Design and Analysis of Optimal Task-Processing Agents

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Theodore P. Pavlic, B.S., M.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2010

Dissertation Committee:

Kevin M. Passino, Advisor

Andrea Serrani

Atilla Eryilmaz

Abstract

This dissertation is given in two parts followed by concluding remarks. The first three chapters describe the generalization of optimal foraging theory for the design of solitary task-processing agents. The following two chapters address the coordinated action of distributed independent agents to achieve a desirable global result. The short concluding part summarizes contributions and future research directions.

Optimal foraging theory (OFT) uses ecological models of energy intake to predict behaviors favored by natural selection. Using models of the long-term rate of energetic gain of a solitary forager encountering a variety of food opportunities at a regular rate, it predicts characteristics of optimal solutions that should be expressed in nature. Several engineered agents can be modeled similarly. For example, an autonomous air vehicle (AAV) that flies over a region encounters targets randomly just as an animal will encounter food as it travels. OFT describes the preferences that the animal is likely to have due to natural selection. Thus, OFT applied to mobile vehicles describes the preferences of successful vehicle designs.

Although OFT has had success in existing engineering applications, rate maximization is not a good fit for many applications that are otherwise analogous to foraging. Thus, in the first part of this dissertation, the classical OFT methods are rediscovered for generic optimization objectives. It is shown that algorithms that are computationally equivalent to those inspired by classical OFT can perform better in

realistic scenarios because they are based on more feasible optimization objectives. It is then shown how the design of foraging-like algorithms provides new insight into behaviors observed and expected in animals. The generalization of the classical methods extracts fundamental properties that may have been overlooked in the biological case. Consequently, observed behaviors that have been previously been called irrational are shown to follow from the extension of the classical methods.

The second part of the dissertation describes individual agent behaviors that collectively result in the achievement of a global optimum when the distributed agents operate in parallel. In the first chapter, collections of agents that are each similar to the agents from the early chapters are considered. These agents have overlapping capabilities, and so one agent can share the task processing burden of another. For example, an AAV patrolling one area can request the help of other vehicles patrolling other areas that have a sparser distribution of targets. We present a method of volunteering to answer the request of neighboring agents such that sensitivity to the relative loading across the network emerges. In particular, agents that are relatively more loaded answer fewer task-processing requests and receive more answers to their own requests. The second chapter describes a distributed numerical optimization method for optimization under inseparable constraints. Inseparable constraints typically require some direct coordination between distributed solver agents. However, we show how certain implementations allow for stigmergy, and so far less coordination is needed among the agents. For example, intelligent lighting, which maintains illumination constraints while minimizing power usage, is one application where the distributed algorithm can be applied directly.

This dissertation is written in memory of my brother Kenny, who I miss dearly. I know no one else who was loved by so many, and I know no one else who deserved his fate less.

Acknowledgments

First, I give thanks to my parents, Paul and Eileen, and my partner Jessie; they have always been supportive and understanding, even when research has reduced the frequency of contact with them. Any success that I have today could not have been possible without them.

My adviser, Professor Kevin M. Passino, deserves thanks not only for his wisdom and guidance but also for his unending patience with me. Through him, I have not only learned engineering, but I have become a better writer and overall thinker. He has strengthened my understanding of how to research effectively and continues to serve as an important role model for me.

Any accurate understanding that I have of behavioral ecology is entirely due to Professor Thomas A. Waite. The tangible and intangible benefits of collaboration with him are too numerous to list. I am an interloper in his field, and he has not only tolerated my intrusion but has welcomed me and provided me with instructions on how I might proceed deeper into new spaces. Exposure to him and his colleagues has left me in awe of the ecological adventures that are common in his field. One colleague of his who also deserves special attention is Ian M. Hamilton; I have been successful contributing to literature in behavioral ecology not in small part due to his guidance and suggestions.

Finally, I thank the other members of my dissertation committee, including Andrea Serrani and Atilla Eryilmaz, for taking the time to evaluate my work. Over the years, I have valued Professor Serrani's persepective on the personal and professional complications of doing academic research in engineering.

Vita

February 28, 1981	Born - Columbus, OH, USA
June 2004	B.S., Elec. & Comp. Engineering
June 2007	M.S., Elec. & Comp. Engineering
2004–present	Dean's Distinguished Univ. Fellow, The Ohio State University
2006–2007	NSF GK-12 Fellow, The Ohio State University
2002, 2003	Analog Design Intern, National Instruments, Austin, Texas
2001	Core Systems Developer, IBM Storage, RTP, North Carolina

Publications

T.P. Pavlic and K.M. Passino. When rate maximization is impulsive. *Behavioral Ecology and Sociobiology*, 64(8):1255–1265. August 2010. DOI:10.1007/s00265-010-0940-1

T.P. Pavlic and K.M. Passino. The sunk-cost effect as an optimal rate-maximizing behavior. *Acta Biotheoretica*. 2010. In press. DOI:10.1007/s10441-010-9107-8

T.P. Pavlic and K.M. Passino. Foraging theory for autonomous vehicle speed choice. *Engineering Applications of Artificial Intelligence*, 22(3):482–489, April 2009. DOI:10.1016/j.engappai.2008.10.017

R.J. Freuler, M.J. Hoffmann, T.P. Pavlic, J.M. Beams, J.P. Radigan, P.K. Dutta, J.T. Demel, and E.D. Justen. Experiences with a comprehensive freshman hands-on

course – designing, building, and testing small autonomous robots. In *Proceedings of the 2003 American Society for Engineering Education Annual Conference & Exposition*, 2003.

Fields of Study

Major Field: Electrical and Computer Engineering

Table of Contents

List of Tables

Table **Page**

List of Figures

Part I: Behaviors of a Solitary Optimal Task-Processing Agent

Foraging theory has been the inspiration for several decision-making algorithms for task-processing agents facing random environments. As nature selects for foraging behaviors that maximize lifetime calorie gain or minimize starvation probability, engineering designs are favored that maximize returned value (e.g., profit) or minimize the probability of not reaching performance targets. In this part of the dissertation, we investigate how foraging theory can be expanded so that it can be used in a wider range of design applications. Then we show how applying foraging theory to a general space of solitary-agent optimization problems has revealed new insights into the biological world.

Prior foraging-inspired designs are direct applications of classical optimal foraging theory (OFT). Here, in Chapter 1, we describe a generalized optimization framework that encompasses the classical OFT model, a popular competitor, and several new models introduced here that are better suited for some task-processing applications in engineering. These new models merge features of rate maximization, efficiency maximization, and risk-sensitive foraging while not sacrificing the intuitive character of classical OFT. However, the central contributions of this chapter are analytical and graphical methods for designing decision-making algorithms guaranteed to be optimal within the framework. Thus, we provide a general modeling framework for solitary agent behavior, several new and classic examples that apply to it, and generic methods (some of which are described in more detail in Section 1.3.3) for design and analysis of optimal task-processing behaviors that fit within the framework. Our results extend the key mathematical features of optimal foraging theory to a wide range of other optimization objectives in biological, anthropological, and technological contexts. This work was originally presented by Pavlic and Passino [84].

Although optimal foraging theory predicts that natural selection should favor animal behaviors that maximize long-term rate of gain, behaviors observed in the laboratory tend to be impulsive. In binary-choice experiments, despite the long-term gain of each alternative, animals favor short handling times. Most explanations of this behavior suggest that there is hidden rationality in impulsiveness. Instead, in Chapter 2, we suggest that simultaneous and mutually exclusive binary-choice encounters are often unnatural and thus immune to the effects of natural selection. Using a simulation of an imperfect forager, we show how a simple strategy (i.e. an intuitive model of animal behavior) that maximizes long-term rate of gain under natural conditions appears to be impulsive under operant laboratory conditions. We then show how the accuracy of this model can be verified in the laboratory by biasing subjects with a short pre-experiment ad libitum high-quality feeding period. We also show a similar behavioral mechanism results in diet preferences that are qualitatively consistent with the digestive rate model of foraging (i.e., foraging under digestive rate constraints). This work was originally presented by Pavlic and Passino [82]; the germ of the work was a simple decision-making heuristic for real-time control of a solitary task-processing agent.

Optimal foraging theory has been criticized for underestimating patch exploitation time. However, as we show in Chapter 3, proper modeling of costs not only answers these criticisms, but it also explains apparently irrational behaviors like the sunk-cost effect. When a forager is sure to experience high initial costs repeatedly, the forager should devote more time to exploitation than searching in order to minimize the accumulation of said costs. Thus, increased recognition or reconnaissance costs lead to increased exploitation times in order to reduce the frequency of future costs,

and this result can be used to explain paradoxical human preference for higher costs. In fact, this result also provides an explanation for how continuing a very costly task indefinitely provides the optimal long-term rate of gain; the entry cost of each new task is so great that the forager avoids ever returning to search. In general, apparently irrational decisions may be optimal when considering the lifetime of a forager within a larger system. This work was originally presented by Pavlic and Passino [83]. Expanding the parameter space to include negative costs was necessary to encompass general objective functions seen in engineering applications.

Chapter 1: Generalizing Foraging Theory for Analysis and Design

Foraging theory has been a source of inspiration for optimization [78, 79], autonomous vehicle control [7, 9, 81, 97], and distributed resource allocation [8, 31, 32, 95]. In each case, automated agents prosecute tasks that are analogous to food encountered by animals in the environment. Just like food, tasks can be scarce, are encountered randomly, carry a random handling time, and carry a random value that is analogous to calorie content. Additionally, when an agent chooses to prosecute a task, it may face increased risk of harm during the handling of the task (e.g., from fatigue or from forces analogous to predation). Just as natural selection will favor animal behaviors that maximize lifetime calorie content or minimize the probability of starvation, engineering design favors decision-making algorithms that maximize accumulated value or minimize the probability of not reaching performance targets. Thus, by translating from biological currencies (e.g., calories) to engineering currencies (e.g., dollars), foraging behaviors shown to be advantageous in nature become optimal algorithms for engineered agents in random environments.

Unfortunately, although an autonomous agent may be easily viewed as a forager, the objectives favored by natural selection are not necessarily good models for optimization in engineering. For example, an eagle in flight may select from prey it

encounters so that it maximizes calories over its lifetime. However, an autonomous air vehicle (AAV) with a finite number of packages to deposit on targets has a much shorter time horizon and thus will prioritize its targets differently. Nevertheless, the simplicity of the intuitive results from optimal foraging theory (OFT) makes it attractive for the design of autonomous decision-making algorithms. In this chapter, we identify the key structures responsible for that simplicity so that optimization objectives that better fit engineering scenarios can lead to similar foraging-like designs. Thus, this chapter extends the work of Andrews et al. [9] who applied the principle results of classical optimal foraging theory directly to AAV cases.

In particular, we describe a generalized framework for the analysis and design of optimal autonomous behaviors of solitary task-processing agents. We also give algorithms for designing behaviors within this framework that are guaranteed to meet sufficiency conditions for optimality. Our framework encompasses two popular models of optimal foraging, which include the prey and patch models that inspired existing solitary agent designs [7, 9, 97]. Four additional models that also fit within the framework are introduced to handle cases that are unfit for classical foraging analysis but are applicable for engineered agent design. Thus, the framework and the generalized optimality algorithms allow for the rapid development of optimal behaviors in new solitary agent contexts (e.g., more applicable for engineering design than science). However, they also provide methods for comparing behaviors that are optimal under different utility functions. For example, we show that when finite-lifetime success thresholds are introduced into optimization objectives, the resulting behaviors have the same form of classical OFT but prioritize targets in an order that varies with the size of the success threshold.

The chapter is structured as follows. In Section 1.1, we introduce the Markov renewal–reward process that characterizes a generic solitary task-processing agent and define the advantage-to-disadvantage function, which is an abstract optimization objective that encapsulates several aspects of existing foraging theory. We also describe the models used in classical OFT and show that their objectives have the structure of an advantage-to-disadvantage function. Additionally, we provide motivating examples from the literature of existing applications of foraging theory to engineering. In Section 1.1.3, we define four new optimization objectives that have an advantage-to-disadvantage structure. Each of these new objectives models a special finite-lifetime task-processing agent with an intake threshold for success (e.g., a military autonomous air vehicle doing automated target processing with a finite arsenal that must reach an accumulated target value by the time its arsenal is depleted). Two of these finite-event models are inspired by classical rate maximization [23, 24, 112], and two are inspired by efficiency maximization. These finite-lifetime objectives may better fit behaviors for autonomous agents that have short missions than the classical OFT that has inspired existing decision-making algorithms. In Section 1.2, a graphical approach to multivariate optimization of advantage-to-disadvantage functions is discussed, and a more rigorous quantitative approach is explored in Section 1.3 (i.e., algorithms are given in Section 1.3.3 that are guaranteed to find an optimal task-processing behavior for particular scenarios). A summarized comparison of optimal behaviors found by those algorithms for each of the six example advantage-to-disadvantage functions is given in Section 1.4. Additionally, simulation results are given that show how behaviors developed with the methods in this chapter have better performance in finite-lifetime scenarios when compared to conventional foraging-inspired task-choice

behaviors. Finally, some concluding remarks and suggestions for future research are given in Section 1.5.

1.1 Model of an Autonomous Task-processing Agent

In this section, we present a model of a task-processing agent and show how it generalizes several foraging-inspired optimization problems from robotics and computer science. The summary of the bio-inspired engineering applications is given in Section 1.1.1, and the related optimization problem from classical foraging theory is presented in Section 1.1.2. Then, in Section 1.1.3, we present four new optimization objectives that are better fit to model desirable behaviors for task-processing agents with finite lifetimes. As these objectives each fit within the generalized framework, they can be solved with the generalized methods described in Section 1.2 and 1.3. Moreover, conversion from a classical OFT-inspired decision-making implementation involves little more than a change of parameters. This conversion process is emphasized in Section 1.4, which compares the results of applying the analytical methods in Section 1.3 to each example optimization objectives described here.

Consider an autonomous agent that can complete $n \in \{1, 2, \dots\}$ types of tasks. For task type $i \in \{1, 2, \dots, n\}$, the agent processes $p_i \in [0, 1]$ fraction of encountered type-i tasks and spends an average of $\tau_i \geq 0$ time processing each selected type-i task. So task-processing behavior is completely characterized by vectors $\vec{p} \triangleq [p_1, p_2, \dots, p_n]^\top$ and $\vec{\tau} \triangleq [\tau_1, \tau_2, \dots, \tau_n]^\top$. Next, let $\mathbb{R}$ be the set of the real numbers, $\mathbb{R}_{\geq 0}$ be the set of nonnegative real numbers, and $\overline{\mathbb{R}}_{\geq 0} \triangleq \mathbb{R}_{\geq 0} \cup \{\infty\}$. For each type $i \in \{1, 2, \dots, n\}$, constraints on feasible behaviors are modeled with constants

$p_i^-, p_i^+ \in [0,1]$ and $\tau_i^-, \tau_i^+ \in \overline{\mathbb{R}}_{\geq 0}$ so that the *feasible set* of behaviors is

$$\mathcal{F} \triangleq \{(\vec{p}, \vec{\tau}) \in [0,1]^n \times \mathbb{R}^n_{\geq 0} : p_i^- \leq p_i \leq p_i^+, \tau_i^- \leq \tau_i \leq \tau_i^+, i \in \{1, 2, \ldots, n\}\}, \quad (1.1)$$

which is a convex separable polyhedron. The optimal behavior will maximize the generic *advantage-to-disadvantage function* [80]

$$J(\vec{p}, \vec{\tau}) \triangleq \frac{A(\vec{p}, \vec{\tau})}{D(\vec{p}, \vec{\tau})} \triangleq \frac{a + \sum_{i=1}^{n} p_i a_i(\tau_i)}{d + \sum_{i=1}^{n} p_i d_i(\tau_i)} \quad (1.2)$$

where the $a \in \mathbb{R}$ and $d \in \mathbb{R}$ are constants and $a_i : [\tau_i^-, \tau_i^+] \mapsto \mathbb{R}$ and $d_i : [\tau_i^-, \tau_i^+] \mapsto \mathbb{R}$ are functions of time τ_i associated with type $i \in \{1, 2, \ldots, n\}$.

1.1.1 Background: Foraging-inspired Task-processing Agents

OFT was popularized by Stephens and Krebs [112]. It is based on the work of Charnov [23], and recently updated results and new applications are summarized by Stephens et al. [115]. OFT assumes that a solitary forager goes through Markov renewal cycles of searching for and responding to foraging opportunities. At every encounter, the forager's energy stores will rise or fall based on the forager's behavior, the environment, and the encountered item. In particular, each prey type $i \in \{1, 2, \ldots, n\}$ is encountered with rate λ_i, and those encounters that are chosen for processing have an average gain $g_i(\tau_i)$ and average cost $c_i(\tau_i)$. During the search time between encounters, the forager pays cost c^s/λ where $\lambda \triangleq \lambda_1 + \lambda_2 + \cdots + \lambda_n$ (i.e., $1/\lambda$ is the average time between encounters, and c^s is the cost paid per unit time searching). If the forager is viewed as an autonomous task-processing agent, then the prey it encounters are the tasks it must choose whether and how long to process. Stephens and Krebs [112] describe two popular special cases of the general problem:

(i) *The prey model.* In this case, it is assumed that tasks (i.e., prey) come in lumps that have fixed processing times (i.e., processing-time bounds are such that $\tau_i^- = \tau_i^+ > 0$ for each type i). The agent (i.e., forager) must only select whether to process or ignore the task.

(ii) *The patch model.* In this case, it is assumed that the agent processes every encountered task (i.e., preference bounds $p_i^- = p_i^+ = 1$ for each type i), but each encountered task is a clumped patch of prey with decreasing marginal returns (e.g., due to depletion of prey within the patch). Hence, the agent must decide how long to process each task.

As described in the selection of examples below, these ecological models of a solitary forager have been used to inspire optimal designs of autonomous mobile vehicles [7, 9, 81], resource allocation strategies for distributed temperature regulation [97], and web sites that attract attention of humans on the Internet [87–89]. In this work, we show how the forager is a special case of a more general task-processing framework. The solutions we provide for this framework apply to a wider set of applications than the original foraging and foraging-inspired cases. Moreover, this generalized framework can be used as a tool to compare the operation and efficacy of different policies.

Autonomous mobile vehicles

Andrews et al. [9] show how both the prey and patch models described by Stephens and Krebs [112] can be used to model an AAV (e.g., for military or surveillance applications). In particular, they consider a Dubins' car [28] model of an air vehicle (e.g., a fixed-wing vehicle that travels at a constant speed and has a maximum turn radius). As it sweeps over the ground, an on-board sensor detects relatively slow

targets below the vehicle. The agent responds to each target detection either with ignorance or by choosing to complete a task for a certain amount of time. Some tasks have a fixed processing time (e.g., dropping bombs or food), and other tasks can be processed continuously by the agent (e.g., reconnaissance). Processing each task is costly to the agent (e.g., due to additional fuel use), but completing task returns a value to the agent's designer (e.g., dollars of profit or some currency encoding priority).

Just as prey can be grouped into types based on returned net energy gain and handling time, these tasks can be grouped into n types based on net value $g_i(\tau_i) - c_i(\tau_i)$ and processing time τ_i for each type $i \in \{1, 2, \ldots, n\}$. Furthermore, Andrews et al. [9] use results from Stone [116] to show that if a vehicle encounters a cluster (i.e., patch) of high-value targets that it may process continuously, the accumulated value $g_i(\tau_i) - c_i(\tau_i)$ of processing the targets in patch type $i \in \{1, 2, \ldots, n\}$ over time τ_i is the area under a decaying exponential (i.e., the density of targets in the patch decays due to the depletion of remaining tasks after processing). Thus, patches of tasks have diminishing marginal returns just like patches of prey in foraging models. So descriptions of optimal animal foraging behavior are also recipes for optimal vehicle task-type (i.e., prey model) and processing-length (i.e., patch model) policies. Andrews et al. use flying-vehicle simulations to verify that policies generated by both the prey model and the patch model perform well in stochastic environments; however, the analogy can be applied to autonomous underwater, outer-space, or ground vehicles as well. For example, a domestic autonomous ground vehicle that can collect trash, clean floors, and organize furniture faces random tasks in its environment that

it must choose whether to process or momentarily ignore while searching for a more valuable task.

On-line implementation of OFT-inspired behaviors: In both the prey model application described by Andrews et al. [9] as well as the temperature regulation example below, the encounter rates with each task type must be estimated before the prey model algorithm is used at each encounter to determine whether tasks should be processed or ignored. When encounter rates are available, the prey model algorithm can be completed in linear time that scales with the number of task types. Additionally, the ratio of the number of encounters with a type to the total time will asymptotically converge to the true encounter rate in the environment, and so a simple method exists for estimating the encounter rate. Although this on-line implementation of the prey model is relatively simple to implement, Pavlic and Passino [82] present a much simpler decision-making heuristic that converges to prey-model-optimal behavior without the need for encounter rate estimation. In particular, they show that an asymptotically optimal forager needs only to compare its present accumulated-gain–total-time ratio to the g_i/τ_i ratio of each encountered task to determine whether the task should be processed or ignored. This heuristic is the natural extension of the conventional patch model implementation to the prey model case. Thus, on-line implementations of OFT-inspired decision-making are suitable for autonomous agents with strict timing requirements and simple computational abilities.

Resource allocation: distributed temperature regulation

Quijano et al. [97] develop a method for applying the prey model to distributed resource allocation, and they test their strategies in a working physical temperature control experiment. Their apparatus consists of eight zones that each include a temperature sensor and a heating element. The zones are arranged so that there is significant cross coupling (i.e., heat from one zone causes the temperature to rise not only at its local sensor but also on the sensors of nearby zones). This apparatus could be a model of a large room with multiple temperature actuators or a building with multiple rooms. Assuming that at most one heating element can be energized at a time, Quijano et al. design a policy for a centralized controller that determines which if any heating element should be activated at each time so that all zones achieve a single desired temperature.

This temperature regulation problem connects to foraging theory by using a "foraging for error" method like the one described by Passino, Passino [78, 79]. At each instant of time, there is an error associated with each zone representing the difference between the desired temperature and the temperature at its sensor. Quijano et al. [97] create an error index that maps all errors to a finite set of integers; that is, they generate a mapping $i(e)$ from error magnitude $e \in \mathbb{R}$ to error type $i \in \{1, n\}$. For each error type i, they also associate a value g_i and a heating time τ_i that both are monotonically increasing with error magnitude (i.e., a higher error magnitude is associated with a higher value and a higher heating time). The centralized controller randomly chooses which zone to monitor at each time. Hence, it encounters each error type just as a forager encounters prey types. At each encounter with error e, it identifies the error type $i(e)$ and the associated value $g_{i(e)}$ and heating time $\tau_{i(e)}$ and uses the prey model to determine whether to activate the zone for the $\tau_{i(e)}$ heating

time or to move to the next zone. Quijano et al. actually implement four such error foragers simultaneously and show that the resulting strategy achieves uniform temperature regulation across all zones and rejects temperature disturbances even under delays and sensor noise.

Similar foraging-inspired resource-allocation algorithms could be used on mobile agents deployed on factory floors that must balance queues of raw materials. If a raw material is loaded into a physical queue from one end only, the queue will frequently be overloaded on that end. A mobile robot that must move around the queue to shift resources from one location to another could prioritize its movements based on the height of each location in the queue compared to the average height. Those areas with the greatest off-average error would be highest value and thus would attract the greatest attention from the re-allocation agent.

Web design

Pirolli [88] gives a summary of so-called "information foraging" analyses of human behavior on the Internet that are based on classical optimal foraging theory. In one example, humans are viewed as foragers that accumulate information from websites that are viewed as patches of information, and it is assumed that humans will allocate time in each web patch according to optimal foraging theory. Hence, web developers must organize content on their web pages in order to maximize the time an optimal information forager should spend using their sites. For example, one of the key results of the patch model of optimal foraging theory is that foragers will spend less time in all patches if the average time between patch encounters decreases. In particular, the forager leaves each patch when the patch marginal returns fall below a particular threshold, and that threshold increases as the search time between patches decreases.

Likewise, if fast search engines return several relevant responses to a search query, the information-foraging human will spend very little time visiting each site before moving to the next site in the search results. Consequently, web sites designed to retain visitors for as long as possible (e.g., to maximize exposure to advertisements) must dynamically arrange content based on the search request so that the site sustains a high level of marginal returns of relevant information.

1.1.2 Classical Optimal Foraging Objective

In Section 1.1.1, we described several examples of how OFT has been used in technological design of autonomous vehicles, resource allocation algorithms, and dynamic web sites. Here, we summarize the classical OFT optimization objective and show how it is a special case of the advantage-to-disadvantage function. We also show how a related but different optimization objective favored by some behavioral ecologists is also an advantage-to-disadvantage function. Later, in Section 1.1.3, we present other optimization objectives that are better suited for engineering applications (e.g., AAV delivery schedules when there are a finite number of packages to deliver to a random set of targets).

OFT studies behaviors that maximize Darwinian fitness, which is an unmeasurable quantity in general. Charnov [23] and Pyke et al. [92] suggest that the lifetime rate of total gain to total time is a sufficient fitness surrogate because it predicts behaviors that achieve maximal foraging gain for minimal foraging time, which are the two objectives from the classic optimization model of natural selection [103]. Unfortunately, for any finite lifetime, this optimization objective strongly depends on precise knowledge of how gain and time covary [23, 80]. So lifetimes are assumed to

be very long (i.e., practically infinite with respect to prey handling and search times) so that the sensitivity of the optimization objective to the covariances is vanishingly small.

In particular, Charnov [23] assumes that encounters with each type come from an independent Poisson counting process. So the process describing all encounters is the *merged* Poisson process, and the energetic intake is modeled by a Markov renewal–reward process corresponding to this merged process. Over a long time, to maximize both cycle gain and number of cycles, the optimal foraging behavior $(\vec{p}, \vec{\tau}) \in \mathcal{F}$ should maximize the stochastic limit of total gain to total time [80]. That is, the behavior should maximize the *rate*

$$\frac{\left(\sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - c_i(\tau_i)\right)\right) - c^s}{1 + \sum_{i=1}^{n} \lambda_i p_i \tau_i}, \tag{1.3}$$

which matches Equation (1.2) with

$$a \triangleq -c^s, \quad a_i(\tau_i) \triangleq \lambda_i \left(g_i(\tau_i) - c_i(\tau_i)\right), \quad d \triangleq 1, \quad \text{and} \quad d_i(\tau_i) \triangleq \lambda_i \tau_i. \tag{1.4}$$

The prey model lets $\tau_i^- \triangleq \tau_i^+$ for each task type $i \in \{1, 2, \ldots, n\}$ and finds the optimal $\vec{p} \in [0, 1]^n$, and the patch model lets $p_i^- \triangleq p_i^+ \triangleq 1$ for each patch type $i \in \{1, 2, \ldots, n\}$ and finds the optimal $\vec{\tau} \in [0, \infty)^n$ [112].

The expectation of ratios: Some observational evidence [e.g., 69] contradicts predictions from the *marginal value theorem (MVT)*, which is the principle result of the patch model [23, 24, 111, 112]. In response, arguments from Templeton and Lawlor [117] have been used as fodder for *expectation-of-ratios* [15, 16, 44] objective

functions of the form

$$\sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i \frac{g_i(\tau_i) - c_i(\tau_i) - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \tau_i} \tag{1.5}$$

which matches Equation (1.2) with

$$a \triangleq 0, \quad a_i(\tau_i) \triangleq \frac{\lambda_i}{\lambda} \frac{g_i(\tau_i) - c_i(\tau_i) - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \tau_i}, \quad d \triangleq 1, \quad \text{and} \quad d_i(\tau_i) \triangleq 0.$$

These two optimization objectives are significantly different, but because they are advantage-to-disadvantage functions, they can both be analyzed with the generic methods presented in this work.

1.1.3 New Objectives for Finite-event Scenario

The success of classical OFT to describe animal foraging behavior is not uniform across species and environments. Likewise, some applications will be ill suited for solutions inspired by OFT. Below, in Section 1.1.3, we focus on criticisms of the OFT formulation for cases where task-processing agents cannot be assumed to have unending operation. Then, in Section 1.1.3, we introduce a novel optimization model of an autonomous task-processing agent that may better fit applications that are less suitable for OFT.

OFT inadequacies in finite-lifetime models

Classical foraging theory is not well suited for modeling finite lifetimes where either success thresholds must be met or only a finite number of tasks can be processed. For example, a small bird may perish from the heat lost during the night if it does not eat enough during the day. Likewise, an AAV dispatched for a finite periods of time (e.g., due to daily fuel constraints) may fail each mission if it ignores too many tasks with a low marginal return (e.g., by avoiding low-profit-per-time tasks in favor of waiting for

high-profit-per-time tasks, it may return too little overall profit in its finite mission time to justify its overall fuel cost). In the infinite lifetime case, future opportunities are certain, and so waiting can be a beneficial tactic. However, in the finite-lifetime case, future opportunities are uncertain, and so successful foragers should be biased toward present returns.

For cases with survival thresholds over short times, Stephens and Charnov [111] describe a *risk-sensitive* forager that maximizes the probability that a net gain threshold will be achieved by some critical time. This risk-sensitive foraging model is also used by Andrews et al. [9] for an AAV application where the vehicle is given a value threshold it must reach by the end of its mission time. Initially, the AAV specializes on targets that have a high value-to-time ratio. However, at the end of its life, if it has not accumulated enough value to reach its goal threshold, it begins to generalize on all targets it encounters. Hence, the risk-sensitive behavior is a perturbation of the rate-maximizing behavior that becomes most pronounced at the end of life (i.e., at the end of an agent's mission). However, the risk-sensitive model not only uses limiting forms of the mean and variance of the accumulated gain, but it is also based on results that follow from the central-limit theorem. Hence, even though the formulation is meant to prescribe behaviors for short-lifetime agents, it is based on assumptions that are only true for agents with long lifetimes.

As discussed by Wajnberg [122], OFT can be used to describe the behavior of an insect that searches for hosts to lay her eggs in. However, it is best suited to model this scenario when typical lifetimes are too short to deplete the egg supply. However, several studies have shown that egg-limited parasitoids are not uncommon [33, 45, 64, 90, 100, 101]. Furthermore, in AAV applications where packages (e.g., bombs or

food bundles) are dropped on targets, the mission will likely be limited by the number of packages able to be stored within the AAV. In Section 1.1.3, we develop a simple task-processing model that fits within the advantage-to-disadvantage framework and accounts for both success thresholds and limitations on number of tasks processed.

Autonomous agent model for finite-event scenario

Consider a task-processing agent similar to the one described in Section 1.1.1. That is, consider an agent that encounters n types of tasks where a task of type $i \in \{1, 2, \dots, n\}$ is characterized by its Poisson encounter rate λ_i, processing preference p_i, average processing time τ_i, average gain $g(\tau_i)$, and average cost $c(\tau_i)$. That agent pays an average search cost c^s/λ between encounters, where $\lambda \triangleq \lambda_1 + \cdots + \lambda_n$ is the encounter rate of the Poisson process resulting from merging the n independent encounter processes for each task type. However, also let $N \in \{1, 2, \dots\}$ be the number of processed encounters in a mission duration. For example, a forager may need to eat or store N items to survive over winter, or a female may have N eggs to lay in encountered hosts, or an AAV must deliver one of N packages to each deserving target. In each case, the time to complete each mission is finite and random, but the number of tasks completed in each mission is fixed at N.

Instead of considering the Markov renewal process that renews at each encounter at a rate of $\lambda_1 + \cdots + \lambda_n$, it is convenient to focus on the Markov renewal process that renews at every processed encounter at the lower rate of $p_1\lambda_1 + p_2\lambda_2 + \cdots + p_n\lambda_n$. The agent mission can be represented by either process, but many cycles of the former process may complete during a single cycle of the latter process. Hence, for this finite-event agent, the expectation of total net gain $G(N)$, cost $C(N)$, and time $T(N)$

are given by

$$\mathrm{E}(G(N)) = N \frac{\left(\sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - c_i(\tau_i)\right)\right) - c^s}{\sum_{i=1}^{n} \lambda_i p_i}, \tag{1.6}$$

$$\mathrm{E}(C(N)) = N \frac{\left(\sum_{i=1}^{n} \lambda_i p_i c_i(\tau_i)\right) + c^s}{\sum_{i=1}^{n} \lambda_i p_i}, \tag{1.7}$$

and

$$\mathrm{E}(T(N)) = N \frac{1 + \sum_{i=1}^{n} \lambda_i p_i \tau_i}{\sum_{i=1}^{n} \lambda_i p_i}. \tag{1.8}$$

These statistics can then be combined to form optimization objectives suitable for different applications. In particular, the finite-event agent can maximize:

(i) *Excess rate.* Because mission durations are finite by definition, success thresholds can be added. Let $G^T \in \mathbb{R}$ be a gain penalty charged to the agent after its N processed encounters (e.g., an autonomous vehicle must accumulate G^T dollars of profit from the first N tasks it randomly encounters and picks for processing). That is, G^T is the value *threshold* the agent must reach to be dispatched on another mission. This threshold will often be positive, but it may be negative (e.g., it may be a handicap allowed to the agent). In this case, optimal behaviors

maximize the ratio of *excess* net gain to total time, which is the advantage-to-disadvantage function

$$\frac{\mathrm{E}(G(N)) - G^T}{\mathrm{E}(T(N))} = \frac{\left(\sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - c_i(\tau_i) - \frac{G^T}{N}\right)\right) - c^s}{1 + \sum_{i=1}^{n} \lambda_i p_i \tau_i}. \tag{1.9}$$

In this case, decreasing threshold G^T to zero or increasing the number of cycles N will make their effect on the optimal behavior negligible. In particular, as $N \to \infty$, finite-event excess-rate maximization is equivalent to classical infinite-time rate maximization. That is, when future opportunities are certain, choices should be made based on the balance between returned gain and required processing time (i.e., marginal rate). However, when future opportunities are uncertain (i.e., low N) or the threshold for success is high (i.e., high G^T), the optimal behavior shifts toward high-gain tasks that better guarantee meeting the success threshold. That is, when the agent is at risk of not meeting its success threshold, it spends relatively more time processing (i.e., earning gain for certain) and relatively less time searching.

(ii) *Time-discounted net gain.* Classical OFT describes behaviors that simultaneously maximize net gain and minimize foraging time. The relative importance of time minimization over gain maximization is varied in order to minimize the *opportunity cost* [51] of each activity. That is, the optimal rate of gain represents the maximum gain that can be returned for each unit of time. An OFT behavior accumulates gain in each activity only if there is no other activity that could return more mean gain for that amount of time. Hence, the optimal rate of gain represents the gain–time tradeoff that minimizes opportunity cost. Instead,

the gain–time tradeoff can be fixed *a priori*. In particular, an optimal behavior might maximize the advantage-to-disadvantage function

$$\mathrm{E}(G(N)) - G^T - w\,\mathrm{E}(T(N)) = N \frac{\left(\sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - c_i(\tau_i) - \frac{G^T}{N} - w\tau_i\right)\right) - c^s - w}{\sum_{i=1}^{n} \lambda_i p_i} \tag{1.10}$$

where discount rate $w \in \mathbb{R}$ is a constant representing the relative importance of the time objective over the gain objective. In cases where $p_1^- = p_2^- = \cdots = p_n^- = 0$, we assume that $c^s + w \geq 0$ to avoid the pathological case where it is best for the forager not to do any processing. We include the threshold G^T for completeness, but it only shifts the objective function by a constant value, and so it has no impact on the optimal solution. That is, when maximizing excess rate above, the relative value of gain and time float with the environment and the success threshold. For high thresholds in environments where encounters return relatively low gain, high-gain opportunities have a greater value. In this case, because the relative gain–time value is fixed, the success threshold has no effect on optimal solutions.

(iii) *Excess efficiency.* Stephens and Krebs [112] criticize using *efficiency* (i.e., benefit-to-cost) objectives because they neglect the impact of time and do not differentiate between behaviors that bring large gains at large costs and small gains at small costs. However, efficiency is a commonly used metric in engineering applications. Additionally, in our finite-event model, the impact of time is explicitly modeled by *cost* functions, and gain thresholds help to differentiate between

high-gain–high-cost and low-gain–low-cost behaviors. So we can define an efficiency metric that answers both concerns of Stephens and Krebs. Let $G_g^T \in \mathbb{R}$ be a minimum total *gross* gain required for success. An optimally efficient behavior will maximize the advantage-to-disadvantage function

$$\frac{\mathrm{E}(G(N)) + \mathrm{E}(C(N)) - G_g^T}{\mathrm{E}(C(N))} = \frac{\sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - \frac{G_g^T}{N} \right)}{c^s + \sum_{i=1}^{n} \lambda_i p_i c_i(\tau_i)}. \tag{1.11}$$

Again, decreasing threshold G_g^T or increasing number of cycles N sufficiently will make their impact on the optimal behavior negligible. If the task-processing agent is given a low success threshold or a large number of tasks to complete, it should not greatly perturb its behavior from the pure efficiency maximizer.

(iv) *Cost-discounted gain.* Just as the gain–time tradeoff can be fixed *a priori*, so can the gain–cost tradeoff. In particular, an optimal behavior could maximize the advantage-to-disadvantage function

$$\mathrm{E}(G(N)) + \mathrm{E}(C(N)) - G_g^T - w\,\mathrm{E}(C(N)) = N \frac{\left(\sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - \frac{G_g^T}{N} - w c_i(\tau_i) \right) \right) - w c^s}{\sum_{i=1}^{n} \lambda_i p_i} \tag{1.12}$$

where discount rate $w \in \mathbb{R}$ is a constant representing the relative importance of the cost objective over the gain objective. Again, we assume that $c^s + w \geq 0$ in cases where $p_1^- = \cdots = p_n^- = 0$ to avoid the pathological case, and we include the G_g^T threshold for completeness.

These four optimization objectives are all advantage-to-disadvantage functions, and they will be graphically examined in the examples from Section 1.2. Results of the application of the algorithms described in Section 1.3 will be given in Section 1.4.

1.2 A Graphical Optimization Approach

It can be instructive to study advantage-to-disadvantage functions graphically, especially when those functions lack properties required for analytical tractability. Here, we extend the graphical optimization approach described by Stephens and Krebs [112] to arbitrary advantage-to-disadvantage functions with arbitrary constraints. We use insights from the graphical process to compare and contrast the example optimization objectives discussed in Section 1.1. An analytical optimization approach is given in Section 1.3 along with algorithms that are guaranteed to find an optimal task-processing behavior for certain scenarios.

Because Equation (1.2) is a ratio, its value can be depicted as the slope of a line, and so optimization is finding the line with the steepest slope. This process is illustrated in Figure 1.1. Here, the shaded area is constructed by plotting the point $(\sum_{i=1}^{n} p_i d_i(\tau_i), \sum_{i=1}^{n} p_i a_i(\tau_i))$ for every $(\vec{p}, \vec{\tau}) \in \mathcal{F}$. For each of those points, the slope of the line connecting it to the point $(-d, -a)$ is equal to the advantage-to-disadvantage function for the corresponding behavior. So optimization consists of rotating a ray originating from $(-d, -a)$ from $-90°$ toward $90°$ and stopping at the angle just before the ray leaves the shaded region for the last time. If $(-d, -a)$ is within the shaded region, the ray will never leave the region between $-90°$ and $90°$ of rotation, and so the $90°$ ray should be used. In general, the shaded region need

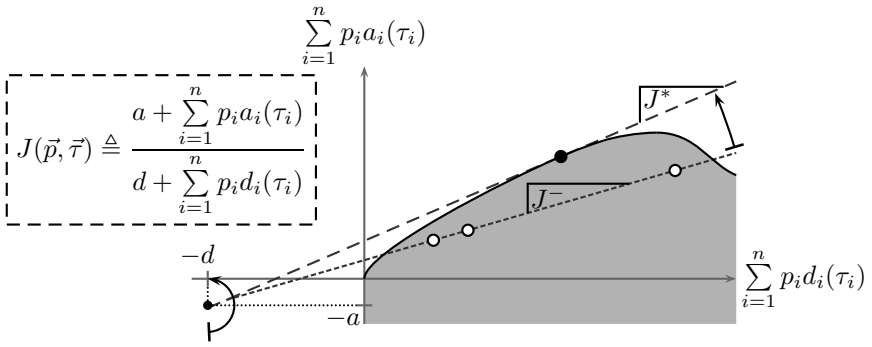


Figure 1.1: Graphical optimization of an advantage-to-disadvantage function. Each point in the shaded region corresponds to a different feasible behavior $(\vec{p}, \vec{\tau})$, and the slope of the line connecting that point to $(-d, -a)$ is the value of the objective function $J(\vec{p}, \vec{\tau})$ for that behavior. Hence, the three open circles correspond to three distinct behaviors that result in the same suboptimal rate J^-. An optimal behavior falls on the $(-d, -a)$-ray with the greatest positive slope. Here, the filled circle corresponds to the unique optimal behavior that results in the optimal rate J^*, which is the slope of the corresponding $(-d, -a)$-ray.

not be convex nor connected, but it should be *closed* (e.g., it could be a finite set of points).

1.2.1 Optimization of the Classical Objective

For the following, let

$$\lambda \triangleq \sum_{i=1}^{n} \lambda_i, \quad \overline{g} \triangleq \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i g_i(\tau_i), \quad \overline{c} \triangleq \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i c_i(\tau_i), \quad \text{and} \quad \overline{\tau} \triangleq \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i \tau_i.$$

The average time between encounters is $1/\lambda$, and λ_i/λ is the probability that an encounter is with a task of type $i \in \{1, 2, \ldots, n\}$. The expected processing gain, processing cost, and processing time for a single encounter are $\overline{g}$, $\overline{c}$, and $\overline{\tau}$, respectively,

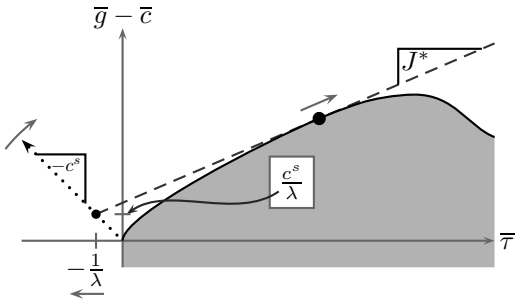


Figure 1.2: Graphical optimization of classical optimization objective. As search cost c^s or interarrival time $1/\lambda$ increases, the mean processing time $\overline{\tau}$ will increase.

and the rate of gain in Equation (1.3) is equivalent to

$$\frac{\left(\sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i \left(g_i(\tau_i) - c_i(\tau_i)\right)\right) - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i \tau_i} = \frac{\overline{g} - \overline{c} - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \overline{\tau}}. \tag{1.13}$$

For all $i \in \{1, 2, \ldots, n\}$, assume that λ_i/λ is constant with respect to λ (i.e., an encounter density); this assumption assists in the qualitative analysis of the impact of parameter changes on the optimal $(\overline{p}, \overline{\tau})$ behavior. Increases in the optimal $\overline{\tau}$ or $\overline{g}$ reflect increased preferences for higher processing times or processing gains, respectively.

Graphical optimization of this function is shown in Figure 1.2 for a given search cost c^s and encounter rate λ. As the average interarrival time $1/\lambda$ or search cost c^s increases, the point $(-1/\lambda, c^s/\lambda)$ anchoring the ray with slope J^* will move to the left. Consequently, the point of tangency between the ray and the feasible behavior frontier will move to the right. That point corresponds to the optimal combination of *average* processing time $\overline{\tau}$ and average net processing gain $(\overline{g} - \overline{c})$. If c^s or $1/\lambda$ increase to beyond the point where c^s/λ matches the $(\overline{g} - \overline{c})$-peak of the feasible

behavior frontier, the optimal average processing time $\overline{\tau}$ will continue to increase although the optimal average net processing gain $(\overline{g} - \overline{c})$ decreases.

In words, small increases in search cost c^s/λ cause the optimal processing time to increase in order to return more average processing gain from each encounter. However, large increases in search cost c^s/λ cause the optimal processing time per encounter to increase in spite of the resulting decreasing average processing gain per encounter. In this region of decreasing average processing gain, the increased average processing time *preempts* the very costly searching (i.e., rather than adding gain from processing, search cost is being removed by searching relatively less). This effect is a result of opportunity cost minimization; there is less opportunity cost for additional processing when searching is itself very costly. Processing tasks not only accumulates gain, but it prevents the loss of gain through searching. A task-processing agent ceases processing a task when it is likely that a task with higher marginal returns will be found quickly. However, when there is a long time between encountered tasks, it is better to burn fuel processing a task longer than burning fuel searching for a new task because gain is accumulated while processing but not while searching.

