

Generalizing foraging theory for analysis and design

Theodore P Pavlic¹ and Kevin M Passino²

Abstract

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. Prior foraging-inspired designs are direct applications of classical optimal foraging theory (OFT). Here, 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 paper 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 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.

Keywords

Agent-based models, biomimicry, decision making, Markov renewal processes, mathematical biology, optimization, solitary agent behavior

1. Introduction

Foraging theory has been a source of inspiration for optimization (Passino 2002, 2005), autonomous vehicle control (Andrews et al. 2004; Quijano et al. 2006; Andrews et al. 2007a; Pavlic and Passino 2009), and distributed resource allocation (Finke et al. 2006; Andrews et al. 2007b; Finke and Passino 2007; Quijano and Passino 2007). 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. In addition, 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 paper, we identify the key structures responsible for

¹Department of Computer Science and Engineering, Ohio State University, Columbus, OH, USA

²Department of Electrical and Computer Engineering, Ohio State University, Columbus, OH, USA

Corresponding author:

Theodore P Pavlic, Department of Computer Science and Engineering, Ohio State University, 395 Drees Laboratories, 2015 Neil Avenue, Columbus, OH 43210, USA.

Email: pavlic.3@osu.edu

that simplicity so that optimization objectives that better fit engineering scenarios can lead to similar foraging-like designs. Thus, this paper extends the work of Andrews et al. (2007a) 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 (Andrews et al. 2004; Quijano et al. 2006; Andrews et al. 2007a). 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 paper is structured as follows. In Section 2, 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. In addition, we provide motivating examples from the literature of existing applications of foraging theory to engineering. In Section 2.3, 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 performing 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 (CR) maximization (Charnov 1973, 1976; Stephens and Krebs 1986), 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 3, a graphical approach to multivariate optimization of advantage-to-disadvantage functions is discussed, and a more rigorous quantitative approach is explored in Section 4. Algorithms based on that approach are given in Appendix A 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 5. In addition, simulation results are given that show how behaviors developed with the methods in this paper have better performance in finite-lifetime scenarios when compared with conventional foraging-inspired task-choice behaviors. Finally, some concluding remarks and suggestions for future research are given in Section 6.

2. 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 2.1, and the related optimization problem from classical foraging theory is presented in Section 2.2. Then, in Section 2.3, we present four new optimization objectives that are better fits 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 Sections 3 and 4. 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 5, which compares the results of applying the analytical methods in Section 4 to each of the 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 $\mathbf{p} \triangleq [p_1, p_2, \dots, p_n]^T$ and $\boldsymbol{\tau} \triangleq [\tau_1, \tau_2, \dots, \tau_n]^T$. Next, let \mathbb{R} be the set of the real numbers, $\mathbb{R}_{\geq 0}$ be the set of non-negative real numbers, and $\bar{\mathbb{R}}_{\geq 0} \triangleq \mathbb{R}_{\geq 0} \cup \{\infty\}$. 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 $\tau_i^-, \tau_i^+ \in \bar{\mathbb{R}}_{\geq 0}$ so that the *feasible set* of behaviors is

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

which is a convex separable polyhedron. The optimal behavior will maximize the generic *advantage-to-disadvantage function* (Pavlic 2007)

$$J(\mathbf{p}, \boldsymbol{\tau}) \triangleq \frac{A(\mathbf{p}, \boldsymbol{\tau})}{D(\mathbf{p}, \boldsymbol{\tau})} \triangleq \frac{a + \sum_{i=1}^n p_i a_i(\tau_i)}{d + \sum_{i=1}^n p_i d_i(\tau_i)}, \quad (2)$$

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

2.1. Background: foraging-inspired task-processing agents

OFT was popularized by Stephens and Krebs (1986). It is based on the work of Charnov (1973), and recently updated results and new applications have been summarized by Stephens et al. (2007). 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 $i \in \{1, 2, \dots, n\}$ is encountered at 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 $\lambda \triangleq \lambda_1 + \lambda_2 + \dots + \lambda_n$ (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 (1986) 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 (Andrews et al. 2004, 2007a; Pavlic and Passino 2009), resource allocation strategies for distributed temperature regulation (Quijano et al. 2006), and Web sites that attract attention of humans on the Internet (Pirolli and Card 1999; Pirolli 2005, 2007). 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.

2.1.1. Autonomous mobile vehicles Andrews et al. (2007a) show how both the prey and patch models described by

Stephens and Krebs (1986) can be used to model an AAV (e.g. for military or surveillance applications). In particular, they consider a Dubins's car (Dubins 1957) 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 a 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 $i \in \{1, 2, \dots, n\}$. Furthermore, Andrews et al. (2007a) use results from Stone (1975) 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 $i \in \{1, 2, \dots, n\}$ 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. (2007a) 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. (2007a) as well as the temperature regulation example described in the following, 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. In addition, 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 (2010) 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.

2.1.2. Resource allocation: distributed temperature regulation Quijano et al. (2006) 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 such as that described by Passino (2002, 2005). 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. (2006) 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.

2.1.3. Web design Pirolli (2007) 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 Web sites 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.

2.2. Classical optimal foraging objective

In Section 2.1, we described several examples of how OFT has been used in the 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 2.3, 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 (1973) and Pyke et al. (1977) suggest that the life-time 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 (Schoener 1971). Unfortunately, for any finite lifetime, this optimization objective strongly depends on precise knowledge of how gain and time covary (Charnov 1973; Pavlic 2007). 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 (1973) 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 $(\mathbf{p}, \boldsymbol{\tau}) \in \mathcal{F}$ should maximize the stochastic limit of total gain to total time (Pavlic 2007). That is, the behavior should maximize the *rate*

$$\frac{\sum_{i=1}^n \lambda_i p_i (g_i(\tau_i) - c_i(\tau_i)) - c^s}{1 + \sum_{i=1}^n \lambda_i p_i \tau_i}, \quad (3)$$

which matches Equation (2) with

$$\begin{aligned} a &\triangleq -c^s, & a_i(\tau_i) &\triangleq \lambda_i (g_i(\tau_i) - c_i(\tau_i)), \\ d &\triangleq 1, & d_i(\tau_i) &\triangleq \lambda_i \tau_i. \end{aligned} \quad (4)$$

The prey model lets $\tau_i^- \triangleq \tau_i^+$ for each task type $i \in \{1, 2, \dots, n\}$ and finds the optimal $\mathbf{p} \in [0, 1]^n$, and the patch model lets $p_i^- \triangleq p_i^+ \triangleq 1$ for each patch type $i \in \{1, 2, \dots, n\}$ and finds the optimal $\boldsymbol{\tau} \in [0, \infty)^n$ (Stephens and Krebs 1986).

