UAV Routing and Coordination in Stochastic, Dynamic Environments

John J. Enright Emilio Frazzoli
Marco Pavone Ketan Savla

Abstract

Recent years have witnessed great advancements in the science and technology for unmanned aerial vehicles (UAVs), e.g., in terms of autonomy, sensing, and networking capabilities. This chapter surveys algorithms on task assignment and scheduling for one or multiple UAVs in a dynamic environment, in which targets arrive at random locations at random times, and remain active until one of the UAVs flies to the target's location and performs an on-site task. The objective is to minimize some measure of the targets' activity, e.g., the average amount of time during which a target remains active. The chapter focuses on a technical approach that relies upon methods from queueing theory, combinatorial optimization, and stochastic geometry. The main advantage of this approach is its ability to provide analytical estimates of the performance of the UAV system on a given problem, thus providing insight into how performance is affected by design and environmental parameters, such as the number of UAVs and the target distribution. In addition, the approach provides provable guarantees on the system's performance with respect to an ideal optimum. To illustrate this approach, a variety of scenarios are considered, ranging from the simplest case where one UAV moves along continuous paths and has unlimited sensing capabilities, to the case where the motion of the UAV is subject to curvature constraints, and finally to the case where the UAV has a finite sensor footprint. Finally, the problem of cooperative routing algorithms for multiple UAVs is considered, within the same queueing-theoretical framework, and with a focus on control decentralization.

I. Introduction

This chapter discusses current solution approaches for the design of cooperative control and task allocation strategies for networks of unmanned aerial vehicles (UAVs). The focus is on *uncertain* and *dynamically changing environments*, in which new task requests are generated in real time, and on routing algorithms with performance guarantees, as opposed to heuristic algorithms.

As a motivating example, consider the following scenario: a team of Unmanned Aerial Vehicles (UAVs) is responsible to investigate possible threats over a region of interest. As possible threats are detected, by intelligence, high-altitude or orbiting platforms, or by ground sensor networks, one of the UAVs must visit its location and investigate the cause of the alarm, in order to enable an appropriate response if necessary. Performing this task may require the UAV not only to fly to the possible threat's location, but also to spend additional time on site. The objective is, in general, to minimize the average time between the appearance of a possible threat and the time one of the UAVs completes the close-range inspection task. In a variation of this problem, which will be referred to as persistent patrolling, the UAVs must detect possible threats using limited-range on-board sensors. Other variations may include priority levels or time windows during which the inspection task must be completed.

In order to perform the required mission, the UAVs (or, more in general, mission control) need to repeatedly solve three *coupled* decision-making problems:

1) **Task allocation among the UAVs:** Which UAV shall pursue each task? What policy is used to assign tasks to UAVs?,

Senior Research Scientist, Kiva Systems jenright@kivasystems.com.
Associate Professor, Laboratory for Information and Decision Systems, Aeronautics and Astronautics Department, Massachusetts Institute of Technology frazzoli@mit.edu.
Assistant Professor, Aeronautics and Astronautics Department, Stanford University pavone@stanford.edu
Assistant Professor, Sonny Astani Department of Civil and Environmental Engineering, University of Southern California ksavla@usc.edu.

2) **Service scheduling for each UAV:** Given the list of tasks to be pursued, what is the most efficient ordering of these tasks?
3) **Loitering paths:** What should UAVs without pending assignments do?

In general, the combined problem, which one can refer to as Dynamic Vehicle Routing, falls within the class of heterogeneous, stochastic (possibly distributed) decision-making problems with uncertain information, with additional complexity stemming from the differential and algebraic constraints on the UAV motion and the local sensing of the environment. This problem is generally *intractable*, and solution approaches have been devised that look either at heuristics algorithms or at approximation algorithms with some guarantee on their performance.

The chapter is structured as follows. Section II presents an overview of current solution approaches for UAV routing in uncertain and dynamic environments. First, the differences and commonalities between static and dynamic UAV routing problems are discussed. Then, some of the main classes of algorithms used in the relevant literature are introduced, namely, heuristic algorithms (without performance guarantees), as well as online algorithms and spatial queueing theory (which provide guaranteed approximations to optimal performance). Section III considers in some detail the application of spatial queueing theory to some prototypical scenarios, involving a single UAV. The main motivation is to show how to model a specific UAV routing problem within this framework and how to solve it. Section IV considers the same scenarios, extending the theory to the case of multiple UAVs. Finally, Section V summarizes the chapter.

II. Approaches for UAV Routing in Dynamic Environments

The objective of this section is to first discuss the differences between static and dynamic environments, and then to present a broad overview of current solution approaches. Broadly speaking, there are three main approaches available in the literature to tackle such Dynamic Vehicle Routing problems. The first approach relies on heuristic algorithms. In the second approach, called "online algorithms," routing policies are designed to minimize the worst-case ratio between their performance and the performance of an optimal offline algorithm which has a priori knowledge of the entire input sequence. In the third approach, the routing problem is embedded within the framework of queueing theory, and routing policies are designed to minimize typical queueing-theoretical cost functions such as the expected time the tasks remain in the queue. Since the generation of tasks and motion of the vehicles is within an Euclidean space, one can refer to this third approach as spatial queueing theory.

A. Static and Dynamic Vehicle Routing

In the recent past, considerable efforts have been devoted to the problem of how to cooperatively assign and schedule tasks that are defined over an extended geographical area (Alighanbari and How, 2008; Arslan et al., 2007; Beard et al., 2002; Moore and Passino, 2007; Smith and Bullo, 2009). In these papers, the main focus is in developing distributed algorithms that operate with knowledge about the task locations and with limited communication between robots. However, the underlying mathematical model is *static*, in that no new tasks arrive over time, and fits within the framework of the static vehicle routing problem, whereby: (i) a team of m vehicles is required to service a set of n tasks in a 2-dimensional space; (ii) each task requires a certain amount of on-site service; (iii) the goal is to compute a set of routes that minimizes the cost of servicing the tasks; see Toth and Vigo (2001) for a thorough introduction to this problem. In general, most of the available literature on routing for robotic networks focuses on static environments and does not properly account for scenarios in which dynamic, stochastic and adversarial events take place.

The problem of planning routes through service tasks that arrive *during* a mission execution is known as the "dynamic vehicle routing problem" (abbreviated as the DVR problem in the operations research literature). There are two key differences between static and dynamic vehicle routing problems. First, planning algorithms should actually provide *policies* (in contrast to pre-planned routes) that prescribe how the routes should evolve as a function of those inputs that evolve in real time. Second, dynamic tasks

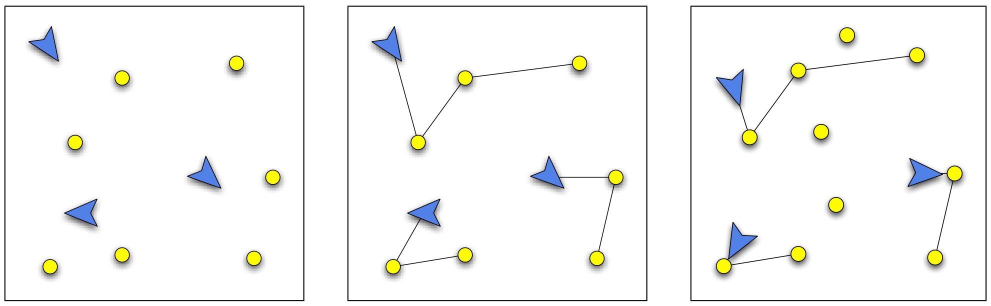


Fig. 1. An illustration of *dynamic routing problems for a robotic system*. From left to right: (i) tasks are generated, (ii) vehicles are assigned to tasks and select routes, (iii) new tasks appear, requiring an update to assignments and routes.

(i.e., tasks that arrive and vary over time) add *queueing phenomena* to the combinatorial nature of vehicle routing. In such a dynamic setting, it is natural to focus on steady-state performance instead of optimizing the performance for a single task. Additionally, system stability in terms of the number of waiting tasks is an issue to be addressed.

B. Heuristic Algorithms

A naïve, yet reasonable approach to devise an heuristic algorithm (i.e., an algorithm without performance guarantees) would be to adapt classic queueing policies to spatial queueing systems. However, perhaps surprisingly, this adaptation is not at all straightforward. For example, policies based on a First-Come First-Served discipline, whereby tasks are fulfilled in the order in which they arrive, are unable to stabilize the system for all stabilizable task arrival rates, in the sense that under such policies the average number of tasks grows over time without bound, even though there exist other policies that would maintain the number of tasks uniformly bounded (Bertsimas and van Ryzin, 1991).

The most widely applied approach is to combine static routing methods (e.g., VRP-like methods or heuristic methods such as nearest neighbor or genetic algorithms) and sequential re-optimization, where the re-optimization horizon is chosen heuristically. (A similar approach, incidentally, is at the core also of the approximation algorithms presented in the following sections.) However, the joint selection of a static routing method and of the re-optimization horizon in presence of UAV and task constraints (e.g., differential motion constraints, or task priorities) makes the application of this approach far from trivial. First, one can show that an erroneous selection of the re-optimization horizon can lead to pathological scenarios where no task *ever* receives service (Pavone, 2010). Second, direct application of VRP-like methods might lead to infeasible paths for vehicles with differential motion constraints. Additionally, performance criteria in dynamic settings commonly differ from those of the corresponding static problems. For example, in a dynamic setting, the time needed to complete a task may be a more important factor than the total vehicle travel cost.

