrictions for each cluster and the depot is assumed to be a member of each cluster. Second, we talk about clustering based on polar angles between customers and the depot to take advantage of specific customer distributions. Third, we talk about a clustering method that incorporates the shape of the clusters formed to combine insights from the first two methods. We present the results obtained from applying our cluster first, route second methods to some benchmark instances and some modified benchmark instances. Finally, we discuss the results of our experiments and conclude.

3.2 Formulation

Here, we give a standard formulation of a capacitated vehicle routing problem.

$\mathscr{I}$ is the index set for customers and depot where depot's index is 0.

$\mathscr{L}$ is the index set for vehicles.

d is the number of vehicles, i.e. $|\mathscr{L}|$.

m is the number of customers, i.e. $|\mathscr{I}|-1$.

q_i is the demand of customer $i \in \mathscr{I}$ which is assumed to be one.

v is the capacity of each vehicle.

c_{ij} is the cost of traveling from customer i to customer j, which is symmetric and equal to the Euclidean distance.

$$x_{ijl} = \begin{cases} 1, & \text{if vehicle } l \text{visits customer } i \text{from customer } j \\ 0, & \text{otherwise} \end{cases}$$

$$y_{il} = \begin{cases} 1, & \text{if customer } i \text{is visited by driver } l \\ 0, & \text{otherwise} \end{cases}$$

The mathematical formulation as an integer programming problem is given below.

$$
\begin{aligned}
\min \quad & \sum_i \sum_j \sum_l c_{ij} x_{ijl} \\
\text{subject to} \quad & \sum_i q_i y_{il} \leq v && \forall\, l \in L \\
& \sum_l y_{il} = \begin{cases} d, & i = 0 \\ 1, & i = 1, \ldots, m \end{cases} \\
& \sum_i x_{ijl} = y_{jl} && j \in I,\, l \in L \\
& \sum_j x_{ijl} = y_{jl} && i \in I,\, l \in L \\
& \sum_{i,j} x_{ijl} \leq |S| - 1 && S \subseteq I \setminus \{0\}, \quad 2 \leq |S| \leq m-1 \\
& x_{ijl} \text{ binary}, \quad y_{il} \text{ binary}
\end{aligned}
$$

In our solution approach, this integer program is decomposed into an assignment problem and d many traveling salesman problems. The heuristic by Fisher and Jaikumar (1981) is a decomposition to the above formulation in the following way.

$$
\begin{aligned}
\min \quad & \sum_l f(y_l) \\
& \sum_i q_i y_{il} \leq capac && \forall\, l \in L \\
& \sum_l y_{il} = \begin{cases} d, & i = 0 \\ 1, & i = 1, \ldots, m \end{cases} \\
& y_{il} \text{ binary}
\end{aligned}
$$

where $f(y_l)$ is the cost for the traveling salesman path for the group of customers $M_l = \{i \in I \,|\, y_{il} = 1\}$. Then, for a given solution to the above problem, for each subset M_l there corresponds a traveling salesman problem. We do not write a mathematical formulation for it since it is standard and can be found in the literature (Rego et al., 2011; Bektas, 2006; Laporte, 1992).

The important part of this decomposition is the objective in the assignment problem, $f(y_l)$. Researchers have used linear functions of $\{y_{il}\}$ to act as a surrogate for $f(y_l)$. As discussed in the following sections, our approach is in the same vein.

3.3 Clustering with Manhattan Distance

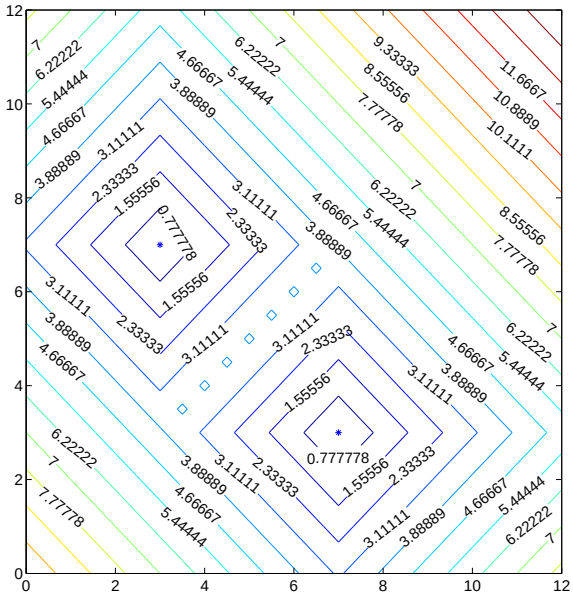
Our method of clustering customers is different from previous work. The Fisher and Jaikumar (1981) method begins by selecting a set of seed customers. Using this set of customers, they compute the cost of inserting a customer to a specific cluster. This makes the quality of the routing dependent on the chosen seed customers and the resulting estimation of route cost. Bramel et al. (1992) allow for updating the choice of seed customers in the optimization procedure for better routing quality. However, this requires a more extensive computational procedure. We use a special technique that updates the seed customers during the algorithm and can obtain high quality clusters within reasonable time.

We present a 1-norm (Manhattan or rectangular distance) clustering algorithm. In the 1-norm, the clustering problem may also be called the k-medians clustering problem. Using 1-norm as a distance measure is more suitable when a vehicle has to travel on a grid to get from point A to point B. This is especially the case for a depot serving a region whose road network resembles a rectangular grid, which is true for metropolitan areas. As typical of k-means clustering algorithms, this algorithm delivers clusters with a convex shape. Moreover, the 1-norm enables the application of a special type of mathematical optimization technique, namely bilinear programming. We formulate the decision to assign drivers to customers that is a clustering problem as a bilinear programming problem and solve it using the Uncoupled Bilinear Program Algorithm technique (Bradley et al., 1997).

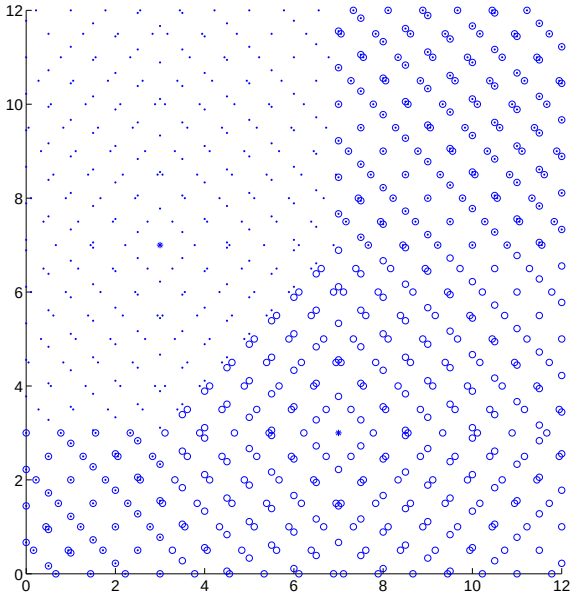
We illustrate the type of clusters 1-norm distance can deliver in Figure 3.3.1. In Figure 3.3.1a we show two hypothetical centers for two clusters, $(3,7)$ and $(7,3)$. The contour lines in this figure represent the distance of each point on the contour line to its nearest center. We see how the clusters are in a grid shaped fashion and convex. In Figure 3.3.1a, we show how every point in the positive quadrant would be assigned under the assumption that cluster sizes are not constrained. The points in the quadrant are classified into two. Those with the dot symbol and those with the circle symbol. There exist some points with both symbols because they are of equal distance to each cluster center.

3.3.1 Formulation

Given a set of m customer locations in $\mathbb{R}^2$ represented by $A \in \mathbb{R}^{m\times 2}$ that require delivery and a number of drivers d, the clustering problem is to solve the following optimization problem:



(a) contour of minimum 1-norm distance to centers $(3,7)$and $(7,3)$.



(b) cluster assignments.

Figure 3.3.1: 1-norm distances to two different centers.

$$\underset{C_l, D_{il}, T_{il}}{\text{minimize}} \quad \sum_{i=1}^{m}\sum_{l=1}^{d} e^T D_{il} T_{il} \tag{3.3.1}$$

$$\text{subject to} \quad -D_{il}^j \leq \left(A_i^T - C_l\right)^j \leq D_{il}^j \qquad \forall\, i, l, j \tag{3.3.2}$$

$$\sum_{l=1}^{d} T_{il} = 1 \qquad \forall\, i \tag{3.3.3}$$

$$\sum_{i=1}^{m} q_i T_{il} \leq v \qquad \forall\, l \tag{3.3.4}$$

$$T_{il} \in \{0,1\} \qquad \forall\, i, l \tag{3.3.5}$$

$$D_{il}^j \geq 0 \qquad \forall\, i, l, j \tag{3.3.6}$$

Here, D_{il} is a dummy variable in $\mathbb{R}^2$ that represents the bounds on the 1-norm distances and e is a vector of ones. D_{il}^j is the jth component of vector D_{il}. So, the term inside the summation is the summation of the bounds on the 1-norm distances. The choice variables are C_l and D_{il} for each $i \in \{1,..,m\}$ and $l \in \{1,...,d\}$. C_l is a 2×1 vector representing the coordinates of cluster center l. The binary assignment variables T_{il} are equal to 1 if customer i is assigned to cluster l and 0 otherwise.

The above formulation is an uncoupled bilinear programming problem when the integer constraints are relaxed. The objective function is bilinear in the variables (D_{il}, T_{il}). An important property of such a function is that when one group of variables are fixed, the objective is linear in the other set of variables. Moreover, the constraints can be separated into two independent groups with no interaction between the variables from each group. The constraints (3.3.2) are those that have to do with the distance of customers to every center. These are independent from the constraints (3.3.3) and (3.3.4), which say that every customer should be assigned to one cluster and cluster size should not be more than its capacity.

Bradley et al. (1997) reformulate the k-medians clustering problem as an uncoupled bilinear programming problem. It is noteworthy to mention that the clustering problem involves integer variables whereas the bilinear programming problem relaxes these variables. They show that the reformulation is equivalent and does not change the optimal solution. However, the implementation of bilinear programming to 1-norm clustering in Bradley et al. (1997) does not have capacity restrictions. For the reformulation to work, we must show the equivalence of the clustering problem to an equivalent bilinear programming problem. We show here that in the *unit* demand case the transformation is still valid. That is, we show that relaxing the integer constraints on T_{il} is possible through Lemma 3.3.1. Consequently, the equivalence is satisfied with Proposition 3.3.2.

Assume $a_{il} = e^T D_{il}$. Given a_{il}, the clustering problem is a decision problem in only T_{il} subject to (3.3.3), (3.3.4) and (3.3.5). Moreover, the objective function is linear in these variables. We have the following:

$$
\begin{aligned}
\underset{T_{il}}{\text{minimize}} \quad & \sum_{i=1}^{m}\sum_{l=1}^{d} a_{il}T_{il} & \\
\text{subject to} \quad & \sum_{l=1}^{d} T_{il} = 1 & \forall\, i \\
& \sum_{i=1}^{m} q_i T_{il} \leq v & \forall\, l \\
& T_{il} \in \{0,1\} & \forall\, i,l
\end{aligned}
$$

If you relax the integer constraints in the above problem, this is equivalent to the transportation problem which is a class of minimum-cost network flow problems. As a result, we can relax the integrality constraints without changing the optimum solution.

Lemma 3.3.1. *(Nemhauser and Wolsey, 1988) Let $V_1 = I = \{1,2,...,m\}$ and $V_2 = L = \{1,2,...,d\}$. Let $P = \{(i,l) : i \in V_1 \text{ and } l \in V_2\}$ be a set of arcs connecting nodes in the customer set I to nodes in the drivers set L . Then, $G = (V_1, V_2, P)$ is a digraph. Moreover, the above problem corresponds to the transportation problem in the relaxed variables $T_{il} \in \mathbb{R}$. In such a situation, the constraint matrix is totally unimodular. Therefore, the integrality requirement is automatically satisfied.*

Proposition 3.3.2. *Let $T_{il} \in \mathbb{R}^1$ be a continuous variable in $(0,1)$ for all $i \in \{1,..,m\}$ and $l \in \{1,...,d\}$. Under such a relaxation, the clustering problem is an instance of a bilinear program.*

3.3.2 Algorithm

We can apply the Uncoupled Bilinear Program Algorithm in Bennett and Mangasarian (1993) to our problem. Let us reexpress the problem in matrix form as follows and name it BP,

$$
\begin{aligned}
\min_{y,x} \quad & y'Qx && (3.3.7)\\
\text{subject to} \quad & A_1 x \leq b_1, && (3.3.8)\\
& A_2 y \leq b_2, && (3.3.9)\\
& A_3 y = b_3, && (3.3.10)\\
& x \geq 0,\ y \geq 0. && (3.3.11)
\end{aligned}
$$

The decision variables in this form are grouped as follows: a vector of variables $x = \left(\left\{D_{il}^{j}\right\}, \{C_l\}\right)$ containing the distance variables, $\left\{D_{il}^{j} : i = 1,...,m \quad l = 1,..,d \quad j = 1,2\right\}$, and the center variables $\{C_l : l = 1,..,d\}$. A vector of variables $y = (\{T_{il}\})$ containing the assignment variables, $\{T_{il} : i = 1,...,m \quad l = 1,..,d\}$.

Let the constraint group (3.3.2) be expressed in matrix form as (3.3.8) and the constraint group (3.3.3), (3.3.4) be expressed as (3.3.9) and (3.3.10). The resulting optimization problem belongs to the class of bilinear programming problems.

Let the bilinear program BP be denoted by $P(Q, A, b)$ where Q, $A = (A_1, A_2, A_3)$ and $b = (b_1, b_2, b_3)$ are the givens in the problem. The Uncoupled Bilinear Program Algorithm alternates between solving two different mathematical programs with respect to each group of constraints while keeping the other variables (which are in the other group of constraints) fixed.

Algorithm 2 Uncoupled Bilinear Program Algorithm

Starting from an initial feasible point, $\left(x^0, y^0\right)$, determine $\left(x^{r+1}, y^{r+1}\right)$ from (x^r, y^r) as follows:

1. Solve for x^{r+1} in problem $P(Q, A, b; y^r)$ only subject to (3.3.8) which is a linear programming problem.
2. Solve for y^{r+1} in problem $P\left(Q, A, b; x^{r+1}\right)$ only subject to (3.3.9).
3. Stop when $\left(y^{r+1}\right)' Q x^{r+1} \geq (y^r)' Q x^r$. That is, the objective does not improve in step $r+1$.

The bilinear program BP can be shown to have a vertex solution (Bennett and Mangasarian, 1993). Moreover, they show that Algorithm 2 has finite termination at either a global minimum or a point satisfying the "minimum principle" (Mangasarian, 1994; Bennett and Mangasarian, 1993). The algorithm involves solving linear programs throughout all iterations. We use a call to CPLEX's internal linear program solver, which is a primal-dual interior point method. These algorithms are known to have polynomial time worst-case complexity. However, the number of iterations in the worst-case can be exponential, for there exists an exponential number of vertices that the algorithm can traverse before reaching the optimum. In practice, run-time is expected to be much better.

3.4 Spherical Clustering

We focused on a clustering problem using the 1-norm in the previous section. This same clustering problem may be called the k-medians problem in other literature. 1-norm (k-medians) clustering focuses on finding tightly packed cluster of customers and optimizes over such a clustering structure. This is suitable for data realizations that have strong features like these. However, data realization may be such that customers are dispersed in ways where the 1-norm solution structure is not that effective. These cases are observed for some benchmark instances used in the literature. The group of instances that correspond to randomly dispersed customers within the Gehring-Homberger-unit benchmark are examples of these.

We borrow from the data mining literature, more specifically text mining, in order to cluster based on the angles data points make with the location of the depot. In this way, we respect the even radial distribution of the data. For this purpose, we first need to talk about similarity measures, and

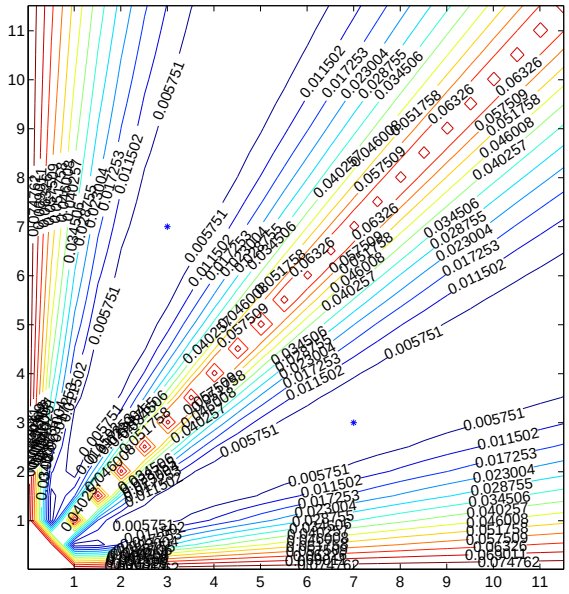
in particular *cosine similarity*.

Consider two points in the plane $x_1 = (a, b)$ and $x_2 = (c, d)$. The cosine similarity between these two points is given by a scalar

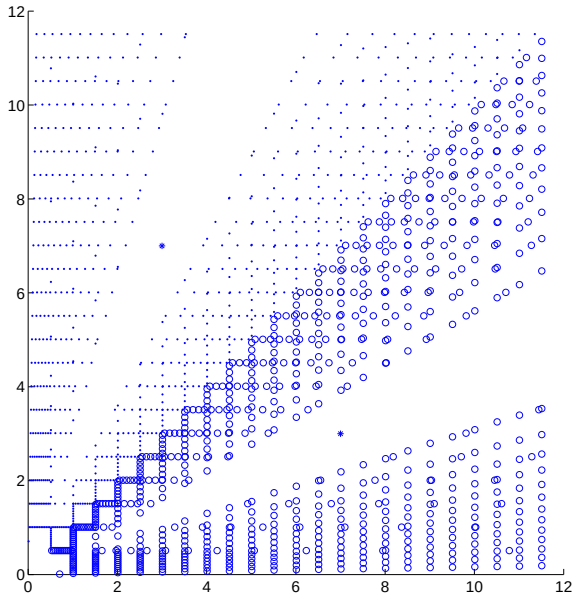
$$cos(\theta) = \frac{x_1 \cdot x_2}{\|x_1\| \, \|x_2\|}.$$

This scalar can be between -1 and 1. If the points lie on exactly opposite rays emanating from the depot, then their similarity is -1. However, if they lie on the same ray, then this similarity will be 1. If it turns out that they are orthogonal to each other, that is there is a 90 degree angle between them, then their similarity score is zero. The intuition with respect to vehicle routing is that effective routes take into account how customers are situated with respect to the depot.

We illustrate the type of clusters cosine distance can deliver in Figure 3.4.1. In Figure 3.4.1a we show two hypothetical centers for two clusters, $(3, 7)$ and $(7, 3)$. The contour lines in this figure represent the distance of each point on the contour line to its nearest center. We see how the clusters are in radial form emanating from the origin representing the depot. In Figure 3.4.1b, we show how every point in the positive quadrant would be assigned under the assumption that cluster sizes are not constrained. The points in the quadrant are classified into two. Those with the dot symbol and those with the circle symbol.



(a) contour of minimum cosine distance to centers $(3,7)$ and $(7,3)$.



(b) points assigned to nearest center.

Figure 3.4.1: Cosine distances to two different centers.

In the next section, we present the optimization problem for the clustering problem for spherical k-means. Then, we present the algorithm we choose to obtain solutions. Finally, we talk about the results from computations we carried out on benchmark instances.

3.4.1 Mathematical Formulation

Here, we present the mathematical formulation for spherical clustering. Given a set of m points in $\mathbb{R}^2$ represented by $A \in \mathbb{R}^{m\times 2}$ and number of clusters, d (number of drivers), we wish to maximize the similarity between all customers that belong to the same cluster. Define a_{ij} as the ith row and jth column of matrix A. It represents the jth coordinate of customer i in a given map. Let $a_i = (a_{i1}, a_{i2})$ denote coordinates of customer i. Define $\widetilde{a_i} = \frac{a_i}{\|a_i\|}$ as the normalized locations for customer i. In other words, we translate the coordinates so that the customers lie on the unit circle around the depot but still on the same ray from the depot. Let T_{il} be binary assignment variables which are equal to 1, if customer i is assigned to cluster l; and 0 otherwise. The variable c_l represents center for cluster l on the unit sphere around the depot.

$$\max_{c_l, T_{il}} \quad \sum_{i=1}^{m}\sum_{l=1}^{d} \widetilde{a_i}^T c_l T_{il} \tag{3.4.1}$$

$$\text{subject to} \quad \sum_{i=1}^{m} q_i T_{il} \leq v, \qquad l = 1,...,d, \tag{3.4.2}$$

$$\sum_{l=1}^{d} T_{il} = 1, \qquad i = 1,..,m, \tag{3.4.3}$$

$$\|c_l\| = 1, \qquad l = 1,...,d. \tag{3.4.4}$$

Constraints (3.4.2) tell us that capacity of each cluster must be met. Constraints (3.4.3) make sure that everyone is assigned to one cluster. The group of constraints (3.4.4) ensure that the chosen centers lie on the unit circle around the depot. These constraints are nonlinear; however, we solve for them analytically during the course of the algorithm which we present in the next section.

3.4.2 Algorithm

Finding an optimal solution to the above problem is NP-complete (Kleinberg et al., 1998). We employ an iterative procedure used in Dhillon and Modha (2001) called the *spherical k-means* algorithm. The worst-case complexity of this algorithm is exponential as there is a call to a branch-and-bound CPLEX routine to solve for the integer programs within the *while* loop.

Algorithm 3 Spherical K-means

Initialization. Initialize iteration counter, $t \leftarrow 0$. Randomly pick $\left\{c_l^t\right\}_{l=1}^d$ on the unit circle around the depot.

while (centers change)

- Find new assignments of customers to centers such that similarity is maximized.

$$\left\{U_{il}^{t+1}\right\}_{i=1,l=1}^{m,d} \in \arg\max_{U_{il}} \left\{\sum_{i=1}^m \sum_{l=1}^d \widetilde{a}_i^T c_l^t U_{il} \ : \text{ subject to (3.4.2) and (3.4.3)}\right\}$$

- Update centers $\left\{c_l^{t+1}\right\}_{l=1}^d$ as

$c_l^{t+1} = \frac{m_l^{t+1}}{\left\|m_l^{t+1}\right\|}$ where m_l^{t+1} is the centroid of customers i such that $U_{il}^{t+1} = 1$.

Stop when centers do not change.

The "update-centers" step in the algorithm 3 uses an analytical solution for the optimum set of centers given the assignments from the previous step in the algorithm. Naturally, this shortens the run-time. This analytical result is shown with the following lemma, Lemma 3.4.1, that is a result of the Cauchy-Schwarz inequality.

Lemma 3.4.1. *(Dhillon and Modha, 2001) Let $X = \{x_1, x_2, x_3, ..., x_k\}$ be a subset of R^n. Let $m \in \mathrm{R}^n$ be the centroid of the set X. Let $c = \frac{m}{\|m\|}$ be the normalized centroid. Then, for any vector $z \in \mathrm{R}^n$, we have the following as a result of the Cauchy-Schwarz inequality*

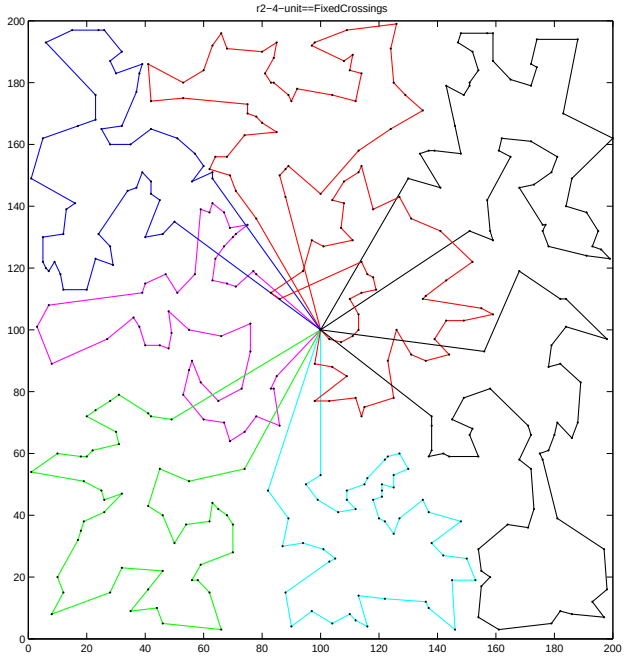
$$\sum_{x \in X} x^T z \leq \sum_{x \in X} x^T c$$

3.5 Clustering Based on Shape (Slender Clustering)

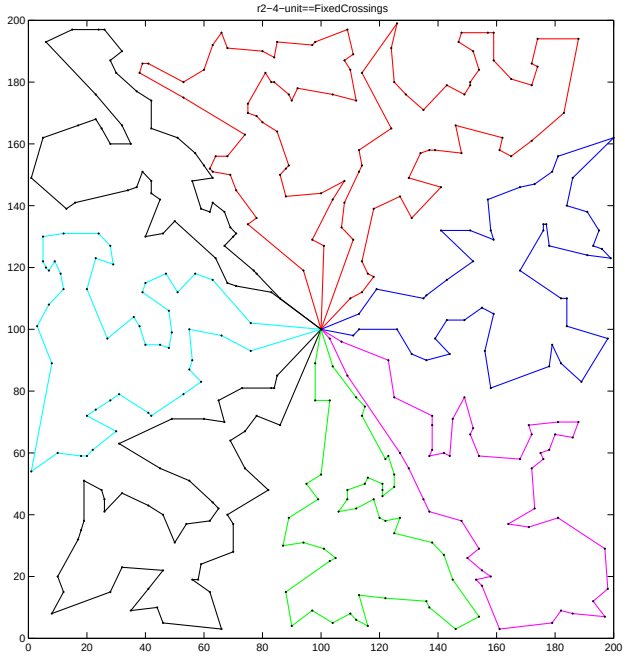
In this section we develop another model of clustering that mitigates some of the problems with k-medians and spherical clustering. The k-medians algorithm focuses on finding clusters that are tightly packed together. This bias becomes less effective when the customer distribution is more radial and evenly spread out. For example, in Figure 3.5.1, we see how the k-medians algorithm clusters customers versus how the spherical algorithm clusters them. It makes sense to cluster customers based on radial distances in such a case as the distribution of customer locations enables a

lower traveling cost. In this case, spherical clustering was more appropriate for routing. However, there are problems clustering solely based on the objective for spherical clustering. In order to generalize over the implications of the previous methods as discussed in Section 3.6.2 and Section 3.6.3, we propose another method that is parametric over the shape of the clusters. We were motivated by the work of Daganzo (1984). The major contribution of his work is to provide analytical guidelines for constructing near-optimal routes. At the time, these guidelines were geared towards human dispatchers. The main idea behind these guidelines is that the optimal slenderness (length / width) of the rectangle covering the customers in a single cluster should be a ratio depending on how far the group of customers are to the depot, the capacity of the vehicles and customer density (Ouyang, 2007b). Specifically, they show that the near-optimum shape of a cluster far from the depot should be elongated towards the depot reflecting a 'thin' route, where as a cluster that is close to the depot can have any shape under optimality.

We model these qualitative guidelines into an objective function whose parameters control the shape of clusters. The modeling requires to translate the original customer locations (cartesian coordinates) into polar coordinates so that clusters elongated towards the depot can be captured in our model as Ouyang (2007b) suggests. The parameters for the model control the influence of the distance between a customer and a cluster center. We measure the distance between a customer and a cluster center by summing the weighted combination of the angular and radial distance between them. It is these parameters that control the weight of each respective distance depending on how far the customer is from the depot. If the weight given to the angular distances is zero, then we do not care about the difference in angle between two customers. If the weight given to the radial distances is zero, then we only care about the angle between two customers.



(a) K-medians routing with VRP cost 3766.



(b) Spherical routing with VRP cost 3501.

Figure 3.5.1: Randomly scattered customers.

In the following section, we detail a mathematical formulation of clustering as an integer program that takes into account the shape of clusters. We give the name *slender* for this method. We refer to it in the following sections with this name interchangeably. When we use the term shape, we mean the ratio of the radial and transversal length of a group of customers considered as a cluster. Then, we discuss the solution methodology we use to tackle this problem. Finally we show empirical results of its effectiveness as compared to the baseline state-of-the-art VRP solver.

3.5.1 Mathematical Formulation

We formulate an integer program that models the shape of clusters in a clustering problem. This formulation can be seen as a discrete facility location problem with a modified objective. This objective reflects the kind of clusters we want to see. We guide the solutions towards the desired cluster shapes using the coefficients which are set a priori. We calibrate those parameters based on experimentation.

We first do some preprocessing from the original data. We translate the location of each customer as well as the location of the depot to polar coordinates, (θ, ρ). The depot is assumed to take the value of $(0, 0)$, corresponding to zero angle (θ) and zero distance (ρ). There is a set of customer indices $\mathscr{I} = \{1, 2, ..., m\}$ and a set of clusters $\mathscr{L} = \{1, 2, .., d\}$.

There are three sets of binary variables:

1. $X_{ilj} \in \{0, 1\}$ to denote the condition that customer $i \in \mathscr{I}$ is assigned to center $l \in \mathscr{L}$ which is chosen to be customer $j \in \mathscr{I}$. Since we choose cluster centers from among the customers, there needs to be a third index running over all the customers allowing for the possibility of each customer to be a cluster center.

2. $U_{il} \in \{0, 1\}$ is equal to 1 if customer i is assigned to center $l \in \mathscr{L}$.

3. $P_{lj} \in \{0, 1\}$ is equal to 1 if center l is chosen to be customer $j \in \mathscr{I}$.