1.2.2 Optimal Behaviors from Alternate Objectives

For simplicity in this graphical analysis, assume the special case of patch problems (i.e., $p_i^- = p_1^+ = p_i^* \triangleq 1$ for each $i \in \{1, 2, \ldots, n\}$). These results can be extended to prey-model problems by translating increased task-processing times to increased preference for task types with higher processing times; these prey-model effects (e.g., preference reversal) are explored in Section 1.4 after the analytical methods in Section 1.3 are introduced. Consider finite-event maximization of:

(i) *Excess rate.* In this case, Equation (1.9) is

$$\frac{\sum_{i=1}^{n} \lambda_i \left(g_i(\tau_i) - c_i(\tau_i)\right) - \sum_{i=1}^{n} \lambda_i \frac{G^T}{N} - c^s}{1 + \sum_{i=1}^{n} \lambda_i \tau_i} = \frac{\overline{g} - \overline{c} - \left(\frac{G^T}{N} + \frac{c^s}{\lambda}\right)}{\frac{1}{\lambda} + \overline{\tau}}, \tag{1.14}$$

which is equivalent to Equation (1.13) with the per-cycle search cost c^s/λ augmented by the per-cycle average success threshold G^T/N. That is, in the patch case, every finite-event task-processing agent that maximizes excess rate can be transformed into an equivalent infinite-time rate maximizer by increasing search cost. So increasing threshold G^T or decreasing number of cycles N will have the same effect on the finite-event excess-rate maximizer as increasing search cost c^s on the infinite-time rate maximizer, and Figure 1.2 also describes this case. This result is consistent with the idea that thresholds induce an exploration cost which is reduced when future opportunities are certain. That is, because the agent receives no gain while searching, searching is a less desirable activity when high gain thresholds must be met.

Stephens and Charnov [111] present a risk-sensitive model of foraging behavior that predicts the optimal combination of gain mean and variance to maximize the probability of reaching a critical energetic threshold. Stephens and Krebs [112] show that optimal risk-sensitive processing times will be:

- greater than rate-maximized processing times when the energetic threshold is less than expected gain. Hence, present gains are increased to reduce lifetime gain variance (i.e., reduce uncertainty).
- less than rate-maximized processing times when the energetic threshold is greater than expected gain. Hence, lifetime gain variance is increased (i.e.,

to increase probability of very high accumulated gain) by increasing number of lifetime encounters at the cost of reduced lifetime mean gain.

- identical to rate-maximized processing times when the energetic threshold is equal to expected gain.

So the time-limited task-processing agent trades per-encounter gain with number of encounters to maximize the probability of reaching a success threshold.

The excess-rate task-processing model modifies the classical rate-maximizing model in a similar way. However, this model has a fixed number of encounters and a variable time, and the gain success threshold is essentially a *forced cost.* Consequently, results are opposite the expected results from risk-sensitivity theory. In particular, when the success threshold is:

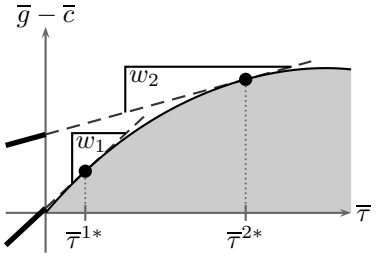
- positive, the agent *increases* processing times. In this context, a positive threshold implies that the agent suffers a *loss* from each processed encounter, and so the opportunity cost of more processing time is reduced. The agent delays the next encounter in order to mitigate the effect of the next positive threshold.
- negative, the agent *decreases* processing times. In this context, a negative threshold implies that the agent receives a *gain* from each processed encounter, and so the opportunity cost of more processing is increased. At this heightened cost, the agent cannot afford to spend more time processing when future negative thresholds are left to be encountered.
- zero, the agent behaves like like a classical rate maximizer.

(ii) *Time-discounted net gain.* Under the patch assumption, the time-discounted net-gain (TDNG) objective function in Equation (1.10) does not have a conenient slope-maximizing graphical interpretation; however, a different graphical method can be used, and this method reveals a relationship between time-discounted net-gain maximization and rate maximization. In particular, Equation (1.10) in the patch case is equivalent to

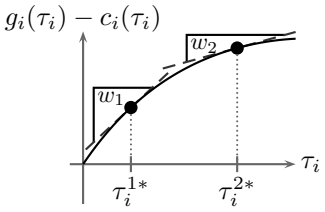
$$N\underbrace{((\overline{g}-\overline{c})-w\overline{\tau})}_{(*)}-\underbrace{N\left(\frac{c^s}{\lambda}+w\frac{1}{\lambda}+\frac{G^T}{N}\right)}_{(**)}.$$

Because $N > 0$ and $(**)$ is constant, TDNG optimization is identical to optimization of $(*) \triangleq (\overline{g}-\overline{c})-w\overline{\tau}$. So possible solutions come from the dark upper frontier in Figure 1.2, which corresponds with the behaviors that maximize $(\overline{g}-\overline{c})$ for a given $\overline{\tau}$ and minimize $\overline{\tau}$ for a given $\overline{g}-\overline{c}$ (i.e., the behavior will be Pareto optimal with respect to these two optimization objectives). The particular solution from this frontier is depends on the selection of $w \in \mathbb{R}$, which is a *cost rate* that converts time into gain.

Graphical TDNG optimization is shown in Figure 1.3. Just as in Figure 1.2, each point in the shaded area of Figure 1.3(a) is the pair $(\overline{\tau}, \overline{g}-\overline{c})$ corresponding to a particular $(\vec{p},\vec{\tau})$ behavior. In this example, the frontier of the shaded area is smooth and continuous, and so it can be represented as a differentiable function $(\overline{g}-\overline{c})(\overline{\tau})$, and the optimal processing average processing time $\overline{\tau}^*$ is the point where $(\overline{g}-\overline{c})'(\overline{\tau}^*) = w$ and $(\overline{g}-\overline{c})'' < 0$. So optimization of smooth frontiers is depicted as finding a point of deceleration that is tangent to a line with slope w. Two such lines are shown in Figure 1.3(a); the thick portions of



(a) Aggregate showing rate relationship.



(b) Optimization for type $i \in \{1, 2, \ldots, n\}$.

Figure 1.3: Time-discounted net-gain optimization. The shaded area used in graphical rate maximization is also used in (a); however, the optimal TDNG behavior corresponds to the point of tangency with a line of slope w. Here, the steeper cost rate $w_1 > w_2$ is associated with a shorter average time $\overline{\tau}^{1*} < \overline{\tau}^{2*}$ because time is more expensive. As shown in (b), for a given w, the optimal TDNG τ_i is the point of tangency between of tangency between $g_i(\tau_i) - c_i(\tau_i)$ and a line of slope w.

those lines correspond to (c^s, λ) combinations where TDNG and rate maximization are equivalent. As discussed by Houston and McNamara [51], if w is set to the maximal value of Equation (1.14) (i.e., the maximum long-term rate of gain), the corresponding gain–time tradeoff will also maximize long-term rate of gain.

In Section 1.3, we give precise analytical methods for optimization of this function. Meanwhile, we observe that because $(\overline{g} - \overline{c}) - w\overline{\tau}$ is a weighted sum, then for each type $i \in \{1, 2, \ldots, n\}$, the optimal processing time τ_i^* is the point that maximizes $g_i(\tau_i) - c_i(\tau_i) - w\tau_i$. So TDNG optimization is equivalent to the decoupled optimization of the n versions of this expression. In particular, for each $i \in \{1, 2, \ldots, n\}$, if optimal processing time $\tau_i^* \in (\tau_i^-, \tau_i^+)$, then it must be that $g_i'(\tau_i) - c_i'(\tau_i) = w$ and $g_i''(\tau_i) - c_i''(\tau_i) < 0$. So optimization of each type has an identical structure as optimization of the aggregate. As shown in Figure 1.3(b),

each optimal processing time is the point of tangency with a line of slope w. As shown by the dark line segments in Figure 1.3(a), once the optimal processing time is found for every type, the line with slope w that intersects $(\overline{\tau}^*, (\overline{g} - \overline{c})^*)$ can be used to find the set of (c^s, λ) combinations that lead to the same optimal behavior in the rate-maximizing case.

(iii) *Excess efficiency.* The graphical optimization approach shows that efficiency maximization and rate maximization can have similar optimal solutions. In these patch problems, Equation (1.11) is

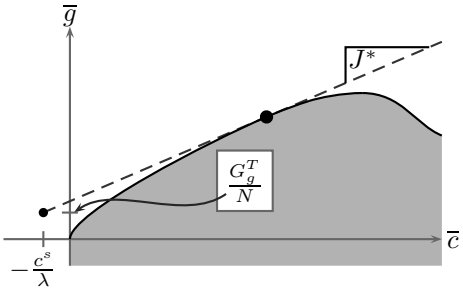
$$\frac{\sum_{i=1}^{n} \lambda_i g_i(\tau_i) - \sum_{i=1}^{n} \lambda_i \frac{G_g^T}{N}}{c^s + \sum_{i=1}^{n} \lambda_i c_i(\tau_i)} = \frac{\overline{g} - \frac{G_g^T}{N}}{\frac{c^s}{\lambda} + \overline{c}}, \tag{1.15}$$

which resembles Equation (1.14) and has optimization depicted by Figure 1.4(a). In particular, if the processing cost functions are monotonically increasing with time, changes in the environment associated with increases in optimal-rate processing time will also be associated with increases in optimal-efficiency processing time. The efficiency defined by Equation (1.15) is equivalent to a long-term rate of gain after time has been converted to a different currency. In this case, those currency conversions vary among types and the environment.

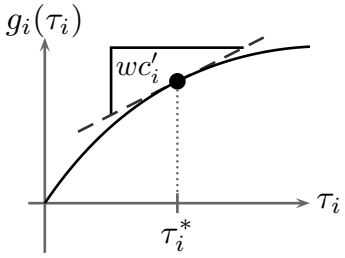
(iv) *Cost-Discounted Gain.* Under the patch assumption, the cost-discounted gain (CDG) function in Equation (1.12) is

$$N\left(\overline{g} - w\overline{c}\right) - N\left(\frac{c^s}{\lambda} + \frac{G_g^T}{\lambda}\right),$$

which is maximized at the same point as $\overline{g} - w\overline{c}$. As in TDNG optimization, optimization of each type can be decoupled from the other types. In particular,



(a) Excess-efficiency maximization.



(b) CDG optimization for type $i \in \{1, \ldots, n\}$ with linear processing cost.

Figure 1.4: Optimization based on efficiency. In (a), excess-efficiency maximization is shown to be similar to excess-rate maximization. Here, as search cost c^s or interarrival time $1/\lambda$ increases, the mean processing cost $\overline{c}$ will increase. So long as processing cost increases are due to processing time increases, mean processing time $\overline{\tau}$ will also increase. This result qualitatively matches what is expected for rate maximization. In (b), cost-discounted gain optimization is shown for one particular type. In this example, the type's processing cost $c_i(\tau_i)$ is depicted as a linear function $c_i'\tau_i$, which makes CDG optimization identical to TDNG optimization when each type's processing time is scaled by c_i'.

for each $i \in \{1, 2, \ldots, n\}$, each optimal processing time τ_i^* maximizes $g_i(\tau_i) - wc_i(\tau_i)$. In the special case where processing costs are linear in time, optimization is depicted by Figure 1.4(b). That is, optimization is nearly identical to the TDNG case except that the processing time in type $i \in \{1, 2, \ldots, n\}$ is scaled by c_i'.

So not only can each of the finite-event optimization objectives be optimized using similar methods, but they all have results that are qualitatively identical to classical rate-maximization results. Hence, these optimization objectives can be used to model behaviors that do not perfectly fit the classical foraging model.

The expectation-of-ratios objective in Equation (1.5) apparently does not have a convenient structure for graphical optimization. In Section 1.3, we give analytical strategies for its optimization. Meanwhile, to motivate a graphical optimization method, we observe that because Equation (1.5) is a weighted sum, then it can be shown that optimization of Equation (1.5) reduces to optimization of $(g_i(\tau_i) - c_i(\tau_i) - c^s/\lambda)/(1/\lambda + \tau_i)$ for each $i \in \{1, 2, \ldots, n\}$. Each of these functions is an advantage-to-disadvantage function nearly identical to Equation (1.14) when $n = 1$, and so the standard graphical optimization procedure can be applied to each type separately. However, although expectation-of-ratio optimization can be completed separately for each type $i \in \{1, 2, \ldots, n\}$, the optimal processing times are still related by global parameters c^s and λ.

1.3 An Analytical Optimization Approach

The graphical approach described in Section 1.2 makes qualitative predictions about average behaviors but is inappropriate for more precise investigations. Here, we apply a more rigorous analysis approach. In particular, we describe the mathematical structure of smooth objective functions at points of optimality. In Section 1.3.3, we provide detailed descriptions of algorithms that are guaranteed to find these points of optimality. However, to highlight the salient features common to all of those algorithms, we first connect the characterization of a generalized task-processing optimum to the popular algorithms used in OFT-type applications (Section 1.3.2). Finally, we summarize the application of algorithms from Section 1.3.3 to the example advantage-to-disadvantage functions described in Section 1.1, and we list some observations about important similarities and differences in the results.

1.3.1 Characterization of Optimal Behaviors

Here, we must characterize the optimality of Equation (1.2) over the set of behaviors in Equation (1.1). We give conditions that guarantee that a behavior is a *strict local maximum* of Equation (1.2). If the optimization objective is *strictly convex*, these conditions describe its *unique global maximum.* Our analysis uses Lagrange multiplier theory [18], and so we assume that a_i and d_i are twice continuously differentiable for each type $i \in \{1, 2, \ldots, n\}$ in an open neighborhood of the optimal behavior.

Take some feasible behavior $(\vec{p}^*, \vec{\tau}^*) \in \mathcal{F}$, and let $A^* \triangleq A(\vec{p}^*, \vec{\tau}^*)$ and $D^* \triangleq D(\vec{p}^*, \vec{\tau}^*)$. For each type $j \in \{1, 2, \ldots, n\}$, assume that

$$p_j^* = \begin{cases} p_j^- & \text{if } D^* a_j(\tau_j^*) < A^* d_j(\tau_j^*), \\ p_j^+ & \text{if } D^* a_j(\tau_j^*) > A^* d_j(\tau_j^*). \end{cases} \tag{1.16}$$

Because $p_j = p_j^-$ or $p_j = p_j^+$ for each type $j \in \{1, 2, \ldots, n\}$, we call Equation (1.16) the *extreme-preference rule.* Additionally, for each type $j \in \{1, 2, \ldots, n\}$, assume that

$$\tau_j^- < \tau_j^* < \tau_j^+ \quad \text{and} \quad \begin{cases} D^* a_j'(\tau_j^*) = A^* d_j'(\tau_j^*) \\ \qquad\qquad \text{and} \\ D^* a_j''(\tau_j^*) < A^* d_j''(\tau_j^*), \end{cases} \tag{1.17a}$$

or

$$\tau_j^* = \tau_j^- \quad \text{and} \quad D^* a_j'(\tau_j^*) < A^* d_j'(\tau_j^*), \tag{1.17b}$$

or

$$\tau_j^* = \tau_j^+ \quad \text{and} \quad D^* a_j'(\tau_j^*) > A^* d_j'(\tau_j^*). \tag{1.17c}$$

The condition in Equation (1.17a) ensures that the interior coordinate is at a stationary point of the objective function with local convexity. The conditions in Equations (1.16), (1.17b), and (1.17c) ensure that the extreme coordinates sit on downward slopes at the edge of the objective function. So Equations (1.16) and (1.17) define *sufficiency conditions* for optimality. Under these conditions, $(\vec{p}^*, \vec{\tau}^*)$ must be a strict local maximum. If Equation (1.2) is convex everywhere, then the behavior is its unique global maximum.

1.3.2 Motivating Interpretations

Detailed algorithms for finding points that meet the described optimality conditions are given in Section 1.3.3. Here, we show how the conditions in Equations (1.16) and (1.17) are natural generalizations of familiar classical foraging theory and present summaries of the existing OFT algorithms to motivate the general cases in Section 1.3.3. Elements of these two cases can be found in each of the generalized algorithms. In particular, task types are ranked by some generalized profitability and then partitioned into take-most and take-few sets, and processing times are found through some generalized marginal value theorem.

Prey model as optimal task-type choice: profitability ordering

When applied to Equation (1.3) for the prey model case (i.e., when it is given that $\tau_i^+ = \tau_i^- = \tau_i^*$ for each type i), the extreme-preference rule in Equation (1.16) is equivalent to

$$p_j^* = \begin{cases} 0 & \text{if } \frac{g_j(\tau_j^*) - c_j(\tau_j^*)}{\tau_j^*} < \frac{\left(\sum_{i=1}^n \lambda_i p_i^* \left(g_i(\tau_i^*) - c_i(\tau_i^*)\right)\right) - c^s}{1 + \sum_{i=1}^n \lambda_i p_i^* \tau_i^*}, \\ 1 & \text{if } \frac{g_j(\tau_j^*) - c_j(\tau_j^*)}{\tau_j^*} > \frac{\left(\sum_{i=1}^n \lambda_i p_i^* \left(g_i(\tau_i^*) - c_i(\tau_i^*)\right)\right) - c^s}{1 + \sum_{i=1}^n \lambda_i p_i^* \tau_i^*}, \end{cases} \tag{1.18}$$

which is the familiar *zero–one rule* [112] where $a_j(\tau_j^*)/d_j(\tau_j^*)$ is the *profitability* $g_j(\tau_j^*)/\tau_j^*$ of type $j \in \{1, 2, \ldots, n\}$. This rule states that if task types are indexed by profitability so that

$$\frac{g_1(\tau_1^*) - c_1(\tau_1^*)}{\tau_1^*} > \frac{g_2(\tau_2^*) - c_2(\tau_2^*)}{\tau_2^*} > \cdots > \frac{g_n(\tau_n^*) - c_n(\tau_n^*)}{\tau_n^*},$$

then there is a critical $k^* \in \{0, 1, \ldots, n\}$ such that

$$p_j^* = \begin{cases} 1 & \text{if } j \leq k^* \\ 0 & \text{if } j > k^*. \end{cases}$$

That is, k^* partitions the set of types $\{1, 2, \ldots, n\}$ into a take-all set $\{1, 2, \ldots, k^*\}$ and a take-none set $\{k^* + 1, \ldots, n\}$. Moreover, it is the optimal rate $J^* \triangleq J(\vec{p}^*, \vec{\tau}^*)$ that partitions the profitabilities in the same manner. That is,

$$\frac{g_1^*}{\tau_1^*} > \cdots > \frac{g_{k^*}}{\tau_{k^*}^*} > J^* > \frac{g_{k^*+1}}{\tau_{k^*+1}^*} > \cdots > \frac{g_n^*}{\tau_n^*}$$

where optimal net gain $g_j^* \triangleq g_j(\tau_j^*) - c_j(\tau_j^*)$ for each $j \in \{1, 2, \ldots, n\}$. This relationship is depicted in Figure 1.5 for a case with $n = 5$. Key results from this analysis are that:

- There is an ordering of task-type preference that is invariant of the environment. If it is optimal to exclude tasks of type k, tasks of type $\ell > k$ must also be excluded. Similarly, if it is optimal to include tasks of type k, tasks of type $j < k$ must also be included. This ordering does not depend on the encounter rates nor the cost of search.

- As the maximum long-term rate of gain J^* decreases (e.g., due to a global decline in encounter rates or an increase in search cost), the optimal task-processing strategy should be more inclusive (i.e., more types should be included in the

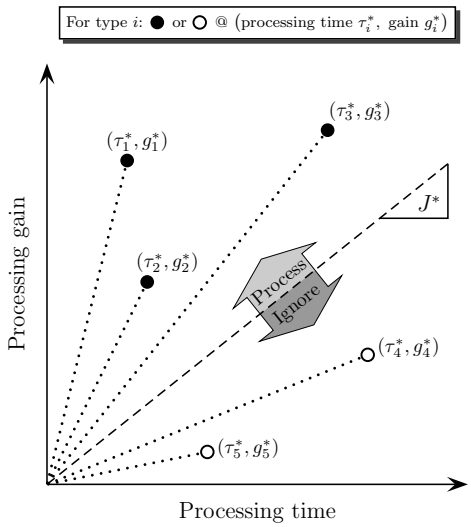


Figure 1.5: Graphical summary of prey model result. For a task type $i \in \{1, 2, 3, 4, 5\}$, the average processing time τ_i^* and average net gain g_i^* is plotted as a dot. The maximum long-term rate of gain J^* is the slope of the dashed line which separates the processed types, 1, 2, and 3, from the ignored types, 4 and 5. The profitability of each type is the slope of the dotted line connecting the origin to its (gain, time)-coordinate.

take-all set). Likewise, as the maximum long-term rate of gain J^* increases, the optimal strategy should be more exclusive.

Moreover, the zero–one rule means that finding the optimal take-all set of task types involves a combinatorial search through a set of 2^n different $\vec{p}$ preference profiles. However, because of the invariant task-type ordering, there are at most $n+1$ possible $\vec{p}$ vectors that must to be checked (i.e., the preference vectors $[0, 0, \ldots, 0]^\top$, $[1, 0, \ldots, 0]^\top$, $[1, 1, \ldots, 0]^\top$, $\ldots$, and $[1, 1, \ldots, 1]^\top$).

Patch model as optimal processing-time choice: marginal value

When Equation (1.17) is applied to Equation (1.3) for the patch model case (i.e., when it is given that $p_i^- = p_i^+ = p_i^* = 1$ for each type i), Equation (1.17a) is equivalent

to

$$\tau_j^* > 0 \quad \text{and} \quad g_j''(\tau_j^*) - c_j''(\tau_j^*) < 0$$

and

$$g_j'(\tau_j^*) - c_j'(\tau_j^*) = \frac{\left(\sum_{i=1}^{n} \lambda_i p_i^* \left(g_i(\tau_i^*) - c_i(\tau_i^*)\right)\right) - c^s}{1 + \sum_{i=1}^{n} \lambda_i p_i^* \tau_i^*}, \tag{1.19}$$

which is the familiar marginal value theorem [23, 24]. Consider the special single-type patch case where $n = 1$ and $p_1^- = p_1^+ = 1$. Then Equation (1.19) is equivalent to

$$g_1'(\tau_1^*) - c_1'(\tau_1^*) = \frac{(g_1(\tau_i^*) - c_1(\tau_1^*)) - \frac{c^s}{\lambda_1}}{\frac{1}{\lambda_1} + \tau_i^*}. \tag{1.20}$$

Additionally, the graphical analysis in Figure 1.2 of this case degenerates so that all behaviors fall on the bold Pareto frontier, and that frontier traces the shape of the net gain function $\tau_1 \mapsto g_1(\tau_1) - c_1(\tau_1)$. The resulting graph is exactly the situation described by Equation (1.20). That is, the optimal task-1 processing time τ_1^* occurs at the point of tangency between the function $g_1 - c_1$ and a ray originating from the point $(-1/\lambda_1, c^s/\lambda_1)$.

1.3.3 Algorithms for Finding an Optimal Generalized Foraging Behavior

Now that we have characterized optimal behaviors, we present three algorithms that find an optimal behavior $(\vec{p}^*, \vec{\tau}^*) \in \mathcal{F}$ for a task-processing agent when certain assumptions are met. Because each algorithm has different requirements than the others, one algorithm may apply to one task-processing scenario better than another. However, all three share the following characteristics:

- For each type $i \in \{1, 2, \ldots, n\}$, the functions a_i and d_i are assumed to be twice continuously differentiable.

- The types are ordered by maximum *generalized profitability* so that

$$\max_{\tau_1 \in [\tau_1^-, \tau_1^+]} \left\{ \frac{a_1(\tau_1)}{d_1(\tau_1)} \right\} > \max_{\tau_2 \in [\tau_2^-, \tau_2^+]} \left\{ \frac{a_2(\tau_2)}{d_2(\tau_2)} \right\} > \cdots > \max_{\tau_n \in [\tau_n^-, \tau_n^+]} \left\{ \frac{a_n(\tau_n)}{d_n(\tau_n)} \right\}.$$

  Determination of this ordering is not simple to do in general, but the assumptions for each case below greatly simplify the task. In two of the three cases, the maximum generalized profitability a_i/d_i occurs at $\tau_i = \tau_i^-$ for all $i \in \{1, 2, \ldots, n\}$. In the other case, maximizing a_i/d_i is equivalent to either maximizing or minimizing a_i for all $i \in \{1, 2, \ldots, n\}$.

- To satisfy the extreme-preference rule from Equation (1.16), the n types are partitioned into a high-preference set and a low-preference set. An *optimal pool size* $k^* \in \{0, 1, \ldots, n\}$ exists so that the k^* types with the highest profitabilities form the high-preference set and the $n - k^*$ other types form the low-preference set. In particular, for each type $i \in \{1, 2, \ldots, n\}$ and each $k \in \{0, 1, \ldots, n\}$, the *conditional preference* p_i^k is so that

$$p_i^k \triangleq \begin{cases} p_i^+ & \text{if } i \leq k, \\ p_i^- & \text{if } i > k, \end{cases} \tag{1.21}$$

  and the optimal behavior $(\vec{p}^*, \vec{\tau}^*)$ will have $p_j^* = p_j^{k^*}$ for all $j \in \{1, 2, \ldots, n\}$.

So after ordering the types appropriately, each algorithm finds an optimal pool size and a set of optimal processing times for that pool size.

Generalized Prey Algorithm

Stephens and Krebs [112] describe a prey model algorithm that finds a $(\vec{p}^*, \vec{\tau}^*)$ to optimize Equation (1.3) when $\vec{\tau}^*$ is known *a priori* (i.e., it is constrained to a single point by the environment). Because τ_i^* is fixed, the functions a_i and d_i are replaced

with constants $a_i(\tau_i^*)$ and $d_i(\tau_i^*)$, respectively. Here, we present a generalized version of the algorithm that does *not* fix $\vec{\tau}^*$. Instead, we only require that the function d_i is *constant* and *nonzero* for each type $i \in \{1, 2, \ldots, n\}$.

Assume that for distinct types $j, k \in \{1, 2, \ldots, n\}$,

(i) The function d_j is constant and nonzero.

(ii) Functions d_j and d_k have the same sign, and constant d is either zero or also has this sign.

(iii) If $d = 0$, then $a < 0$.

(iv) If d_j is positive, then a_j has a maximum, and if d_j is negative, then a_j has a minimum (i.e., profitability function a_j/d_j has a maximum).

These assumptions guarantee that the objective function has a maximum.

Using item (iv), for each type $j \in \{1, 2, \ldots, n\}$, let τ_j^* be the point that maximizes the generalized profitability function a_j/d_j. Also assume that:

(v) The indices are ordered by generalized profitability so that

$$\frac{a_1(\tau_1^*)}{d_1(\tau_1^*)} > \frac{a_2(\tau_2^*)}{d_2(\tau_2^*)} > \cdots > \frac{a_n(\tau_n^*)}{d_n(\tau_n^*)}.$$

Finally, to ensure strict local convexity of the solution, assume that:

(vi) For any $k \in \{0, 1, \ldots, n-1\}$,

$$\frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^*)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^*)} \neq \frac{a_{k+1}(\tau_{k+1}^*)}{d_{k+1}(\tau_{k+1}^*)}$$

where p_i^k is defined by Equation (1.21). By these assumptions, there is an optimal pool size $k^* \in \{0, 1, \ldots, n\}$ such that

$$k^* \triangleq \min \left(\left\{ \left\{ k \in \{0, 1, \ldots, n-1\} : \underbrace{\frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^*)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^*)} > \frac{a_{k+1}(\tau_{k+1}^*)}{d_{k+1}(\tau_{k+1}^*)}}_{(*)} \right\} \cup \{n\} \right\} \right). \tag{1.22}$$

So k^* can be found iteratively by choosing the smallest $k \in \{0, 1, \ldots, n-1\}$ that satisfies the underbraced expression $(*)$. Then, the behavior $(\vec{p}^*, \vec{\tau}^*)$ with

$$p_j^* \triangleq p_j^{k^*} = \begin{cases} p_j^+ & \text{if } j \leq k^*, \\ p_j^- & \text{if } j > k^* \end{cases}$$

for each type $j \in \{1, 2, \ldots, n\}$ will be optimal. That is, the optimal behavior gives highest preference to the k^* types with highest profitability and ignores the other $n - k^*$ types. So given the n generalized profitabilities, an optimal behavior can be found by iterating through no more than $n + 1$ candidate behaviors.

Alternate Generalized Prey Algorithm

The algorithm in Section 1.3.3 cannot be used with the expectation-of-ratios function in Equation (1.5) because it has $d_i \equiv 0$ for each type $i \in \{1, 2, \ldots, n\}$. Here, we provide a similar algorithm to handle this case and others so long as $d \neq 0$. The algorithm assigns an *infinite* generalized profitability to each type $i \in \{1, 2, \ldots, n\}$ with $d_i \equiv 0$ and treats all other types in the same manner as in Section 1.3.3. That is, types are ranked by their *extended* generalized profitabilities.

Assume that for distinct types $j, k \in \{1, 2, \ldots, n\}$,

(i) The function d_j is constant and *possibly* zero.

(ii) The constant $d \neq 0$.

(iii) If $d_j \neq 0$, then it has the same sign as d.

(iv) If d is positive, then a_j has a maximum, and if d is negative, then a_j has a minimum (i.e., function a_j/d has a maximum).

These assumptions are nearly identical to the ones in Section 1.3.3. Here, cases with $d = 0$ are excluded in order to include cases with $d_i \equiv 0$ for at least one type $i \in \{1, 2, \ldots, n\}$.

Take $(\vec{p}^*, \vec{\tau}^*) \in \mathcal{F}$. Using item (iv), let τ_j^* be the point that maximizes a_j/d for each type $j \in \{1, 2, \ldots, n\}$. Also assume that:

(v) The indices are ordered by *extended generalized profitability* so that there exists some $\ell, u \in \{0, 1, \ldots, n+1\}$ with $\ell < u$ and

$$d_j(\tau_j^*) = 0 \quad \text{and} \quad \frac{a_j(\tau_j^*)}{d} > 0 \quad \text{for each type } j \in \{1, \ldots, \ell\}$$

and

$$\frac{a_{\ell+1}(\tau_{\ell+1}^*)}{d_{\ell+1}(\tau_{\ell+1}^*)} > \frac{a_2(\tau_{\ell+2}^*)}{d_2(\tau_{\ell+2}^*)} > \cdots > \frac{a_{u-2}(\tau_{u-2}^*)}{d_{u-2}(\tau_{u-2}^*)} > \frac{a_{u-1}(\tau_{u-1}^*)}{d_{u-1}(\tau_{u-1}^*)}$$

and

$$d_j(\tau_j^*) = 0 \quad \text{and} \quad 0 > \frac{a_j(\tau_j^*)}{d} \quad \text{for each type } j \in \{u, \ldots, n\}.$$

For strict local convexity of the solution, assume that:

(vi) For each type $k \in \{\ell, \ell+1, \ldots, u-2\}$,

$$\frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^*)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^*)} \neq \frac{a_{k+1}(\tau_{k+1}^*)}{d_{k+1}(\tau_{k+1}^*)}$$

where p_i^k is defined by Equation (1.21). By these assumptions, the optimal pool size $k^* \in \{\ell, \ell+1, \ldots, u-2\}$ is so that

$$k^* \triangleq \min\left(\left\{\left\{k \in \{\ell, \ell+1, \ldots, u-2\} : \underbrace{\frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^*)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^*)} > \frac{a_{k+1}(\tau_{k+1}^*)}{d_{k+1}(\tau_{k+1}^*)}}_{(*)}\right\} \cup \{u-1\}\right).$$

So k^* can be found iteratively by choosing the smallest $k \in \{\ell, \ell+1, \ldots, u-2\}$ that satisfies the underbraced expression $(*)$. Then, the behavior $(\vec{p}^*, \vec{\tau}^*)$ with

$$p_j^* \triangleq p_j^{k^*} = \begin{cases} p_j^+ & \text{if } j \leq k^*, \\ p_j^- & \text{if } j > k^* \end{cases}$$

for each type $j \in \{1, 2, \ldots, n\}$ will be optimal. So the optimal behavior can also be found with a search through no more than $n+1$ candidates.

Generalized Patch Algorithm

The algorithms in Sections 1.3.3 and 1.3.3 cannot be used with Equation (1.3) in the patch case because the function $d_i(\tau_i) \triangleq \tau_i$ for each type $i \in \{1, 2, \ldots, n\}$ (i.e., it is *not constant*). Here, we give a generalized patch algorithm that can be used when each generalized profitability function comes from a certain class of *decreasing* functions.

Assume that for distinct types $j, k \in \{1, 2, \ldots, n\}$,

(i) The profitability function is *strictly decreasing* so that $(a_j(\tau_j)/d_j(\tau_j))' < 0$ for any $\tau_j \in (\tau_j^-, \tau_j^+)$.

(ii) The convexities of a_j and d_j are such that $(a_j'(\tau_j)/d_j'(\tau_j))' < 0$ for any $\tau_j \in (\tau_j^-, \tau_j^+)$.

(iii) For any $\tau_j \in (\tau_j^-, \tau_j^+)$, $d_j(\tau_j) \neq 0$ and

- If $d_j(\tau_j) > 0$, then $d_j'(\tau_j) > 0$ (i.e., positive functions are rising).
- If $d_j(\tau_j) < 0$, then $d_j'(\tau_j) < 0$ (i.e., negative functions are falling).

So the *magnitude* of d_j is nonzero and increasing everywhere on its interior.

(iv) For any $\tau_j \in (\tau_j^-, \tau_j^+)$ and any $\tau_k \in (\tau_k^-, \tau_k^+)$, $d_j(\tau_j)$ and $d_k(\tau_k)$ have the same sign, and constant d is either zero or also has this sign.

These assumptions allow for the case where $d_i(\tau_i^-) = 0$ for some type $i \in \{1, 2, \ldots, n\}$. So to guarantee that the objective function is well defined, also assume that:

(v) If $d_1(\tau_1^-) = d_2(\tau_2^-) = \cdots = d_n(\tau_n^-) = 0$, then $d \neq 0$.

By the assumptions, for each type $i \in \{1, 2, \ldots, n\}$, the profitability function $a_i(\tau_i)/d_i(\tau_i)$ will be well defined for all $\tau_i \in (\tau_i^-, \tau_i^+)$, but it may have a singularity at $\tau_i = \tau_i^-$, and so we *extend* the *initial profitability* so that

$$\frac{a_i(\tau_i^-)}{d_i(\tau_i^-)} \triangleq \lim_{\tau_i \to \tau_i^-} \frac{a_i(\tau_i)}{d_i(\tau_i)}.$$

When there is no singularity or when the singularity is removable, this limit will be finite. That is, the types can be partitioned into a set with unbounded profitabilities and a set with bounded profitabilities whose bounds can be ordered. So assume that:

(vi) The indices are ordered so that there exists some $\ell \in \{0, 1, \ldots, n-1\}$ where

$$\frac{a_j(\tau_j^-)}{d_j(\tau_j^-)} = \infty \quad \text{for each type } j \in \{1, \ldots, \ell\}$$

and

$$\infty > \frac{a_{\ell+1}(\tau_{\ell+1}^-)}{d_{\ell+1}(\tau_{\ell+1}^-)} > \frac{a_2(\tau_{\ell+2}^-)}{d_2(\tau_{\ell+2}^-)} > \cdots > \frac{a_{n-1}(\tau_{n-1}^-)}{d_{n-1}(\tau_{n-1}^-)} > \frac{a_n(\tau_n^-)}{d_n(\tau_n^-)}.$$

Next, for each type $j \in \{1, 2, \ldots, n\}$ and any $k \in \{0, 1, \ldots, n\}$, define τ_j^k so that

$$\frac{a_j'(\tau_j^k)}{d_j'(\tau_j^k)} = \frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^k)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^k)}$$

or let

$$\tau_j^k \triangleq \begin{cases} \tau_j^- & \text{if } \dfrac{a_j'(\tau_j^k)}{d_j'(\tau_j^k)} < \dfrac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^k)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^k)}, \\ \tau_j^+ & \text{if } \dfrac{a_j'(\tau_j^k)}{d_j'(\tau_j^k)} > \dfrac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^k)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^k)} \end{cases}$$

where p_i^k is defined by Equation (1.21). These definitions represent a *generalized marginal value theorem.* That is, τ_i^k represents the optimal patch residence time in patches of type i given that the optimal pool size is k; it is well defined by the assumption in item (ii). Again, to guarantee strict convexity of the objective function at the optimal behavior, assume that:

(vii) For any $k \in \{\ell, \ell + 1, \ldots, n - 1\}$,

$$\frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^k)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^k)} \neq \frac{a_{k+1}(\tau_{k+1}^-)}{d_{k+1}(\tau_{k+1}^-)}.$$

Finally, define optimal pool size $k^* \in \{\ell, \ell + 1, \ldots, n\}$ so that

$$k^* \triangleq \min \left(\left\{ k \in \{\ell, \ell + 1, \ldots, n - 1\} : \underbrace{\frac{a + \sum_{i=1}^{n} p_i^k a_i(\tau_i^k)}{d + \sum_{i=1}^{n} p_i^k d_i(\tau_i^k)} > \frac{a_{k+1}(\tau_{k+1}^-)}{d_{k+1}(\tau_{k+1}^-)}}_{(*)} \right\} \cup \{n\} \right).$$

So k^* can be found iteratively choosing the smallest $k \in \{\ell, \ell + 1, \ldots, n - 1\}$ that satisfies the underbraced expression $(*)$. At each iteration, the processing times are

chosen using the generalized marginal value theorem. Then, the behavior $(\vec{p}^*, \vec{\tau}^*)$ with

$$\tau_j^* \triangleq \begin{cases} \tau_j^- & \text{if } j \leq \ell, \\ \tau_j^{k^*} & \text{if } j > \ell, \end{cases} \quad \text{and} \quad p_j^* \triangleq p_j^{k^*} = \begin{cases} p_j^+ & \text{if } j \leq k^*, \\ p_j^- & \text{if } j > k^* \end{cases}$$

for each type $j \in \{1, 2, \ldots, n\}$ will be optimal. So finding the behavior is equivalent to solving no more than $n+1$ generalized marginal value theorem (i.e., patch) problems where the highest profitabilities are chosen as high preference types in each iteration.

1.4 Examples: Theory and Application

Here, we examine the consequences of objective function choice on the design of decision-making behaviors for task processing. In particular, we apply methods from Section 1.3 to the example functions from Section 1.1. In Section 1.4.1, the salient theoretical differences between the resulting optimal behaviors are compared. In Section 1.4.2, the results from a mobile agent simulation are presented to compare the performance of a conventional foraging-inspired task-selection behavior with a similar behavior developed using the refined methods described in this chapter.

1.4.1 Comparison of Theoretical Results

Applying the behavioral-design algorithms from Section 1.1 yields the generalized profitabilities and MVT conditions summarized in Table 1.1. In each case, task types are assigned indices ordered by decreasing maximum generalized profitability, and any interior optimal processing time will satisfy the generalized MVT condition. Comparing each row reveals features distinctive to each associated objective function, and noting similarities reveals important structural features of classes of objective functions.

Objective	Generalized Profitability	Generalized MVT Condition
Classical	$\frac{g_i(\tau_i)-c_i(\tau_i)}{\tau_i}$	$g_i'(\tau_i) - c_i'(\tau_i) = J(\vec{p}, \vec{\tau})$
ER	$\frac{g_i(\tau_i)-c_i(\tau_i)-G^T/N}{\tau_i}$	$g_i'(\tau_i) - c_i'(\tau_i) = J_{ER}(\vec{p}, \vec{\tau})$
TDNG	$g_i(\tau_i) - c_i(\tau_i) - w\tau_i$	$g_i'(\tau_i) - c_i'(\tau_i) = w$
EE	$\frac{g_i(\tau_i)-G_g^T/N}{c_i(\tau_i)}$	$\frac{g_i'(\tau_i)}{c_i'(\tau_i)} = J_{EE}(\vec{p}, \vec{\tau})$
CDG	$g_i(\tau_i) - wc_i(\tau_i)$	$g_i'(\tau_i) = wc_i'(\tau_i)$
EoR	$\frac{g_i(\tau_i)-c_i(\tau_i)-\frac{c^s}{\lambda}}{\frac{1}{\lambda}+\tau_i}$	$g_i'(\tau_i) - c_i'(\tau_i) = \frac{g_i(\tau_i)-c_i(\tau_i)-\frac{c^s}{\lambda}}{\frac{1}{\lambda}+\tau_i}$

Table 1.1: Sample optimization results for type $i \in \{1, 2, \ldots, n\}$. The six rows correspond to the five objective functions discussed: long-term rate of gain (Classical), excess rate (ER), time-discounted net gain (TDNG), excess efficiency (EE), cost-discounted gain (CDG), and expectation of ratios (EoR). Likewise, J_{ER} and J_{EE} refer to the ER and EE objective functions, and J refers to the classical optimization objective. In all cases, an optimal behavior will have types ranked by maximum generalized profitability and will meet the generalized MVT condition.

The classical MVT condition in the first row states that the optimal processing time occurs when the instantaneous rate of gain in each patch drops to the long-term rate of gain. This feature is mirrored in generalized MVT conditions for the excess rate (ER) case in the second row as well as the excess efficiency (EE) case in the fourth row. For all three cases, the optimal behavior for one task type is coupled to the optimal behavior for another task type due to the mutual effects on the environmental average. This feature is due to the presence of decision variables in the denominator of the corresponding advantage-to-disadvantage objective functions.

Because the corresponding advantage-to-disadvantage functions do not have decision variables in their denominators, the generalized MVT condition for the time-discounted-net-gain (TDNG) case, the cost-discounted-gain (CDG) case, and the

expectation-of-ratios (EoR) case state that the optimal processing times can be determined independently of each other (i.e., processing time determination is separable). However, the optimal times are modulated by a common environmental parameter. In the TDNG and CDG cases, it is the discount factor w that represents the relative importance of gain maximization and time or cost minimization. Hence, in these two cases, the encounter rates and search cost have no impact on the optimal behavior. Thus, by fixing the discount factor, the opportunity cost of searching is also fixed and thus does not vary with the environment. However, in the EoR case, even though optimal processing times can be determined independently, they all simultaneously respond to changes in search cost or encounter rates in a qualitatively similar way as the optimal processing times in the classical case. In fact, for the single-type patch case, the EoR and classical cases match.

In Section 1.2, it was shown how in the patch case, ER optimization is identical to classical optimization if the c^s/λ search cost is augmented by the G^T/N per-task threshold. However, as shown in the second row of the table, in prey or general cases, the profitability ordering for the ER and classical cases will not match. In the patch case, higher success thresholds imply longer optimal processing times because of a greater premium on accumulating gain to reach the threshold. Similarly, for the general ER case, higher success thresholds lead to a shift in profitability orderings toward task types with higher gain. For example, classical long-term rate maximization does not differentiate between two task types with $(g_1(\tau_1), \tau_1) = (\$5, 5\,\mathrm{s})$ and $(g_2(\tau_2), \tau_2) = (\$25, 25\,\mathrm{s})$. However, when given a threshold of $G^T = \$10$ over $N = 1$ tasks, ER maximization properly prefers the latter task type that is guaranteed to

reach the $G^T = \$10$ threshold. Maximization of the EE objective has a similar feature; as the gross threshold per task G_g^T/N ratio increases, task types with greater gross gain are preferred more.

The invariance of profitability ordering is a key result of classical OFT. Although the risk-sensitive foraging model of Stephens and Charnov [111] that is applied to an autonomous vehicle problem by Andrews et al. [9] does predict that time-limited foragers facing success thresholds will tend to generalize and include task types that would otherwise be excluded by a rate maximizer, it does not predict that foragers should ever change specializations. However, the ER maximization analysis above suggests that task types that a task-processing agent would specialize on at low thresholds may be excluded entirely from very high threshold cases. A similar preference reversal is also predicted by a stochastic dynamic programming analysis of foragers facing mortality (i.e., finite lifetimes) by Iwasa et al. [53]. As discussed in Section 1.1.3, the risk-sensitive models of Stephens and Charnov still make subtle assumptions about long task-processing missions with many tasks processed. Hence, the invariance of task-type ordering may be a result of the many-task long-run-time assumptions present in popular foraging models. In engineering applications where there are relatively few tasks or high success thresholds, bio-inspired task-type ordering should be evaluated carefully.

1.4.2 Simulation Results

Table 1.2 shows simulation results from for five different finite-event task-choice (i.e., prey model) strategies with each of four different net gain success thresholds. These simulations are similar to those by Andrews et al. [9] of a fixed-wing AAV

searching continuously over an area for tasks to process (e.g., targets for package deposit, objects to collect); however, they apply equally as well to other mobile vehicle scenarios. The statistics in the tables were generated from 300 Monte Carlo samples for each of the 5×4 cases. The three rows that correspond to each gain threshold G^T show the mean and standard error of the mean (SEM) for total net gain and total time accumulated in each run as well as the percentage of runs where the total gain met or exceeded the success threshold. Each run terminated immediately after the simulated agent completed exactly $N = 300$ tasks. The particular numerical details of the simulation (e.g., encounter rates, gains, times) are given in the caption of the table. Because the simulation represents a task-choice problem (i.e., lumped tasks where each task type has fixed mean processing time and net gain), the average net gain $(g_i(\tau_i) - c_i(\tau_i))$ for each task type i has been abbreviated g_i.