The expectation of ratios. Some observational evidence (e.g. Nonacs 2001) contradicts predictions from the *marginal value theorem* (MVT), which is the principle result of the patch model (Charnov 1973, 1976; Stephens and Charnov 1982; Stephens and Krebs 1986). In response, arguments from Templeton and Lawlor (1981) have been used as fodder for *expectation-of-ratios* (EoR) (Harder and Real 1987; Bateson and Kacelnik 1996; Bateson and Whitehead 1996) objective functions of the form

$$\sum_{i=1}^n \frac{\lambda_i}{\lambda} p_i \frac{g_i(\tau_i) - c_i(\tau_i) - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \tau_i}, \quad (5)$$

which matches Equation (2) with

$$\begin{aligned} a &\triangleq 0, & a_i(\tau_i) &\triangleq \frac{\lambda_i}{\lambda} \frac{g_i(\tau_i) - c_i(\tau_i) - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \tau_i}, & d &\triangleq 1, \\ & & d_i(\tau_i) &\triangleq 0. \end{aligned}$$

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.

2.3. 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. In Section 2.3.1, we focus on criticisms of the OFT formulation for cases where task-processing agents cannot be assumed to have unending operation. Then, in Section 2.3.2, we introduce a novel optimization model of an autonomous task-processing agent that may better fit applications that are less suitable for OFT.

2.3.1. 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.

To handle cases with survival thresholds over short times, Stephens and Charnov (1982) 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. (2007a) 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 (2006), 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 (Rosenheim and Rosen 1991; Minkenberg et al. 1992; Fletcher et al.

1994; Prokopy et al. 1994; Rosenheim 1996; Heimpel and Rosenheim 1998). 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 2.3.2, 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.

2.3.2. Autonomous agent model for finite-event scenario

Consider a task-processing agent similar to that described in Section 2.1. 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 $\lambda \triangleq \lambda_1 + \dots + \lambda_n$ 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 $\lambda_1 + \dots + \lambda_n$, it is convenient to focus on the Markov renewal process that renews at every processed encounter at the lower rate of $p_1\lambda_1 + p_2\lambda_2 + \dots + p_n\lambda_n$. 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

$$E(G(N)) = N \frac{\sum_{i=1}^n \lambda_i p_i (g_i(\tau_i) - c_i(\tau_i)) - c^s}{\sum_{i=1}^n \lambda_i p_i}, \quad (6)$$

$$E(C(N)) = N \frac{\sum_{i=1}^n \lambda_i p_i c_i(\tau_i) + c^s}{\sum_{i=1}^n \lambda_i p_i}, \quad (7)$$

and

$$E(T(N)) = N \frac{1 + \sum_{i=1}^n \lambda_i p_i \tau_i}{\sum_{i=1}^n \lambda_i p_i}. \quad (8)$$

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

$$\frac{E(G(N)) - G^T}{E(T(N))} = \frac{\sum_{i=1}^n \lambda_i p_i (g_i(\tau_i) - c_i(\tau_i)) - \frac{G^T}{N} - c^s}{1 + \sum_{i=1}^n \lambda_i p_i \tau_i}. \quad (9)$$

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 $N \rightarrow \infty$, finite-event excess-rate (ER) 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* (Houston and McNamara 1999) 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

$$E(G(N)) - G^T - wE(T(N)) = \frac{\sum_{i=1}^n \lambda_i p_i \left(g_i(\tau_i) - c_i(\tau_i) - \frac{G^T}{N} - w\tau_i \right) - c^s - w}{\sum_{i=1}^n \lambda_i p_i}, \quad (10)$$

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 $p_1^- = p_2^- = \dots = p_n^- = 0$, 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 ER 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 (1986) 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. In addition, 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

$$\frac{E(G(N)) + E(C(N)) - G_g^T}{E(C(N))} = \frac{\sum_{i=1}^n \lambda_i p_i \left(g_i(\tau_i) - \frac{G_g^T}{N} \right)}{c^s + \sum_{i=1}^n \lambda_i p_i c_i(\tau_i)}. \quad (11)$$

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

$$E(G(N)) + E(C(N)) - G_g^T - wE(C(N)) = \frac{\sum_{i=1}^n \lambda_i p_i \left(g_i(\tau_i) - \frac{G_g^T}{N} - wc_i(\tau_i) \right) - wc^s}{\sum_{i=1}^n \lambda_i p_i}, \quad (12)$$

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 $p_1^- = \dots = p_n^- = 0$ 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 3. Results of the application of the algorithms described in Section 4 will be given in Section 5.

3. 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 (1986) 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 2. An analytical optimization approach is given in Section 4 along with algorithms that are guaranteed to find an optimal task-processing behavior for certain scenarios.

Because Equation (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. 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 $(\mathbf{p}, \boldsymbol{\tau}) \in \mathcal{F}$. 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 not be convex nor connected, but it should be closed (e.g. it could be a finite set of points).

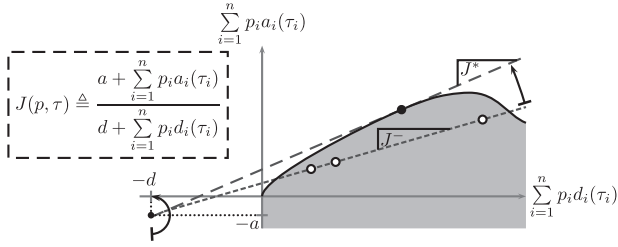


Fig. 1. Graphical optimization of an advantage-to-disadvantage function. Each point in the shaded region corresponds to a different feasible behavior $(\mathbf{p}, \boldsymbol{\tau})$, and the slope of the line connecting that point to $(-d, -a)$ is the value of the objective function $J(\mathbf{p}, \boldsymbol{\tau})$ 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.