C. Online Algorithms

An online algorithm is one that operates based on input information available up to the current time. Thus, these algorithms are designed to operate in scenarios where the entire input is not known at the outset, and new pieces of the input should be incorporated as they become available. The distinctive feature of the online algorithm approach is the method used to evaluate an algorithm's performance, which is called *competitive analysis* (Sleator and Tarjan, 1985). In competitive analysis, the performance of an online algorithm is compared to the performance of a corresponding offline algorithm (i.e., an algorithm that has *a priori* knowledge of the entire input) in the worst-case scenario. Specifically, an online algorithm is c-competitive if its cost on *any* problem instance is at most c times the cost of an optimal offline algorithm:

$$\text{Cost}_{\text{online}}(I) \leq c\,\text{Cost}_{\text{optimal offline}}(I), \quad \forall \text{ problem instances } I.$$

In the recent past, several dynamic vehicle routing problems have been successfully studied in this framework, under the name of the online traveling repairman problem (Irani et al., 2004; Jaillet and Wagner, 2006; Krumke et al., 2003), and many interesting insights have been obtained. However, the online algorithm approach has some disadvantages. First, competitive analysis is a *worst-case* analysis, hence, the results are often overly pessimistic for normal problem instances. Moreover, in many applications there is some probabilistic problem structure (e.g., distribution of the inter-arrival times, spatial distribution of future tasks, distribution of on-site service times etc.), that can be advantageously exploited by the vehicles. In online algorithms, this additional information is not taken into account. Second, competitive analysis is used to bound the performance relative to the optimal offline algorithm, and thus it does not give an absolute measure of performance. In other words, an optimal online algorithm is an algorithm with minimum "cost of causality" in the worst-case scenario, but not necessarily with the minimum worst-case cost. Finally, many important real-world constraints for DVR, such as time windows, priorities, differential constraints on vehicle's motion and the requirement of teams to fulfill a task "have so far proved to be too complex to be considered in the online framework" (Golden et al., 2008, page 206). Some of these drawbacks have been recently addressed by Van Hentenryck et al. (2009) where a combined stochastic and online approach is proposed for a general class of combinatorial optimization problems and is analyzed under some technical assumptions.

D. Spatial Queueing Theory

Spatial queueing theory embeds the dynamic vehicle routing problem within the framework of queueing theory and overcomes some of the limitations of the online algorithm approach; in particular, it allows to take into account several real-world constraints, such as time constraints and priorities. The name *spatial queueing theory* stems from the fact that the generation of the tasks and the motion of the servers (i.e., UAVs) happens in a metric space. This chapter concentrates on an algorithmic approach to spatial queueing theory whose objective is to *synthesize* an efficient control policy, whereas in standard queueing theory the objective is usually to *analyze* the performance of a specific policy. Within this context, an efficient policy is one whose *expected* performance is either optimal or within a constant factor of the optimum. Specifically, the expected performance of a policy is the expected value of the performance over all possible inputs (i.e., task arrival sequences). A policy performs within a constant factor κ of the optimum if the ratio between the policy's expected performance and the optimal expected performance is upper bounded by κ.

In order to make the model tractable, tasks are usually considered "statistically independent" and their arrival process is assumed stationary (with possibly unknown parameters). These assumptions, however, can be unrealistic in some scenarios, in which case the online algorithms approach may represent a better alternative. Pioneering work in this context is that of Bertsimas and van Ryzin (1991, 1993a,b), who introduced queueing methods to solve the simplest DVR problem (a vehicle moves along straight lines and visits tasks whose time of arrival, location and on-site service are stochastic; information about task location is communicated to the vehicle upon task arrival); see also the earlier related work (Psaraftis, 1980). Recently, by integrating ideas from dynamics, combinatorial optimization, teaming, and distributed algorithms, this approach has been applied to scenarios with complex models for the tasks such as time constraints, service priorities and translating tasks, problems concerning robotic implementation such as adaptive and decentralized algorithms, complex vehicle dynamics, limited sensing range, and team forming, and even integration of humans in the design space, see Bullo et al. (2011) and references therein.

An interesting feature of this approach is that the performance analysis of these algorithms usually yields scaling laws for quality of performance in terms of mission parameters. These scaling laws can serve as useful guidelines for operators to select mission parameters when feasible (e.g., number of UAVs, sensing range, etc.) to provide a desired quality of service.

III. Spatial Queueing Theory: the Single-Server Case

This section presents the basic ideas and tools for an algorithmic approach to spatial queuing theory (more details can be found in, e.g., Bullo et al. (2011)). This approach consists of three main steps, namely development of a spatial queueing model, establishment of fundamental limitations of performance, and design of algorithms with performance guarantees. More specifically, the formulation of a model entails detailing four main aspects:

1) A model for the *dynamic* component of the environment: this is usually achieved by assuming that new events are generated (either adversarially or stochastically) by an exogenous process.
2) A model for targets/tasks: tasks are usually modeled as points in a physical environment distributed according to some (possibly unknown) distribution, might require a certain level of on-site service time, and can be subject to a variety of constraints, e.g., time windows, priorities, etc.
3) A model for the UAVs and their motion: besides their number, one needs to specify whether the UAVs are subject to algebraic (e.g., obstacles) or differential (e.g., minimum turning radius) constraints, sensing constraints, and fuel constraints. Also, UAVs might be able to communicate directly only with other UAVs (or static nodes) that lie within a certain radius, or might not have any communication capability (e.g., when cheap micro-UAVs are used, or stealthiness is required). Finally, the control could be centralized (i.e., coordinated by a central station) or decentralized.
4) Performance criterion: examples include the minimization of the waiting time before service, loss probabilities, expectation-variance analysis, etc.

Once the model is formulated, one seeks to characterize fundamental limitations of performance (in the form of lower bounds for the best achievable cost); the purpose of this step is essentially twofold: it allows to quantify the degree of optimality of a routing algorithm and provides structural insights into the problem. As for the last step, the design of a routing algorithm usually relies on a careful combination of static routing methods with sequential re-optimization. Desirable properties for the static methods are: (i) the static problem can be solved (at least approximately) in polynomial time, and (ii) the static method is amenable to a statistical characterization (this is essential for the computation of performance bounds). Formal performance guarantees on a routing algorithm are then obtained by quantifying the ratio between an upper bound on the cost delivered by that algorithm and a lower bound for the best achievable cost. Such a ratio, being an estimate of the degree of optimality of the algorithm, should be close to one and possibly independent of systems's parameters (i.e., a *constant factor* guarantee, as defined in Section II).

In the remainder of this section, three problems will be considered, all involving only one UAV (the server in the queueing model: in the remainder of this chapter the terms "server" and "UAV," as well as "task" and "target," will be used interchangeably). In the first problem the UAV moves along continuous paths and visits targets whose time of arrival, location and on-site service are stochastic; information about target location is communicated to the UAV upon target arrival. In the second problem, the motion of the UAV is subject to differential constraints, but still the UAV has full knowledge of newly arrived targets. Finally, in the third problem, the information available to the UAV is limited, i.e., the UAV is not aware of a target's existence or location upon its arrival time, but must first detect it using on-board sensors.

The purpose of these examples is twofold: on the one hand, to provide some concrete examples about how to apply spatial queuing theory to devise UAV routing algorithms in dynamic settings, on the other hand to provide guidelines for UAV routing in a variety of scenarios of interest. The case of multi-UAV coordination will be addressed in the next section.

A. UAV routing with no motion constraints and unlimited sensing

Consider a basic scenario where one UAV moves along continuous paths (with no differential constraints, e.g., on the curvature) and visits spatially-localized targets whose time of arrival, location and on-site service are stochastic; information about target location is communicated to each UAV upon target arrival. This problem has been studied in the literature as the Dynamic Traveling Repairman Problem (DTRP) in Bertsimas and van Ryzin (1991), and can be summarized as follows:

The single-server DTRP problem: In a geographical region $\mathcal{Q}$ of area A, a dynamic process generates spatially localized tasks. The process generating tasks is modeled as a spatio-temporal Poisson process, i.e., (i) the time between each pair of consecutive events has an exponential distribution with intensity $\lambda > 0$ and (ii) upon arrival, the locations of tasks are independently and uniformly distributed in $\mathcal{Q}$. The location of the new tasks is assumed to be immediately available to the UAV. The UAV provides service in $\mathcal{Q}$, flying at constant speed v; the UAV is assumed to have unlimited fuel and task-servicing capabilities. Each task requires an independent and identically distributed amount of on-site service with finite mean duration $\bar{s} > 0$. A task is completed when the UAV moves to its location and performs its on-site service. The objective is to design a *routing policy* that maximizes the Quality of Service delivered by the UAV in terms of the average steady-state time delay $\overline{T}$ between the generation of a task and the time it is completed (in general, in a dynamic setting, the focus is on the quality of service as perceived by the "end user," rather than, for example, fuel economies achieved by the UAV). Other quantities of interests are the average number $\overline{N}$ of tasks waiting to be completed and the waiting time $\overline{W}$ of a task before its location is reached by a UAV. These quantities, however, are related according to $\overline{T} = \overline{W} + \bar{s}$ (by definition) and by Little's law, stating that $\overline{N} = \lambda\overline{W}$, for stable queues (Little, 1961).

One comment is in order: the queueing models used to model UAV routing problems are inherently different from traditional, non-spatial queuing models. In particular, one might be tempted to consider the queuing model for the single-server DTRP as a standard $M/G/1$ queue (where M stands for Poisson arrival process, G indicates that the service times are identically and *independently* distributed according to a general distribution, and 1 is the number of servers). The main reason is that in UAV routing the "service time" has both a *travel* and an *on-site* component. Although the on-site service requirements are "statistically" independent (by assumption), the travel times generally are not.

Stability: Before proceeding further, it is necessary to ensure the stability of the system. The system is considered stable if the expected number of waiting tasks is uniformly bounded at all times, or equivalently, that tasks are removed from the system at least at the same rate at which they are generated. In the case at hand, the time to complete a task is the sum of the time to reach its location (which depends on the routing policy) plus the time spent at that location in on-site service (which is independent of the routing policy). Since, by definition, the service time is no shorter than the on-site service time $\bar{s}$, then a weaker necessary condition for stability is $\varrho := \lambda\bar{s} < 1$; the quantity ϱ measures the fraction of time the UAV is performing on-site service. Remarkably, it turns out that this is also a sufficient condition for the single-server DTRP; note that this stability condition is independent of the size and shape of $\mathcal{Q}$, and of the speed of the vehicle.

Lower bounds: To derive lower bounds, the main difficulty usually consists in bounding (possibly in a statistical sense) the amount of time spent to reach a target location. The derivation of these bounds becomes simpler in asymptotic regimes, i.e., looking at cases when $\varrho \to 0^+$ and $\varrho \to 1^-$, which are often called "light load" and "heavy load" conditions, respectively.