We compute coefficients for the objective function whose parameters control the slenderness of the clusters. They assure that we get thin and long clusters elongated towards the depot or fat and wide clusters. The data given to the optimization problem is customer demand, $\{q_i\}_{i \in \mathscr{I}}$ and input coefficients $\left\{\delta_{ilj}\right\}_{i \in \mathscr{I}, l \in L, j \in \mathscr{I}}$ that are precomputed as follows

$$\delta_{ilj} = \alpha^1_{il}\theta_{ij} + \alpha^2_{il}\rho_{ij} \tag{3.5.1}$$

where

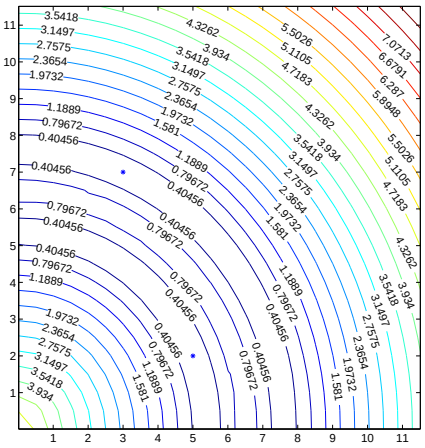
$$\theta_{ij} = \pi - \left|\pi - \left|\theta_i - \theta_j\right|\right| \qquad \text{for all } i, j \in \mathscr{I}. \tag{3.5.2}$$

$$\rho_{ij} = \left|\rho_i - \rho_j\right| \qquad \text{for all } i, j \in I. \tag{3.5.3}$$

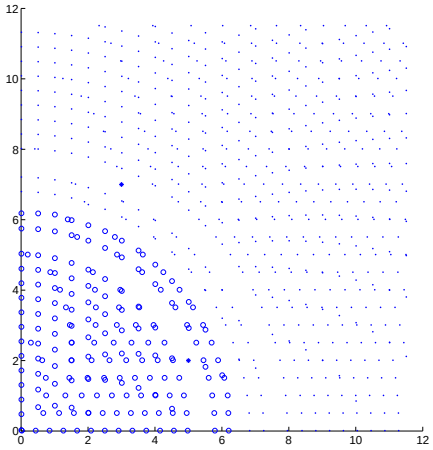
$$\alpha_{il}^1 + \alpha_{il}^2 = 1 \qquad \text{for all } i \in I, l \in \mathscr{L}. \tag{3.5.4}$$

We control how each distance, angular (θ_{ij}) and radial (ρ_{ij}), influences the objective via the weights $\left(\alpha_{il}^1, \alpha_{il}^2\right)$ associated with each respectively. The δ_{ilj} are the coefficients in 3.5.1 that belong to the objective of the clustering. These coefficients weigh the cost of assigning customer i to center l that is chosen to be customer j. They depend on the angular difference between customer i and customer j, θ_{ij}; and they depend on the difference in the distances to the depot of customer i and customer j, ρ_{ij}. The parameters ρ_{ij} are computed in the obvious way given by 3.5.3. On the other hand, the paramaters θ_{ij} are computed as the acute angle between customer i and customer j, which can be computed analytically by the expression in 3.5.2. These weights sum up to one as given in 3.5.4. If α_{il}^1 is relatively higher, then assigning customer i to center l cares more about the angular difference in terms of increasing the objective. If α_{il}^2 is relatively higher, then it is more preferable to have customer i and center l to be of similar distance to the depot.

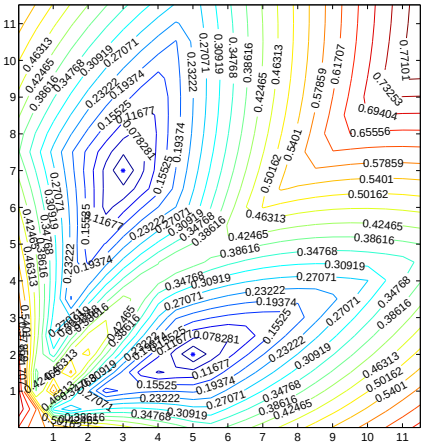
We illustrate the type of clusters slender distance can deliver in Figure 3.5.2. In Figure 3.5.2a we show two hypothetical centers for two clusters, $(3,7)$ and $(5,2)$ with differing distance from the origin or depot. The contour lines in this figure represent the distance of each point on the contour line to its nearest center. We see how the clusters are in near-circular form. This is due to the low weight, $\left(\alpha_{il}^1, \alpha_{il}^2\right) = (0.05, 0.95)$, assigned to angular distance between points. If the points in the positive quadrant are assigned to their nearest center, we get the clusters in Figure 3.5.2b. On the other hand, when there is high emphasis given to angular distance with $\left(\alpha_{il}^1, \alpha_{il}^2\right) = (0.95, 0.05)$, then we get contour lines as in Figure 3.5.2c that resembles contour lines we observe for cosine distance used in spherical clustering. Hence, the resulting clusters look like in Figure 3.5.2d.



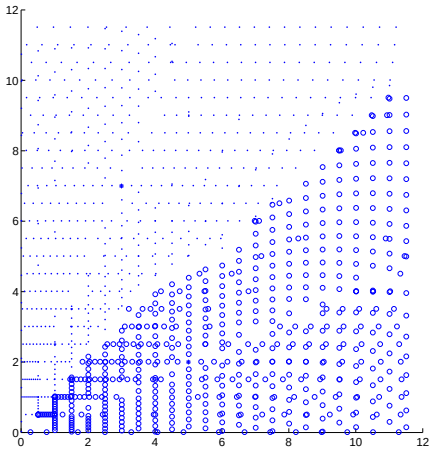
(a) contour of minimum slender distance with weights $(\alpha_{il}^1, \alpha_{il}^2) = (0.05, 0.95)$ to centers $(3,7)$ and $(5,2)$.



(b) points assigned to nearest center.



(c) contour of minimum slender distance with weights $(\alpha_{il}^1, \alpha_{il}^2) = (0.95, 0.05)$ to centers $(3,7)$ and $(5,2)$.



(d) points assigned to nearest center.

Figure 3.5.2: Slender distances to two different centers.

The integer program formulation, denoted by SL1, can be written as

$$\min_{\{X_{ilj}\},\{U_{il}\},\{P_{lj}\}} \quad \sum_{j\in\mathscr{I}}\sum_{l\in\mathscr{L}}\sum_{i\in\mathscr{I}} \delta_{ilj}X_{ilj} \tag{3.5.5}$$

$$\text{subject to} \quad \sum_{i\in\mathscr{I}} q_i U_{il} \leq v \qquad \forall\, l \in \mathscr{L} \tag{3.5.6}$$

$$\sum_{l\in\mathscr{L}} U_{il} = 1 \qquad \forall\, i \in \mathscr{I} \tag{3.5.7}$$

$$U_{il} + P_{lj} - 1 \leq X_{ilj} \qquad \forall\, i,\, j \in \mathscr{I},\, \forall\, l \in \mathscr{L} \tag{3.5.8}$$

$$X_{ilj} \leq U_{il} \qquad \forall\, i,\, j \in \mathscr{I},\, \forall\, l \in \mathscr{L} \tag{3.5.9}$$

$$X_{ilj} \leq P_{lj} \qquad \forall\, i,\, j \in \mathscr{I},\, \forall\, l \in \mathscr{L} \tag{3.5.10}$$

$$\sum_{j\in\mathscr{I}} P_{lj} = 1 \qquad \forall\, l \in \mathscr{L} \tag{3.5.11}$$

$$\sum_{j\in\mathscr{I}}\sum_{l\in\mathscr{L}} P_{lj} = d \tag{3.5.12}$$

In the above formulation, we have the usual capacity constraints (3.5.6) and the customer assignment constraints (3.5.7). The constraints (3.5.8), (3.5.9), (3.5.10) model the condition that if $X_{ilj} = 1$, then $U_{il} = 1$ and $P_{lj} = 1$. Constraints (3.5.11) only allow one customer to be chosen as a center. Finally, constraints (3.5.12) force the customers representing centers to be uniquely chosen.

During the course of our solution method, the above integer program reduces to a simpler problem when the customer centers are fixed. In such a case, we solve for the following mathematical program denoted by SL2:

$$\min_{\{U_{il}\}} \quad \sum_{l\in\mathscr{L}}\sum_{i\in\mathscr{I}} \delta_{il}U_{il} \tag{3.5.13}$$

$$\text{subject to} \quad \sum_{i\in\mathscr{I}} q_i U_{il} \leq v \qquad \forall\, l \in \mathscr{L} \tag{3.5.14}$$

$$\sum_{l\in\mathscr{L}} U_{il} = 1 \qquad \forall\, i \in \mathscr{I} \tag{3.5.15}$$

3.5.2 Solution Method

In Section 3.5.1, we formulated an integer program that can be considered as a discrete capacitated facility location problem with distances precomputed in a special way. We use the iterative

improvement scheme to obtain solutions. The outline of the procedure is given in Algorithm 4. This algorithm has an exponential worst-case complexity as it includes a branch-and-bound procedure for the integer program within the *while* loop. In the course of the algorithm, some customers represent centers of each cluster, which we call *center-customers.*

Algorithm 4 Slender Shape Clustering

Initialization. Initialize iteration counter, $t \leftarrow 0$. Randomly pick k customers to be *center-customers*.

- Compute δ_{il} induced by the initial choice of center-customers using equations 3.5.1, 3.5.2, 3.5.3, and 3.5.4.

while center-customers have changed

1. Find new assignments of customers to centers from integer program SL2.
2. Update centers $\left\{c_l^{t+1}\right\}_{l=1}^{d}$ as

   $c_l^{t+1} = m_l^{t+1}$ where m_l^{t+1} is the centroid of customers i such that $U_{il}^{t+1} = 1$.

3. Find closest customers to every center. Let these be new center-customers.

end

3.6 Computational Experiments

We first discuss the computational results for each solution approach separately in each subsection. Second, we make observations for comparing these methods in the last subsection. We applied the algorithms presented in the previous sections to small-sized splittable demand and unsplit demand cases. For large-scale problems, we only tested those instances where demand is allowed to be split, i.e. unit demand is assumed.

3.6.1 Data Set and Implementation

We implemented our algorithms in MATLAB-R2012a on a 64-bit Intel (R) Core (TM) i7-3770 CPU with 16GB RAM. At each iteration of the algorithm, an integer programming solver was called using the CPLEX (12.04)-MATLAB interface. The number of drivers is a free paramater to choose in our algorithms. We fix it at the minimum number of drivers to satisfy all delivery requests. We applied each algorithm ten times and pick the best clustering. We compared this method against a state-of-the-art vehicle routing solver in the VRPH library, the record-to-record vehicle routing

heuristic (Groër et al., 2010). Its effectiveness is very close to the best results obtained for several benchmarks thus far in the literature. The code for this metaheuristic is open source. We refer to it as the RTR heuristic. It can be considered as a state of the art for vehicle routing problems.

We tested our algorithm on several benchmark instances. We were mainly interested in problems with more realistic features regarding customer distribution and with a large number of stops. We use the benchmark problems in Gehring and Homberger (1999), but remove the time window constraints and treat the problems as capacitated problems like Mester and Bräysy (2007). Moreover, we assume all demand is equal to one. After such modifications, the benchmark can be categorized into clustered (c), random-clustered (rc), and random (r) problems. Within each category, there are problems that require many number of vehicles and less number of vehicles depending on the capacity parameter in each problem. We analyze routing policies with respect to customer distribution. We denote this testset as the Gehring-Homberger-unit testset.

Second, we use the benchmark of Augerat et al. (1995). These are smaller-sized problems, however the location of the depot is more varied with respect to the center of the data in each test instance. We hypothesize that the Slender algorithm objective parameters can be tuned to give better results depending on how the depot is located. We denote the original testset with A. We also modify the benchmark by assuming unit demand for all customers. We call this modified testset the A-unit testset.

3.6.2 Clustering with Manhattan Distance

We summarize our results in tables. The first column in each table is the name of the specific instance. The 'Cost' columns contain the VRP length of the solution delivered by each algorithm, RTR, k-medians, and 'k-medians + RTR'. The 'Gap' columns are percentage difference in VRP length of each algorithm with respect to the baseline (10 RTR). The tables contain the computation time for solving each instance under 'Time' as well as the number of vehicles used in the solutions under '# of vehicles'.

We ran Algorithm 2 on the benchmark instances. Table 3.6.1 contains results obtained from the Gehring-Homberger-unit testset. The results for the testset A and A-unit are in Table 3.6.3 and Table 3.6.2, respectively. On the small scale instances, the k-medians algorithm does 5% worse on average than the RTR solver. On the large scale instances, its performance on average is 4% worse.

One group of experiments involved using the routing solutions obtained from the k-medians algorithm as initial solutions provided to the RTR solver. We ran the RTR solver for 1 iteration on this initial solution and compared it to running the RTR 10 times with different initial solutions that it generates internally. We see that when the k-medians initial solution is used, its performance increases but still does worse by 1.9%. The poor performance of the k-medians algorithm can be attributed to several things. Considering the large scale random instances, the k-medians algorithm finds poor solutions that cannot be easily explained because they are visually similar to the metaheuristic solutions. Essentially, the metaheuristic RTR was able to find better solutions because of the neighborhood operations it uses to continuously improve a starting solution. Moreover, k-

medians is not well suited for random instances because the clustering solution is very susceptible to initial cluster centers. As the clustering tendency of the data increases as in "c1-2-unit" in Figure 3.6.1, it does a better job in finding the correct clusters. However, the metaheuristic has a competitive edge in finding hard-to-find improvements in routing through neighborhood search where the k-medians objective can be blind about. The north-south bound purple route on the left in the figure seems like an odd choice for a route, while the metaheuristic did not choose such a route.

The number of vehicles k-medians algorithm uses is equal to the minimum number required to satisfy total customer demand. For this reason, at times, the number of vehicles k-medians uses is one less than the number of vehicles used in the RTR solutions. The runtime for the k-medians algorithm is about twice as long as the RTR on average, 114.7 and 54.2 seconds respectively. When you consider the k-medians + RTR routine, runtime is on average 130.4 seconds. The k-medians algorithm is relatively slower because it spends time optimizing over the clustering objective with ten repetitions of the clustering algorithm.

	10 RTR			*k*-medians				*k*-medians + RTR			
Problem	Cost	# of vehicles	Time	Cost	Gap	Time	# of vehicles	Cost	Gap	# of vehicles	Time
c1_10_unit	37623.35	84	76.78	38953.66	0.04	255.94	84	38090.78	0.01	84	283.60
c1_2_unit	2470.42	17	13.28	2615.99	0.06	27.84	17	2542.73	0.03	17	31.14
c1_4_unit	6524.06	34	26.45	6738.12	0.03	47.68	34	6613.51	0.01	34	54.66
c1_6_unit	13562.36	55	41.01	14138.95	0.04	106.03	55	13702.66	0.01	55	118.01
c1_8_unit	22817.89	69	60.47	23600.79	0.03	173.28	67	22942.52	0.01	67	192.66
c2_10_unit	12516.75	21	228.77	12992.33	0.04	66.87	20	12774.81	0.02	20	98.46
c2_2_unit	1431.92	5	33.82	1487.54	0.04	31.58	4	1478.25	0.03	4	37.20
c2_4_unit	3013.49	9	78.69	3179.83	0.06	42.31	8	3175.08	0.05	8	53.36
c2_6_unit	5909.05	13	108.93	5937.90	0.005	67.39	12	5916.13	0.001	12	85.35
c2_8_unit	8907.03	17	175.80	9128.28	0.02	107.71	16	9064.81	0.02	16	135.39
r1_10_unit	43272.54	92	85.50	44225.03	0.02	271.91	91	43810.75	0.01	91	300.43
r1_2_unit	2878.12	17	13.38	3066.35	0.07	33.17	17	2990.53	0.04	17	36.45
r1_4_unit	6999.61	34	27.02	7305.38	0.04	66.62	34	7106.70	0.02	34	73.61
r1_6_unit	14864.88	51	41.80	15336.28	0.03	96.18	50	15180.97	0.02	50	108.33
r1_8_unit	26720.35	67	58.43	27427.12	0.03	226.19	67	27070.71	0.01	67	249.07
r2_10_unit	16680.81	20	110.89	17338.53	0.04	152.28	19	16926.99	0.01	19	185.96
r2_2_unit	1732.04	4	18.86	1670.43	-0.04	36.79	4	1661.74	-0.04	4	41.05
r2_4_unit	3597.71	9	39.69	3766.70	0.05	46.55	8	3761.08	0.05	8	55.97
r2_6_unit	6703.80	11	65.45	7205.88	0.07	76.06	11	7005.11	0.04	11	92.95
r2_8_unit	11303.65	15	87.36	11801.40	0.04	119.53	15	11585.29	0.02	15	145.05
rc1_10_unit	39763.97	84	77.88	41565.90	0.05	325.51	84	40529.80	0.02	84	354.25
rc1_2_unit	2771.88	17	13.30	2944.68	0.06	36.40	17	2804.50	0.01	17	39.72

Table 3.6.1: *K*-medians on testset Gehring-Homberger-unit.

	10 RTR			*k*-medians				*k*-medians + RTR			
Problem	Cost	# of vehicles	Time	Cost	Gap	Time	# of vehicles	Cost	Gap	# of vehicles	Time
rc1_4_unit	7056.94	34	27.41	7256.68	0.03	59.84	34	7091.61	0.00	34	67.00
rc1_6_unit	14862.49	55	43.70	15506.99	0.04	126.28	55	15184.53	0.02	55	138.98
rc1_8_unit	27041.27	74	65.99	27687.52	0.02	203.90	73	27240.53	0.01	73	223.65
rc2_10_unit	14730.63	18	138.24	15526.64	0.05	118.62	18	15152.95	0.03	18	151.53
rc2_2_unit	1583.85	5	18.77	1594.23	0.01	29.54	4	1593.32	0.01	4	34.10
rc2_4_unit	3339.25	9	39.83	3465.22	0.04	55.14	8	3418.53	0.02	8	64.47
rc2_6_unit	6029.80	12	69.47	6366.02	0.06	69.47	11	6244.78	0.04	11	86.24
rc2_8_unit	9897.86	16	95.33	10652.80	0.08	106.86	15	10256.43	0.04	15	131.98
-/+					1-, 0 , 24+				1-, 0 , 24+		
Ave.			54.25		0.0385	114.70			0.019		130.44

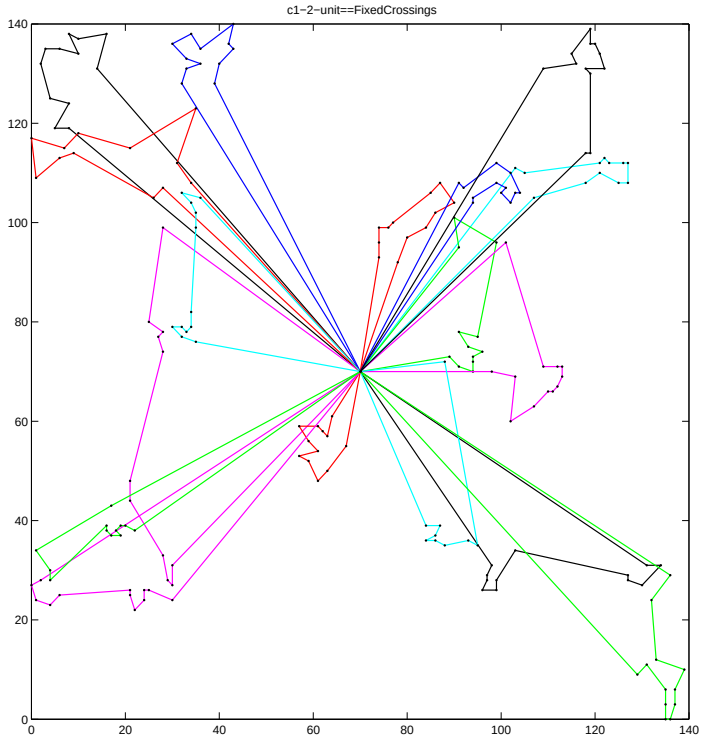
Table 3.6.1 continued.

Problem	RTR	# of vehicles	Time	K-medians	Kmedians Gap	Time	# of vehicles	Kmedians + RTR	Kmedians + RTR Gap	# of vehicles	Time
An32k5_unit	824	5	1.35	910.73	0.11	4.62	5	830	0.01	5	4.81
An33k5_unit	688	5	1.33	702.57	0.02	3.97	5	661	-0.04	5	4.13
An33k6_unit	749	6	1.32	791.26	0.06	4.04	6	740	-0.01	6	4.21
An34k5_unit	780	5	1.40	765.63	-0.02	4.57	5	765	-0.02	5	4.75
An36k5_unit	891	5	1.59	888.20	0.00	4.35	5	886	-0.01	5	4.53
An37k5_unit	697	5	1.65	741.70	0.06	5.08	5	700	0.00	5	5.30
An38k5_unit	709	5	1.77	803.55	0.13	4.79	5	706	0.00	5	5.01
An39k5_unit	820	5	1.92	949.12	0.16	5.04	5	893	0.09	5	5.26
An39k6_unit	867	6	1.89	948.29	0.09	4.43	6	894	0.03	6	4.64
An44k6_unit	929	6	2.41	963.32	0.04	5.22	6	930	0.00	6	5.47
An45k6_unit	903	6	2.43	918.77	0.02	4.48	6	912	0.01	6	4.76
An45k7_unit	1094	7	2.44	1190.58	0.09	4.94	7	1128	0.03	7	5.48
An46k7_unit	948	7	2.60	1025.94	0.08	5.15	7	963	0.02	7	5.44
An48k7_unit	1155	7	2.95	1155.89	0.00	4.50	7	1153	0.00	7	4.82
An53k7_unit	1027	7	3.44	1122.09	0.09	5.29	7	1060	0.03	7	5.66
An54k7_unit	1130	7	3.61	1167.15	0.03	5.21	7	1123	-0.01	7	5.59
An55k9_unit	1082	9	3.35	1168.87	0.08	5.07	9	1126	0.04	9	5.43
An60k9_unit	1283	9	3.89	1382.30	0.08	5.16	9	1301	0.01	9	5.56
An61k9_unit	984	9	3.89	1128.29	0.15	5.06	9	1012	0.03	9	5.47
An62k8_unit	1292	8	3.98	1365.51	0.06	5.29	8	1319	0.02	8	5.73
An63k10_unit	1229	9	3.99	1353.33	0.10	5.25	9	1289	0.05	9	5.70
An63k9_unit	1650	9	4.09	1675.60	0.02	5.57	9	1628	-0.01	9	5.99
An64k9_unit	1511	9	4.10	1572.38	0.04	5.52	9	1526	0.01	9	5.93
An65k9_unit	1076	8	4.06	1172.01	0.09	4.73	8	1157	0.08	8	5.15
An69k9_unit	1165	9	4.49	1234.59	0.06	7.01	9	1213	0.04	9	7.52
An80k10_unit	1853	10	5.02	1943.38	0.05	5.99	10	1921	0.04	10	6.56
-/+					2-, 0 , 24+				8-, 0 , 18+		
Ave.			2.883		0.065	5.012			0.017		5.342

Table 3.6.2: K-medians on testset A-unit.

Problem	RTR	# of vehicles	Time	K-medians	K-medians Gap	Time	# of vehicles	K-medians + RTR	K-medians + RTR Gap	# of vehicles	Time
An32k5	827	5	1.427	845.143	0.022	6.207	5	784	-0.052	5	6.377
An33k5	675	5	1.394	700.904	0.038	6.166	5	680	0.007	5	6.326
An33k6	743	6	1.353	811.191	0.092	5.391	6	743	0.000	6	5.555
An34k5	791	5	1.429	820.394	0.037	8.066	5	808	0.021	5	8.238
An36k5	805	5	1.610	877.203	0.090	7.282	5	805	0.000	5	7.508
An37k5	679	5	1.732	738.461	0.088	4.930	5	679	0.000	5	5.128
An37k6	960	6	1.702	1041.642	0.085	7.966	6	994	0.035	6	8.174
An38k5	736	5	1.818	734.683	-0.002	5.267	5	730	-0.008	5	5.470
An39k5	841	5	1.943	896.942	0.067	5.550	5	857	0.019	5	6.056
An39k6	855	6	1.930	872.781	0.021	5.932	6	834	-0.025	6	6.211
An44k6	957	7	2.388	1032.273	0.079	6.740	6	1029	0.075	6	7.020
An45k6	958	6	2.479	961.347	0.003	7.805	6	949	-0.009	6	8.112
An45k7	1194	7	2.653	1241.150	0.039	12.134	7	1203	0.008	7	12.487
An46k7	922	7	2.636	976.600	0.059	6.835	7	957	0.038	7	7.127
An48k7	1097	7	2.985	1143.703	0.043	8.405	7	1135	0.035	7	8.778
An53k7	1038	7	3.423	1131.947	0.091	8.068	7	1106	0.066	7	8.577
An54k7	1176	7	3.503	1288.568	0.096	8.768	7	1184	0.007	7	9.160
An55k9	1085	9	3.506	1174.356	0.082	8.750	9	1130	0.041	9	9.132
An60k9	1363	9	4.092	1444.983	0.060	8.918	9	1407	0.032	9	9.377
An61k9	1044	10	3.953	1144.257	0.096	16.235	9	1100	0.054	9	16.633
An62k8	1338	8	4.128	1392.653	0.041	8.935	8	1360	0.016	8	9.376
An63k10	1332	10	4.147	1404.603	0.055	9.225	10	1346	0.011	10	9.684
An63k9	1636	10	4.206	1666.067	0.018	10.426	9	1646	0.006	9	10.866
An64k9	1426	9	4.077	1486.082	0.042	9.790	9	1455	0.020	9	10.238
An65k9	1202	9	4.150	1246.687	0.037	10.029	9	1238	0.030	9	10.462
An69k9	1172	9	4.454	1258.295	0.074	10.660	9	1194	0.019	9	11.147
An80k10	1821	10	5.239	1872.799	0.028	11.525	10	1855	0.019	10	12.290
-/+					1-, 0 , 26+				4-, 3 , 20+		
Ave.			2.902		0.055	8.371			0.017		8.723

Table 3.6.3: *K*-medians on testset A.



(a) K-medians routing solution for highly clustered customers.

Figure 3.6.1: c1-2-unit.

3.6.3 Spherical Clustering

We summarize our results in tables. The first column in each table is the name of the specific instance. The 'Cost' columns contain the VRP length of the solution delivered by each algorithm, RTR, spherical, and 'spherical+ RTR'. The 'Gap' columns are percentage difference in VRP length of each algorithm with respect to the baseline (10 RTR). The tables contain the computation time for solving each instance under 'Time' as well as the number of vehicles used in the solutions under '# of vehicles'.

On the small size instances from Augerat et al. (1995), as shown in Table 3.6.5 and Table 3.6.6, spherical k-means results in routes which are 8.8% and 6.1% worse than the baseline. It is 5.4% worse than the baseline, on average, for the Gehring-Homberger-unit problems as can be seen in Table 3.6.4. If you consider the "spherical k-means + RTR" algorithm, then these values all decrease to 4.3%, 2.9% and 2.8% respectively.

Focusing on Gehring-Homberger-unit instances with prefix "c1", i.e., clustered, the spherical algorithm does particularly worse. This is because within those instances, there are "pockets" of locations that lie on the same ray from the depot but are in fact far from each other. Nevertheless, the spherical algorithm tends to cluster these locations together because they lie within a very narrow

interval of rays with respect to the depot. For those instances that start with “r”, i.e. random, the random scatteredness of the customers makes it suitable for the solution structure of the spherical algorithm to deliver better routes, and in some cases it even beats the RTR solver. For example, Figure 3.6.2 is a good example. Overall, however, the spherical algorithm does better than RTR solver in fewer cases for the Gehring-Homberger-unit.

Problem	RTR	# of vehicles	Time	Spherical	Spherical Gap	Time	# of vehicles	Spherical + RTR	Spherical + RTR Gap	# of vehicles	Time
c1_10_unit	37623.35	84	76.78	42963.95	0.142	51.46	84	41459.95	0.102	84	60.88
c1_2_unit	2470.41	17	13.28	2738.27	0.108	6.14	17	2669.95	0.081	17	7.47
c1_4_unit	6524.06	34	26.45	7184.92	0.101	12.13	34	6982.87	0.07	34	14.94
c1_6_unit	13562.36	55	41.01	15950.2	0.176	22.47	55	15245.3	0.124	55	26.96
c1_8_unit	22817.89	69	60.47	26467.54	0.16	33.46	67	25462.19	0.116	67	39.66
c2_10_unit	12516.75	21	228.77	12600.19	0.006	30.10	20	12556.07	0.003	20	56.22
c2_2_unit	1431.92	5	33.82	1451.54	0.01	5.80	4	1441.82	0.007	4	9.38
c2_4_unit	3013.49	9	78.69	2952.96	-0.02	8.15	8	2948.53	-0.02	8	16.37
c2_6_unit	5909.05	13	108.93	5662.57	-0.04	14.83	12	5640.28	-0.04	12	26.30
c2_8_unit	8907.03	17	175.80	8659.24	-0.02	16.55	16	8639.54	-0.03	16	16.12
r1_10_unit	43272.54	92	85.5	50553.6	0.168	41.33	91	47719.34	0.103	91	50.76
r1_2_unit	2878.12	17	13.38	3017.91	0.049	5.94	17	3001.23	0.043	17	7.37
r1_4_unit	6999.61	34	27.02	7640.79	0.092	11.39	34	7501.64	0.072	34	14.28
r1_6_unit	14864.88	51	41.8	16862.87	0.134	16.27	50	16134.34	0.085	50	20.84
r1_8_unit	26720.35	67	58.43	30470.61	0.14	25.1	67	29028.33	0.086	67	31.58
r2_10_unit	16680.81	20	110.89	15594.01	-0.065	16.88	19	15429.92	-0.075	19	28.89
r2_2_unit	1732.04	4	18.86	1710.02	-0.013	4.66	4	1705.16	-0.016	4	6.51
r2_4_unit	3597.71	9	39.69	3501.98	-0.027	6.52	8	3491.49	-0.03	8	10.41
r2_6_unit	6703.8	11	65.45	6423.5	-0.042	9.39	11	6413.78	-0.043	11	15.81
r2_8_unit	11303.65	15	87.36	10638.58	-0.059	14.02	15	10557.71	-0.066	15	22.82
rc1_10_unit	39763.97	84	77.88	46499.27	0.169	39.36	84	43060.32	0.083	84	48.35
rc1_2_unit	2771.88	17	13.3	2992.17	0.079	7.2	17	2838.38	0.024	17	8.58
rc1_4_unit	7056.94	34	27.41	7483.7	0.06	11.53	34	7444.01	0.055	34	14.34

Table 3.6.4: Spherical algorithm tested on benchmark Gehring-Homberger-unit.

Problem	RTR	# of vehicles	Time	Spherical	Spherical Gap	Time	# of vehicles	Spherical + RTR	Spherical + RTR Gap	# of vehicles	Time
rc1_6_unit	14862.49	55	43.7	17773.67	0.196	16.96	55	16386.83	0.103	55	21.63
rc1_8_unit	27041.27	74	65.99	31585.49	0.168	27.25	73	28948.9	0.071	73	33.52
rc2_10_unit	14730.63	18	138.24	14298.21	-0.029	18.69	18	14272.55	-0.031	18	30.2
rc2_2_unit	1583.85	5	18.77	1602.27	0.012	4.63	4	1599.23	0.01	4	6.47
rc2_4_unit	3339.25	9	39.83	3269.63	-0.021	6.15	8	3260.22	-0.024	8	10.06
rc2_6_unit	6029.8	12	69.47	6119.6	0.015	9.64	11	6110.05	0.013	11	16.22
rc2_8_unit	9897.86	16	95.33	9725.49	-0.017	14.66	15	9648.32	-0.025	15	23.11
-/+					11-, 0 , 19+				11-, 0 , 19+		
Ave.			54.252		0.054	17.329			0.028		22.866

Table 3.6.4 continued.