3.1. Optimization of the classical objective

For the following, let

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

and $\bar{\tau} \triangleq \sum_{i=1}^n \frac{\lambda_i}{\lambda} p_i \tau_i$.

The average time between encounters is $1/\lambda$, and λ_i/λ is the probability that an encounter is with a task of type $i \in \{1, 2, \dots, n\}$. The expected processing gain, processing cost, and processing time for a single encounter are \bar{g} , \bar{c} , and $\bar{\tau}$, respectively, and the rate of gain in Equation (3) is equivalent to

$$\frac{\sum_{i=1}^n \frac{\lambda_i}{\lambda} p_i (g_i(\tau_i) - c_i(\tau_i)) - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \sum_{i=1}^n \frac{\lambda_i}{\lambda} p_i \tau_i} = \frac{\bar{g} - \bar{c} - \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \bar{\tau}}. \quad (13)$$

For all $i \in \{1, 2, \dots, n\}$, 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 $(\bar{p}, \bar{\tau})$ behavior. Increases in the optimal $\bar{\tau}$ or \bar{g} reflect increased preferences for higher processing times or processing gains, respectively.

Graphical optimization of this function is shown in Figure 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 $\bar{\tau}$ and average net processing gain $(\bar{g} - \bar{c})$. If c^s or $1/\lambda$ increase to beyond

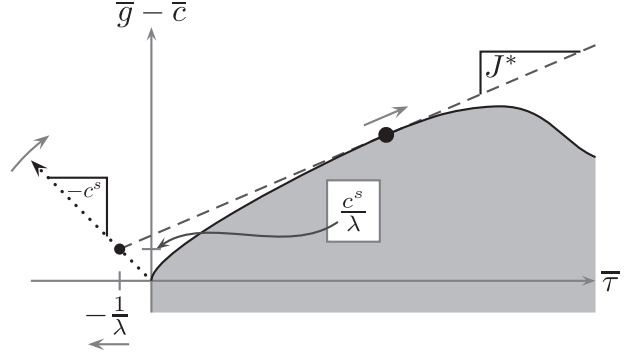


Fig. 2. Graphical optimization of the classical optimization objective. As search cost c^s or interarrival time $1/\lambda$ increases, the mean processing time $\bar{\tau}$ will increase.

the point where c^s/λ matches the $(\bar{g} - \bar{c})$ -peak of the feasible behavior frontier, the optimal average processing time $\bar{\tau}$ will continue to increase although the optimal average net processing gain $(\bar{g} - \bar{c})$ 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.

3.2. Optimal behaviors from alternative objectives

For simplicity in this graphical analysis, assume the special case of patch problems (i.e. $p_i^- = p_i^+ = p_i^* \triangleq 1$ for each $i \in \{1, 2, \dots, n\}$). 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 5 after the analytical methods in Section 4 are introduced. Consider finite-event maximization of the following:

(i) *Excess rate*. In this case, Equation (9) is

$$\frac{\sum_{i=1}^n \lambda_i (g_i(\tau_i) - c_i(\tau_i)) - \sum_{i=1}^n \lambda_i \frac{G^T}{N} - c^s}{1 + \sum_{i=1}^n \lambda_i \tau_i} = \frac{\bar{g} - \bar{c} - \frac{G^T}{N} + \frac{c^s}{\lambda}}{\frac{1}{\lambda} + \bar{\tau}}, \quad (14)$$

which is equivalent to Equation (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 ER 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 ER maximizer as increasing search cost c^s on the infinite-time rate maximizer, and Figure 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 (1982) 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 (1986) 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 ER task-processing model modifies the CR 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 a CR maximizer.

(ii) *Time-discounted net gain*. Under the patch assumption, the time-discounted net-gain (TDNG) objective function in Equation (10) does not have a convenient 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 (10) in the patch case is equivalent to

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

Because $N > 0$ and $(**)$ is constant, TDNG optimization is identical to optimization of $(*) \triangleq (\bar{g} - \bar{c}) - w\bar{\tau}$. So possible solutions come from the dark upper frontier in Figure 2, which corresponds with the behaviors that maximize $(\bar{g} - \bar{c})$ for a given $\bar{\tau}$ and minimize $\bar{\tau}$ for a given $\bar{g} - \bar{c}$ (i.e. the behavior will be Pareto optimal with respect to these two optimization objectives). The particular solution from this frontier is dependent 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 3. Just as in Figure 2, each point in the shaded area of Figure 3(a) is the pair $(\bar{\tau}, \bar{g} - \bar{c})$ corresponding to a particular (p, τ) behavior. In this example, the frontier of the shaded area is smooth and continuous, and so it can be represented as a differentiable function $(\bar{g} - \bar{c})(\bar{\tau})$, and the optimal processing average processing time $\bar{\tau}^*$ is the point where $(\bar{g} - \bar{c})(\bar{\tau}^*) = w$ and $(\bar{g} - \bar{c}) < 0$. 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 3(a); the thick portions of those lines correspond to (c^s, λ) combinations where TDNG and rate maximization are equivalent. As discussed by Houston and McNamara (1999), if w is set to the maximal value of Equation (14) (i.e. the

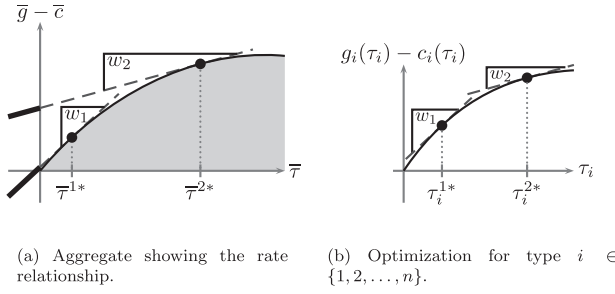


Fig. 3. Time-discounted net-gain optimization. The shaded area used in graphical rate maximization is also used in (a); however, the optimal TDNG behavior corresponds to the point of tangency with a line of slope w . Here, the steeper cost rate $w_1 > w_2$ is associated with a shorter average time $\bar{\tau}^{1*} < \bar{\tau}^{2*}$ 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 .

maximum long-term rate of gain), the corresponding gain–time tradeoff will also maximize long-term rate of gain.