For example, consider first the case in which $\varrho \to 0^+$ (light load regime). The median of $\mathcal{Q}$ is defined as the point that minimizes the expected distance to a random point sampled uniformly from $\mathcal{Q}$; this distance can be written as $H_1^*\sqrt{A}$, where H_1^* is a constant that only depends on the shape of $\mathcal{Q}$ (see also Appendix A). Assuming that the UAV would have enough time to return to the median location before the appearance of each new task (in other words, assuming light load conditions), the expected system time can be lower bounded as

$$\overline{T} \geq \frac{H_1^*\sqrt{A}}{v} + \bar{s} \qquad (\text{as } \varrho \to 0^+).$$

Consider now the case in which $\varrho \to 1^-$ (heavy load). Let $\overline{D}$ be the average travel distance per task for some routing policy. By using arguments from geometrical probability (independent of algorithms), one can show that $\overline{D} \geq \beta_2\sqrt{A}/\sqrt{2\overline{N}}$ as $\varrho \to 1^-$, where β_2 is a constant that will be specified later. As discussed, for stability one needs $\bar{s} + \overline{D}/v < 1/\lambda$. Combining the stability condition with the bound on the average travel distance per task, one obtains

$$\bar{s} + \frac{\beta_2 \sqrt{A}}{v\sqrt{2\overline{N}}} \leq \frac{1}{\lambda}.$$

Since, by Little's law, $\overline{N} = \lambda \overline{W}$ and $\overline{T} = \overline{W} + \bar{s}$, one finally obtains (recall that $\varrho = \lambda \bar{s}$):

$$\overline{T} \geq \frac{\beta_2^2}{2} \frac{A}{v^2} \frac{\lambda}{(1-\varrho)^2} + \bar{s}, \qquad (\text{as } \varrho \to 1^-).$$

This lower bound allows to draw the following conclusions for the single-server DTRP: (i) the condition $\varrho < 1$ is also sufficient for stability, and (ii) the Quality of Service, which is proportional to $1/(1-\varrho)^2$, degrades much faster as the target load increases than in a non-spatial queueing systems (where the growth rate is proportional to $1/(1-\varrho)$).

Routing for the single-server DTRP: Consider the following routing policy for the single-server DTRP, based on a partition of $\mathcal{Q}$ into $p \geq 1$ sub-regions $\{\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_p\}$ of equal area A/p. Such a partition can be obtained, e.g., as sectors centered at the median of $\mathcal{Q}$. Define a cyclic ordering for the sub-region, such that, e.g., if the vehicle is in region $\mathcal{Q}_i$ the "next" region is $\mathcal{Q}_j$, where j follows i in the cyclic ordering (in other words, $j = (i+1)\text{mod}p$).

1) If there are no outstanding targets, move to the median of the region $\mathcal{Q}$.
2) Otherwise, visit the "next" sub-region; subregions with no tasks are skipped. Compute a minimum-length path from the UAV's current position through all the outstanding tasks in that subregion. Complete all tasks on this path, ignoring new tasks generated in the meantime.
3) Repeat.

In the above policy, two static optimization methods are applied, depending on whether or not there are outstanding tasks. Computing the median of $\mathcal{Q}$ is a standard problem in geometric optimization (see Appendix A). The problem of computing the shortest path through a number of points is related to the well-known Traveling Salesman Problem (TSP). While the TSP is a prototypically hard combinatorial problem, it is well known that the Euclidean version of the problem is very easy to approximate; see Appendix B for more details. Furthermore, the length $\text{ETSP}(n)$ of a Euclidean TSP through n points independently and uniformly sampled in $\mathcal{Q}$ is known to satisfy the following property:

$$\lim_{n\to\infty} \text{ETSP}(n)/\sqrt{n} = \beta_2 \cdot \sqrt{A}, \qquad \text{almost surely},$$

where $\beta_2 \approx 0.712$ is a constant; this is the same constant that appears in the lower bound. The convergence to this limit is very fast: Larson and Odoni (1981) report that for "fairly compact and fairly convex" regions, the estimate $\text{ETSP}(n) \approx \beta_2\sqrt{nA}$ is within a few percent from the true value for as few as 15 points.

It can be shown (see, e.g., Bertsimas and van Ryzin (1993a)) that, using the above routing policy, the average system time $\overline{T}$ satisfies

$$\overline{T} = \frac{H_1^* \sqrt{A}}{v} + \bar{s}, \qquad (\text{as } \varrho \to 0^+),$$

$$\overline{T} \leq \gamma(p) \frac{A}{v^2} \frac{\lambda}{(1-\varrho)^2} + \bar{s}, \qquad (\text{as } \varrho \to 1^-),$$

where $\gamma(1) = \beta_2^2$, and $\gamma(p) \to \beta_2^2/2$ for large p. These results critically exploit the statistical characterization of the length of an optimal TSP tour. Hence, the proposed policy achieves a quality of service that is arbitrarily close to the optimal one, in the asymptotic regimes of light or heavy load.

While the simple policy stated above is provably optimal in light and heavy load, in the sense that no policy can provide a strictly lower system time, several variations have been proposed that improve the performance in other scenarios or operating conditions, at the expense of some additional complexity in implementation and analysis. The interested reader can find more information in Bullo et al. (2011) and references therein.

B. *UAV routing with motion constraints and unlimited sensing*

In this section, the complexity of the single-server DTRP problem is increased, by imposing differential constraints on the trajectories that the UAV can follow.

The single-server Dubins DTRP problem: In this version of the problem, the task generation process and performance metrics are assumed to be the same as in the standard DTRP problem. On the other hand, the UAV is modeled as a non-holonomic vehicle, constrained to move on the plane at constant speed v, along paths of bounded curvature. In particular, the instantaneous radius of curvature is constrained to be no less than ρ. This model is often referred to as the Dubins vehicle, in recognition of Dubins' work in computing minimum-length paths for such model (Dubins, 1957), and is typically considered appropriate to model the kinematics of UAVs (Beard et al., 2002; Chandler et al., 2000). The UAVs are assumed to be identical, and have unlimited range. In the course of this chapter, the term *Dubins frame* shall be used to refer to a coordinate frame with the origin attached to the Dubins vehicle and its first axis aligned with the vehicle's velocity vector. For simplicity, the region $\mathcal{Q}$ will be assumed to be a rectangle of height H and width W, with $H < W$, and $WH = A$. The on-site service time will be assumed to be identically zero, e.g., as in fly-by requests. This version of the DTRP will be henceforth called the Dubins DTRP.

Lower bounds: The lower bounds from the basic case still hold. Note that in this case, since $\bar{s} = 0$, the load is entirely determined by λ: in the light-load case $\lambda \to 0^+$, and in the heavy-load case $\lambda \to +\infty$. So, for example, the light-load lower bound is

$$\overline{T} \geq H_1^*/v, \qquad (\text{as } \lambda \to 0^+).$$

For the heavy-load case, it is possible to derive a lower bound specific to the Dubins DTRP. Let $\overline{D}_\rho(n)$ be the expected distance along a Dubins path, between a UAV situated in the interior of $\mathcal{Q}$ and the closest among n points independently and uniformly sampled from $\mathcal{Q}$. Reachability arguments show that, for large n,

$$\overline{D}_\rho(n) \geq \overline{\gamma}_{\mathrm{D}} \left(\frac{\rho A}{n}\right)^{1/3},$$

where $\overline{\gamma}_{\mathrm{D}} = \frac{3}{4}\sqrt[3]{3}$. In the case $\bar{s} = 0$, stability requires $\overline{D}_\rho(\overline{N})/v \leq 1/\lambda$, and Little's condition states that $\overline{N} = \lambda \overline{T}$. Hence, for large λ,

$$\frac{1}{\lambda} \geq \frac{\overline{D}_\rho(\overline{N})}{v} \geq \overline{\gamma}_{\mathrm{D}} \left(\frac{\rho A}{\lambda \overline{T}}\right)^{1/3} \frac{1}{v},$$

and finally

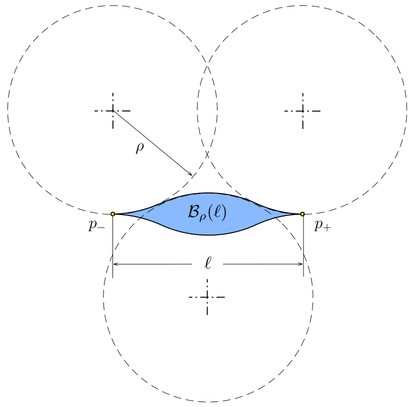
$$\overline{T} \geq \overline{\gamma}_{\mathrm{D}}^3 \frac{\rho A}{v^3} \lambda^2, \qquad (\text{as } \lambda \to +\infty).$$

Routing for the single-server Dubins DTRP problem: Consider the following routing policy for the single-server Dubins DTRP problem. As in the case of the "standard" single-server DTRP, the policy is based on a partition of $\mathcal{Q}$. However, in this case the partition is computed through a tiling of the plane into "beads" of length

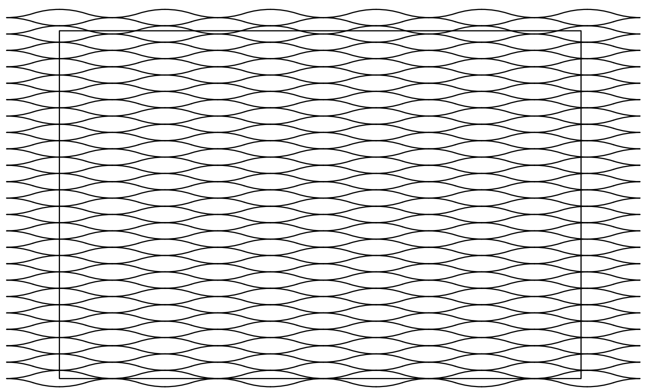
$$\ell = \min\left\{\frac{7-\sqrt{17}}{4}\left(1 + \frac{7\pi\rho H}{3A}\right)^{-1} \frac{v}{\lambda}, 4\rho\right\},$$

aligned along the width direction of $\mathcal{Q}$, see Figure 2. All beads with a non-empty intersection with $\mathcal{Q}$ are ordered in a cyclic fashion, in such a way that beads sharing a cusp are adjacent in the ordering. The beads are constructed in such a way that a Dubins vehicle arriving at point p_- (refer Figure 2) with heading towards point p_+ can service at least one target anywhere inside the bead and reach point p_+

with the same heading as it had when it arrived at point p_-. This feature allows the Dubins vehicle to service at least one target per bead in a cyclic fashion.