Problem	RTR	# of vehicles	Time	Spherical	Spherical Gap	Time	# of vehicles	Spherical + RTR	Spherical + RTR Gap	# of vehicles	Time
An32k5	827	5	1.43	932.31	0.127	4.7	5	784	-0.052	5	4.93
An33k5	675	5	1.39	737.05	0.092	3.54	5	728	0.079	5	3.7
An33k6	743	6	1.35	785.63	0.057	4.99	6	784	0.055	6	5.29
An34k5	791	5	1.43	793.8	0.004	5	5	787	-0.005	5	5.23
An36k5	805	5	1.61	943.44	0.172	4.44	5	851	0.057	5	4.65
An37k5	679	5	1.73	736.44	0.085	4.63	5	670	-0.013	5	4.84
An37k6	960	6	1.7	1078.15	0.123	4.47	6	1054	0.098	6	4.67
An38k5	736	5	1.82	774.74	0.053	5.43	5	742	0.008	5	5.66
An39k5	841	5	1.94	870.87	0.036	6.37	5	842	0.001	5	6.6
An39k6	855	6	1.93	960.81	0.124	4.82	6	857	0.002	6	5.05
An44k6	957	7	2.39	1056.86	0.104	5.68	6	982	0.026	6	5.94
An45k6	958	6	2.48	1094.89	0.143	6.97	6	1062	0.109	6	7.25
An45k7	1194	7	2.65	1273.89	0.067	8.93	7	1274	0.067	7	9.23
An46k7	922	7	2.64	950.23	0.031	4.97	7	945	0.025	7	5.26
An48k7	1097	7	2.98	1174.16	0.07	4.94	7	1144	0.043	7	5.28
An53k7	1038	7	3.42	1067.08	0.028	6.95	7	1034	-0.004	7	7.3
An54k7	1176	7	3.5	1253.94	0.066	12.23	7	1246	0.06	7	12.61
An55k9	1085	9	3.51	1176.47	0.084	9.72	9	1144	0.054	9	10.12
An60k9	1363	9	4.09	1543.97	0.133	141.03	9	1518	0.114	9	141.46
An61k9	1044	10	3.95	1181.17	0.131	27.05	9	1129	0.081	9	27.49
An62k8	1338	8	4.13	1410.99	0.055	33.02	8	1349	0.008	8	33.45
An63k10	1332	10	4.15	1468.99	0.103	85.18	10	1440	0.081	10	85.63
An63k9	1636	10	4.21	1782.53	0.09	45.83	9	1760	0.076	9	46.29
An64k9	1426	9	4.08	1660.91	0.165	53.58	9	1488	0.043	9	54.05
An65k9	1202	9	4.15	1231.72	0.025	19.66	9	1222	0.017	9	20.09
An69k9	1172	9	4.45	1249.3	0.066	11.07	9	1227	0.047	9	11.55
An80k10	1821	10	5.24	2076.5	0.14	297.53	10	1961	0.077	10	298.1
-/+					0-, 0 , 27+				4-, 0 , 23+		
Ave.			2.902		0.088	30.471			0.043		30.804

Table 3.6.5: Spherical algorithm tested on benchmark A.

Problem	RTR	# of vehicles	Time	Spherical	Spherical Gap	Time	# of vehicles	Spherical + RTR	Spherical + RTR Gap	# of vehicles	Time
An32k5_unit	824	5	1.35	920.14	0.117	2.64	5	847	0.028	5	2.81
An33k5_unit	688	5	1.33	676.32	-0.017	2.21	5	647	-0.06	5	2.37
An33k6_unit	749	6	1.32	771.33	0.03	2.38	6	749	0	6	2.53
An34k5_unit	780	5	1.4	808.76	0.037	2.25	5	785	0.006	5	2.42
An36k5_unit	891	5	1.59	887.15	-0.004	2.2	5	884	-0.008	5	2.41
An37k5_unit	697	5	1.65	703.31	0.009	2.52	5	701	0.006	5	2.72
An38k5_unit	709	5	1.77	737.09	0.04	2.84	5	691	-0.025	5	3.07
An39k5_unit	820	5	1.92	888.38	0.083	2.32	5	880	0.073	5	2.56
An39k6_unit	867	6	1.89	906.58	0.046	2.49	6	893	0.03	6	2.7
An44k6_unit	929	6	2.41	988.49	0.064	2.58	6	980	0.055	6	2.84
An45k6_unit	903	6	2.43	972.47	0.077	2.42	6	964	0.068	6	2.77
An45k7_unit	1094	7	2.44	1209.74	0.106	2.2	7	1134	0.037	7	2.5
An46k7_unit	948	7	2.6	1027.2	0.084	2.26	7	966	0.019	7	2.55
An48k7_unit	1155	7	2.95	1175.86	0.018	2.22	7	1162	0.006	7	2.53
An53k7_unit	1027	7	3.44	1082.69	0.054	2.29	7	1046	0.019	7	2.65
An54k7_unit	1130	7	3.61	1198.23	0.06	2.19	7	1193	0.056	7	2.56
An55k9_unit	1082	9	3.35	1174.28	0.085	2.18	9	1134	0.048	9	2.55
An60k9_unit	1283	9	3.89	1416.9	0.104	2.89	9	1354	0.055	9	3.29
An61k9_unit	984	9	3.89	1027.96	0.045	2.7	9	1019	0.036	9	3.15
An62k8_unit	1292	8	3.98	1494.63	0.157	2.82	8	1440	0.115	8	3.3
An63k10_unit	1229	9	3.99	1286.03	0.046	4	9	1268	0.032	9	4.52
An63k9_unit	1650	9	4.09	1785.93	0.082	2.55	9	1736	0.052	9	3.04
An64k9_unit	1511	9	4.1	1674.11	0.108	2.1	9	1549	0.025	9	2.57
An65k9_unit	1076	8	4.06	1100.55	0.023	2.47	8	1096	0.019	8	2.89
An69k9_unit	1165	9	4.49	1173.39	0.007	2.93	9	1164	-0.001	9	3.39
An80k10_unit	1853	10	5.02	2079.18	0.122	2.68	10	1974	0.065	10	3.29
-/+					2-, 0 , 24+				4-, 1 , 21+		
Ave.			2.883		0.061	2.513			0.029		2.845

Table 3.6.6: Spherical algorithm tested on benchmark A-unit.

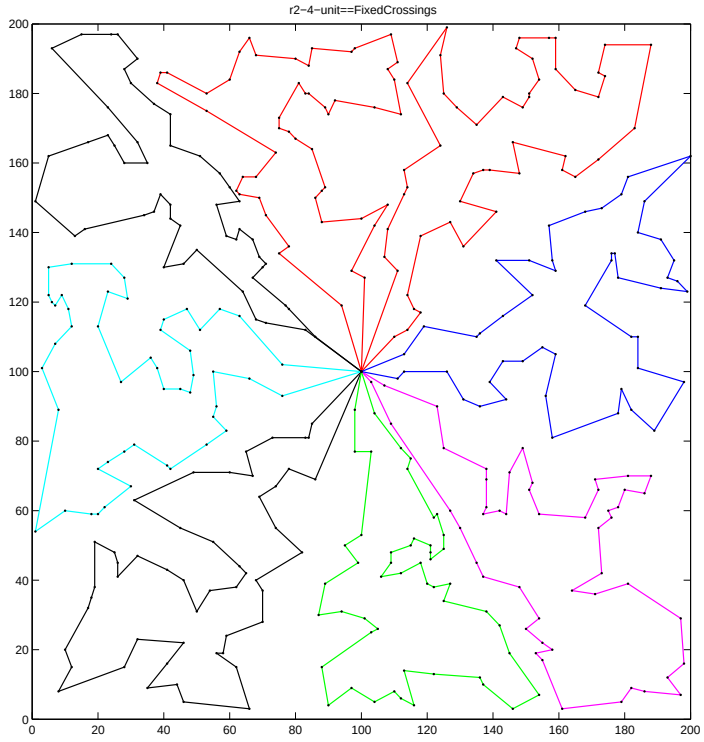


Figure 3.6.2: Spherical routing with VRP cost 3501 for Gehring-Homberger-unit benchmark instance r2-4-unit.

3.6.4 Slender-based Clustering

We first test the hypothesis that higher emphasis on angular distance delivers lower cost routing. Then, we carry out experiments to see the effect of different α_{il}^1, α_{il}^2 for i, l.

3.6.4.1 Uniform Bias for Slenderness

We conduct a group of experiments to see how various parameters in the objective affected the routing solution. First, we test for a single coefficient of θ_{il} in equation (3.5.1), in other words, a single scalar in $(0, 1)$ for all i, l $(j = l)$. We do not expect to see that a small emphasis on θ_{il} would deliver better results because we observed that routing solutions in which vehicles cover a large angle tends to be more costly. We choose four distinct values for $\alpha_{il} \in \{0.4, 0.5, 0.7, 0.9\}$. Table 3.6.7 contains the average gap to the best RTR result from ten runs for each benchmark A-unit, A, and Gehring-Homberger-unit. Additionally, it shows how many times the slender algorithm was better and the run-time of the slender algorithm as well as the run-time for the baseline algorithm in seconds.

Across all instances from different benchmarks, a higher the emphasis on θ_{il} meant better routing. Moreover, it turns out that it is easier to solve the optimization problems within the slender algorithm when θ_{il} takes a higher value. In Table 3.6.7, we see this as decreasing run-time as α_{il} increases. When there is unit demand (benchmark A-unit and Gehring-Homberger-unit), the slender algorithm tends to do better than when there is heterogeneous demand (benchmark A). The slender algorithm with $\alpha_{il} = 0.9$ does better than the baseline in nine instances for the Gehring-Homberger-unit, in one instance for benchmark A and in one instance for benchmark A-unit. The "slender + RTR" algorithm with $\alpha_{il} = 0.9$ does better than the baseline in 11 instances for the Gehring-Homberger-unit, in four instances for benchmark A and in six instances for benchmark A-unit. In all cases, the slender algorithm and the "slender + RTR" does better in fewer cases than the baseline algorithm (10 RTR).

		RTR	$\alpha_{il} = 0.4$		$\alpha_{il} = 0.5$		$\alpha_{il} = 0.7$		$\alpha_{il} = 0.9$	
			Slender	Slender + RTR	Slender	Slender + RTR	Slender	Slender + RTR	Slender	Slender + RTR
A-unit	ave. gap		0.13	0.033	0.106	0.027	0.056	0.013	0.044	0.017
	better(-)/worse(+)		0-, 0 , 26+	3-, 0 , 23+	0-, 0 , 26+	3-, 1 , 22+	0-, 0 , 26+	5-, 0 , 21+	1-, 0 , 25+	6-, 0 , 20+
	time (sec.)	2.883	4.823	5.165	4.088	4.429	3.367	3.703	3.482	3.823
A	ave. gap		0.155	0.05	0.122	0.046	0.063	0.025	0.067	0.028
	better(-)/worse(+)		0-, 0 , 27+	2-, 0 , 25+	0-, 0 , 27+	3-, 1 , 23+	0-, 0 , 27+	5-, 1 , 21+	1-, 0 , 26+	4-, 0 , 23+
	time (sec.)	2.902	20.502	20.841	12.901	13.245	8.582	8.917	8.055	8.391
Gehring-Homberger-unit	ave. gap		0.112	0.066	0.087	0.058	0.047	0.027	0.019	0.001
	better(-)/worse(+)		0-, 0 , 30+	0-, 0 , 30+	0-, 0 , 30+	0-, 0 , 30+	1-, 0 , 29+	2-, 0 , 28+	9-, 0 , 21+	11-, 1 , 17+
	time (sec.)	54.252	40.746	48.234	38.517	45.979	35.078	42.414	34.787	42.265

Table 3.6.7: All benchmark results for uniform bias coefficients.

3.6.4.2 Slenderness Based on Distance to Depot

In the previous section, it is evident that the emphasis on the angular distance needs to be high in order to obtain better routings. In this section, we focus on this aspect and carry out computations with high emphasis on angular distance, but with a finer set of parameters. In addition, we wanted to see whether differentiation per customer for α_{il} would improve the results. The intuition is that those customers farther away from the depot should have a higher emphasis on their θ_{il} in the objective function of the clustering problem so that the vehicle visiting them would not cover a large angle far from the depot leading to a long distance route.

The important part in this algorithm are the coefficients: $(\alpha_{il}\theta_{il} + (1-\alpha_{il})\rho_{il})$ for every i,l. The parameter space we searched was:

$$\alpha_{il} = \frac{\rho_i}{max(\rho_i)} \times b + a$$

a	0.3	0.3	0.5	0.5	0.5	0.6	0.6	0.7	0.7	0.8	0.8
b	0.6	0.65	0.3	0.4	0.45	0.3	0.35	0.2	0.25	0.15	0.19

If we look at Table 3.6.8, we see that, as compared to k-medians solutions, the range of parameters where we do marginally better is towards higher a and higher b. In other words, a good value for a would be around 0.5 or 0.6. So, we may say, when there is heterogeneous demand the angularity effect of good routes diminishes so that $a = 0.5$ turns out to be one of the best parameter settings. Note, these are only comparing the clustering solutions (cluster-first, route-second, NO RTR) to k-medians. The number of instances where slender does better than the k-medians solution is less frequent than the same problems with unit demand which we can see in Figure 3.6.3. In the figure, the x-axis measures how far the depot is from the center of the data, i.e., if the depot is closer to the boundary of the data points, then the measure is closer to 1. As for the 'A-unit' benchmark, when we compare the routing obtained from the k-medians to each of the slender-clustering methods with different parameters, we can make several observations. First, the better parameters tend to be the ones with higher values for $a \sim 0.7, 0.8$. Second, across all the parameter configurations where the slender-clustering did relatively better, it tends to consistently do worse on the test instances with the depot closer to the boundary of the dataset. So, there is a more clean relationship between the parameter settings which do worse and the location of the depot. This is reflected by the higher slope of the trend lines in Figure 3.6.4. In this case, the slender algorithm can be doing something wrong systematically. We see in Table 3.6.8 for the Gehring-Homberger-unit dataset that the parameters that do best have high values for a, $\{a,b\} = \{(0.7,0.2),(0.8,0.15)\}$. However, this problem set has their depot centered very near the center of the data. The slender algorithm beats k-medians solution 60% of the time and it delivers a routing cost that is, on average, 0.65% better. Other parameter settings deliver very close results on average which are less than 1% better than k-medians.

Our second group of experiments involved using our slender-based routing solutions as initial solutions for the RTR solver and comparing this to the RTR solver rerun 10 times with different

initial solutions which are internally found by the solver. The latter is the baseline for these comparisons. For the benchmark A, the slender algorithm beats the baseline, again, on a similar range of parameter settings, where the best ones occur at $a = \{0.5, 0.8\}$. This can be seen by looking at Table 3.6.9. If we analyze Table 3.6.9, we can observe that the slender algorithm did best for parameters $\{a,b\} = \{(0.8,0.19),(0.8,0.15),(0.5,0.45)\}$; for a taking value 0.5, it tends to do better on problem instances where the depot is closer to the boundary . The slender algorithm beats the baseline RTR in about 1/4 of the cases, and on average has worse VRP cost by 2.5%. As for the A-unit instances, the best parameters tend to lie towards high a. The slender algorithm beats the baseline in about 1/3 of the cases. In the best parameter configuration, $\{a,b\} = \{(0.7,0.25)\}$, slender-algorithm tends to do worse on cases with depot closer to the boundary. The slender algorithm is worse than the baseline on average by 1.35% in the best parameters, but only 1% worse for $\{a,b\} = \{(0.6,0.35)\}$. Third, on the Gehring-Homberger-unit instances, the best results are obtained again for similar parameter values, $\{a,b\} = \{(0.7,0.2),(0.8,0.15),(0.8,0.19)\}$. For these parameters, the VRP cost is on average less than 1% worse than the baseline.

a	b	-	0	+
0.3	0.6	4	0	23
0.3	0.65	4	0	23
0.5	0.3	12	0	15
0.5	0.4	8	1	18
0.5	0.45	12	0	15
0.6	0.3	12	0	15
0.6	0.35	9	0	18
0.7	0.2	9	0	18
0.7	0.25	9	0	18
0.8	0.15	8	0	19
0.8	0.19	8	0	19

(a) Benchmark A

a	b	-	0	+
0.3	0.6	8	0	18
0.3	0.65	7	0	19
0.5	0.3	12	0	14
0.5	0.4	15	0	11
0.5	0.45	12	1	13
0.6	0.3	15	1	10
0.6	0.35	16	0	10
0.7	0.2	18	1	7
0.7	0.25	16	0	10
0.8	0.15	14	0	12
0.8	0.19	13	0	13

(b) Benchmark A-unit

a	b	-	0	+
0.3	0.6	2	0	28
0.3	0.65	2	0	28
0.5	0.3	2	0	23
0.5	0.4	7	0	23
0.5	0.45	5	0	25
0.6	0.3	13	0	17
0.6	0.35	14	0	16
0.7	0.2	18	0	12
0.7	0.25	15	0	15
0.8	0.15	17	0	13
0.8	0.19	14	0	16

(c) Benchmark Gehring-Homberger-unit

Table 3.6.8: Slender algorithm better(-) or worse(+) compared to *k*-medians.

a	b	-	0	+
0.3	0.6	1	0	26
0.3	0.65	3	0	24
0.5	0.3	2	0	25
0.5	0.4	1	0	26
0.5	0.45	5	0	22
0.6	0.3	1	3	23
0.6	0.35	4	1	22
0.7	0.2	3	2	22
0.7	0.25	4	0	23
0.8	0.15	5	0	22
0.8	0.19	5	1	21

(a) Benchmark A

a	b	-	0	+
0.3	0.6	3	1	22
0.3	0.65	7	0	19
0.5	0.3	7	0	19
0.5	0.4	8	0	18
0.5	0.45	7	0	19
0.6	0.3	7	1	18
0.6	0.35	8	1	17
0.7	0.2	6	1	19
0.7	0.25	9	1	16
0.8	0.15	7	0	19
0.8	0.19	4	1	21

(b) Benchmark A-unit

a	b	-	0	+
0.3	0.6	1	0	29
0.3	0.65	1	0	29
0.5	0.3	1	0	24
0.5	0.4	1	0	29
0.5	0.45	3	0	27
0.6	0.3	2	0	28
0.6	0.35	3	0	27
0.7	0.2	3	0	27
0.7	0.25	12	0	18
0.8	0.15	13	0	17
0.8	0.19	14	0	16

(c) Benchmark Gehring-Homberger-unit

Table 3.6.9: Slender algorithm better(-) or worse(+) compared to justRTR.

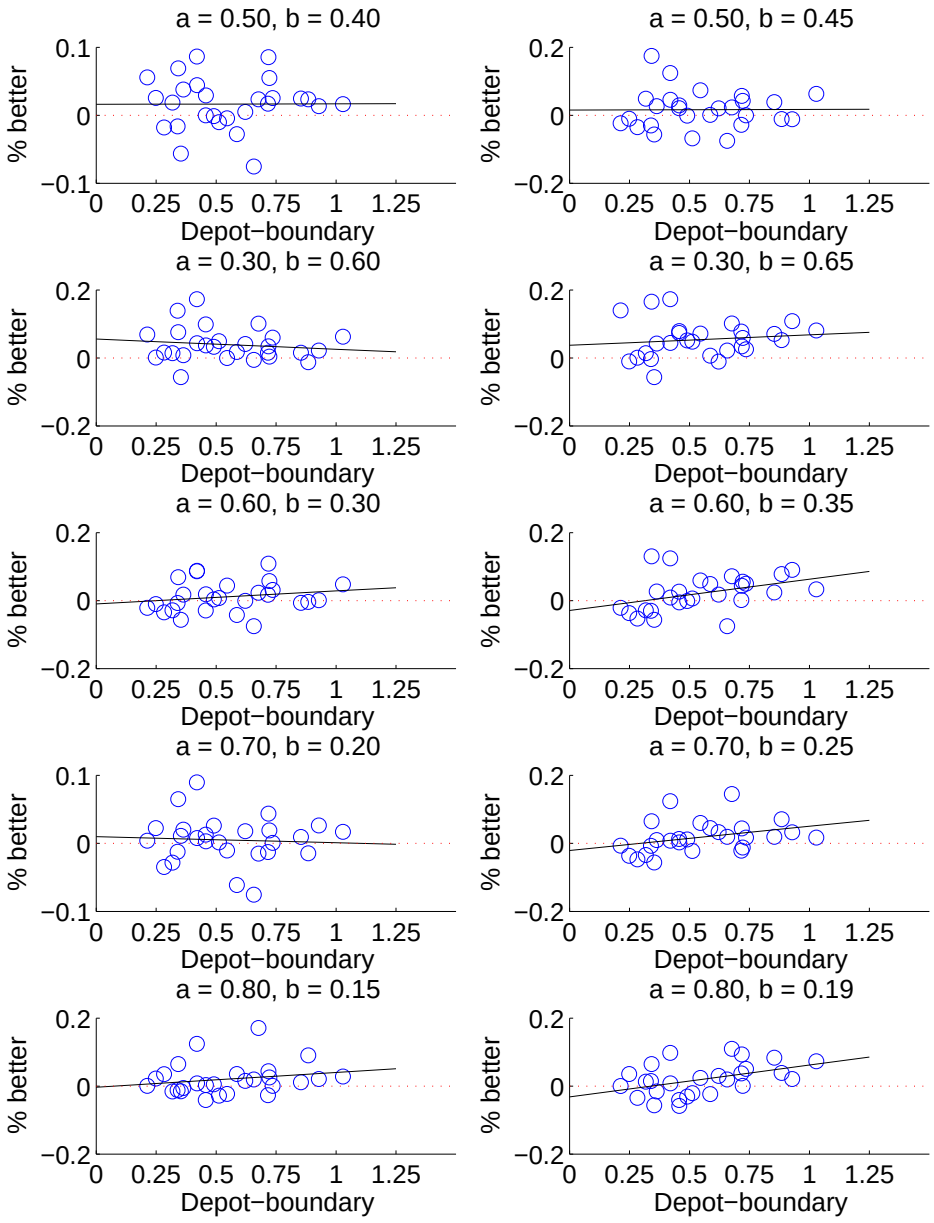


Figure 3.6.3: Slender algorithm % better than k-medians. Benchmark A.

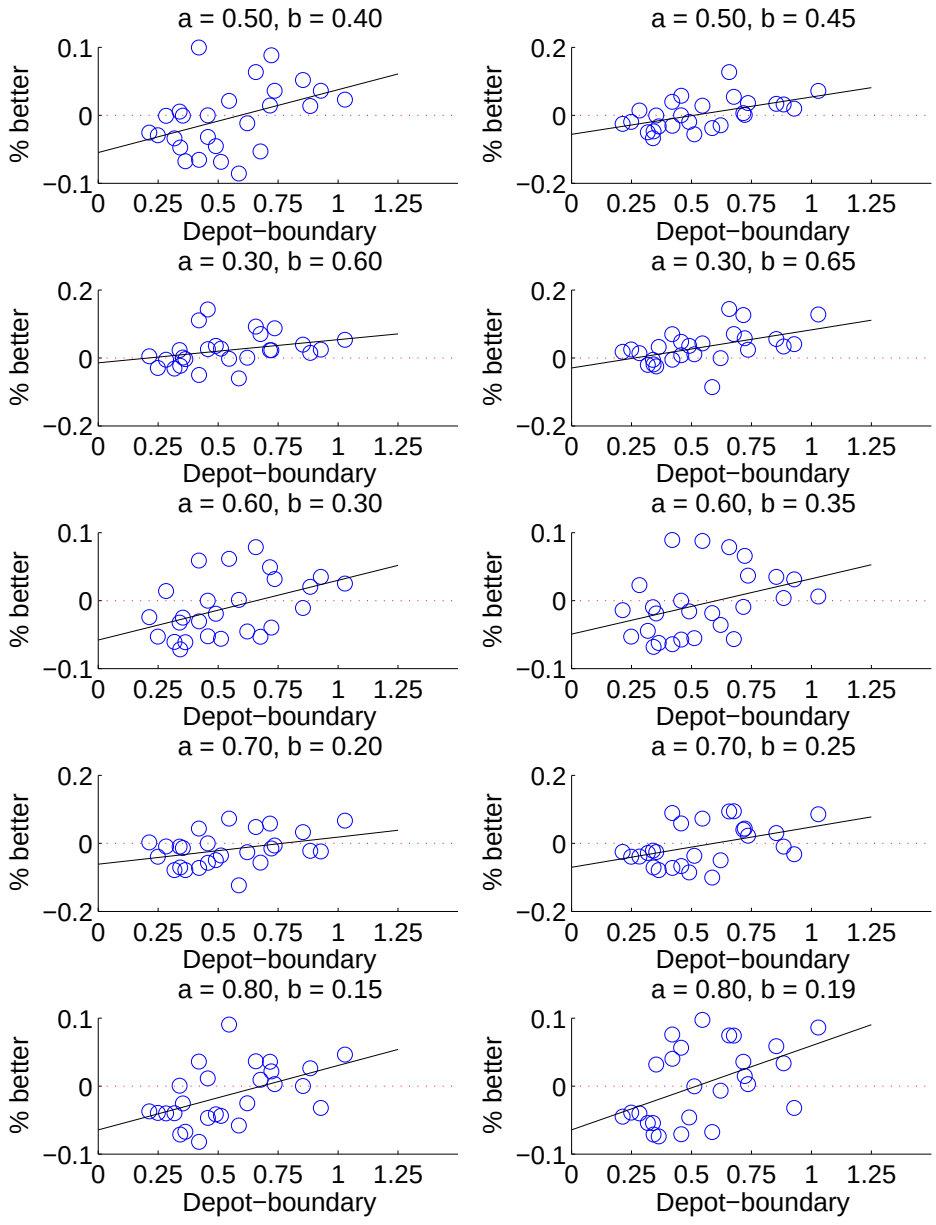


Figure 3.6.4: Slender algorithm % better than k-medians. Benchmark A-unit.

3.6.5 Results on Very Large Instances

In this section, we focus on the scalability of the SL method. SL and SL-RTR's greatest value comes when considering very large scale instances. To show this, we focus on the runtime of the SL algorithm in which the best clustering out of 10 is chosen and then routed. We use test instances composed of very large size instances. For instance, waste collection and mail and newspaper

delivery are some areas where very large instances can be found in practice (Kytöjoki et al., 2007). We test the SL algorithm on these instances and compare its runtime to RTR to see if we can observe a divergence in runtime.

We generate 18 very large problems, where customers are partially clustered and partially uniformly distributed. Then we test the SL and RTR heuristic on each instance. Each instance contains a number of customers from the set $\{2000, 3000, 5000\}$ and a minimum number of vehicles from the set $\{20, 50, 100\}$. See Table 3.6.10 for specific customer size, number of vehicles, and instance size for every test instance. The minimum number of vehicles is computed by dividing total demand by vehicle capacity.

In Figure 3.6.5, we show the runtime of the Slender algorithm versus the RTR heuristic on these instances as well as the GH-u instances. On the x-axis, we measure instance size by the number of customers multiplied by the number of vehicles. The y-axis shows the computation time in log(seconds). The squares represent runtime for an instance of the RTR. The stars represent the same for the Slender algorithm.

The Slender algorithm is more scalable than RTR in runtime (in log seconds) because not only does its runtime grow close to linearly, but is significantly less on these very large problems with a minor loss in solution quality. To the contrary, the runtime for the RTR heuristic is instance dependent and highly variable. The RTR spends much time on instances that have a higher number of customers and fewer vehicles. These instances require more customers per route. Table 3.6.10 contains the relative solution quality (positive gap means RTR better) and runtime of the SL algorithm for every instance. Instance size is measured by the number of customers times the number of vehicles. The shape parameter chosen was $\alpha_{il}^{1} = 0.9$.

The value of the clustering approach is greatest when we consider its solutions as initial feasible solutions for RTR. The SL-RTR algorithm does marginally better than the RTR on these very large instances (Table 3.6.10) versus the GH-u instances (Table 3.6.7). Figure 3.6.5 reflects the disadvantages of using a metaheuristic in that it requires tuning across the parameters (number of customers, number of vehicles) representing the problem instances. Our approach mitigates this and is less susceptible to variations in the number of customers and the number of vehicles.

Problem	# of Customers	Size	SL gap	SL # vehicles	SL sec.	SL-RTR gap	SL-RTR # vehicles	SL-RTR sec.	RTR sec.
vl-1	2000	40000	0.035	20	57.88	-0.002	20	107.79	502.49
vl-2	2000	40000	0.062	20	60.11	0.013	20	109.82	498.32
vl-3	3000	60000	0.051	20	94.61	0.001	20	246.52	1585.40
vl-4	2000	60000	0.033	20	101.14	-0.026	20	249.75	1488.34
vl-5	2000	100000	0.022	50	131.84	-0.006	50	152.31	231.87
vl-6	5000	100000	0.051	20	179.01	-0.012	20	828.89	6646.81
vl-7	5000	100000	0.057	50	133.24	0.005	50	152.92	206.15
vl-8	5000	100000	0.046	20	178.05	-0.020	20	840.18	6378.33
vl-9	3000	150000	0.025	50	222.93	-0.016	50	268.45	511.87
vl-10	3000	150000	0.030	50	221.24	-0.007	50	265.69	478.63
vl-11	2000	200000	0.030	100	287.71	0.000	100	304.35	183.30
vl-12	2000	200000	0.038	100	292.82	0.004	100	309.83	170.92
vl-13	5000	250000	0.029	50	455.20	-0.018	50	600.52	1532.73
vl-14	5000	250000	0.039	50	408.82	-0.011	50	555.07	1479.50
vl-15	3000	300000	0.028	100	480.67	-0.007	100	512.49	377.23
vl-16	3000	300000	0.035	100	470.73	-0.006	100	503.36	324.63
vl-17	5000	500000	0.030	100	1038.77	-0.007	100	1117.65	840.13
vl-18	5000	500000	0.017	100	879.33	-0.018	100	957.65	823.17
average			0.036		316.34	-0.007		449.07	1347.77
deviation			0.012		273.34	0.009		310.47	1943.55

Table 3.6.10: Very large instances.

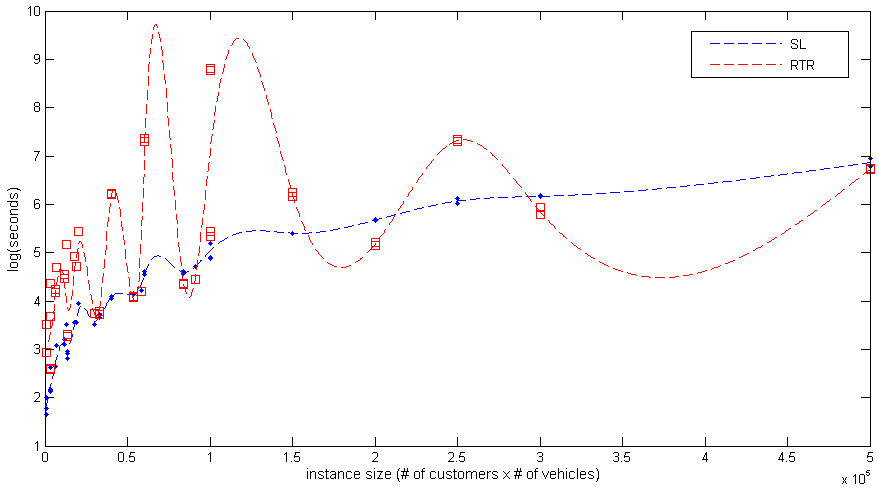


Figure 3.6.5: Runtime for Slender and RTR. Squares represent RTR and stars represent the Slender algorithm.