In Section 4, we give precise analytical methods for optimization of this function. Meanwhile, we observe that because $(\bar{g} - \bar{c}) - w\bar{\tau}$ is a weighted sum, then for each type $i \in \{1, 2, \dots, n\}$, 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 $i \in \{1, 2, \dots, n\}$, if the 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 to the optimization of the aggregate. As shown in Figure 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 3(a), once the optimal processing time is found for every type, the line with slope w that intersects $(\bar{\tau}^*, (\bar{g} - \bar{c})^*)$ 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 (11) is

$$\frac{\sum_{i=1}^n \lambda_i g_i(\tau_i) - \sum_{i=1}^n \lambda_i \frac{G_g^T}{N}}{c^s + \sum_{i=1}^n \lambda_i c_i(\tau_i)} = \frac{\bar{g} - \frac{G_g^T}{N}}{\frac{c^s}{\lambda} + \bar{c}}, \quad (15)$$

which resembles Equation (14) and has optimization depicted by Figure 4(a).

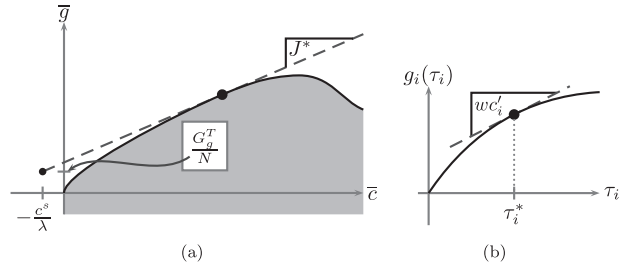


Fig. 4. Optimization based on efficiency. In (a), excess-efficiency maximization is shown to be similar to excess-rate maximization. Here, as search cost c^s or interarrival time $1/\lambda$ increases, the mean processing cost \bar{c} will increase. So long as processing cost increases are due to processing time increases, mean processing time $\bar{\tau}$ will also increase. This result qualitatively matches what is expected for rate maximization. In (b), cost-discounted gain (CDG) optimization is shown for one particular type, $i \in \{1, \dots, n\}$ with linear processing cost. 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 .

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 (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 (12) is

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

which is maximized at the same point as $\bar{g} - w\bar{c}$. As in TDNG optimization, optimization of each type can be decoupled from the other types. In particular, for each $i \in \{1, 2, \dots, n\}$, 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 4(b). That is, optimization is nearly identical to the TDNG case except that the processing time in type $i \in \{1, 2, \dots, n\}$ 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 CR maximization results. Hence, these optimization objectives can be used to model behaviors that do not perfectly fit the classical foraging model.

The EoR objective in Equation (5) apparently does not have a convenient structure for graphical optimization. In Section 4, we give analytical strategies for its optimization. Meanwhile, to motivate a graphical optimization

method, we observe that because Equation (5) is a weighted sum, then it can be shown that optimization of Equation (5) reduces to optimization of $(g_i(\tau_i) - c_i(\tau_i) - c^s/\lambda) / (1/\lambda + \tau_i)$ for each $i \in \{1, 2, \dots, n\}$. Each of these functions is an advantage-to-disadvantage function nearly identical to Equation (14) when $n = 1$, and so the standard graphical optimization procedure can be applied to each type separately. However, although EoR optimization can be completed separately for each type $i \in \{1, 2, \dots, n\}$, the optimal processing times are still related by global parameters c^s and λ .

4. An analytical optimization approach

The graphical approach described in Section 3 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 Appendix A, we provide detailed descriptions of algorithms that are guaranteed to find these points of optimality. However, for brevity, in this section we connect the characterization of a generalized task-processing optimum to the popular algorithms used in OFT-type applications. Finally, we summarize the application of algorithms from Appendix A to the example advantage-to-disadvantage functions described in Section 2, and we list some observations about important similarities and differences in the results.

4.1. Characterization of optimal behaviors

Here, we must characterize the optimality of Equation (2) over the set of behaviors in Equation (1). We give conditions that guarantee that a behavior is a *strict local maximum* of Equation (2). If the optimization objective is *strictly convex*, these conditions describe its *unique global maximum*. Our analysis uses Lagrange multiplier theory (Bertsekas 1995), and so we assume that a_i and d_i are twice continuously differentiable for each type $i \in \{1, 2, \dots, n\}$ in an open neighborhood of the optimal behavior.

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

$$p_j^* = \begin{cases} p_j^- & \text{if } D^* a_j(\tau_j^*) < A^* d_j(\tau_j^*), \\ p_j^+ & \text{if } D^* a_j(\tau_j^*) > A^* d_j(\tau_j^*). \end{cases} \quad (16)$$

Because $p_j = p_j^-$ or $p_j = p_j^+$ for each type $j \in \{1, 2, \dots, n\}$, we call Equation (16) the *extreme-preference rule*. In addition, for each type $j \in \{1, 2, \dots, n\}$, assume that

$$\tau_j^- < \tau_j^* < \tau_j^+ \quad \text{and} \quad \begin{cases} D^* a_j(\tau_j^*) = A^* d_j(\tau_j^*) \\ \quad \text{and} \\ D^* a_j(\tau_j^*) < A^* d_j(\tau_j^*), \end{cases} \quad (17a)$$

or

$$\tau_j^* = \tau_j^- \quad \text{and} \quad D^* a_j(\tau_j^*) < A^* d_j(\tau_j^*), \quad (17b)$$

or

$$\tau_j^* = \tau_j^+ \quad \text{and} \quad D^* a_j(\tau_j^*) > A^* d_j(\tau_j^*). \quad (17c)$$

The condition in Equation (17a) ensures that the interior coordinate is at a stationary point of the objective function with local convexity. The conditions in Equations (16), (17b), and (17c) ensure that the extreme coordinates sit on downward slopes at the edge of the objective function. So Equations (16) and (17) define *sufficiency conditions* for optimality. Under these conditions, $(\mathbf{p}^*, \boldsymbol{\tau}^*)$ must be a strict local maximizer. If Equation (2) is convex everywhere, then the behavior is its unique global maximizer.