(a)



(b)

Fig. 2. (a) Construction of the "bead". The figure shows how the upper half of the boundary is constructed, the bottom half is symmetric. (b) Bead tiling of the plane, with the beads aligned with the width of $\mathcal{Q}$.

1) If there are no outstanding targets, loiter on a circular trajectory of radius ρ about the median of the region $\mathcal{Q}$. The direction of loitering is irrelevant, and can be chosen in such a way that the loitering circle is returned to in minimum time after servicing the target.
2) Otherwise, visit beads in order, servicing at least one target per non-empty bead. Shortcuts skipping empty beads can be taken, as long as the cyclic ordering of the beads is preserved.
3) Repeat.

In the light-load case, in which most of the time there are no outstanding tasks, it is clear that the above policy attempts to replicate the policy used in the basic case, with no differential constraints. In the heavy-load case, the construction of the beads and their cyclic ordering ensure that: (i) the number of beads in $\mathcal{Q}$ is proportional to λ^3, and hence the rate at which tasks are generated within a single bead is proportional to λ^{-2}, and (ii) the length of the cycle is proportional to λ^2, hence the rate at which the UAV is able to complete tasks within a single bead (at least once per cycle), is proportional to λ^{-2}. Indeed, the proposed policy is able to stabilize the system, and in fact provides a provable constant-factor approximation to the optimal system time.

Summarizing, it can be shown that, using the stated policy, the average system time $\overline{T}$ satisfies

$$\overline{T} \leq \frac{H_1^* \sqrt{A} + 7/3\pi\rho}{v}, \qquad (\text{as } \lambda \to 0^+),$$

and

$$\overline{T} \leq 71 \frac{\rho A}{v^3} \left(1 + \frac{7}{3}\pi \frac{\rho}{W}\right)^3 \lambda^2, \qquad (\text{as } \lambda \to +\infty).$$

In other words, the performance of the stated policy is approximately optimal in light load if the minimum turning radius ρ is negligible with respect to $\sqrt{A}$. In the heavy-load case, the policy is provably stabilizing, and is within a constant factor of the optimum.

C. UAV Routing with Limited Sensing Capabilities

In some applications, UAVs are not aware of new tasks upon their arrival; rather, the UAVs must search for and detect the tasks with limited-range on-board sensors before being able to complete them. This section is devoted to this problem, which can be called *Persistent Patrolling* problem.

The Persistent Patrolling Problem (PPP): In this version of the problem, the task generation process and performance metrics are the same as in the DTRP problem, as well as the model for the motion of the UAVs, which can fly at constant speed along any continuous path, with no differential constraints imposed. However, the UAVs are not aware of the location of new tasks until they detect them using limited-range on-board sensor. For the sake of simplicity, the sensing region of a UAV is modeled as a disk of radius σ centered at the position of the UAV. Other shapes of the sensor footprint can be considered with minor modifications to specifics of the algorithms, without affecting their performance significantly. The analysis will focus on the case in which σ is small when compared to the region $\mathcal{Q}$ (or, more precisely, to a characteristic length, e.g., $\sqrt{A}$). Also, in order to highlight some interesting aspects of the problem, the task generation process will be modeled with a non-uniform spatial distribution, described by a probability density function φ supported on $\mathcal{Q}$.

This problem is related to the well-studied problem of optimal search (Stone, 1975). However, the fact that tasks are dynamically generated changes the problem in a fundamental way. In particular, new tasks may be generated in areas that have already been explored, hence requiring the UAV to return to previously visited locations. Problems of this nature have been studied, e.g., in Song et al. (2010), considering uniform distribution of tasks; however, standard sweep methods do not work well if φ is not uniform. Motivated by search, patrolling, or foraging applications, other works such as (Cannata and Sgorbissa, 2011; Mathew and Mezic, 2009; Mesquita, 2010) give algorithms that ensure that the distribution of the UAV (over time) asymptotically matches a given distribution, often taken to be the distribution of an underlying stochastic process. However, as discussed in the following, for the case at hand the desired spatial distribution of the UAV's position is dependent on, but not equivalent to, the spatial distribution φ of tasks.

Lower Bounds: The problem will be discussed with a focus on the case in which $\sigma \to 0^+$, i.e., the sensing range is very small, and hence search time is the main factor determining the system time. It can be shown that in this case

$$\overline{T} \geq \frac{1}{4v\sigma}\left(\int_{\mathcal{Q}} \sqrt{\varphi(q)}\, dq\right)^2 + \bar{s}.$$

Since $\left(\int_{\mathcal{Q}} \sqrt{\varphi(q)}\, dq\right)^2 \leq \int_{\mathcal{Q}} \varphi(q)\, dq = 1$ (e.g., by Jensen's inequality), any non-uniformity in the task distribution is beneficial in terms of detection time, and should be exploited by the patrolling UAV.

Routing for the single-server PPP: In the case in which the density φ is uniform the optimal patrolling strategy consists of following a "lawnmower pattern," i.e., a cyclic path that allows the vehicle's footprint to cover all of Q. For small σ (i.e., ignoring boundary effects), and uniform φ, this achieves the lower bound on the system time, and is hence optimal.

The case in which φ is not uniform, the design of a good patrolling strategy is more involved. As in previous cases, a routing policy can be designed based on an appropriate partition of the environment. For simplicity of exposition, assume that the distribution φ is piecewise constant, i.e., there is a partition $\{\mathcal{Q}_1, \mathcal{Q}_2, \dots, \mathcal{Q}_l\}$ of $\mathcal{Q}$ such that $\varphi(q) = \varphi_i$ for all $q \in \mathcal{Q}_i$, $i = 1, \dots, l$. Further partition each sub-region $\mathcal{Q}_i$ into $p_i = \lceil \bar{p}/\sqrt{\varphi_i} \rceil$ tiles of equal area, $i = 1, \dots, l$, where $\bar{p}$ is a number that is large enough that the values $\bar{p}/\sqrt{\varphi_i}$ are well approximated by integers. Define a cyclic ordering of the subregions Q_i, and cyclic orderings of the p_i tiles within each subregion, $i = 1, \dots, l$. Each of these ordering defines uniquely the "next" sub-region (or tile), given the current sub-region (or tile).

1) If any outstanding tasks have been revealed, use the single-server DTRP routing policy to complete these tasks.
2) Otherwise, move to the "next" tile in the "next" subregion, and sweep this tile using a lawnmower pattern. (The initial tile and subregion can be chosen arbitrarily.)
3) Repeat.

The idea in the above policy is to ensure that the time-averaged "density" of the UAV matches $\sqrt{\varphi}$. As $\sigma \to 0^+$, the system time using this patrolling strategy matches the lower bound, and is hence optimal.

Interestingly, in the case in which σ is given, and $\varrho \to 1^-$, it turns out that the sensing-range limitation does not constrain the system's performance. In other words, in heavy-load conditions, the rate at which new tasks will be detected while servicing previously-detected targets is high enough that sensing limitations do not make a difference, and the performance for the "standard" single-server DTRP is recovered.

IV. Spatial Queueing Theory: the Multi-Server Case

The previous section presented an algorithmic approach to spatial queueing theory for single-server problems; this section extends such an approach to the multi-server case. The focus is on *control decentralization*: in fact, a decentralized architecture can provide robustness to failures of single servers, and can guarantee better time efficiency; also, it might reduce the total implementation and operation cost, increase reactivity and system reliability, and add flexibility and modularity with respect to the centralized counterpart.

The extension to the multi-server case coupled with the constraint of control decentralization add significant challenges with respect to the single-server case. Fortunately, there are a number of cases where a simple, yet systematic approach allows to lift single-server routing policies to *decentralized* multi-server routing policies with provable performance guarantees. The idea is to have the servers partition the workspace into regions of dominance via a decentralized partitioning algorithm, and then have each server follow a single-server policy within its own region. Specifically, one defines an m*-partition* of $\mathcal{Q}$ as a collection of m closed subsets $\{Q_i\}_{i=1}^m$ with disjoint interiors, and whose union is $\mathcal{Q}$. Given a single-server policy π and an m-partition of $\mathcal{Q}$, a π*-partitioning policy* is a multi-server policy such that (i) one server is assigned to each subregion (thus, there is a one-to-one correspondence between servers and subregions), and (ii) each server executes the single-server policy π to service demands that fall within its own subregion. Note that a partitioning policy is parametrized by the single-server policy π and by the m-partition of $\mathcal{Q}$, possibly computed with a decentralized partitioning algorithm. Which partitions should one consider, and to what extent this *decoupling* strategy affects optimality? These questions are discussed in detail in Pavone et al. (2009), where the authors illustrate a number of scenarios, partitioning schemes, and *decentralized* partitioning algorithms whereby one can retain optimality, or at least some degree of optimality, under this (systematic) decomposition.

In the following, paralleling the structure of the previous section, the multi-server dynamic routing problem will be studied for three problems: (1) the simplest case of teams of UAVs without motion constraints and with unlimited sensing, (2) the case of teams of UAVs with differential motion constraints and with unlimited sensing, and (3) the case of teams of UAVs without motion constraints but with limited information about the environment. In all three cases the aforementioned partitioning procedure will be pivotal for the design of provably-efficient multi-UAV routing strategies.

A. Multi-UAV routing with no motion constraints and unlimited sensing

The problem has the same definition of the DTRP problem, with the exception that m UAVs are available to provide service to the targets.