3.6.6 General Discussion

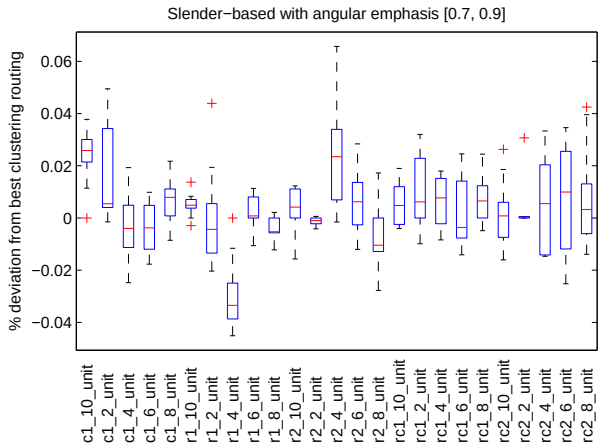
There are three methods used in this chapter. Each has strengths and weaknesses in terms of capturing clusters that should go together in a route. The k-medians algorithm in Section [[# does better on average across the large-scale benchmark instances. When it does worse than RTR, it does not do as worse as spherical clustering. Additionally, it tends to do more worse on instances with low number of vehicles than instances with a high number of vehicles. In contrast, the spherical algorithm does particularly well when the number of drivers is relatively low. Also, it performs better when customers are randomly dispersed. We can see this when we compare it against the RTR solver for the random instances (r1) and some random-clustered (rc1, rc2) instances. However, when the number of drivers is relatively large then spherical clustering tends to deliver routes that are too thin with respect to the depot, and hence does poorly. Nevertheless, k-medians clustering results tend to be on average better than spherical clustering.

Analyzing the routing solutions that RTR delivers on its own, it is evident that there needs to be some overlap of routes as the number of drivers increases in order to be efficient. For this reason, the slender-based clustering algorithm tends to do better than spherical and marginally better than k-medians for some choice of parameters. This is because the routings from the slender-based algorithm respects the tendency for efficient routes to be thinner but occasionally overlap.

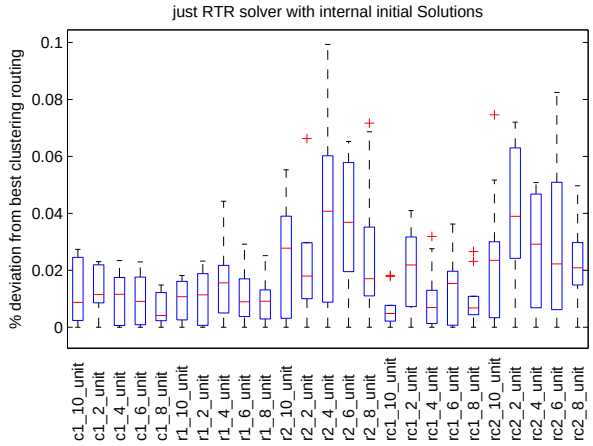
The implications of the above observations suggest that measuring how randomly customers are scattered in a service region can help determine the correct clustering method to use. When customers are highly clustered, methods like k-medians clustering possibly coupled with local search heuristics is preferable. On the other hand, when there is more randomness in how customers are dispersed, then a clustering algorithm that emphasizes the angularity of clusters tends to be preferable. The measurement of randomness of customer dispersion can be automated on computers by

using indexes like the Silhouette coefficient (Rousseeuw, 1987). Based on this index, the preferred algorithm can be applied.

Additionally, the clustering solutions obtained are susceptible to the local optima trap. For this reason, the vehicle routing solutions as a result of clustering are susceptible to variation also. We tested the degree of this variation by routing each clustering obtained from the clustering algorithm instead of choosing the best clustering and calculating the routing afterwards. We give the boxplot of the variation for a choice of parameters that did the best for the slender-based algorithm in Figure 3.6.6a and the initial solution susceptibility of the RTR solver in Figure 3.6.6b.



(a) Slender-based solution variation.



(b) RTR solver solution variation.

Figure 3.6.6: VRP cost variation.

We see from the boxplots that the initial solution susceptibility is smaller for the slender-based

clustering although the RTR solver performs better in terms of routing cost. However, there is significant variation in the routing costs for the clustering method which suggests that paralellization can be beneficial. For instance, the clustering method could be applied to the problem ten times which outputs ten different clusterings. We can process each single clustering on a separate processor and apply TSP routing to those clusters on separate processors. This could further improve the run-time of the algorithm.

3.7 Conclusion

Overall, the clustering techniques discussed in this paper are applicable in day-to-day routing. They are capable of delivering zones within the service region for every business day given customer demand for that day. The efficiency with which they do routing is slightly worse than a state-of-the-art vehicle routing solver. Clustering methods can benefit parcel delivery companies with various business goals other than minimum cost routing. For example, vehicle dispatchers carry out routing subject to contiguity considerations for service regions in a dynamic setting. This is a recent trend in big logistics companies such as the United Parcel Services (UPS) where they value contiguity restrictions for routes served by their drivers. In this light, clustering is more likely to be beneficial within this broader set of objectives.

Chapter 4

Dynamic Consistency

4.1 Introduction

Our focus in this chapter is to consider the issue of consistency. Driver-customer consistency depends on the history of the service provided to a customer by different drivers. We model the problem as a dynamic clustering (assignment) problem. We present clustering techniques that assign customers to drivers in a consistent manner across time and deliver compact clusters. This dynamic construction of consistent clusters would be useful for a cluster-first, route-second approach to vehicle routing problems with multiple objectives such as routing cost and consistency. We use historical information on driver services to find those drivers who have served each customer most of the time. We propose two solution methods that assign drivers to customers that takes this information into account in order to create consistency. The model we define in this paper is only partially solved. Specifically, no routing is considered.

We do not know of any other study that has quantified the degree of consistency using entropy related measures and studied it in a dynamic model with no fixed routing or pre-clustering of customers. Our work is unique in that it models the consistency problem in a "soft" way rather than imposing constraints (Groer et al., 2009) or exclusively assigning regions to drivers beforehand (Zhong et al., 2007).

In Section 4.2, we introduce concepts for consistency and the dynamic vehicle routing problem with stochastic customers who request delivery. We focus on creating consistency using clustering techniques, but do not consider routing customers to obtain a full VRP solution. In Section 4.3, we present two solution approaches that can create consistency over a period of time. Simulation results will be delivered in Section 4.4. The final section, Section 4.5, will contain our thoughts on possible future work and concluding remarks.

4.2 Concepts and Model

4.2.1 Consistency

Customer preferences toward consistent service can depend on various things. Customers tend to value the same or a small set of drivers delivering their packages over time (Campbell and Thomas, 2008b). Additionally, customers prefer to receive their packages when they are home and most readily available. In the VRP literature, this preference is usually modeled as constraints on the time window when a customer can accept his or her delivery. We think about these preferences as a long-term utility that the customer accrues, so it is a function of the history of service a customer receives. We summarize the history of driver assignments to a customer as a vector of the frequency with which a driver has visited the customer.

Typically, customers are loyal to the logistics companies that they receive service from. This is true unless customer preferences are not satisfied for a significant amount of time. The likelihood for customers switching their preferred logistics provider should increase the more the service they receive gives them incentive to search for other delivery options, holding all other factors constant. Those logistics companies that do not want to lose their customer base need to provide as high customer satisfaction as possible and in a *consistent* manner over a length of time as long as a customer is a client (Bowersox et al., 2002; Smith, 2002, 2004; Clinton, 2011; UPS, 2007).

A measure that quantifies consistency should be characterized with certain properties. We explain the properties a measure should have in an example. Assume we are given a customer in a region that is served by a logistics company that operates with 3 different drivers. Assume the customer has been loyal for 7 periods with this company. Let us consider the following 3 different ways in which this customer's demand can be satisfied with the 3 drivers.

$$\text{Solution 1: } (1,1,1,1,1,1,1)$$

$$\text{Solution 2: } (1,2,1,2,1,2,1)$$

$$\text{Solution 3: } (1,1,2,1,2,3,1)$$

If you look at Solution 1, you see that only 1 driver has served the customer whenever the customer requested delivery. This case should correspond to the most extreme score since the logistics company cannot do any better in terms of driver consistency. It is also the case that it does not matter if the driver who served this customer was driver 1, 2 or 3 as long as it is only a single driver across the 7 time periods.

When you consider solution 2, the score this case should receive is less than the first solution. This is because there is some variation of drivers serving the customer. Solution 3 is the case that should receive the lowest score out of the 3 cases. The difference between solution 2 and solution 3 is that the number of times driver 2 serves this customer is one greater than in solution 3. However,

driver 2 is still the second most frequent driver who visits this customer. Driver 3 is the least frequent driver visiting the customer.

Generally speaking, from a consistency point of view, it is not just the most frequent driver that brings forward value for the customer, but also the second most frequent driver and third most frequent driver can bring forward value in terms of consistency for the customer. A mathematical function that can represent this property should be chosen.

We propose to use entropy related measures borrowing from the data-mining literature. There are various sorts of entropy-based measures used in the clustering literature depending on the specific study in which it is applied. The similarity between clusterings can be modeled as mutual information, normalized mutual information and other indices that researchers have come up with for their domain of study. We have incorporated normalized mutual information and simple entropy measures into our clustering algorithms.

Entropy related measures take probability vectors as input and output a scalar value. In our model, from a given history of driver assignments for a single customer, we derive the percentage of times that each driver has served this customer. We represent this information in a vector. This vector has elements less than or equal to one and sums to one given any time period. Therefore, we can treat it as a probability vector or as a vector that holds the frequencies with which each driver has visited this customer. As a result, every customer has an associated level of entropy measuring consistency. This embodies the properties discussed before. Mathematically, they measure the level of uncertainty in a probability distribution (or frequency distribution). A uniform distribution would correpsond to the worst case since each outcome (driver) has an equal probability or frequency. As the probability distribution diverges from the uniform distribution, entropy level gets lower and lower. They are represented as a convex combination of logarithms. The base of the logarithms can be chosen so that it gives a number between 0 and 1. Given a history vector of assignments represented as visiting frequencies, $p = (p_1, p_2, ..., p_d)$, the entropy of such a vector is

$$E = -\sum p_i log\,(p_i)\,.$$

For instance, if we consider the three solutions again, we can derive the consistency (entropy) level of each as follows:

$$\text{Solution 1: } (1,1,1,1,1,1,1) \longrightarrow (100\%, 0\%, 0\%) \longrightarrow entropy = 0$$

$$\text{Solution 2: } (1,2,1,2,1,2,1) \longrightarrow (57\%, 43\%, 0\%) \longrightarrow entropy = 0.9852$$

$$\text{Solution 3: } (1,1,2,1,2,3,1) \longrightarrow (57\%, 28\%, 15\%) \longrightarrow entropy = 1.3788$$

Looking at solution 2 and solution 3, we see the effect of the second most frequent driver, 43% and 28% respectively. Solution 2 has a lower entropy because given the most frequent driver, driver 1 with 57%, this solution has more skewed frequencies over drivers 2 and 3 compared to solution 3.

This entropy function has the nice property of being concave. It obtains the value zero when the probability vector puts all the probability mass on a single entry like solution 1 above. Moreover, the value of the function decreases as more probability mass is concentrated over fewer outcomes in the probability vector. This captures consistency in terms of the most frequent second driver, third driver and so on.

4.2.2 Model

We present a dynamic clustering model to give context to our solution approaches for creating consistency. Creating consistency is an assignment type of problem. It depends on who was served by which driver. Therefore, we can look at it as a type of clustering (assignment) problem. We are mainly interested in clustering techniques that can create consistency across time.

Consider a spatially distributed set of locations (customers) who are served by a single depot in a given geographic area. Customers generate service requests that are served by vehicles based at the depot. The subset of customers who generate demand for a given day is stochastic. At the beginning of every period demand and associated customers have become known before d trucks leave to serve delivery requests. This situation applies to, for example local distribution networks like UPS and FedEx. The dispatcher at the depot has information on the customers needing to be visited for that day and the associated demand. The dispatcher does not know about the future requests that will be made. His/her job is to assign customers with demand requests to drivers. We assume that each location generates only one unit of demand per day. Each individual demand will be modeled as a Bernoulli random variable with parameter p_i, where demand is independently distributed.

The horizon begins at $t = 0$. The final period is denoted by T. Every period a set of customers Q_t generate demand. Observing this, the dispatcher makes driver assignments. We keep track of the number of times a customer i has been assigned to driver j throughout all time periods, $0, ..., T$. We call such a variable a history or a history of assignments, H_t. Let $\mathscr{M}$ represent the map where the geographic region is situated. We have location parameters for all customers as well as the depot's location. Let $\mathscr{N}$ be the set of customers in this map. Let n be the number of customers in this map. Each coordinate of a customer is represented by L, which is a matrix of size $(n \times 2)$ where each row contains the x coordinate and y coordinate of a customer. For each customer i, the Bernoulli parameter is p_i. Further, p_i and p_j are independent for any i and j. Let m be the number of drivers available every day at the depot. H_0 is the matrix of size $(n \times d)$ representing initial history of driver assignments made to each customer where entry $H_0(i, j)$ is the number of times customer i has been assigned to driver j up until period 0. The information flows as follows: We step into period t and we know what H_t is. After, we observe Q_t, i.e. which customers need service this period, we make a decision based on these two variables, H_t and Q_t.

State Variables

State space consists of the information a decision maker needs to know to be able to make a correct decision. In our problem, with the long-run consistency as the objective, this includes keeping track of the history of assignments. Let H_t be the matrix of size $(n \times d)$ representing the history of assignments up to period t. Q_t is a binary vector $(n \times 1)$ representing the set of customers requesting service in period t. S_t is the state variable in period t (after Q_t is realized), which is a combination of H_t and Q_t, i.e. $S_t = [H_t\, Q_t]$.

To give an example, let us consider a situation when we are in period $t = 5$, and the number of customers is $n = 5$, and drivers $m = 3$. Also, the set of customers that request service this period has become known, hence we know Q_5. Then, a state $S_5 = [H_5\, Q_5]$ looks like the following:

$$H_5 = \begin{bmatrix} 2 & 2 & 1 \\ 3 & 0 & 0 \\ 0 & 4 & 1 \\ 2 & 0 & 2 \\ 3 & 1 & 0 \end{bmatrix}, \; Q_5 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}.$$

These two variables are interpreted as follows. In matrix H_5, Customer 1 (i.e. row 1) has been assigned to Driver 1 (column 1) two times, to Driver 2 (column 2) two times and once to Driver 3 (column 3). In vector Q_5, Customer 1 (row 1) has made a request for period 5, Customer 2 (row 2) has not made a request.

Decision Variables

Given a state, S_t, the decision is to cluster the customers whose $q_t^i = 1$, where q_t^i is ith component of Q_t. Further, define $\overline{Q_t} \subseteq Q_t$ such that, if $q_t^i = 1$, then $i \in \overline{Q}_t$. Let $A_t(S_t) = \{x_{ij}\}$ denote the set of decision variables at S_t where $x_{ij} = 1$ if customer i is assigned to driver j. Hence, the latter is an assignment notation only. Now,

$$\sum_{j=1}^{d} x_{ij} = 1 \quad \forall\, i \in \overline{Q}_t, \tag{4.2.1}$$

$$\sum_{i=1}^{n} x_{ij} \leq C \quad \forall\, j \tag{4.2.2}$$

$$A_t(S_t) = \left\{ \{x_{ij}\} \in \{0,1\}^{N \times m} \;\middle|\; \text{subject to (4.2.1) and (4.2.2).} \right\}. \tag{4.2.3}$$

The admissible set of actions are all clusters such that every customer that has a request is assigned to one driver only and each driver has at least one customer to serve.

Transition function

The transition $P(S_{t+1}|S_t, A_t(S_t))$ from $S_t = (H_t, Q_t)$ to $S_{t+1} = (H_{t+1}, Q_{t+1})$ works as follows:

- Q_{t+1} is an independent random variable of S_t since we assume independence of realizations for demand across all time periods. Hence, it is governed by $Ber(i) = p_i$ for all $i \in N$. The probability of Q_{t+1} is given by

$$P(Q_{t+1} = (y_i)_{i=1}^n) = \prod_{i=1}^{n} (1-p_i)^{1-y_i} p_i^{y_i}$$

  where $y_i \in \{0,1\}\ \forall i$.

- H_{t+1} depends deterministically on H_t, Q_t and $A_t(S_t)$. The transition is as follows: Given $H_t(i,:) = \left(h_{i1}^t, h_{i2}^t, ..., h_{id}^t\right)$, which is the ith row of H_t. If $Q_t(i,:) = 0$, then $H_{t+1}(i,:) = \left(h_{i1}^t, h_{i2}^t, ..., h_{id}^t\right)$. If $Q_t(i,:) = 1$, then

$$H_{t+1}(i,:) = \left(h_{i1}^t + x_{i1}, h_{i2}^t + x_{i2}, ..., h_{ij}^t + x_{ij}, ..., h_{id}^t + x_{id}\right).$$

Contribution Function

The per-period contribution function is composed of two parts. The first component accounts for the compactness of clusters formed in period t. The second component accounts for the value of the consistency achieved in period t. We do not explicitly define how we measure consistency at this point in the per-period contribution function, but we defer the reader to the next section where we explicitly define how we account for consistency.We express the contribution function as

$$C_t\left(S_t, \{x_{ij}\}\right) = \sum_{i=1}^{n}\sum_{j=1}^{d} c_{ij}x_{ij} + \beta \cdot consistency\left(\{x_{ij}\}\right)$$

where $\{x_{ij}\} \in A_t(S_t)$ and c_{ij} is the contribution to the clustering objective of assigning customer i to cluster j. Note that there is a scalar β in front of the consistency term, which models the degree of trade-off between the two objectives in the contribution function.

Bellman Equation

The full horizon objective we wish to maximize in period 0 is

$$\max_{\pi \in \prod} \left\{ \mathbb{E}\left(\sum_{t=0}^{T} C_t\left(S_t, A_t^{\pi}(S_t)\right) \middle| s_0 \right) \right\}$$

where $A_t^{\pi}(S_t)$ is a particular decision/policy rule that is admissible, i.e. a collection of particular

decisions at every state that can be visited in $t = 1,2,,,T$ starting from s_0. $\prod$ is the set of all of these policy rules.

We are ready to write the Bellman equation of this formulation:

$$V_t(S_t) = \max_{a \in A_t(S_t)} \{C_t(S_t, A_t) + \mathbb{E}(V_{t+1}(S_{t+1}) | S_t)\},$$

$$P(S_{t+1} | S_t, A_t(S_t)).$$

4.3 Solution Methods

Creating consistency in a multi-period setting depends *only* on how customers are assigned. We solve for assignment policies using a clustering approach. We present two clustering approaches that create consistency dynamically. First, we use normalized mutual information as a consistency measure that clusters customers based on consistency at the cluster level. Second, we measure consistency at the customer level, and do clustering using customer level consistency information. We will employ a clustering scheme whose objective balances the trade-off between compact clusters and consistency.

4.3.1 Normalized Mutual Information Approach

4.3.1.1 NMI

First, we propose a measure of consistency that is based on the notion of normalized mutual information. This index is generally used as a similarity measure to compare two clusterings within the data-mining literature. We leverage the idea of the similarity between clusterings and apply this notion to multi-period vehicle routing as measuring consistency over time. See Wagner and Wagner (2007) for a discussion of different methods the data mining literature uses to compare clusterings. This can be seen as an aggregate measure of consistent service where as in the next subsection we focus on a more refined measure of consistency based on customer level consistency. We can interpret the value of the normalized mutual information measure between clusterings as the degree of consistency between them. The higher this value, the higher the consistency. Unfortunately, NMI is a nonlinear combinatoric function of the decision variable. Instead, we make the optimization indirectly, and we include a measure of distance in the clustering objective to encourage consistency that in turn leads to high NMI. The details of the procedure will be explained below.

Assuming current period t, we summarize the service customers receive in a matrix. Assume we know the set of customers needing service for period t and they are indexed by the set Q. Let m denote the number of these customers. Let H_t^Q be the matrix with history of assignments made to each customer that needs to serviced right now, i.e. superscript Q and subscript t. Let X_t^Q be the $0-1$ matrix where if $X(i,j) = 1$, then customer $i \in Q$ is assigned to driver j. Then, a decision to allocate customers to drivers leads to a new history at $t+1$, given by $H_t^Q + X_t^Q$. In order to compute

the NMI value of such a decision we manipulate this matrix and get a new matrix for period t which we call $Conf_t$.

We compute several numbers using the entries in this $Conf_t$ matrix. Let $W = |Conf_t|$ be the summation of all the entries. Let $P(i)$ be the sum of row i divided by W, and $P(j)$ be the sum of column j divided by W. We compute the entropy associated with the rows in this matrix by

$$HR = -\sum P(i)log(P(i)). \tag{4.3.1}$$

We compute the entropy associated with columns in this matrix by

$$HC = -\sum P(j)log(P(j)). \tag{4.3.2}$$

Again, the base for the logarithms can be chosen so that these give values between 0 and 1.

The $Conf$ matrix is a $(d \times d)$ matrix whose \$i-j\$th entry is given by

$$Conf_{ij}^t = \left[H_{\bullet i}^Q\right]^\intercal X_{\bullet j}. \tag{4.3.3}$$

The entry $Conf_t(i,j)$ is equal to the total number of times all the customers that have been assigned to driver i until period t that are assigned to driver j. Each row corresponds to the summation of the rows of H_t that have been assigned to driver $r \in \{1,2,..,m\}$. The mutual information between the rows and columns of $Conf_t$ is given by

$$I = \sum_{i=1}^{m}\sum_{j=1}^{m} P(i,j)\, log_2\left(\frac{P(i,j)}{P(i)P(j)}\right) \tag{4.3.4}$$

where $P(i,j) = Conf_t(i,j)/W$. Finally, the normalized mutual information is given by

$$NMI = \frac{I}{\sqrt{HR \times HC}}. \tag{4.3.5}$$

We illustrate how to obtain the $Conf$ matrix in the following example:

$$H_5 = \begin{bmatrix} 2 & 2 & 1 \\ 3 & 0 & 0 \\ 0 & 4 & 1 \\ 2 & 0 & 2 \\ 3 & 1 & 0 \end{bmatrix}, \; Q_5 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

Assume we are given the state $S_5 = (H_5, Q_5)$ above. Let the decision variable take a value such that

the assignments are $\{x_{11}, x_{23}, x_{32}, x_{41}, x_{52}\} = (1,1,1,1,1)$, so we get $[x_{ij}] = X$ as follows:

$$X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

$Conf_5(2,2)$ will be equal to

$$Conf^t_{2,2} = \left[H^Q_{\bullet 2}\right]^\intercal X_{\bullet 2} = \begin{bmatrix} 2 & 0 & 4 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}.$$

$$Conf_5 = \begin{bmatrix} 2+2 & 2+0 & 1+2 \\ 0+3 & 4+1 & 1+0 \\ 3 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 4 & 2 & 3 \\ 3 & 5 & 1 \\ 3 & 0 & 0 \end{bmatrix}.$$

In this example, $W = 21$. $HR = 0.91$, $HC = 0.94$, and $I = 0.16$. Therefore, the normalized mutual information turns out to be 0.17. This value is close to zero telling us that there is not much mutual information between the rows and columns in the $Conf$ matrix, which means that our cumulative decisions have lead to relatively low consistency in the driver-customer assignments.

In general, NMI is a number in the range $[0,1]$. It is a measure of the agreement between two clusterings. By appropriately summarizing history, we compare today's assignments with our summary of the history for the service region, i.e. the $Conf$ matrix. If this agreement is in line with history, then there is more consistency in our solutions across time.

4.3.1.2 Clustering

We do not directly optimize over the NMI measure, which is a complicated expression as previously shown. Our strategy is to cluster customers in such a way that customers with similar history profile end up in the same cluster provided that they are not very far apart. Then, we assign drivers to each cluster so that consistency is maximized.

We use well-known distance functions used in data mining to measure dissimilarity between data objects to derive a heuristic. We use cosine distance to measure the distance between the histroy profile of two different customers. The histroy profiles are represented as frequency vectors and we are interested if whether or not two history profiles are aligned or not. Cosine distance measures the degree of this alignment. We briefly define the cosine distance here, which is equal to 1 minus the cosine similarity. Given two vectors $x, y \in \mathbb{R}^n$, the cosine distance between these

vectors is defined to be

$$1 - cos\,(x,y) = 1 - \frac{\langle x,y\rangle}{\| x \| \| y \|},$$

where the numerator is the dot product between x and y, and the denominator is the product of the Euclidean norm of each vector. Cosine similarity measures the angle between two vectors in a real space. In general, the cosine similarity is a number between -1 and 1.

Consider the following example to see why it is useful for our model: Let $h_1 = (3,4,0,0)$ and $h_2 = (1,1,4,5)$. If we compute the cosine distance between these vectors we get a cosine distance of 0.78. When we make $h_2 = (0,0,4,5)$, the cosine distance obtains its maximum value of 1. However, since we do not have vectors of history with negative values we will never compute a cosine similarity of -1. Cosine distance will always be between 0 and 1 for all the possible realizations of history in our model.

Given a state in our model, our strategy is to first cluster customers and then assign drivers. Each step involves an optimization. The clustering step is described below.

The presentation of the objective requires the notation $d_{euc}(.)$ for Euclidean distance, $d_{cos}(.)$ for cosine distance. Say we are given a state $S_t = (H_t, Q_t)$. Let H_t^Q denote the rows of H_t where $Q(i) = 1$. These are the histories of those customers that have made a request in period t. Given this H^Q matrix and L, we can a write clustering problem with the following minimization objective:

$$\sum_{i=1}^{m} \sum_{x \in C_i} d\,(x,\, \mu\,(C_i)) = \sum_{i=1}^{m} \sum_{x \in C_i} d_{euc}\,(x_{loc},\, \mu_{loc}\,(C_i)) + \beta\, d_{cos}\,(x_{hist},\, \mu_{hist}\,(C_i))\,,$$

where x_{loc} represents a customers location obtained from matrix L, and $\mu_{loc}(C_i)$ is the location component of the seed point for cluster i which is also called the centroid of cluster C_i. Likewise, x_{hist} comes from H_Q and μ_{hist} is the history component of the centroid of C_i. The history component of a centroid is the mean of the history profiles of all customers in a cluster.

$d_{euc}\,(x_{loc},\, \mu_{loc}\,(C_i))$ is the standard Euclidean distance squared between location of customer x and the location of the centroid of C_i.

$d_{cos}\,(x_{hist},\, \mu_{hist}\,(C_i))$ is the cosine distance between history of customer x and the history component of the centroid of C_i.

β is a balancing parameter which we use to bias the clustering solution to be more history oriented, hence more consistency or to be more location oriented, hence lower traveling cost, at least approximately.

The constraints of this problem are standard; every customer with a request must be assigned to one and only one cluster.

The motivation of the above formulation comes from the idea that if we can leverage the historical data of assignments and include this into our objective as a second component of distance between customers, then customers that have been assigned to the same driver more or less in the past should be made to group in the same cluster. This is done using a second distance function that

measures the distance between two customers in terms of their history of assignments. If we are not able to put similar customers into same clusters, then making a driver assignment would entail a loss of consistency in service since at least one group would have to be served by a different driver than the one mainly observed in their history, which is an unwanted situation.

4.3.1.3 Assignment of Drivers

Now, once we compute a clustering solution, we are ready to assign drivers to clusters in light of the NMI consistency measure described in the previous section.

In order to write the optimization problem, we need to compute a $Conf$ matrix similar to the one in the previous section. The only difference is that the columns represent clusters and *not* individual drivers. We denote $Conf$ as follows:

$$Conf = \begin{bmatrix} c_{11} & c_{12} & . & . & c_{1m} \\ c_{21} & c_{22} & . & . & c_{2m} \\ . & . & . & . & . \\ . & . & . & . & . \\ c_{m1} & c_{m2} & . & . & c_{mm} \end{bmatrix}.$$

The interpretation of c_{ij} is the total number of times customers in cluster j have been assigned to driver i up to period t. What we do is to find a matching function between drivers and clusters. This is done by solving the following integer programming formulation:

$$\max \quad \sum_{i=1}^{m}\sum_{j=1}^{m} c_{ij} z_{ij}$$

$$\text{subject to} \quad \sum_{i=1}^{m} z_{ij} = 1 \quad \forall j$$
$$\sum_{j=1}^{m} z_{ij} = 1 \quad \forall i$$

We call the greedy solution to this problem as *greedy* and the optimal one *optimal*. Also, we note that the constraint matrix of this problem is totally unimodular.

This problem formulation is motivated by the idea that, if we are to match drivers with clusters, we should obtain a high trace from the resulting matrix when we order the columns from driver 1 to driver m. A high trace value would be associated with a high NMI value in the future since in this way we create more unevenness in the future possible realizations of this matrix. To illustrate this point, consider the following 3 (3×3) matrices.

$$Matrix1 = \begin{bmatrix} 32 & 2 & 24 \\ 34 & 29 & 0 \\ 4 & 27 & 30 \end{bmatrix} \quad Matrix2 = \begin{bmatrix} 15 & 0 & 0 \\ 0 & 13 & 0 \\ 0 & 0 & 14 \end{bmatrix}$$

$$Matrix3 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The first matrix has an NMI value of 0.27. In contrast to the large numbers in this matrix, the dispersion of the numbers is somewhat more even compared to the other two extreme examples. This matrix tells us that we have not done a good job of assigning driver 1 customers since when we try to match driver one with one of the columns, we are stuck between column 1 and column 2 and we would have to give up on some consistency. So in order to determine the best possible matching we solve the formulated problem above that would direct us to higher consistency. The second example has NMI of 1 which is the highest possible value that can be obtained. This matrix has done perfectly in assigning all the customers served by driver 1 to only one cluster, similarly for the other clusters. This makes our job easier in terms obtaining higher consistency. Moreover, matrix 3 has done a good job for Driver 1 and Driver 3's customers in terms of history but not in terms of Driver 2's customers. The NMI value obtained in this case is 0.67.

4.3.2 Individual Entropy Approach

In the previous section, we used a consistency measure that is computed from an aggregation of individual assignments. This is done because we want to evaluate today's clustering in terms of consistency. However, fundamentally we are interested in consistency at a finer level, from an individual perspective. Also, the question is how to combine all these individual consistency measures (NMI is one particular option). We propose to use entropy at an individual level (computed from an assignment vector for each customer) to measure the degree of certainty in that vector, and call this measurement the consistency for this customer. In order to get the consistency level for a group of customers, we can simply add their individual consistency measures. It can be seen as a finer way of measuring the same thing. However, the aggregate consistency measure will not belong to the $[0,1]$ interval anymore, but will be a sum of such numbers. NMI is advantegous because it shrinks the size of the problem for the consistency decision. Additionally, NMI is nonlinear in our decision variables (see below) where as the newer measure is simpler. NMI is highly nonlinear and nonconvex.

Say we have a vector of assignments in period t for customer i, $h(i) = (h_1, h_2, ..., h_n)$. So we have n drivers at our disposal. Then, the entropy of this vector can be used to represent the consistent service this customer has received up to now:

$$H_t(h(i)) = Entropy_t(i) = -\sum p(k) log_2(p(k))$$

where $p(k) = h_k / \{\sum_{r=1}^{n} h_r\}$. So this $p(k)$ is the percentage of time that customer i has been served by driver k.