Along with the five strategies used to generate Table 1.2, some additional trivial strategies that are relatively simple to analyze can be used a benchmarks. For example:

(a) An agent seeking to achieve its G^T success threshold in its N runs could wait to accept only tasks of the type 4 because that type has the highest average net gain $g_4 = 100$. The average searching time for each of these N tasks is $1/\lambda_4 = 10$ time units for each there is a c^s search cost per unit time, and so the average total gain after $N = 300$ tasks is $(N)(g_4 - c^s/\lambda_4) = (300)(100 - 0.1/0.1) = 29700$, and the average total time is $(N)(\tau_4 + 1/\lambda_4) = (300)(110 + 1/0.1) = 36000$ time units. This strategy meets each of the four G^T thresholds given, but each mission is much longer. In particular, despite having an average mission time

$N = 300$ tasks per mission, 100 Monte Carlo samples						
$G^T = 6000$		Take all	CR	ER	eCR	eER
	ΣG:	16555 ± 35	10954 ± 17	20520 ± 24	11172 ± 113	16534 ± 46
	$@G^T$:	100%	100%	100%	100%	100%
	ΣT:	11107 ± 42	4399 ± 9	9242 ± 13	4541 ± 55	9855 ± 32
$G^T = 9000$		Take all	CR	ER	eCR	eER
	ΣG:	16565 ± 30	10946 ± 16	20473 ± 25	11218 ± 128	18119 ± 38
	$@G^T$:	100%	100%	100%	98%	100%
	ΣT:	11119 ± 42	4391 ± 8	9227 ± 13	4567 ± 63	11668 ± 43
$G^T = 13500$		Take all	CR	ER	eCR	eER
	ΣG:	16642 ± 33	10958 ± 16	25153 ± 11	11270 ± 103	18647 ± 44
	$\mathbf{@G^T}$**:**	**100%**	**0%**	**100%**	**5%**	**100%**
	ΣT:	11158 ± 38	4393 ± 8	15645 ± 42	4586 ± 50	12779 ± 46
$G^T = 16500$		Take all	CR	ER	eCR	eER
	ΣG:	16546 ± 34	10993 ± 16	25141 ± 14	10965 ± 91	18796 ± 39
	$\mathbf{@G^T}$**:**	**55%**	**0%**	**100%**	**0%**	**100%**
	ΣT:	11092 ± 40	4421 ± 8	15605 ± 53	4440 ± 43	13120 ± 44

Table 1.2: Simulation results for prey-model-inspired agent simulation. Statistics are generated by taking 100 Monte Carlo samples of a mobile agent with a mission that ends after processing $N = 300$ tasks. Each agent faces an environment with with a search cost rate $c^s = 0.1$ value currency per unit time and five prey-model task types described by the 3-tuples $(\lambda_1, g_1, \tau_1) = (0.5, 30, 10)$, $(\lambda_2, g_2, \tau_2) = (0.25, 50, 20)$, $(\lambda_3, g_3, \tau_3) = (0.4, 80, 35)$, $(\lambda_4, g_4, \tau_4) = (0.1, 100, 110)$, $(\lambda_5, g_5, \tau_5) = (0.8, 55, 50)$ of encounter rate (per unit time), average net gain (value currency), and average process time (unit time). Five different task-choice scenarios are tested: the "Take all" strategy processes all encountered tasks; the classical rate (CR) strategy uses the standard prey model from classical OFT; the excess rate (ER) strategy uses a prey model based on ER maximization; the estimated classical rate (eCR) strategy uses a simple heuristic described by Pavlic and Passino [82] that converges to the prey model result; the estimated excess rate (eER) uses a modified form of the eCR heuristic applied to ER maximization. The four scenarios shown differ in their success threshold G^T. Each ΣG row gives the sample mean and standard error of the mean (SEM) for the total accumulated gain for each of the five different strategies in each of the four different scenarios. Similarly, the ΣT rows give the sample mean and SEM for total time, and the $@G^T$ rows give the proportion of runs that met or exceeded the corresponding success threshold G^T. Particularly notable $@G^T$ rows have been emphasized in bold.

of more than double the average mission time of the excess rate (ER) strategy for the $G^T = 16500$ case, it returns less than 20% more value.

(b) An agent could wait to accept only tasks of type 1 because that type has the highest profitability (i.e., net-gain–processing-time ratio). For $N = 300$ tasks, the average total gain is then 8940, and the average total time is 3600 time units. Despite this strategies high total-gain–total-time ratio, it completes the $N = 300$ tasks so quickly that it does not meet three of the example G^T thresholds from Table 1.2.

(c) An agent could wait to accept only tasks of type 3 because that type has the highest ER profitability for the $G^T = 16500$ and $N = 300$ case (i.e., $\arg\max_i (g_i - G^T/N)/\tau_i = 3$). In this case, the average total gain is then 23925, and the average total time is 11250 time units. This simple strategy achieves all four success thresholds in less than a third of the time required for the strategy in item (a) that also is uniformly successful.

Both of the single-type strategies in items (a) and (c) are successful, but they depend upon a low search cost rate c^s and a high encounter rate for their preferred task type. If the environment is relatively sparse in tasks of the desired type, the agent will engage primarily in costly searching as it ignores encounters with other types that may be more frequent. A better strategy is to balance the benefits of waiting for more profitable types with the benefits of reducing costly search time. Additionally, reducing the time of missions allows mobile agents to be re-deployed more quickly thus increasing the value returned overall.

Hence, the strategies in Table 1.2 represent different methods of prioritizing all task types to achieve success thresholds to avoid pitfalls of the single-type case.

- The take-all strategy is provided as a multiple-type benchmark. An agent following the take-all strategy does not discriminate; the agent processes every task encounter and the mission ends after exactly N encounters. As this strategy does not depend upon the success threshold G^T, its performance does not vary across different G^T selections. Consequently, for $G^T = 16500$, the strategy does worse than others that avoid low-gain tasks and instead search longer for high-gain tasks.

- The classical rate (CR) strategy uses the classic prey model algorithm for task-type choice. In this simulation, the strategy uses *a priori* knowledge of the encounter rate λ_i and the profitability g_i/τ_i of each task type i to group task types into take-all and take-none sets. The encounter rates could also be estimated as in Andrews et al. [9]; however, this idealized case is presented here for comparison to the estimated classical rate (eCR) strategy described below. Because the CR strategy is based on a rate-maximization assumption, the CR strategy has a very high total-gain–total-time ratio. Hence, for missions limited by time as opposed to number of tasks, it would likely return relatively high gain. However, when task-processing opportunities are limited, the strategy gives too much priority to task types with low processing times.

- The excess rate (ER) strategy uses the generalized prey model algorithm for task-type choice. That is, it is identical to the CR strategy except that the

realized net gain from each task type i is $g_i - G^T/N$. Consequently, its task-choice priorities vary with G^T. Thus, moving from $G^T = 9000$ to $G^T = 13500$ causes a shift in task-choice priorities that leads to a behavior mode that has a longer total mission time but also returns a higher total gain. As with the CR strategy, the ER simulation here is performed with *a priori* knowledge of encounter rates to compare its performance with the estimated ecess rate (eER) strategy described below.

- The estimated classical rate (eCR) strategy uses the simple behavioral heuristic described by Pavlic and Passino [82] to make process–ignore decisions. The heuristic makes no use of encounter rates. Instead, it compares the recognized profitability to the present total-gain–total-time ratio in order to determine whether an encountered task should be processed. The eCR strategy has similar performance as the CR strategy.

- The estimate excess rate (eER) strategy modifies the eCR behavioral heuristic to match the ER maximization case. Consequently, its performance follows behind the performance of the ER strategy.

Thus, the ER and eER strategies show that simple strategies exist that adapt to different mission success thresholds by waiting longer for high-gain tasks without depending on maximally long mission times. Moreover, the intuitive nature and simple implementation of these strategies is ported from classical OFT through the generalized framework described in this chapter.

1.5 Conclusions

In this chapter, we summarize several applications of foraging-inspired decision making in robotics (e.g., autonomous air vehicles) and computer science (e.g., resource allocation, web design), and we demonstrate that while the resulting algorithms are intuitive and simple to implement, the OFT optimization objectives themselves may not match engineering problems. We then introduce a single advantage-to-disadvantage optimization objective that generalizes several of the existing objectives used in OFT, and we also give four new models of finite-run-time optimality and show how each of them are special cases of advantage-to-disadvantage optimization. Each finite-run-time objective function includes a success threshold that mixes elements of classical rate maximization with shortfall minimization (i.e., risk sensitivity). Additionally, these four models provide optimization frameworks for the design of task-processing agents that can only engage in a finite number of tasks (e.g., a vehicle that can only deliver a finite number of packages to a practically infinite number of possible targets). As we show in simulation, the general framework allows for the design decision-making algorithms with similar attractive structures as OFT-inspired algorithms but better performance in engineering applications.

We also show how a generic optimization framework provides a substrate on which different optimal task-processing behaviors can be compared. For example, our analysis shows a relationship between rate and efficiency maximization, two approaches that are usually viewed in opposition to each other. Additionally, our analysis shows

how the introduction of success thresholds challenges the invariance of task-type preference ordering, which is a key result of classical optimal foraging theory. These comparisons reveal which key features of different optimization metrics can lead to vastly different behaviors in application.

Most applications of foraging theory to engineering focus on problems amenable to optimal prey (i.e., task-type) choice or patch residence time (i.e., task-processing time) solutions. However, Pavlic and Passino [81] apply foraging behaviors described by Gendron and Staddon [35] to a fixed-wing AAV that may also choose its search speed. In this case, the speed of the vehicle affects its detection accuracy. Increased speed increases the encounter rate with task types that are easy to detect but decreases the encounter rate with task types that are difficult to detect, and so predicting the optimal combination of task-type choice and search speed is nontrivial. A further complication is that increased speed can have increased costs (e.g., in fuel or calories). Pavlic and Passino are able to extend the methods of Gendron and Staddon from two task types to an arbitrary number of task types, but it comes at the cost of a simplistic model of detection accuracy. However, the optimization objective is an advantage-to-disadvantage function with an additional decision variable representing search speed. Extending the methods described here to handle this case is a valuable future direction that should provide more insights into complex task-processing behavior (e.g., when more realistic models of detection accuracy are included).

Chapter 2: When Rate Maximization Is Impulsive

As discussed by Schoener [103] and Pyke et al. [92], an animal must make choices that put upward pressure on both the number of prey encountered in its finite lifetime and the gain (e.g., calories or another surrogate for Darwinian fitness) returned from each prey. To attain maximum gain in minimum time, optimal foraging theory (OFT) predicts that natural selection will favor behaviors that maximize the long-term rate of gain [112]. However, in operant binary-choice experiments in the laboratory [e.g., 1, 15, 20, 41, 62, 98, 104, 107, 110], animals will often make impulsive choices that favor short handling times regardless of foraging gain. Conversely, as reviewed by Giraldeau and Caraco [38, pp. 155–167], in experiments that do not force animals to make binary-choice decisions, animals not only maximize their rate of gain but also respond to changes in the environment by dynamically adjusting their behavior to maintain maximal long-term rate of gain. That is, natural selection has gone so far as to bestow on animals the ability to calculate rate-maximizing behaviors in real time. In this chapter, we propose a simple behavioral strategy consistent with real-time rate maximization in nature and show how it appears to be impulsive in laboratory binary-choice experiments. Additionally, we show how rate maximization in the laboratory can be restored with appropriate pre-experiment treatment. If our

hypothesis is shown to be empirically robust, then impulsiveness may be a behavioral nuance that is immune to the effects of natural selection.

2.1 Background

2.1.1 Impulsiveness Without Discounting

As reviewed by Ainslie [2], early theories of impulsiveness are based on the assumption of temporal discounting. Animals are assumed to discount future rewards by some decreasing function of the time until the reward. As long as the discounting function is sufficiently concave, a smaller-sooner reward can have greater value than a larger-later reward. However, Stephens [109] argues that realistic discount rates will be too shallow to impact laboratory experiments. In particular, animals with relatively long lifetimes may value rewards today more than rewards tomorrow, but a difference in delay of less than a minute should not be enough to cause a preference reversal. Additionally, Henly et al. [46] show that likelihood of interruptions, which is often used to justify the discounting hypothesis, has negligible impact on impulsiveness observed in the laboratory.

To explain impulsiveness without discounting, recent attention focuses on how laboratory methods may artificially bias subjects toward impulsive behaviors. Arguing that impulsiveness is the result of an informational constraint, Stephens and Anderson [110] and Stephens et al. [114] show how a simple rule leads to impulsive decisions in the typical binary-choice schedule and rate-maximizing decisions in a sequential-choice schedule, but evidence from Stephens and McLinn [113] suggests that animals do not use the same choice rule in both contexts. In a different experiment, pigeons whose attention is trained on a larger-later option lose their impulsive tendencies [104],

which leads Monterosso and Ainslie [65] to suggest that impulsiveness may be the result of attention and not deliberate choice. To investigate state-dependent effects, Houston and McNamara [50] use a dynamic programming model to show that an impulsive strategy minimizes probability of starvation when an animal is in a low-energy state. This result agrees with studies [e.g., 20, 107] that show that animals are more likely to prefer smaller-sooner alternatives when they are subjected to greater food deprivation. Because food deprivation accompanies conventional operant methodology, this result may explain all observed laboratory impulsiveness; however, the model of Houston and McNamara is based on a sequential-choice assumption that is rarely met in the mutually exclusive binary-choice experiments used in most tests of impulsiveness. Here, we describe a simple behavioral mechanism that leads to impulsiveness under typical operant conditioning and rate maximization otherwise. As this single mechanism is highly influenced by both the subject's attention and energetic state, it synthesizes ideas from the three different approaches to explaining impulsiveness without discounting.

2.1.2 A Graph of the Prey Model

A detailed discussion of classical OFT is given by Stephens and Krebs [112]. Here, we summarize a central OFT result and recreate a useful graphical interpretation first presented by Charnov [25]. Consider a forager that searches through its environment for $n \in \mathbb{N}$ types of prey. Encounters with prey of type $i \in \{1, 2, \ldots, n\}$ come from an independent Poisson process with rate $\lambda_i > 0$, and those type-i prey that are chosen for processing return an average g_i gain (e.g., calories) to the forager after an average h_i handling time. When not handling prey, the forager pays a cost c per unit

search time (i.e., c is a cost rate). Assuming that natural selection favors foraging behaviors that maximize lifetime gain, the optimal behavior should trade large reward per encounter for increased number of encounters (i.e., less time per encounter), and so an optimal behavior maximizes the long-term average rate

$$R \triangleq \frac{-c + \sum_{i=1}^{n} \lambda_i p_i g_i}{1 + \sum_{i=1}^{n} \lambda_i p_i h_i} \tag{2.1}$$

where $p_i \in [0, 1]$ represents the probability that the forager processes an encounter with a prey of type i. Each behavior is represented by a particular collection $(p_1, p_2, \ldots, p_n)$ representing the diet of the forager.

The central result of this so-called contingency model [25, 91] or prey model [112] is that if prey types are ordered so that

$$\frac{g_1}{h_1} > \frac{g_2}{h_2} > \cdots > \frac{g_{n-1}}{h_{n-1}} > \frac{g_n}{h_n},$$

then an optimal behavior is to process all encounters with prey of type $m \leq k^*$ and to ignore all encounters with prey of type $m > k^*$ where $k^* \in \{0, 1, 2, \ldots, n\}$ is the smallest k that satisfies

$$\frac{-c + \sum_{i=1}^{k} \lambda_i g_i}{1 + \sum_{i=1}^{k} \lambda_i h_i} > \frac{g_{k+1}}{h_{k+1}}.$$

Consequently,

$$\overbrace{\frac{g_1}{h_1} > \frac{g_2}{h_2} > \cdots > \frac{g_{k^*}}{h_{k^*}}}^{\text{Processed types}} > \left(\overbrace{R^* \triangleq \frac{-c + \sum_{i=1}^{k^*} \lambda_i g_i}{1 + \sum_{i=1}^{k^*} \lambda_i h_i}}^{\text{Optimal rate}} \right) > \overbrace{\frac{g_{k^*+1}}{h_{k^*+1}} > \cdots > \frac{g_{n-1}}{h_{n-1}} > \frac{g_n}{h_n}}^{\text{Ignored types}} \tag{2.2}$$

where the optimal rate R^* partitions the list of profitabilities (i.e., the ratio g_i/h_i for each type i) into an acceptable set and an unacceptable set. This result, which

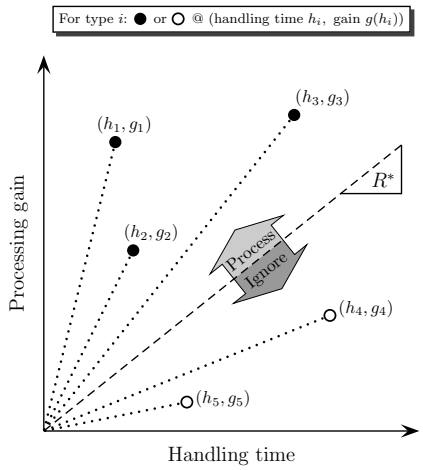


Figure 2.1: Graphical summary of prey model result. For a prey type $i \in \{1, 2, 3, 4, 5\}$, the average handling time h_i and average gain g_i is plotted as a dot. The maximum long-term rate of gain R^* is the slope of the dashed line which separates the processed types, 1, 2, and 3, from the ignored types, 4 and 5. The profitability of each type is the slope of the dotted line connecting the origin to its (gain, time)-coordinate.

is known as the zero–one rule [112], is shown graphically in Figure 2.1. For each type, a solid dot is plotted with the type's average handling time as its abscissa and the type's average processing gain as its ordinate, and so the slope of each dotted line corresponds to the profitability of the corresponding type. If the optimal long-term rate of gain is R^*, then the dashed line with slope R^* divides the plane into an acceptable upper region and an unacceptable lower region. In this example, the three steep lines corresponding to types 1, 2, and 3 are contained in the acceptable region, and the shallow lines corresponding to types 4 and 5 are contained in the unacceptable region, and so $k^* = 3$ in Equation (2.2). As encounter rates of acceptable types increase or decrease, the dashed line rotates to exclude or include more types.

2.1.3 Justification for Adaptive Rate-maximization Model

Optimal foraging theory describes the ultimate behaviors favored by natural selection [112]. In the example in Figure 2.1, a heritable preference for only the three highest-profitability types will readily spread through future generations of the population. If an environmental disaster causes encounter rates to fall sharply, then more-inclusive heritable preferences present in the background population will successfully invade and dominate future generations of the population. So optimal foraging theory posits that rate-maximizing behaviors will be the adaptive outcome of the gradual process of natural selection. However, Giraldeau and Livoreil [39] show that birds in the laboratory respond to a sharp change in the environment with a new behavior that matches the long-term rate-maximizing behavior predicted by OFT. This response is consistent with real-time calculation of the optimal rate-maximizing behavior. Moreover, in a survey by Sih and Christensen [105] of 74 recent foraging studies, 22 studies are given that show animals flexibly adapting to their experimental scenarios in a way that is at least qualitatively consistent with optimal foraging theory. Thus, OFT may also be viewed as describing the proximate outcome of dynamic behaviors that adapt to changing environments.

2.1.4 Other Ostensible Violations of Rate Maximization

Impulsiveness describes only a subset of behaviors that appear to violate rate maximization. In the recent survey by Sih and Christensen [105] of 60 foraging studies originally surveyed by Stephens and Krebs [112] as well as 74 more recent studies, optimal foraging theory is shown to have mixed success in describing animal behavior. Sih and Christensen argue that the prey model is a poor descriptor of the

foraging behaviors of animals with mobile prey. However, even in herbivores [48, 121] and molluscivores [94, 119] that have essentially immobile prey, optimal foraging theory fails to predict the observed diet preferences. In these cases, the digestive rate model [DRM; 47, 49, 121] is shown to better predict animal diet preference, but the DRM itself is the result of generalizing the prey model (i.e., rate maximization) to include the effect of digestive constraints.

The DRM of Hirakawa [47] is a correction of related foraging constraint models from Pulliam [91], Stephens and Krebs [112], and Verlinden and Wiley [121]. It maximizes the same rate expression from Equation (2.1); however, it also introduces the material intake rate

$$X \triangleq \frac{\sum_{i=1}^{n} \lambda_i p_i k_i}{1 + \sum_{i=1}^{n} \lambda_i p_i h_i} \tag{2.3}$$

where k_i is the average quantity (e.g., mass) of ingested material from prey of type i. Depending on whether the material is a nutrient or a toxin, the intake rate X will have a lower or upper bound, respectively. In the molluscivore shorebird examples described by van Gils et al. [119] and Quaintenne et al. [94], X represents the intake rate of ballast material (e.g., shells of bivalves and snails). When the gizzard of one of these cropping birds is full, the bird must pause its foraging behavior until its gizzard has emptied enough for it to continue. Thus, these shorebirds have a maximum intake rate of ballast material that is not negligible. A key result of the DRM is that prey types should not be ordered by profitability as in Equation (2.2); instead, they should be ordered by so-called (digestive) quality which is defined by g_i/k_i for each type i. The DRM gives an algorithm to find the prey type whose quality partitions the group into take-always and take-never groups. However, only a

fraction of encounters with prey of the partitioning type should be ingested. Hence, even though the DRM is the result of rate maximization under a constraint, its result both re-orders prey preferences and appears to violate the zero–one rule. Although our main focus is to present a simple model of foraging that explains impulsiveness in the laboratory, we shall also show how our simplified model has similar qualitative features as the DRM. In particular, our DRM-consistent behavioral model reconciles the differences between rate maximization and DRM using an internal handling time approach, which is similar to the ecological–physiological hybrid method described by Whelan and Brown [125].

2.2 Model

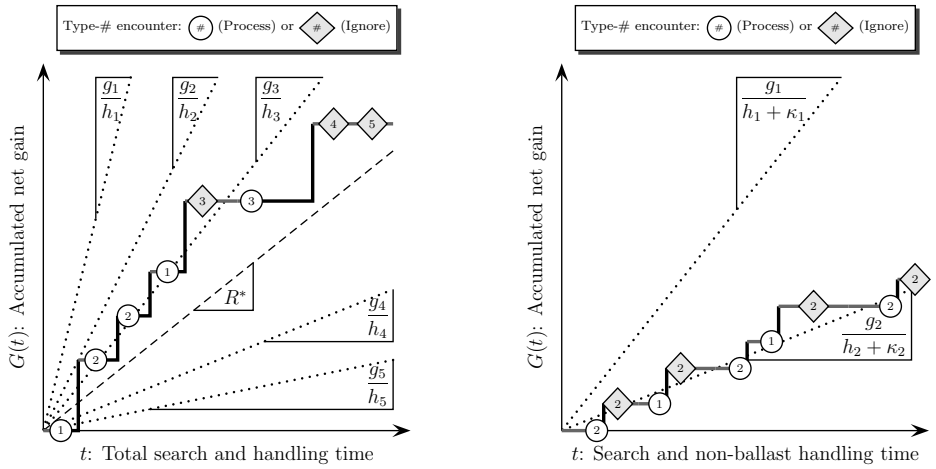
2.2.1 State-based Real-time Adaptive Rate-maximization

Using a version of the prey model of Stephens and Krebs [112] put into an engineering context, Andrews et al. [7] and Quijano et al. [97] implement a heuristic algorithm that achieves real-time rate maximization through computation of an estimated optimal behavior. In particular, on encounter with a prey of type i, the algorithm estimates the encounter rate λ_i and then updates its exclusion rule based on Equation (2.2) assuming that the present λ_i estimate is correct. That is, the algorithm calculates the optimal rate R^* based on the estimated rates and then updates its prey model exclusion policy based on that estimated optimal rate.

Instead of estimating each of the n encounter rates to find the critical k^*, our algorithm keeps a running estimate of the long-term rate of gain R by dividing total accumulated gain by total time. Then, on each encounter, the forager compares the encountered prey's profitability with the current R estimate and rejects the prey if

its profitability is lower than R. By accepting prey with super-R profitability and rejecting those with sub-R profitability, processed prey only cause the estimated R to rise. Between processed encounters, the R estimate falls due to the effect of increased time without increased gain. Over time, the oscillating R estimate converges on the optimal R^*; hence, the process–ignore policy converges to the prey model described by Equation (2.2). This method is essentially an application of the patch model of Stephens and Krebs [112] applied to a forager whose processing-length decision is limited to leaving immediately or staying for a patch's entirety. Not only does this strategy have less computational complexity than those based on encounter-rate estimation, but it has an intuitive state-based interpretation; when the forager's present energy stores are low (i.e., when R is low due to low accumulated gain or high total running time), the forager becomes more inclusive. We caution that this is a loaded interpretation. Among other things, it neglects the effects of non-foraging activities (e.g., reproduction) that both depress present energy stores and take time. However, although a forager's instantaneous energy stores may be a poor estimator of R, low-frequency trends in the energy store signal should correlate well with R. For example, it may be maladaptive for momentary hunger to drive foraging decisions, but it persistent hunger is an important signal that the forager should generalize more. Furthermore, we show later that a small adjustment to this model accounts for optimal foraging under digestive rate constraints. In that case, a stronger connection exists between energy state and R (e.g., high R may indicate that less bulky foods should be preferred).

Our algorithm is depicted in Figure 2.2(a). The solid line represents the trajectory of the gain of the forager over time. Each dotted line has a slope matching one of the



(a) Classical model.

(b) DRM case.

Figure 2.2: Adaptive long-term maximizer trajectory. The *solid line* represents a forager's accumulated gain over time. Each encounter with a new prey is shown with as a *circle* or a *diamond* surrounding the prey's type. Encounters shown with a *circle* are chosen for processing and encounters shown with a *diamond* are ignored. Hence, jumps in the gain trajectory occur at circled encounters. Encounters that occur below the prey type's profitability slope are processed; otherwise, encounters are ignored. In (a), profitability is defined in the classical way as g_i/h_i for each type i, and the gain trajectory converges on a line with slope R^*, which is the maximum long-term rate of gain. In (b), profitability is defined as $g_i/(h_i + \kappa_i)$ (or, alternatively, g_i/κ_i with similar results) where κ_i has units of time and is proportional to k_i, and the gain trajectory converges on the profitability line of the partially preferred prey type.

n profitabilities. The estimated rate R at each encounter is the slope of an imaginary line drawn from the origin to the gain trajectory at that time. If the profitability of an encountered prey is steeper than the present R estimate, then the prey is processed, and the current R estimate increases; otherwise, the prey is ignored. Encounters are shown in this depiction as points on the gain trajectory. Those encounters that are processed are shown as circles, and encounters that are ignored are shown as diamonds. So an encounter is processed if its graph location, which corresponds to the estimated R at the time of the encounter, is below the dotted profitability line of its task type. For this example trajectory, the first four encounters are chosen for processing because they fall below their corresponding profitability lines. The fifth encounter is with a prey of type 3, and it is ignored because it falls above the third profitability line. Even though the sixth encounter is also with a prey of type 3, it is chosen for processing because enough time has passed to depress the estimated rate R. As time continues, the estimated rate R eventually converges to the maximum long-term rate of gain R^*, which is shown as the slope of the dashed line.

2.2.2 State-based Real-time Adaptive Model Consistent with DRM

With only a small modification, the adaptive rate-maximization model can be made to be consistent with the qualitative predictions of the DRM. In particular, for each type i, let ballast time κ_i be a parameter with units of time that is proportional to the quantity k_i. For example, κ_i may be viewed as the average amount of time prey of type i spend in the gizzard of a molluscivore shorebird. If the profitability g_i/h_i for each type i used by the forager to determine whether a particular encounter should be accepted or rejected is replaced by the $g_i/(h_i + \kappa_i)$ or simply g_i/κ_i, then

the gain trajectory will follow (i.e., continually switch across) one of the profitability lines. The prey type corresponding to the switching line will also represent the prey that is only partially preferred. The other prey types will be preferred via the zero–one rule. This modified process is depicted in Figure 2.2(b). Hence, the introduction of internal handling time κ_i mitigates the need for a hard digestive rate constraint, which is similar to the ecological–physiological hybrid model described by Whelan and Brown [125] that also models internal handling time. Coopting their example, we notice that in our model, as R increases (e.g., stomach fills), calorie-dense foods (e.g., cookies) may be preferable over high-bulk foods (e.g., salad) even though profitabilities may be similar, which matches conventional human experience.

Although this algorithm may not accurately described the proximate mechanisms for decision making in foragers with digestive constraints, it may provide helpful intuition for visualizing DRM results. Additionally, if the ballast-time proportionality constant (i.e., κ_i/k_i for each type i) is identical across all types and digestive profitability is defined as $g_i/(h_i+\kappa_i)$ for each type i, then this model behaves as a classical rate maximizer when h_i is much larger than κ_i and as a digestively constrained forager when κ_i is much larger than h_i. Hence, the previous model is a limiting case of this model; as ballast times become vanishingly small, the acceptance proportion of the partially accepted prey type approaches unity or nullity. Thus, this simple behavioral model reconciles differences between classical optimal foraging theory models and digestive constraint models. Moreover, this model justifies the otherwise questionable use of energy availability (e.g., hunger) as an estimator of R. In the case of pure rate maximization, an animal in a should only use its energy state as an estimator of a low R if its energy stores are persistently low. However, when using digestive

profitability (i.e., when including ballast effects), using energy stores as an estimator of R implicitly accounts for digestive constraints; less bulky foods should be preferred when R is high.

2.3 Results

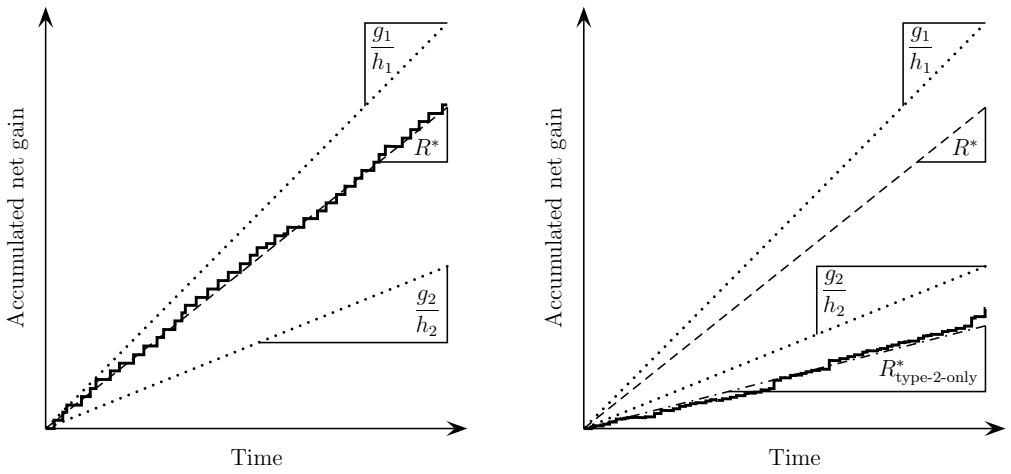
The most prevalent foraging models used in OFT make the simplifying assumption that encounters with more than one prey at a time occur with zero probability [23]. This assumption does not lack realism. A forager that searches for sparse stationary morsels of food may rarely encounter items simultaneously, and even when it does, its choices need not be mutually exclusive. If such an environment is the source of natural selection, then it makes sense that the animal's behavior may be poorly adapted for the mutually exclusive, simultaneous encounter, binary-choice experiments of the conventional operant laboratory. Here, we show how our real-time algorithm is impulsive under traditional laboratory conditions and maximizes long-term rate of gain under natural conditions. We also show how our DRM-inspired real-time algorithm results in foraging preferences consistent with results from the DRM. All simulations were implemented using MATLAB ((R14SP3) The MathWorks, Natick, MA, USA), and all MATLAB source code is available as an electronic supplement (see S1 Appendix).

2.3.1 Simulation: Simultaneous Encounters Lead to Suboptimality

A simulated trajectory for our algorithm over 100-s is shown in Figure 2.3(a) where $n = 2$, $(\lambda_1, g_1, h_1) = (2\text{-s}^{-1}, 5, 2\text{-s})$, $(\lambda_2, g_2, h_2) = (5\text{-s}^{-1}, 1, 1\text{-s})$, and $c = 0.1\text{-s}^{-1}$. So type 1 has the highest profitability and type 2 has the lowest average handling time. For each type, the gain and handling time have nonzero variance and positive

correlation. Additionally, on an encounter with a single prey, there is a 5% probability that the forager will make an algorithmic mistake. In this simulation, encounters with each type come from separate and independent Poisson processes, and so simultaneous encounters occur with zero probability. However, in the rare case of a simultaneous encounter, this forager arbitrarily attends to the prey with the lower average handling time first with a 95% probability. This assumption of biased attention is motivated by the training experiment of Siegel and Rachlin [104] and the suggestion by Monterosso and Ainslie [65] that impulsiveness may be the result of attention span and not deliberate choice. In this example, the steady-state foraging behavior accepts all high-profitability encounters and rejects all low-profitability encounters, and the gain trajectory behaves like a line with R^* slope. Because this behavior eventually achieves the maximum long-term rate of gain, it is eligible for proliferation by natural selection.

In Figure 2.3(b), a simulated trajectory for the same behavior that maximizes its long-term rate in Figure 2.3(a) is shown when facing simultaneous and mutually exclusive encounters at rate $\lambda = 2\text{-s}^{-1}$. Because it attends to the prey with the lower average handling time first and the experimenter removes the other prey on each encounter, the trajectory achieves a suboptimal long-term rate $R^*_{\text{type-2-only}}$, and the experimenter concludes that the forager is not a rate maximizer. However, the rate $R^*_{\text{type-2-only}}$ is the maximum rate in an environment with only the low-profitability type. Under these experimental conditions, the estimated rate is initially very low, and so the low-profitability prey are initially acceptable for processing. Because the forager attends to these prey first and the experimenter immediately removes the other prey, the new estimated rate R remains too low to exclude the next encountered low-profitability type. Hence, the forager never experiences high-profitability prey gains



(a) Behavior facing merged Poisson encounters.

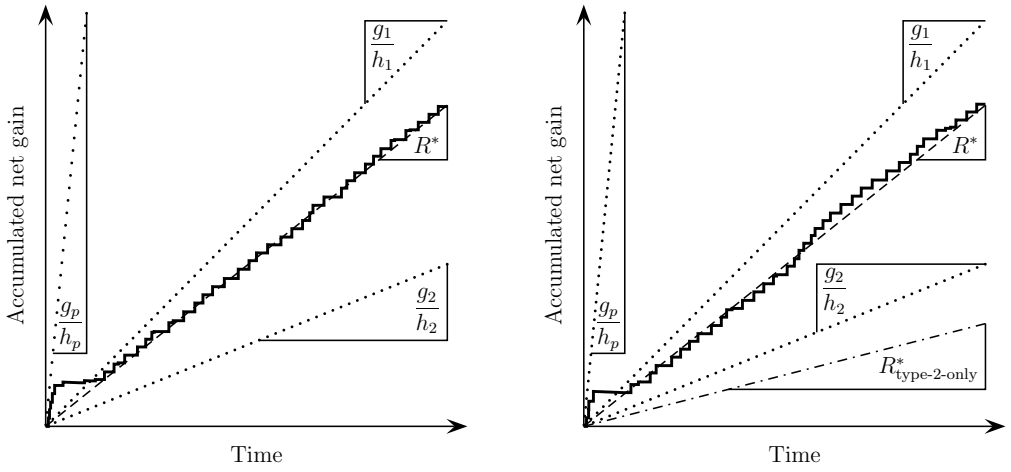
(b) Behavior facing simultaneous encounters.

Figure 2.3: Simulation of an adaptive and impulsive foraging behavior facing different encounter processes. In the 100-s foraging bout, the forager encounters high-profitability prey of type 1 ($g_1 = 5$, $h_1 = 2$-s) and low-handling-time prey of type 2 ($g_2 = 1$, $h_2 = 1$-s). During search, it pays cost $c = 0.1$-s^{-1}. On an encounter with a prey, it is chosen for processing if its profitability is greater than the quotient of the accumulated gain and total lifetime, and the forager makes a mistake in this calculation with 5% probability. On simultaneous encounters, the forager attends to the prey with lowest handling time first with 95% probability (i.e., its attention is impulsive). In (a), encounters come from a merged Poisson process with rates $\lambda_1 = 2$-s^{-1} and $\lambda_2 = 5$-s^{-1}, and the behavior is optimal. In (b), encounters are simultaneous and mutually exclusive with rate $\lambda = 2$-s^{-1}, and the same behavior is suboptimal. In particular, the performance in (b) is optimal in an environment of only type 2 prey.

and is destined for poor performance. If these operant laboratory conditions (i.e., starvation followed by sequences of mutually exclusive binary-choice trials) are unlike those in nature, this short-time attentive foraging strategy may still maximize its long-term rate of gain in nature and thus will still be present in the animal population.

2.3.2 Simulation: Pre-experiment Feeding Restores Optimality

This hypothesis can be tested by preloading the forager's gain trajectory immediately before the experiment (e.g., to alleviate any persistent hunger signals that drive the forager to be maximally inclusive). The trajectory in Figure 2.4(a) is the result of a second simulation with merged Poisson encounters; however, during the first two seconds of the experiment, the forager faces very frequent encounters with prey of a very high profitability g_p/h_p. As a consequence, the trajectory rises very quickly and then plateaus off until the estimated rate of gain falls below the profitability of type 1. After that time, the forager follows a typical rate-maximizing trajectory. Similarly, the trajectory in Figure 2.4(b) is the result of a second simulation with simultaneous and mutually exclusive encounters, but the forager is given the same initial two-second feeding period in order to bias its behavior into a rate-maximizing mode. As with the previous case, the gain rises steeply and then plateaus until the point where the estimated gain falls below the profitability of type 1. After that time, the previously impulsive behavior follows a true rate-maximizing trajectory. That is, because the estimated rate is so high, the forager never attends to the low-profitability type; it ceases to appear impulsive. This restoration of rationality will rarely occur under conventional operant methods because those subjects are typically intentionally deprived of food outside of the experimental apparatus [e.g., 20, 41, 104, 107, 110, 113].



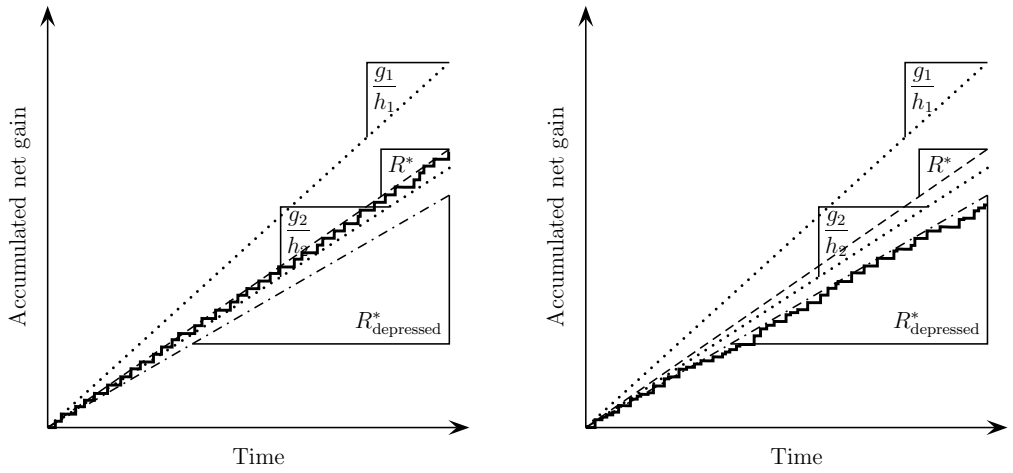
(a) Behavior facing merged Poisson encounters.

(b) Behavior facing simultaneous encounters.

Figure 2.4: Simulation of an impulsive foraging behavior after initial ad libitum feeding period. In most of the 100-s foraging bout, the forager encounters high-profitability prey of type 1 ($g_1 = 5$, $h_1 = 2$-s) and low-handling-time prey of type 2 ($g_2 = 1$, $h_2 = 1$-s). However, during the first 2-s of the bout, the forager encounters type-p prey of very high profitability and very high rate ($\lambda_p = 10\text{-s}^{-1}$, $g_p = 5$, $h_p = 0.2$). During search, it pays cost $c = 0.1\text{-s}^{-1}$. On an encounter with a prey, it is chosen for processing if its profitability is greater than the quotient of the accumulated gain and total lifetime, and the forager makes a mistake in this calculation with 5% probability. On simultaneous encounters, the forager attends to the prey with lowest handling time first with 95% probability (i.e., its attention is impulsive). In (a), encounters come from a merged Poisson process with rates $\lambda_1 = 2\text{-s}^{-1}$ and $\lambda_2 = 5\text{-s}^{-1}$, and in (b), encounters are simultaneous and mutually exclusive with rate $\lambda = 2\text{-s}^{-1}$. In both cases, the behavior is optimal.

2.3.3 Simulation: Equal-opportunity Foragers and Simultaneous Encounters

Another set of simulated simultaneous-encounter trajectories of our algorithm is shown in Figure 2.5. However, in these examples, the forager has unbiased attentiveness. That is, in the case of a simultaneous encounter with two prey, the forager attends first to one of them at random following a uniform distribution. Under merged Poisson encounters, this behavior will still maximize the forager's long-term rate of gain because simultaneous encounters are so rare. However, when encounters are artificially made by an experimenter to be simultaneous and mutually exclusive, the foraging behavior is not guaranteed to maximize the long-term rate. In particular, runs can be put into two groups characterized by Figures 2.5(a) and 2.5(b). In the former case of Figure 2.5(a), the gain accumulates initially fast enough to exclude the low-profitability type from consideration by the algorithm, and the foraging behavior achieves the optimal long-term rate of gain R^*. In the latter case of Figure 2.5(b), the equal-opportunity foraging behavior achieves a long-term rate of gain $R^*_{\text{depressed}}$ equal to the optimal long-term rate of gain when encounter rates are halved. Hence, forcing animals to make mutually exclusive decisions about repeated simultaneous encounters has the effect of scaling encounter rates by the animal's attention pattern. However, as with the impulsive behavior, a rate-maximizing result like the one shown in Figure 2.5(a) can be forced by preloading the forager with an ad libitum feeding period to ensure the initial behavior is as exclusive as possible (i.e., ensuring that the initial rate R estimate is relatively high). Again, treating animals with pre-experiment feeding is uncommon in conventional operant laboratory methods because animals are starved to enforce compliance with the experimental apparatus. Unfortunately,



(a) Rate-maximizing run.

(b) Maladaptive run.

Figure 2.5: Equal-opportunity behavior facing simultaneous encounters. During the 100-s foraging bout, the forager encounters high-profitability prey of type 1 ($g_1 = 4.5$, $h_1 = 2$-s) and low-handling-time prey of type 2 ($g_2 = 1.6$, $h_2 = 1$-s). All encounters are simultaneous and are generated by a Poission process with a 1.7-s^{-1} encounter rate. During search, the forager pays cost $c = 0.1$-s^{-1}. On an encounter with a prey, it is chosen for processing if its profitability is greater than the quotient of the accumulated gain and total lifetime, and the forager makes a mistake in this calculation with 5% probability. On simultaneous encounters, the forager randomly attends to one of the encountered prey. In the typical run shown in (a), the gain trajectory converges to the rate-maximizing optimal result. In another typical run shown in (b), the gain trajectory converges to a result congruent with rate maximization if encounter rates were halved.

it is possible that starved preferences are overly inclusive; a starved individual may eat foods that it would rarely choose willingly in its natural environment.

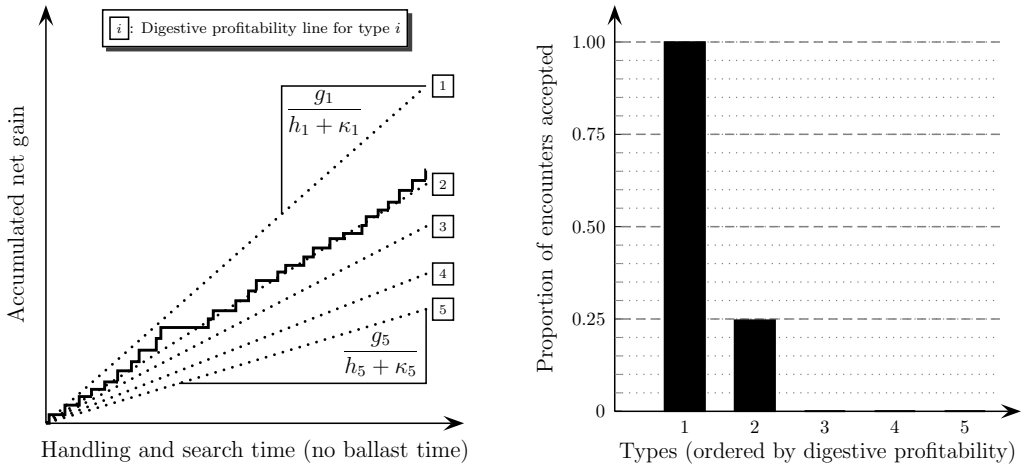
2.3.4 Simulation: DRM-inspired Rule has DRM-like Preferences

In Figure 2.6(a), the gain trajectory for a forager using the DRM-style decision algorithm is shown. The gain trajectory switches across the profitability line for type 2. Moreover, as shown in Figure 2.6(b), the prey types are partitioned into a set of always-accepted types, always-rejected types, and a single partially accepted type (i.e., type 2 in this example). Similar results can be found by defining digestive profitability as g_i/κ_i for each type i, which guarantees the same type ordering as the DRM model provided that the proportionality constant (i.e., κ_i/k_i for each type i) is identical across all types. However, when digestive profitability is defined as $g_i/(h_i + \kappa_i)$ for each type i, then this forager behaves as a traditional rate maximizer when h_i is significantly larger than κ_i.