Lower bounds: The lower bounds (and the techniques to derive them) are similar to the ones for the single-server case. Consider first the light load case (i.e., $\varrho \to 0^+$). The m-median of $\mathcal{Q}$ is defined as the set of m points that minimizes the expected distance between a random point sampled uniformly from $\mathcal{Q}$ and the closest point in such set (in other words, the m-median of $\mathcal{Q}$ is the global minimizer $P_m^* := \operatorname{argmin}_{(p_1,\dots,p_m)\in\mathcal{Q}^m} \mathbb{E}\left[\min_{k\in\{1,\dots,m\}} \|p_k - q\|\right]$). This distance can be written as $H_m^*(\mathcal{Q})\sqrt{A/m}$, where $H_m^*(\mathcal{Q})$ lies in the interval $[0.3761,\, c(\mathcal{Q})]$ where $c(\mathcal{Q})$ is a constant that depends only on the shape of $\mathcal{Q}$. The m-median of $\mathcal{Q}$ induces a Voronoi partition that is called *Median Voronoi Tessellation*. Recall that the *Voronoi Diagram* $\mathcal{V}(P_m^*) = (V_1, \dots, V_m)$ of $\mathcal{Q}$ generated by points $P_m^* = (p_1, \dots, p_m)$ is defined by

$$V_i = \Big\{ q \in \mathcal{Q} |\, \|q - p_i\| \leq \|q - p_j\|, \, \forall j \neq i, \; j \in \{1, \dots, m\} \Big\}.$$

See Figure 3 for an example of a Voronoi Diagram. The expected system time in light load can then be

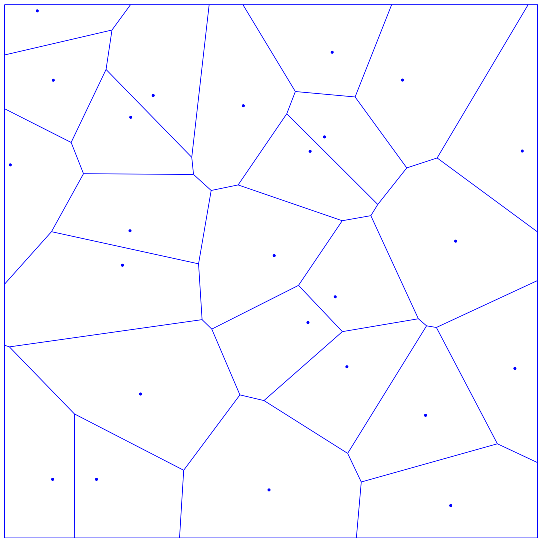


Fig. 3. A Voronoi Diagram with 25 generators.

lower bounded as

$$\overline{T} \geq \frac{H_m^*(\mathcal{Q})\sqrt{A}}{v\sqrt{m}} + \overline{s}, \qquad (\text{as } \varrho \to 0^+).$$

Hence, in light-load, the optimal system time scales with the square root of the number of UAVs. In the heavy-load case, one can show

$$\overline{T} \geq \frac{\beta_2^2}{2} \frac{A}{v^2} \frac{\lambda}{m^2 \, (1 - \varrho/m)^2}, \qquad (\text{as } \varrho/m \to 1^-).$$

A salient feature of the above lower bound is that it scales *quadratically* with the number of servers; note, however, that congestion effects are not included in this model.

Routing for the multi-server DTRP: The design of a partitioning policy relies on three basic steps, i.e.,

1) characterization of an optimal (or constant-factor optimal) single-server routing policy;
2) characterization of efficient partitioning schemes;
3) design of (possibly decentralized) algorithms for workspace partitioning.

Consider Step 2. Let π be an optimal (or constant-factor optimal) single-server policy for the DTRP. Assuming heavy-load conditions (when the problem resembles one of workload balance), one can show that a "π-partitioning policy" that uses m-partitions whose subregions have *equal area* (i.e., *equitable* m-partitions) has the same optimality properties as π. This remarkable result extends to the case where the distribution of targets is not uniform, targets have priorities, targets have time windows, end even to the case of non-holonomic UAVs, as illustrated in the next section. This result, however, only holds under the heavy-load assumption; when the load is only moderate, the shape of subregions can have a significant effect. In moderate load, a solution that turns out to be effective for the DTRP is to adopt equitable partitioning policies in which the subregions are "fat," i.e., with a small diameter for a given area, rather than long and thin. Finally, in light-load conditions, when the problem resembles one of geometric optimization, the relevant partitions are the *Median Voronoi Tessellations* defined in the previous lower bound section.

This discussion leads to the following (centralized) routing policy for the multi-server DTRP:

1) Compute an m median of $\mathcal{Q}$, and the corresponding Voronoi partition.
2) Assign one vehicle to each Voronoi region,
3) Each UAV executes the single-server DTRP policy in its own subregion.

Using the above routing policy, the average system time $\overline{T}$ satisfies

$$\overline{T} = \frac{H_m^*(\mathcal{Q})\sqrt{A}}{\sqrt{m}v} + \bar{s}, \qquad (\text{as } \varrho \to 0^+),$$

$$\overline{T} \leq \gamma \frac{A}{v^2} \frac{\lambda}{m^2 \, (1-\varrho/m)^2}, \qquad (\text{as } m \to +\infty \text{ and } \varrho/m \to 1^-).$$

where γ is the optimality factor of the single-server routing policy.

Note that the heavy-load result above relies on the fact that for large m each Voronoi region in a Median Voronoi Tesselation has the same area (see Appendix A). For general m, this may not be the case; furthermore, one might wonder whether equitable partitions can be computed in a decentralized fashion (this corresponds to the third step in the aforementioned policy design procedure).

In the solution proposed in Pavone et al. (2011), Power Diagrams are the key geometric concept to obtain, in a decentralized fashion, *equitable and median Voronoi* partitions (or "good" approximations when they do not exist). Define

$$P_W := \Big((p_1, w_1), \dots, (p_m, w_m)\Big) \in (\mathcal{Q} \times \mathbb{R})^m.$$

The pair (p_i, w_i) is called a *power point*. The *Power Diagram* $\mathcal{V}(P_W) = (V_1, \dots, V_m)$ of $\mathcal{Q}$ generated by power points P_W is defined by

$$V_i(P_W) = \Big\{ q \in \mathcal{Q} |\, \|q - p_i\|^2 - w_i \leq \|q - p_j\|^2 - w_j, \, \forall j \neq i, \; j \in \{1, \dots, m\} \Big\}.$$

The set P_W is the set of *power generators* of $\mathcal{V}(P_W)$, and V_i is the power cell of the i-th power generator. When all weights are the same, the Power Diagram coincides with the Voronoi Diagram. One can show that given $m \geq 1$ distinct points $(p_1, \dots, p_m)$ in $\mathcal{Q}$, there exist weights w_i, $i \in \{1, \dots, m\}$, such that the set of power points $\Big((p_1, w_1), \dots, (p_m, w_m)\Big)$ generates a Power Diagram that is equitable, i.e., where all power regions have the same area. The basic idea, then, is to associate to each UAV i a *virtual* power generator (virtual generator for short) (p_i, w_i); then, the power cell V_i becomes the region of dominance for UAV i (see Figure 4). A virtual generator (p_i, w_i) is simply an artificial (or logical) variable whose value is locally controlled by the ith UAV. In general, the position of an UAV and the position of its virtual generator are *distinct*, i.e., the position of an UAV inside its own region of dominance V_i is independent from the position of its virtual generator (see Figure 4). Note that an equitable power diagram can be obtained by just changing the values of the weights (while keeping the generators's positions fixed). Thus, the degrees of freedom given by the positions of the generators can be used to "steer" the partition toward an equitable and *median* Voronoi partition. Specifically, the idea is to construct an energy function with the properties that (1) it depends on the weights and positions of the virtual generators, and (2) all its critical points correspond to vectors of weights and positions yielding an equitable power diagram. Then, each UAV updates its own virtual generator by updating the weight according to a *decentralized* gradient-descent law (with respect to the energy function) and by updating the virtual generator's position so as to steer the partition toward a median Voronoi diagram, when this motion does not increase the disagreement between the areas of the neighboring regions (see Pavone et al. (2011) for details). Under this decentralized partitioning algorithm an equitable partition is always achieved, hence the resulting

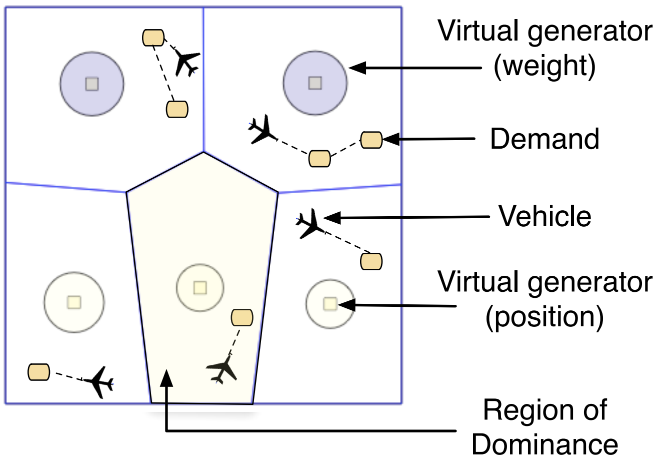


Fig. 4. Vehicles, virtual generators, tasks/demands and regions of dominance. A positive weight w is represented by a yellow circle with radius $\sqrt{w}$; a negative weight w is represented by a blue circle with radius $\sqrt{|w|}$. Note that the position of a UAV and the position of its virtual generator are, in general, distinct.

partitioning policy is optimal in heavy load. Furthermore, if an m-median of $\mathcal{Q}$ that induces a Voronoi partition that is equitable exists, this partitioning algorithm will locally converge to it, thus the resulting decentralized partitioning policy is locally optimal in light-load.

B. Multi-UAV routing with motion constraints and unlimited sensing

The definition of the problem is the same as the one for the single-server Dubins DTRP problem, but now there are m UAVs providing service. In the following, for brevity, this problem is referred to as the m-Dubins DTRP.