This measure has some useful properties:

1. $H_t(i)(p(1), p(2), ..., p(n)) = H_t(i)(p(n), p(1), ..., p(2))$. It is symmetric with respect to

permutation of probabilities, hence, with respect to permutation of elements of vector h_i.

2. It is continuous in p.

3. $H^n(p(1), p(2), ..., p(n)) \leq H^n\left(\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n}\right)$. The measure is maximal if all outcomes have equal percentages.

4.3.2.1 Examples

Say we have $n = 4$ drivers.

1. $h_1 = (x, x, x, x)$. Entropy of this vector is 2, no matter what x variable takes on as long as $x \neq 0$.

2. $h_2 = (y, y, y, 0)$. Entropy of this vector is 1.585, no matter what y values take, as long as $y \neq 0$.

In terms of the idea of consistency, h_2 is always going to have a better consistency measure than h_1, independent of $x, y,$ and, $n \geq 2$. This makes sense because h_2 has made equal assignments to only 3 drivers where as h_1 has made the same to all the drivers available. Also, since entropy obtains its maximum value when all percentages are equal (bullet 3 in previous section), then it can be normalized with this maximum value so that the measure takes on a value between 0 and 1.

4.3.2.2 Individual Measure

Within the context of making decisions over time, creating consistency with this new measure is possible. Let H and X be as before. So, a decision gives the new history of $H + X$ at time $t + 1$. So from an individual customer i perspective the new level of entropy in its assignments is

$$Entropy_{t+1}(i) = -\sum p(k) log_2(p(k))$$

where

$$p(k) = \frac{H_{ik} + X_{ik}}{\sum_k (H_{ik} + X_{ik})}$$

So, given a period t, say customer i has a history vector of assignments of $h_t(i) = (h_1, h_2, ..., h_n)$. A history of this individual can change only to n-different values depending on who serves this customer this period, namely $h_{t+1}(i) = (h_1 + 1, h_2, ..., h_n)$ or $h_{t+1}(i) = (h_1, h_2 + 1, ..., h_n)$ or $\ldots$, or $h_{t+1}(i) = (h_1, h_2, ..., h_n + 1)$. So we can compute each of these vector's entropy level, and denote it by c_{ik}, meaning that for customer i, if $X_{ik} = 1$, c_{ik} will be obtained for entropy, hence consistency. Finally, the 'Individual' objective makes the problem linear. It also is a good measure, maybe better measure than NMI because it is at a finer level. We can tie the labeling and clustering into one optimization (as opposed to cluster first and then assign labels, section 4.3.1.1) in a simpler way.

4.3.2.3 Clustering

Assume we are in period t. Given a set of m customer locations in R^2 represented by $A \in R^{m \times 2}$ that require delivery and a given number of clusters, d (number of drivers), the clustering problem is to

solve the following optimization problem:

$$\underset{C_l, D_{il}, T_{il}}{\text{minimize}} \quad \sum_{i=1}^{m} \sum_{l=1}^{d} [1-\alpha \; 1-\alpha \; \alpha]^T D_{il} T_{il} \tag{4.3.6}$$

$$\text{subject to} \quad -D_{il}^j \leq \left(\left(A_i^t\right)^T - C_l \right)^j \leq D_{il}^j, \quad i = 1,..,m \,\&\, l = 1,...,d, \; j = 1,2 \tag{4.3.7}$$

$$\lambda_{il}^t \leq D_{il}^3, \quad i = 1,..,m \,\&\, l = 1,...,d \tag{4.3.8}$$

$$\sum_{l=1}^{d} T_{il} = 1, \;\; T_{il} \geq 0, \; i = 1,..,m \,\&\, l = 1,...,d \tag{4.3.9}$$

Here, D_{il} is a dummy variable in R^{2+1} where the first two components represent the bounds on the 1-norm distances and last component is the bound on the entropy term (our consistency measure). e is a vector $[1-\alpha \; 1-\alpha \; \alpha]^T$ of size 3×1. α is the weight given to the objective of the entropy (consistency) term. D_{il}^j is the jth component of vector D_{il}. So, the term inside the minimum in the summation is the summation of the bounds on the 1-norm distances and the bound on the entropy of assigning point i to cluster/driver l. λ_{il}^t is the entropy obtained by assigning point i to driver l which is precomputed and is given for the clustering problem above. These sets of constraints guide the objective towards being highly consistent and depend on the history of assignments made until period t. The choice variables are C_l and D_{il} for each $i \in \{1,..,m\}$ and $l \in \{1,...,k\}$.

The way in which historical data is incorporated is through the set of constraints 4.3.8. They depend on the matrix H from the previous section. From this matrix, we compute the bounds on the constraints 4.3.8 as follows. Given the ith row of matrix H, $H_i^t = \left(h_{i1}^t, h_{i2}^t, ..., h_{id}^t\right)$,

$$\lambda_{il}^t = IE\left(h_{i1}^t, .., h_{il}^t + 1, ..., h_{id}^t\right)$$

where IE is the information entropy function

$$IE = -\sum_{l=1}^{d} p_l log\left(p_l\right)$$

where p_l is the lth entry of the normalized H_i^t vector. Clustering is applied for all periods up to period T, representing the length of the operation horizon.

We apply the iterative k-medians like solution method where we make customer assignments to cluster centers and adjust cluster centers until cluster centers do not change. The difference to a standard k-medians method is in the objective minimized that contains a consistency component weighed by α.

4.4 Simulation Results

4.4.1 Normalized Mutual Information

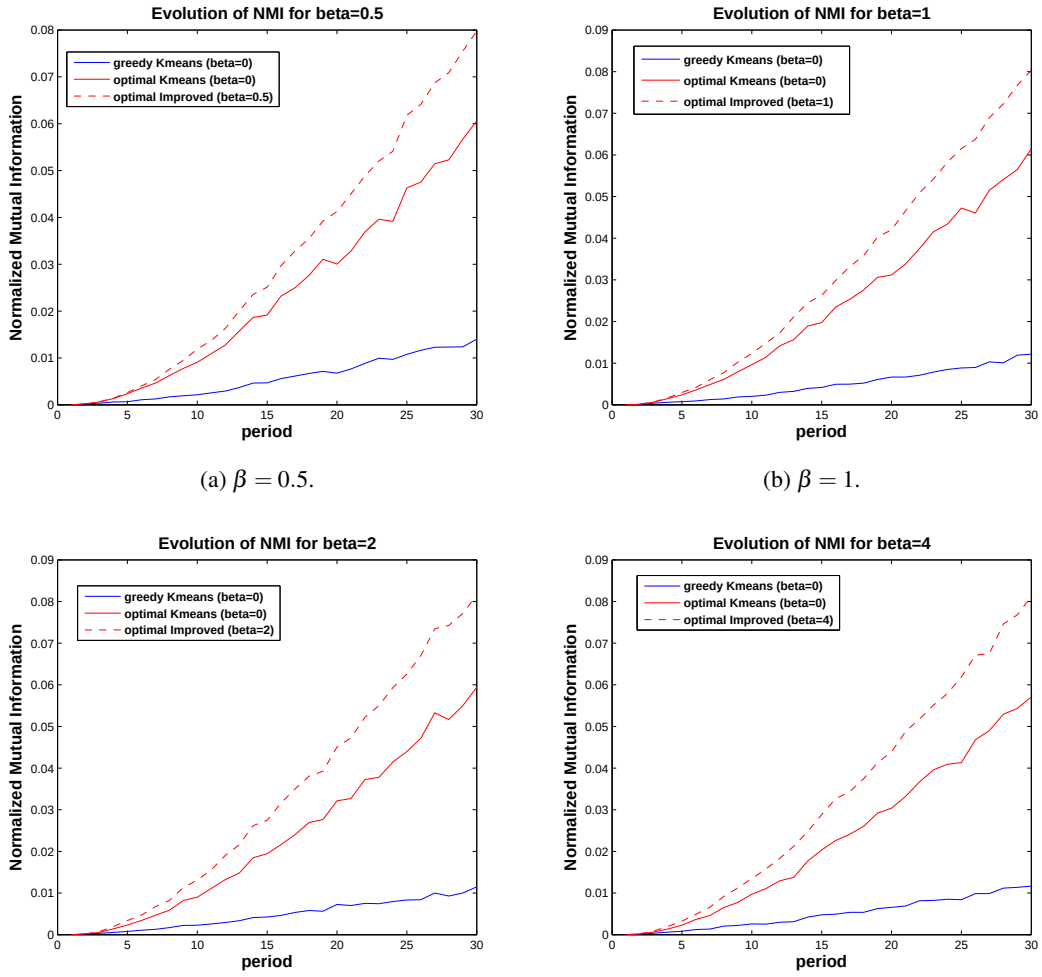
We provide simulation results and show the effectiveness of our approximation in creating consistent behavior in driver-to-customer assignments. We have not fully solved the problem in Section 3, but gained insight in what kind of policies would generate consistent behaviour.

The parameter configurations involved are the size of the map, the number of customers, the Bernoulli parameter vector, *Ber* and the initial history, H_0. One important observation is that the range of the Euclidean distance and the range of the distance for the history component should be similar, hence a square grid on $[0,1]$ is considered. The cosine distance also has a range of $[0,1]$ but the Euclidean distance does not necessarily. In order to balance this bias, we consider such dimensions for the hypothetical geographic region.

4.4.1.1 Experiment 1

In this simulation, we study if the way we model the objective affects our ability to obtain higher consistency by changing the parameter, β. Intuitively, the greater β, the greater consistency should be observed. The beta parameter is used to bias the objective towards history, like in experiment 1, however we do not require it to be a convex combination of location component and the history component. We call the objective with $\beta = 0$, a *Kmeans* objective. It is solely aimed at traveling cost efficiency. When $\beta > 0$, we call this objective an *improved* objective since it balances traveling cost efficiency with consistent service.

The simulation experiment is carried out with 80 potential customers at various locations in the square $[0,1] \times [0,1]$. They are chosen to be situated with a uniform probability on this grid. Their demand probability vector *Bern* consists of success parameters chosen uniformly between 0 and 1 for each customer. Each period $m = 5$ drivers serve the region. There are 30 periods. The parameter β takes on 4 discrete values of $\{0.5, 1, 2, 4\}$. These results are dependent on the initilization of H_0, which here is a uniform matrix with 10 assignments to each driver for each customer.



(a) $\beta = 0.5$.

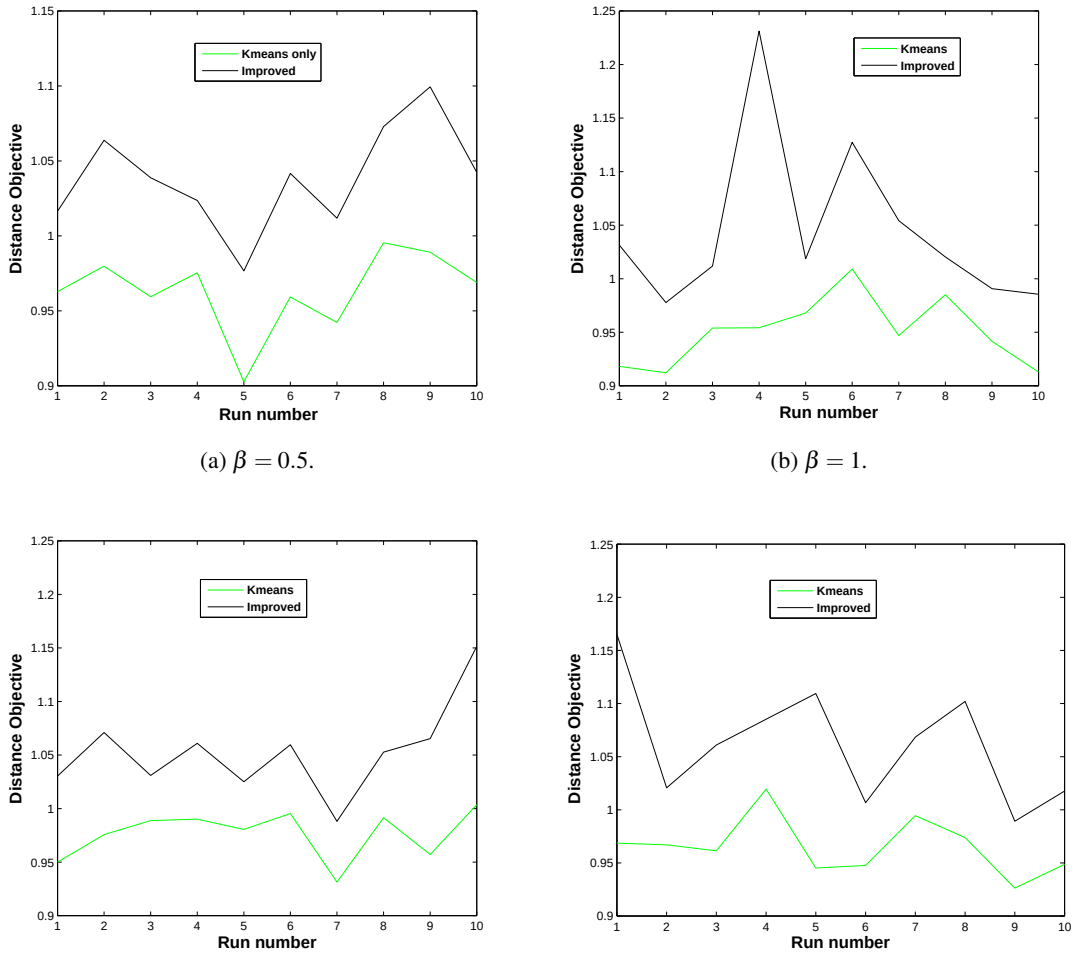
(b) $\beta = 1$.

(c) $\beta = 2$.

(d) $\beta = 4$.

Figure 4.4.1: NMI series with β varied.

In Figure 4.4.1, which shows consistency results versus a range of parameter values for β, the first thing to note is that the *Improved* scheme always does better than the *Kmeans* scheme across all parameter values. It is important to note that the scale of the Euclidean distance and cosine distance should be the same for this to occur. Otherwise, the biasing parameter β has to be rescaled accordingly in order to see cases where the improved scheme does better in terms of consistent assignment of customers to drivers. There is no clear improvement in the difference between the *improved* scheme and the *Kmeans* scheme with respect to the β values, on the other hand.



(a) $\beta = 0.5$.

(b) $\beta = 1$.

(c) $\beta = 2$.

(d) $\beta = 4$.

Figure 4.4.2: Routing efficiency *k-means* vs. *Improved*

In Figure 4.4.2, the lines represent average traveling cost observed in each of the runs. The difference seems to be constant around $7-8$ %, for all the β values. As we had expected according to our intuition, there is a trade-off between gaining consistency and lowering routing cost. We can interpret the following: the NMI series for the *Improved* scheme builds up quicker than the *Kmeans* scheme at a cost of about $7-8$ % routing efficiency. The NMI series will keep on building consistency value and increase almost monotonically with respect to the number of periods.

4.4.1.2 Individual Customer Assignments

Since we demonstrated that the improved scheme does better in terms of NMI on average, we can look into how consistent the assignments have been on a more granular level based on individual customers. The following table compares an optimal-Kmeans with an optimal-Improved scheme. There exist 80 customers who have received service for 30 days by 5 drivers. Figure 4.4.3 shows first 22 customers from the output. We can see that the assignments are very close to each other in

general but there are major differences for some. For instance, customer 16 has been assigned to Driver 1 35 times in the *Improved* scheme where as the same customer has been assigned between Driver 1 and Driver 3, 24 and 21 times respectively. There are also other customers that have a significant difference in the assigment distribution.

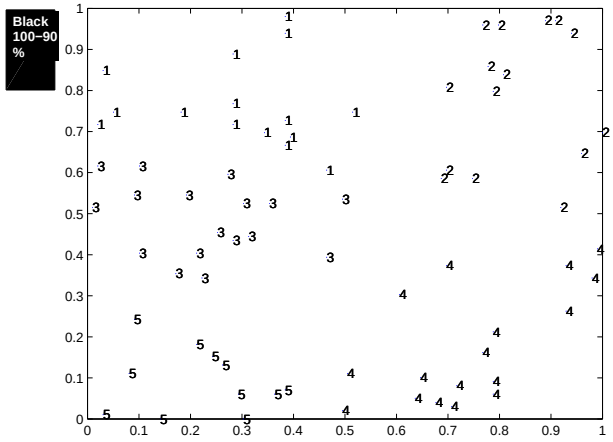
	Optimal *Kmeans*					Optimal *Improved*						
ID	Driver 1	Driver 2	Driver 3	Driver 4	Driver 5	Driver 1	Driver 2	Driver 3	Driver 4	Driver 5	Total	Total
1	10	10	10	23	10	10	10	10	23	10	63	63
2	11	10	10	18	10	10	10	10	19	10	59	59
3	10	10	17	10	10	10	10	17	10	10	57	57
4	11	10	10	14	10	10	10	10	15	10	55	55
5	10	39	10	10	10	10	39	10	10	10	79	79
6	10	12	10	10	10	10	12	10	10	10	52	52
7	10	10	10	35	10	10	10	10	35	10	75	75
8	11	10	11	10	10	12	10	10	10	10	52	52
9	10	13	10	10	10	10	13	10	10	10	53	53
10	10	10	34	10	10	10	10	34	10	10	74	74
11	10	10	10	10	10	10	10	10	10	10	50	50
12	10	39	10	10	10	10	39	10	10	10	79	79
13	10	10	10	10	13	10	10	10	10	13	53	53
14	36	10	14	10	10	40	10	10	10	10	80	80
15	10	10	10	10	37	10	10	10	10	37	77	77
16	24	10	21	10	10	35	10	10	10	10	75	75
17	10	10	19	10	10	10	10	19	10	10	59	59
18	30	10	15	10	10	35	10	10	10	10	75	75
19	14	10	20	10	10	10	10	24	10	10	64	64
20	10	10	10	10	12	10	10	10	10	12	52	52
21	10	27	10	10	11	10	28	10	10	10	68	68
22	17	10	11	10	10	18	10	10	10	10	58	58

Figure 4.4.3: Individual assignments at the end of 30 periods.

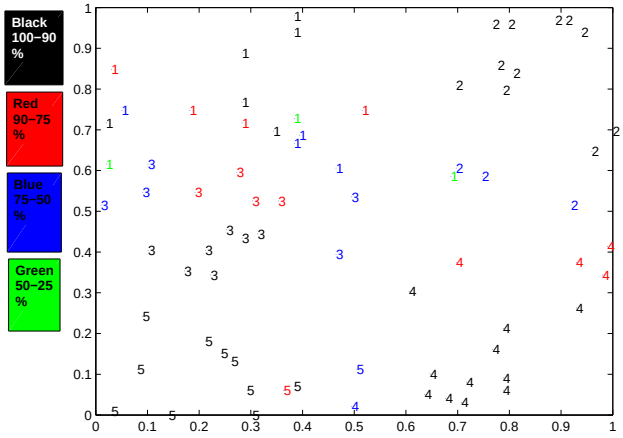
Overall, we see that the optimal-Improved scheme does as well as or better than the optimal-Kmeans scheme. We compare the maximum number in each row to come to this conclusion. As an example, Customer 18 has been assigned to a single driver 30 times and 35 times the most, under *Kmeans* and *Improved* respectively.

4.4.1.3 Service Zones

We are also interested in seeing how the service zones of each driver evolve over time. How stable are they? We give results that indicate *qualitative* answers to these questions. In the figures, the points represent customer locations and the attached numbers are the drivers who have served that customer the most times. A point is black if this driver has served the customer 100%-90% of the time, red if 90%-75% of the time, blue for 75%-50%, green for 50%-25%.



(a) *Improved*.



(b) *Kmeans*.

Figure 4.4.4: Service areas by service rate, H_0 non-random.

In Figure 4.4.4, the service regions are based on H_0 being a constant matrix whose elements are all equal to 10. So, customers start with a uniform prior in terms of assignment distributions in period 0. In Figure 4.4.4a, we see that the *Improved* scheme has done perfectly at 90% level of assigning customers consistently to a single driver. In Figure 4.4.4b, we see that *Kmeans* did not do as well in terms of creating consistency. We can see that Driver 5, 4 and 2 have served customers in a more consistent way, since their region encompasses more "black" customers than other kinds. Drivers 3 and 1 on the other hand have been less stable in serving customers. We see such a sharp difference in terms of service regions between each scheme mainly because of the initial prior and possibly because of the demand realization over the course of 30 periods.

Figure 4.4.5 is based on the optimal-Improved scheme where the initilization of H_0 is a random matrix with elements chosen between 0 and 10 as initial customer-driver history. There exist 80 customers, 30 periods and 5 drivers.

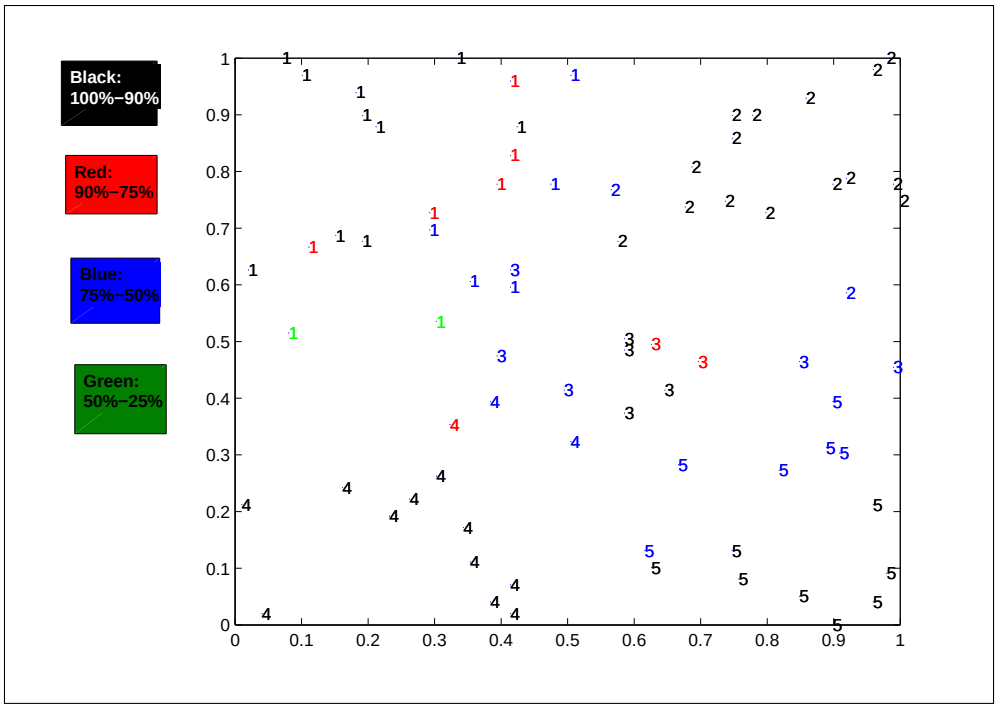


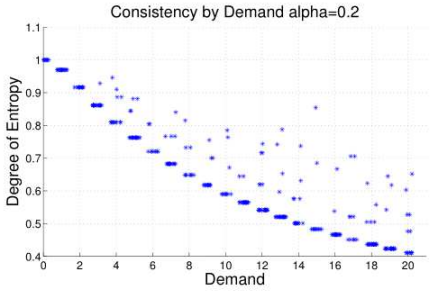
Figure 4.4.5: Service areas by service rate, H_0 random.

If you look into the figure closely, between the regions of "black" customers are the customers that are of other types. These customers seem to lie on the boundary between core groups of customers ("black") and have more variable assignment of drivers. Also, this figure suggests that the effectiveness of the *Improved* scheme depends on the initial history although this figure is based on a different map.

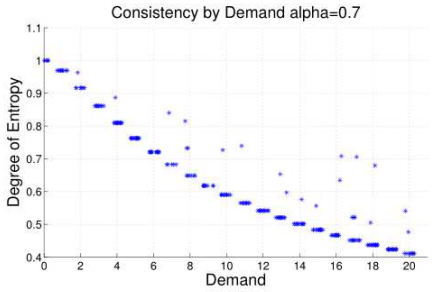
4.4.2 Individual Entropy Approach

We carried out simulations to see how responsive this approach was to modeling consistency in this way. For an operation horizon of 20 periods and a customer base of 80 locations, we generated random request for delivery every period according to a different bernoulli distibution for evey customer.

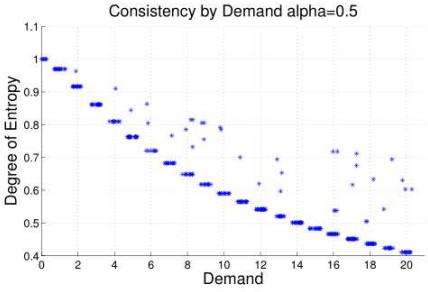
In Figure 4.4.6, we see how consistency levels at the end of 20 periods are distributed with respect to the demand each location generated. For those customers with higher demand, i.e. those that request delivery more frequently, the consistency level is greater or in other words degree of entropy is lower. Additionally, this effect is stronger for higher number of customers as the emphasis on the consistency term in the clustering objective is gradually increased from 0.2 to 0.9.



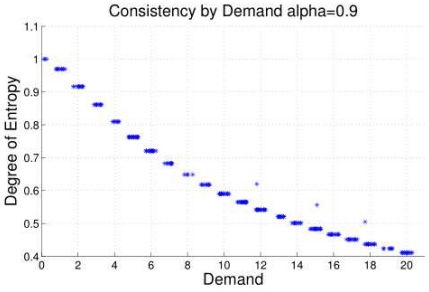
(a) $\alpha = 0.2$.



(b) $\alpha = 0.7$.



(c) $\alpha = 0.5$.



(d) $\alpha = 0.9$.

Figure 4.4.6: Consistency by demand.

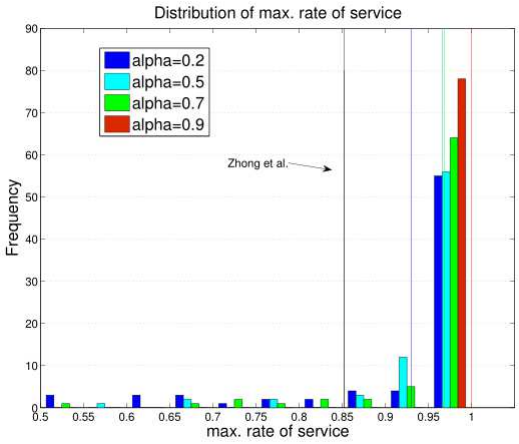


Figure 4.4.7: Maximum service rate by drivers.

The trade-off the α parameter controls in the clustering objective turned out to be on average 3.42 as can be seen in figure 4.4.8. For every one unit improvement in the consistency objective, the solution methodology forgoes 3.42 units in the clustering (compactness) objective.

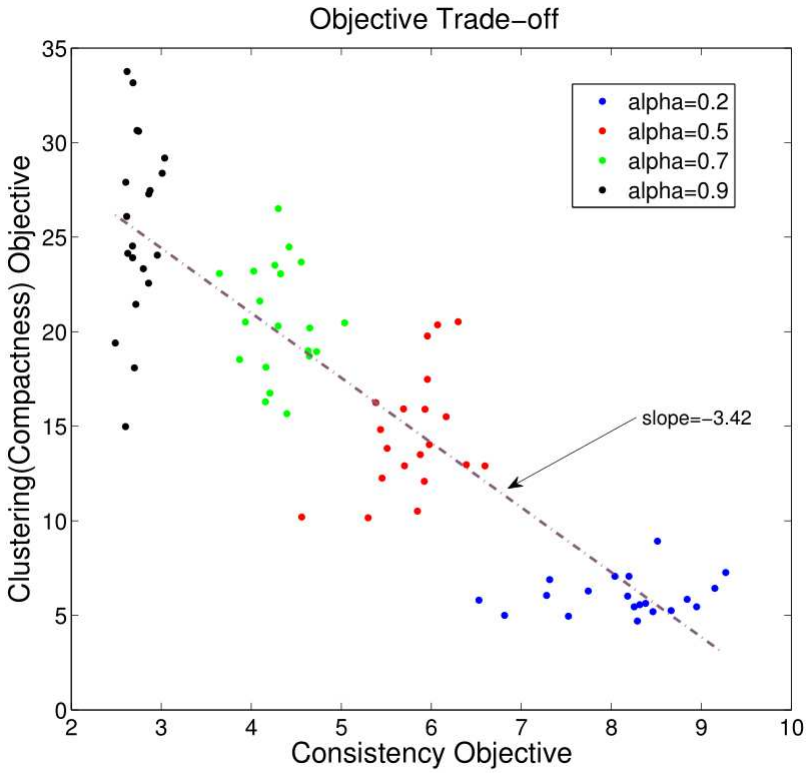


Figure 4.4.8: Pareto curve.

4.5 Conclusion

We have proposed and provided results of two policies that would create consistent behaviour in routing while balancing this with a surrogate cost efficiency. Unlike Zhong et al. (2007), we build on consistent service in an unrestricted way. Furthermore, our results suggest that the policy we have proposed is successful in creating consistency. Our future work would entail combining the methodology proposed here with a cluster-first, route-second VRP heuristic. We consider solving the actual daily routing problem with a fast local search heuristic and be able to compute more accurate route costs this way. Moreover, we would like to consider fast clustering algorithms that do a good job in obtaining good solutions within reasonable computation time.

Chapter 5

Stochastic Orienteering with An Application to Fish Trawling

5.1 Introduction

In this chapter, we address the problem of how to route a fishing vessel as it trawls for fish over a number of fishing areas. In this problem, we incorporate data by using forecasting models built using historical catch data. We integrate previous observations made during a fishing trip into the decision of where to harvest for fish.

A fishing trawler departs from an origin port, visits several fishing grounds, and lands its catch at a destination port. At each period, the fishing trawler needs to decide whether to keep fishing in the current fishing area or move onto a new one. We assume the fishing trawler harvests one or more species over several fishing grounds. The harvest is uncertain but we have historical data with which to estimate it. The per unit value of each species is known, p_l. The objective of the fishing trawler is to maximize the expected total value of catch over all species by traversing between the fishing grounds starting from an origin port to a destination port, while respecting time and capacity constraints.

A common solution approach to handle uncertainty is to use an a priori solution approach (fixed-route policy). This approach produces a routing solution that is easier to compute for large scale routing problems. See (Campbell and Thomas, 2008a) for a review. While an a priori solution approach has its benefits, a dynamic solution approach would constitute an improvement over fixed-route policies because it enables modification of the routing plan as observations (catches) are made. There has been recent work (Goodson et al., 2013b) that develop dynamic solutions based on fixed-route policies in a rollout framework. Rollout procedures are heuristic solution approaches that select the next best action at a current state using an estimate of the rewards-to-go for the set of states approachable from this current state. We employ their solution methodology and apply it to our stochastic orienteering problem.

Rollout algorithms rely on the valuation of the future at every step of the decision process.

Further, this valuation depends on how uncertainty is quantified. Using raw data, various statistical and data mining (data science) methods can be employed to aid the rollout algorithm. We explore several data analysis methods suitable for our data and validate the correct approach that creates higher value.