2.4 Discussion

2.4.1 Binary-choice Impulsiveness can be Sequentially Optimal in Nature

In the laboratory, animals forced to make a choice between two mutually exclusive alternatives show an impulsive tendency for short handling time even when that behavior fails to maximize long-term rate of gain. We have shown that animals that are impulsive in binary-choice experiments can still be rate maximizers in nature. Our hypothesis is that on an encounter with a prey, the animal chooses to process the prey based on whether the prey's gain–time ratio is greater than the animal's



(a) Gain trajectory.

(b) Diet preferences.

Figure 2.6: Simulation of a DRM-inspired foraging behavior that has DRM-like results. During the 100-s foraging bout, the forager encounters prey of types 1 ($\lambda_1 = 1\text{-s}^{-1}$, $g_1 = 3$, $h_1 = 0.5\text{-s}$, $\kappa_1 = 5\text{-s}$), 2 ($\lambda_2 = 0.75\text{-s}^{-1}$, $g_2 = 5$, $h_2 = 2\text{-s}$, $\kappa_2 = 5\text{-s}$), 3 ($\lambda_3 = 0.75\text{-s}^{-1}$, $g_3 = 4$, $h_3 = 0.25\text{-s}$, $\kappa_3 = 4\text{-s}$), 4 ($\lambda_4 = 1\text{-s}^{-1}$, $g_4 = 4$, $h_4 = 0.5\text{-s}$, $\kappa_4 = 3\text{-s}$), and 5 ($\lambda_5 = 0.1\text{-s}^{-1}$, $g_5 = 5$, $h_5 = 0.1\text{-s}$, $\kappa_5 = 3\text{-s}$) according to a five-way merged Poisson process. The prey types are ordered by digestive profitability, which is defined as $g_i/(h_i + \kappa_i)$ for each type i. In this example scenario, there is no search cost (i.e., $c = 0\text{-s}^{-1}$). On simultaneous encounters (which occur with probability 0 due to the merged Poisson process), the forager attends to the prey with lowest handling time first (i.e., its attention is impulsive). The gain trajectory is shown in (a), and the proportion of encounters that are accepted is shown in (b). As shown, prey of type 1 are always accepted, prey of types 3, 4, or 5 are always rejected, and prey of type 2 are accepted on 25% of encounters with them.

present accumulated gain–total time ratio. In the rare case when multiple prey are encountered at the same time, the animal arbitrarily attends to the one with shortest handling time first. Because these cases are so rare, the animal's attention pattern has little impact on its lifetime success. However, because laboratory encounters are always simultaneous and mutually exclusive, the food-deprived animal fails to make good decisions.

To test this hypothesis, animals can be given a short ad libitum feeding period directly before testing. This feeding period should raise their gain–time ratios so that the purely impulsive choice is not a viable alternative. If this hypothesis is robust, then impulsiveness may not be a curiously irrational behavior so much as a behavioral polymorphism that is normally masked to the effects of natural selection. Additionally, this simple behavioral algorithm may provide intuition for understanding complex state-based animal behavior. An important future direction is the testing of this theory in the laboratory; however, it is consistent with previous experiments [e.g., 20, 107] that show that impulsiveness is reduced when animals have more access to food. Additionally, the R process is is a discrete-time denumerable Markov chain, and it is likely that the convergence we have demonstrated in simulation can be shown analytically via mathematical proof. To establish confidence that this algorithm is a robust rate maximizer, the stochastic convergence of R should be analytically characterized.

A notable weakness of the central hypothesis of this chapter is that it takes for granted that the animal attends first to prey with short handling times when facing simultaneous encounters. The salient feature of this assumption is that the animal's attention modulates the perceived encounter rates of encountered prey, which

is consistent with the attention-training experiment of Siegel and Rachlin [104] and the conclusion of Monterosso and Ainslie [65] that impulsiveness may be linked to attention and not choice. If the forager only sees simultaneous encounters with a low-profitability short-time prey and a high-profitability long-time prey and always attends to the short-time prey first, it is as if the high-profitability prey has a null encounter rate. Moreover, an equal-opportunity forager facing all simultaneous encounters may also appear to be overly inclusive to short-time prey. During the period of time where its initial rate estimate is low enough to consider the low-profitability short-time prey, the forager forced to make mutually exclusive choices on each simultaneous encounter will behave as if it were facing encounter rates that are halved and thus will continue to be overly inclusive. It is true that a forager that attends first to high-profitability prey on simultaneous encounters will outperform others in binary-choice experiments. However, if simultaneous encounters are rare in nature, it is unlikely that natural selection will have shaped these attention mechanisms, and a wide variation of attention schemes may be present in a background population. Furthermore, the assumption that animals exhibit their natural preferences when starved seems flawed regardless of the underlying behavioral mechanism. This chapter serves as an example of how starved preferences can be unique.

2.4.2 A Mechanism Consistent with Digestive Rate Model

We have also shown how a simple modification to the our rate-maximizing behavior leads to adaptive diet choices that are qualitatively consistent with the digestive rate model of digestively constrained rate maximizers. In particular, the mechanism automatically orders types by digestive profitability, which is a quantity similar to

digestive quality in the DRM, and results in partial preference for a single prey type that partitions the other prey types into a set of always-accepted and always-ignored types. To construct digestive profitability, we defined an internal ballast time that is likely proportional to the material quantity used in DRM constraints. Hence, digestive profitability for each type is the ratio of its average gain to the sum of its handling time and ballast time. Like the approach of Whelan and Brown [125], this model achieves qualitatively similar results as DRM without a hard digestive rate constraint. Like the short-time attentive rate-maximizing mechanism we introduced, this DRM-consistent mechanism may also be prone to maladaptive impulsiveness in the case of simultaneous encounters. However, our objective in presenting this simple DRM-consistent behavioral model is to suggest intuition for visualizing DRM results and reconciling differences between ecological and physiological schools of foraging thought. For example, our model shows clearly how a forager with high energy stores may prefer the less bulky of two items with equal profitability while a forager with low energy stores will accept both. An important future direction is to precisely define what a ballast time is in nature.

Chapter 3: The Sunk-cost Effect as an Optimal Rate-maximizing Behavior

Influential work by Schoener [103] postulates that natural selection favors behaviors on a continuum from foraging time minimization to net energetic gain maximization. Techniques developed by Pyke et al. [92] and Charnov [23] use gain-to-time rate maximization to quantitatively predict which behavior from this continuum will be favored in particular cases. These techniques have been popularized by Stephens and Krebs [112] as the *prey* and *patch* models of classical optimal foraging theory (OFT). They respectively describe which prey foragers should include in their diet and how long foragers should exploit a patch of prey, which are the two central questions of solitary foraging theory.

The rate-maximizing prey and patch models have mixed success in explaining behavioral observations in the field. The prey model accurately describes patterns of preference, and the patch model makes accurate predictions about how foraging durations should change when background parameters change, but the patch model has poor success in many cases when predicting actual foraging durations [69, 105, 112]. In a review by Nonacs [69], several examples are picked that show foraging durations tend to be longer than expected by the classical patch model. Nonacs concludes that OFT is incomplete and that optimal behavior is not described by

rate maximization. However, recent studies show how observed behaviors that are inconsistent with classical OFT are indeed rate maximizing under an adjusted foraging model. For example, shorebirds that appear to violate classical OFT are shown to be rate maximizers when explicitly modeling digestive constraints [119] and the value of information discovery [118].

Here, we show how explicitly modeling foraging costs can answer Nonacs' criticisms of the classical OFT's underpredictions of foraging duration. Additionally, we show how the same modifications lead to foraging theoretic explanations of the *sunk-cost effect* [10, 11, 56, 108], which describes behaviors that invest more time in the more costly of two otherwise equivalent resources. This effect is also known as the *Concorde fallacy* [27] because it describes an apparent logical fallacy analogous to the continued development of a supersonic jet that never returns a profit. However, by giving a foraging theoretic explanation, we show that the fallacy is actually an optimal behavior. This explanation is also consistent with observations of animals that commit longer feeding times after moving into areas where prey requires greater energy to acquire [68]. Additionally, if a forager's uncertainty in estimating patch quality is modeled as an initial average cost associated with each patch type, the extended foraging durations predicted by our augmented OFT is consistent with the "wait for good news" behaviors predicted and observed in supposed Bayesian foragers [73, 74, 118].

This chapter is organized as follows. First, in Section 3.1, we summarize the principal results from classical OFT. Next, in Section 3.2, we present a common empirical criticism of OFT and provide a mathematical response supporting OFT. In Section 3.3, we augment the traditional graphical analysis method from OFT

to include effects of search and in-patch costs. This analysis shows how explicitly including costs leads to longer patch residence times that are more consistent with observed behavior. We use similar graphical methods in Section 3.4 to provide a detailed explanation of the sunk-cost effect as an adaptive rate-maximizing behavior. Finally, in Section 3.5, we give some concluding remarks.

3.1 Classical Optimal Foraging Theory

Optimal foraging theory models a forager that faces $n \in \mathbb{N}$ types of prey. The forager encounters prey of type $i \in \{1, 2, \ldots, n\}$ according to an independent Poisson counting process with rate $\lambda_i > 0$. Each encountered prey item of type i is picked for handling with probability $p_i \in [0, 1]$, and those chosen items are exploited for $\tau_i \geq 0$ time on average. The forager pays a cost (e.g., energy) of c^s per unit search time, but it receives an average reward of $g_i(\tau_i)$ after handling an item of type i. Assuming that natural selection favors foragers that maximize their lifetime gain, behaviors should trade increased reward per encounter for increased number of encounters in a lifetime. So the optimal behavior will maximize the rate

$$\frac{-c^s + \sum_{i=1}^{n} \lambda_i p_i g_i(\tau_i)}{1 + \sum_{i=1}^{n} \lambda_i p_i \tau_i}. \tag{3.1}$$

The *prey model* treats the case where τ_i is fixed for each type i and the forager must make a binary choice to handle or ignore each encountered prey. The *patch model* treats the opposite case where $p_1 = p_2 = \cdots = p_n = 1$ (i.e., all encountered prey are exploited) and the forager must choose when to stop exploiting each prey. A central result of the patch model is the *marginal value theorem* (MVT) [23, 24], which states

that nonzero exploitation times $\tau_1^*, \tau_2^*, \ldots, \tau_n^*$ that make

$$g_i'(\tau_i^*) = \frac{-c^s + \sum_{i=1}^{n} \lambda_i g_i(\tau_i^*)}{1 + \sum_{i=1}^{n} \lambda_i \tau_i^*} \quad \text{and} \quad g_i''(\tau_i^*) < 0 \tag{3.2}$$

for each type i will correspond to an optimal patch foraging behavior. That is, behaviors that equate per-prey rate of gain with per-lifetime rate of gain will appropriately balance present reward with future opportunity cost so as to be optimal over a lifetime.

3.2 OFT Criticism and Explicit Processing Costs

Nonacs [69] bases his criticisms on the observation that Equation (3.2) implies that $g_1'(\tau_1^*) = g_2'(\tau_2^*) = \cdots = g_n'(\tau^*)$. That is, if behaviors in nature could be described by the MVT, then every per-type rate of average gain (i.e., "speed" of gain) should be equal to some global *giving-up density* (GUD). In the studies that Nonacs reviews, foragers stop exploiting different prey types at different speeds, which leads him to the conclusion that the MVT does not hold in reality.

As discussed by Stephens and Krebs [112], the function g_i is not meant to be a gross observable reward to the forager. Instead, it models the energetic reward to the forager minus the internal handling cost, which is usually not externally observable. To make this distinction clearer, we explicitly introduce an average handling cost $c_i(\tau_i)$ for exploiting prey of type i for an average of τ_i time, and we let $g_i(\tau_i)$ be the corresponding average observable reward. Thus, the forager should maximize

$$\frac{-c^s + \sum_{i=1}^{n} \lambda_i p_i \left(g_i(\tau_i) - c_i(\tau_i)\right)}{1 + \sum_{i=1}^{n} \lambda_i p_i \tau_i}. \tag{3.3}$$

In this framework, the MVT states that times $\tau_1^*, \ldots, \tau_n^*$ that make

$$g_i'(\tau_i^*) - c_i'(\tau_i^*) = \frac{-c^s + \sum_{i=1}^{n} \lambda_i \left(g_i(\tau_i^*) - c_i(\tau_i^*)\right)}{1 + \sum_{i=1}^{n} \lambda_i \tau_i^*} \quad \text{and} \quad g_i''(\tau_i^*) - c_i''(\tau_i^*) < 0 \tag{3.4}$$

will correspond to an optimal patch behavior. From Equation (3.4), it must be that $g_1'(\tau_1^*) - c_1'(\tau_1^*) = g_2'(\tau_2^*) - c_2'(\tau_2^*) = \cdots = g_n'(\tau_n^*) - c_n'(\tau_2^*)$. The rare case when the observable rewards are such that $g_1'(\tau_1^*) = g_2'(\tau_2^*) = \cdots = g_n'(\tau_n^*)$ is when the internal handling cost rates are also all equal, and so a global GUD is exceptional. Differences in terminal handling densities across types reflect differences in the handling burden of those types. For example, for types with the same reward–time curve and linear handling cost functions (i.e., $c_i(\tau_i) \triangleq c_i \tau_i$ for type i and constant $c_i > 0$), smaller marginal handling costs (i.e., $c_i'(\tau_i) = c_i$ for type i) should lead to longer optimal exploitation times—foragers spend relatively less time in patches with steep handling-cost functions.

3.3 Graphical Optimization and Long Residence Times

The intuitive graphical approach frequently used by Stephens and Krebs [112] for analysis of the single-type case can be extended to our model, which makes all costs explicit and uses multiple types. To do so, we define $\lambda \triangleq \lambda_1 + \lambda_2 + \cdots + \lambda_n$ to be the average Poisson encounter rate for all prey types combined. Then $1/\lambda$ is the average search time between encounters, and

$$\overline{g} \triangleq \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i g_i(\tau_i), \quad \overline{c} \triangleq \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i c_i(\tau_i), \quad \text{and} \quad \overline{\tau} \triangleq \sum_{i=1}^{n} \frac{\lambda_i}{\lambda} p_i \tau_i$$

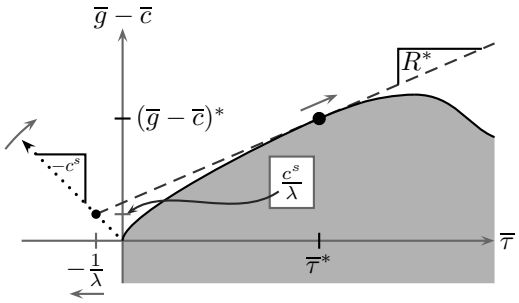


Figure 3.1: Effect of searching on optimal patch residence times. As search cost c^s or average interarrival time $1/\lambda$ increases, the slope R^* of the optimal ray will decrease, and the optimal average exploitation time $\overline{\tau}^*$ will increase.

are the average per-encounter handling gain, cost, and time. Equation (3.3) then becomes

$$\frac{\overline{g}-\overline{c}-\frac{c^s}{\lambda}}{\frac{1}{\lambda}+\overline{\tau}}. \tag{3.5}$$

Graphical optimization of Equation (3.5) is shown in Figure 3.1. Each point in the shaded area corresponds to a particular choice of preferences, $p_1, p_2, \ldots, p_n$, and exploitation times, $\tau_1, \ldots, \tau_n$. The upper boundary of the shaded area is the optimal frontier on which all optimal behaviors are found. Optimization consists of rotating a ray originating from $(-1/\lambda, c^s/\lambda)$ counter-clockwise from a $-90°$ angle to the least upper bound of the angles less than $90°$ where the ray intersects the shaded region. The slope of the optimal ray R^* is the optimal value of Equation (3.5), and the corresponding values $(\overline{g}-\overline{c})^*$ and $\overline{\tau}^*$ are the optimal per-encounter average net handling gain and exploitation time. Hence, this graphical interpretation of rate maximization allows for predictions of how perturbing parameters of the model will impact the average handling time $\overline{\tau}$. An increase in the average handling time may be due to an increase in the optimal exploitation time for each patch (i.e., changes in τ_i^* for each

type i) or due to additional patch types being added to the optimal diet (i.e., changes in p_i^* for each type i) or a mixture of the two effects. This graphical approach does not suggest how to predict each individual τ_i^* or p_i^*; it is only meant to investigate how certain parameter perturbations will affect general foraging trends. Specific optimal diet content and patch exploitation times can be found using algorithms from classical OFT [23, 24, 80, 112].

When Stephens and Krebs [112] consider graphical solutions to the single-type case, they frequently discuss how changes in encounter rate λ_1 lead to changes in optimal behavior. For multiple types, the overall rate λ can be changed while holding the *density* λ_i/λ constant for each type i. The graphical example in Figure 3.1 shows that the optimal average exploitation time $\overline{\tau}^*$ will increase if either the overall encounter rate λ decreases or the search cost rate c^s increases. In both cases, to maximize lifetime reward, the forager must increase present gains to compensate for future search losses—the *opportunity cost* of more exploitation decreases when search time or cost increase. Likewise, if the average per-encounter handling cost $\overline{c}$ is increased, the $\overline{g} - \overline{c}$ curve will be shifted down the vertical axis and the optimal average exploitation time $\overline{\tau}^*$ will increase. This prediction suggests an explanation for the ostensibly anomalous observation by Nolet et al. [68] that tundra swans spend more time feeding in areas of deep water where more energy is required to acquire similar prey as in shallow water. The increased costs necessarily decrease the maximal long-term rate of gain, which is the opportunity cost of increased exploitation, and so patches are exploited longer. This effect is investigated in more detail in Section 3.4.

3.4 The Sunk-cost Effect

Graphical optimization of Equation (3.3) helps explain apparently irrational behaviors discussed in the fields of economics, psychology, and biology in terms of lifetime gain maximization. In economics, the propensity to continue a costly task after paying some initial cost is sometimes called an *escalation commitment*, *escalation behavior*, or *escalation error*, but it is more commonly known as the *sunk-cost effect* [56, 108]. This nomenclature is also used by psychologists [e.g., 10]. In fact, psychologists Arkes and Ayton [11] note that the sunk-cost effect is equivalent to the *Concorde fallacy* first described by biologists Dawkins and Carlisle [27]. Although none of these terms are used, the same phenomena is also observed by Nolet et al. [68]. In particular, tundra swans must expend more energy to "up-end" to feed on deep-water tuber patches than they do to "head-dip" to feed on shallow-water patches; however, contrary to the expectations of Nolet et al., the swans feed for a longer time on each high-cost deep-water patch. In every context, the observation of the sunk-cost effect is an enigma because intuition suggests that this behavior is suboptimal. Here, we show how optimization of Equation (3.3) predicts the sunk-cost effect for certain scenarios; a common element of every case is a large initial cost.

3.4.1 Initial Costs: Recognition, Acquisition, Reconnaissance

For simplicity, we make a graphical argument with the assumptions that $n = 1$, $c^s = 0$, and $p_1 = 1$ (i.e., the patch case). We also revert to interpreting g_1 as the average net handling gain (i.e., the sum of an observable gain and an internal cost). Under these assumptions,

$$\frac{g_1(\tau_1)}{\frac{1}{\lambda_1} + \tau_1} \tag{3.6}$$

is the objective function for optimization. Similar arguments can be made about the general multivariate form in Equation (3.3).

Consider the case when gain function g_1 is initially:

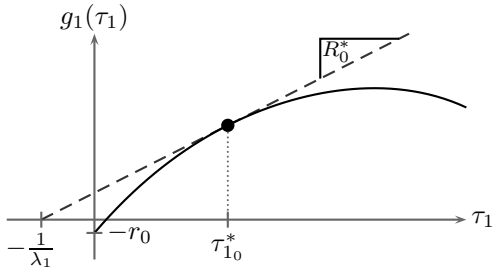
(i) negative (i.e., $g_1(0) < 0$),

(ii) increasing (i.e., $g_1'(0) > 0$), and

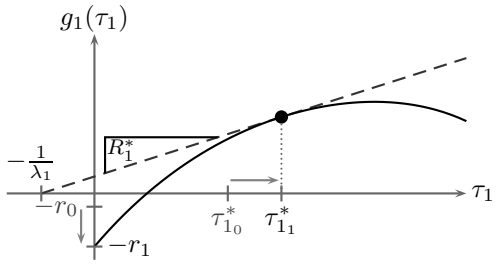
(iii) decelerating (i.e., $g_1''(0) < 0$).

Functions meeting items (ii) and (iii) are treated by Stephens and Krebs [112]. However, these functions are all initially zero. Here, we let the average handling gain $g_1(\tau_1)$ be initially negative to reflect some initial cost. For example, rather than treating recognition cost as a separate quantity [e.g., 112, pp. 79–81], we consider them to be a characteristic of the gain returned to the forager from handling each item (e.g., an initial energy expenditure required simply for access to or identification of a prey). Alternatively, as in the case of the tundra swans observed by Nolet et al. [68], the initial cost may model the extra energy required to acquire prey (e.g., by "up-ending" instead of simply "head-dipping"). Furthermore, just as Olsson and Brown [73] show how Bayesian foragers receive a relative "foraging benefit of information" for spending extra time in a patch, the initial cost here may be viewed as an average penalty due to uncertainty about the patch gain function.

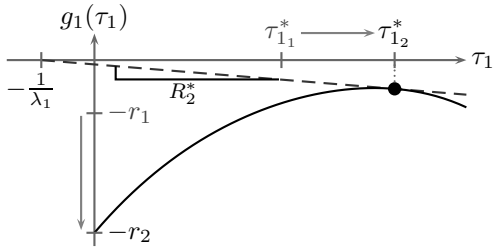
The graphical optimization procedure for such a initially negative gain function is shown in Figure 3.2. The same gain function is shown in Figures 3.2(a), 3.2(b), and 3.2(c) except that initial recognition cost is low, moderate, and high, respectively. When the initial cost increases from r_0 to r_1, the optimal exploitation time increases from $\tau_{1_0}^*$ to $\tau_{1_1}^*$. Even when the initial cost increases from r_1 to r_2 and makes the gain



(a) Function with low initial cost $r_0 > 0$.



(b) Function with high initial cost $r_1 > r_0$.



(c) Function with very high initial cost $r_2 > r_1$.

Figure 3.2: Different recognition costs $r_2 > r_1 > r_0$ have different optimal exploitation times $\tau_{1_2}^* > \tau_{1_1}^* > \tau_{1_0}$ and optimal rates $R_2^* < R_1^* < R_0^*$. For two equally shaped gain functions, the one with the higher initial cost will also have the higher optimal residence time.

function strictly negative, the exploitation time still increases from $\tau^*_{1_1}$ to $\tau^*_{1_2}$. This propensity to exploit items longer when the initial cost is increased is exactly the sunk-cost effect [10, 11]. However, it also maximizes the long-term net rate of gain, and so it is the rational behavior.

This result can be explained using an *opportunity cost* interpretation [51, 112]. The optimal rate R^* is an opportunity cost for spending additional time exploiting an item. It represents the maximum expected gain possible per unit time, and so it is costly to exploit items of a particular type for so long that the type's rate of average gain falls below R^*. When the initial cost of handling increases on all encounters, the total gain decreases, and so the opportunity cost for additional exploitation decreases (e.g., $R^*_1 < R^*_0$). Thus, the marginal handling gain must decrease further in order for marginal costs and marginal benefits to equalize. This effect manifests itself through the increase in optimal exploitation time.

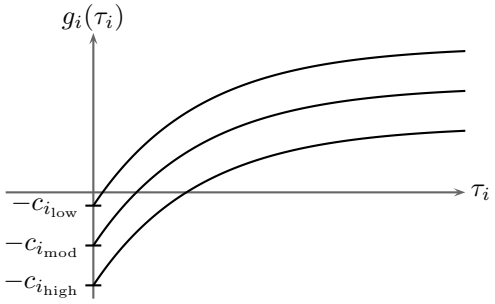
By increasing exploitation time when initial cost increases, the forager reduces the amount of time it spends paying high costs. Leaving patches at an earlier time is equivalent to volunteering for paying recognition costs more frequently. It is not the increased initial cost on a single encounter that is important; it is the increased initial cost on all encounters that causes the decrease in opportunity cost for additional exploitation.

3.4.2 Human and Nonhuman Examples

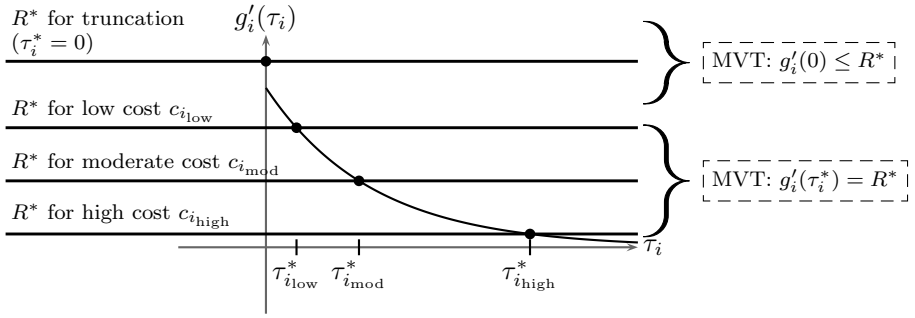
Feeding Time Increases in Areas of High-Cost Prey: In a study by Nolet et al. [68], the foraging behavior of tundra swans in shallow water is compared to the behavior in deep water. The swans forage on belowground tubers, and thus the

swans need only to "head-dip" in shallow water whereas they need to "up-end" in deep water to find and retrieve the tubers. The model that the authors use to explain the tundra swan behavior predicts that there will be a decrease in feeding time in areas where feeding has larger power requirements. However, in deep water where the power requirements to up-end are apparently larger than the requirements to head-dip, the observed tundra swans spent longer times feeding on tuber patches. Although this result is a contradiction of the model used by Nolet et al., it follows directly from the discussion in Section 3.4.1. In particular, let $n \geq 1$ (i.e., multiple tuber patch types are available) and $p_1 = \cdots = p_n = 1$ (i.e., encounters are not ignored). For each patch type i, the average net gain function g_i is assumed to be initially negative in order to model the energetic burden of acquiring the belowground tubers. Hence, we use the classical marginal value theorem result in Equation (3.2) applied to initially negative gain functions.

The curve in Figure 3.3(a) models the average value g_i of a tuber patch of type i after exploiting the patch for time τ_i. The average cost c_i of a patch entry reduces the maximum possible long-term rate of gain R^* from all patches. The MVT predicts that the optimal exploitation time τ_i^* occurs when $g_i'(\tau_i^*) = R^*$, which is depicted in Figure 3.3(b) for three different R^* maximum rates corresponding to three different c_i entry costs. As the average entry cost of a patch type increases, the maximum rate decreases, and so average exploitation time increases. When R^* is very high either due to low cost c_i or high return rates from other patches, the optimal commitment time drops to zero because the shallow value function is dominated by steep returns from other options.



(a) Gain $g_i(\tau_i)$ shifts following initial cost c_i.



(b) Marginal rate of gain $g_i'(\tau_i)$ is invariant of constant c_i.

Figure 3.3: Optimal exploitation time for patch type i with acquisition cost c_i. Increased costs lead to decreased overall return rate R^*, which leads to greater exploitation time in each costly patch. That is, when most encountered patches are costly to enter, exploitation time in each of these high-cost patches will be long in order to reduce their frequency. However, when enough other low-cost patches are available to raise the overall return rate sufficiently high, exploitation time to patches with relatively shallow marginal returns may decrease to zero. That is, when there are enough high-quality patches, time is better spent searching for them.

Human Preference for High Price: One recorded example of the sunk-cost effect in humans comes from studying behavior at the cinema. In a controlled study, Arkes and Blumer [10] charge movie patrons different prices for tickets to see a particular movie. They show that the likelihood that people attend the movie is positively correlated with the cost of the ticket, and they conclude that this behavior is an irrational mistake. However, this result can also be predicted by the classical marginal value theorem result in Equation (3.2) applied to initially negative gain functions[1]. Again, consider the case where $n \geq 1$ (i.e., multiple types are available) and $p_1 = \cdots = p_n = 1$ (i.e., encounters are not ignored). For each activity type i, let the gain function $g_i(\tau_i)$ be parameterized by *commitment time* τ_i that represents the average time the individual is committed to participating in that type of activity. That is, if the commitment time for a type of activity is low, there is a low probability that the individual will complete the activity, and thus the average gain returned from this type of activity will also be low. Likewise, if the commitment time for the activity is high, it will likely be attended, and so the value returned will level off as it returns no value after it is attended. Hence, the analysis summarized in Figure 3.3 also applies here. As ticket price increases for this set of individuals who are forced to buy the ticket, they become more strongly committed to attending the movie[2].

This example matches classical economic intuition. As the price a consumer pays increases, the overall purchasing frequency decreases. The longer commitment to a higher-priced movie reflects reluctance to make more purchases. Over the long term,

[1]Here, to be consistent with Arkes and Blumer [10], we do not allow encounters to be ignored, and so initial costs are forced and the pure patch model predicts the optimal behavior. The combined prey–patch model better fits reality as ticket purchasing opportunities can be ignored.

[2]If the experimenters allowed for encounters to be ignored (i.e., if participants could choose to not purchase a ticket), movies with zero commitment times would also have zero ticket sales.

this behavior maximizes the net value the consumer holds collectively in goods and capital. Behaving in any other way would be reckless when viewed from a long-term opportunity-cost perspective. In particular, exploitation times are longer in order to accumulate more gain to justify returning to a search that will likely result in finding more "prey" that force high initial costs. Thus, the sunk-cost effect may seem nonsensical for any single purchase, but it is adaptive when considering an economic or ecological system view.

3.4.3 Escalation Behavior

The propensity to continue a costly task ad infinitum is an extreme form of the sunk-cost effect known as *escalation behavior* or *escalation error* [56, 108]. In fact, the *Concorde fallacy* gets its name from the example given by Dawkins and Carlisle [27] of continued government investment on a supersonic jet that has no promise of ever returning a profit. Here, we show how escalation behavior can be an adaptive rate-maximizing strategy under special circumstances. These examples presume that the optimal long-term rate of gain R^* is negative, and so they are a better fit for economic examples where prolonged deficits are allowed. However, as several studies show that foragers can dynamically adjust behavior to maximize the present estimate of long-term rate of gain [38, 39, 105], these examples may also be consistent with short-term trends in behavior when foragers are temporarily subjected to high-cost environments. Additionally, this analysis may apply to similar mathematical models of value maximization when behaviors must choose from a set of costly options. For example, in spiders that choose to join existing webs or spin their own [54], small spiders face potentially dangerous competition with large spiders in new webs

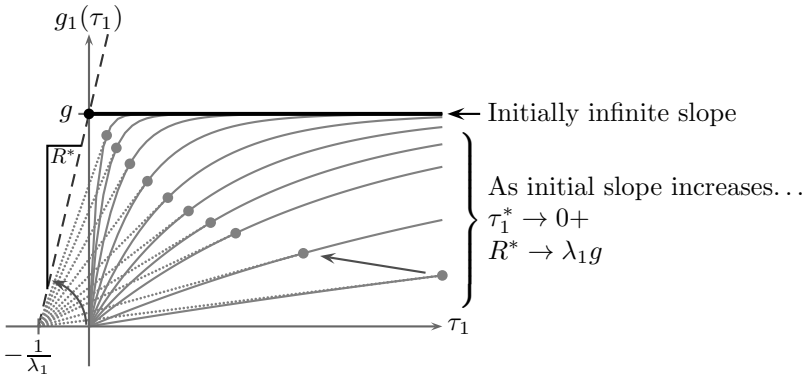


Figure 3.4: Analysis of positive constant gain function. The gain function is the limit of smooth asymptotic gain functions of increasing steepness. The limiting optimal behavior is to leave the patch immediately. Staying longer returns no additional gain, and leaving immediately decreases the time between subsequent high-quality patches.

but also face high costs in building new webs. Also, studies of women in abusive relationships suggests their decisions to leave the relationship are affected by fear of violence toward pets that remain [29]. In both of these cases, the optimal behavior may be accompanied by costs that might seem prohibitive in isolation. As in Section 3.4.1, we use the simplified optimization objective given by Equation (3.6).

In Figure 3.4, a positive constant gain function is shown as the limit of asymptotic gain functions of increasing steepness. Because of the asymptotic upper limit, sharp increases in gain correspond to sharp decreases in marginal returns. So the optimal patch residence time decreases as steepness increases because marginal returns fall so quickly. In the limiting case, when the gain function is constant, the optimal behavior is to leave immediately because staying longer cannot result in any additional gain. In particular, it is better to invest time searching for new patches that guarantee an entry reward g at regular rate λ_1 (i.e., long-term rate of gain $\lambda_1 g$) rather than

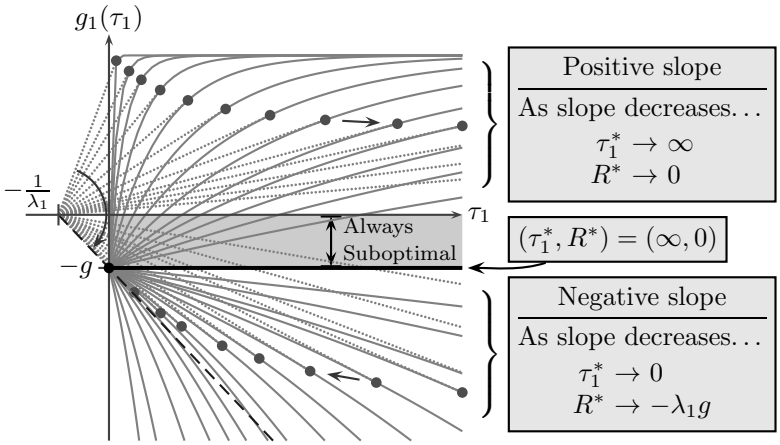


Figure 3.5: Analysis of negative constant gain function. Escalation error can be viewed as the limit of increasingly shallow gain functions that are initially negative. Zero patch residence time is not restored until steepness is *sufficiently negative*. For this class of functions, optimal points will never be found in the shaded region between the horizontal axis and the constant gain function.

staying in any single patch that provides no additional gain (i.e., long-term rate of gain $g/\infty \approx 0$).

The opposite effect is demonstrated in Figure 3.5, which shows a negative gain function as the limit of asymptotic gain functions of decreasing steepness. As the steepness of each gain function is reduced, the optimal patch residence time increases without bound. So the optimal behavior for a constant gain function that is negative is to remain forever in each encountered patch. It is better to pay a single recognition cost once and remain in the patch forever rather than paying the recognition cost repeatedly at each new encounter.

As shown in the lower portion of Figure 3.5, even when the gain function is initially decreasing and convex, the optimal behavior is not to immediately leave the patch unless the initial slope is sufficiently negative. It is better for the forager

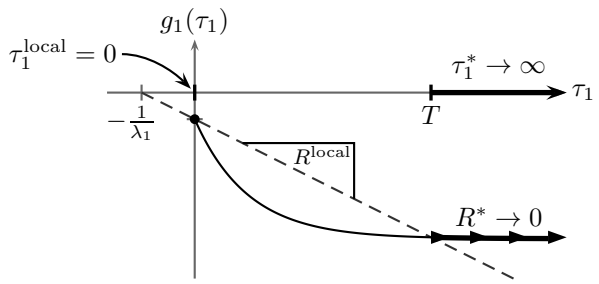


Figure 3.6: Initially negative concave gain function has latent escalation behavior that dominates the early local maximum. Leaving the patch immediately returns a higher long-term gain than staying a short time longer, but it returns a lower gain than staying for any time greater than T. Hence, escalation behavior is optimal because it prevents paying any future entry costs.

to pay additional costs within the patch rather than face the $-\lambda_1 g$ lifetime rate of gain associated with immediately leaving the patch. However, escalation behavior is avoided in this case because the convex gain function eventually becomes too steep. A different effect occurs for initially negative and initially steep concave functions like the one shown in Figure 3.6. Here, immediate withdrawal from the patch is a locally optimal behavior. That is, the objective function decreases as the exploitation time is increased slightly. However, as exploitation time increases, the objective function reaches its global minimum and then starts increasing. Moreover, there exists a time T such that exploiting these patches for any time $\tau_1 \geq T$ will be favorable to immediate withdrawal. Again, it is better to exploit patches with a shallow negative gain longer than to search for more patches that have a steep gain.

In the supersonic jet construction example [27], the costly behavior may be adaptive because it prevents starting other costlier ventures. More generally, successful politicians in power may induce endless and massive spending in one area for no other reason than to avoid new spending in other areas. In a foraging context, a forager

that must pay a large energetic cost to enter a patch (e.g., due to high predation risk, having to swim upstream, or high uncertainty about patch quality) may hesitate to ever leave that patch if the habitat contains few other types of patches. Similarly, social group members may resist the temptation to leave a low-quality group if joining any other group is accompanied by costly antagonistic violence [e.g., 29, 54]. In general, in a set of apparently bad choices, the choice that has the lowest eventual marginal losses will minimize the very long-term losses.

3.5 Conclusions

Optimal foraging theory introduces quantitative analysis into the study of behavior in a way that complements intuition. We have shown how explicitly modeling foraging costs can improve the accuracy of rate-maximizing OFT methods and answer the criticisms of Nonacs [69] that rate maximization too often underestimates patch residence time. As verified by van Gils et al. [118], the Bayesian forager modeled by Olsson and Holmgren [74] accurately predicts observed patch residence times. This result is investigated by Olsson and Brown [73] who derive a "foraging benefit of information" that is an additional reward that Bayesian foragers receive for staying longer in patches to gather information. As these models investigate foraging behavior using numerical dynamic programming techniques, they lack the intuition of classical OFT. However, if the discovery penalty of the information gathering process is modeled as an initial in-patch cost, then relatively simple classical OFT methods also predict longer exploitation times.

We also show how explicitly including large initial in-patch costs in classical OFT methods explains why sunk-cost effects like those observed by Nolet et al. [68] are

rational rate-maximizing foraging behaviors. This result seems counter-intuitive, but it follows from using a farsighted decision-making model based on comparing present returns to the opportunity cost of not searching. When encounters with costly patches are likely, it is better to spend more time in each patch in order to reduce the accumulation rate of said costs. This interpretation may also explain human preference for high price [e.g., 10] and the escalation of costly behaviors ad infinitum [e.g., 27, 29]. Sunk-cost behaviors are criticized because they are examples of making decisions based on past investments; however, they should be better understood as decisions that reduce the frequency of similar future investments.

Part II: Optimal Distributed Task Processing

In the following two chapters, we depart from the case of a solitary task-processing agent with a local optimization objective. Here, we discuss groups of distributed agents. The communication between agents is limited, and their actions are largely uncoordinated. However, the ensemble of their individual behaviors results in desirable global properties. Hence, these intelligent algorithms allow development resources to be focused on improving the task-processing capabilities as opposed to mechanisms required for reliable coordination. Additionally, these distributed behaviors serve as prototypes for understanding emergent phenomena already observed in nature. In both cases described here, behaviors of a group of distributed agents converge upon an optimal resource allocation. The first case treats a narrow set of problems that have separable configuration spaces and seek Nash optimal resource allocations, and the second case treats a broad range of problems with polyhedral configuration spaces where Pareto optimal resource allocations are desired.

In Chapter 4, we introduce a novel framework for the analysis and design of distributed agents that must complete externally generated tasks but also can volunteer to process tasks encountered by other agents. Example applications include flexible manufacturing systems and autonomous air vehicles. In both cases, agents have overlapping capabilities encounter, but each agent is responsible for a different incoming flow of tasks (e.g., widget requests or pop-up targets). It is desirable that agents cooperate to share task load, but relatively loaded agents should accept fewer tasks from others and vice versa. A distributed asynchronous volunteering policy is presented that dynamically adjusts task flow around the network of agents. It is shown that even though agents independently adjust their tendency to volunteer to process tasks from other agents, the set of all volunteering tendencies converges to the unique Nash

equilibrium of a cooperation game. An artificial cooperation trading economy ensures that at the equilibrium, non-zero cooperation tendencies are possible and vary across agents. In particular, an agent with relatively high task encounter rate not only provides more incentive for connected neighbors to cooperate with it but also has less incentive to volunteer to cooperate with other agents. The framework is shown via simulation to be applicable to autonomous air vehicles, and the mathematical results of the chapter are also shown to be consistent with classic studies of cooperation from science. This work was originally presented by Pavlic and Passino [85].

In Chapter 5, we investigate the general case of minimizing a cost function over a set of non-separable constraints. Although the results we describe are generic primal space numerical optimization methods amenable to parallelization, we focus on the specific application of intelligent lighting. In particular, a system contains a set of lighting agents and sensor agents. Each sensor is associated with a particular minimum illumination constraint; however, it is desirable that the system meets these constraints using minimum total power across the lights. Sensors are unable to communicate directly with other sensors; however, the lights themselves can serve as a shared memory bank that each sensor can use to infer the demands of the other sensors. We show how the single-sensor version of this problem is isomorphic to the ideal free distribution (IFD) model of social foraging (i.e., allocating a fixed number of animals to food sources to meet a nutrient constraint). Furthermore, we show how a generalization of the IFD matches the general lighting problem and use the IFD to inspire a distributed behavior that converges to the optimal allocation. The salient feature of these numerical solution methods is that they are allowed to temporarily

violate their constraints, and their small perturbations away from the feasible set provide information that allows them to return to the space at a less costly allocation. Other example applications include optimal power dispatch (i.e., allocating power from a set of distributed generators to supply an aggregate consumer demand). Additionally, our characterization of optimal generalized IFD allocations may suggest how eusocial insect societies that have several nutrient constraints may appear to violate known rational allocation strategies but actually fit a generalized IFD.

Chapter 4: Cooperative Task Processing

We consider a network of autonomous agents for which some agents are responsible for processing tasks from one or more external sources. When a task arrives at one of these agents, the agent may advertise the task to other agents connected to it. If none of the connected agents volunteer to process the task, it must be processed by the advertising agent; otherwise, the task is processed by one of the volunteering agents. Agents that volunteer for tasks may themselves be connected to incoming task flows for which they can advertise task encounters. In general, an agent in the network may advertise task encounters to others, volunteer to process advertised tasks from others, or do both. Our challenge is to define a distributed asynchronous algorithm for automatically tuning how often each agent volunteers to process advertised tasks so that the set of volunteering tendencies across the network converges to a Nash equilibrium. In the following, we review existing cooperative processing work and discuss why those approaches are not adequate for solving the problem we formulate here.

Grid computing [22] is one existing approach for achieving cooperative task processing across a group of networked task-processing agents. System designers work

under the assumption of heterogeneous agents with conflicting priorities. They borrow from the economic theories of mechanism design [61, Chapter 23] and implementation theory [76, Chapter 10] to design mechanisms (e.g., brokering agents) and protocols that either encourage resource sharing [5, 42, 106, 123] or discourage exploitation [75, 93] among groups of agents. The common element of these different methods of *distributed algorithmic mechanism design* (DAMD) [30] is that the designer has no direct control over individual agents; instead, they control the structure of the interactions between given agents on a given network. Hence, DAMD is not appropriate for the design of the task-processing networks themselves.

Methods exist for the design of networks of interconnected task-processing agents that have desirable task flow characteristics. For example, a *flexible manufacturing system* (FMS) includes several machines that switch their current processing to one of several input task flows and then produce output task flows for other machines in the system. Perkins and Kumar [86] show that distributed scheduling policies exist that guarantee such systems will have finite upper bounds on all buffers of tasks. Similarly, Cruz [26] shows how special network elements can be combined to form queueing systems with output traffic flows that are guaranteed to have finite burstiness constraints so long as the input flows also satisfy similar constraints. These methods are not intended to describe how agents can dynamically adjust task flow to exploit unused processing ability on idle connected agents.

Because an optimal task flow configuration may be unknown, inaccessible, or changing over time, task-processing agents may need to use feedback to acquire and

stabilize the optimal task-handling behavior. For example, a set of autonomous air vehicles (AAV) deployed for distributed search, surveillance, or task processing can coordinate their actions in order to converge on a holistically optimal behavior [31, 32, 37]. However, the coordination required between agents can be prohibitive. Additionally, the single optimality criteria being maximized ignores fatigue on individual agents. For example, in a smart power grid [52], it may be desirable for distributed power stations to share load; however, a single overloaded station should not result in a cascade of self-sacrificing failures. Here, non-cooperative game theory is used to develop totally asynchronous and distributed algorithms for task-processing agents that both respect local processing priorities while also sharing the processing burden of highly loaded neighbors.