4.2. Motivating interpretations

Detailed algorithms for finding points that meet the described optimality conditions are given in Appendix A. Here, we show how the conditions in Equations (16) and (17) are natural generalizations of familiar classical foraging theory and present summaries of the existing OFT algorithms to motivate the general cases in Appendix A. 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.

4.2.1. Prey model as optimal task-type choice: profitability ordering When applied to Equation (3) for the prey-model case (i.e. when it is given that $\tau_i^+ = \tau_i^- = \tau_i^*$ and $p_i^- = 0$ and $p_i^+ = 1$ for each type i), the extreme-preference rule in Equation (16) is equivalent to

$$p_j^* = \begin{cases} 0 & \text{if } \frac{g_j(\tau_j^*) - c_j(\tau_j^*)}{\tau_j^*} < \frac{\sum_{i=1}^n \lambda_i p_i^* (g_i(\tau_i^*) - c_i(\tau_i^*)) - c^s}{1 + \sum_{i=1}^n \lambda_i p_i^* \tau_i^*}, \\ 1 & \text{if } \frac{g_j(\tau_j^*) - c_j(\tau_j^*)}{\tau_j^*} > \frac{\sum_{i=1}^n \lambda_i p_i^* (g_i(\tau_i^*) - c_i(\tau_i^*)) - c^s}{1 + \sum_{i=1}^n \lambda_i p_i^* \tau_i^*}, \end{cases} \quad (18)$$

which is the familiar *zero-one rule* (Stephens and Krebs 1986) where $a_j(\tau_j^*)/d_j(\tau_j^*)$ is the *profitability* $g_j(\tau_j^*)/\tau_j^*$ of type $j \in \{1, 2, \dots, n\}$. This rule states that if task types are indexed by profitability so that

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

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

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

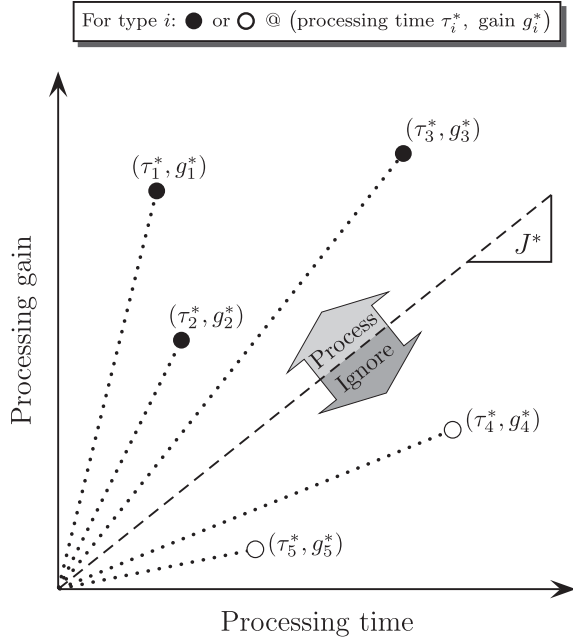


Fig. 5. Graphical summary of the prey-model result. For a task type $i \in \{1, 2, 3, 4, 5\}$, the average processing time τ_i^* and average net gain g_i^* are plotted as a dot. The maximum long-term rate of gain J^* is the slope of the dashed line which separates the processed types, 1, 2, and 3, from 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.

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

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

where optimal net gain $g_j^* \triangleq g_j(\tau_j^*) - c_j(\tau_j^*)$ for each $j \in \{1, 2, \dots, n\}$. This relationship is depicted in Figure 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 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 \mathbf{p} preference profiles. However, because of the invariant task-type ordering, there are at most $n + 1$ possible \mathbf{p} vectors that must to be checked (i.e. the preference vectors $[0, 0, \dots, 0]^T$, $[1, 0, \dots, 0]^T$, $[1, 1, \dots, 0]^T$, ..., and $[1, 1, \dots, 1]^T$).

4.2.2. Patch model as an optimal processing-time choice: marginal value When Equation (17) is applied to Equation (3) for the patch model case (i.e. when it is given that $p_i^- = p_i^+ = p_i^* = 1$ and $\tau_i^- = 0$ and $\tau_i^+ = \infty$ for each type i), Equation (17a) is equivalent to

$$\begin{aligned} \tau_j^* > 0 \quad \text{and} \quad g_j(\tau_j^*) - c_j(\tau_j^*) &< 0 \\ \text{and} \\ g_j(\tau_j^*) - c_j(\tau_j^*) &= \frac{\sum_{i=1}^n \lambda_i p_i^* (g_i(\tau_i^*) - c_i(\tau_i^*)) - c^s}{1 + \sum_{i=1}^n \lambda_i p_i^* \tau_i^*}, \end{aligned} \quad (19)$$

which is the familiar marginal value theorem (Charnov 1973, 1976). Consider the special single-type patch case where $n = 1$, $p_1^- = p_1^+ = 1$, $\tau_1^- = 0$, and $\tau_1^+ = \infty$. Then Equation (19) is equivalent to

$$g_1(\tau_1^*) - c_1(\tau_1^*) = \frac{g_1(\tau_1^*) - c_1(\tau_1^*) - \frac{c^s}{\lambda_1}}{\frac{1}{\lambda_1} + \tau_1^*}. \quad (20)$$

In addition, the graphical analysis in Figure 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 $\tau_1 \rightarrow g_1(\tau_1) - c_1(\tau_1)$. The resulting graph is exactly the situation described by Equation (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 $(-1/\lambda_1, c^s/\lambda_1)$.