This work makes several contributions including taking into account correlation of rewards in the field of stochastic orienteering problems. We model the correlation into an MDP formulation by incorporating previous observations as a state variable. We propose short term forecasting models for South Baltic sea cod harvesting. We provide a prescription of a forecasting method combined with a routing strategy to use for cod harvesting specifically for the South Baltic sea. We shed light on the relationship between the quality of a prediction model and quality of routing obtained as a result.

In the following sections, we formulate the fishing vessel routing problem as a Markov Decision Problem (MDP). The MDP formulation can be found in Section 5.2. We use a rollout solution method coupled with variable neighborhood search to solve the MDP, which is explained in Section 5.3. We present predictive methods and the results of our solution approach applied to real cod-catch data for the Baltic region in Section 5.4.

5.2 Model Formulation

Let $\mathscr{G} = (\mathscr{N}, \mathscr{E})$ be a given complete graph. The nodes are represented by the set $\mathscr{N} = \{u\} \cup \{1,2,...,n\} \cup \{w\}$, where u represents the origin port and w represents the destination port. Let $i \in \{1,2,..,n\}$ represent a fishing ground. Fishing grounds are understood to be regions in the sea. However, we use the center point of a rectangular region to represent an area for harvesting. Let $\mathscr{E} = \{(i,j) \,|\, i, j \in \mathscr{N}\}$ represent the set of arcs between any pair of nodes. A fishing trawler starts a trip from node u, visits a subset of nodes in $\mathscr{N} \setminus \{u,w\} = \{1,2,...,n\}$, and arrives at the destination port w. There is an associated travel time $t\,(i,\, j)$ with every arc $(i,j) \in \mathscr{E}$. Let T denote the travel time duration limit that the fishing trawler is facing for a fishing expedition. This duration limit can be something set managerially in a company. We assume a vessel contains enough fuel and does not need to refill before this limit is reached. Let G be the finite available capacity in the fishing trawler in terms of the amount of fish it can hold after which it must travel to the destination port.

The catch between nodes for the same species or between species may be correlated. After arriving at a fishing ground, one unit of time is spent searching and towing (fishing) the current fishing ground. A catch is sampled for the first unit of time, and then a decision of whether or not to harvest the area is made. The process is repeated after each tow.

Time is assumed to be continuous. A decision epoch is triggered at the beginning of a new fishing cycle, which can correpsond to an arrival to a fishing ground and having sampled the area once or a continuation of fishing at the current fishing ground. The random time of decision epoch k, T_k, marks the end of period $k-1$ and the beginning of period k. While a trawler is in a fishing ground, time is incremented in equal discrete amounts (a fishing cycle) for each decision epoch

associated with the decision to whether or not keep harvesting the same area.

5.2.1 States

Each period has an associated pre-decision state s_k and a post-decision state s_k^a. There is a deterministic transition between s_k and s_k^a as a result of decision a. Time is updated depending on a either by upon arrival and completing the first tow/sample at the new area marking the beginning of period $k+1$ or at the end of the fishing cycle at the current fishing ground marking the beginning of period $k+1$. The post-decision state s_k^a corresponds to the state of the system after we have made a decision but before any new information (observations, which are catches in our case) has arrived. The transition between post-decision state s_k^a to pre-decision state s_{k+1} is random and associated with a random reward equal to minimum of remaining vessel capacity and current catch, $min\{R_k(s_k,a), g_k\}$. The process repeats until the pre-decision state is in the set of absorbing states, S_K. These states correspond to the vessel's current location being the destination port.

The state of the system captures all relevant information that is needed to determine the available actions and associated rewards (catches) as well as to compute the expected future value of being at that state. There are two possible ways to compute this value: either through the transition probabilities or by having access to a simulation process. The state needs to capture the current node of the fishing trawler. This is represented as variable $q \in \{\{u\} \cup \{1,2,...,n\} \cup \{w\}\}$. The remaining available capacity at decision epoch k in the fishing trawler needs to be known as well. Let $g \in [0,G]$ denote the scalar for the left-over capacity. The arrival time $t \in [0,T]$ to a node needs to be captured by the state variable as well. Moreover, the history of observed and unobserved catches need to be captured for every species. Let $c_i^l =?$, if catch rate at fishery i for species l is not observed yet. If the catch rate has been observed, possibly several times, then $c_i^l \in \{[0,\infty)\}^{h_i}$ where h_i is the number of observations made. The set of possibilities is represented by $\left\{\{?\} \cup \{[0,\infty)\}^h\right\}^{n\times l}$. At decision epoch k, a state s_k in state space $S = \{\{1,2,...,n\} \cup \{w\}\} \times [0,G] \times [0,T] \times \left\{\{?\} \cup \{[0,\infty)\}^h\right\}^{n\times l}$ takes the form $[q,g,t,c]$, where q is an integer, g is a scalar, t is a non-negative scalar, and $c = \left[c_i^l\right]_{i=1,l=1,o=1}^{n,m,h_i}$. The initial state is given by $s_0 = \left[u,G,0,(?)^{n\times l}\right]$. The set of absorbing states are those that correspond to when the trawler has arrived to the destination port, $S_K = \left\{ [q=w,g,t,c] \;\middle|\; g \le G, \quad t \le T, \quad c \in \left\{\{?\} \cup \{[0,\infty)\}^{h_i}\right\}^{n\times l}, \quad h_i \in \mathbb{Z}^+ \quad \forall i \right\}$.

5.2.2 Actions

An action, taken at each decision epoch, is to either stay or go to a new region, a. Let $I\{x = q_k\}$ be an indicator function such that

$$I\{x=q_k\} = \begin{cases} 1 & \text{, if } x = q_k \\ 0 & \text{, otherwise} \end{cases}.$$

The set of admissible actions allowed at state s_k are

$$\mathscr{A}(s_k) = \Bigg\{ a \in \{\mathscr{N} \setminus \{u\}\} : a \notin \Bigg\{ x \in \{\mathscr{N} \setminus \{u\}\} : t_k + I\{x = q_k\}\Delta t + (1 - I\{x = q_k\})(t(q_k, x) + \Delta t + t(x, w)) > T \Bigg\} \Bigg\}, \tag{5.2.1}$$

where condition (5.2.1) disallows staying or going to new areas if it is going to violate the time duration limit. In such a case, the vessel is forced to travel to destination port. Although not explicitly modeled in the action space, if capacity is met at the end of a fishing cycle, we assume the vessel is forced to travel to destination port even though the duration limit may not be reached. From that point on, the sample path corresponds to zero value, hence this assumption does not change anything. We incorporate this implicit constraint into our solution approach directly.

5.2.3 Transition to Post-Decision State

Given that pre-decision state is currently $s_k = [q_k, g_k, t_k, c_k]$ and action $a \in \mathscr{A}(s_k)$ is selected, the deterministic transition to post-decision state s_k^a entails updating the current node and time:

$$q_k^a = a, \tag{5.2.2}$$

$$t_k^a = t_k + \Big(1 - I\{a = q_k\}\Big)\big(t(q_k, a) + \Delta t\big) + I\{a = q_k\}\Delta t. \tag{5.2.3}$$

5.2.4 Rewards

A transition from pre-decision state s_k to post-decision state s_k^a results in a reward for species l $R_{k,l}(s_k, a)$.

The total reward can be expressed as

$$R_k(s_k, a) = min\Big\{\sum_{l=1}^{m} p_l c_a^l,\, g_k\Big\}. \tag{5.2.4}$$

Let $Y_k = \sum_{l=1}^{m} p_l c_a^l$. The expected reward from taking action a is

$$\mathbb{E}\Big[min\Big\{\sum_{l=1}^{m} p_l c_a^l,\, g_k\Big\}\Big|\, c_k\Big] = \int_0^{g_k} Y \times P(Y|c_k)\, dY + g_k \left[\int_{g_k}^{\infty} P(Y|c_k)\, dY\right]. \tag{5.2.5}$$

5.2.5 Transition to Pre-Decision State

At decision epoch $k+1$, a random transition from s_k^a to pre-decision state s_{k+1} is made. Let $s_{k+1} = ([q_{k+1}, g_{k+1}, t_{k+1}, c_{k+1}])$ and $s_k^a = \left(\left[q_k^a, g_k^a, t_k^a, c_k^a\right]\right)$. Then,

$$g_{k+1} = \left\{ g_k^a - min\left(\sum_{l=1}^{m} \{c\}_{q_k}^l, g_k^a \right) \right\} \quad , \tag{5.2.6}$$

$$\forall i \in \{1,2,...,n\}, \forall l \in \mathscr{M} \quad \{c_{k+1}\}_i^l = \begin{cases} \{c_k^a\}_i^l & , i \neq q_k^a \\ \left[\{c\}_i^l, \{c_k^a\}_i^l \right] & , i = q_k^a \end{cases}, \tag{5.2.7}$$

$$P(c|c_k) = f(c|c_k). \tag{5.2.8}$$

where c is a new observation vector given observations up to decision epoch k. The conditional probability of observing c given observations up to period k, c_k^a is dependent on the probability distribution f which in turn is dependent on the underlying probability of catch in each fishing ground.

5.2.6 Criterion and Objective

We seek a deterministic Markovian policy that maximizes the total expected reward. A policy $\pi \in \prod$ is a function $\delta^\pi(s) : s \longrightarrow \mathscr{A}(s)$ that maps each state to an action. Let $\prod$ be the set of all Markovian deterministic policies. The decision criterion for a policy is given by

$$V_0^\pi = \mathbb{E}\left\{ \sum_{k=0}^{K} R_k(s_k, \delta^\pi(s_k)) \,\middle|\, s_0 \right\}. \tag{5.2.9}$$

We seek a policy $\pi^\star$ such $V_0^{\pi^\star} \geq V_0^\pi$ for all $\pi \in \prod$.

5.3 Solution Method

In this section, we describe the solution approach for the Markov decision problem presented in the previous section. An optimal solution, which is a function, to the MDP is characterized by the Bellman optimality equation 5.3.1. This function, $\pi(s_k)$, is said to be optimal if it maximizes the right hand side of equation 5.3.1 for all states s_k. The Bellman equation relates the value of being in state s_k, $V_k(S_k)$, with the value of being in all the possible future states (optimal rewards-to-go) if an optimal policy (function) is followed. However, typically, this entails solving for a function that can have an exponentially large domain which is not computationally feasible. One suboptimal methodology to address this challenge is to use rollout. In a rollout technique, one constructs a suboptimal solution sequentially by searching through feasible actions to take in the current state based on approximate rewards-to-go at the current state. One evaluates a feasible action by generating and evaluating many policies to follow from the current state in order to gauge the value of taking this action at this state.

We use the rollout algorithm framework as it is used in Goodson et al. (2013b). This particular rollout framework includes methods that make use of the difference between pre- and post-decision state variables in order to overcome the dimensionality brought forward by an exponentially large

state space.

$$V_k(S_k) = \max_{\pi(s_k)} \{R_k(s_k, a) + V_{k+1}(S_{k+1}(S_k, a))\} \tag{5.3.1}$$

Here, we give a high level description of the solution approach we use. We refer the reader to Goodson et al. (2013b) and the manuscript Goodson et al. (2014) for details.

We repeat some definitions from Section 5.2 here for convenience. Let Π be the set of all deterministic Markovian policies. A deterministic Markov policy π is a sequence of decision rules: $\pi = \left(\delta_0^\pi, \delta_1^\pi, \delta_2^\pi, ..., \delta_K^\pi\right)$ where each $\delta_k^\pi : s_k \longrightarrow \mathscr{A}(s_k)$ is a function that maps a state, s_k, to an action choice a in $\mathscr{A}(s_k)$. So, a policy tells us what action is to be chosen at each decision epoch k. A *restricted policy class* is a subset of all the deterministic markovian policies, $\overline{\Pi} \subset \Pi$. A *heuristic* is any method to select decision rules for all subsequent decision epochs starting from state s denoted by $\mathscr{H}(s)$. Note, s can be a pre- or post-decision state as defined in Section 5.2. A heuristic policy for a given heuristic $\mathscr{H}(s)$ is $\pi_{\mathscr{H}(s)} = \left(\delta_k^{\mathscr{H}(s)}, \delta_{k+1}^{\mathscr{H}(s)}, ..., \delta_K^{\mathscr{H}(s)}\right)$. We search for a routing policy within a space of restricted heuristic policies for the fish trawler.

A rollout algorithm returns a rollout policy that is constructed sequentially (Algorithm 5). Starting from an initial state s_0, the rollout algorithm iterates until a terminal state is reached. At each iteration, the rollout decision rule chosen delivers a current action to take (Line 6). Then, system dynamics lead to a new state for this chosen action (Line 7). The action is chosen (Line 5) based on suboptimal heuristic policies generated by a heuristic routine for states reachable from the current state, s_k (Lines 3, 4). In our case, an action is a fishing ground to harvest, where we allow for the current fishing ground to be in the action space, i.e. to stay at the current fishing ground for another period. The system dynamics govern how much catch, c_{k+1}, is made after each fishing cycle at a fishing ground (Line 7).

Algorithm 5 Rollout Algorithm

1: *Initialize* $k \leftarrow 0$, $s_k \leftarrow s_0$.
2: **while** s_k is not a terminal state
3: **for** all states reachable from s_k t steps into the future
4: Apply **HeuristicRoutine**$(\cdot)$ to generate fixed-route policy $\pi^{s_k}(v)$.
5: Select action **a** according to respective decision rule.
6: $\delta_k^{rollout}(s_k) \leftarrow a$.
7: $s_{k+1} \leftarrow transition(s_k, a, c_{k+1})$.
8: $k \leftarrow k+1$.

We consider the class of a priori policies as the restricted policy class. In our context, an a priori policy means that the fishing vessel will harvest a fixed sequence of fishing grounds. The corresponding decision rules can be expressed in the following way using the notation we introduced before. Given a fixed sequence of fishing grounds v, we have:

$$\delta_k^{\pi(v)}(s_k) = \begin{cases} v_k & \text{, if } g_k > 0 \;\&\; T_k + t(q_k, v_k) + \Delta t + t(v_k, w) \leq Tmax. \\ w & \text{, otherwise.} \end{cases} \tag{5.3.2}$$

According to equation 5.3.2, a fixed sequence of fishing areas $v = (v_1, v_2, ..., v_n)$ is visited as long as current capacity (g_k) and the trip duration limit $(Tmax)$ are not violated.

We employ two types of decision rules (Line 5) to generate a rollout policy. First, a pre-decision state decision rule, which is when $t = 0$ in Line 3. This corresponds to applying the heuristic routine once for the current state. Whichever action the heuristic routine delivers becomes our next action to take. Second, a post-decision state decision rule that corresponds to applying the heuristic routine from every post-decision state reachable by following a feasible action in the current state. A post-decision state is reached by taking a half-step, that is, we have not observed catch for decision epoch $k+1$ but we have moved to a new fishing ground. The action that delivers the best approximate rewards-to-go becomes the next action to take in the rollout policy. See Figure 5.3.1 for a summary of the discussion so far.

While searching from among candidate policies, each policy needs to be evaluated. One method to evaluate a policy is to use a certainty equivalence approach where we make a point-estimate for the value of the future state variables. We use survey data to make these predictions for the Baltic cod in Section 5.4.

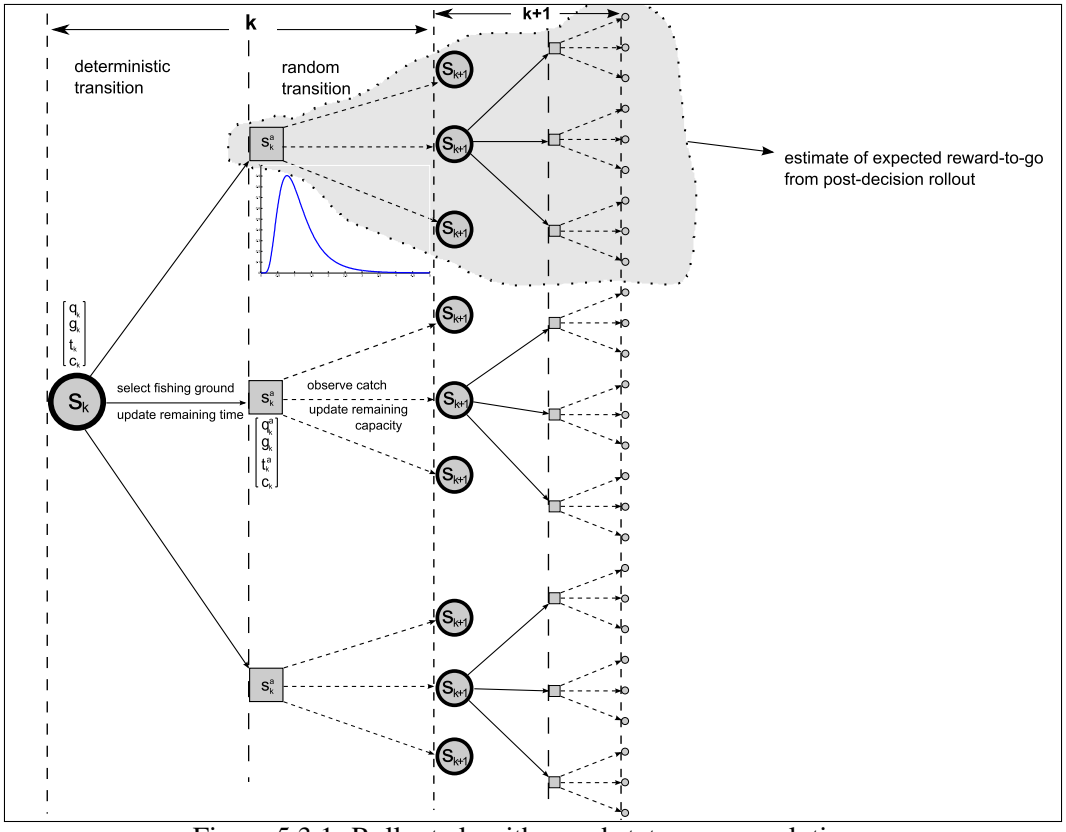


Figure 5.3.1: Rollout algorithm and state space evolution.

The rollout algorithm makes use of a heuristic routine that searches for locally-optimal a priori policies at a given state. The dimension of the problems we consider can be large enough that complete enumeration of all feasible routing policies is computationally prohibitive. We, therefore, make use of a variable neighborhood search method (Thomas and Manni, 2014) that explores the

space of all feasible routing policies. Goodson et al. (2013b) employ a local search procedure that uses a relocation neighborhood and a first-improving acceptance criterion.

A useful conceptualization of the solution approach is to consider a matrix whose rows represent nodes on a graph and whose columns represent time points. See Figure 5.3.2 for a representation. Based on previous observations (represented as stars in the figure), the rollout algorithm evaluates future feasible paths that traverses the cells of this matrix where a visit to a matrix cell accrues a random reward at a particular time point at a particular node. The evaluation of these cells hinges on the forecast used to evaluate future time periods.

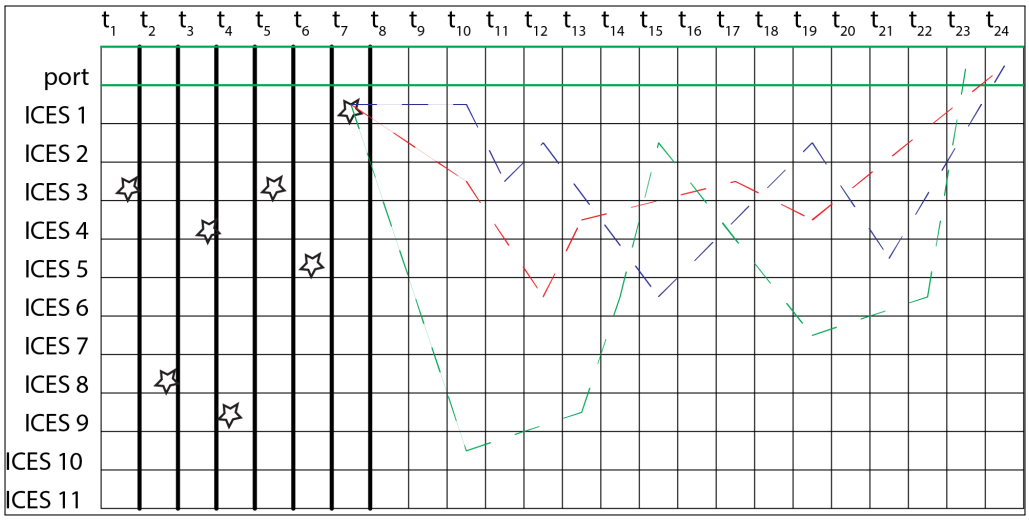


Figure 5.3.2: Path evaluation.

We next discuss the application of our solution method to the Baltic Sea cod case.

5.4 Data Analysis for Baltic Sea Cod

In this section, we discuss the data and prediction algorithms used to predict the efficacy of future fishing locations. Our methods incorporate the spatial and temporal correlation of catch between fishing locations in the South Baltic sea.

The Baltic International Trawl Survey (BITS) is conducted by the International Council for the Exploration of the Sea (ICES). The survey is aimed at accurately assessing the stock abundance for target species of cod and flounder. The survey reflects the variability of the pattern distribution of target species based on ICES areas and depth layers. Traditionally, the survey is conducted twice annually, first in spring (15 February - 31 March) and second in November (1 November - 30 November). However, additional surveys have been conducted for other time periods since 2001. The survey data is collected by 23 different vessels. One of the main focuses of the survey is the evaluation of fishing efficiency (catch per unit effort, i.e. cpue).

The data set we use can be obtained from the ICES data portal website (`http://www.ices.dk/marine-data/data-portals/Pages/DATRAS.aspx`). It contains a total of 400,647 records spanning trawls from 1991 to 2014. The records contain information about catch per haul per length

of species including other information. See Table 5.4.1 for details. Out target variable is the CPUE index (in number of fish). There are 261 records with missing depth field values. There are 279 records with missing Subarea field values. Sex values do not exist in this data set. The data set is created after an extensive data screening process, so outliers are not likely to be due to noise or mismeasurement. About ten percent of the samples are zero catches.

Field Name	Field Description	Field Type
Survey	BITS	categorical
Year	years 1991 to 2014	integer
Quarter	1,2,3,4	categorical
Ship	Ship codes at `http://vocab.ices.dk/?ref=3`.	categorical
Gear	Gear codes at `http://vocab.ices.dk/?ref=2`.	categorical
HaulNo	Number identifying a unique haul	integer
ShootLat	Latitude where fishing haul began	numerical
ShootLon	Longitude where fishing haul began	numerical
DateTime	Date and time of haul	numerical
Depth	Depth at which the trawl sampled	numerical
Area	ICES subdivision (statistical region)	categorical
Subarea	Depth stratum code	categorical
DayNight	Day or Night during haul	categorical
AphiaID	`http://www.marinespecies.org/`	categorical
Species	Latin Name for species caught	categorical
Sex	Sex codes at `http://vocab.ices.dk/?ref=17`.	categorical
LngtCls	millimetres	numerical
CPUE-number-per-hour	Catch in numbers per hour of hauling	numerical

Table 5.4.1: CPUE data set information.

One important characteristic of the target variable (cpue) is that there is a drastic difference of the cpue index based on day or night. Figures 5.4.1 and 5.4.2 show the effect of daylight on the cpue index. Figure 5.4.1 shows that there is more opportunity during the day to make larger catches where central south Baltic sea is a hot spot in terms of cod abundance. Figure 5.4.2 shows that harvesting during the night is associated with a higher probability of less catch.

In Figure 5.4.3, we see the dispersion of the trawl locations in the South Baltic sea over the years. It includes ICES areas 25 and 26 where most trawls have been carried out. As can be seen from the figure, not all regions are equally sampled. More observations are made where cod species is believed to be in more abundance. These observations are made using a stratified sampling approach where each strata is a combination of a statistical rectangle and sea depth level. The survey has taken into account the distribution pattern of these species when planning where to carry out trawling operations.[1]

5.4.1 Prediction

In this section, we present spatio-temporal methods for catch prediction. We first discuss three simple forecasting methods. Then, we consider an exponential smoothing method. After, we discuss

[1] `https://datras.ices.dk/Documents/Manuals/Manuals.aspx`

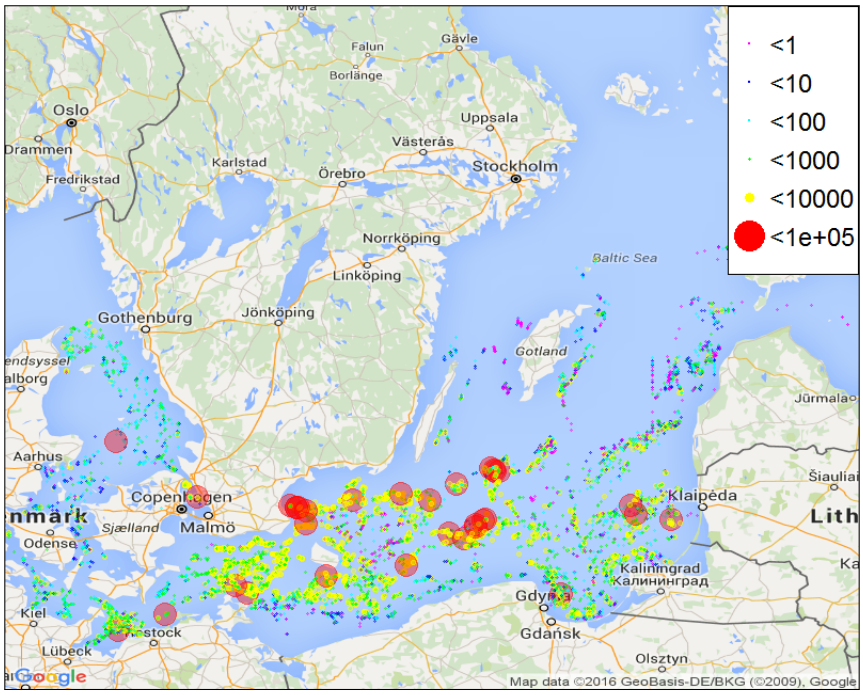


Figure 5.4.1: Cpue during the day.

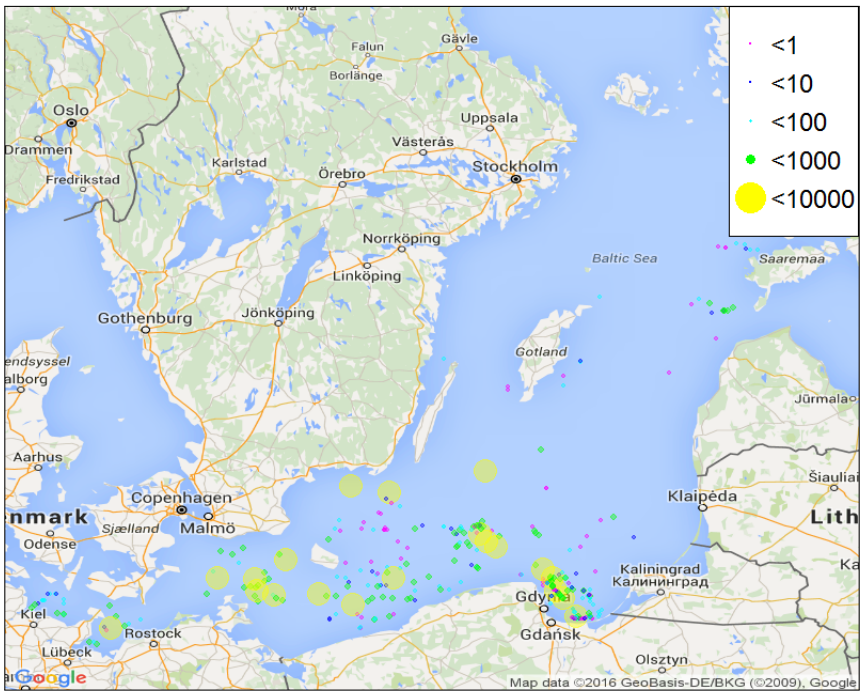


Figure 5.4.2: Cpue during the night.

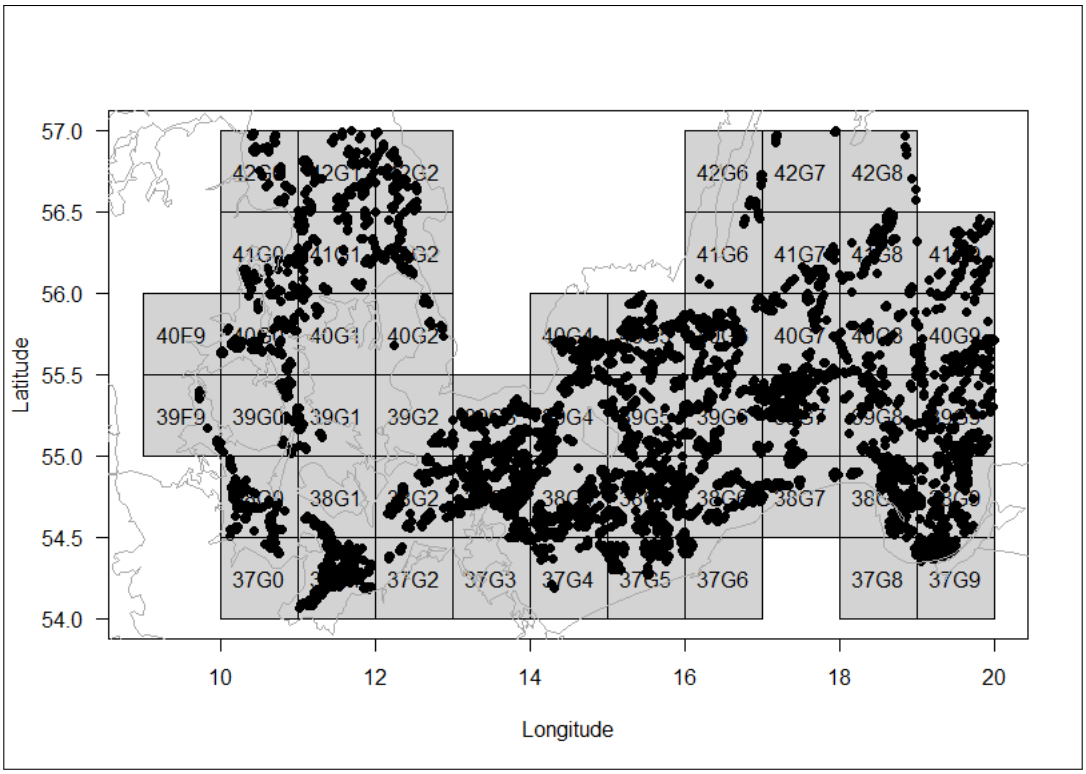


Figure 5.4.3: South Baltic Sea haul locations.

kriging methods, which are geostatistical methods for spatial and temporal prediction. Finally, we consider a neural network approach that predicts catch using a different model for every ICES rectangle. This model is intended to model different catch probability based on specific location information as opposed to kriging models, which model catch prediction based only on geographical distance between locations. Here, we focus on a general model with all the relevant statistical rectangles included.