Non-cooperative game theory has been traditionally used to design optimal control strategies [14]; however, it can also be used to design simple selfish strategies that nonetheless assist neighbors. Several such techniques already exist for designing policies on nodes in *ad hoc* multi-hop communication networks [3, 4, 21]. In these cases, nodes can forward packets from other nodes in order to reduce network congestion or improve communication diversity, but nodes resist using all local resources for assisting other nodes. A salient feature of these forwarding networks is that packets can be duplicated or dropped at any time. Hence, these networks are ill-equipped to model task-processing scenarios where tasks that enter the network must be assigned and processed by exactly one agent. Instead, our approach passes volunteering requests around a network and uses an economics-inspired task-processing network game to determine how best to respond to these requests. The resulting volunteering policy

is sensitive to both local processing requests and the presence of other agents on the network that can volunteer as well.

This chapter is organized as follows. In Section 4.1, the task-processing network framework is defined and example task-processing networks are described. The optimization game is presented in Section 4.2, and an asynchronous distributed computation method that ensures convergence to the game's Nash equilibrium is given in Section 4.3. In Section 4.4, results from a simulated task-processing network of autonomous air vehicles are presented, and conclusions and areas of future research are discussed in Section 4.5.

4.1 Task-Processing Network

In the following, we use the real numbers $\mathbb{R}$, the natural numbers $\mathbb{N} \triangleq \{1, 2, \dots\}$, the whole numbers $\mathbb{W} \triangleq \{0, 1, 2, \dots\}$, and derived symbols like the non-negative real numbers $\mathbb{R}_{\geq 0}$. Take a finite but arbitrarily large set $\mathcal{A} \subset \mathbb{N}$ of *task-processing agents* and a set $\mathcal{P} \subseteq \{(i,j) \in \mathcal{A}^2 : i \neq j\}$ of directed arcs connecting distinct agents. For each agent $i \in \mathcal{A}$, $\mathcal{V}_i \triangleq \{j \in \mathcal{A} : (j,i) \in \mathcal{P}\}$ and $\mathcal{C}_i \triangleq \{j \in \mathcal{A} : (i,j) \in \mathcal{P}\}$ are respectively the sets of *conveyors* and *cooperators* connected to agent i. Hence, $\mathcal{V} \triangleq \{j \in \mathcal{A} : \mathcal{C}_j \neq \emptyset\} = \bigcup_{i \in \mathcal{A}} \mathcal{V}_i$ and $\mathcal{C} \triangleq \{i \in \mathcal{A} : \mathcal{V}_i \neq \emptyset\} = \bigcup_{j \in \mathcal{A}} \mathcal{C}_j$ are respectively the sets of all conveyors and cooperators in the network. Assume that:

1. For all $i \in \mathcal{A}$, there exists a finite and possibly empty set $\mathcal{Y}_i \subset \mathbb{N}$ of *task types* such that for all $k \in \mathcal{Y}_i$, tasks of type k arrive at agent i from an external source at average rate $\lambda_i^k \in \mathbb{R}_{>0}$. Each external source of tasks is assumed to be independent of all other sources.

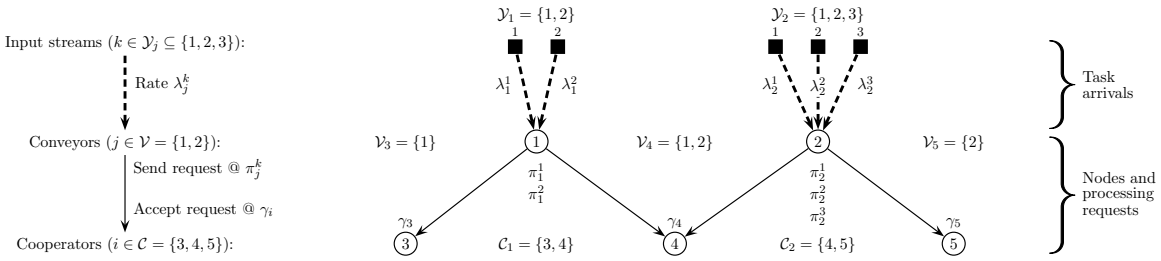


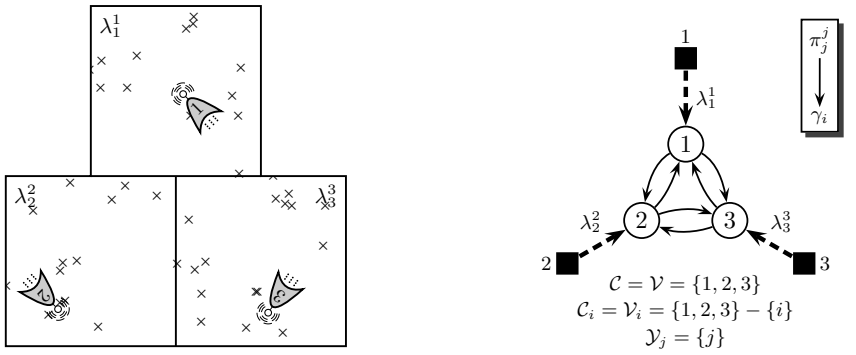
Figure 4.1: Simple flexible manufacturing system example.

2. If $j \in \mathcal{V}$, then there exist $k \in \mathcal{Y}_j$ with $\pi_j^k \neq 0$ where $\pi_j^k \in [0,1]$ represents the probability that conveyor j advertises an incoming k-type task to its connected cooperators $\mathcal{C}_j$. If $j \in \mathcal{V}$ does not advertise a task to its connected cooperators, the task will be processed by agent j.

3. If $i \in \mathcal{C}$, then there is some $\gamma_i \in [0,1]$ that represents the probability that agent i will volunteer for an advertised task from one of its connected conveyors $\mathcal{V}_i$. Any task arriving at conveyor $j \in \mathcal{V}$ that is advertised to cooperators $\mathcal{C}_j$ will be processed with uniform probability by exactly one of the cooperators that volunteer for it; if no cooperators volunteer for the task, then it is processed by conveyor j.

The graph $\mathcal{G} \triangleq (\mathcal{A}, \mathcal{P})$, rates, and probabilities defined above characterize a *task-processing network* (TPN).

The simple TPN shown in Figure 4.1 represents a flexible manufacturing system (FMS) similar to the systems described by Perkins and Kumar [86]. Tasks of types 1, 2, and 3 arrive according to independent Poisson processes. Type-1 and type-2 tasks arrive at agent 1, and all three types of tasks arrive at agent 2. For tasks



(a) AAV-patrolled territories.

(b) AAV task-processing network.

Figure 4.2: A task-processing network formed by three autonomous air vehicles (AAV). (a) AAV-patrolled territories (b) Corresponding task-processing network

of type $k \in \mathcal{Y}_1 = \{1, 2\}$, agent 1 advertises task arrivals to agents 3 and 4 with probability π_1^k. Likewise, agent 2 advertises arrivals of tasks of type $k \in \mathcal{Y}_2 = \{1, 2, 3\}$ to agents 4 and 5 with probability π_2^k. The system designer can choose different probabilities for each task type based on the specialized abilities of each agent. Each agent $i \in \{3, 4, 5\}$ volunteers for an advertised task with probability γ_i independent of task type. Hence, in this TPN, agents 1 and 2 are conveyors and agents 3, 4, and 5 are cooperators.

In the FMS example, the set of conveyors and the set of cooperators are disjoint. In a general TPN, an agent can be both a cooperator and a conveyor. For example, the fully-connected TPN shown in Figure 4.2(b) models an autonomous air vehicle (AAV) patrol scenario shown in Figure 4.2(a) that is similar to others in resource allocation literature [31, 32, 37]. Each AAV $i \in \{1, 2, 3\}$ continuously searches its territory for tasks (e.g., targets) to process, and these tasks are generated (i.e., found) at rate $\lambda_i > 0$. When a task is found, the AAV advertises the task to both of its

neighbors. If neither neighbor volunteers for processing, the AAV processes the task itself. In this fully-connected topology, all agents are both cooperators and conveyors. Although this network has several cycles, tasks do not move around the network — if a volunteering cooperator is given a task for processing, it cannot generate a new task-processing request for that task; it must process it itself.

Task-processing networks describe a broad range of applications. The AAV example above can also serve as a model of a mobile software agent [57, 58, 60, 99, 126] that patrols for tasks to process or any general group of networked processors [36]. Additionally, by converting encounter rates to energetic rates (i.e., power demand), TPNs can model the behavior of smart power grids [52] made up of stations that request assistance from neighbors. That is, cooperator stations adjust additional supply provided in response to demand requests from remote conveyor stations.

4.2 Cooperation Game Among Selfish Agents

In a task-processing network, the probability (i.e., *cooperation willingness*) $\gamma_i \in [0, 1]$ that cooperator $i \in \mathcal{C}$ will volunteer for an advertised task from its connected conveyors must be chosen. It is assumed that this choice must be done in a distributed fashion and it is impractical for agents to coordinate in order to maximize some global utility. So each agent independently chooses a cooperation policy that maximizes its individual utility (i.e., agents are selfish). Hence, optimality is given in terms of the Nash equilibrium [19, Section 3.5.1].

To inform each cooperator how to choose this policy, the network's designer assigns cost and rewards to agent operations in a common currency (e.g., proportional to dollars of net profit) that is called *points* here. In particular,

- Agent $i \in \mathcal{A}$ receives $(b_i^k - c_i^k)$ net points for processing a locally generated task of type $k \in \mathcal{Y}_i$.

- Conveyor $i \in \mathcal{V}$ receives r_i^k when a task of type $k \in \mathcal{Y}_i$ from i is processed by a $\mathcal{C}_i$ cooperator.

- If cooperator $j \in \mathcal{C}_i$ volunteers and is selected to process a task of type $k \in \mathcal{Y}_i$ from conveyor $i \in \mathcal{V}$, then cooperator j pays cost c_{ij}^k to process that task.

However, these costs and benefits alone do not provide cooperators with any incentive to volunteer to process conveyor tasks, and so a payment mechanism is required. Consider conveyor $j \in \mathcal{V}$ and task type $k \in \mathcal{Y}_j$. If one or more cooperators in $\mathcal{C}_j$ volunteer frequently to process requests from agent j, the other cooperators in the set should conserve resources by volunteering infrequently. To ensure this qualitative behavior, each cooperator $i \in \mathcal{C}_j$ receives volunteering payment $q_{ij}^k p_j^k(Q_j)$ from conveyor $j \in \mathcal{V}_i$ where:

- $Q_j \triangleq \sum_{k \in \mathcal{C}_j} \gamma_k$ is the *total quantity of cooperation willingness* available to conveyor j.

- $p_j^k(Q_j)$ is a decreasing *payment function* that represents the price that conveyor j pays to its connected cooperators each time they volunteer for a task of type $k \in \mathcal{Y}_j$.

- $q_{ij}^k \in \mathbb{R}_{>0}$ is a value factor that scales payment $p_j^k(Q_j)$ from conveyor j into the currency of cooperator $i \in \mathcal{C}_j$ (i.e., i perceives $q_{ij}^k p_j^k(Q_j)$ value from the contribution $p_j^k(Q_j)$ from j).

So if any cooperator $i \in \mathcal{C}_j$ increases its cooperation willingness γ_i, it increases how often it receives payment $p_j^k(Q_j)$ while also decreasing the payment itself. For each cooperator $i \in \mathcal{C}_j$, these two pressures encourage cooperation willingness (i.e., $\gamma_i > 0$) and resource conservation (i.e., $\gamma_i < 1$).

To maximize net points earned over a long run time, each agent chooses a policy that maximizes its own expected rate of point accumulation. So for a given vector $\vec{\gamma} = [\gamma_{c_1}, \gamma_{c_2}, \ldots, \gamma_{c_{|\mathcal{C}|}}]^\top \in [0,1]^{|\mathcal{C}|}$ of cooperation policies (where unique $c_k \in \mathcal{C}$ for all $k \in \{1, 2, \ldots, |\mathcal{C}|\}$), the utility (i.e., long-term rate of point gain) returned to cooperator $i \in \mathcal{C}$ is

$$U_i(\vec{\gamma}) \triangleq \overbrace{b_i + \underbrace{\left(1 - \prod_{j \in \mathcal{C}_i}(1-\gamma_j)\right)}_{\Pr(\text{Volunteer from } \mathcal{C}_i | \text{Advertisement from } i)} r_i - Q_i p_i(Q_i)}^{\text{Conveyor part — constant with respect to } \gamma_i} \tag{4.1a}$$

$$+ \underbrace{\gamma_i \sum_{j \in \mathcal{V}_i} \big(p_{ij}(Q_j) - \overbrace{\text{SOBP}_1(\mathcal{C}_j - \{i\})}^{\Pr(i \text{ awarded task from } j | i \text{ volunteers})} c_{ij}\big)}_{\text{Cooperator part — } \gamma_i \text{ and } Q_j \text{ vary with } \gamma_i} \tag{4.1b}$$

where

$$b_i \triangleq \sum_{k \in \mathcal{Y}_i} \lambda_i^k \left(b_i^k - c_i^k\right), \tag{4.1c}$$

$$r_i \triangleq \sum_{k \in \mathcal{Y}_i} \lambda_i^k \pi_i^k \left(r_i^k - \left(b_i^k - c_i^k\right)\right), \tag{4.1d}$$

$$p_i(Q_i) \triangleq \sum_{k \in \mathcal{Y}_i} \lambda_i^k \pi_i^k p_i^k(Q_i), \tag{4.1e}$$

are the costs and benefits of local processing on $i \in \mathcal{V}$, and

$$c_{ij} \triangleq \sum_{k \in \mathcal{Y}_j} \lambda_j^k \pi_j^k c_{ij}^k, \tag{4.1f}$$

$$p_{ij}(Q_j) \triangleq \sum_{k \in \mathcal{Y}_j} \lambda_j^k \pi_j^k q_{ij}^k p_j^k(Q_j). \tag{4.1g}$$

are the costs and benefits to $i \in \mathcal{C}$ for volunteering for tasks exported from $j \in \mathcal{V}_i$. The expression for SOBP is given in Definition 4.1.

Definition 4.1 (Sum of binomial products)**.** *Let $\mathcal{I}$ be a finite index set and $\Omega \triangleq \{\gamma_i\}_{i\in\mathcal{I}}$ be an indexed family where $\gamma_i \in [0,1]$ for each $i \in \mathcal{I}$. For $\Gamma \subseteq \mathcal{I}$ and $g \in \mathbb{N}$, the* sum of binomial products

$$\mathrm{SOBP}_g(\Gamma) \triangleq \sum_{\ell=0}^{|\Gamma|} \frac{1}{g+\ell} \sum_{\substack{\mathcal{C}\subseteq\Gamma \\ |\mathcal{C}|=\ell}} \left(\left(\prod_{i\in\mathcal{C}} \gamma_i \right) \left(\prod_{k\in\Gamma-\mathcal{C}} (1-\gamma_k) \right) \right). \tag{4.2}$$

For a cooperator $i \in \mathcal{C}$, $\mathrm{SOBP}_1(\mathcal{C}_j - \{i\})$ is the probability that cooperator i is chosen to process an advertised task from conveyor $j \in \mathcal{V}_i$ when it is given that it volunteers for the task. Hence, for $j \in \mathcal{V}_i$, the impact of cost rate c_{ij} decreases as other cooperators from $\mathcal{C}_j$ increase their own cooperation willingness because the probability that agent i will be selected decreases. So for a conveyor $j \in \mathcal{V}$, its connected cooperators $\mathcal{C}_j$ form a Cournot oligopoly [67] (i.e., a set of independent agents that provide a service for a demand-driven price) with a positive externality [13] (i.e., the cost of processing decreases as more cooperators enter the market). The underbraced *cooperator part* of Equation (4.1b) shows that cooperator i must set its cooperation willingness γ_i (i.e., its quantity of supplied cooperation) based on the summed returns from several such markets.

4.3 Distributed Computation of the Nash Equilibrium

Let $n \triangleq |\mathcal{C}|$. Because there is no coordination between players, the n-dimensional play space is the Cartesian product $\prod_{i\in\mathcal{C}}[0,1] = [0,1]^n$, and the collection of cooperation policies across all cooperators is the vector $\vec{\gamma} \triangleq [\gamma_{c_1}, \gamma_{c_2}, \ldots, \gamma_{c_n}]^\top \in [0,1]^n$ (where unique $c_k \in \mathcal{C}$ for all $k \in \{1,2,\ldots,n\}$). For each $i \in \mathcal{C}$, it is assumed that the utility

function $U_i : [0,1]^n \mapsto \mathbb{R}$ is twice-continuously differentiable, and so, by Weirstrass' theorem, U_i is bounded above and below and achieves its extrema. Following Bertsekas and Tsitsiklis [19, Proposition 5.7 from Chapter 3], the Nash equilibria of the cooperation game can be found by solving n separate one-dimensional variational inequality problems. In particular, $\vec{\gamma}^* \in [0,1]^n$ is a Nash equilibria of the cooperation game if and only if, for all $i \in \mathcal{C}$,

$$(\gamma_i - \gamma_i^*)\nabla_i U_i(\vec{\gamma}^*) \leq 0 \quad \text{for all } \gamma_i \in [0,1] \tag{4.3}$$

where the block gradient (i.e., the ith row of the gradient)

$$\nabla_i U_i(\vec{\gamma}) = \sum_{j \in \mathcal{V}_i} \Big(\overbrace{p_{ij}(Q_j) + \gamma_i p'_{ij}(Q_j)}^{\frac{\partial}{\partial \gamma_i}(\gamma_i p_{ij}(Q_j))} - \text{SOBP}_1(\mathcal{C}_j - \{i\}) c_{ij} \Big).$$

So in a local neighborhood of the Nash equilibrium $\gamma^* \in [0,1]^n$, any unilateral perturbation of a coordinate of γ^* will result in equal or reduced utility.

The existence of a solution to the n simultaneous nonlinear equations in Equation (4.3) is not guaranteed in general and may be difficult to find analytically. However, variational inequalities over product spaces are well suited for parallel and asynchronous computation [19]. Under special conditions on each utility function, a unique Nash equilibrium is guaranteed to exist, and each of its coordinates in Equation (4.3) can be computed independently in the distributed and asynchronous fashion described by Assumption 4.1.

Assumption 4.1 (Totally asynchronous distributed iteration)**.** *Take* $(c_1, c_2, \ldots, c_n) \triangleq \mathcal{C}$ *to represent the* n *distinct cooperators of* $\mathcal{C}$*. Let* $\mathcal{T} \triangleq \mathbb{W}$ *to be the indices of a sequence of physical times, and let* $\{\vec{\gamma}(t)\}_{t \in \mathcal{T}} \triangleq \{(\gamma_{c_1}(t), \gamma_{c_2}(t), \ldots, \gamma_{c_n}(t))\}$ *be a sequence of iterated calculations in the* $[0,1]^n$ *play space. For each* $i \in \mathcal{C}$*, subset* $\mathcal{T}^i \subseteq \mathcal{T}$

corresponds to the times when coordinate $\gamma_i(t)$ *is computed. Additionally, for each* $i, j \in \mathcal{C}$ *and each* $t \in \mathcal{T}$*, there is an index* $\tau_j^i(t) \in \mathcal{T}$ *of the least-outdated version of coordinate* γ_j *available for the computation of coordinate* γ_i *with transition mapping* $T_i : [0,1]^n \mapsto [0,1]$ *at time* t *such that* $0 \leq \tau_j^i(t) \leq t$*. That is, an outdated state estimate*

$$\vec{\gamma}^i(t) \triangleq (\gamma_{c_1}^i(t), \gamma_{c_2}^i(t), \ldots, \gamma_{c_n}^i(t)) \triangleq (\gamma_{c_1}(\tau_{c_1}^i(t)), \gamma_{c_2}(\tau_{c_2}^i(t)), \ldots, \gamma_{c_n}(\tau_{c_n}^i(t)))$$

is available for the computation $\gamma_i(t+1) = T_i(\vec{\gamma}^i(t))$ *for each* $t \in \mathcal{T}$ *and* $i \in \mathcal{C}$*. It is assumed that*

1. *Set* $\mathcal{T}^i$ *is countably infinite (i.e.,* $|\mathcal{T}^i| = |\mathcal{T}| = |\mathbb{N}|$*) for all* $i \in \mathcal{C}$*.*

2. *If subsequence* $\{t_k\}$ *of* $\mathcal{T}^i$ *is such that* $\lim_{k\to\infty} t_k = \infty$*, then* $\lim_{k\to\infty} \tau_j^i(k) = \infty$ *for all* $i, j \in \{1, 2, \ldots, n\}$*. That is,* $\liminf_{t\to\infty} \tau_j^i(t) = \infty$ *for all* $i, j \in \{1, 2, \ldots, m\}$*.*

For all $t \in \mathcal{T}$*, sequence* $\{\vec{\gamma}(t)\}$ *is generated by the* totally asynchronous distributed iteration (TADI)

$$\gamma_i(t+1) \triangleq \begin{cases} T_i(\vec{\gamma}^i(t)), & \text{if } t \in \mathcal{T}^i, \\ \gamma_i(t), & \text{if } t \notin \mathcal{T}^i \end{cases} \tag{4.4}$$

where $\vec{\gamma}(t) \triangleq (\gamma_{c_1}(t), \gamma_{c_2}(t), \ldots, \gamma_{c_n}(t))$*. For each* $i \in \mathcal{C}$*, the transition mapping* $T_i : [0,1]^n \mapsto [0,1]$ *in Equation (4.4) is defined by*

$$T_i(\vec{\gamma}) \triangleq \min\{1, \max\{0, \gamma_i + \sigma_i \nabla_i U_i(\vec{\gamma})\}\}$$

where $\sigma_i \in \mathbb{R}_{>0}$ *is a* step-size *parameter that scales movement along the utility gradient* $\nabla_i U_i$*.*

4.3.1 Conditions for Distributed Convergence

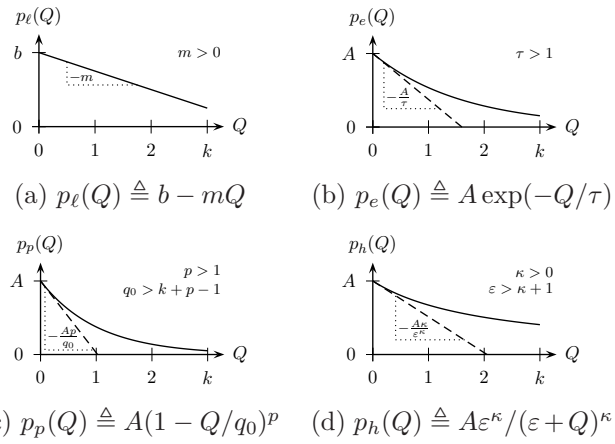
The TADI-generated $\{\vec{\gamma}(t)\}$ sequence represents the collective motion of n self-interested agents that each climb their respective utility gradient in order to maximize their expected rate of point return. That is, Equation (4.4) may be viewed as a dynamical system model of coupled agents that each take independent actions. In particular, it can be shown that there exists a constant $\underline{\text{SOBP}} > 0$ such that $\text{SOBP}_1(\Gamma) \geq \underline{\text{SOBP}}$ for all $\Gamma \subseteq \mathcal{C}$. So, assuming that $c_{ij} > 0$ and payment $p_{ij} \equiv 0$ for all $i, j \in \mathcal{A}$, the response of the system reaches $\vec{\gamma}(T) = \vec{0}$ in some finite time $T \in \mathbb{W}$. That is, the intrinsic agent behavior is not to cooperate. For each $i \in \mathcal{C}$, it is desirable to find a control law, which is implemented through the choice of payment function, to destabilize the no-cooperation equilibrium and provide feedback to stabilize the Nash equilibrium. It will be shown that functions satisfying Definition 4.2 the necessary characteristics.

Definition 4.2 (Stabilizing payment function)**.** *For $k \in \mathbb{N}$, a* stabilizing payment function (SPF) *$p : [0, k] \mapsto \mathbb{R}$ is a twice-continuously-differentiable function such that:*

1. *It is strictly decreasing. In particular, $p'(Q) \triangleq \mathrm{d}p(Q)/\mathrm{d}Q < 0$ for all $Q \in [0, k]$.*
2. *It is convex. In particular, $p''(Q) \triangleq \mathrm{d}^2p(Q)/\mathrm{d}^2Q \geq 0$ for all $Q \in [0, k]$.*
3. *Its convexity is eventually dominated by its slope. In particular,*

$$\gamma p''(Q) \leq -p'(Q) \text{ for all } Q \in [\gamma, k - (1 - \gamma)] \text{ with } \gamma \in [0, 1]. \tag{4.5}$$

The set of SPFs is closed under conical combinations (i.e., it is a filled cone). So for $i \in \mathcal{C}$, if p_{ij} is an SPF for all $j \in \mathcal{V}_i$, then the sum $\sum_{j \in \mathcal{V}_i} p_{ij}(Q_j)$ is itself an SPF.



(a) $p_\ell(Q) \triangleq b - mQ$ (b) $p_e(Q) \triangleq A\exp(-Q/\tau)$

(c) $p_p(Q) \triangleq A(1 - Q/q_0)^p$ (d) $p_h(Q) \triangleq A\varepsilon^\kappa/(\varepsilon + Q)^\kappa$

Figure 4.3: Sample stabilizing payment (i.e., inverse demand) functions. (a) $p_\ell(Q) \triangleq b - mQ$ (b) $p_e(Q) \triangleq A\exp(-Q/\tau)$ (c) $p_p(Q) \triangleq A(1 - Q/q_0)^p$ (d) $p_h(Q) \triangleq A\varepsilon^\kappa/(\varepsilon + Q)^\kappa$

Additionally, by the definition of $p_{ij}(Q_j)$ in Equation (4.1g), if $p_j^k(Q_j)$ is an SPF for all $j \in \mathcal{V}$ and $k \in \mathcal{Y}_j$, then $p_{ij}(Q_j)$ will also be an SPF for all $i \in \mathcal{C}$.

Proposition 4.1 (Sufficient conditions for stabilization)**.** *Take $k \in \mathbb{N}$ and function $p : [0, k] \mapsto \mathbb{R}$. If $0 \leq p''(Q) < -p'(Q)$ for all $Q \in [0, k]$, then p is a stabilizing payment function.*

Four example SPFs are shown in Figure 4.3. Each payment function meets the conditions of Proposition 4.1; however, the weaker condition 3 of Definition 4.2 is met for $\varepsilon \geq \kappa$ in (d). Additionally, the polynomial function in (c) is an extension of the linear function in (a).

Convergence to the Nash equilibrium depends not only on the structure of the payment functions but also on the structure of the TPN graph itself. Sufficient convergence conditions for a TPN network and its payment functions are given in Theorem 4.1, which uses Definition 4.3 to describe the topological constraints on the TPN graph.

Definition 4.3 (k-conveyor)**.** *Conveyor $i \in \mathcal{V}$ is a* k-conveyor *if it has exactly $k \in \mathbb{N}$ outgoing connections to cooperators (i.e., if $k = |\mathcal{C}_i|$).*

Theorem 4.1 (Convergence of cooperation)**.** *Assume that*

1. *For all $i \in \mathcal{C}$ and $j \in \mathcal{V}_i$, p_{ij} is a stabilizing payment function.*
2. *For all $j \in \mathcal{V}$, $|\mathcal{C}_j| \leq 3$ (i.e., no conveyor can have more than 3 outgoing links to cooperators).*
3. *For $i \in \mathcal{C}$ and $j \in \mathcal{V}_i$, if j is a 3-conveyor, then there must be some $k \in \mathcal{V}_i$ that is a 2-conveyor.*

Define $T : [0,1]^n \mapsto [0,1]^n$ by $T(\vec{\gamma}) \triangleq (T_1(\vec{\gamma}), T_2(\vec{\gamma}), \ldots, T_n(\vec{\gamma}))$ where, for each $i \in \mathcal{C}$,

$$T_i(\vec{\gamma}) \triangleq \min\{1, \max\{0, \gamma_i + \sigma_i \nabla_i U_i(\vec{\gamma})\}\}, \tag{4.6a}$$

where

$$\frac{1}{\sigma_i} \geq 2|\mathcal{V}_i| \max_{k \in \mathcal{V}_i} |p'_{ik}(0)| \tag{4.6b}$$

for all $\vec{\gamma} \in [0,1]^n$. If

$$\min_{j \in \mathcal{V}_i} |p'_{ij}(|\mathcal{C}_j|)| > \left(|\mathcal{V}_i| - \frac{1}{2}\right) \max_{j \in \mathcal{V}_i} |c_{ij}|, \quad \text{for all } i \in \mathcal{C}, \tag{4.7}$$

then the TADI sequence $\{\vec{\gamma}(t)\}$ generated with mapping T and the outdated estimate sequence $\{\vec{\gamma}^i(t)\}$ for all $i \in \mathcal{C}$ each converge to the unique Nash equilibrium of the cooperation game.

Proof of Theorem 4.1. By assumption 1 (i.e., all payment functions are stabilizing), for any $\vec{\gamma} \in [0,1]^n$ and $i \in \mathcal{C}$,

$$\begin{aligned}\nabla^2_{ii} U_i(\vec{\gamma}) &= \sum_{j \in \mathcal{V}_i} \left(2p'_{ij}(Q_j) + \gamma_i p''_{ij}(Q_j)\right) \\ &= \sum_{j \in \mathcal{V}_i} \overbrace{p'_{ij}(Q_j)}^{<0} + \sum_{j \in \mathcal{V}_i} \overbrace{\left(p'_{ij}(Q_j) + \gamma_i p''_{ij}(Q_j)\right)}^{\leq 0} < 0,\end{aligned}$$

and

$$\begin{aligned}\nabla^2_{ii} U_i(\vec{\gamma}) &= \sum_{j \in \mathcal{V}_i} \overbrace{\left(2p'_{ij}(Q_j)\right)}^{<0} + \overbrace{\gamma_i p''_{ij}(Q_j)}^{\geq 0} \geq -2 \sum_{j \in \mathcal{V}_i} |p'_{ij}(Q_j)| \\ &\geq -2 \sum_{j \in \mathcal{V}_i} \max_{k \in \mathcal{V}_i} |p'_{ik}(0)| = -2|\mathcal{V}_i| \max_{k \in \mathcal{V}_i} |p'_{ik}(0)| \\ &\geq -2|\mathcal{V}_i| \max_{k \in \mathcal{V}_i} |p'_{ik}(0)|. \end{aligned} \tag{4.8}$$

So by the assumed limits on step size σ_i given in Equation (4.6b), $0 > \nabla^2_{ii} U_i(\vec{\gamma}) \geq -1/\sigma_i$ for all $i \in \mathcal{C}$.

Next, we bound the cross terms $\nabla^2_{i\ell} U_i$ of the utility Hessian. These bounds require the introduction of SOMS in Definition 4.4, which represents the slope of the SOBP. Moreover, Lemmas 4.1, 4.2, and 4.3 put bounds on SOMS and precisely relate it to the SOBP.

Definition 4.4 (Sum of monomial sums)**.** *Let $\mathcal{I}$ be a finite index set and $\Omega \triangleq \{\gamma_i\}_{i \in \mathcal{I}}$ be an indexed family where $\gamma_i \in [0,1]$ for each $i \in \mathcal{I}$. For $\Gamma \subseteq \mathcal{I}$ and $h \in \mathbb{N}$, the* sum of monomial sums

$$\mathrm{SOMS}_h(\Gamma) \triangleq \sum_{\ell=0}^{|\Gamma|} (-1)^\ell \frac{1}{h+\ell} \sum_{\substack{\mathcal{C} \subseteq \Gamma \\ |\mathcal{C}|=\ell}} \left(\prod_{i \in \mathcal{C}} \gamma_i \right). \tag{4.9}$$

Lemma 4.1 (SOBP_1 derivative)**.** *If $\Gamma \subseteq \mathcal{I}$ and $k \in \Gamma$, then $\partial\, \mathrm{SOBP}_1(\Gamma)/\partial \gamma_k = -\,\mathrm{SOMS}_2(\Gamma - \{k\})$.*

Lemma 4.2 (SOMS floor)**.** *For* $\Gamma \subseteq \mathcal{I}$ *and* $h \in \mathbb{N}$, $\mathrm{SOMS}_h(\Gamma) \geq (1/h) \prod_{k=1}^{|\Gamma|} k/(h+k)$.

Lemma 4.3 (SOMS ceiling)**.** *For* $\Gamma \subseteq \mathcal{I}$ *and* $h \in \mathbb{N}$, $\mathrm{SOMS}_h(\Gamma) \leq 1/h$.

Take $\vec{\gamma} \in [0,1]^n$ and cooperator $i \in \mathcal{C}$. For another cooperator $\ell \in \mathcal{C} - \{i\}$, if $\ell \notin \mathcal{C}_j$ (i.e., ℓ is not an outgoing cooperator for j), then $\partial Q_j / \partial \gamma_\ell = 0$ and $\partial\, \mathrm{SOBP}_1(\mathcal{C}_j - \{i\})/\partial \gamma_\ell = 0$ where $Q_j \triangleq \sum_{k \in \mathcal{C}_j} \gamma_k$ and SOBP is from Definition 4.1. So by introducing SOMS from Lemma 4.1,

$$\begin{aligned} 0 &\leq \sum_{\substack{\ell \in \mathcal{C} \\ \ell \neq i}} \left| \nabla^2_{i\ell} U_i(\vec{\gamma}) \right| \\ &= \sum_{\substack{\ell \in \mathcal{C} \\ \ell \neq i}} \left| \sum_{j \in \mathcal{V}_i} [\ell \in \mathcal{C}_j] \left(\begin{array}{l} p'_{ij}(Q_j) + \gamma_i p''_{ij}(Q_j) \\ \quad + \underbrace{\mathrm{SOMS}_2(\mathcal{C}_j - \{i, \ell\})}_{\partial / \partial \gamma_\ell \, \mathrm{SOBP}_1(\mathcal{C}_j - \{i\})} c_{ij} \end{array} \right) \right| \end{aligned}$$

where $[\cdot]$ is the Iverson bracket (i.e., $[S] = 1$ or $[S] = 0$ when statement S is true or false). By Lemmas 4.2 and 4.3, $0 < \mathrm{SOMS}_2(\Gamma) \leq 1/2$ for all $\Gamma \subseteq \mathcal{C}$. Hence,

$$\begin{aligned} \sum_{\substack{\ell \in \mathcal{C} \\ \ell \neq i}} \left| \nabla^2_{i\ell} U_i(\vec{\gamma}) \right| &\leq \sum_{\substack{\ell \in \mathcal{C} \\ \ell \neq i}} \sum_{j \in \mathcal{V}_i} [\ell \in \mathcal{C}_j] \left(\underbrace{\left| p'_{ij}(Q_j) + \gamma_i p''_{ij}(Q_j) \right|}_{\leq 0} + \frac{1}{2} |c_{ij}| \right) \\ &= \sum_{j \in \mathcal{V}_i} \left(\left| p'_{ij}(Q_j) + \gamma_i p''_{ij}(Q_j) \right| + \frac{1}{2} |c_{ij}| \right) \sum_{\substack{\ell \in \mathcal{C} \\ \ell \neq i}} [\ell \in \mathcal{C}_j] \\ &= \sum_{j \in \mathcal{V}_i} \left(\left| p'_{ij}(Q_j) + \gamma_i p''_{ij}(Q_j) \right| + \frac{1}{2} |c_{ij}| \right) (|\mathcal{C}_j| - 1) \,. \end{aligned}$$

However, by assumption 2, each conveyor $j \in \mathcal{V}$ has no more than three outgoing connections to cooperators (i.e., $|\mathcal{C}_j| \leq 3$). Additionally, by assumption 3, if $j \in \mathcal{V}_i$ is a 3-conveyor (i.e., it has 3 outgoing cooperator connections), then there must be some other conveyor $m \in \mathcal{V}_i - \{j\}$ that is a 2-conveyor. So letting $m \in \mathcal{V}_i$ be the

2-conveyor that is guaranteed to exist,

$$\sum_{\substack{\ell\in\mathcal{C}\\ \ell\neq i}} \left|\nabla^2_{i\ell}U_i(\vec{\gamma})\right| \leq \overbrace{2\sum_{j\in\mathcal{V}_i-\{m\}}\left(\underbrace{\left|p'_{ij}(Q_j)+\gamma_i p''_{ij}(Q_j)\right|}_{\leq 0}+\frac{1}{2}|c_{ij}|\right)}^{\substack{\text{Doubled contribution to sum from}\\ \text{other cooperators connected to}\\ \text{assumed 3-conveyors in } \mathcal{V}_i-\{m\}}}$$
$$\underbrace{+\left|\underbrace{p'_{im}(Q_m)+\gamma_i p''_{im}(Q_m)}_{\leq 0}\right|+\frac{1}{2}|c_{im}|}_{\substack{\text{Contribution to sum from}\\ \text{other cooperator of 2-conveyor } m\in\mathcal{V}_i}}.$$

By Definition 4.2 of an SPF,

$$\begin{aligned}\sum_{\substack{\ell\in\mathcal{C}\\ \ell\neq i}} \left|\nabla^2_{i\ell}U_i(\vec{\gamma})\right| &\leq -\sum_{j\in\mathcal{V}_i-\{m\}}\left(2p'_{ij}(Q_j)+\gamma_i p''_{ij}(Q_j)\right)\\ &\quad -\left(p'_{im}(Q_m)+\gamma_i p''_{im}(Q_m)\right)\\ &\quad +\left(\overbrace{|\mathcal{V}_i-\{m\}|}^{m\in\mathcal{V}_i}+\frac{1}{2}\right)\max_{j\in\mathcal{V}_i}|c_{ij}|\\ &= -\sum_{j\in\mathcal{V}_i-\{m\}}\left(2p'_{ij}(Q_j)+\gamma_i p''_{ij}(Q_j)\right)\\ &\quad -\left(p'_{im}(Q_m)+\gamma_i p''_{im}(Q_m)\right)\\ &\quad +\left(|\mathcal{V}_i|-\frac{1}{2}\right)\max_{j\in\mathcal{V}_i}|c_{ij}|.\end{aligned}$$

or, equivalently,

$$\begin{aligned}\sum_{\substack{\ell\in\mathcal{C}\\ \ell\neq i}} \left|\nabla^2_{i\ell}U_i(\vec{\gamma})\right| &\leq -\sum_{j\in\mathcal{V}_i}\left(2p'_{ij}(Q_j)+\gamma_i p''_{ij}(Q_j)\right)\\ &\quad +\left(2p'_{im}(Q_m)+\gamma_i p''_{im}(Q_m)\right)\\ &\quad -\left(p'_{im}(Q_m)+\gamma_i p''_{im}(Q_m)\right)\\ &\quad +\left(|\mathcal{V}_i|-\frac{1}{2}\right)\max_{j\in\mathcal{V}_i}|c_{ij}|.\end{aligned}$$

Thus

$$\sum_{\substack{\ell\in\mathcal{C}\\ \ell\neq i}} \left|\nabla^2_{i\ell}U_i(\vec{\gamma})\right| \leq -\overbrace{\sum_{j\in\mathcal{V}_i}\left(2p'_{ij}(Q_j)+\gamma_i p''_{ij}(Q_j)\right)}^{\nabla^2_{ii}U_i(\vec{\gamma})}+\overbrace{p'_{im}(Q_m)}^{<0}+\left(|\mathcal{V}_i|-\frac{1}{2}\right)\max_{j\in\mathcal{V}_i}|c_{ij}|.$$

So

$$\sum_{\substack{\ell \in \mathcal{C} \\ \ell \neq i}} \left|\nabla^2_{i\ell} U_i(\vec{\gamma})\right| \leq -\nabla^2_{ii} U_i(\vec{\gamma}) - \underbrace{\left(\min_{j \in \mathcal{V}_i} |p'_{ij}(Q_j)| - \left(|\mathcal{V}_i| - \frac{1}{2}\right) \max_{j \in \mathcal{V}_i} |c_{ij}|\right)}_{> 0 \text{ by Equation (4.7)}}$$

where, by the assumption in Equation (4.7), the underbraced expression is strictly positive. The desired result follows from this inequality and Equation (4.8). In particular, as shown by Bertsekas and Tsitsiklis [19, Propositions 1.1, 1.11, and 5.1 from Chapter 3], T is a maximum-norm contraction mapping with unique fixed point $\vec{\gamma}^*$ that is the Nash equilibrium of the cooperation game. Futhermore, the TADI sequence $\{\vec{\gamma}(t)\}_{t \in \mathcal{T}}$ generated by T converges to $\vec{\gamma}^*$ [19, Proposition 2.1 from Chapter 6]. □

4.3.2 Interpretations

As shown in the proof of Theorem 4.1, every 3-conveyor contributes two payment slope p'_{ij} terms to $\nabla^2_{i\ell} U_i$ that are canceled by the two slope terms in $\nabla^2_{ii} U_i$. Hence, when 3-conveyors are connected to a cooperator, the cooperator loses control of its utility gradient along its cooperation coordinate unless there exists a 2-conveyor that it can dominate. So 2-conveyors are themselves stabilizers that allow a cooperator $i \in \mathcal{C}$ to focus its decision making on the conveyors in $\mathcal{V}_i$ for which there is only one other cooperator competing for payment. For example, in the complex TPN in Figure 4.4, the 3-conveyors in the network (e.g., 2, 4, 7, and 10) could destabilize the gradient ascent if the 2-conveyors (e.g., 1, 5, 6, 8, 9, and 11) were not also present. It may be possible to weaken Theorem 4.1 to allow for conveyors with $n > 3$ outgoing connections to cooperators so long as the slopes of the n-conveyor payment functions can be dominated by those of other 1-conveyors.

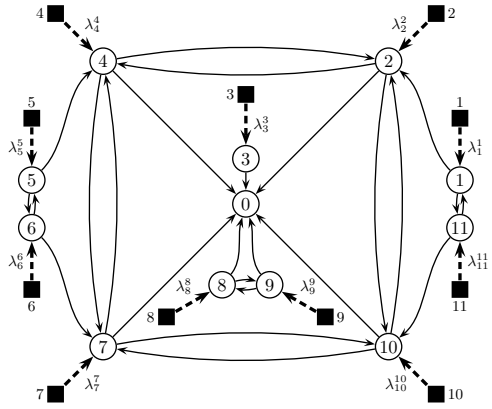


Figure 4.4: Many-agent task-processing network with stable topology.

The restriction in Equation (4.7) is similar to the network generalization of Hamilton's rule [43] discussed by Ohtsuki et al. [72] and Nowak [70]. In their case, the graph consists of individuals at graph nodes that have relationships modeled by graph links. Each individual is either a cooperator, which pays a cost to deliver a benefit to a neighbor, or a defector, which pays no cost and delivers no benefit. Behavioral strategies spread by way of birth–death processes, and they show that a sufficient condition for cooperation to spread is that the benefit-to-cost ratio is greater than the average degree (i.e., number of neighbors connected to each node) of the network. Here, the task-processing network is not a substrate for birth–death processes; however, the rule in Equation (4.7) plays a similar role relating payment, cost, and cooperator degree. Moreover, it is a sufficient condition for individual gradient ascent to converge upon a stable cooperation policy. Hence, just as Hamilton's rule allows scientists to reason about cooperation in natural networks, it also ensures that automata will find stable cooperation strategies in artificial networks.

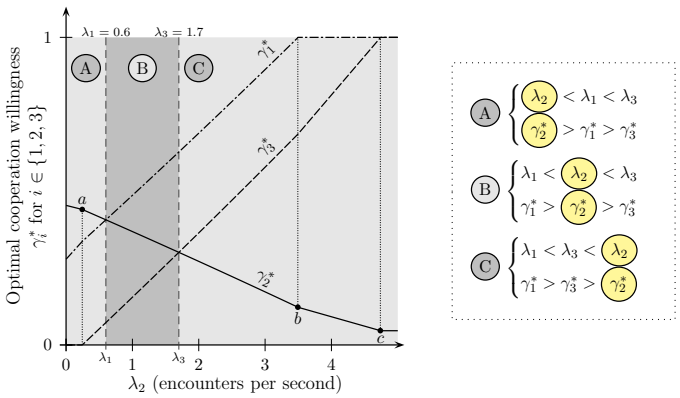


Figure 4.5: AAV optimal cooperation willingness as encounter rates vary.

4.4 Simulation of Cooperative AAV Scenario

Consider the AAV scenario shown in Figure 4.2. Assume that $\pi_i^k = 1$, $c_{ij}^\ell = 0.1$, and $q_{ij}^\ell = 1$ for all $i \in \mathcal{A}$, $j \in \mathcal{A} - \{i\}$, $k \in \mathcal{Y}_i$, and $\ell \in \mathcal{Y}_j$. Also assume that $\lambda_1^1 = 0.6$, $\lambda_3^3 = 1.7$, $0 < \lambda_2 \leq 5$, and the linear payment function $p_i^i(Q_i) \triangleq 1 - Q_i/\lambda_i^i$ for all $i \in \mathcal{A}$. Hence, the three otherwise equivalent agents face different task encounter rates, and their payment functions have slopes that are inversely proportional to each encounter rate. So agents associated with higher encounter rates have a higher demand for cooperation and thus have inelastic payment functions (i.e., cooperation retains its high value even when a high quantity is available).