5. 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 4 to the example functions from Section 2. In Section 5.1, the salient theoretical differences between the resulting optimal behaviors are compared. In Section 5.2, 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 paper.

5.1. Comparison of theoretical results

Applying the behavioral-design algorithms from Section 2 and Appendix A yields the generalized profitabilities and

MVT conditions summarized in Table 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.

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 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 TDNG case, the CDG case, and the 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 3, 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 $(g_1(\tau_1), \tau_1) = (\$5, 5 \text{ s})$ and $(g_2(\tau_2), \tau_2) = (\$25, 25 \text{ s})$. 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 (1982) that is applied to an autonomous vehicle problem by Andrews et al. (2007a) 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. (1984). As discussed in Section 2.3.1, the risk-sensitive models of Stephens and Charnov (1982) 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.

5.2. Simulation results

Table 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. (2007a) 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 2, some additional trivial strategies that are relatively simple to analyze can be used as benchmarks. For example:

Table 1. Sample optimization results for type $i \in \{1, 2, \dots, n\}$. 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 for interior processing times.

Objective	Generalized profitability	Generalized MVT condition
Classical	$\frac{g_i(\tau_i) - c_i(\tau_i)}{\tau_i}$	$g_i(\tau_i) - c_i(\tau_i) = J(\mathbf{p}, \mathbf{\tau})$
ER	$\frac{g_i(\tau_i) - c_i(\tau_i) - G^T/N}{\tau_i}$	$g_i(\tau_i) - c_i(\tau_i) = J_{ER}(\mathbf{p}, \mathbf{\tau})$
TDNG	$g_i(\tau_i) - c_i(\tau_i) - w\tau_i$	$g_i(\tau_i) - c_i(\tau_i) = w$
EE	$\frac{g_i(\tau_i) - G^T/N}{c_i(\tau_i)}$	$\frac{g_i(\tau_i)}{c_i(\tau_i)} = J_{EE}(\mathbf{p}, \mathbf{\tau})$
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 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 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 (2010) 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 G 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 T 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.

$N = 300$ tasks per mission, 100 Monte Carlo samples						
$G^T = 6,000$		Take all	CR	ER	eCR	eER
	G :	16,555 \pm 35	10,954 \pm 17	20,520 \pm 24	11,172 \pm 113	16,534 \pm 46
	$@G^T$:	100%	100%	100%	100%	100%
	T :	11,107 \pm 42	4,399 \pm 9	9,242 \pm 13	4,541 \pm 55	9,855 \pm 32
$G^T = 9,000$		Take all	CR	ER	eCR	eER
	G :	16,565 \pm 30	10,946 \pm 16	20,473 \pm 25	11,218 \pm 128	18,119 \pm 38
	$@G^T$:	100%	100%	100%	98%	100%
	T :	11,119 \pm 42	4,391 \pm 8	9,227 \pm 13	4,567 \pm 63	11,668 \pm 43
$G^T = 13,500$		Take all	CR	ER	eCR	eER
	G :	16,642 \pm 33	10,958 \pm 16	25,153 \pm 11	11,270 \pm 103	18,647 \pm 44
	$@G^T$:	100%	0%	100%	5%	100%
	T :	11,158 \pm 38	4,393 \pm 8	15,645 \pm 42	4,586 \pm 50	12,779 \pm 46
$G^T = 16,500$		Take all	CR	ER	eCR	eER
	G :	16,546 \pm 34	10,993 \pm 16	25,141 \pm 14	10,965 \pm 91	18,796 \pm 39
	$@G^T$:	55%	0%	100%	0%	100%
	T :	11,092 \pm 40	4,421 \pm 8	15,605 \pm 53	4,440 \pm 43	13,120 \pm 44

(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 $g_4 = 100$. For each of these N tasks, there is

a c^s search cost per unit time and an average searching time of $1/\lambda_4 = 10$ time units; 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 $(N)(\tau_4 + 1/\lambda_4) = (300)(110 + 1/0.1) = 36,000$ 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 of more than double the average mission time of the ER strategy for the $G^T = 16,500$ 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 8,940, and the average total time is 3,600 time units. Despite the high total gain–total time ratio of this strategy, it completes the $N = 300$ tasks so quickly that it does not meet three of the example G^T thresholds from Table 2.
- (c) An agent could wait to accept only tasks of type 3 because that type has the highest ER profitability for the $G^T = 16,500$ 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 23,925, and the average total time is 11,250 time units. This simple strategy achieves all four success thresholds in less than a third of the time required for the strategy in (a) that also is uniformly successful.

Both of the single-type strategies in (a) and (b) 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. In addition, 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 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 $G^T = 16,500$, the strategy does worse than others that avoid low-gain tasks and instead search longer for high-gain tasks.
- The 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. (2007a); however, this idealized case is presented here for comparison with the estimated classical rate (eCR) strategy described in the following. 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 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 $G^T = 9,000$ to $G^T = 13,500$ 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 excess rate (eER) strategy described in the following.
- The eCR strategy uses the simple behavioral heuristic described by Pavlic and Passino (2010) 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 to the CR strategy.
- The 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 paper.

6. Conclusions

In this paper, we have summarized 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 CR maximization with shortfall minimization (i.e. risk sensitivity). In addition, 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 of 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. In addition, 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 finding the optimal prey (i.e. task-type) choice or patch residence time (i.e. task-processing time). However, Pavlic and Passino (2009) apply foraging behaviors described by Gendron and Staddon (1983) 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 non-trivial. 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).

Funding

The work in this paper was partially supported by the National Science Foundation under grant number EECS-0931669.

Conflict of interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We thank Thomas A. Waite for his helpful insights and instruction in behavioral ecology. We are also grateful to Ian M Hamilton

and three anonymous reviewers for their helpful comments and suggestions about this manuscript.