We use catch data from 2000 to 2012 for training and testing purposes. As our problem involves making future predictions, we train a forecasting method using a sliding window data partitioning approach. More specifically, we sort data points from oldest to most recent. A single training set is created using N observations starting from the first observation in this sorted data. The testing set is composed of the next M observations. This gives us a single training/testing partition. The second partition (training/testing pair) is created by selecting N observations starting from the second observation in the sorted data set, and selecting the next M observations for testing. We take $N = 100$ and $M = 15$. For some of the methods, we use all cumulative observations associated with a test set instead of the previous N as described before. We assess the generalization error of the forecasting method using the training/testing folds created in this manner.

Some of the training/testing partitions created contain observations that belong to different quarters in a year. For instance, training observations can fall into quarter 1 of 2011, and the testing observations fall into quarter 3 of 2011. This is because cod harvesting seasons are in quarter 1 and quarter 3 of the year. We remove such cases from our folds as this does not correspond to a real-life

fishing trip that starts and finishes in a single season.

We use three types of simple forecasting (SF) methods. First, we use a moving average type of simple forecasting in which a prediction is made using previous observations (SF-MA) by taking the average of the observations associated with the ICES rectangle of the test point in the training partition. Second, we make a prediction for a test point by taking the average of all cumulative previous observations (SF-Global). Third, we make a prediction for a test point by assigning the historical cumulative average of the ICES rectangle to which the test point is associated with (SF-Spatial).

We next present exponential smoothing, kriging, and a neural network model for prediction. After, we provide the prediction performance of each model.

5.4.1.1 Exponential Smoothing

We use the date-time, latitude, and longitude features for prediction. We first rescale these values to $[0,1]$. To make a prediction for a test point, we compute its distance to each training point. To come up with a smoothed prediction, we use the nearest 5, 10, and 20 observations. Starting from the oldest observation, we iteratively smooth observations according to formula 5.4.1. The prediction for the test point is obtained by computing the final value using this recursive formula.

$$\hat{y}_t = \alpha x_t + (1-\alpha)\,\hat{y}_{t-1} \tag{5.4.1}$$

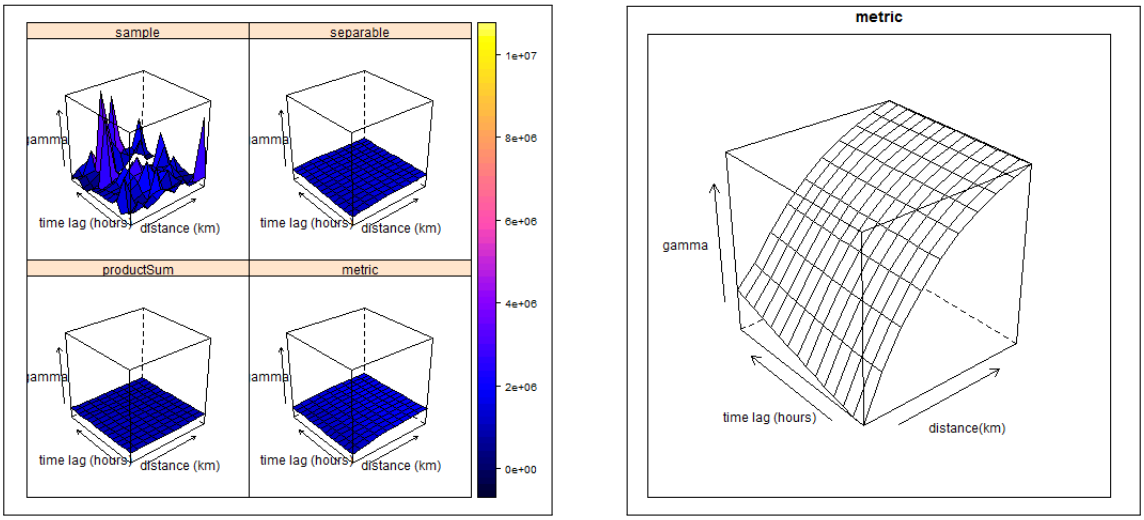
where $\alpha = e^{\mu}$ and μ is the normalized euclidean distance between two nearest observations.

5.4.1.2 Kriging

Kriging is a well-known method for spatial (and temporal) interpolation that uses previous observations to predict unsampled (future) observations. An important function used in Kriging is the variogram function. The variogram function models the structure of the correlation over spatial and temporal distances between observations. One typically employs a parametric form for the variogram function. It is generally assumed that nearby observations are more highly correlated than distant observations and this assumption is reflected in the parametric functions that have been proposed in the literature (Cressie and Wikle, 2015). We use three different parametric forms for the spatiotemporal variogram (De Cesare et al., 2001). These are the separable, metric, and product-sum form. Each of these forms combine simpler one-dimensional variogram functions. We use the exponential one-dimensional variogram for this purpose. To estimate a variogram model numerically, we use the gstat package in the CRAN repository (Pebesma, 2004). Using data from 2007 to 2012, we first compute a sample variogram function and then fit a parametric variogram model to this sample variogram. A sample variogram is estimated by discretizing the distances in the temporal and spatial coordinates. We use time lags of 3 hours with a temporal depth of 24 hours. For the spatial dimension, we use a maximum distance of 100 km binned by 20 km.

There are three main parameters that control the goodness of the fit. These are the range, sill,

and the nugget parameter of the variogram. The range controls the distance at which the variogram surface becomes flat. The sill parameter controls the height of the variogram. The nugget parameter controls where the variogram cuts through the vertical axis. We present the fitted values for these parameters in Table 5.4.2. The fit error is close in all three cases; however, it is marginally better with the metric variogram. Figure 5.4.4 shows the sample variogram and the three fitted variograms over spatial and temporal lags. The metric variogram is zoomed in on in Figure 5.4.4b.



(a) Sample variogram versus fitted variograms.

(b) Fitted metric variogram (zoomed in).

Figure 5.4.4: Fitted variograms.

	Separable	ProductSum	Metric
Sill	1.55E+06	5.00E-01 5.00E-01	1.01E+06
Range	1.50E+02 3.00E+01	3.91E+01 4.27E+01	2.00E+02
Nugget	3.07E-01 3.07E-01	1.10E+01 5.50E-01	9.58E+05
MSE	1.25E+12	1.25E+12	1.25E+12

Table 5.4.2: Fitted variogram parameters.

5.4.1.3 Neural Network Prediction Model

In contrast to kriging prediction that models correlation based only on pairwise spatial and temporal distance between observations, we use separate neural networks to take into account potentially different dependency for different locations. Our modeling strategy is to use previous observations at one region and other regions to forecast catch at this region for every period. We take a period to be of length 3 hours.

Although the survey covers a wide area, a great portion of the sea is unsurveyed for various time periods during a season. In order to be able to make forecasts for unsurveyed areas for every period, we build a predictive model in two stages.

In the first stage, we use a model that can impute values for unsurveyed locations at a particular time point. We use data before 2000 to train models for imputation. The predictive models are trained with data after 2000. Based on this, we form a hybrid dataset that contains real observations and imputed observations for particular space-time pairs. The forecasting model that uses previous observations will be based on this hybrid dataset. We discuss each stage next.

5.4.1.4 Filling Missing Values for Unsurveyed Space-time

We use two different methods to fill in missing unsurveyed space-time values. First, we use a standard a two layer feed-forward neural network whose first layer uses a radial basis transfer function and the second layer (output layer) uses a linear transfer function. This neural network is trained using features such as latitude, longitude, and temporal features. Second, we use kernel regression that uses nearby sample points weighted by a kernel function.

We vary the number of hidden nodes from 50 to 1000 for the neural network. We create 10 folds of training and testing partitions for cross validation. The observations are allocated into each set randomly. We present the performance of each imputation method in Table 5.4.3.

Model (hidden neurons)	Cross Validation MSE
50	4.92E+06
100	4.88E+06
250	4.87E+06
500	4.87E+06
750	4.85E+06
1000	4.88E+06
Kernel Regression	4.84E+06

Table 5.4.3: Average MSE.

According to performance results in Table 5.4.3, the kernel regression method obtained the lowest mean squared error. We choose this model for our second stage learning.

5.4.1.5 Neural Network for Cod Forecasts

In the second stage, we train a separate feedforward network for every region. Let us focus on the network associated with a single region to describe the architecture of the networks. The input layer is composed of nodes representing previous lagged observations for every region, while the output layer consists of a single node representing current catch at this region. Each hidden node in the hidden layer represents a radial basis transfer fuction that processes the input it receives from the input layer and forwards its output to the output layer. The output layer processes the input received from the hidden layer using a linear transfer function. We use such a function in the output layer to predict a continuous variable that can take arbitrary values from 0 to infinity.

The data used to train the network is a hybrid dataset that uses both real observations as well as imputed values for various space-time lags. We employ backpropagation algorithms implemented in the Matlab neural toolbox for training. We use rolling-time horizon validation.

We carry out experiments to determine the neural network structure that performs best. We include as input previous catch from all regions in the prediction for region i. The number of lagged observations included for each region is most recent 8 periods (lags 1-8, previous 24 hours). All other regions have previous day lags (lags 9-16, between 24-48 hours ago) included as input. Since there are 84 ICES rectangles in the dataset, the input layer is of dimension 672 (84*8) as a result. Each period corresponds to a 3 hour length of time. We use the mean squared error measure to evaluate the models in each experiment.

We present the performance measure of correlation and root mean squared error (RMSE) for each predictive model in Table 5.4.4. Kriging interpolation with the metric covariance function form does best. Moreover, simple forecasting performs relatively well. The hybridNeural technique we propose does not do as well. However, neither of these methods obtain high correlation with the cpue observations. This can be due to model selection as well as lack of more fine-grained data with a higher frequency of sampling over spatial and temporal coordinates.

Model	RMSE	R2 (correlation)	RMSE std. dev	R2 std. dev
Kriging: Metric	1007.60	0.45	871.90	0.20
Kriging: Separable	1026.10	0.39	857.90	0.18
SF-Spatial	1048.00	0.50	792.00	0.20
Kriging: Product-Sum	1058.70	0.39	852.20	0.17
SF-Global	1164.40	-	834.00	-
SF-MA	1212.50	0.44	1030.90	0.22
hybridNeural	1285.50	0.22	950.00	0.35
ES (nearest 20)	1485.60	0.33	1935.70	0.15
ES (nearest 10)	1552.10	0.36	1866.18	0.17
ES (nearest 5)	1717.90	0.37	3723.00	0.19

Table 5.4.4: Average performance of predictive models.

5.4.2 Prescription

In this section, we carry out experiments to compare routing policy performance with respect to predictive models. We use three different routing policies. These are the a priori, predecision, and postdecision routing policies. In combination with each type of route generating policy, we use the Kriging (Metric), Kriging (Separable), SF-Spatial, and hybridNeural prediction methods. The predictive methods are useful to evaluate future cod catchability at various regions. The routing heuristics discussed in Section 5.3 choose to visit a fishing region based on this evaluation of future cod catchability.

5.4.2.1 Experimental Design

To test the performance of different combinations of a routing strategy and a predictive model, we use March 2013 data to create test (validation) instances that correspond to a weekly trip in March of 2013. These serve to evaluate which policy plus predictive model combination will perform better in reality, thus have better generalizable performance.

For March 2013 in particular, the survey has been conducted only for 37 different ICES rectangles. Accordingly, our test instances use these ICES rectangles only. A major challenge is how to handle the case if the decision heuristic chooses to visit an ICES rectangle with no actual observation for a particular time period. In fact, starting from March 1, 2013 until March 21, 2013, there are only 166 sample points in our data set. However, a complete experimental test case should consist of an observation every three hours for all 37 ICES rectangles from March 1st to March 21st. In order to overcome the sparsity and make a fair comparison of the various prediction and decision heuristics, we use standard interpolation techniques to complete our observations. We use leave-one-out (LOO) cross-validation to estimate the true error rate of several interpolation techniques. The LOO cross-validation method has a better estimation ability of the true error rate from among model evaluation methods such as the hold-out method, k-fold cross-validation, and repeated subsampling.

We use all sample points from 1991 to 2014 that fall in the months of February, March, and April. Additionally, we use features including hour, day, month, latitude, and longitude to interpolate cpue in March of 2013. We experiment with various interpolation methods including K-nearest neighbors, radial basis function interpolation, generalized regression neural networks, and linear interpolation. We find that K-nearest neighbors with $K = 3$ neighbors has the smallest average RMSE measure (Table A1 in the Appendix). Consequently, we create test instances based on this specific interpolation method.

We use one week's worth of data to create each test instance whose beginning date is shifted one period (3 hours) into the future. In each test instance, we treat observations that fall into the first two periods as known to the decision maker.

The test instances assume a vessel with 165 hours time to complete a trip, a tank capacity to hold $50,000$ fish, and we assume that a vessel visits a fishing region at most twice at any particular time before the duration limit is reached. We use the geographical location of Rostock (Germany) from which vessels operate and land their catch.

We solve a total of 107 test instances. Each time we use a different combination of the four predictive models Kriging (Metric), Kriging (Separable), SF-Spatial, and hybridNeural. For each of the predictive models, we use the three routing strategies a priori, predecision, and postdecision. We refer to a combination of a predictive model and routing strategy as a prescriptive model, in short.

We implement our algorithms using C++ and execute them on a high performance cluster system. The computations are carried out on 2.6GHz machines with 16 cores and 64 GB RAM. These machines run on the CentOS 6.4 Linux operating system. The heuristic routine is chosen to be restarted 100 times before delivering a final solution.

5.4.2.2 Results

In this section, we present the quality of the average reward that can be earned if a particular combination of a predictive method and rollout method is used. We focus on predictions made by Kriging (Metric), Kriging (Separable), SF-Spatial, and hybridNeural.

Upon our initial experiments, we observe that rerouting strategies do not necessarily lead to better outcomes. However, (Goodson et al., 2014) show that when exact solution evaluation is possible, then a rollout algorithm can be fortified to deliver better results than an a priori solution. Although we cannot evaluate a solution exactly, nonetheless, we implement rollout with fortification, which is a simple variation to a rollout algorithm that updates its policy only if it has found a better policy to follow. See Goodson et al. (2013a, 2014) and Bertsekas (2013) for more detail.

We present the average runtime (in seconds) for a single epoch in Table 5.4.5. The rows correspond to the rollout method and columns are organized by predictive method. The a priori heuristic takes about 5 seconds, on average. The predecision heuristic takes on average close to 3 seconds per epoch. In the initial epochs, the predecision heuristic takes longer to execute than at later epochs when the number of feasible routes has decreased significantly. The postdecision heuristic takes the most time per epoch since it applies the heuristic from every feasible action at a given state. On average, it takes about 67 seconds per epoch. Note, adding to the computation time, the algorithm communicates with an external program that generates predictions at the beginning of every epoch.

	hybridNeural	Kriging (Metric)	Kriging (Separable)	SF-Spatial
a priori	5.01	6.10	7.49	4.93
predecision	2.18	2.78	2.80	5.45
postdecision	61.40	72.97	71.59	64.72

Table 5.4.5: Average runtime per epoch for each combination (seconds).

In Table 5.4.6, we present the average reward earned over all the instances for every predictive and rollout method. The rerouting strategies do better across all the predictive methods. Moreover, Kriging (Metric) and SF-Spatial do about equally well for predecision and postdecision state policy generation. Kriging (Metric) interpolation performs best with an average of about $41,891$ cod caught during a one-week trip in March 2013. SF-Spatial produces an average of about $41,793$ cod. Kriging (Separable) produces an average of $41,416$ cod. The hybridNeural method is only able to deliver $30,512$ cod, on average.

	SF-Spatial	Kriging (Metric)	Kriging (Separable)	hybridNeural
a priori	38,971.75	39,279.47	40,697.21	28,519.61
predecision	42,822.41	43,318.33	42,138.93	31,632.78
postdecision	43,587.46	43,076.29	41,411.91	31,386.47
ave. reward	41,793.87	41,891.36	41,416.02	30,512.96

Table 5.4.6: Average reward for each combination (number of cod).

Figure 5.4.5 shows the distribution and mean reward earned by each routing strategy. Although

there is a visual overlap between the different strategies, Table 5.4.7 shows that the a priori strategy is statistically worse than the predecision and postdecision strategy. However, there is no statistical difference between the predecision and postdecision strategies.

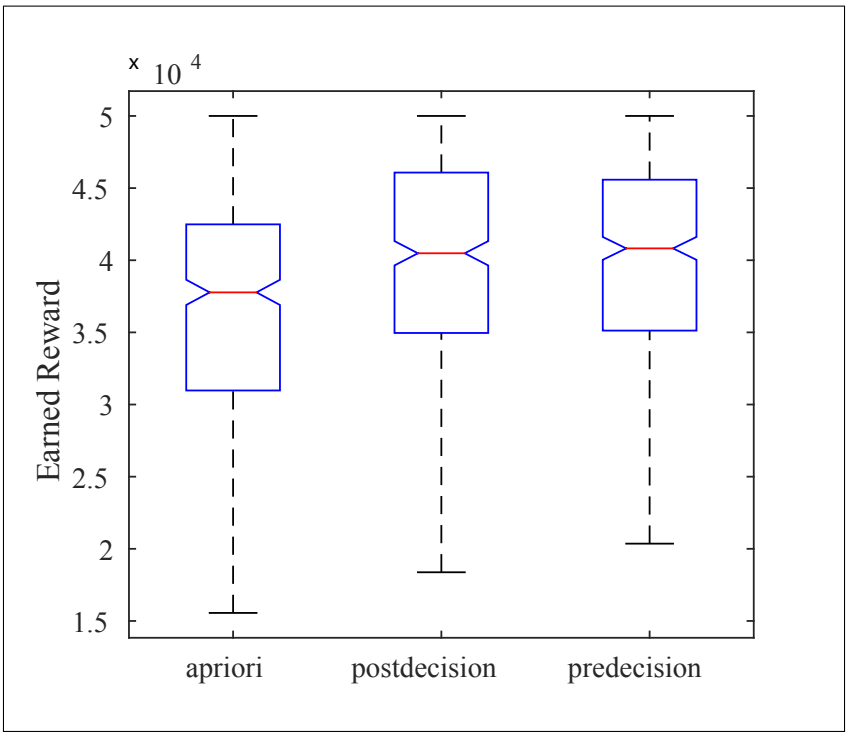


Figure 5.4.5: Mean reward comparison.

	Lower 95%	Mean Difference	Upper 95%	p-value
(a priori − postdecision)	-4,215.50	-2,998.50	-1,781.50	2.40E-08
(a priori − predecision)	-4,328.10	-3,111.10	-1,894.10	7.16E-09
(postdecision − predecision)	-1,329.60	-112.5811	1,104.40	9.74E-01

Table 5.4.7: Pairwise differences in mean reward.

A predictive method, used in the application of the heuristic, does better ultimately because it enables the variable neighborhood search heuristic to compare solutions correctly, i.e. routes that would yield higher reward in actuality (according to the test instance) are ranked higher by the heuristic evaluation of each route. It is interesting to understand the relationship between the quality of prediction and the quality of routing.

For the following discussion, we consider the predecision routing strategy to compare all the predictive methods. In Figure 5.4.6, a point corresponds to an RMSE error of a specific prediction model on the groundtruth of a test instance and the associated reward as a result of applying the predecision routing strategy on that test instance. There is a point for each of the prediction models from Table 5.4.4 and test instance. There is a center point associated with each prediction model marked by a colored shape. These centers are the mean RMSE and mean reward for each prediction model. We can see that higher reward is, on average, associated with better squared error in RMSE.

The dashed line is a linear fit through all the points and the continuous line is a quadratic fit through all the points. There is a lower bound to the RMSE that can be attained (no prediction model is going to achieve zero error); as a result, the quadratic line seems to be a better fit. Moreover, in Figure 5.4.7, the relationship between prediction performance in terms of correlation and higher reward is positive and linear. This is also a strong indication between predictive performance and better optimization.

The top predictive methods that lead to good routing performance on the groundtruth test instances are SF-MA, Kriging-Metric, SF-Spatial, Kriging-Separable, and Kriging-ProductSum. Their groundtruth RMSE measures as well as optimization performance are close to each other. Note, although the cross-validation measure for SF-MA is relatively worse in Table 5.4.4, SF-MA's predictions are marginally better on the test instances (green plus sign in Figure 5.4.6). However, the groundtruth RMSE measure for SF-Spatial, Kriging-Metric, Kriging-Separable, and Kriging-ProductSum are among the best (centers in the lower right), which is as expected based on their cross-validation performances (Table 5.4.4). These are also associated with the top routing performance.

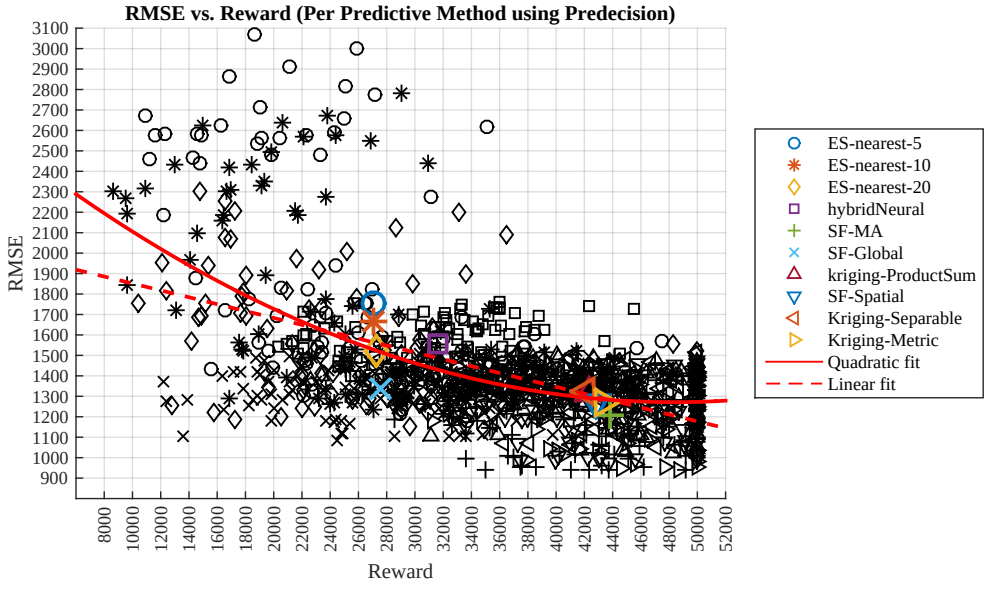


Figure 5.4.6: RMSE vs. earned reward (per predictive method).

In Figures 5.4.8a, 5.4.8b, and 5.4.9 we present the geographical route prescribed by the a priori, predecision, and postdecision heuristic between March 1st, 2013 to March 8th, 2013, respectively. These paths suggested by the heuristics seem to capture the regions where more abundant cod is likely to be during the day (Figure 5.4.1).

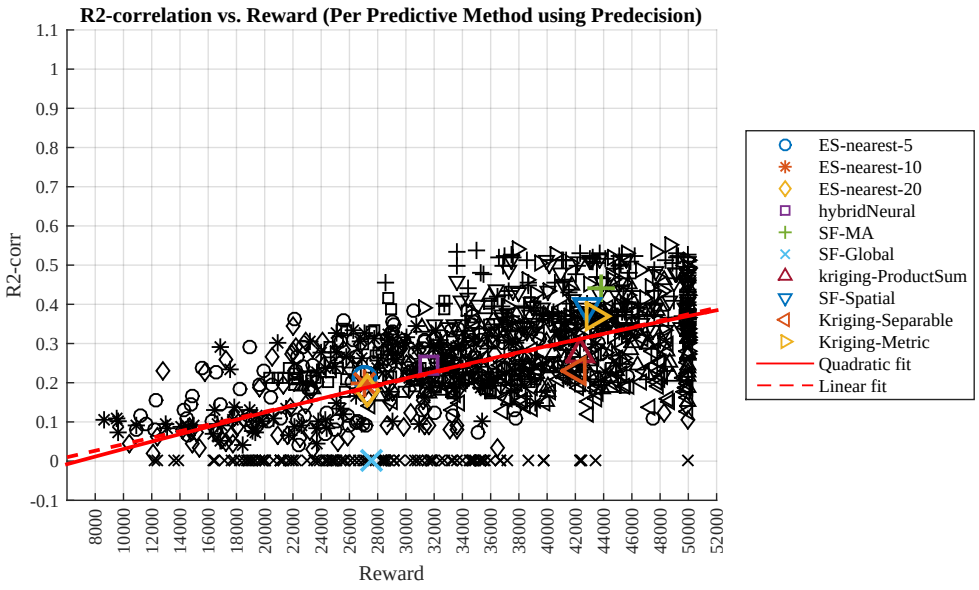
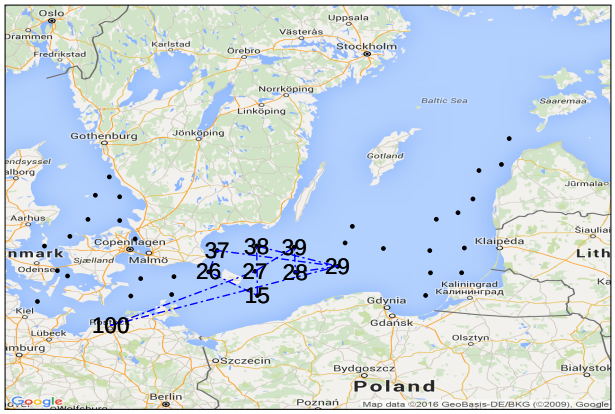


Figure 5.4.7: ρ vs. earned reward (per predictive method).



(a) A priori, number of cod = 47,232.

(b) Predecision, number of cod = 50,000.

Figure 5.4.8: Kriging routes.

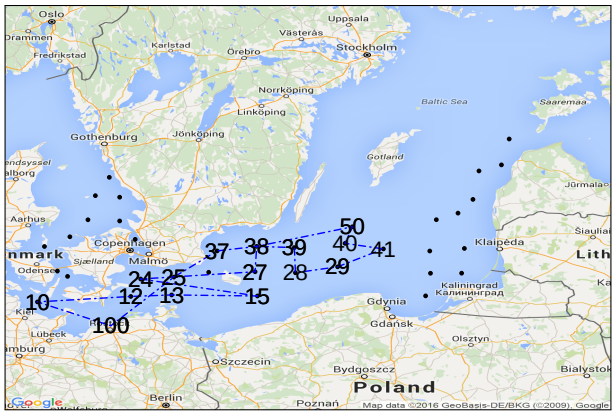


Figure 5.4.9: Kriging, Postdecision, number of cod = 50,000.

5.5 Conclusion and Future Work

Our experimental results rely on test instances generated by interpolation. We have tested the quality of our estimation with LOO cross-validation; however, there is room for improving the estimation. In K-nearest neighbor interpolation, features and/or neighbors can be weighted. In addition, nearest neighbors can be chosen using mutual information over the features.

A potentially fruitful question is to consider machine learning techniques that focus on ranking outcomes instead of minimizing traditional error measures. Any solution approach that compares its best solution to the incumbent solution, under uncertainty, is influenced by the probability of making a correct comparison which is more of a ranking problem than an estimation problem. Techniques that focus on such a goal could be valuable (Burges et al., 2005; Cao et al., 2007; He et al., 2008; Liu, 2009).

We used a certainty equivalence approach while searching for better solutions. This approach associates a scalar to every solution, which may contain a significant bias with respect to the true value of a solution. Monte carlo simulation is one method that can reduce the bias in the evaluation of routing solutions. It works by simulating the reward that could be earned along a policy and taking their average. Note, this will significantly increase computational time unless the algorithm is carefully designed.

We have shown that rerouting strategies are beneficial even under relatively simple stochastic dynamic policy classes such as the a priori policy class. We used a real data set to build predictive models and generate test instances to validate our solution approach. We explored the relationship between the quality of prediction and routing performance. We suggest that a better predictive performance (squared error) implies higher earnable average reward. This suggestion is in line with the analytical discussion on approximate objective function evaluation in Secomandi (2003), although we do not prove the regularity conditions assumed therein in order for rollout strategies to be preferable. Moreover, the correlation performance of a models predictions is also important.

Chapter 6

Summary

In this thesis, we propose several methodologies for routing problems that integrate data-analytic methods into their solution approach.

In Chapter 3, we propose a scalable cluster-first, route-second approach to the vehicle routing problem. Our approach focuses on the shape of the clusters to obtain efficient routes. We model the objective function of a standard clustering problem by a special distance function that takes into account polar and angular coordinates between customer locations. Our solution approach is easy to maintain and competitive to benchmark VRP solver.

In Chapter 4, we incorporate data to create consistency in parcel delivery problems across multiple periods. We do this by adding extra constraints to a clustering problem that optimizes over the degree of consistency. We quantify the degree of consistency mathematically by the information entropy function, which models consistency in a lexicographic fashion. Our solution approach is less restrictive than existing approaches.

In Chapter 5, we focus on a fish trawling problem that allows for correlated catch across fishing regions. We incorporate data via forecasting to reduce error on solution evaluation during local search. We shed light on the relationship between predictive performance and quality of routing. Qualitatively, if a model performs better in RMSE and correlation, it tends to deliver better quality routes, on average.

Appendix

Method	RMSE	Correlation
Knn, K=3	2859.2*	0.62*
Kernel regression	3600.1	0.18
Linear Interpolation	3788.7	0.33
Rbf Interpolation	3888.3	0.003
Grnn	4525.7	0.19

Table A1: Cross-validation performance.

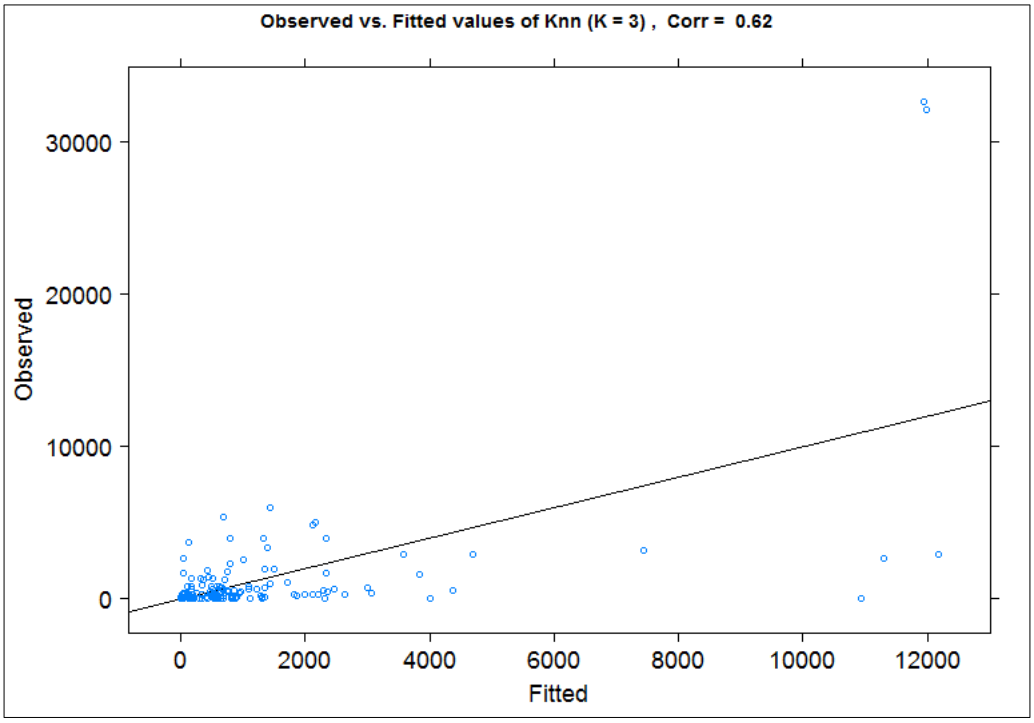


Figure A1: Observed vs. fitted values for 2013 March cpue data.