A conservative choice of step size $\sigma_\ell \triangleq 1/(4 \max_{i \in \mathcal{A}, j \in \mathcal{V}_i} p'_{ij}([0,0,0]^\top))$ for all $\ell \in \mathcal{A}$ yields a convergent TADI for this scenario. MATLAB simulation results summarized in Figure 4.5 show how the resulting Nash equilibrium $\vec{\gamma}^*$ depends upon the AAV encounter rates. In particular, the Nash equilibrium has the desirable feature that $\lambda_i > \lambda_j$ implies that $\gamma_i^* < \gamma_j^*$ for all $i, j \in \mathcal{A}$. So agents that are locally busy are less willing to cooperate, and agents that are relatively idle are more willing to cooperate.

In Figure 4.5, as λ_2 increases, payment function p_2 to agents 1 and 3 becomes shallower and causes the optimal γ_1^* and γ_3^* to increase. However, as γ_1^* (or γ_3^*) increases, payment $p_3(Q_3)$ (or $p_1(Q_1)$) to agent 2 is depressed and γ_2^* decreases. Moreover, at point b when the ascent of γ_1^* truncates, the rate that γ_2^* decreases shallows. At point c when γ_3^* also truncates, the γ_2^* graph flattens entirely. Hence, to reduce the load on the saturated cooperators, agent 2 reciprocates for their cooperation by not reducing its own cooperation level to zero. So even though each agent's own encounter rate has no direct relationship to its TADI-directed movement, a desirable coupling between encounter rates and optimal cooperation levels emerges from the cyclic relationships on the network.

4.5 Conclusion

A framework for cooperative task processing on a network has been presented. Using this framework, a particular totally asynchronous cooperative control policy was shown to stabilize the Nash equilibrium of a cooperation game. By introducing a cooperation-trading economy into the formulation, the agents individually climb their own local utility functions yet still achieve an equilibrium where task processing is shared among different agents. The present work adjusts each agent's overall cooperation willingness in order to maximize economic returns over a lifetime of task encounters and processing requests. Future work should address the case where each agent associates a different cooperation willingness with each of its connected conveyors. Likewise, forwarding probabilities could be considered to be decision variables that should be adjusted across each agent's connected cooperators. The present work associates only one decision variable with each distributed agent, and so it makes the

simplifying assumption that those variables come from a Cartesian product space. However, if future frameworks place multiple decision variables on a single distributed agent, that assumption may be relaxed.

A weakness of the present work is that it implicitly assumes that agents either have infinite processing capacity or that all tasks have negligible processing time. Processing and switching durations are central motivations for the work of Perkins and Kumar [86] just as finite capacity motivates the work of Cruz [26]. Effects like these can be added to this model by explicitly modeling the average processing time of each task. In particular, the present work optimizes the long-term rate of gain of each agent based on rewards issued at the instant each task arrives, and this rate will be depressed by the processing time of each task. Moreover, the time spent processing a task represents a opportunity cost due to the lost time available for encountering other tasks that return higher profit. Because each arrival is independent, the study of average reward rates of Markov renewal–reward processes by Johns and Miller [55] can be used to model the long-term rate of gain when considering task processing times. Hence, the utility functions discussed in this work can be easily modified to include these effects. If analytically tractable, optimal results can be found that account for appreciable processing time.

Chapter 5: The MultiIFD as Distributed Gradient Descent for Constrained Optimization

Distributed optimization methods are typically unconstrained or, as was the case in Chapter 4, constrained within a configuration space that is a Cartesian product of independent subspaces corresponding to each agent. Otherwise, each agent must strongly coordinate its next action with its neighbors so as to prevent infeasible solutions. Consequently, parallelization of optimization problems with inseparable constraints typically involves re-casting the problem in its dual space where the optimal Lagrange multipliers can be solved for instead [19]. Because the multiplier associated with each constraint is independent of the other multipliers, the dual space is a separable Cartesian product and thus amenable to parallelization. Unfortunately,

- Distributed dual-space solvers still must collectively elect particular agents to operate on each coordinate of the dual space. This task is especially complicated when the dual and primal spaces have different dimensions.

- Even though the dual space is separable, the optimization problem itself may lack separability. In fact, the translation into the dual space can destroy primal space functional separability (e.g., a diagonal quadratic cost function in the primal space can translate into a quadratic cost function with cross coupling).

- After the optimal multipliers have been discovered, they must all be broadcast to all agents so that the optimal primal-space solution can be found.

- During the calculation of the optimal multipliers for a set of system parameters, those system parameters may change. If they change quickly, the relatively long delay from one multiplier calculation to the next can lead to inappropriate oscillations in the optimal solutions for the system. These jumps can be mitigated by generating intermediate solutions that incorporate the intermediate multiplier estimates; however, *ad hoc* methods like these create difficulties in quantifying the robustness of the system.

Consequently, despite the myriad methods of solving constrained optimization problems numerically, the inseparability of constraints presents many problems to distributed implementations.

Thus, in this concluding chapter, we describe a novel numerical approach to nonlinear optimization under constraints that is designed specifically for certain parallel implementations. Ultimately, despite the optimization algorithm used, the presence of inseparable constraints requires some coordination between agents. Here, the coordination is implemented stigmergically. That is, agent actions leave a residue in a shared location, and other agents respond appropriately to that residue. Coordination thus emerges from the agents even though they lack direct communication. However, it is a necessary condition of this algorithm that system trajectories can temporarily leave the constraint set. In principle, the distributed algorithm can be adapted to so that the ultimate error bounds vanish; however, the non-equilibrium behavior must be allowed to deviate from the feasible set. Examples of systems where this behavior is acceptable include intelligent lighting, where illumination constraints

can temporarily be violated, and eusocial insect foraging, where nutrient surpluses and deficits are temporarily acceptable.

This chapter is organized as follows. In Section 5.1, the most generic form of the optimization problem is presented. The optimal solutions of the problem are characterized using classical optimization theory, and example applications where the problem is relevant are described. Additionally, the conventional dual-space optimization approach is summarized. In Section 5.2, we present our distributed primal-space algorithm and the assumptions required for its convergence to the desired optimum point. Finally, in Section 5.3, we present experimental results for verification.

Notation: We use conventional mathematical notation in our discussion. The natural numbers are denoted by $\mathbb{N} \triangleq \{1, 2, \ldots\}$, and the whole numbers are denoted by $\mathbb{W} \triangleq \{0, 1, 2, \ldots, \}$. For function $F : \mathcal{X} \mapsto \mathbb{R}$ where $\mathcal{X} \subseteq \mathbb{R}^n$, at point $\vec{x} \in \mathcal{X}$, the gradient $\nabla F(\vec{x}) \triangleq [\nabla_1 F(\vec{x}), \nabla_2 F(\vec{x}), \ldots, \nabla_n F(\vec{x})]^\top \triangleq [\partial F(\vec{x})/\partial x_1, \partial F(\vec{x})/\partial x_2, \ldots, \partial F(\vec{x})/\partial x_n]^\top$. For a vector $\vec{x}$, the norm $\|\vec{x}\| \triangleq \|\vec{x}\|_2 \triangleq \sqrt{\vec{x}^\top \vec{x}}$. Other notation will be defined as needed.

5.1 The Optimization Problem

The most basic version of the focal optimization problem consists of a cost functional to minimize and a set of linear inequality constraints. In particular,

- Let $n \in \mathbb{N}$ be the dimension of the domain of the cost functional. For each $i \in \{1, 2, \ldots, n\}$, there exist constants $\underline{x}_i, \overline{x}_i \in \mathbb{R}_{\geq 0}$ such that $\overline{x}_i \geq \underline{x}_i \geq 0$. Furthermore, the set $\mathcal{X} \triangleq \prod_{i=1}^{n} [\underline{x}_i, \overline{x}_i]$ (i.e., $\mathcal{X}$ is a Cartesian product of n intervals of the real line). Then $F : \mathcal{X} \mapsto \mathbb{R}$ is the cost function to minimize.

- Let $m \in \mathbb{N}$ be the number of linear inequality constraints. That is, for each $j \in \{1, 2, \ldots, m\}$, there is a minimum level $c_j \in \mathbb{R}_{\geq 0}$ and a vector $\vec{a}_j$ that is normal to the hyperplanar boundary of the feasible set $\mathcal{C}_j \triangleq \{\vec{x} \in \mathcal{X} : \vec{a}_j^\top \vec{x} \geq c_j\}$. That is, $\vec{a}_j$ represents the contribution of motion $\vec{\Delta} \in \mathbb{R}^n$ to reducing or increasing the constraint deficit. It is assumed that there is some $j \in \{1, 2, \ldots, n\}$ such that $\sum_{i=1}^{n} a_{ji} \geq 0$. Collect each constraint normal vector into $m \in n$ matrix $A \triangleq [\vec{a}_1, \vec{a}_2, \ldots, \vec{a}^n]^\top$ and each minimum level into vector $\vec{c}_j \triangleq [c_1, c_2, \ldots, c_m]^\top$. The intersection of the feasible sets $\cap_{j=1}^{m} \mathcal{C}_j = \{\vec{x} \in \mathcal{X} : A\vec{x} \geq \vec{c}\}$ is a compact convex polyhedron.

Thus, the non-linear optimization problem over linear constraints is to

$$\begin{aligned} \text{minimize} \quad & F(\vec{x}) \\ \text{subject to} \quad & A\,\vec{x} \geq \vec{c}. \end{aligned} \tag{5.1}$$

The existence solutions to this problem requires additional constraints on the shape of the cost function. These constraints will accumulated in the next sections as necessary.

5.1.1 Characterization of Optimal Solutions

Let F be convex and continuously differentiable. An optimal solution to Equation (5.1) must exist; however, there may be several optima. Let $\vec{x}^* \in \mathcal{X}$ be an optimal solution. By the Karush–Kuhn–Tucker (KKT) conditions [18], there must exist a scalar Lagrange multiplier $\lambda_j^* \in \mathbb{R}_{\geq 0}$ for each constraint $j \in \{1, 2, \ldots, m\}$ and scalar Lagrange multipliers $\mu_i^*, \nu_i^* \in \mathbb{R}_{\geq 0}$ respectively for the lower and upper bounds $\underline{x}_i, \overline{x}_i$ for each $i \in \{1, 2, \ldots, n\}$ such that

$$\begin{aligned} \nabla F(\vec{x}^*) = {} & \lambda_1^* \vec{a}_1 + \lambda_2^* \vec{a}_2 + \cdots + \lambda_m^* \vec{a}_m \\ & + (\mu_1^* - \nu_1^*)\vec{e}_1 + (\mu_2^* - \nu_2^*)\vec{e}_2 + \cdots + (\mu_n^* - \nu_n^*)\vec{e}_n \end{aligned}. \tag{5.2}$$

where $\vec{e}_i$ is an elementary vector for $i \in \{1, 2, \ldots, n\}$ with $e_{ii} \triangleq 1$ and $e_{ij} = 0$ for all $j \in \{1, 2, \ldots, n\} - \{i\}$. Thus, when $\vec{x}^*$ is in the interior of $\mathcal{X}$, the gradient $\nabla F(\vec{x}^*)$ is a conical combination of the vectors normal to each active constraint. Moreover, if the gradient is bounded away from this cone so that some directions are unsupported by all constraints, the additional support will be provided by the upper and lower bounds (i.e., the elementary vectors are normal to separable auxiliary constraints in the system).

5.1.2 Example Applications

The non-linear optimization problem in Equation (5.1) can be shown to be a more general case of existing problems in biology and engineering. We explore the animal distribution problem at length first and then translate its conclusions by analogy to a classical power problem and an intelligent lighting problem. In each case, a finite resource is being optimally distributed across to a set of tasks (e.g., foragers to food patches, power demand to generators, power supplied to lights). A central contribution of this chapter is the generalization of the classical one-resource–one-constraint allocation model to use several constraints for the one resource. However, Moore et al. [66] extend he one-resource–one-constraint model to consider several resources each with a single constraint. Thus, combining the two approaches to support multiple resources that each have multiple constraints should represent a broad variety of resource allocation problems.

Social Foraging: The Ideal Free Distribution

The ideal free distribution (IFD) of social foraging theory was originally introduced by Fretwell and Lucas [34], and a review of recent biological advances in the theory

is given by Stephens et al. [115, Box 10.1 by Ian M. Hamilton]. Recently, engineering extensions of IFD theory have been used to solve autonomous resource allocation problems [e.g., 31, 66, 95, 96]. Here, we show that the IFD is a special case of Equation (5.1). Thus, the methods presented in this chapter can be used to find distributed IFD solutions. Furthermore, the viewing Equation (5.1) as a general IFD suggests other resource allocation problems.

In the basic IFD model, each of a group of $N \in \mathbb{N}$ foragers is free to move among n food patches. Food arrives at each food patch at some rate, and then that food is available to be distributed among the foragers within the patch. Thus, whenever a new forager enters a patch, the per-unit-time food available to each of the previous occupants of the group decreases. In the IFD model, it is assumed that each ideal forager that is free to move among these n patches has perfect knowledge of function $r_i : \{0, 1, \ldots, N\} \mapsto \mathbb{R}$ that maps the number of foragers in patch $i \in \{1, 2, \ldots, n\}$ to the rate of food available to each occupant of that patch. It is assumed that when the rate in one patch is higher than the rate in another patch, some sufficiently small number of animals immediately move from the higher-rate patch to the lower-rate patch so that the imbalance between all rates decreases to its minimum. Consequently, the patches with a relatively high food capacity will have a relatively high occupancy.

For analytical tractability, the integer programming problem of the IFD is approximated by the real optimization problem

$$\begin{aligned} \text{minimize} \quad & \max\{s_i(x_i) : i \in \{1, 2, \ldots, n\}\} \\ \text{subject to} \quad & x_1 + x_2 + \cdots + x_n = N \end{aligned} \tag{5.3}$$

where $x_i \in \mathbb{R}_{\geq 0}$ and the suitability function $s_i : [0, N] \mapsto \mathbb{R}$ is a continuous and strictly monotonic real extension of the rate function r_i for each $i \in \{1, 2, \ldots, n\}$.

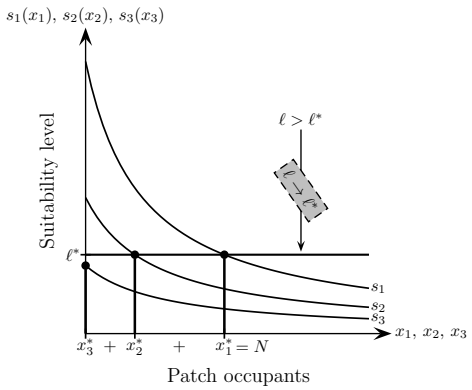


Figure 5.1: Graphical depiction of the IFD solution. For patches 1, 2, and 3, the suitability functions s_1, s_2, and s_3 are plotted on the same axes. A horizontal line is plotted at a suitability level $\ell \geq \max\{s_i(0) : i \in \{1, 2, 3\}\}$ and lowered until it reaches a level $\ell = \ell^*$ where $x_1^* + x_2^* + x_3^* \triangleq s_1^{-1}(\ell^*) + s_2^{-2}(\ell^*) + 0 = N$ where N is the total population size. Then $\vec{x}^* = [x_1^*, x_2^*, x_3^*]^\top$ represents the equilibrium IFD solution at the equilibrium suitability level ℓ^*.

Consequently, the equilibrium distribution $\vec{x}^* \triangleq [x_1, x_2, \ldots, x_n]^\top$ such that

$$\begin{cases} x_i^* = 0 & \text{if } s_i(0) < \ell^* \\ s_i(x_i^*) = \ell^* & \text{otherwise} \end{cases} \tag{5.4}$$

where $\ell^* \in \mathbb{R}_{\geq 0}$ is a constant representing the equilibrium suitability for all occupied patches. Because $x_1 + \cdots x_n = N > 0$, at least one patch must be occupied, and so the equilibrium is well defined. Furthermore, because $\ell^* \geq 0$, changing the equality constraint that $\sum_{i=1}^n x_i = N$ in Equation (5.3) to the the inequality $\sum_{i=1}^n x_i \leq N$ has no impact on the solution. Moreover, the minimax problem is equivalent to the maximin problem provided the inequality (or, equivalently, equality) constraint is such that $\sum_{i=1}^n x_i \geq N$.

The solution to the IFD problem is often depicted graphically as in Figure 5.1. In particular, the suitability functions are all plotted on the same axes. Initially, a horizontal line at suitability level ℓ is drawn above the maximum suitability level. As

that line is decreased, it eventually intersects each suitability function $i \in \{1, 2, \ldots, n\}$ at a level x_i. The IFD distribution $\vec{x}^* = [x_1, x_2, \ldots, x_n]^\top$ and suitability level $\ell^* = \ell$ when $x_1 + x_2 + \cdots + x_n = N$.

Numerically, IFD solution methods search the positive real half-line for ℓ^*. For each $i \in \{1, 2, \ldots, n\}$, the restricted inverse $s_i^{-1} : [0, s_i(0)] \mapsto [0, N]$ of the continuous and strictly monotonic suitability function s_i exists. Thus, at each numerical iteration, the suitability level candidate ℓ can be used to generate the corresponding candidate occupation level $x_i = s_i^{-1}(\ell)$ for each $i \in \{1, 2, \ldots, n\}$ where $s_i(0) > \ell$ ($x_i = 0$ otherwise). The suitability candidate ℓ converges on ℓ^* as the $x_1 + \cdots x_n$ converges on N.

In principle, the numerical solution to Equation (5.3) can be reduced from a search over the real half-line to a search over the N possible numbers of truncated patches. Consider the case when the suitability functions are ranked and shaped so that truncation of patch $i \in \{1, 2, \ldots, n\}$ implies truncation of patch $j \in \{i+1, \ldots, n\}$ for all $\vec{x} \in [0, N]^n$. By the ordering,

- Patch 1 will always be occupied. If $s_1(N) < s_2(0)$, then patch 2 cannot be truncated, and so patch 2 must be occupied as well. Otherwise, $x_1^* = N$ and $x_i^* = 0$ for all $i \in \{2, 3, \ldots, n\}$.

- If only patch 1 and patch 2 are occupied, they are occupied at candidate levels $x_1^+, x_2^+ \in [0, N]$ where $s_1(x_1^+) = s_2(x_2^+)$ such that $x_1^+ + x_2^+ = N$. Hence, candidate equilibrium suitability level ℓ^+ is such that $s_1^{-1}(\ell^+) + s_2^{-1}(\ell^+) = N$. In many cases, this equation can be solved for ℓ^+ analytically. If $\ell^+ < s_3(0)$, then this process must continue for the candidate set of patches 1, 2, and 3. Otherwise, $\ell^* = \ell^+$ and $x_1^* = s_1^{-1}(\ell^*)$ and $x_2^* = s_2^{-1}(\ell^*)$.

For example, if each patch $i \in \{1, 2, \ldots, n\}$ has a suitability function $s_i(x_i) = a_i/x_i$ where $a_i \in \mathbb{R}_{>0}$, then no patches will ever be truncated and $x_i^* = Na_i/(a_1 + a_2 + \cdots + a_n)$ where the equilibrium suitability ℓ^* is the average $(a_1 + a_2 + \cdots + a_n)/N$. A precise $O(N)$ algorithm for the case of generalized hyberbolic suitability functions is presented by Quijano and Passino [95].

IFD as Area Maximization: The equilibrium solution of the IFD can be summarized as

$$s_i(x_i^*) = \ell^* - \mu_i^* \tag{5.5}$$

for all $i \in \{1, 2, \ldots, n\}$ where $\mu_i^* \in \mathbb{R}_{\geq 0}$ such that $\mu_i^* = 0$ if $x_i^* = 0$. That is, there exists at least one occupied patch i so that $0 < s_i(x_i^*) = \ell^*$ is the equilibrium suitability level. A different patch $j \in \{1, 2, \ldots, n\}$ is either occupied at the same suitability level (i.e., $s_j(x_j^*) = \ell^*$) or truncated at a lower suitability level (i.e., $s_j(0) = \ell^* - \mu_i^* \leq \ell^*$). The n equations of the form of Equation (5.5) can then be collected into

$$[s_1(x_1^*), s_2(x_2^*), \ldots, s_n(x_n^*)]^\top = \ell^* - \mu_1^* \vec{e}_1 - \mu_2^* \vec{e}_2 - \cdots - \mu_n^* \vec{e}_n \tag{5.6}$$

which is the KKT characterization of the optimal point $\vec{x}^*$ to the problem

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \int_0^{x_i} s_i(\tau)\, \mathrm{d}\tau \\ \text{subject to} \quad & x_1 + x_2 + \cdots + x_n \leq N. \end{aligned} \tag{5.7}$$

Thus, the IFD that is traditionally viewed as a minimax problem over an equality constraint may also be viewed as maximizing the total area underneath the suitability curves subject to an inequality constraint that is always active. For example, a eusocial insect colony may need to maximize its total foraging gain, but it can allocate no more than N of its foragers. In the sequel, we revisit the definition of suitability

and show how a slightly different formulation suggests how to pick N for a given environment.

IFD as Special Minimization Case: The IFD can be recast in terms of price as opposed to suitability. That is, because suitabilities are effectively bounded away from zero, patch $i \in \{1, 2, \ldots, n\}$ with the highest suitability $s_i(x_i)$ may also be viewed as the patch with the lowest price $p_i : [0, N] \mapsto \mathbb{R}_{\geq 0}$ where $p_i(x_i) \triangleq 1/s_i(x_i)$. Thus, as the quantity of food demanded increases due to an increase in the number of foragers N, the equilibrium price $\lambda^* \triangleq 1/\ell^*$ increases. In particular, the equilibrium solution of the IFD can then be summarized as

$$\frac{1}{s_i(x_i^*)} \triangleq \boxed{p_i(x_i^*) = \lambda^* + \mu_i^*} \triangleq \frac{1}{\ell^*} + \mu_i^* \tag{5.8}$$

for all $i \in \{1, 2, \ldots, n\}$ where $\mu_i^* \in \mathbb{R}_{\geq 0}$ such that $\mu_i^* = 0$ if $x_i^* = 0$. This solution is depicted in Figure 5.2 for the same case from Figure 5.1 (i.e., the hyperbolic suitability functions are depicted as affine price functions). That is, there exists at least one occupied patch i so that $0 < p_i(x_i^*) = \lambda^*$ is the equilibrium price. A different patch $j \in \{1, 2, \ldots, n\}$ is either occupied at the same price (i.e., $p_j(x_j^*) = \lambda^*$) or truncated at a higher entry price (i.e., $p_j(0) = \lambda^* + \mu_i^* \geq \lambda^*$). The n equations of the form of Equation (5.8) can then be collected into

$$[p_1(x_1^*), p_2(x_2^*), \ldots, p_n(x_n^*)]^\top = \lambda^* + \mu_1^* \vec{e}_1 + \mu_2^* \vec{e}_2 + \cdots + \mu_n^* \vec{e}_n \tag{5.9}$$

which is the KKT characterization of the optimal point $\vec{x}^*$ to the problem

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} \int_0^{x_i} p_i(\tau) \, \mathrm{d}\tau \\ \text{subject to} \quad & x_1 + x_2 + \cdots + x_n \geq N. \end{aligned} \tag{5.10}$$

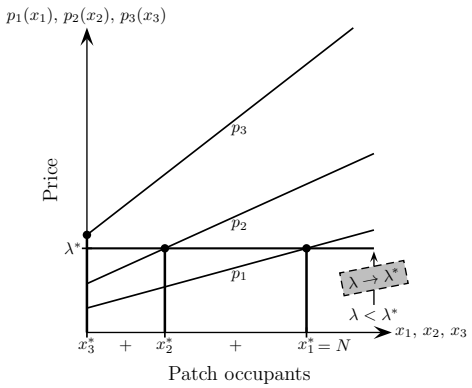


Figure 5.2: Graphical depiction of price-minimizing IFD solution. For patches 1, 2, and 3, the price (i.e., inverse suitability) functions p_1, p_2, and p_3 are plotted on the same axes. A horizontal line is plotted at a price $\lambda \leq \min\{p_i(0) : i \in \{1, 2, 3\}\}$ and raised until it reaches a price $\lambda = \lambda^*$ where $x_1^* + x_2^* + x_3^* \triangleq p_1^{-1}(\lambda^*) + p_2^{-2}(\lambda^*) + 0 = N$ where N is the minimum population size. Then $\vec{x}^* = [x_1^*, x_2^*, x_3^*]^\top$ represents the equilibrium IFD solution at the equilibrium price λ^*. Although the axes have been scaled, the solution here matches the one shown in Figure 5.1. In particular, the hyperbolic suitability functions are represented by affine price functions here.

Thus, this inequality realization of the IFD minimizes the cost (i.e., total price) required to satisfy a minimum N foragers. For example, given that the food requirements of N foragers will be serviced by n food patches, the N foragers will allocate themselves to minimize the price of each unit of food (e.g., they shift out of patches in order to minimize the time to wait to accumulate each unit of food). The inequality constrained optimization problem in Equation (5.10) matches the focal minimization problem of this chapter in Equation (5.1) with $m = 1$ and $\vec{a}_1 = [1, 1, \ldots, 1]^\top$.

Incorporating a Single Nutrient Constraint: The use of price (i.e., inverse suitability) functions to characterize food patches allows the IFD problem to be recast as a cost minimization problem of the form of Equation (5.1). However, this general structure also supports adding nutrient constraints to the IFD model. For example, a

eusocial insect colony may require a minimum level c_1 of a particular nutrient that can be found in one of the n food patches. Food patch $i \in \{1, 2, \ldots, n\}$ returns $a_{1i} \in \mathbb{R}_{\geq 0}$ units of the nutrient per forager allocated to the patch. If it is beneficial to meet the nutrient requirement with the fewest number of foragers, the nutrient-constrained IFD is the solution to

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} x_i \\ \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq c_1. \end{aligned} \tag{5.11}$$

Thus, this IFD projects the $[0, 0, \ldots, 0]^n$ origin onto the $\vec{a}_1^{\top}\vec{x} \geq c_1$ constraint space according to the $\|\cdot\|_1$ Manhattan distance. Because there is no upper bound on the number of occupants in a single patch, the equilibrium strategy allocates all necessary foragers to the food patch $i \in \{1, 2, \ldots, n\}$ with the highest return rate a_{1i}. Assuming that it is desirable to spread foragers across patches (e.g., to prevent a catastrophe at one patch from killing a large group that is responsible for returning all of the nutrient to the colony), then it may be desirable to

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} x_i^2 \\ \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq c_1, \end{aligned} \tag{5.12}$$

which does not depend on the assumption of a fixed population size N. This optimization problem uses the Euclidean distance $\|\cdot\|_2$ to project the origin onto the constraint space. Consequently, the equilibrium distribution populates all patches with foragers. Just as with Equation (5.11), the price functions are symmetric across the n food patches; for each patch $i \in \{1, 2, \ldots, n\}$ of the Euclidean projection problem in Equation (5.12), the price (i.e., inverse suitability) function $p_i(x_i) = \mathrm{d}(x_i^2)/\mathrm{d}x_i = 2x_i$. However, the asymmetric weighting of the constraint results in asymmetric occupancy levels. Thus, there can be no equilibrium price or suitability level. Hence, because

costs are not equalized across patches, it appears like the solution to Equation (5.12) cannot be an IFD. However, it is the case that the equilibrium solution $\vec{x}^*$ will be such that

$$\frac{2x_i^*}{a_{1i}} = \frac{2x_j^*}{a_{1j}} \tag{5.13}$$

for all $i, j \in \{1, 2, \ldots, n\}$. Moreover, for each $i \in \{1, 2, \ldots, n\}$,

$$x_i^* = c_1 \frac{a_{1i}}{a_{11}^2 + a_{12}^2 + \cdots + a_{1n}^2}.$$

Therefore, although prices (i.e., suitabilities) are not balanced, each weighted price function in is balanced as in Equation (5.13) at a level λ^* where

$$\frac{p_i(x_i^*)}{a_{1i}} = \lambda^* = \frac{2c_1}{a_{11}^2 + a_{12}^2 + \cdots + a_{1n}^2}$$

where $p_i(x_i) = 2x_i$ for all $i \in \{1, 2, \ldots, n\}$. Moreover, the solution to the nutrient-constrained Euclidean IFD problem in Equation (5.12) matches the solution to the classical IFD problem in Equation (5.7) with suitability and population defined as

$$s_i(x_i) = \frac{a_{1i}}{2x_i} \qquad \text{and} \qquad N = x_1^* + x_2^* + \cdots + x_n^* = c_1 \frac{a_{11} + a_{12} + \cdots + a_{1n}}{a_{11}^2 + a_{12}^2 + \cdots + a_{1n}^2}$$

where these suitabilities are balanced at an equilibrium level $\ell^* = 1/\lambda^*$. That is, N is the minimum number of foragers needed to meet the nutrient constraint at a uniform suitability level. So the IFD generated from nutrient-constrained minimization not only matches high occupation levels with high capacities, but it adjusts the required population size N automatically with changes in environmental parameters. It is unlikely that a eusocial insect colony always allocates a fixed number of workers to foraging; hence, it is attractive that this generalization of the IFD predicts both distribution and total population size. To test the predictions of this model, the number of colony workers allocated to foraging could be measured as the quality of patches is modulated.

Multiple Nutrient Constraints: In the preceding discussion, we showed how a cost minimization problem under a single nutrient constraint can be translated into an equivalent IFD problem where the minimum necessary population size is parameterized by the variables from the environment. Under a single nutrient constraint, a quantity analogous to suitability is equalized across all food patches. However, as we will now show, if multiple simultaneous nutrient constraints are added to processes that would normally generate balanced IFD solutions, the equilibrium distribution will not be balanced. Thus, it must remain in the framework of Equation (5.1).

Following the Euclidean projection example of Equation (5.12), let there be $m \in \mathbb{N}$ nutrients where a eusocial insect colony must acquire at least $c_j \in \mathbb{R}_{\geq 0}$ of nutrient j across all n patches. Each food patch $i \in \{1, 2, \ldots, n\}$ is assumed to return $a_{ji} \in \mathbb{R}_{\geq 0}$ of nutrient j per each unit forager occupied in the patch. Thus, the colony must allocate foragers to

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} x_i^2 \\ \text{subject to} \quad & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq c_1, \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq c_2, \\ & \qquad \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq c_m, \end{aligned} \tag{5.14}$$

which matches Equation (5.1) with $F(\vec{x}) = \|\vec{x}\|_2^2$ and $\vec{a}_j = [a_{j1}, a_{j2}, \ldots, a_{jn}]^\top$ for each $j \in \{1, 2, \ldots, m\}$. Thus, the equilibrium solution $\vec{x}^*$ is such that

$$2x_i^* = \lambda_1^* a_{1i} + \lambda_2^* a_{2i} + \cdots + \lambda_m^* a_{mi} + \mu_i^*$$

where $\lambda_j^* \in \mathbb{R}_{\geq 0}$ and $\mu_i^* \in \mathbb{R}_{\geq 0}$ for each $i \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, m\}$. Because each price function vanishes at the origin, there will be no truncated patches

and so $\mu_i^* \equiv 0$. More importantly, because of the multiple constraints, there will be no uniform equilibrium price (nor suitability) level across all n patches. If the coupling between nutrient constraint vectors is weak (i.e., the patches that contribute much of one constraint contribute very little of any other), then there will be price (i.e., inverse suitability) shear between groups of patches; in particular, the patches most strongly associated with each nutrient will come to a weighted price (and weighted suitability) equilibrium for only that group. Assuming that eusocial insect colonies do have multiple nutrient constraints that come from a single set of food patches, allocations may appear to violate predictions of the IFD even though the foraging behaviors are nonetheless consistent with minimum suitability maximization (i.e., maximum price minimization). Combining these ideas with the several-population IFD approach (i.e., multiple resources that each have one constraint) of Moore et al. [66] and Quijano and Passino [96] should lead to a very general model of animal foraging distributions (and thus resource allocation in general).

Economic Power Dispatch

The basic economic dispatch problem in power engineering is summarized by Bergen and Vittal [17]. There are $n \in \mathbb{N}$ generators that generate the $P_G \in \mathbb{R}_{\geq 0}$ power demanded by a given community. At each generator $i \in \{1, 2, \ldots, n\}$, the cost $C_i(P_i)$ of generating $P_i \in \mathbb{R}_{\geq 0}$ units of power is assumed to come from the convex function $C_i : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$. Thus, the optimal allocation $\vec{P}^* = [P_1, P_2, \ldots, P_n]^\top$ is the solution to the problem

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} C_i(P_i) \\ \text{subject to} \quad & P_1 + P_2 + \cdots P_n = P_G. \end{aligned} \tag{5.15}$$

However, this problem is identical to the price-IFD in Equation (5.10). In fact, Bergen and Vittal provide explanatory figures similar to Figure 5.2 without any reference to the IFD. Thus, the methods from this chapter are amenable to the distributed solution of the economic power dispatch problem. Moreover, the multiple-constraint foraging discussion motivates other economic power dispatch problems. For example, consider $m = 2$ communities, and let community $j \in \{1, 2\}$ demand P_{Gi} units of power and receive a_{ji} fraction of its power from generator $i \in \{1, 2, \ldots, n\}$. Due to network effects (e.g., distance, connectivity), the vector $\vec{a}_1 \neq \vec{a}_2$. Thus, the extension of Equation (5.15) for this problem is to

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} C_i(P_i) \\ \text{subject to} \quad & a_{11}P_1 + a_{12}P_2 + \cdots a_{1n}P_n \geq P_{G1}, \\ & a_{21}P_1 + a_{22}P_2 + \cdots a_{2n}P_n \geq P_{G2}, \end{aligned} \tag{5.16}$$

where one or both constraints may be active. When both constraints are active, the optimal solution is the point along the intersection of the two hyperplanes that minimizes the cost function. The solution to this problem cannot be solved using the IFD-like methods of Bergen and Vittal; however, it does have the form of Equation (5.1).

Intelligent Lighting

In the built environment, light is usually provided by several overhead artificial sources as well as windows that provide variable levels of natural light. Intelligent lighting systems have been proposed that use precise control of individual lights to meet occupant preferences. Although some schemes are designed to control the color of light [e.g., 12], most approaches are ostensibly for reducing the power used by

artificial sources [e.g., 40, 63, 77, 124]. However, few power-saving methods are constructed within an optimization framework; instead, they save power solely by eliminating preference surpluses (i.e., no optimization is done within the user preference hyperplane). Thus, recent intelligent lighting research focuses on measuring, meeting, and maintaining user preferences [e.g., 40, 77]. Those that do include a power or energy component into their algorithms are either *ad hoc* [63] or take the optimization procedure for granted [124]. For example, Wen and Agogino [124] suggest an allocation policy for $n \in \mathbb{N}$ lights above $m \in \mathbb{N}$ sensors that sets the light output $d_i \in [\underline{d}_i, \overline{d}_i] \subseteq \mathbb{R}_{\geq 0}$ on each light $i \in \{1, 2, \ldots, n\}$ according to the solution $\vec{d}^* = [d_1, d_2, \ldots, d_n]^\top$ of the problem

$$\begin{aligned} \text{minimize} \quad & \|\vec{d}\|_1 \\ \text{subject to} \quad & L\,\vec{d} = \vec{E}. \end{aligned} \tag{5.17}$$

where the matrix $L \in \mathbb{R}_{\geq 0}^{m \times n}$ represents the relative influence of each of the n lights on the m sensors, and $\vec{E} \in \mathbb{R}_{\geq 0}^{m}$ represents the desired level on those sensors. However,

- No claims are made about the existence of a solution to the linear programming problem. When a feasible solution is not available, their approach changes the equality constraint to two inequality constraints that bound the sensor readings within a tolerance region of $\vec{E}$, and that tolerance region grows until feasibility is returned to the system. However, it is nonsensical to formulate this intelligent lighting optimization problem with an upper bound on the sensor readings. If the goal is reduction of energy use and certain minimum lighting constraints must be met, then some constraints must be allowed to deactivate. Alternatively, hybrid actuators that can both provide light and remove it (e.g., by converting excess natural or artificial light into electricity) can in principle

maintain the equality constraints; however, these actuators are not part of this formulation. Moreover, because the tolerance adjustment procedure is two sided and symmetric, it can lead to large areas of darkness as some lights attempt to compensate for the positive biases provided by natural light.

- The choice of the Manhattan distance $\|\cdot\|_1$ as the objective function to minimize is unjustified. As discussed in the foraging examples above, 1-norm minimization leads to small clusters of high occupancy even though other allocations exist that reduce power use and point-source effects. Additionally, as discussed in Section 5.1.3, minimizing the Euclidean distance $\|\cdot\|_2$ subject to the inequality constraint $L\vec{d} \geq \vec{E}$ finds the least-squares fit $\vec{d}^*$ to the system of equations $L\vec{d} = \vec{E}$. Hence, the need for a tolerance growth procedure is mitigated by choosing minimizing the Euclidean norm (i.e., the instantaneous power) instead.

- A centralized lighting algorithm that distributes its answers periodically to slave lights is used. Decentralized methods are proposed that duplicate the optimization procedure on every node of the network and require cyclic computation. It is suggested that the linear programming can be done collaboratively among the actuators; however, no mechanisms for this distributed computation are given.

So intelligent lighting research is presently a vacuum for rigorous distributed optimization results. Here, we formulate an intelligent lighting optimization problem similar to Equation (5.17) of the form of Equation (5.1). Hence, the distributed optimization methods presented in this chapter are amenable to intelligent lighting. Furthermore, we use intelligent lighting as a motivating example in the description of the algorithm behavior.

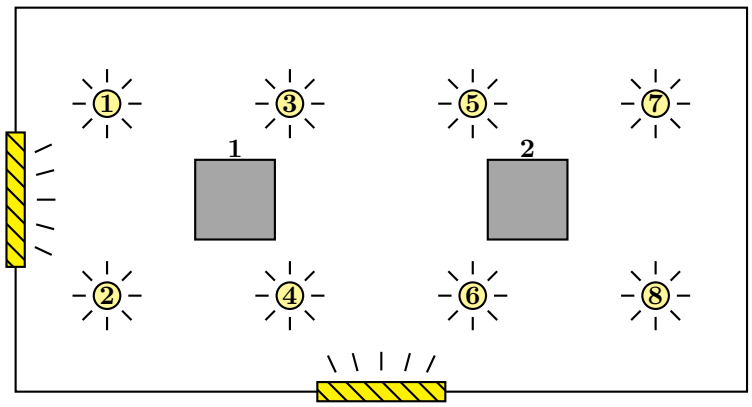


Figure 5.3: Top view of prototypical lighting system with $n = 8$ lights and $m = 2$ sensors. Lights, which are shown as circles around their identities, are mounted in the ceiling of the room. Sensors, which are shown as patches underneath their identities, are located at a distance beneath the lights. There are two disturbance sources (e.g., windows) shown as hatched rectangles.

Optimal Intelligent Lighting: Let there be $n \in \mathbb{N}$ lights positioned above $m \in \mathbb{N}$ sensors. The lights and sensors can be in any position, but one possible configuration is the room depicted in Figure 5.3. In general,

- Each light $i \in \{1, 2, \ldots, n\}$ is actuated by control $x_i \in [\underline{x}_i, \overline{x}_i] \subseteq \mathbb{R}_{\geq 0}$ where $\underline{x}_i, \overline{x}_i \in \mathbb{R}_{\geq 0}$ with $\overline{x}_i \geq \underline{x}_i \geq 0$.

- For each sensor $j \in \{1, 2, \ldots, m\}$, if the control for light $i \in \{1, 2, \ldots, n\}$ increases from x_i to $x_i + \delta_i$ with $\delta_i \in \mathbb{R}_{>0}$ while all other lighting controls remain constant, the reading of sensor j increases by $a_{ji}\delta_i$.

- Associated with each sensor $j \in \{1, 2, \ldots, m\}$ is a minimum sensor reading $\gamma_j \in \mathbb{R}_{\geq 0}$ and a positive sensor bias $b_j \in \mathbb{R}_{\geq 0}$ due to an exogenous light source (e.g., the sun); it is taken for granted that b_j is constant or changes at a slow rate relative to the speed of the control system. Assuming that $\vec{a}_j$ and the

present value of control variables $\vec{x}$ is available to sensor $j \in \{1, 2, \ldots, m\}$, the value of bias b_j can be estimated by subtracting the expected sensor value from the actual sensor value. Thus, the effective minimum constraint $c_j \triangleq \gamma_j - b_j$ must be supplied by the n lights for each sensor $j \in \{1, 2, \ldots, m\}$.

So the intelligent lighting system must solve Equation (5.1) where $A \triangleq [\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_m]^\top$ and cost function F is chosen by the system designer. This chapter considers a general class of objective functions; however, a natural choice is to minimize the instantaneous power $F(\vec{x}) = \|\vec{x}\|_2^2$. Mathematically, this choice projects the origin (i.e., all lights off) onto the constraint set using the Euclidean distance. Moreover, for each $i \in \{1, 2, \ldots, n\}$, if x_i represents a control voltage (e.g., the RMS voltage of a dimmed AC power signal) across a linear lighting element, this policy will minimize the power used by a group of identical lights. However, even if x_i is a reference variable for an inner-loop control system on the light (e.g., the input to a dimmable fluorescent ballast or LED control system as opposed to the RMS voltage across an incandescent light), minimization of $\|\cdot\|_2^2$ will create diffuse pools of light that still may likely use less power than single point sources.

5.1.3 Conventional Dual-Space Optimization Methods

The conventional method for parallelizing non-separable optimization problems is to parallelize the dual optimization problem that may have greater separability [19]. We study this approach for Equation (5.1) here. For simplicity, the upper and lower bounds on each $\vec{x}$ will not be treated explicitly here; however, matrix A and constraint vector $\vec{c}$ can be augmented to include them. Although the minimization problem in Equation (5.1) has a polyhedral constraint set with edges that are oblique in general,

the dual problem to

$$\begin{aligned} \text{maximize} \quad & \inf_{\vec{x}\in\mathcal{X}} \left(F(\vec{x}) + \vec{\lambda}^\top(\vec{c} - A\vec{x})\right) \\ \text{subject to} \quad & \vec{\lambda} \geq [0, 0, 0, \ldots, 0]^\top \end{aligned} \tag{5.18}$$

has a constraint set that is a Cartesian product of real half spaces. In principle, each of the m separable half spaces can be assigned to a different distributed agent for parallelized computation. However, the assignment of the m separable spaces to the n agents may be non-trivial especially when $m \neq n$. Additionally, computations in each space may still depend upon computations in another space. Parallelized computations that must be completed sequentially do not capitalize on the benefits of parallelization; in fact, due to the extra communication overhead, such problems may be better solved on a single agent. Here, we consider how dual space optimization methods may be used with the Euclidean distance $\|\cdot\|_2$ and the Manhattan distance $\|\cdot\|_1$.

Euclidean Minimization as Pseudoinverse Generator

Consider Equation (5.1) with the $F(\vec{x}) = \|\vec{x}\|_2^2 = \vec{x}^\top\vec{x}$ objective function. So the primal problem is to

$$\begin{aligned} \text{minimize} \quad & \vec{x}^\top\vec{x} \\ \text{subject to} \quad & A\,\vec{x} \geq \vec{c}. \end{aligned}$$

The dual of this problem is to

$$\begin{aligned} \text{maximize} \quad & \inf_{\vec{x}\in\mathcal{X}} \left(\vec{x}^\top\vec{x} + \vec{\lambda}^\top(\vec{c} - A\vec{x})\right) \\ \text{subject to} \quad & \vec{\lambda} \geq [0, 0, 0, \ldots, 0]^\top \end{aligned} \tag{5.19}$$

where Lagrange multiplier vector $\vec{\lambda} \triangleq [\lambda_1, \lambda_2, \ldots, \lambda_m]^\top \in \mathbb{R}^m_{\geq 0}$. This problem is more attractive for parallelization because its constraint space is separable. Moreover, for this particular dual problem, for any multiplier $\vec{\lambda}$, the quadratic argument of the

infimum has the unique minimum

$$\vec{x}^* = \frac{1}{2} A^\top \vec{\lambda}$$

which corresponds to the KKT condition that the gradient $2\vec{x}$ is a conical combination $A^\top \vec{\lambda}$ of the vectors normal to each constraint space. Applying this minimum to Equation (5.19) yields

$$\begin{aligned} \text{minimize} \quad & \frac{1}{4} \vec{\lambda}^\top A A^\top \vec{\lambda} - \vec{c}^\top \vec{\lambda} \\ \text{subject to} \quad & \vec{\lambda} \geq [0, 0, 0, \ldots, 0]^\top \end{aligned}$$

which itself is a quadratic program minimized over the positive orthant of the multiplier space. Distributed methods exist (e.g., projected gradient descent) for solving this constrained problem, but their amenability for parallelization depends on the structure of the A matrix [19]. For illustrative purposes, assume that $\lambda_j^* > 0$ for each $j \in \{1, 2, \ldots, m\}$ (i.e., all constraints in the primal problem solution are active). Then this quadratic problem has unconstrained minimum $\lambda^* = 2(AA^\top)^{-1}\vec{c}$. So

$$\vec{x}^* = \frac{1}{2} A^\top \vec{\lambda}^* = \overbrace{A^\top (AA^\top)^{-1}}^{\substack{\text{Moore–Penrose} \\ \text{pseudoinverse of } A}} \vec{c}.$$

where the overbraced expression is the psuedoinverse of A. That is, $\vec{x}^*$ is the least-squares fit to the system of equations $A\vec{x} = \vec{c}$. However, even if A has a sparse separable structure, calculation of the pseudoinverse of A generally cannot leverage the benefits of parallel computation.