References

- Andrews BW, Passino KM and Waite TA (2004) Foraging theory for decision-making system design: task-type choice. In *Proceedings of the 43rd IEEE Conference on Decision and Control*, Vol. 5, Nasssau, Bahamas, pp. 4740–4745.
- Andrews BW, Passino KM and Waite TA (2007a) Foraging theory for autonomous vehicle decision-making system design. *Journal of Intelligent and Robotic Systems* 49: 39–65.
- Andrews BW, Passino KM and Waite TA (2007b) Social foraging theory for robust multiagent system design. *IEEE Transactions on Automation Science and Engineering* 4: 79–86.
- Bateson M and Kacelnik A (1996) Rate currencies and the foraging starling: the fallacy of the averages revisited. *Behavioral Ecology* 7: 341–352.
- Bateson M and Whitehead SC (1996) The energetic costs of alternative rate currencies in the foraging starling. *Ecology* 77: 1303–1307.
- Bertsekas DP (1995) *Nonlinear Programming*. Belmont, MA: Athena Scientific.
- Charnov EL (1973) *Optimal Foraging: Some Theoretical Explorations*. Ph.D. thesis, University of Washington.
- Charnov EL (1976) Optimal foraging: the marginal value theorem. *Theoretical Population Biology* 9: 129–136.
- Dubins LE (1957) 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.
- Finke J and Passino KM (2007) Stable cooperative vehicle distributions for surveillance. *Journal of Dynamic Systems, Measurement, and Control* 129: 597–608.
- Finke J, Passino KM and Sparks AG (2006) Stable task load balancing for cooperative control of networked autonomous air vehicles. *IEEE Transactions on Control Systems Technology* 14: 789–803.
- Fletcher JP, Hughes JP and Harvey IF (1994) Life expectancy and egg load affect oviposition decisions of a solitary parasitoid. *Proceedings: Biological Sciences* 258: 163–167.
- Gendron RP and Staddon JER (1983) Searching for cryptic prey: the effect of search rate. *American Naturalist* 121: 172–186.
- Harder LD and Real LA (1987) Why are bumble bees risk averse? *Ecology* 68: 1104–1108.
- Heimpel GE and Rosenheim JA (1998) Egg limitation in parasitoids: a review of the evidence and a case study. *Biological Control* 11: 160–168.
- Houston AI and McNamara JM (1999) *Models of Adaptive Behavior*. Cambridge: Cambridge University Press.
- Iwasa Y, Suzuki Y and Matsuda H (1984) Theory of oviposition strategy of parasitoids. I. effect of mortality and limited egg number. *Theoretical Population Biology* 26: 205–227.
- Minkenberg OPJM, Tatar M and Rosenheim JA (1992) Egg load as a major source of variability in insect foraging and oviposition behavior. *Oikos* 65: 134–142.
- Nonacs P (2001) State dependent behavior and the marginal value theorem. *Behavioral Ecology* 12: 71–83.
- Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Systems Magazine* 22: 52–67.

- Passino KM (2005) *Biomimicry for Optimization, Control, and Automation*. London: Springer-Verlag.
- Pavlic TP (2007) *Optimal Foraging Theory Revisited*. Master's thesis, The Ohio State University, Columbus, OH, http://www.ohiolink.edu/etd/view.cgi?acc_num=osu1181936683.
- Pavlic TP and Passino KM (2009) Foraging theory for autonomous vehicle speed choice. *Engineering Applications of Artificial Intelligence* 22: 482–489.
- Pavlic TP and Passino KM (2010) When rate maximization is impulsive. *Behavioral Ecology and Sociobiology* 64: 1255–1265.
- Pirolli P (2005) Rational analyses of information foraging on the web. *Cognitive Science* 29: 343–373.
- Pirolli P (2007) *Information Foraging Theory: Adaptive Interaction with Information*. New York: Oxford University Press.
- Pirolli P and Card S (1999) Information foraging. *Psychological Review* 106: 643–675.
- Prokopy RJ, Roitberg BD and Vargas RI (1994) Effects of egg load on finding and acceptance of host fruit in *Ceratitis capitata* flies. *Physiological Entomology* 19: 124–132.
- Pyke GH, Pulliam HR and Charnov EL (1977) Optimal foraging: a selective review of theory and tests. *Quarterly Review of Biology* 52: 137–154.
- Quijano N, Andrews BW and Passino KM (2006) Foraging theory for multizone temperature control. *IEEE Computational Intelligence Magazine* 1: 18–27.
- Quijano N and Passino KM (2007) The ideal free distribution: theory and engineering application. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 37: 154–165.
- Rosenheim JA (1996) An evolutionary argument for egg limitation. *Evolution* 50: 2089–2094.
- Rosenheim JA and Rosen D (1991) Foraging and oviposition decisions in the parasitoid *Aphytis lingnanensis*: distinguishing the influences of egg load and experience. *Journal of Animal Ecology* 60: 873–893.
- Schoener TW (1971) Theory of feeding strategies. *Annual Review of Ecology and Systematics* 2: 369–404.
- Stephens DW, Brown JS and Ydenberg RC (eds) (2007) *Foraging: Behavior and Ecology*. Chicago, IL: University of Chicago Press.
- Stephens DW and Charnov EL (1982) Optimal foraging: some simple stochastic models. *Behavioral Ecology and Sociobiology* 10: 251–263.
- Stephens DW and Krebs JR (1986) *Foraging Theory*. Princeton, NJ: Princeton University Press.
- Stone LD (1975) *Theory of Optimal Search*. New York: Academic Press.
- Templeton AR and Lawlor LR (1981) The fallacy of the averages in ecological optimization theory. *American Naturalist* 117: 390–393.
- Wajnberg É (2006) Time allocation strategies in insect parasitoids: from ultimate predictions to proximate behavioral mechanisms. *Behavioral Ecology and Sociobiology* 60: 589–611.

Appendix A: Algorithms for finding an optimal behavior

Now that we have characterized optimal behaviors, we present three algorithms that find an optimal behavior

$(\mathbf{p}^*, \boldsymbol{\tau}^*) \in \mathcal{F}$ 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 $i \in \{1, 2, \dots, n\}$, the functions a_i and d_i are assumed to be twice continuously differentiable.
- The types are ordered by maximum *generalized profitability* so that

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

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 $i \in \{1, 2, \dots, n\}$. In the other case, maximizing a_i/d_i is equivalent to either maximizing or minimizing a_i for all $i \in \{1, 2, \dots, n\}$.