Bibliography

Driving success: Why the UPS model for managing 103,500 drivers is a competitive advantage. http://pressroom.ups.com/mediakits/popups/factsheet/0,1889,1201,00.html, 2007.

Meet ORION, Software That Will Save UPS Millions By Improving Drivers' Routes. http://www.forbes.com/sites/alexkonrad/2013/11/01/meet-orion-software..., May 2014.

N Absi, C Archetti, S Dauzère-Pérès, and D Feillet. A two-phase iterative heuristic approach for the production routing problem. *Transportation Science*, 49(4):784–795, November 2015. 00012.

Agostinho Agra, Marielle Christiansen, Rosa Figueiredo, Lars Magnus Hvattum, Michael Poss, and Cristina Requejo. The robust vehicle routing problem with time windows. *Computers & Operations Research*, 40(3):856–866, March 2013.

Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. OPTICS: Ordering Points to Identify the Clustering Structure. SIGMOD '99, pages 49–60. ACM, August 1999.

D Applegate, R Bixby, V Chvátal, and W Cook. Concorde TSP Solver, 2006. *http://www. tsp. gatech. edu/concorde*, 2006.

P. Augerat, J.M. Belenguer, E. Benavent, A. Corberan, D. Naddef, and G. Rinaldi. Computational results with a branch and cut code for the capacitated vehicle routing problem. *Rapport de recherche - IMAG*, 1, 1995.

Nabila Azi, Michel Gendreau, and Jean-Yves Potvin. A dynamic vehicle routing problem with multiple delivery routes. *Annals of Operations Research*, 199(1):103–112, 2012.

Elizabeth A Babcock and Ellen K Pikitch. A dynamic programming model of fishing strategy choice in a multispecies trawl fishery with trip limits. *Canadian Journal of Fisheries and Aquatic Sciences*, 57(2):357–370, February 2000.

Barrie M. Baker and MA Ayechew. A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30(5):787–800, 2003.

Roberto Baldacci, Daniele Vigo, Paolo Toth, James J Cochran, Louis A Cox, Pinar Keskinocak, Jeffrey P Kharoufeh, and J Cole Smith. Exact Solution of the Capacitated Vehicle Routing Problem. In *Wiley Encyclopedia of Operations Research and Management Science*. John Wiley & Sons, Inc., 2010.

Gulay Barbarosoglu and Demet Ozgur. A tabu search algorithm for the vehicle routing problem. *Computers & Operations Research*, 26(3):255–270, 1999.

J E Beasley. Fixed Routes. *The Journal of the Operational Research Society*, 35(1):pp. 49–55, 1984.

J E Beasley and N Christofides. Vehicle routing with a sparse feasibility graph. *European Journal of Operational Research*, 98(3):499–511, May 1997.

Tolga Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, June 2006.

K P Bennett and O L Mangasarian. Bilinear separation of two sets in n-space. *Computational Optimization and Applications*, 2(3):207–227, 1993.

David E. Benson. *Load Plan Selection For Package Express Fleets*. Ph.d., Univerity of Michigan, 2001.

Russell W. Bent and Pascal Van Hentenryck. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52(6):977–987, December 2004.

Gerardo Berbeglia, Jean-François Cordeau, and Gilbert Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, April 2010.

Jean Berger and Mohamed Barkaoui. A hybrid genetic algorithm for the capacitated vehicle routing problem. In *Genetic and Evolutionary Computation GECCO 2003*, pages 646–656. Springer, 2003.

Sophie Bertrand, Arnaud Bertrand, Renato Guevara-Carrasco, and François Gerlotto. Scale-invariant movements of fishermen: The same foraging strategy as natural predators. *Ecological Applications*, 17(2):331–337, March 2007.

Dimitri P Bertsekas. Rollout algorithms for discrete optimization: A survey. In *Handbook of Combinatorial Optimization*, pages 2989–3013. Springer, 2013.

D J Bertsimas and G Van Ryzin. Stochastic and dynamic vehicle routing in the Euclidean plane with multiple capacitated vehicles. *Operations Research*, pages 60–76, 1993.

Dimitris J Bertsimas and Garrett van Ryzin. A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane. *Operations Research*, 39(4):pp. 601–615, 1991.

Nicolas Bez, Emily Walker, Daniel Gaertner, Jacques Rivoirard, Philippe Gaspar, and Carl Walters. Fishing activity of tuna purse seiners estimated from vessel monitoring system (VMS) data. *Canadian Journal of Fisheries & Aquatic Sciences*, 68(11):1998–2010, November 2011.

J Bishop, W N Venables, C M Dichmont, and D J Sterling. Standardizing catch rates: is logbook information by itself enough? *ICES Journal of Marine Science: Journal du Conseil*, 65(2): 255–266, March 2008.

Janet Bishop, W N Venables, and You-Gan Wang. Analysing commercial catch and effort data from a Penaeid trawl fishery: A comparison of linear models, mixed models, and generalised estimating equations approaches. *Fisheries Research*, 70(23):179–193, December 2004.

Donald J Bowersox, David J Closs, and M Bixby Cooper. *Supply chain logistics management*, volume 2. McGraw-Hill New York, 2002.

Burcin Bozkaya, Erhan Erkut, and Gilbert Laporte. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research*, 144(1):12–26, January 2003.

Paul S Bradley, Olvi L Mangasarian, and W Nick Street. Clustering via concave minimization. *Advances in neural information processing systems*, pages 368–374, 1997.

Julien Bramel and David Simchi-Levi. A location based heuristic for general routing problems. *Operations Research*, 43(4):649–660, August 1995.

Julien Bramel and David Simchi-Levi. Probabilistic analyses and practical algorithms for the vehicle routing problem with time windows. *Operations Research*, 44(3):501, May 1996.

Julien Bramel and David Simchi-Levi. On the effectiveness of set covering formulations for the vehicle routing problem with time windows. *Operations Research*, 45(2):295, March 1997.

Julien Bramel, Edward G Coffman Jr., Peter W Shor, and David Simchi-Levi. Probabilistic Analysis of the Capacitated Vehicle Routing Problem with Unsplit Demands. *Operations Research*, 40(6): 1095–1106, November 1992.

Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96. ACM, 2005.

A M Campbell and B W Thomas. Challenges and advances in a priori routing. *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 123–142, 2008a.

Ann M Campbell and Barrett W Thomas. Probabilistic Traveling Salesman Problem with Deadlines. *Transportation Science*, 42(1):1–21, February 2008b.

Ann M Campbell, Michel Gendreau, and Barrett W Thomas. The orienteering problem with stochastic travel and service times. *Annals of Operations Research*, 186(1):61–81, June 2011.

Ann Melissa Campbell and Martin W P Savelsbergh. A decomposition approach for the inventory-routing problem. *Transportation Science*, 38(4):488–502, November 2004.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM, 2007.

J. G. Carlsson. Dividing a Territory Among Several Vehicles. *INFORMS Journal on Computing*, 24(4):565–577, October 2011.

Ping Chen, Hou-kuan Huang, and Xing-Ye Dong. Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem. *Expert Systems with Applications*, 37(2):1620–1627, March 2010.

T.William Chien. Operational estimators for the length of a traveling salesman tour. *Computers & Operations Research*, 19(6):469–478, August 1992.

N Christofides. Fixed routes and areas for delivery operations. *International Journal of Physical Distribution*, 1:87–93, 1971.

Fan R K Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

Gérard C Clarens and VF Hurdle. An operating strategy for a commuter bus system. *Transportation Science*, 9(1):1–20, 1975.

Steven R Clinton. Importance of technology investments in the logistics service providers: A case study of UPS and its use of online tools. *Journal of Applied Business Research (JABR)*, 24(2): 67–80, 2011.

Lee G. Cooper and Giovanni Giuffrida. Turning datamining into a management science tool: New algorithms and empirical results. *Management Science*, 46(2):249–264, 2000.

J-F Cordeau, M Gendreau, G Laporte, J-Y Potvin, and F Semet. A guide to vehicle routing heuristics, 2002.

Jean-François Cordeau and Mirko Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39(9):2033–2050, September 2012.

Jean-François Cordeau, Michel Gendreau, and Gilbert Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30(2):105–119, September 1997.

Noel Cressie and Christopher K Wikle. *Statistics for spatio-temporal data*. John Wiley & Sons, 2015.

Benoit Crevier, Jean-François Cordeau, and Gilbert Laporte. The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, 176(2):756–773, January 2007.

Carlos F Daganzo. The distance traveled to visit N points with a maximum of C stops per vehicle: An analytic model and an application. *Transportation Science*, 18(4):331–350, 1984.

Carlos F Daganzo and Alan L Erera. On planning and design of logistics systems for uncertain environments. In *New Trends in Distribution Logistics*, pages 3–21. Springer, 1999.

Carlos F. Daganzo, Vikash V. Gayah, and Eric J. Gonzales. The potential of parsimonious models for understanding large scale transportation systems and answering big picture questions. *EURO Journal on Transportation and Logistics*, 1(1-2):47–65, April 2012.

L De Cesare, DE Myers, and D Posa. Estimating and modeling space–time correlation structures. *Statistics & Probability Letters*, 51(1):9–14, 2001.

L. Delesie and L. Croes. Operations research and knowledge discovery: a data mining method applied to health care management. *International Transactions in Operational Research*, 7(2): 159–170, 2000.

I S Dhillon and D S Modha. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1):143–175, 2001.

Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Kernel K-means: Spectral clustering and normalized cuts. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 551–556, New York, NY, USA, 2004. ACM.

KarlF. Doerner and Verena Schmid. Survey: Matheuristics for Rich Vehicle Routing Problems. In MaríaJ. Blesa, Christian Blum, Günther Raidl, Andrea Roli, and Michael Sampels, editors, *Hybrid Metaheuristics SE - 15*, volume 6373 of *Lecture Notes in Computer Science*, pages 206–221. Springer Berlin Heidelberg, 2010.

Rodolfo Dondo and Jaime Cerdá. A sweep-heuristic based formulation for the vehicle routing problem with cross-docking. *Computers & Chemical Engineering*, 48(0):293–311, January 2013.

Martin W Dorn. Fishing behavior of factory trawlers: a hierarchical model of information processing and decision-making. *{ICES} Journal of Marine Science: Journal du Conseil*, 58(1): 238–252, January 2001.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

Lanah Evers, Kristiaan Glorie, Suzanne van der Ster, Ana Isabel Barros, and Herman Monsuur. A two-stage approach to the orienteering problem with stochastic weights. *Computers & Operations Research*, 43:248–260, March 2014.

Awi Federgruen and Paul Zipkin. A combined vehicle routing and inventory allocation problem. *Operations Research*, 32(5):1019–1037, October 1984.

Miguel Andres Figliozzi. Planning approximations to the average length of vehicle routing problems with varying customer demands and routing constraints. *Transportation Research Record: Journal of the Transportation Research Board*, 2089(-1):1–8, December 2008.

Marshall L Fisher and Ramchandran Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11(2):109–124, June 1981.

C Fraley and A E Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, January 1998.

Roberto De Franceschi, Matteo Fischetti, and Paolo Toth. A new ILP-based refinement heuristic for vehicle routing problems. *Mathematical Programming*, 105(2-3):471–499, 2006.

Guojun Gan, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*, volume 20. Siam, 2007.

Hermann Gehring and Jörg Homberger. A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In *Proceedings of EUROGEN99*, volume 2, pages 57–64, 1999.

Michel Gendreau, Alain Hertz, and Gilbert Laporte. A tabu search heuristic for the vehicle routing problem. *Management science*, 40(10):1276–1290, 1994.

Michel Gendreau, François Guertin, Jean-Yves Potvin, and René Séguin. Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. *Transportation Research Part C: Emerging Technologies*, 14(3):157–174, June 2006.

Gianpaolo Ghiani, Emanuele Manni, and Barrett W Thomas. A comparison of anticipatory algorithms for the dynamic and stochastic traveling salesman problem. *Transportation Science*, 46 (3):374–387, October 2011.

Billy E Gillett and Leland R Miller. A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, 22(2):340–349, April 1974.

M Girolami. Mercer kernel-based clustering in feature space. *Neural Networks, IEEE Transactions on*, 13(3):780–784, May 2002.

Bruce Golden, S. Raghavan, and Edward Wasil, editors. *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*. Springer US, Boston, MA, 2008.

Bruce L Golden, Edward A Wasil, James P Kelly, and I-Ming Chao. The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In *Fleet management and logistics*, pages 33–56. Springer, 1998.

Justin C Goodson, Jeffrey W Ohlmann, and Barrett W Thomas. Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand. *European Journal of Operational Research*, 217(2):312–323, March 2012.

Justin C Goodson, Jeffrey W Ohlmann, and Barrett W Thomas. Rollout policies for dynamic solutions to the multivehicle routing problem with stochastic demand and duration limits. *Operations Research*, 61(1):138–154, February 2013a.

Justin C Goodson, Jeffrey W Ohlmann, and Barrett W Thomas. Rollout policies for dynamic solutions to the multivehicle routing problem with stochastic demand and duration limits. *Operations Research*, 61(1):138–154, 2013b.

Justin C Goodson, Barrett W Thomas, and Jeffrey W Ohlmann. A Generalized Rollout Algorithm Framework for Stochastic Dynamic Programming. Working Paper, 2014.

Salvatore Greco, Benedetto Matarazzo, and Roman Slowinski. Multicriteria classification by dominance-based rough set approach. *Handbook of data mining and knowledge discovery.Oxford University Press, New York*, 2002.

C Groer, B Golden, and E Wasil. The consistent vehicle routing problem. *Manufacturing & service operations management*, 11(4):630–643, 2009.

Chris Groër, Bruce Golden, and Edward Wasil. The Consistent Vehicle Routing Problem. *Manufacturing & Service Operations Management*, 11(4):630–643, October 2009.

Chris Groër, Bruce Golden, and Edward Wasil. A library of local search heuristics for the vehicle routing problem. *Mathematical Programming Computation*, 2(2):79–101, April 2010.

W J Guerrero, C Prodhon, N Velasco, and C A Amaya. A relax-and-price heuristic for the inventory-location-routing problem. *International Transactions in Operational Research*, pages n/a–n/a, April 2014.

Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: An Efficient Clustering Algorithm for Large Databases. SIGMOD '98, pages 73–84. ACM, August 1998.

Walton M Hancock and Franklin H Bayha. The learning curve. *Handbook of industrial engineering*, pages 1585–1598, 1992.

M A Haughton. The performance of route modification and demand stabilization strategies in stochastic vehicle routing. *Transportation Research Part B: Methodological*, 32(8):551–566, 1998.

M A Haughton. Quantifying the benefits of route reoptimisation under stochastic customer demands. *Journal of the Operational Research Society*, 51(3):320–332, 2000.

M A Haughton and A J Stenger. Modeling the customer service performance of fixed-routes delivery systems under stochastic demand. *Journal of Business Logistics*, 19:155–172, 1998.

Chuan He, Cong Wang, Yi-Xin Zhong, and Rui-Fan Li. A survey on learning to rank. In *2008 International Conference on Machine Learning and Cybernetics*, volume 3, pages 1734–1739. IEEE, 2008.

Daniel S Holland and Jon G Sutinen. Location choice in New England trawl fisheries: Old habits die hard. *Land Economics*, 76(1):133–149, February 2000.

Lars M. Hvattum, Arne Lø kketangen, and Gilbert Laporte. Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science*, 40 (4):421–438, November 2006.

Lars Magnus Hvattum, Arne Lø kketangen, and Gilbert Laporte. A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems. *Networks*, 49(4):330–340, July 2007.

Lars Magnus Hvattum, Arne Lø kketangen, and Gilbert Laporte. Scenario tree-based heuristics for stochastic inventory-routing problems. *INFORMS Journal on Computing*, 21(2):268–285, October 2008.

Soumia Ichoua, Michel Gendreau, and Jean-Yves Potvin. Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science*, 40(2):211–225, May 2006.

Taylan Ilhan, Seyed M. R. Iravani, and Mark S. Daskin. The orienteering problem with stochastic profits. *IIE Transactions*, 40(4):406–421, February 2008. 00018.

Arif Imran, Said Salhi, and Niaz A Wassan. A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, 197(2): 509–518, September 2009.

Jianyong Jin, Teodor Gabriel Crainic, and Arne Lø kketangen. A parallel multi-neighborhood cooperative tabu search for capacitated vehicle routing problems. *European Journal of Operational Research*, 222(3):441–451, November 2012.

G Karypis, Eui-Hong Han, and V Kumar. Chameleon: hierarchical clustering using dynamic modeling, 1999.

Jon Kleinberg, Christos Papadimitriou, and Prabhakar Raghavan. A microeconomic view of data mining. *Data Mining and Knowledge Discovery*, 2(4):311–324, December 1998.

Duncan Knowler. A review of selected bioeconomic models with environmental influences in fisheries. *Journal of Bioeconomics*, 4(2):163–181, May 2002.

Attila A Kovacs, Bruce L Golden, Richard F Hartl, and Sophie N Parragh. The generalized consistent vehicle routing problem. *Transportation Science*, June 2014.

Ohseok Kwon, Bruce Golden, and Edward Wasil. Estimating the length of the optimal TSP tour: an empirical study using regression and neural networks. *Computers & Operations Research*, 22 (10):1039–1046, 1995.

Jari Kytöjoki, Teemu Nuortio, Olli Bräysy, and Michel Gendreau. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, 34(9):2743–2757, September 2007.

Daniel E Lane. A partially observable model of decision making by fishermen. *Operations Research*, 37(2):240–254, March 1989.

G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, October 2009.

G Laporte, M Gendreau, J-Y. Potvin, and F Semet. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, 7(4-5):285–300, September 2000a.

G Laporte, M Gendreau, J-Y. Potvin, and F Semet. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, 7(4-5):285–300, September 2000b.

Gilbert Laporte. The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231–247, June 1992.

M Laszlo and S Mukherjee. Minimum spanning tree partitioning algorithm for microaggregation. *Knowledge and Data Engineering, IEEE Transactions on*, 17(7):902–911, 2005.

Hongtao Lei, Gilbert Laporte, and Bo Guo. Districting for Routing with Stochastic Customers Districting for Routing with Stochastic Customers. 2011.

J K Lenstra and A H G Rinnooy Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, June 1981.

Jinfeng Li, Kanliang Wang, and Lida Xu. Chameleon based on clustering feature tree and its application in customer segmentation. *Annals of Operations Research*, 168(1):225–245, April 2009.

X Li and S Olafsson. Discovering dispatching rules using data mining. *Journal of Scheduling*, 8(6): 515–527, 2005.

R H Lin. Potential use of FP-growth algorithm for identifying competitive suppliers in SCM. *Journal of the Operational Research Society*, 60(8):1135–1141, 2009.

Fuh-Hwa Franklin Liu and Sheng-Yuan Shen. A route-neighborhood-based metaheuristic for vehicle routing problem with time windows. *European Journal of Operational Research*, 118(3): 485–504, November 1999.

Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

Sandro Lorini, Jean-Yves Potvin, and Nicolas Zufferey. Online vehicle routing and scheduling with dynamic travel times. *Computers & Operations Research*, 38(7):1086–1090, July 2011.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA, 1967.

O L Mangasarian. *Nonlinear programming*, volume 10. Society for Industrial Mathematics, 1994.

Yannis Marinakis. Multiple phase neighborhood search-GRASP for the capacitated vehicle routing problem. *Expert Systems with Applications*, 39(8):6807–6815, June 2012.

Yannis Marinakis and Magdalene Marinaki. A hybrid genetic particle swarm optimization algorithm for the vehicle routing problem. *Expert Systems with Applications*, 37(2):1446–1455, March 2010.

S. Meisel and D. Mattfeld. Synergies of operations research and data mining. *European Journal of Operational Research*, 206(1):1–10, 2010.

S Meisel, U Suppa, and D Mattfeld. GRASP based approximate dynamic programming for dynamic routing of a vehicle. In *Proceedings of MIC*, 2009.

Jorge E Mendoza, Bruno Castanier, Christelle Guéret, Andrés L Medaglia, and Nubia Velasco. A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, 37(11):1886–1898, November 2010.

Jorge E Mendoza, Bruno Castanier, Christelle Guéret, Andrés L Medaglia, and Nubia Velasco. Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, 45(3):346–363, February 2011.

JorgeE. Mendoza and JuanG. Villegas. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. *Optimization Letters*, 7(7):1503–1516, 2013.

David Mester and Olli Bräysy. Active guided evolution strategies for large-scale vehicle routing problems with time windows. *Computers & Operations Research*, 32(6):1593–1614, June 2005.

David Mester and Olli Bräysy. Active-guided evolution strategies for large-scale capacitated vehicle routing problems. *Computers & Operations Research*, 34(10):2964–2975, October 2007.

Tom Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

Yuichi Nagata and Olli Braysy. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. *Networks*, 54(4):205–215, 2009.

George L Nemhauser and Laurence A Wolsey. *Integer and combinatorial optimization*, volume 18. Wiley New York, 1988.

Gordon F. Newell and Carlos F. Daganzo. Design of multiple-vehicle delivery tours-i a ring-radial network. *Transportation Research Part B: Methodological*, 20(5):345–363, 1986a.

Gordon F. Newell and Carlos F. Daganzo. Design of multiple vehicle delivery tours-ii other metrics. *Transportation Research Part B: Methodological*, 20(5):365–376, 1986b.

Tom Nishida and Ding-Geng Chen. Incorporating spatial autocorrelation into the general linear model with an application to the yellowfin tuna (Thunnus albacares) longline CPUE data. *Fisheries Research*, 70(23):265–274, December 2004.

Yanfeng Ouyang. Design of vehicle routing zones for large-scale distribution systems. *Transportation Research Part B: Methodological*, 41(10):1079–1093, December 2007a.

Yanfeng Ouyang. Design of vehicle routing zones for large-scale distribution systems. *Transportation Research Part B: Methodological*, 41(10):1079–1093, December 2007b.

Edzer J. Pebesma. Multivariable geostatistics in s: the gstat package. *Computers and Geosciences*, 30:683–691, 2004.

Victor Pillac, Michel Gendreau, Christelle Guéret, and Andrés L Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, February 2013.

Jean-Yves Potvin, Ying Xu, and Ilham Benyahia. Vehicle routing and scheduling with dynamic travel times. *Computers & Operations Research*, 33(4):1129–1137, April 2006.

Eric N Powell, Allison J Bonner, Bruce Muller, and Eleanor A Bochenek. Vessel time allocation in the US Illex illecebrosus fishery. *Fisheries Research*, 61(13):35–55, March 2003.

W B Powell. A stochastic formulation of the dynamic assignment problem, with an application to truckload motor carriers. *Transportation Science*, 30(3):195–219, 1996.

W B Powell, P Jaillet, and A Odoni. Stochastic and dynamic networks and routing. *Handbooks in Operations Research and Management Science*, 8:141–295, 1995.

R Prellezo, I Lazkano, M Santurtún, and A Iriondo. A qualitative and quantitative analysis of selection of fishing area by Basque trawlers. *Fisheries Research*, 97(12):24–31, April 2009.

Christian Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002, 2004.

Christian Prins, Caroline Prodhon, Angel Ruiz, Patrick Soriano, and Roberto Wolfler Calvo. Solving the capacitated location-routing problem by a cooperative lagrangean relaxation-granular tabu search heuristic. *Transportation Science*, 41(4):470–483, November 2007.

H N Psaraftis. Vehicle routing: Methods and studies. *Chapter: Dynamic Vehicle Routing Problem, Elsevier Science Publishers BV*, 1988.

H.N. Psaraftis. Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61 (1):143–164, 1995.

Andre E Punt and Ray Hilborn. Fisheries stock assessment and decision analysis: the Bayesian approach. *Reviews in Fish Biology and Fisheries*, 7(1):35–63, March 1997.

César Rego, Dorabela Gamboa, Fred Glover, and Colin Osterman. Traveling salesman problem heuristics: Leading methods, implementations and latest advances. *European Journal of Operational Research*, 211(3):427–441, June 2011.

Jacques Renaud and Fayez F Boctor. A sweep-based algorithm for the fleet size and mix vehicle routing problem. *European Journal of Operational Research*, 140(3):618–628, August 2002.

Francesc Robust, Carlos F. Daganzo, and Reginald R. Souleyrette II. Implementing vehicle routing models. *Transportation Research Part B: Methodological*, 24(4):263–286, August 1990. 00102.

Peter J Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, November 1987.

Tommaso Russo, Antonio Parisi, Marina Prorgi, Fabrizio Boccoli, Innocenzo Cignini, Maurizio Tordoni, and Stefano Cataudella. When behaviour reveals activity: Assigning fishing effort to métiers based on VMS data using artificial neural networks. *Fisheries Research*, 111(12):53–64, September 2011.

M W Savelsbergh, M Goetschalckx, and Georgia Institute of Technology. Material Handling Research Center. *A comparison of the efficiency of fixed versus variable vehicle routes*. Material Handling Research Center, Georgia Institute of Technology, 1993.

P Sawicki and J Zak. Technical diagnostic of a fleet of vehicles using rough set theory. *European Journal of Operational Research*, 193(3):891–903, 2009.

Milner B Schaefer. Some considerations of population dynamics and economics in relation to the management of the commercial marine fisheries. *Journal of the Fisheries Research Board of Canada*, 14(5):669–681, May 1957.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, July 1998.

N Secomandi. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27(11-12):1201–1225, 2000.

N Secomandi. A rollout policy for the vehicle routing problem with stochastic demands. *Operations Research*, pages 796–802, 2001.

Nicola Secomandi. Analysis of a rollout approach to sequencing problems with stochastic routing applications. *Journal of Heuristics*, 9(4):321–352, 2003.

X Y Shao, Z H Wang, P G Li, and C X J Feng. Integrating data mining and rough set for customer group-based discovery of product configuration rules. *International Journal of Production Research*, 44(14):2789–2811, 2006.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

Karen Smilowitz, Maciek Nowak, and Tingting Jiang. Workforce management in periodic delivery operations. *Transportation Science*, 47(2):214–230, 2013.

Alan D Smith. Loyalty and e-marketing issues. *Quarterly Journal of Electronic Commerce*, 3(2): 149, June 2002.

Alan D. Smith. Empirical exploration for a product data management (PDA) system at a major telecommunications firm. *Industrial Management + Data Systems*, 104(5/6):513–525, 2004.

Kenneth Sörensen and Marc Sevaux. A practical approach for robust and flexible vehicle routing using metaheuristics and monte carlo sampling. *Journal of Mathematical Modelling and Algorithms*, 8(4):387–407, 2009.

Mariam Tagmouti, Michel Gendreau, and Jean-Yves Potvin. A dynamic capacitated arc routing problem with time-dependent service costs. *Transportation Research Part C: Emerging Technologies*, 19(1):20–28, February 2011.

Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

C.D. D Tarantilis, F. Stavropoulou, and P.P. P Repoussis. A template-based tabu search algorithm for the consistent vehicle routing problem. *Expert Systems with Applications*, 39(4):4233–4239, March 2012.

Lydia C L Teh, Louise S L Teh, and Michael J Meitner. Preferred resource spaces and fisher flexibility: Implications for spatial management of small-scale fisheries. *Human Ecology*, 40(2): 213–226, April 2012.

B W Thomas. Dynamic Vehicle Routing. *Wiley Encyclopedia of Operations Research and Management Science*, 2011.

Barrett W Thomas. Waiting strategies for anticipating service requests from known customer locations. *Transportation Science*, 41(3):319–331, August 2007.

Barrett W Thomas and Emanuele Manni. Scheduled penalty variable neighborhood search. *Computers & Operations Research*, 52:170–180, 2014.

Barrett W Thomas and Chelsea C White. Anticipatory route selection. *Transportation Science*, 38 (4):473–487, November 2004.

Gregorio Tirado, Lars Magnus Hvattum, Kjetil Fagerholt, and Jean-François Cordeau. Heuristics for dynamic and stochastic routing in industrial shipping. *Computers & Operations Research*, 40 (1):253–263, January 2013.

T L Tseng, C C Huang, F Jiang, and J C Ho. Applying a hybrid data-mining approach to prediction problems: A case of preferred suppliers prediction. *International Journal of Production Research*, 44(14):2935–2954, 2006.

Peter Turchin. *Quantitative analysis of movement: measuring and modeling population redistribution in animals and plants*, volume 1. Sinauer Associates Sunderland, Massachusetts, USA, 1998.

Pascal Van Hentenryck, Russell Bent, Luc Mercier, and Yannis Vergados. Online stochastic reservation systems. *Annals of Operations Research*, 171(1):101–126, August 2009.

Pascal Van Hentenryck, Russell Bent, Eli Upfal, and Pascal Hentenryck. Online stochastic optimization under time constraints. *Annals of Operations Research*, 177(1):151–183, September 2010.

S Wagner and D Wagner. *Comparing clusterings: An overview*. Universität Karlsruhe, Fakultät für Informatik, 2007.

Wei Wang, Jiong Yang, Richard Muntz, and Others. STING: A statistical information grid approach to spatial data mining. volume 97, pages 186–195, August 1997.

Xiaochun Wang, Xiali Wang, and D M Wilkes. A divide-and-conquer approach for minimum spanning tree-based clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 21(7): 945–958, 2009.

C D J Waters. Vehicle-scheduling problems with uncertainty and omitted customers. *Journal of the Operational Research Society*, pages 1099–1108, 1989.

Henning Winker, Sven E Kerwath, and Colin G Attwood. Comparison of two approaches to standardize catch-per-unit-effort for targeting behaviour in a multispecies hand-line fishery. *Fisheries Research*, 139:118–131, March 2013.

K F Wong and J E Beasley. Vehicle routing using fixed delivery areas. *Omega*, 12(6):591–600, 1984.

Jiefeng Xu and James P. Kelly. A network flow-based tabu search heuristic for the vehicle routing problem. *Transportation Science*, 30(4):379–393, 1996.

Shu Zhang, Jeffrey Ohlmann, and Barrett Thomas. Dynamic Orienteering on a Network of Queues. *Tippie College of Business Publications*, September 2014a. 00000.

Shu Zhang, Jeffrey W. Ohlmann, and Barrett W. Thomas. A priori orienteering with time windows and stochastic wait times at customers. *European Journal of Operational Research*, 239(1):70–79, November 2014b.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, {SIGMOD} '96, pages 103–114, New York, NY, USA, 1996. ACM.

H. Zhong, R. W. Hall, and M. Dessouky. Territory Planning and Vehicle Dispatching with Driver Learning. *Transportation Science*, 41(1):74–89, February 2007.

Yingjie Zhong and Michael H Cole. A vehicle routing problem with backhauls and time windows: a guided local search solution. *Transportation Research Part E: Logistics and Transportation Review*, 41(2):131–144, March 2005.