Lower Bounds: In contrast to the standard DTRP, the lower bound for the m-Dubins DTRP does not follow easily from the results for the single-server Dubins DTRP. Consider first the light load case. In this case, the lower bound is a function of a dimensional parameter called *nonholonomic vehicle density*:

$$d_\rho := \frac{\rho^2 m}{A},$$

which is proportional to the ratio of disk whose radius is the turning radius, and the area of $\mathcal{Q}$ available per vehicle. Hence, d_ρ is representative of the significance of the turning radius with respect to the typical distance that an individual vehicle travels. One can give the following lower bounds for the m-Dubins DTRP:

$$\overline{T} \geq \frac{H_m^*(\mathcal{Q})}{v}\left(\frac{A}{m}\right)^{1/2}, \qquad (\text{as } \lambda \to 0^+ \text{ and } d_\rho \to 0^+),$$

$$\overline{T} \geq \frac{3\sqrt[3]{3}}{4v}\left(\frac{\rho A}{m}\right)^{1/3}, \qquad (\text{as } \lambda \to 0^+ \text{ and } d_\rho \to +\infty).$$

The first bound is obtained by approximating the Dubins distance (i.e., the length of the shortest feasible path for a Dubins vehicle) with the Euclidean distance. The second lower bound is obtained by explicitly taking into account the Dubins turning cost. Note that when the motion constraint becomes "active" as $d_\rho \to +\infty$, the lower bound scales with $m^{-1/3}$.

Reachability arguments show that when λ is large (i.e., in heavy load), a lower bound for the heavy-load

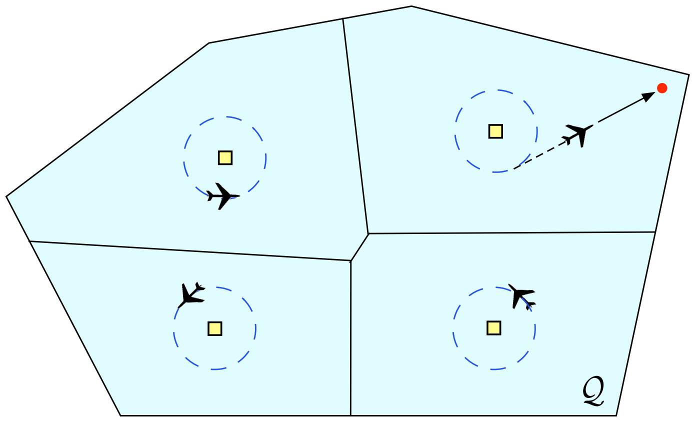


Fig. 5. Illustration of the the light-load policy for low values of non-holonomic density. The squares represent elements of $P_m^*(\mathcal{Q})$, the m-median of $\mathcal{Q}$. Each vehicle loiters about its respective generator at a radius ρ. The regions of dominance are the Voronoi partition generated by $P_m^*(\mathcal{Q})$. In this figure, a task has appeared in the subregion roughly in the upper-right quarter of the domain. The vehicle responsible for this subregion has left its loitering orbit and is en route to service the demand.

case is as follows:

$$\overline{T} \geq \frac{81}{64}\frac{\rho A}{v^3}\frac{\lambda^2}{m^3}, \qquad (\text{as } \lambda/m \to +\infty).$$

Hence, for the m-Dubins DTRP the lower bound scales with the inverse of the *cube* of the number of UAVs (in contrast to the standard DTRP where the scaling is inverse of the square); note, again, that the model does not include congestion effects.

Routing Policies for the m-Dubins DTRP: Consider, first, the case when λ is small (i.e., light load) and the non-holonomic density is low; in this case the problem resembles the standard multi-server DTRP problem. Accordingly, the relevant partition scheme is a Median Voronoi Tesselation, and the following partitioning policy is efficient in this case.

1) Compute a m-median of $\mathcal{Q}$,
2) Assign one UAV to each Voronoi region,
3) Each UAV visits the demands in the Voronoi region in the order in which they arrive. When no demands are available, the UAV returns to the median and loiters on a circular trajectory of radius ρ centered at the median of the subregion.

This policy is illustrated in Figure 5. The performance of this policy is given by

$$\overline{T} \leq \frac{H_m^*(\mathcal{Q})}{v}\left(\frac{A}{m}\right)^{1/2}, \qquad (\text{as } \lambda \to 0^+ \text{ and } d_\rho \to 0^+),$$

i.e., this policy is optimal. Hence, in this asymptotic regime the system time scales as $1/(v\sqrt{m})$.

In light-load, when the non-holonomic density is large, one should instead consider *dynamic* partitions; in particular, an effective policy is as follows:

1) Bound the environment $\mathcal{Q}$ with a rectangle of minimum *height*, where height denotes the smaller of the two side lengths of a rectangle. Let W and H be the *width* and *height* of this bounding rectangle, respectively. Divide $\mathcal{Q}$ into strips of width w, where

$$w = \min\left\{\left(\frac{4}{3\sqrt{\rho}}\frac{WH + 10.38\rho H}{m}\right)^{2/3}, 2\rho\right\}.$$

2) Orient the strips along the side of length W.
3) Construct a closed Dubins path which runs along the longitudinal bisector of each strip, visiting all strips in top-to-bottom sequence, making U-turns between strips at the edges of $\mathcal{Q}$, and finally returning to the initial configuration. The m UAVs loiter on this path, equally spaced, in terms of path length.

A depiction of this policy is shown in Figure 6. At the instant a target arrives, one constructs a circle of radius ρ which is tangent to the loitering path and intersects the target. The UAV responsible for visiting the target is the one closest in terms of loitering path length to the point of departure, at the time of target-arrival. After a UAV has serviced a target, it must return to its place in the loitering pattern.

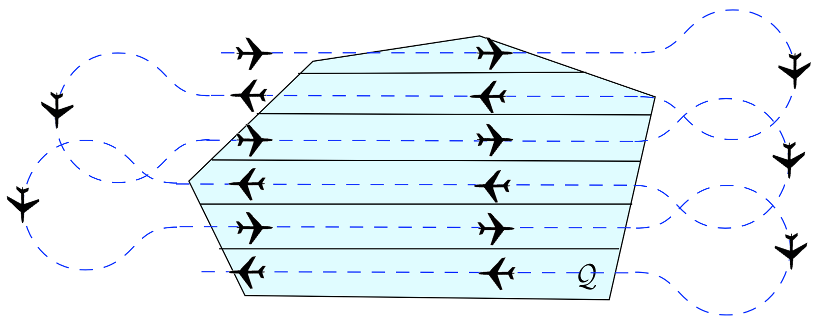


Fig. 6. Illustration of the the light-load policy for large non-holonomic densities. The segment providing closure of the loitering path (returning the UAVs from the end of the last strip to the beginning of the first strip) is not shown here for clarity of the drawing.

Note that the dynamic partition associated with a particular UAV is in fact fixed in the reference frame of that UAV and that in the global frame these partitions could be regarded as a *dynamic version of an equitable partition* of $\mathcal{Q}$, modulo the boundary effects. The reason for using dynamic equitable partitions is as follows. For large values of non-holonomic density, the area per vehicle is small in comparison to the disk of radius ρ, and hence the efficient regions of responsibility for the UAVs in such a regime coincide with their *small-time* reachable sets. Since the UAVs modeled as a Dubins vehicle can not stall, a simple way to ensure that the regions of responsibility are contained inside the small-time reachable sets all the time is to have the regions of responsibility fixed in the reference frame of the UAVs.

The system time of the above policy that uses dynamic partitions satisfies

$$\overline{T} \leq \begin{cases} \frac{1.238}{v}\left(\frac{\rho WH+10.38\rho^2 H}{m}\right)^{1/3} + \frac{W+H+6.19\rho}{mv} & \text{for } m \geq 0.471\left(\frac{WH}{\rho^2}+\frac{10.38H}{\rho}\right), \\ \frac{WH+10.38\rho H}{4\rho m v} + \frac{W+H+6.19\rho}{mv} + \frac{1.06\rho}{v} & \text{otherwise.} \end{cases}$$

Hence this policy is within a constant factor of the optimal in the asymptotic regime where $\lambda/m \to 0^+$ *and* $d_\rho \to +\infty$. Moreover, in such asymptotic regime the optimal system time scales as $1/(v\sqrt[3]{m})$.

Finally, for the heavy load, the relevant partitions are *additively-weighted* equitable partitions, as follows:

1) Divide the environment into regions of dominance with lines parallel to the bead rows. Let the area and height of the ith vehicle's region be denoted with A_i and H_i. Place the subregion dividers in such a way that

$$A_i + \frac{7}{3}\pi\rho H_i = \frac{1}{m}\left(A + \frac{7}{3}\pi\rho H\right), \qquad \text{for all } i \in \{1,\dots,m\}.$$

2) Allocate one subregion to every vehicle.

3) Each vehicle executes the single vehicle policy in its own region, where the size of bead in region i is chosen to be $\ell_i = \min\{C_{\mathrm{BT},i} v/\lambda_i, 4\rho\}$, with $\lambda_i = \lambda A_i/A$ being the arrival rate in region i, and $C_{\mathrm{BT},i} = \frac{7-\sqrt{17}}{4}\left(1 + \frac{7\pi\rho H_i}{3A_i}\right)^{-1}$.

The system time for this policy satisfies

$$\overline{T} \leq 71 \frac{\rho A}{v^3}\left(1 + \frac{7\pi\rho H}{3A}\right)^3 \frac{\lambda^2}{m^3}, \qquad (\text{as } \lambda/m \to +\infty).$$

Hence this policy is within a constant factor of the optimal in heavy-load, and that the optimal system time in this case scales as $\lambda^2/(mv)^3$.

Note that all the three above policies are partitioning policies; the partitions used are, however, quite different, ranging from Median Voronoi Tesselations, to dynamic partitions, to additively-weighted equitable partitions. Median Voronoi Tesselations and additively-weighted equitable partitions can be computed in a *decentralized* fashion with the partitioning algorithm discussed for the multi-DTRP problem (indeed, a Median Voronoi Tesselation can be computed even without inter-agent communication). No decentralized algorithms have been developed so far for dynamic partitions.