- To satisfy the extreme-preference rule from Equation (16), the n types are partitioned into a high-preference set and a low-preference set. An *optimal pool size* $k^* \in \{0, 1, \dots, n\}$ 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 $i \in \{1, 2, \dots, n\}$ and each $k \in \{0, 1, \dots, n\}$, the *conditional preference* p_i^k is so that

$$p_i^k \triangleq \begin{cases} p_i^+ & \text{if } i \leq k, \\ p_i^- & \text{if } i > k, \end{cases} \quad (21)$$

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

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

A.1. Generalized prey algorithm

Stephens and Krebs (1986) describe a prey-model algorithm that finds a $(\mathbf{p}^*, \boldsymbol{\tau}^*)$ to optimize Equation (3) when $\boldsymbol{\tau}^*$ 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 $\boldsymbol{\tau}^*$. Instead, we only require that the function d_i is *constant* and *non-zero* for each type $i \in \{1, 2, \dots, n\}$.

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

- (i) The function d_j is constant and non-zero.

- (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 (iv), for each type $j \in \{1, 2, \dots, n\}$, 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

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

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

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

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

where p_i^k is defined by Equation (21). By these assumptions, there is an optimal pool size $k^* \in \{0, 1, \dots, n\}$ such that

$$k^* \triangleq \min \left(\left\{ k \in \{0, 1, \dots, n-1\} : \underbrace{\frac{a + \sum_{i=1}^n p_i^k a_i(\tau_i^*)}{d + \sum_{i=1}^n p_i^k d_i(\tau_i^*)}}_{(*)} > \frac{a_{k+1}(\tau_{k+1}^*)}{d_{k+1}(\tau_{k+1}^*)} \right\} \cup \{n\} \right). \quad (23)$$

So k^* can be found iteratively by choosing the smallest $k \in \{0, 1, \dots, n-1\}$ that satisfies the underbraced expression (*). Then, the behavior $(\mathbf{p}^*, \boldsymbol{\tau}^*)$ 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 $j \in \{1, 2, \dots, n\}$ 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.

A.2. Alternative generalized prey algorithm

The algorithm in Appendix A.1 cannot be used with the EoR function in Equation (5) because it has $d_i \equiv 0$ for each type $i \in \{1, 2, \dots, n\}$. Here, we provide a similar algorithm to handle this case and others so long as $d = 0$. The algorithm assigns an *infinite* generalized profitability to each

type $i \in \{1, 2, \dots, n\}$ with $d_i \equiv 0$ and treats all other types in the same manner as in Appendix A.1. That is, types are ranked by their *extended* generalized profitabilities.

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

- (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 those in Appendix A.1. Here, cases with $d = 0$ are excluded in order to include cases with $d_i \equiv 0$ for at least one type $i \in \{1, 2, \dots, n\}$.

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

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

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

and

$$\frac{a_{\ell+1}(\tau_{\ell+1}^*)}{d_{\ell+1}(\tau_{\ell+1}^*)} > \frac{a_2(\tau_2^*)}{d_2(\tau_2^*)} > \dots > \frac{a_{u-2}(\tau_{u-2}^*)}{d_{u-2}(\tau_{u-2}^*)} > \frac{a_{u-1}(\tau_{u-1}^*)}{d_{u-1}(\tau_{u-1}^*)}$$

and

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

For strict local convexity of the solution, assume that:

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

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

where p_i^k is defined by Equation (21). By these assumptions, the optimal pool size $k^* \in \{\ell, \ell+1, \dots, u-2\}$ is so that

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

So k^* can be found iteratively by choosing the smallest $k \in \{\ell, \ell+1, \dots, u-2\}$ that satisfies the underbraced expression (*). Then, the behavior $(\mathbf{p}^*, \boldsymbol{\tau}^*)$ 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 $j \in \{1, 2, \dots, n\}$ will be optimal. So the optimal behavior can also be found with a search through no more than $n + 1$ candidates.

A.3. Generalized patch algorithm

The algorithms in Appendices A.1 and A.2 cannot be used with Equation (3) in the patch case because the function $d_i(\tau_i) \triangleq \tau_i$ for each type $i \in \{1, 2, \dots, n\}$ (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 $j, k \in \{1, 2, \dots, n\}$:

- (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 $\tau_j \in (\tau_j^-, \tau_j^+)$, $d_j(\tau_j) = 0$ 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 non-zero 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 $i \in \{1, 2, \dots, n\}$. So to guarantee that the objective function is well defined, also assume that:

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

By the assumptions, for each type $i \in \{1, 2, \dots, n\}$, 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

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

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 $\ell \in \{0, 1, \dots, n-1\}$ where

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

and

$$\begin{aligned} \infty &> \frac{a_{\ell+1}(\tau_{\ell+1}^-)}{d_{\ell+1}(\tau_{\ell+1}^-)} > \frac{a_{\ell+2}(\tau_{\ell+2}^-)}{d_{\ell+2}(\tau_{\ell+2}^-)} > \dots > \frac{a_{n-1}(\tau_{n-1}^-)}{d_{n-1}(\tau_{n-1}^-)} \\ &> \frac{a_n(\tau_n^-)}{d_n(\tau_n^-)}. \end{aligned}$$

Next, for each type $j \in \{1, 2, \dots, n\}$ and any $k \in \{0, 1, \dots, n\}$, 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

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

where p_i^k is defined by Equation (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 (ii). Again, to guarantee strict convexity of the objective function at the optimal behavior, assume that:

- (vii) For any $k \in \{0, \dots, n-1\}$,

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

Finally, define optimal pool size $k^* \in \{0, \dots, n\}$ so that

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

So k^* can be found iteratively choosing the smallest $k \in \{0, \dots, n-1\}$ that satisfies the underbraced expression (*). At each iteration, the processing times are chosen using the generalized marginal value theorem. Then, the behavior $(\mathbf{p}^*, \boldsymbol{\tau}^*)$ 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 $j \in \{1, 2, \dots, n\}$ 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.