Manhattan Minimization with Euclidean Proximate

Consider Equation (5.1) with the $F(\vec{x}) = \|\vec{x}\|_1 = \sum_{i=1}^n \vec{x} = \vec{1}^\top \vec{x}$ objective function where $\vec{1} \triangleq \sum_{i=1}^n \vec{e}_i = [1, 1, \ldots, 1]^\top \in \{1\}^n$. So the primal problem is to

$$\begin{aligned} \text{minimize} \quad & \vec{1}^\top \vec{x} \\ \text{subject to} \quad & A\vec{x} \geq \vec{c}. \end{aligned}$$

The dual of this problem is to

$$\begin{aligned} \text{maximize} \quad & \inf_{\vec{x} \in \mathcal{X}} \left(\vec{1}^\top \vec{x} + \vec{\lambda}^\top (\vec{c} - A\vec{x}) \right) \\ \text{subject to} \quad & \vec{\lambda} \geq [0, 0, 0, \ldots, 0]^\top, \end{aligned} \tag{5.20}$$

but this problem is considerably more difficult to solve due to the primal cost function not being strictly convex. However, an iterative method can transform this problem into a sequence of quadratic minimizations. In particular, if the primal problem is re-cast as

$$\begin{aligned} \text{minimize} \quad & \vec{1}^\top \vec{x} + \gamma (\vec{x} - \vec{y})^\top (\vec{x} - \vec{y}) \\ \text{subject to} \quad & A\,\vec{x} \geq \vec{c} \end{aligned}$$

where $\gamma \in \mathbb{R}_{\geq 0}$ and $\vec{y} \in \mathcal{X}$ is an estimate of optimal solution $\vec{x}^*$, then the same method that was used to find the optimal Euclidean projection can be used. After each iteration, the estimate $\vec{y}$ is updated with the optimal result from the previous iteration. This method converges on an optimal solution to the primal problem; however, it is not any more parallelizable as the dual space method for the Euclidean minimization.

5.2 Parallelizable Primal-Space Algorithm

The parallel solver of Equation (5.1) that we introduce here depends on a special monotonicity present in the examples discussed in Section 5.1.2. In particular, along with the configuration space $\mathcal{X} = \prod_{i=1}^{n} [\underline{x}_i, \overline{x}_i]$, also define origin space $\mathcal{X}_0 \triangleq \prod_{i=1}^{n} [0, \overline{x}_i]$ as the convex Cartesian extension of $\mathcal{X}$ to include the origin. The function F is replaced with its continuous extension to this new space $F : \mathcal{X}_0 \mapsto \mathbb{R}$.

Hence, Equation (5.1) is

$$\begin{aligned} \text{minimize} \quad & F(\vec{x}) \\ \text{subject to} \quad & A\,\vec{x} \geq \vec{c} \\ & E\,\vec{x} \geq [\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_n]^\top \end{aligned} \tag{5.21}$$

where the elementary matrix $E \triangleq [\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n]^\top$.

It has already been assumed that F is convex and continuously differentiable. Here, we also assume that each component of the gradient F is monotonically increasing. That is, for each $i \in \{1, 2, \ldots, n\}$ and $\vec{x}, \vec{y} \in \mathcal{X}_0$,

$$x_i \geq y_i \iff \nabla_i F(\vec{x}) \geq \nabla_i F(\vec{y}).$$

Consequently, $\nabla F(\vec{x}) \geq \nabla F(\vec{0})$ for all $\vec{x} \in \mathcal{X}$ where $\vec{0} \triangleq [0, 0, 0, \ldots, 0]^\top$ is the origin. Moreover, excluding the pathological case where F is constant, if $\nabla F(\vec{x}) = \vec{0}$, then $\vec{x} = \vec{0}$. Thus, optimization problem in Equation (5.21) is equivalent to projecting the origin $\vec{0}$ onto the constraint set $\mathcal{X}$ along the cost function F. The origin itself can only be a solution to the optimization problem if all constraints are inactive (i.e., $\vec{c} = \vec{0}$); otherwise, at least one constraint will be active.

5.2.1 Lighting Agents

Using the lighting discussion in Section 5.1.2 as a motivating example, we develop the distributed optimization algorithm here in the context of intelligent lights. In particular, let there be $n \in \mathbb{N}$ lighting agents and $m \in \mathbb{N}$ sensor agents that each have access to the n lighting control variables (e.g., voltages) $\vec{x} \triangleq [x_1, x_2, \ldots, x_n]^\top \in \mathcal{X}$. There is also a sufficiently small parameter $\delta \in \mathbb{R}_{>0}$ that is chosen such that $\delta < c_j / (\vec{a}_j^\top \vec{1})$ for all $j \in \{1, 2, \ldots, m\}$. This parameter will be used in the algorithms

defined below; roughly, it represents a trade between algorithm convergence speed (i.e., high δ) and accuracy (i.e., low δ).

In principle, several different implementations of the discrete-time systems described here are possible. The salient feature of this algorithm is that the actual lighting levels are a shared memory of the system, and elements of that shared memory can be increased or decreased in very specific ways. Likewise, in the following discussion, we assume that there are *lighting agents* that can only decrease the value of a single light associated to them and *sensor agents* that can increase the value of all lights. That is,

- Each lighting agent $i \in \{1, 2, \ldots, n\}$ has the ability to decrease the value of component x_i until that value is truncated at its lower bound $\underline{x}_i$.
- Each sensor agent $j \in \{1, 2, \ldots, m\}$ has the ability to increase the value of all components of the vector $\vec{x}$ until they are each truncated at their upper bounds.

However, it is only the scheduling of the interaction with the shared memory that is important. For example, other valid implementations may only use sensor agents to broadcast sensor readings to lighting agents that perform both types of access.

Commissioning

In the discussion below, each sensor $j \in \{1, 2, \ldots, m\}$ is assumed to have access to its constraint normal vector $\vec{a}_j$. For example, lighting agent i is implemented as a part of the control mechanism for component x_i, and that mechanism also services requests from each sensor agent. During a pre-runtime discovery mode of influence matrix A, each component x_i is set at two non-zero test levels. If only one component changes at a time and the identity of that light is broadcast instantaneously to all

sensors, then each sensor $j \in \{1, 2, \ldots, m\}$ can determine its influence vector $\vec{a}_j$ using the detected changes in its sensor readings.

Timing

Each agent is modeled by a discrete-time system acting at generally independent time steps. That is, light agent $i \in \{1, 2, \ldots, n\}$ acts at the infinite set of times $\mathcal{T}^{\ell i} \triangleq \{t_1^{\ell i}, t_2^{\ell i}, \ldots\}$ where $t_k^{\ell i} \in \mathbb{R}_{\geq 0}$ and $t_j^{\ell i} > t_k^{\ell i}$ for all $j, k \in \mathbb{N}$ with $j > k$. Likewise, sensor agent $j \in \{1, 2, \ldots, m\}$ acts at the infinite set of times $\mathcal{T}^{sj}$ which is defined in the analogous way. Let the set of times $\mathcal{T} \triangleq \{t_1, t_2, \ldots\} = \bigcup_{i=1}^{n} \mathcal{T}^{\ell i} \cup \bigcup_{j=1}^{m} \mathcal{T}^{sj}$ for which the lighting variable $\vec{x}$ is updated by one or more agents. As with the individual agent action times, if $j < k$ and $t_j, t_k \in \mathcal{T}$, then $t_j < t_k$. Thus, the $\vec{x}$ system evolves as the discrete-time equation $\vec{x}[k+1] = G(\vec{x}[k])$ where the notation $\vec{x}[k]$ represents the value of $\vec{x}$ at time $t_k \in \mathcal{T}$. Furthermore, the notation $[k]$ alone represents the time $t_k \in \mathcal{T}$ corresponding to the kth update of the $\vec{x}$ vector. Whether each sensor or lighting agent must know some or all of the components of the vector $\vec{x}$ at each time $[k]$ depends on the particular algorithm. However, each sensor $j \in \{1, 2, \ldots, m\}$ can detect its accurate sensor reading $\vec{a}_j^{\top} \vec{x}[k]$ at each time $[k]$.

In this chapter, we focus on two related timing schedules.

- We first consider the *sequential* case before generalizing to other timing schedules. In the sequential case,
  - For any $i, n \in \{1, 2, \ldots, n\}$ and $j, o \in \{1, 2, \ldots, m\}$, the sets $\mathcal{T}^{\ell i}$, $\mathcal{T}^{\ell n}$, $\mathcal{T}^{sj}$, and $\mathcal{T}^{so}$ are all disjoint.

– If the action at time $[k]$ is due to a given agent and there are m sensors and n lights (i.e., $m+n$ total agents), the action at time $[k+m+n]$ will also be due to that agent.

Consequently, the vector $\vec{x}$ is only modified by one agent at a time (i.e., there are no simultaneous events), and the order of the $n+m$ updates is fixed and periodic. For example, this case follows from the assumption that there exists a $\Delta_t \in \mathbb{R}_{>0}$ such that $t_{k+1}^{\ell i} = t_k^{\ell i} + \Delta_t$ and $t_{k+1}^{sj} = t_k^{sj} + \Delta_t$ for each lighting agent $i \in \{1, 2, \ldots, n\}$ and each sensor $j \in \{1, 2, \ldots, m\}$, and that the initial action time of each agent is different.

- The *sequentially simultaneous* case is a modification to the sequential case to allow simultaneous events. In the sequentially simultaneous case,

  – For each time $[k]$, $\mathcal{A}^\ell[k]$ is the set of all lighting agents that run simultaneously, and $\mathcal{A}^s[k]$ is the set of all sensor agents that run simultaneously.

  – There exists $M \in \{1, 2, \ldots, m+n\}$ such that for any $[k]$, $\mathcal{A}^\ell[k] = \mathcal{A}^\ell[k+M]$ and $\mathcal{A}^s[k] = \mathcal{A}^s[k+M]$ and $\mathcal{A}^\ell[k] \cap \mathcal{A}^\ell[o] = \mathcal{A}^s[k] \cap \mathcal{A}^s[o] = \emptyset$ for any $o \in \{1, 2, \ldots, M\}$.

  Consequently, simultaneous events can occur at any time, but the ordering of events is consistent. For example, this case follows from the assumption that there exists a $\Delta_t \in \mathbb{R}_{>0}$ such that $t_{k+1}^{\ell i} = t_k^{\ell i} + \Delta_t$ and $t_{k+1}^{sj} = t_k^{sj} + \Delta_t$ for each lighting agent $i \in \{1, 2, \ldots, n\}$ and each sensor $j \in \{1, 2, \ldots, m\}$, but that the initial action time of each agent may be the same as a set of other agents.

5.2.2 Motivation: Optimization by Normal Support of Variable Gravity

Before we introduce our focal distributed optimization algorithm, we discuss the germ of a related *gravitational gradient* algorithm. We will use it as a foil to highlight important features of our algorithm. In the sequential case of the gravitational gradient algorithm,

- If lighting agent $i \in \{1, 2, \ldots, n\}$ acts at time $[k+1]$, then

$$\vec{x}[k+1] = \vec{x}[k] - d_i[k] \nabla_i F(\vec{x}) \vec{e}_i. \tag{5.22}$$

  This behavior requires that each lighting behavior have access to every component of $\vec{x}$ at time $[k]$. However, if it is assumed that $f_i'(x_i) \triangleq \nabla_i F(\vec{x})$ only depends on component x_i for each $i \in \{1, 2, \ldots, n\}$, then each lighting agent i only needs access to component x_i at time $[k]$. Unless otherwise noted, the scalar $d_i[k] \equiv \delta$.

- If sensor $j \in \{1, 2, \ldots, m\}$ acts at time $[k+1]$, then

$$\vec{x}[k+1] = \vec{x}[k] + \begin{cases} \sigma_j[k]\, \vec{a}_j & \text{if } \vec{a}_j^\top \vec{x}[k] \leq c_j, \\ 0 & \text{otherwise.} \end{cases} \tag{5.23}$$

  where $\sigma_j[k] > 0$. For example, $\sigma_j[k] \triangleq (c_j - \vec{a}_j^\top \vec{x}[k])/\|\vec{a}_j\|_2^2$ ensures that $\vec{a}_j^\top \vec{x}[k+1] = c_j$.

This behavior mimics the behavior of a falling object in a gravitational field with force proportional to the gradient ∇F. The gravitational energy (or height) of the object is analogous to the value of the cost function. Hence, a uniform gravitational field is analogous to a linear cost function. The constraint vectors are normal to planes in this space. The KKT conditions of the system require that at equilibrium, the gradient

is a conical combination of the constraint vectors which is parameterized by the corresponding Lagrange multipliers. Likewise, at static equilibrium when an object is at rest, the gravitational force on it must be balanced by a positive combination of the forces normal to its supporting surfaces. The distributed behavior above mimics this idea by allowing the control vector $\vec{x}$ to descend along the gradient until being supported in a direction normal to each active constraint.

5.2.3 The MultiIFD: Optimization Under Uniform Gravity

Here, we assume that there exists some $b_i \in \mathbb{R}_{>0}$ such that $\nabla_i F(\vec{x}) \geq b_i > 0$ for each $i \in \{1, 2, \ldots, n\}$ (e.g., this assumption is met if the lower bound $\underline{x}_i > 0$ for all $i \in \{1, 2, \ldots, n\}$). In the sequential case for this *MultiIFD* algorithm,

- If lighting agent $i \in \{1, 2, \ldots, n\}$ acts at time $[k+1]$, then

$$\vec{x}[k+1] = \vec{x}[k] - d_i[k]\, \vec{e}_i \tag{5.24}$$

  where $d_i[k] \equiv \delta$ unless otherwise noted. Thus, the lighting agent may operate without knowledge of the gradient or the states of the other lights.

- If sensor $j \in \{1, 2, \ldots, m\}$ acts at time $[k+1]$, then

$$\vec{x}[k+1] = \vec{x}[k] + \begin{cases} \sigma_j[k]\, \vec{v}_j & \text{if } \vec{a}_j^\top \vec{x}[k] \leq c_j, \\ 0 & \text{otherwise} \end{cases} \tag{5.25}$$

  where direction

$$\vec{v}_j[k] \triangleq \left[\frac{a_{j1}}{\nabla_1 F(\vec{x}[k])}, \quad \frac{a_{j2}}{\nabla_2 F(\vec{x}[k])}, \quad \cdots \quad \frac{a_{jn}}{\nabla_n F(\vec{x}[k])} \right]^\top \tag{5.26}$$

  and $\sigma_j[k] > 0$. For example, $\sigma_j[k] \triangleq (c_j - \vec{a}_j^\top \vec{x}[k])/(\vec{a}_j^\top \vec{v}_j[k])$ ensures that $\vec{a}_j^\top \vec{x}[k+1] = c_j$.

For each $j \in \{1, 2, \ldots, m\}$, the component v_{ji} represents the suitability of light $i \in \{1, 2, \ldots, n\}$ under the assumption that constraint j is the only active constraint. That is, each sensor allocates its next lighting increase proportional to each suitability. Thus, the collection of m parallel sensors is the collective action of multiple single-constraint IFD solvers.

The MultiIFD algorithm can be derived from the gravitational gradient algorithm. In particular, for dimension $i \in \{1, 2, \ldots, n\}$, if x_i motion in the gravitational gradient space is scaled by $\nabla_i F(\vec{x})$, then gravitational gradient motion is restored. Furthermore, the MultiIFD algorithm transforms a non-linear optimization with linear constraints into an equivalent linear optimization with non-linear constraints.

In both algorithms, the lighting constraints represent the boundary between a decaying behavior driven by the lighting agents and a growth behavior driven by the switched-mode sensor agents. A good algorithm design will have functions $d[k]$ and $\sigma[k]$ that maintain a narrow boundary between these two regions where solutions are guaranteed to be attracted to the optimal solution to Equation (5.1). Although the existence of such convergent behaviors can be demonstrated in both algorithms, control of non-equilibrium (i.e., speed) and equilibrium (i.e., ultimate error bounds) characteristics is simpler to implement in the MultiIFD case. For example, the decay rate in the gravitational gradient algorithm varies and is unknown by the sensors without coordination or inference; consequently, choosing the sensor scalar $\sigma[k]$ sufficiently large to ensure both below-constraint growth and sufficiently small to ensure small ultimate error bounds is nontrivial and may require ongoing coordination with the lights. In contrast, the MultiIFD decay rate is a fixed aspect of the background environment and is thus is more easily compensated for by the sensor responses. In

both cases, the sensors implicitly require information about system gradient. However, in the MultiIFD case, the lighting agents have a much simpler implementation.

Constraint-responsive behaviors: When components of $\vec{x}$ are sufficiently large to deactivate all constraints and thus switch off all sensor responses, the decay rate of each light can be increased to increase the sensitivity of the system. One such mechanism increases $d_i[k]$ on lighting agent $i \in \{1, 2, \ldots, n\}$ after several agent cycles. When constraints are reached and sensors are switched back on, the $d_i[k]$ will return to its nominal values. Similarly, the sensor scalar $\sigma_j[k]$ on sensor j can be adjusted to slow movement toward constraints when sufficiently far from each constraint.

5.2.4 Stability of the MultiIFD

Here, we demonstrate that the MultiIFD algorithm forces trajectories of the system to remain within the vertices of a hypercube that continually approaches an enters a bounded region around the equilibrium $\vec{x}^*$ of Equation (5.21). The theoretical work in this section focuses on the case when a single constraint is active; however, the result can be extended to multiple constraints as shown in Section 5.3.

Unless otherwise noted, the following results use the sequentially simultaneous timing schedule with period M. It is assumed that $j \in \{1, 2, \ldots, m\}$ is the single active constraint. That is, either $m = 1$ or for every $\ell \in \{1, 2, \ldots, m\} - \{j\}$, $\vec{a}_\ell^\top \vec{x}[k] \geq c_\ell$; likewise, it is assumed that the trajectory is sufficiently far from component bounds to cause truncation. The next cycle where the state $\vec{x}$ is updated by sensor agent j is $[k+1]$. Additionally, the MultiIFD scalar

$$\sigma_j[k] \triangleq \frac{c_j - \vec{a}_j^\top \vec{x}[k]}{\vec{a}_j^\top \vec{v}_j[k]}.$$

Fixed-hypercorner Characterization of Solutions

If $\vec{x}[k]$ is such that $\vec{a}_j^\top \vec{x}[k] > c_j$, then a no sensors will be active and the consistent action of the light agents will eventually reduce sensor j to its constraint. Hence, assume that $\vec{x}[k]$ be such that $\vec{a}_j^\top \vec{x}[k] \leq c_j$. Then, at the event time $[k+M]$ when sensor j after all other agents have also executed,

$$\begin{aligned}\vec{a}_j^\top \vec{x}[k+M] &= \vec{a}_j^\top \left(\vec{x}[k] + \frac{c_j - \vec{a}_j^\top \vec{x}[k]}{\vec{a}_j^\top \vec{v}_j[k]} \vec{v}_j[k] - \vec{1}\delta \right) \\ &= \vec{a}_j^\top \vec{x}[k] + \vec{a}_j^\top \frac{c_j - \vec{a}_j^\top \vec{x}[k]}{\vec{a}_j^\top \vec{v}_j[k]} \vec{v}_j[k] - \vec{a}_j^\top \vec{1}\delta \\ &= c_j - \vec{a}_j^\top \vec{1}\delta.\end{aligned}$$

That is, the state $\vec{x}[k+M]$ will be an element of the hyperplane $\mathcal{P}_j \triangleq \{\vec{x} \in \mathcal{X} : \vec{a}_j^\top \vec{x} = c_j - \vec{a}_j^\top \vec{1}\delta\}$, which is parallel to the constraint boundary for sensor j. The system will return to line every $[k+NM]$ time where $N \in \mathbb{N}$. Let $q \in \{1, 2, \ldots, M\}$. Then

$$x[k+M+q] = x[k+M] + \frac{c_j - \vec{a}_j^\top \vec{x}[M]}{\vec{a}_j^\top \vec{v}_j[k]} \vec{v}_j[k] - \delta \sum_{i \in \mathcal{A} \subseteq \{1,2,\ldots,n\}} \vec{e}_i.$$

Thus, after the state enters the hyperplane $\mathcal{P}_j$ at time $[k+M]$, its trajectories are confined to the union of of hyperplanes

$$\mathcal{C}_j \{\vec{x} \in \mathcal{X} : \vec{a}_j^\top \vec{x} = c_j - \delta \vec{a}_j^\top \sum_{i \in \mathcal{A}} \vec{e}_i, \mathcal{A} \subseteq \{1, 2, \ldots, n\}\}$$

that are traced by the vertices of the δ-hypercube that slides along the $\vec{a}_j^\top \vec{x} = c_j$ constraint boundary. As $\mathcal{C}_j$ is a reduced-order surface on which trajectories are confined, it is a sliding mode of the system [127].

Optimal Fixed Hypercorner: Assume that $\vec{v}_j[k+M] \propto \vec{1}$ and $x[k+M] \in \mathcal{P}_j$. Then

$$\vec{x}[k+M+M] = \vec{x}[k+M] + \frac{c_j - \vec{x}[k+M]}{\vec{a}_j^\top \vec{v}_j[k]} \vec{v}_j[k] - \delta\vec{1}$$

$$
\begin{aligned}
&= \vec{x}[k+M] + \frac{\delta \vec{a}_j^\top \vec{1}}{\vec{a}_j^\top \vec{v}_j[k]} \vec{v}_j[k] - \delta \vec{1} \\
&= \vec{x}[k+M] + \delta \vec{1} - \delta \vec{1} \\
&= \vec{x}[k+M]
\end{aligned}
$$

Thus, any point $\vec{x}[k] \in \mathcal{P}_j$ such that $\vec{v}_j[k] \propto \vec{1}$ is a fixed point of the algorithm when only considering each M^{th} step after reaching $\mathcal{P}_j$. If $\vec{x}[k]$ is anchored to a particular point in $\mathcal{P}_j$, then the corresponding points of $\mathcal{C}_j$ are anchored as well. Thus, the system rests at an oscillatory state within the vertices of a single hypercube in $\mathcal{C}_j$. However, if $\vec{v}_j[k] \propto \vec{1}$, then $\vec{x}[k] \propto \nabla F(\vec{x}[k])$. Hence, the fixed point $\vec{x}[k]$ is an optimal point for the equivalent problem with c_j replaced with $c_j - \delta \vec{a}_j^\top \vec{1}$. Denote this fixed point by $\vec{x}^+$ and let λ_j^+ such that $\vec{x}^+ = \lambda_j^+ \nabla F(\vec{x}^+)$.

Ensuring Descent

As before, let $\vec{x}[k+M] \in \mathcal{P}_j$ is in its sliding mode.

- Let $i \in \{1, 2, \ldots, n\}$ and assume that $f_i'(\vec{x}[k+M]) > f_i'(\vec{x}^+) = \lambda_j^+ a_{ji}$. By the monotonicity of the gradient, $x_i[k+M] > x_i^+$, which implies that $\vec{a}_j^\top \vec{x}[k+M] > \vec{a}_j^\top x_i^+$. However, this conclusion is a contradiction because $\vec{x}[k+M] \in \mathcal{P}_j$.

- Let $i \in \{1, 2, \ldots, n\}$ and assume that $f_i'(\vec{x}[k+M]) < f_i'(\vec{x}^+) = \lambda_j^+ a_{ji}$. By the monotonicity of the gradient, $x_i[k+M] < x_i^+$, which implies that $\vec{a}_j^\top \vec{x}[k+M] < \vec{a}_j^\top x_i^+$. However, this conclusion is a contradiction because $\vec{x}[k+M] \in \mathcal{P}_j$.

Therefore, there exist $i, \ell \in \{1, 2, \ldots, n\}$ such that

$$
v_{ji}[k+M] = \frac{a_{ji}}{f_i'(\vec{x}[k+M])} > \frac{1}{\lambda_j^+} > \frac{a_{j\ell}}{f_\ell'(\vec{x}[k+M])} = v_{j\ell}[k+M]. \tag{5.27}
$$

Let $\vec{\pi}_j[k] \triangleq (\vec{a}^\top \vec{1})/(\vec{a}^\top \vec{v}_j[k]) \vec{v}_j[k]$. Then

$$
\vec{x}[k+M] = \vec{x}[k] + \delta \left(\vec{\pi}_j[k] - \vec{1} \right). \tag{5.28}
$$

Additionally,

$$\vec{a}_j^\top \vec{\pi}_j[k] = \frac{\vec{a}^\top \vec{1}}{\vec{a}^\top \vec{v}_j[k]} \vec{v}_j[k] = \vec{a}_j^\top \vec{1}. \tag{5.29}$$

However, by Equation (5.27), there must exist $i, \ell \in \{1, 2, \ldots, n\}$ such that $\pi_{ji}[k] \neq \pi_{j\ell}[k]$. Hence, by Equation (5.29), if $i = \arg\max_o v_{jo}[k]$ and $\ell = \arg\min_o v_{jo}[k]$, then $\pi_{ji}[k] > 1$ and $\pi_{j\ell}[k] < 1$. Thus, by Equation (5.28),

$$\max\{x_i[k+2M] - x_i^+ : i \in \{1, 2, \ldots, n\}\} < \max\{x_i[k+M] - x_i^+ : i \in \{1, 2, \ldots, n\}\}.$$

Moreover, because $v_{ji}(\vec{x}) = a_{ji}/f_i'(\vec{x})$ is continuous for all $i \in \{1, 2, \ldots, n\}$, there exists some δ_0 such that $v_{ji}(\vec{x}[k+M]) > v_{j\ell}(\vec{x}[k+M]) \implies v_{ji}(\vec{x}[k+2M]) > v_{j\ell}(\vec{x}[k+2M])$ for all δ with $0 < \delta < \delta_0$. Therefore, the MultiIFD converges asymptotically to the fixed hypercube.

Ultimate Bounds

We have shown that gradient continuity and monotonicity allow the MultiIFD to drive vertices of a hypercube to a fixed position anchored at a point $\vec{x}^+$ that is optimal subject to the constraint that $\vec{a}_j^\top \vec{x} \geq c_j - \delta \vec{a}_j^T \vec{1}$. However, additional information about the convexity of the cost function is needed to represent the distance between $\vec{x}^+$ and the desired optimal point $\vec{x}^*$. For example, the gradient of the cost function $F(\vec{x}) = \|\vec{x}\|_2^2$ at both $\vec{x}^*$ and $\vec{x}^+$ is proportional to both $\vec{x}^*$, $\vec{x}^+$, and $\vec{a}_j$. Thus, the point $\vec{x}^+$ that anchors one corner of the cube is normal to the c_j constraint hyperplane at $\vec{x}^*$ while the adjacent corner $\vec{x}^+ + 1\vec{\delta}$ falls on the hyperplane. Consequently, in experimental simulation studies, after a critical threshold, each component of Euclidean minimizing trajectories was always under the optimal solution component by no more than 2δ.

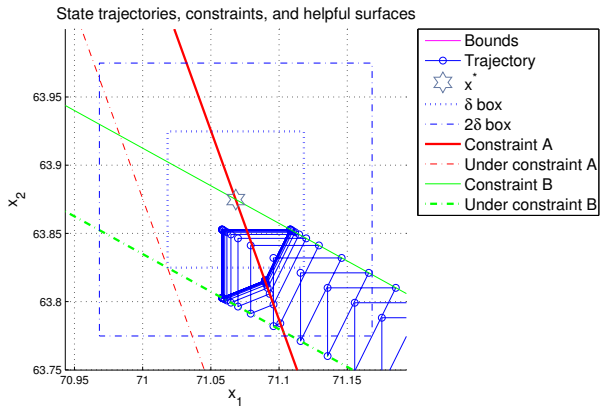


Figure 5.4: The phase-plane trajectories of two lights in a two-sensor simulation. The trajectory enters from the right where only constraint B is active. When the trajectory enters the neighborhood of constraint A, it also becomes active and ceases the motion of B so that the equilibrium solution falls with a 2δ-sized box of the optimal solution, which is shown in the very center.

5.3 Results

Characteristics of the distributed lighting algorithm were investigated in simulation as well as on a tabletop intelligent lighting testbed. A few of the results from those investigations are included here.

5.3.1 Simulation Results

In the figure in Figure 5.4, a two-light–two-scenario scenario is shown. A more general simulation with eight sensors and eight lights is shown in Figure 5.5. The constraint trajectories approach their constraint surfaces and then remain on them. While on the constraint surfaces, the light trajectories move to minimize cost.

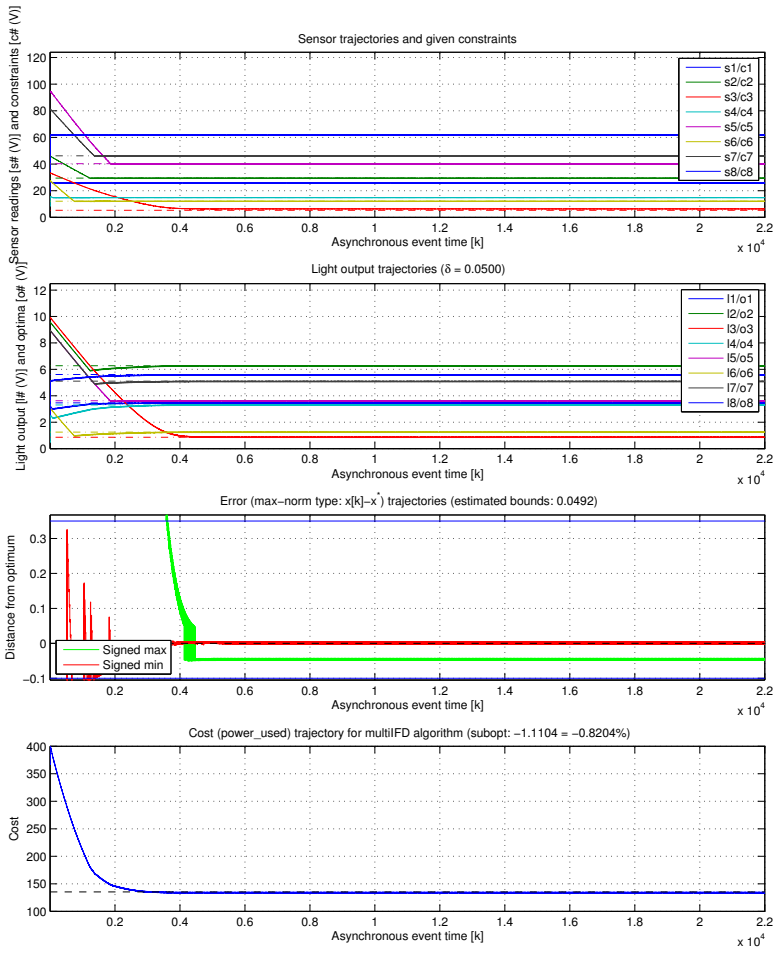
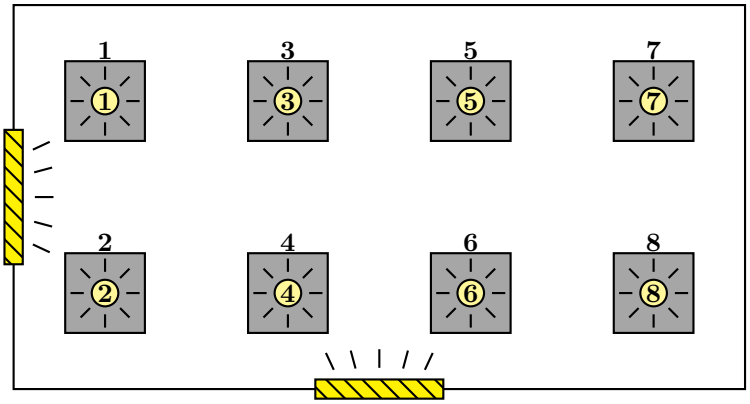


Figure 5.5: Statistics from a lighting simulation with eight lights and eight sensors. The optimal solution meets every constraint, but not all constraints are active. The top of the graph shows the eight sensor trajectories as they approach their minimum constraint levels. The second plot from the top shows the light trajectories as they approach their optimal values. The third plot from the top shows the signed maximum and signed minimum error. The maximum error converges to -0.0492, which is within a δ bound of the optimal solution. The bottom plot shows the cost trajectory; it continually decreases until reaching its minimum within less than 1%.

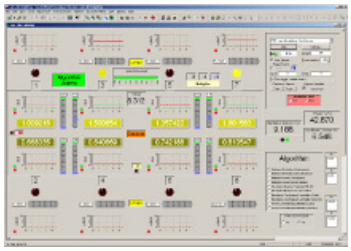
5.3.2 Experimental Results

Experimental validation of the MultiIFD algorithm for distributed lighting was performed using the hardware-in-the-loop apparatus in Figure 5.6. Each algorithm was tested on a single dSPACE RTI1104 DSP (i.e., distributed controls were simulated on a single embedded controller). At the beginning of each run for sixteen seconds (i.e., 2 seconds per light), lights and sensors automatically commissioned themselves using a two-point linearization method. Otherwise, no information was provided to the lights or sensors about relative location of other agents.

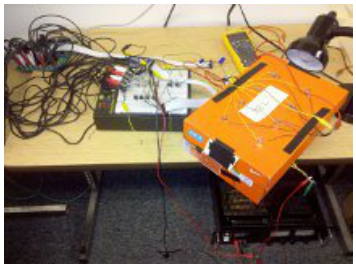
Results for a sample distributed power-minimization experiment with $n = 8$ and $m = 2$ are shown in Figure 5.7. When the commissioning process ends, the controller generates a high overshoot because of the combined action of both sensors that start grossly under constraint. As discussed in Section 5.2.3, these effects can be mitigated by causing the sensors to be less aggressive when under constraint and the lights more aggressive after long lapses in sensor communication. So long as the behavior matches the ideal behavior in the neighborhood of the constraint, the sliding mode optimization continues. Results from a version of the algorithm with these overshoot mechanisms installed is shown in Figure 5.8. For comparison, the results of a centralized dual space optimization procedure is shown in Figure 5.9. To reduce the chattering in the dual space implementation, Lagrage multiplier estimates are used between values produced from the minimization problem. The equilibrium results in both of the distributed cases match the centralized case.



(a) Depiction



(b) User interface

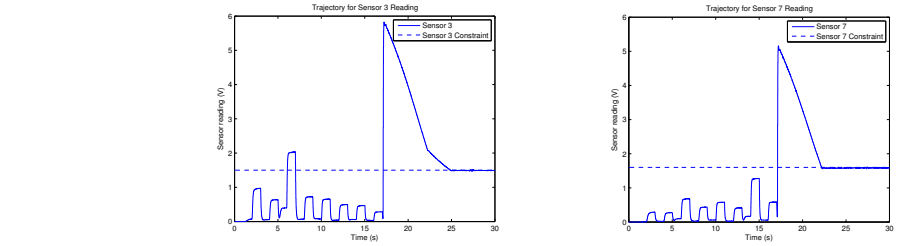


(c) Testbed electronics

(d) Testbed lights

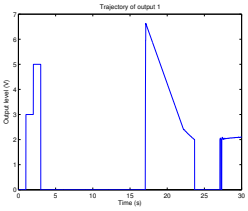
(e) Testbed sensors

Figure 5.6: Experimental lighting testbed with $n = 8$ lights and $m = 8$ sensors. (a) Lights, which are shown as circles around their identities, are mounted in the ceiling of the room and sensors. Sensors, which are shown as patches underneath their identities, are located at a distance beneath the lights. There are two disturbance sources (e.g., windows) shown as hatched rectangles. (b) The graphical user interface designed in dSPACE ControlDesk, which shows an animation of real-time data. (c) The control electronics being actuated by a dSPACE RTI1104 DSP programmed by MATLAB. (d) The incandescent lights inside the testbed. (e) The cadmium sulfide (CdS) photoresistor sensors inside the testbed.

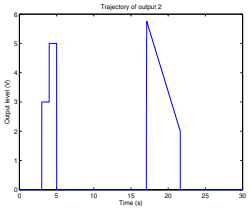


(a) Sensor 3

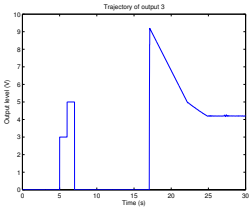
(b) Sensor 7



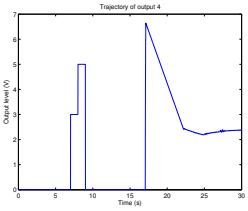
(c) Output 1



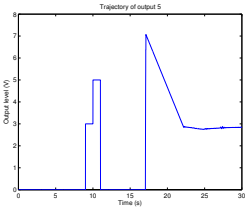
(d) Output 2



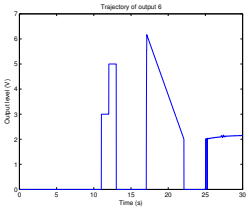
(e) Output 3



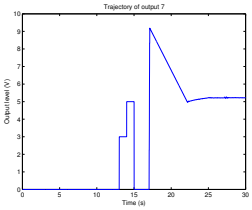
(f) Output 4



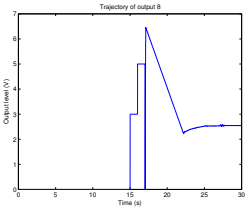
(g) Output 5



(h) Output 6

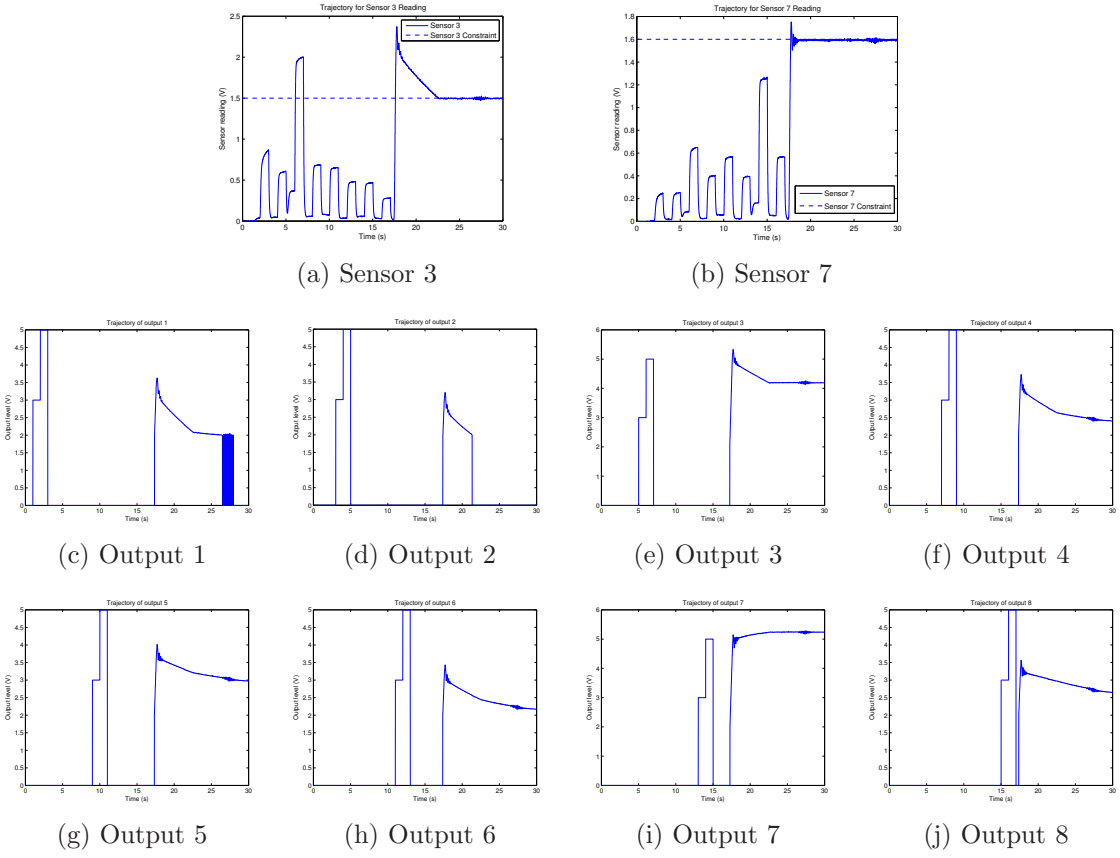


(i) Output 7



(j) Output 8

Figure 5.7: Experimental results for distributed power minimization. In the first sixteen seconds of the experiment, a two-point linearization procedure tests the influence of each light on each sensor. Immediately after the procedure completes, the power minimization algorithm starts. Large transients are observable at the beginning of the experiment followed by smooth tracking of the illumination constraint. (a) Sensor reading and constraint for sensor three. (b) Sensor reading and constraint for sensor seven. (c) Output level for light 1 (d) Output level for light 2 (e) Output level for light 3 (f) Output level for light 4 (g) Output level for light 5 (h) Output level for light 6 (i) Output level for light 7 (j) Output level for light 8

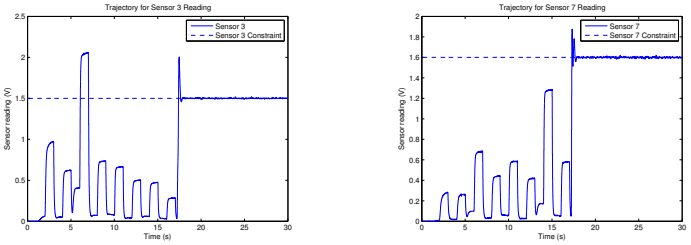


(a) Sensor 3 (b) Sensor 7

(c) Output 1 (d) Output 2 (e) Output 3 (f) Output 4

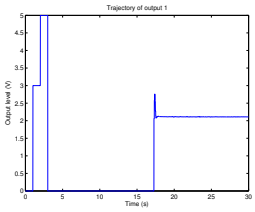
(g) Output 5 (h) Output 6 (i) Output 7 (j) Output 8

Figure 5.8: Experimental results for distributed power minimization with overshoot mitigation. In the first sixteen seconds of the experiment, a two-point linearization procedure tests the influence of each light on each sensor. Immediately after the procedure completes, the power minimization algorithm starts. To prevent large transients, sensors are less aggressive when far under constraint and lights are more aggressive. (a) Sensor reading and constraint for sensor three. (b) Sensor reading and constraint for sensor seven. (c) Output level for light 1 (d) Output level for light 2 (e) Output level for light 3 (f) Output level for light 4 (g) Output level for light 5 (h) Output level for light 6 (i) Output level for light 7 (j) Output level for light 8

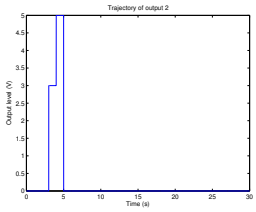


(a) Sensor 3

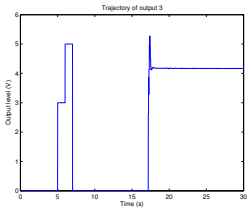
(b) Sensor 7



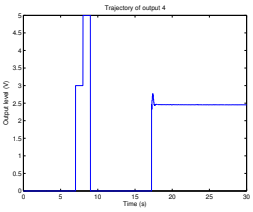
(c) Output 1



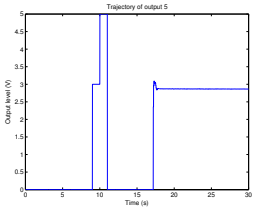
(d) Output 2



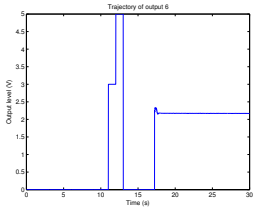
(e) Output 3



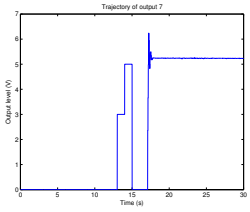
(f) Output 4



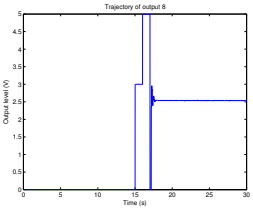
(g) Output 5



(h) Output 6



(i) Output 7



(j) Output 8

Figure 5.9: Experimental results for centralized power minimization with chattering mitigation. In the first sixteen seconds of the experiment, a two-point linearization procedure tests the influence of each light on each sensor. Immediately after the procedure completes, the centralized dual-space power minimization algorithm starts. (a) Sensor reading and constraint for sensor three. (b) Sensor reading and constraint for sensor seven. (c) Output level for light 1 (d) Output level for light 2 (e) Output level for light 3 (f) Output level for light 4 (g) Output level for light 5 (h) Output level for light 6 (i) Output level for light 7 (j) Output level for light 8

Part III: Summary and Conclusions

The thread linking the collected work in this dissertation together is that biomimicry should be expansive and not simply translational. Generalized models that have specializations for both biological and engineering application accelerate research in and collaboration between both areas. The hypothetical *generalized mollusk* [6, 102] is an illustrative example. Mollusks vary greatly in physiological structure and behavior, and yet any two species from the mollusk phylum have a tremendous amount in common. Consequently, mollusk research is necessarily varied but requires active dissemination of information to other researchers in order to prevent duplication. Facing this challenge, invertebrate researchers have created a fictitious generalized mollusk that serves as substrate for understanding the similarities and differences among members of the phylum. Despite the fact that the generalized mollusk is no more real than a unicorn, it is one of the first animals studied in depth in undergraduate invertebrate zoology courses and textbooks. The idea behind the work in this dissertation is that biologists, anthropologists, sociologists, and engineers can benefit from having their own generalized mollusk, a generalized optimal task-processing agent. Here, in Chapter 6, we summarize contributions presented in this dissertation. We outline future research directions in Chapter 7.