It is instructive to compare the scaling of the optimal system time with respect to λ, m and v for the m-DTRP and for the m-Dubins DTRP. Such comparison is shown in Table I. One can observe that

TABLE I
A COMPARISON BETWEEN THE SCALING OF THE OPTIMAL SYSTEM TIME FOR THE MULTI-SERVER DTRP AND FOR THE m-DUBINS DTRP.

	$\overline{T}^*$, m-DTRP	$\overline{T}^*$, m-Dubins DTRP
Heavy load ($\lambda/m \to +\infty$)	$\Theta\left(\frac{\lambda}{m^2 v^2}\right)$	$\Theta\left(\frac{\lambda^2}{m^3 v^3}\right)$
Light load ($\lambda/m \to 0^+$)	$\Theta\left(\frac{1}{v\sqrt{m}}\right)$	$\Theta\left(\frac{1}{v\sqrt{m}}\right)$ if $d_\rho \to 0^+$ $\Theta\left(\frac{1}{v\sqrt[3]{m}}\right)$ if $d_\rho \to +\infty$

in heavy-load the optimal system time for the m-Dubins DTRP is of the order $\lambda^2/(mv)^3$, whereas for the m-DTRP it is of the order $\lambda/(mv)^2$. This analysis rigorously establishes the following intuitive fact: bounded-curvature constraints make the optimal system much more sensitive to increases in the demand generation rate. Perhaps less intuitive is the fact that the optimal system time is also more sensitive with respect to the number of vehicles and the vehicle speed in the m-Dubins DTRP as compared to the m-DTRP. In the light load, the optimal system time for the m-DTRP is of the order $1/(v\sqrt{m})$, which is the same for the m-DTRP but only when the non-holonomic density is small. When the non-holonomic density is high, the optimal system time is of the order $1/(v\sqrt[3]{m})$, i.e., it is more sensitive to the number of vehicles than in the low non-holonomic density case. This suggests the existence of a critical value of d_ρ below which the partitioning policy using Median Voronoi partitions is efficient, and above which the partitioning policy that uses dynamic partitions is efficient. The details of such phase transitions can be found in Enright et al. (2009).

C. Multi-UAV Routing with Limited Sensing Capabilities

The definition of the problem is the same as in the single-server Persistent Patrolling Problem, with the only difference being the number of UAVs providing service.

Lower Bounds: As in the single-server case, the discussion will focus on the case in which $\sigma \to 0^+$. In this case the system time is bounded by

$$\overline{T} \geq \frac{1}{4mv\sigma}\left(\int_{\mathcal{Q}} \sqrt{\varphi(q)}\, dq\right)^2 + \bar{s}.$$

Routing policies for the m-server PPP: Optimal static partitioning policies can be designed using partitions that are simultaneously equitable with respect to φ and to $\sqrt{\varphi}$. (Such partitions can always be found.) However, a simpler strategy is based on a "dynamic" partition, as follows:

1) If any outstanding tasks have been revealed, use the multi-server DTRP routing policy to complete these tasks.
2) Otherwise, move to the "next" tile in the "next" subregion, and sweep this tile using a lawnmower pattern. (The initial tile and subregion for each vehicle are chosen in such a way that vehicles are uniformly spaced along the sub-region/tile cycle.)
3) Repeat.

The above policy matches the lower bound, and is hence optimal as $\sigma \to 0^+$. In the case in which σ is finite, and $\varrho \to 1^-$, the sensing-range limitation does not constrain the system's performance, and the performance for the "standard" multi-server DTRP is recovered. As discussed above, no decentralized algorithms have been developed so far for dynamic partitions.

V. Conclusions

In this chapter, we presented a dynamic vehicle routing framework for coordination of UAVs performing spatially distributed tasks in uncertain and dynamic environments. The technical approach relies on using algorithmic spatial queueing theory for designing efficient algorithms for single vehicles, and then augmenting it with spatial partitioning policies to extend it to multiple vehicles. To illustrate this approach, a variety of problems were considered, ranging from the simplest case with no motion or sensing constraints, to the cases with motion differential constraints and with limited sensing constraints. Additionally, multi-UAV versions for each of these cases were considered. For each scenario, UAV routing policies have been provided, which are provably optimal or approximately optimal in the asymptotic regimes of light and heavy load. The dependence of the performance of these algorithms on mission parameters such as the number, speed, sensor footprint and turning radius of the UAVs, and the dimension of the workspace was discussed. A system designer could interpolate between such scaling laws for the extreme cases as established in this chapter, to choose a policy that best fits given problem specifications. Moreover, the algorithms and their underlying principles presented in this chapter for canonical scenarios could also provide guidelines to design algorithms for other scenarios not explicitly considered in this chapter.

References

P. K. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Computing Surveys*, 30(4):412–458, 1998.

M. Alighanbari and J. P. How. A robust approach to the UAV task assignment problem. *International Journal on Robust and Nonlinear Control*, 18(2):118–134, 2008.

D. Applegate, R. Bixby, V. Chvátal, and W. Cook. On the solution of traveling salesman problems. In *Documenta Mathematica, Journal der Deutschen Mathematiker-Vereinigung*, pages 645–656, Berlin, Germany, 1998. Proceedings of the International Congress of Mathematicians, Extra Volume ICM III.

S. Arora. Nearly linear time approximation scheme for Euclidean TSP and other geometric problems. In *Proc. 38th IEEE Annual Symposium on Foundations of Computer Science*, pages 554–563, 1997.

G. Arslan, J. R. Marden, and J. S. Shamma. Autonomous vehicle-target assignment: A game theoretic formulation. *ASME Journal on Dynamic Systems, Measurement, and Control*, 129(5):584–596, 2007.

R. W. Beard, T. W. McLain, M. A. Goodrich, and E. P. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Trans. on Robotics and Automation*, 18(6):911–922, 2002.

J. Beardwood, J. Halton, and J. Hammersley. The shortest path through many points. *Proceedings of the Cambridge Philoshopy Society (Mathematical and Physical Sciences)*, 55(4):299–327, 1959.

D. J. Bertsimas and G. J. van Ryzin. A stochastic and dynamic vehicle routing problem in the Euclidean plane. *Operations Research*, 39:601–615, 1991.

D. J. Bertsimas and G. J. van Ryzin. Stochastic and dynamic vehicle routing in the Euclidean plane with multiple capacitated vehicles. *Operations Research*, 41(1):60–76, 1993a.

D. J. Bertsimas and G. J. van Ryzin. Stochastic and dynamic vehicle routing with general interarrival and service time distributions. *Advances in Applied Probability*, 25:947–978, 1993b.

F. Bullo, E. Frazzoli, M. Pavone, K. Savla, and S.L. Smith. Dynamic vehicle routing for robotic systems. *Proceedings of the IEEE*, 99(9):1482 –1504, 2011.

G. Cannata and A. Sgorbissa. A minimalist algorithm for multirobot continuous coverage. *IEEE Trans. on Robotics*, 27(2), 2011.

P. Chandler, S. Rasmussen, and M. Pachter. UAV cooperative path planning. In *AIAA Conf. on Guidance, Navigation, and Control*, 2000.

N. Christofides. Bounds for the travelling-salesman problem. *Operations Research*, 20:1044–1056, 1972.

Z. Drezner, editor. *Facility Location: A Survey of Applications and Methods*. Series in Operations Research. Springer, 1995. ISBN 0-387-94545-8.

L.E. Dubins. On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79:497–516, 1957.

J. J. Enright, K. Savla, E. Frazzoli, and F. Bullo. Stochastic and dynamic routing problems for multiple UAVs. *AIAA J. of Guidance, Control, and Dynamics*, 32(4):1152–1166, 2009.

B. Golden, S. Raghavan, and E. Wasil. *The Vehicle Routing Problem: Latest Advances and New Challenges*, volume 43 of *Operations Research/Computer Science Interfaces*. Springer, 2008. ISBN 0387777776.

S. Irani, X. Lu, and A. Regan. On-line algorithms for the dynamic traveling repair problem. *Journal of Scheduling*, 7(3): 243–258, 2004.

P. Jaillet and M. R. Wagner. Online routing problems: Value of advanced information and improved competitive ratios. *Transportation Science*, 40(2):200–210, 2006.

D. S. Johnson, L. A. McGeoch, and E. E. Rothberg. Asymptotic experimental analysis for the held-karp traveling salesman bound. In *Proc. 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 341–350, 1996.

S. O. Krumke, W. E. de Paepe, D. Poensgen, and L. Stougie. News from the online traveling repairman. *Theoretical Computer Science*, 295(1-3):279–294, 2003.

R. Larson and A. Odoni. *Urban Operations Research*. Prentice Hall, Englewood Cliffs, NJ, 1981.

S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21: 498–516, 1973.

J. D. C. Little. A Proof for the Queuing Formula: L= λW. *Operations Research*, 9(3):pp. 383–387, 1961. ISSN 0030364X. URL http://www.jstor.org/stable/167570.

G. Mathew and I. Mezic. Spectral multiscale coverage: A uniform coverage algorithm for mobile sensor networks. In *Proceedings of the 48th IEEE Control and Decision Conference*, pages 7872–7877, Shanghai, China, December 2009.

N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM Journal on Computing*, 13(1):182–196, 1984. ISSN 0097-5397.

A.R. Mesquita. *Exploiting Stochasticity in Multi-agent Systems*. PhD thesis, University of California at Santa Barbara, Santa Barbara, CA, 2010.

B. J. Moore and K. M. Passino. Distributed task assignment for mobile agents. *IEEE Transactions on Automatic Control*, 52 (4):749–753, 2007.

C. H. Papadimitriou. Worst-case and probabilistic analysis of a geometric location problem. *SIAM Journal on Computing*, 10 (3), August 1981.

M. Pavone. *Dynamic Vehicle Routing for Robotic Networks*. PhD thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, June 2010.

M. Pavone, K. Savla, and E. Frazzoli. Sharing the load. *IEEE Robotics and Automation Magazine*, 16(2):52–61, 2009.

M. Pavone, A. Arsie, E. Frazzoli, and F. Bullo. Distributed algorithms for environment partitioning in mobile robotic networks. *IEEE Trans. on Automatic Control*, 56(8):1834–1848, 2011.

G. Percus and O. C. Martin. Finite size and dimensional dependence of the Euclidean traveling salesman problem. *Physical Review Letters*, 76(8):1188–1191, 1996.

H. N. Psaraftis. Dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14(2):130–154, 1980.

D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2): 202–208, 1985.

S. L. Smith and F. Bullo. Monotonic target assignment for robotic networks. *IEEE Trans. on Automatic Control*, 54(9): 2042–2057, 2009.

D. Song, C. Y. Kim, and J. Yi. Stochastic modeling of the expected time to search for an intermittent signal source under a limited sensing range. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

J. M. Steele. Probabilistic and worst case analyses of classical problems of combinatorial optimization in Euclidean space. *Mathematics of Operations Research*, 15(4):749, 1990.

L.D. Stone. *Theory of Optimal Search*. Academic Press, New York, NY, 1975.

P. Toth and D. Vigo, editors. *The Vehicle Routing Problem*. Monographs on Discrete Mathematics and Applications. SIAM, 2001. ISBN 0898715792.

P. Van Hentenryck, R. Bent, and E. Upfal. Online stochastic optimization under time constraints. *Annals of Operations Research*, 177(1):151–183, 2009.

Eitan Zemel. Probabilistic analysis of geometric location problems. *Annals of Operations Research*, 1(3), October 1984.

Appendix

A. *The continuous multi-median problem*

Given a set $\mathcal{Q} \subset \mathbb{R}^d$ and a vector $P = (p_1, \dots, p_m)$ of m distinct points in $\mathcal{Q}$, the expected distance between a random point q, generated according to a probability density function φ, and the closest point in P is given by

$$\begin{aligned} H_m(P, \mathcal{Q}) &:= \mathrm{E}\left[\min_{i \in \{1,\dots,m\}} \|p_i - q\|\right] \\ &= \sum_{i=1}^{m} \int_{\mathcal{V}_i(P,\mathcal{Q})} \|p_i - q\| \varphi(q) dq, \end{aligned}$$

where $\mathcal{V}(P, \mathcal{Q}) = (\mathcal{V}_1(P, \mathcal{Q}), \dots, \mathcal{V}_m(P, \mathcal{Q})$ is the Voronoi partition of the set $\mathcal{Q}$ generated by the points P. In other words, $q \in \mathcal{V}_i(P, \mathcal{Q})$ if $\|q - p_i\| \leq \|q - p_k\|$, for all $k \in \{1, \dots, m\}$. The set $\mathcal{V}_i$ is referred to as the Voronoi cell of the generator p_i. The function H_m is known in the locational optimization literature as the continuous Weber function or the continuous multi-median function; see (Agarwal and Sharir, 1998; Drezner, 1995) and references therein.

The m-median of the set $\mathcal{Q}$, with respect to the measure induced by φ, is the global minimizer

$$P_m^*(\mathcal{Q}) = \operatorname*{argmin}_{P \in \mathcal{Q}^m} H_m(P, \mathcal{Q}).$$

Let $H_m^*(\mathcal{Q}) = H_m(P_m^*(\mathcal{Q}), \mathcal{Q})$ be the global minimum of H_m. It is straightforward to show that the map $P \mapsto H_1(P, \mathcal{Q})$ is differentiable and strictly convex on $\mathcal{Q}$. Therefore, it is a simple computational task to compute $P_1^*(\mathcal{Q})$. It is convenient to refer to $P_1^*(\mathcal{Q})$ as the median of $\mathcal{Q}$. On the other hand, the map $P \mapsto H_m(P, \mathcal{Q})$ is differentiable (whenever $(p_1, \dots, p_m)$ are distinct) but not convex, thus making the solution of the continuous m-median problem hard in the general case. It is known (Agarwal and Sharir, 1998; Megiddo and Supowit, 1984) that the discrete version of the m-median problem is NP-hard for $d \geq 2$. Gradient algorithms for the continuous m-median problems can be designed by means of the equality

$$\frac{\partial H_m(P, \mathcal{Q})}{\partial p_i} = \int_{\mathcal{V}_i(P,\mathcal{Q})} \frac{p_i - q}{\|p_i - q\|} \varphi(q) dq.$$

The set of critical points of H_m contains all configurations $(p_1, \dots, p_m)$ with the property that each p_i is the generator of the Voronoi cell $\mathcal{V}_i(P, \mathcal{Q})$ as well as the median of $\mathcal{V}_i(P, \mathcal{Q})$. We refer to such Voronoi diagrams as *median Voronoi diagrams*. It is possible to show that a median Voronoi diagram always exists for any bounded convex domain $\mathcal{Q}$ and density φ.

The dependence of $H_m^*(\mathcal{Q})$ on m plays a crucial role in the design and analysis of algorithms relying on geometric optimization. However, finding the exact relationship for the general case is difficult; hence, it is of great interest to provide bounds on $H_m^*(\mathcal{Q})$. This problem is studied thoroughly in Papadimitriou (1981) for square regions and in Zemel (1984) for more general compact regions. It is known that, in the asymptotic case ($m \to +\infty$), $H_m^*(\mathcal{Q}) = c_{\text{hex}}\sqrt{A/m}$ almost surely, where $c_{\text{hex}} \approx 0.377$ is the first moment of a hexagon of unit area about its center. This optimal asymptotic value is achieved by placing the m points at the centers of the hexagons in a regular hexagonal lattice within $\mathcal{Q}$ (the honeycomb heuristic). Working towards the above result, it is also known that for any $m \in \mathbb{N}$:

$$\frac{2}{3}\sqrt{\frac{A}{\pi m}} \leq H_m^*(\mathcal{Q}) \leq c(\mathcal{Q})\sqrt{\frac{A}{m}},$$

where $c(\mathcal{Q})$ is a constant depending on the shape of $\mathcal{Q}$.

B. The Euclidean Traveling Salesman Problem

The Euclidean TSP is formulated as follows: given a finite set D of n points in $\mathbb{R}^d$, find the minimum-length closed curve through all points in D. In graph theoretical language, a tour of the point set D is a spanning cycle of the complete graph with vertex set P; the length of a tour is the sum of all Euclidean distances between points in the tour.

The asymptotic behavior of stochastic TSP problems for large n exhibits the following interesting property. Let $\mathrm{ETSP}(n)$ be a random variable returning the length of the Euclidean TSP tour through n points, independently and uniformly sampled from a compact set $\mathcal{Q}$ of unit area; in Beardwood et al. (1959) it is shown that there exists a constant β_2 such that, almost surely,

$$\lim_{n\to+\infty} \frac{\mathrm{ETSP}(n)}{\sqrt{n}} = \beta_2.$$

In other words, the optimal cost of stochastic TSP tours approaches a deterministic limit, and grows as the square root of the number of points to be visited; the current best estimate of the constant appearing in the limit is $\beta_2 = 0.7120 \pm 0.0002$, see Johnson et al. (1996); Percus and Martin (1996). Similar results hold in higher dimensions, and for non-uniform point distributions: from Steele (1990), the limit (B) takes the general form

$$\lim_{n\to+\infty} \frac{\mathrm{ETSP}(n)}{n^{1-1/d}} = \beta_d \int_{\mathcal{Q}} \bar{\varphi}(q)^{1-1/d}\, dq \qquad \text{almost surely,}$$

where $\bar{\varphi}$ is the density of the absolutely continuous part of the distribution φ from which the n points are independently sampled. Notice that the bound holds for all compact sets: the shape of the set only affects the convergence rate to the limit. According to Larson and Odoni (1981), if $\mathcal{Q}$ is a "fairly compact and fairly convex" set in the plane, the estimate $\mathrm{ETSP}(n) \approx \beta_2\sqrt{n}$ for values of n as low as 15. Remarkably, the asymptotic cost of the stochastic TSP for uniform point distributions is an upper bound on the asymptotic cost for general point distributions, i.e.,

$$\lim_{n\to+\infty} \frac{\mathrm{ETSP}(n)}{n^{1-1/d}} \leq \beta_d.$$

This follows directly from an application of Jensen's inequality, i.e.,

$$\int_{\mathcal{Q}} \bar{\varphi}(q)^{1-\frac{1}{d}}\, dq \leq \left(\int_{\mathcal{Q}} \bar{\varphi}(q)\, dq\right)^{1-\frac{1}{d}} \leq \varphi(\mathcal{Q})^{1-\frac{1}{d}} = 1.$$

The TSP is known to be NP-hard, which suggests that there is no general algorithm capable of finding the optimum tour in an amount of time polynomial in the size of the input. Even though the exact optimal solutions of a large TSP can be very hard to compute, several exact and heuristic algorithms and software tools are available for the numerical solution of Euclidean TSPs.

The most advanced TSP solver to date is arguably `concorde` (Applegate et al., 1998). Heuristic polynomial-time algorithms are available for constant-factor approximations of TSP solutions, such as Christofides' algorithm, providing a 3/2 approximation factor (Christofides, 1972). On a more theoretical side, Arora (1997) proved the existence of polynomial-time approximation schemes, providing a $(1+\varepsilon)$ constant-factor approximation for any $\varepsilon > 0$.

A modified version of the Lin-Kernighan heuristic (Lin and Kernighan, 1973) is implemented in `linkern`; this powerful solver yields approximations in the order of 5% of the optimal tour cost very quickly for many instances. For example, in numerical experiments on a 2.4 GHz Pentium machine, approximations of random TSPs with 1,000 points typically required about two seconds of CPU time. Both `concorde` and `linkern` are written in ANSI C and, at the time of writing, are freely available for academic research use at `http://www.tsp.gatech.edu/concorde/index.html`.

In this chapter, several routing policies were presented requiring on-line solutions of large TSPs. Practical implementations of the algorithms will rely on heuristics, such as Lin-Kernighan's or Christofides'. If a constant-factor approximation algorithm is used, the effect on the asymptotic performance guarantees of our algorithms can be simply modeled as a scaling of the constant β_d.