Chapter 6: Contributions

6.1 Generalized Solitary Optimal Task-Processing Agents

Existing work that applies solitary optimal foraging theory to artificial systems [7–9, 32, 81, 97] has been successful insofar as it has filled a void. That is, both solitary foragers [112] and autonomous vehicles [7, 9, 81, 97] encounter tasks to process (e.g., food items) according to some stochastic process. In environments where there is a high rate of encounter with high-value tasks, fewer low-value tasks should be chosen for processing because the high opportunity cost. However, when fewer high-value tasks are available, all encountered tasks should be processed. Thus, the criteria for accepting encountered tasks needs to be modulated in some intelligent way based on environmental parameters. Foraging theory for engineering provides logical justifications for choices made when designing these criteria. However, those justifications are based on assumptions grounded in natural systems. In particular, the optimal forager makes decisions that maximize the stochastic limit of its energy intake rate, an objective function that is likely a proximate of Darwinian fitness for a forager with a long lifetime [23, 112]. However, an artificial task-processing agent will often have a finite lifetime either due to fuel or other resource constraints. For example, an autonomous air vehicle may have a relatively small number of packages to deliver to

targets in a mission that is time constrained by fuel load. Hence, autonomous agents may behave similar to foragers (i.e., processing and ignoring randomly encountered tasks), but their decisions should be shaped by optimization of application specific utility functions and not necessarily by rate maximization.

In the work [84] reproduced in Chapter 1, we have identified the salient theoretical features of optimal foraging theory and created a generalized solitary task-processing agent. Like a natural forager, the agent must choose which tasks (e.g., food items) to process and how long to process each task (e.g., when to leave a patch of food items); however, the objective measure of optimality need not be driven by natural processes. So rather than maximizing the rate of energy gain, the generalized solitary task-processing agent presented in this dissertation optimizes an abstract advantage-to-disadvantage function that is a special composition of other parameterized functions of the environment. The class of advantage-to-disadvantage functions includes the long-term rate of gain used by behavioral ecologists, but it also includes other rates of gain, efficiency measures, and objectives that incorporate a success threshold below which a mission is considered a failure. We also develop three linear-time algorithms for finding optimal behaviors for advantage-to-disadvantage functions that meet the mathematical assumptions of each algorithm. The behaviors produced by these algorithms have the same structure as the optimal behaviors familiar to behavioral ecologists. In particular,

- Each of them ranks task types by a generalized profitability and chooses a critical profitability that separates the task types into always-accept and always-ignore groups.

- Each of them defines a generalized marginal value function that declines during task processing until reaching a critical threshold that indicates the task-processing agent should cease processing the task.

However, the critical switching thresholds vary with the choice of objective function. So an autonomous agent already designed to have foraging-like preferences can be easily modified to behave optimally with respect to different objective functions. For demonstration, we simulated:

(i) an autonomous vehicle using a classical optimal foraging strategy

(ii) an autonomous vehicle using an objective function that includes a critical gain threshold

(iii) an autonomous vehicle that indiscriminately processes all encountered tasks

For an infinite-time horizon, the vehicle in (i) that uses the classical objective maximizes gain under the assumption that future opportunities are certain. That is, it forgoes some additional gain in the present in order to maximize its total number of encounters. However, in the simulations we perform, each autonomous vehicle mission is ended after a given finite number of tasks are completed (e.g., after all packages have been delivered to targets). In this scenario, the vehicle in (ii) that incorporates the success threshold outperforms both the rate-maximizing behavior in (i) and the indiscriminate behavior in (iii). Thus, our generalized solitary task-processing framework allows autonomous vehicle designers to customize the intuitive structures of optimal foraging theory for their own applications.

6.2 Ecological Rationality

Biologists and anthropologists have limited ability to scientifically explain observed irrational behavior because of the possibility that all exogenous effects have not been controlled for [1, 2, 10, 11, 15, 20, 27, 41, 46, 56, 62, 68, 69, 73, 74, 98, 104, 107–110, 118]. Suboptimal foraging behavior may be due to unmodeled sexual or predation-risk-mitigation benefits or developmental limitations or myriad other factors outside of the foraging model. However, engineered agents have relatively tight controls on variation and can be designed to follow behavioral rules to a fault. When even these automata exhibit the same ostensibly irrational decision making as observed in animals, there is reason to believe that the behavior may actually be rational. Hence, the engineering design process is itself a scientific exploration of behavior and presents an opportunity for engineers to contribute to natural science. In this dissertation, we review work [82, 83] that shows how two such irrational behaviors are consistent with predictions from optimality models when model assumptions are violated or when the parameter space is sufficiently large. Both of these observations followed from the study of generalized task-processing agents in theory and simulation, and thus a framework for understanding optimal task-processing agents has value in both design and behavioral analysis.

6.2.1 Computationally Simple Implementations and Impulsiveness

In operant binary-choice experiments in the laboratory, animal subjects are repeatedly given a choice between two food options in order to identify the animal's preferences. Prior to the experiment, the subjects are trained usually through starvation-and-reward conditioning so that they are familiar with the operation of the experimental apparatus (e.g., they recognize that pressing certain buttons is associated with being given certain food items). In these experiments, when given a mutually exclusive choice between two food items [e.g., 1, 15, 20, 41, 62, 98, 104, 107, 110], animals will often prefer the items with the shortest processing time regardless of its foraging gain. However, a rate-maximizing behavior would give preference to items with the highest gain-to-time ratio regardless of the actual handling time. Moreover, as reviewed by Giraldeau and Caraco [38, pp. 155–167], in experiments that do not force animals to make binary-choice decisions, animals flexibly maximize their rate of gain by dynamically adjusting their behavior in response to changes in the environment so that they maintain maximal long-term rate of gain. Furthermore, Stephens and Anderson [110] and Stephens et al. [114] show that animals that are impulsive in binary-choice schedules return to rate-maximizing preferences in sequential-choice schedules (i.e., experiments where animals can ignore an item now to ensure that an item of higher gain is encountered later).

In the work [82] reproduced in Chapter 2, we present a computationally simple decision-making heuristic that converges to the maximal long-term rate of gain under Poisson encounter assumptions (i.e., simultaneous encounters occurring with zero probability). This heuristic can be used to implement optimal behaviors from the

generalized task-processing agent framework. However, we show that if the decision rule is implemented on an agent facing repeated mutually exclusive binary-choice encounters, then a second asymptotic equilibrium is generated, and so the long-term performance is a function of initial condition. If it is initialized with a high accumulated gain, it behaves optimally as a rate-maximizing specialist. However, if it is initialized with low gain, it converges upon a suboptimal generalist equilibrium. In particular, the decision rule initially over generalizes and thus prevents its gain from ever reaching a point where specialization is warranted. We suggest that if a similar decision rule is used in animals, then the operant procedure of starving the animals before the experiment may predispose them to suboptimal behavior. Moreover, animals that have evolved in environments where prey is stationary (e.g., herbivores) will have not been tuned by natural selection for mutually exclusive prey choice.

The decision-making heuristic we present in Chapter 2 need not only apply to rate maximization. As we discuss, models of foraging under digestive-rate constraints [47] result in foraging preferences notably different from the unconstrained preferences. In particular, animals that have a material-processing-rate limit still have preferences ordered by a form of profitability, but their partitioning task type is associated with a partial preference. For example, a shorebird that forages for mollusks with a high bulk-to-food ratio will process all high-calorie mollusks and ignore all low-calorie mollusks, but a subunity fraction of intermediate mollusks will be processed [118, 119]. These partial preferences are only predicted by rate-maximization theory when additional bulk constraints are added. We show that our decision-making heuristic also predicts partial preferences for an intermediate task type when its profitability is changed to include a temporal effect of bulk processing. Additionally, this approach

is consistent with new optimality models of digestive-rate-constrained foragers [125]. Hence, the work reproduced in this dissertation not only provides simple implementation tools for task-processing agent design, but it suggests reasons for observed irrational decision-making in animals.

6.2.2 Over-processing of Tasks

As reviewed by Nonacs [69], the predictions about optimal task-processing length from classical foraging theory tend to be short compared to observations of the behavior of certain classes of animals. Thus, Nonacs argues that optimal foraging theory is incomplete. Moreover, anthropologists and economists [10, 11, 56, 108] have shown that human tendency to continue a task is positively correlated with that task's cost, which is an ostensibly irrational sunk-cost effect. Recently, animals [68] have been shown to exhibit the same cost-sensitive effect. In the work [83] reproduced in Chapter 3, we show how augmenting classical optimal foraging theory to better model foraging costs not only predicts the extension of task-processing length, but it shows how increased task-processing length will be correlated with increased environmental costs.

In classical optimal foraging theory [112], the gain returned from a patch of food is assumed to be non-negative and decelerating. Thus, the per-patch instantaneous rate of gain is an initially positive decreasing function of marginal returns. When the marginal returns reach the long-term average rate of gain of the environment, the forager leaves the patch. If the time between encounters increases or the maximum gain of each encounter decreases, the long-term average rate of gain for the environment

will also decrease, and thus the animal will spend more time in each patch. Consequently, as we show in Chapter 3, if patch gain is allowed to be initially negative (e.g., due to recognition costs or initial energetic expenditure needed to enter the patch), the maximum patch gain will be decreased, and so longer processing times are expected. This prediction fits the study of tundra swans by Nolet et al. [68] particularly well. Each swan forages on tubers buried in soil that varying depths throughout a lake. Each tuber has the same nutritional value regardless of its position, but tubers buried in deep water require a swan to completely tip its body in order to reach it as opposed to only lowering its neck. Nolet et al. observe that when swans move into deep areas to forage, the time spent in the energetically unfavorable up-ended position is actually increased. The rate-maximizing explanation for this behavior is that the swans are spending more time achieving positive foraging gains underwater to reduce the encounter rate with each costly tuber patch. Thus, by expanding the parameter space traditionally used in optimal foraging theory, predictions from our generalized task-processing agent framework better match observations in nature.

6.3 Nash Optimal Cooperative Task-Processing

In the work [85] reproduced in Chapter 4, optimal task-processing on a network is considered. Design and analysis methods exist for networks of autonomous task-processing agents [26, 86], but these methods focus on maintaining bounded queue lengths in the system and not on adjusting task flow to optimally allocate incoming tasks to available agents. Grid computing [22, 30] proposes allocation methods for maintaining optimal resource allocation in a network of autonomous agents, but the approach focuses on the design of protocols for a network of third-party

agents. Optimal message passing algorithms exist for *ad hoc* multi-hop communication networks [3, 4, 21], but these algorithms are inappropriate for application to task-processing networks where tasks cannot be duplicated nor dropped. The optimal group task-processing algorithms that do exist [31, 32, 37] require frequent communication and coordination among agents, which may be prohibitive for some scenarios. Thus, the work discussed in the dissertation focuses on the design of decentralized distributed task-processing agents that achieve desirable system characteristics without much explicit coordination between agents.

In the approach presented in this dissertation, each task-processing agent is responsible for a set of locally encountered tasks. The agent accumulates some value for each locally encountered task that is processed, but it can forgo paying the cost of processing the task (e.g., a cost proportional to fuel use) if another agent agrees to process it instead. Likewise, the agent can process tasks encountered at remote agents, but it must also pay the processing cost of that task. In order for agents to make their decisions totally asynchronously, the constraints on their decision variables (i.e., whether to volunteer to process remotely encountered tasks) must be separable, and thus a Nash equilibrium approach is used. The Nash equilibrium behavior of the natural system is to never volunteer for remotely generated tasks, and so a fictitious trading economy is introduced. In this economy, agents that volunteer for remote tasks are paid a price that decreases with the number of volunteers for the remote tasks. Consequently, at low volunteering levels, there is incentive for each agent to volunteer more, and at high volunteering levels, there is incentive for each agent to volunteer less. In the work from Chapter 4, constraints on the task-processing network topology and on the fictitious trading economy are given that guarantee totally

asynchronous convergence to the Nash equilibrium of the system. Moreover, it is shown that the competitive equilibrium has features that are favorable to the performance of the group as a whole. In an autonomous air vehicle example given, a vehicle facing a relatively high rate of encounters with tasks volunteers less to process the tasks of its neighbors, and its neighbors volunteer to process its tasks more. Thus, despite being driven by local utility functions, the decentralized group as a whole has features similar to groups of load balancing agents. Moreover, the conditions that guarantee convergence to these competitive equilibria with cooperative features may help to explain similar structures in natural cooperative groups. In fact, similar conditions exist in cooperative birth–death networks used to study emergent altruism [59, 70–72].

6.4 Pareto Optimal Constrained Distributed Resource Allocation

As discussed by Bertsekas and Tsitsiklis [19], distributed numerical optimization is best suited for problems with separable configuration spaces. Otherwise, agents must coordinate actions so that changes in variables on one agent properly constrain the changes in variables on a different agent at each numerical iteration. Consequently, optimization problems with non-separable constraints are usually re-cast into a dual space where each optimization variable is associated with a particular constraint. For example, an optimization problem that operates over an n-dimensional space with m constraints has a dual optimization problem that is unconstrained over an m-dimensional configuration space. Unfortunately, the separability of the primal cost function does not imply the separability of the dual cost function. So parallel distributed optimization may still be infeasible in the dual space.

To mitigate the complications introduced by parallel distributed optimization, the work in Chapter 4 uses distributed numerical optimization to solve for a Nash equilibrium. That is, each of n agents has independent constraints and iterates toward locally optimal solutions. Ideally, a distributed numerical optimization algorithm would solve for Pareto equilibria. That is, each agent iterates toward solutions that increase not only its local utility function but also the utility function of other agents on the network. As shown by Verkama et al. [120], distributed Pareto optimization is possible even when each of n agents is not equipped with knowledge of the utility functions on other agents. However, the agents must communicate proposals and responses to coordinate their motion to guarantee mutual benefits. Moreover, the configuration space is assumed to have separable constraints.

So in Chapter 5, a distributed numerical method for constrained Pareto optimization is introduced. The approach iterates over the primal space, but it also includes aspects of dual-space methods. In particular, it is assumed that n independent agents maintain uniform motion within the n-dimensional primal space. It then assumes that each of m additional agents corresponds to one of the m constraints. Whenever the uniform motion of the n agents causes one of the m constraints to be violated, the corresponding constraint agent restores the system back to the constraint; however, its restoration direction is slanted away from the gradient. Consequently, the repeated violation and reassertion of each constraint eventually leads the system to come to rest near its Pareto optimal solution. None of the n agents coordinate with each other, and each of the m agents has no knowledge of the existence of any of the other m agents. Thus, coordination between constraint agents is stigmergic. It is shown that these methods are particularly amenable to intelligent lighting [e.g., 40, 63, 77, 124]

where distributed lights and sensors must maintain appropriate lighting levels while minimizing power usage. However, the constrained optimization problem is a generalization of social foraging distributions from behavioral ecology [34, 115], economic dispatch problems in power systems engineering [17], and distributed resource allocation problems in autonomous vehicles [31, 66, 95, 96]. Thus, Chapter 5 suggests distributed algorithms that can be used in a wide range of constrained optimization applications.

Chapter 7: Future Directions

7.1 Generalized Task-Processing Agents

The generalized solitary task-processing agent described in Chapter 1 provides an analysis framework for optimal task-type and task-processing-length choice. However, the forager described by Gendron and Staddon [35] also must choose its search speed in an environment mixed with some conspicuous and some cryptic prey. Because the forager has limited detection capabilities at high speed, the optimal speed choice may be less than the forager's maximum speed. If the most cryptic prey are also the ones with lowest profitability, the forager should ignore these prey and travel at maximum speed, but the choice of prey types to ignore is determined partly by the search speed as it impacts the encounter rate of all types. So choosing the two problems of choosing optimal task-type preference and optimal search speed are coupled. As discussed by Pavlic and Passino [81], this problem applies to autonomous air vehicles as well. The advantage-to-disadvantage optimization framework described in this dissertation assumes fixed encounter rates and thus fixed speeds. Hence, an important future direction is to generalize the results of Gendron and Staddon and Pavlic and Passino to include a parameter analogous to search speed that modulates encounter rates.

Decision-Making Heuristics: The optimal decision-making heuristic in Chapter 2 is shown in simulation to converge to the optimal long-term rate of gain so long as it faces encounters according to a Poisson process. Otherwise, disjoint suboptimal equilibria are generated that can attract the fixation of the otherwise optimal task-processing agent. The trajectories generated by the decision-making heuristic come from a stochastic process formed by events which depend upon the outcome of prior events. That is, because the decision-making heuristic is sensitive to the present state of the task-processing agent and also affects the future state, events in each realization of the process are coupled. Consequently, the generated stochastic process has low analytical tractability. If the heuristic is to be used in real engineering implementations, its convergence characteristics need to be better understood. So an important future direction is to apply rigorous stochastic convergence analysis methods to the decision-making heuristic discussed in Chapter 2.

7.2 Cooperative Task-Processing Agents

The basic cooperative task-processing networks described in Chapter 4 can be expanded in several useful directions. First, each agent adjusts a single volunteering preference that applies to all agents that submit task-processing requests to it. Consequently, the convergence analysis is simplified because each decision variable is a scalar. However, richer behaviors are possible if agents can adjust per-agent volunteering preferences. For example, an agent may cooperate differently with an agent that frequently submits task-processing requests than it will with an agent that rarely submits requests. An additional weakness of the present cooperative task-processing network formulation is that it assumes agents either have infinite processing capacity

or that tasks take negligible processing time. These issues are the central motivations of the queueing and flexible manufacturing system of Cruz [26] and Perkins and Kumar [86]. Likewise, it is an important future direction to incorporate the processing time of each task into the problem formulation. Following the example from optimal foraging theory, it may be possible to associate a processing time with each encountered task type and model the stochastic limit of the long-term rate of gain on each agent. If this utility function has a tractable structure, it may be possible to show distributed Nash equilibrium computation results.

7.3 Distributed Optimization with Constraints

The distributed optimization algorithm described in Chapter 5 has attractive results in simulation and in experimental studies on test systems (e.g., an intelligent lighting testbed). However, more analytical results are needed to understand its robustness characteristics. The present theoretical analysis implies that ultimate error bounds on the fixed set of the algorithm depend upon the particular cost function being minimized, and the geometric characteristics of the fixed set when the Euclidean norm is minimized suggests a tight error bound; however, exactly how the gradient shape, simultaneity of events, and order of events affect the algorithm convergence is not well understood. Moreover, although the present framework allows for simultaneous events, it does not allow the order of those events nor the events which are simultaneous to change. An assumption of partial asynchrony as described by Bertsekas and Tsitsiklis [19] would be a beneficial adjustment to the present theory. Additionally, the case for multiple active constraints needs to be investigated with the same rigor as the case of a single active constraint. Once these issues are handled, the

formulation may be adjusted to support non-linear constraint surfaces. The present results suggest that the algorithm should also support convex constraint surfaces, but this conjecture needs to be verified analytically. Additionally, if non-linear constraint surfaces are used, more complex automatic commissioning procedures need to be investigated. Finally, although intelligent lighting is a natural application for this distributed algorithm, other applications (e.g., optimal economic dispatch in power systems) are possible and should be explored.

Bibliography

[1] George W. Ainslie. Impulse control in pigeons. *Journal of the Experimental Analysis of Behavior*, 21(3):485–489, May 1974.

[2] George W. Ainslie. Specious reward: a behavioral theory of impulsiveness and impulse control. *Psychological Bulletin*, 82(4):463–496, July 1975.

[3] Eitan Altman, Arzad A. Kherani, Pietro Michiardi, and Refik Molva. Non-cooperative forwarding in ad-hoc networks. In *Proceedings of Networking*, volume 3462 of *Lecture Notes in Computer Science*, pages 486–498, 2005.

[4] Eitan Altman, Anurag Kumar, D. Kumar, and R. Venkatesh. Cooperative and non-cooperative control in IEEE 802.11 WLANs. In *Proceedings of the 19th International Teletraffic Congress*, Beijing, August 29 – September 2, 2005.

[5] Yair Amir, Baruch Awerbuch, Amnon Barak, R. Sean Borgstrom, and Arie Keren. An opportunity cost approach for job assignment in a scalable computing cluster. *IEEE Transactions on Parallel and Distributed Systems*, 11(7):760–768, July 2000. DOI:10.1109/71.877834.

[6] Donald Thomas Anderson. *Invertebrate Zoology*. Oxford University Press, 2001. ISBN 978-0195513684.

[7] Burton W. Andrews, Kevin M. Passino, and Thomas A. Waite. Foraging theory for decision-making system design: task-type choice. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, volume 5, pages 4740–4745, Nasssau, Bahamas, December 14–17, 2004. ISBN 0-7803-8682-5. DOI:10.1109/CDC.2004.1429539.

[8] Burton W. Andrews, Kevin M. Passino, and Thomas A. Waite. Social foraging theory for robust multiagent system design. *IEEE Transactions on Automation Science and Engineering*, 4(1):79–86, January 2007.

[9] Burton W. Andrews, Kevin M. Passino, and Thomas A. Waite. Foraging theory for autonomous vehicle decision-making system design. *Journal of Intelligent and Robotic Systems*, 49(1):39–65, May 2007. DOI:10.1007/s10846-007-9138-9.

[10] Hal Arkes and Catherine Blumer. The psychology of sunk cost. *Organizational Behavior and Human Decision Processes*, 35:124–140, 1985.

[11] Hal R. Arkes and Peter Ayton. The sunk cost and Concorde effects: are humans less rational than lower animals? *Psychological Bulletin*, 125(5):591–600, 1999.

[12] Maiko Ashibe, Mitsunori Miki, and Tomoyuki Hiroyasu. Distributed optimization algorithm for lighting color control using chroma sensors. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 2008. SMC 2008.*, Singapore, 2008. DOI:10.1109/ICSMC.2008.4811270.

[13] Michael Bacharach. *Economics and the Theory of Games*. Macmillan, London, 1976.

[14] Tamer Başar and Geert Jan Olsder. *Dynamic Noncooperative Game Theory*. Number 23 in Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, second edition, 1999. ISBN 0-89871-429-X.

[15] Melissa Bateson and Alex Kacelnik. Rate currencies and the foraging starling: the fallacy of the averages revisited. *Behavioral Ecology*, 7(3):341–352, 1996. DOI:10.1093/beheco/7.3.341.

[16] Melissa Bateson and Siân C. Whitehead. The energetic costs of alternative rate currencies in the foraging starling. *Ecology*, 77(4):1303–1307, June 1996.

[17] Arthur R. Bergen and Vijay Vittal. *Power Systems Analysis*. Prentice Hall, Upper Saddle River, NJ, second edition, 2000. ISBN 0-13-691990-1.

[18] Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, 1995.

[19] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, Massachusetts, 1997.

[20] C. M. Bradshaw and E. Szabadi. Choice between delayed reinforcers in a discrete-trials schedule: the effect of deprivation level. *Quarterly Journal of Experimental Psychology*, 44(1):1–16, 1992.

[21] Levente Buttyán and Jean-Pierre Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *Mobile Networks and Applications*, 8:579–592, 2003.

[22] Rajkumar Buyya. *Economic-based Distributed Resource Management and Scheduling for Grid Computing*. PhD thesis, Monash University, Melbourne, Australia, April 2002.

[23] Eric L. Charnov. *Optimal Foraging: Some Theoretical Explorations*. PhD thesis, University of Washington, 1973.

[24] Eric L. Charnov. Optimal foraging: the marginal value theorem. *Theoretical Population Biology*, 9(2):129–136, April 1976.

[25] Eric L. Charnov. Optimal foraging: attack strategy of a mantid. *American Naturalist*, 110(971):141–151, January–February 1976.

[26] Rene L. Cruz. A calculus for network delay, part II: network analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991. DOI:10.1109/18.61110.

[27] Richard Dawkins and Tamsie R. Carlisle. Parental investment, mate desertion and a fallacy. *Nature*, 262(5564):131–133, July 8, 1976.

[28] Lester 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.

[29] Catherine A. Faver and Elizabeth B. Strand. To leave or to stay? *Journal of Interpersonal Violence*, 18(12):1367–1377, 2003. DOI:10.1177/0886260503258028.

[30] Joan Feigenbaum and Scott Shenker. Distributed algorithmic mechanism design: recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communication*, pages 1–13, Atlanta, Georgia, USA, September 28, 2002. ISBN 1-58113-587-4. DOI:10.1145/570810.570812.

[31] Jorge Finke and Kevin M. Passino. Stable cooperative vehicle distributions for surveillance. *Journal of Dynamic Systems, Measurement, and Control*, 129(5): 597–608, 2007. DOI:10.1115/1.2767656.

[32] Jorge Finke, Kevin M. Passino, and Andrew G. Sparks. Stable task load balancing for cooperative control of networked autonomous air vehicles. *IEEE Transactions on Control Systems Technology*, 14(5):789–803, September 2006. DOI:10.1109/TCST.2006.876902.

[33] John P. Fletcher, John P. Hughes, and Ian F. Harvey. Life expectancy and egg load affect oviposition decisions of a solitary parasitoid. *Proceedings: Biological Sciences*, 258(1352):163–167, November 22, 1994.

[34] Stephen Dewitt Fretwell and Henry L. Lucas, Jr. On territorial behavior and other factors influencing habitat distribution in birds: I. theoretical development. *Acta Biotheoretica*, 19(1):16–36, March 1969. DOI:10.1007/BF01601953.

[35] Robert P. Gendron and John E. R. Staddon. Searching for cryptic prey: the effect of search rate. *American Naturalist*, 121(2):172–186, February 1983.

[36] Alvaro E. Gil and Kevin M. Passino. Stability analysis of network-based cooperative resource allocation strategies. *Automatica*, 42(2):245–250, 2005. DOI:10.1016/j.automatica.2005.09.015.

[37] Alvaro E. Gil, Kevin M. Passino, Sriram Ganapathy, and Andrew Sparks. Cooperative task scheduling for networked uninhabited air vehicles. *IEEE Transactions on Aerospace and Electronic Systems*, 44(2):561–581, April 2008. DOI:10.1109/TAES.2008.4560207.

[38] Luc-Alain Giraldeau and Thomas Caraco. *Social Foraging Theory*. Princeton University Press, Princeton, NJ, 2000.

[39] Luc-Alain Giraldeau and Barbara Livoreil. Game theory and social foraging. In Lee Alan Dugatkin and Hudson Kern Reeve, editors, *Game Theory and Animal Behavior*, pages 16–37. Oxford University Press, New York, 1998.

[40] J. Granderson, Y.-J. Wen, A. M. Agogino, and K. Goebel. Towards demand-responsive intelligent lighting with wireless sensing and actuation. In *Proceedings of the IESNA (Illuminating Engineering Society of North America) 2004 Annual Conference*, pages 265–274, Tampa, FL, 2004.

[41] Leonard Green, E. B. Fisher Jr., Steven Perlow, and Lisa Sherman. Preference reversal and self-control: choice as a function of reward amount and delay. *Behaviour Analysis Letters*, 1:43–51, 1981.

[42] Daniel Grosu and Anthony T. Chronopoulos. Algorithmic mechanism design for load balancing in distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(1):77–84, February 2004. DOI:10.1109/TSMCB.2002.805812.

[43] W. D. Hamilton. The genetical evolution of social behavior. I. *Journal of Theoretical Biology*, 7(1):1–16, 1964.

[44] Lawrence D. Harder and Leslie A. Real. Why are bumble bees risk averse? *Ecology*, 68(4):1104–1108, 1987.

[45] George E. Heimpel and Jay A. Rosenheim. Egg limitation in parasitoids: a review of the evidence and a case study. *Biological Control*, 11(2):160–168, February 1998. DOI:10.1006/bcon.1997.0587.

[46] Samual E. Henly, Allison Ostdiek, Erika Blackwell, Sarah Knutie, Aimee S. Dunlap, and David W. Stephens. The discounting-by-interruptions hypothesis: model and experiment. *Behavioral Ecology*, 19(1):154–162, 2008. DOI:10.1093/beheco/arm110.

[47] Hirofumi Hirakawa. Diet optimization with a nutrient or toxin constraint. *Theoretical Population Biology*, 47(3):331–346, June 1995.

[48] Hirofumi Hirakawa. Digestion-constrained optimal foraging in generalist mammalian herbivores. *Oikos*, 78(1):37–47, February 1997.

[49] Hirofumi Hirakawa. How important is digestive quality? a correction of Verlinden and Wiley's digestive rate model. *Evolutionary Ecology*, 11(2):249–251, March 1997.

[50] Alasdair I. Houston and John M. McNamara. The choice of two prey types that minimizes the probability of starvation. *Behavioral Ecology and Sociobiology*, 17(2):135–141, July 1985. DOI:10.1007/BF00299245.

[51] Alasdair I. Houston and John M. McNamara. *Models of Adaptive Behavior*. Cambridge University Press, Cambridge, 1999.

[52] Ali Ipakchi and Farrokh Albuyeh. Grid of the future. *IEEE Power and Energy Magazine*, 7(2):52–62, March/April 2009. DOI:10.1109/MPE.2008.931384.

[53] Yoh Iwasa, Yoshito Suzuki, and Hiroyuki Matsuda. Theory of oviposition strategy of parasitoids. I. effect of mortality and limited egg number. *Theoretical Population Biology*, 26(2):205–227, October 1984. DOI:10.1016/0040-5809(84)90030-3.

[54] Elizabeth M. Jakob. Individual decisions and group dynamics: why pholcid spiders join and leave groups. *Animal Behaviour*, 68(1):9–20, July 2004. DOI:10.1016/j.anbehav.2003.06.026.

[55] M. V. Johns and R. G. Miller, Jr. Average renewal loss rates. *Annals of Mathematical Statistics*, 34(2):396–401, June 1963.

[56] Chandra Kanodia, Robert Bushman, and John Dickhaut. Escalation errors and the sunk cost effect: an explanation based on reputation and information asymmetries. *Journal of Accounting Research*, 27(1):59–77, Spring 1989.

[57] Danny B. Lange. Mobile objects and mobile agents: the future of distributed computing? In Eric Jul, editor, *ECOOP'98 — Object-Oriented Programming: 12th European Conference, Brussels, Belgium, July 20–24, 1998, Proceedings*, volume 1445 of *Lecture Notes in Computer Science*, pages 1–12, Berlin, July 20–24, 1998. Springer. ISBN 978-3-540-64737-9. DOI:10.1007/BFb0054083.

[58] Danny B. Lange and Mitsuru Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 42(3):88–89, March 1999. DOI:10.1145/295685.298136.

[59] Erez Lieberman, Christoph Hauert, and Martin A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, January 20, 2005. DOI:10.1038/nature03204.

[60] Rajiv T. Maheswaran, Orhan Çağri Imer, and Tamer Başar. Agent mobility under price incentives. In *Proceedings of the 38th IEEE Conference on Decision and Control*, volume 4, pages 4020–4025, Phoenix, Arizona, USA, December 7–10, 1999. DOI:10.1109/CDC.1999.827989.

[61] Andreu Mas-Colell, Michael Dennis Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, New York, 1995.

[62] C. F. McDiarmid and M. E. Rilling. Reinforcement delay and reinforcement rate as determinants of schedule preference. *Psychonomic Science*, 2:195–196, 1965.

[63] Mitsunori Miki, Emi Asayama, and Tomoyuki Hiroyasu. Intelligent lighting system using visible-light communication technology. In *Proceedings of the 2006 IEEE Conference on Cybernetics and Intelligent Systems*, pages 1–6, 2006. DOI:10.1109/ICCIS.2006.252257.

[64] Oscar P. J. M. Minkenberg, Marc Tatar, and Jay A. Rosenheim. Egg load as a major source of variability in insect foraging and oviposition behavior. *Oikos*, 65(1):134–142, October 1992.

[65] John Monterosso and George W. Ainslie. Beyond discounting: possible experimental models of impulse control. *Psychopharmacology*, 146:339–347, 1999.

[66] Brandon J. Moore, Jorge Finke, and Kevin M. Passino. Optimal allocation of heterogeneous resources in cooperative control scenarios. *Automatica*, 45(3): 711–715, March 2009. DOI:10.1016/j.automatica.2008.09.007.

[67] Walter Nicholson. *Microeconomic Theory: Basic Principles and Extensions*. Dryden Press, Fort Worth, TX, fifth edition, 1992.

[68] Bart A. Nolet, Oscar Langevoord, Richard M. Bevan, Kirsten R. Engelaar, Marcel Klaassen, Roef J. W. Mulder, and S. Van Dijk. Spatial variation in tuber depletion by swans explained by differences in net intake rates. *Ecology*, 82(6):1655–1667, June 2001. DOI:10.1890/0012-9658(2001)082[1655:SVITDB]2.0.CO;2.

[69] Peter Nonacs. State dependent behavior and the marginal value theorem. *Behavioral Ecology*, 12(1):71–83, January 2001.

[70] Martin A. Nowak. Five rules for the evolution of cooperation. *Science*, 314 (5805):1560–1563, December 8, 2006. DOI:10.1126/science.1133755.

[71] Martin A. Nowak and Robert M. May. Evolutionary games and spatial chaos. *Nature*, 359(6398):826–829, October 29, 1992. DOI:10.1038/359826a0.

[72] Hisashi Ohtsuki, Christoph Hauert, Erez Lieberman, and Martin A. Nowak. A simple rule for the evolution of cooperation on graphs. *Nature*, 441(7092): 502–505, 2006. DOI:10.1038/nature04605.

[73] Ola Olsson and Joel S. Brown. The foraging benefits of information and the penalty of ignorance. *Oikos*, 112(2):260–273, February 2006. DOI:10.1111/j.0030-1299.2006.13548.x.

[74] Ola Olsson and Noél M. A. Holmgren. The survial-rate-maximizing policy for Bayesian foragers: wait for good news. *Behavioral Ecology*, 9(4):345–353, 1998.

[75] Andy Oram, editor. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly & Associates, Sebastopol, CA, 2001. ISBN 978-0-596-00110-0.

[76] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. MIT Press, Cambridge, MA, 1994. ISBN 0-262-15041-7.

[77] Meng-Shiuan Pan, Lun-Wu Yeh, Yen-Ann Chen, Yu-Hsuan Lin, and Yu-Chee Tseng. A WSN-based intelligent light control system considering user activities and profiles. *IEEE Sensors Journal*, 8(10):1710–1721, October 2008. DOI:10.1109/JSEN.2008.2004294.

[78] Kevin M. Passino. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems Magazine*, 22(3):52–67, 2002.

[79] Kevin M. Passino. *Biomimicry for Optimization, Control, and Automation*. Springer-Verlag, London, 2005.

[80] Theodore P. Pavlic. Optimal foraging theory revisited. Master's thesis, The Ohio State University, Columbus, OH, 2007. URL `http://www.ohiolink.edu/etd/view.cgi?acc_num=osu1181936683`.

[81] Theodore P. Pavlic and Kevin M. Passino. Foraging theory for autonomous vehicle speed choice. *Engineering Applications of Artificial Intelligence*, 22: 482–489, 2009. DOI:10.1016/j.engappai.2008.10.017.

[82] Theodore P. Pavlic and Kevin M. Passino. When rate maximization is impulsive. *Behavioral Ecology and Sociobiology*, 64(8):1255–1265, August 2010. DOI:10.1007/s00265-010-0940-1.

[83] Theodore P. Pavlic and Kevin M. Passino. The sunk-cost effect as an optimal rate-maximizing behavior. *Acta Biotheoretica*, 2010. DOI:10.1007/s10441-010-9107-8. In press.

[84] Theodore P. Pavlic and Kevin M. Passino. Generalizing foraging theory for analysis and design. *International Journal of Robotics Research*, 2010. Submitted.

[85] Theodore P. Pavlic and Kevin M. Passino. Cooperative task processing. *IEEE Transactions on Automatic Control*, 2010. Submitted.

[86] James R. Perkins and P. R. Kumar. Stable, distributed, real-time scheduling of flexible manufacturing/assembly/disassembly systems. *IEEE Transactions on Automatic Control*, 34(2):139–148, February 1989. DOI:10.1109/9.21085.

[87] Peter Pirolli. Rational analyses of information foraging on the web. *Cognitive Science*, 29(3):343–373, 2005.

[88] Peter Pirolli. *Information Foraging Theory: Adaptive Interaction with Information*. Oxford University Press, New York, NY, 2007.

[89] Peter Pirolli and Stuart Card. Information foraging. *Psychological Review*, 106 (4):643–675, 1999.

[90] R. J. Prokopy, B. D. Roitberg, and R. I. Vargas. Effects of egg load on finding and acceptance of host fruit in *Ceratitis capitata* flies. *Physiological Entomology*, 19(2):124–132, 1994. DOI:10.1111/j.1365-3032.1994.tb01085.x.

[91] H. Ronald Pulliam. On the theory of optimal diets. *American Naturalist*, 108 (959):59–74, January–February 1974.

[92] Graham H. Pyke, H. Ronald Pulliam, and Eric L. Charnov. Optimal foraging: a selective review of theory and tests. *Quarterly Review of Biology*, 52(2):137–154, 1977.

[93] Dongyu Qiu and Rayadurgam Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. In Raj Yavatkar, Ellen W. Zegura, and Jennifer Rexford, editors, *Proceedings of the ACM SIGCOMM 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 367–378, Portland, Oregon, USA, August 30 – September 3, 2004. ISBN 1-58113-862-8. DOI:10.1145/1015467.1015508.

[94] Gwenael Quaintenne, Jan A. van Gils, Pierrick Bocher, Anne Dekinga, and Theunis Piersma. Diet selection in a molluscivore shorebird across Western Europe: does it show short- or long-term intake rate-maximization? *Journal of Animal Ecology*, 79(1):53–62, January 2010. DOI:10.1111/j.1365-2656.2009.01608.x.

[95] Nicanor Quijano and Kevin M. Passino. The ideal free distribution: theory and engineering application. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):154–165, February 2007. DOI:10.1109/TSMCB.2006.880134.

[96] Nicanor Quijano and Kevin M. Passino. Honey bee social foraging algorithms for resource allocation: theory and application. *Engineering Applications of Artificial Intelligence*, 2010. To appear.

[97] Nicanor Quijano, Burton W. Andrews, and Kevin M. Passino. Foraging theory for multizone temperature control. *IEEE Computational Intelligence Magazine*, 1(4):18–27, November 2006. DOI:10.1109/MCI.2006.329704.

[98] Howard Rachlin and Leonard Green. Commitment, choice and self-control. *Journal of the Experimental Analysis of Behavior*, 17(1):15–22, January 1972.

[99] Dickon Reed, Ian Pratt, Paul Menage, Stephen Early, and Neil Stratford. Xenoservers: accountable execution of untrusted programs. In *Proceedings of the Seventh Workshop on Hot Topics in Operating Systems*, pages 136–141, Rio Rico, Arizona, USA, March 28–30, 1999. DOI:10.1109/HOTOS.1999.798390.

[100] Jay A. Rosenheim. An evolutionary argument for egg limitation. *Evolution*, 50 (5):2089–2094, October 1996.

[101] Jay A. Rosenheim and David Rosen. Foraging and oviposition decisions in the parasitoid *Aphytis lingnanesis*: distinguishing the influences of egg load and experience. *Journal of Animal Ecology*, 60(3):873–893, October 1991.

[102] Edward E. Ruppert, Richard S. Fox, and Robert D. Barnes. *Invertebrate Zoology: A Functional Evolutionary Approach*. Brooks/Cole Publishing, Belmont, CA, seventh edition, 2004. ISBN 978-0030259821.

[103] Thomas W. Schoener. Theory of feeding strategies. *Annual Review of Ecology and Systematics*, 2:369–404, 1971.

[104] Eric Siegel and Howard Rachlin. Soft commitment: self-control achieved by response persistence. *Journal of the Experimental Analysis of Behavior*, 64(2): 117–128, September 1995.

[105] Andrew Sih and Bent Christensen. Optimal diet theory: when does it work, and when and why does it fail? *Animal Behaviour*, 61(2):379–390, February 2001. DOI:10.1006/anbe.2000.1592.

[106] Reid G. Smith. The contract net protocol: high-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12): 1104–1113, December 1980. DOI:10.1109/TC.1980.1675516.

[107] Mark Snyderman. Optimal prey selection: the effects of food deprivation. *Behaviour Analysis Letters*, 3:359–369, 1983.

[108] Barry M. Staw. The escalation of commitment to a course of action. *Academy of Management Review*, 6(4):577–587, October 1981.

[109] David W. Stephens. Discrimination, discounting and impulsivity: a role for an informational constraint. *Philosophical Transactions of the Royal Society B*, 357(1427):1527–1537, 2002. DOI:10.1098/rstb.2002.1062.

[110] David W. Stephens and Dick Anderson. The adaptive value of preference for immediacy: when shortsighted rules have farsighted consequences. *Behavioral Ecology*, 12(3):330–339, 2001.

[111] David W. Stephens and Eric L. Charnov. Optimal foraging: some simple stochastic models. *Behavioral Ecology and Sociobiology*, 10:251–263, 1982.

[112] David W. Stephens and John R. Krebs. *Foraging Theory*. Princeton University Press, Princeton, NJ, 1986.

[113] David W. Stephens and Collen M. McLinn. Choice and context: test a simple short-term choice rule. *Animal Behaviour*, 66(1):59–70, July 2003. DOI:10.1006/anbe.2003.2177.

[114] David W. Stephens, Benjamin Kerr, and Esteban Fernández-Juricic. Impulsiveness without discounting: the ecological rationality hypothesis. *Proceedings of the Royal Society B*, 271(1556):2459–2465, 2004. DOI:10.1098/rspb.2004.2871.

[115] David W. Stephens, Joel S. Brown, and Ronald C. Ydenberg, editors. *Foraging: Behavior and Ecology*. University of Chicago Press, Chicago, IL, 2007.

[116] L. D. Stone. *Theory of Optimal Search*. Academic Press, 1975.

[117] Alan R. Templeton and Lawrence R. Lawlor. The fallacy of the averages in ecological optimization theory. *American Naturalist*, 117(3):390–393, March 1981.

[118] Jan A. van Gils, Ingrid W. Schenk, Oscar Bos, and Theunis Piersma. Incompletely informed shorebirds that face a digestive constraint maximize net energy gain when exploiting patches. *American Naturalist*, 161(5):777–793, May 2003. DOI:10.1086/374205.

[119] Jan A. van Gils, Sem R. de Rooij, Jelmer van Belle, Jaap van der Meer, Anne Dekinga, Theunis Piersma, and Rudi Drent. Digestive bottleneck affects foraging decisions in red knots *Calidris canutus*. I. prey choice. *Journal of Animal Ecology*, 74(1):105–119, January 2005. DOI:10.1111/j.1365-2656.2004.00903.x.

[120] Markku Verkama, Harri Ehtamo, and Raimo P. Hämäläinen. Distributed computation of pareto solutions in n-player games. *Mathematical Programming*, 74: 29–45, 1996.

[121] Chris Verlinden and R. Haven Wiley. The constraints of digestive rate: an alterantive model of diet selection. *Evolutionary Ecology*, 3(3):264–272, July 1989. DOI:10.1007/BF02270727.

[122] Éric Wajnberg. Time allocation strategies in insect parasitoids: from ultimate predictions to proximate behavioral mechanisms. *Behavioral Ecology and Sociobiology*, 60(5):589–611, September 2006. DOI:10.1007/s00265-006-0198-9.

[123] Carl A. Waldspurger, Tad Hogg, Bernardo A. Huberman, Jeffrey O. Kephart, and W. Schott Stornetta. Spawn: a distributed computational economy. *IEEE Transactions on Software Engineering*, 18(2):103–117, February 1992. DOI:10.1109/32.121753.

[124] Yao-Jung Wen and Alice M. Agogino. Wireless networked lighting systems for optimizing energy savings and user satisfaction. In *Proceedings of Wireless Hive Networks Conference*, Austin, TX, USA, 2008.

[125] Christopher J. Whelan and Joel S. Brown. Optimal foraging and gut constraints: reconciling two schools of thought. *Oikos*, 110(3):481–496, September 2005. DOI:10.1111/j.0030-1299.2005.13387.x.

[126] James E. White. Telescript technology: mobile agents. In Dejan Milojičić, Frederick Douglis, and Richard Wheeler, editors, *Mobility: Processes, Computers, and Agents*, pages 460–493. ACM Press, New York, 1999. ISBN 0-201-37928-7.

[127] K. David Young, Vadim I. Utkin, and Ümit Özgüner. A control engineer's guide to sliding mode control. *IEEE Transactions on Control Systems Technology*, 7 (3):328–342, May 1999. DOI:10.1109/87.761053.