Online Robot Task Switching Under Diminishing Returns

Jens Wawerla, and Richard T. Vaughan

Simon Fraser University, Burnaby, BC, Canada
{jwawerla, vaughan}@sfu.ca

Abstract

We investigate the task switching problem of a robot maximizing its long-term average rate of return on work performed. We propose an online method to maximize the average gain rate based on only past experience. For that we alter the formulation from optimal foraging theory and recursively include estimates of global task qualities. We demonstrate and analyze our method on a puck-foraging example. In simulation experiments under a variety of conditions we show that our method performs well compared to results obtained by brute force method using post-processed foraging data.

Introduction

Many robot applications require a robot to make task switching decisions in order to maximize its reward. Often this reward is a diminishing function of the time spent performing the task. These diminishing returns can either be caused by (i) exhausting a given task, for example having delivered all mail in a given building or by (ii) increasing difficulty to perform the task, e.g. it will be more and more difficult for a vacuum cleaning robot[1] to remove dirt as it cleans the floor. In fact it will be virtually impossible for a vacuum cleaning robot to remove all dirt particles and thus this task has no well defined intrinsic end point.

In both situations the robot has to decide when it is profitable to terminate the current task, pay a switching cost, and start a new task that yields higher rewards. The switching cost can come in form of an opportunity cost or an actual cost such as energy expenditure, transit toll or task acquisition cost. In other words the robot has to decide when to switch tasks in order maximize its long-term average reward rate. This decision depends on a number of factors: how good is the current task, how high is the switching cost and what is the average payoff function for tasks in the robot's environment?

In an earlier paper (Wawerla and Vaughan, 2009) we proposed a task switching policy based on the Marginal-Value Theorem (MVT) (see Sec. Marginal-Value Theorem). This policy required the robot to perform exploration steps in order to evaluate the average quality of the available tasks. We showed that the performance of the proposed policy was about 80% of that obtained by a near optimal policy discovered by brute force search.

In this paper we propose a recursive task switching policy based on locally available information only, hence no explicit exploration phase and thus no exploration/exploitation trade-off is required.

The policy is applicable to other task switching situations that exhibit diminishing returns. We choose foraging as an example task, since it is a canonical task in autonomous robotics (Cao et al., 1997). Robot foraging often means multi-agent *central place foraging* (Stephens et al., 2007), where foraged items are delivered to single privileged location. In contrast in this paper and our previous work (Wawerla and Vaughan, 2009) we use solitary, instant-consumption foraging in a patchy environment: a single robot immediately consumes items once they are encountered obtaining a reward without the need to deliver them to a centralized location. Items to be foraged are not distributed uniformly, but in patches defined for Behavioural Ecology as "*an homogeneous resource containing area separated from others by areas containing little or not resources*" (Danchin et al., 2008).

Marginal-Value Theorem

In behavioural ecology the task switching problem is often discussed in terms of optimal foraging theory (Stephens and Krebs, 1986) as a patch leaving decision. In this context patches are subject to diminishing returns and thus require the forager to make decisions about changing patches. In this case the task switching cost the inter-patch travel cost. An important result of optimal foraging theory is the Marginal-Value Theorem (MVT). Charnov and Orians (1973); Charnov (1976) proposed the MVT to model foraging decisions made by animals. His key result is the following patch leaving rule: "*when the intake rate in any patch drops to the average rate for the habitat, the animal should move on to another patch*" (Charnov and Orians, 1973). As

[1] We assume the robot gets rewarded for the amount of dirt collected and not for time spent vacuuming.

a consequence an optimal forager should exploit patches for a longer time as the inter-patch travel time increases and for a shorter time as the entire environment becomes more profitable. The simplicity of this rule makes it very appealing as a task-switching rule for robots, but the theorem and its validity has been widely and controversially discussed, for example by Green (1984); McNamara (1982); Stephens and Krebs (1986). Some of these issues make an implementation of the MVT as a robot task switching policy impossible. The main problems are:

- How to measure the marginal gain rate (the derivative of the gain rate) if the reward comes in discrete lumps. Andrews et al. (2007) suggest calculating the slope of the gain function between the last gain function change and the one two changes prior. In our tests (not shown) this method proved ineffective due to the stochastic nature of puck encounter during random foraging in patches with randomly placed pucks. In previous work (Wawerla and Vaughan, 2009) we used the expected value of a beta distribution over time-steps in which the robot found a puck and those in which it did not, as a proxy for the instantaneous rate. While we were able to build a task switching policy around this estimated gain rate, it is not the instantaneous gain rate. Thus leaving a patch once this estimated gain rate equals the long-term average rate does not maximize the long-term gain rate.

- The true long-term average gain rate for a given environment is usually unknown to the forager: all it can know is the average gain rate it experiences. This experience is a result of the foragers behaviour, yet the MVT requires the forager to base it's patch leaving decision on the obtainable long-term average gain rate. This circular dependency necessitates that the forager explores the action space in order to find the maximum long-term average gain rate. Previously (Wawerla and Vaughan, 2009) we used this circular dependency and turned the foraging task into a multi-armed bandit problem and applied standard ϵ-greedy methods (Sutton and Barto, 1998) to tackle the exploration-exploitation trade-off.

Stephens and Krebs (1986) summarize these problems as "*The MVT survives not as a rule for foragers to implement, but as a technique that finds the rate-maximizing rule from a known set of rules*". Since the MVT does not provide an implemetable policy, behavioural ecologists proposed other patch-leaving rules. (1) **number rule**, "leave after catching *n* items" (Gibb, 1958); (2) **fixed residence time rule** "leave after being in a patch for *t* time" (Krebs, 1973); (3) **give up time rule** "leave after *t* time has elapsed since the last encounter" (Krebs et al., 1974); (4) **rate rule** "leave when the instantaneous intake rate drops to a critical value *r*" (McNamara, 1982). Rules 1-3 have the advantage that the decision is based on values that are easily measurable by the forager.

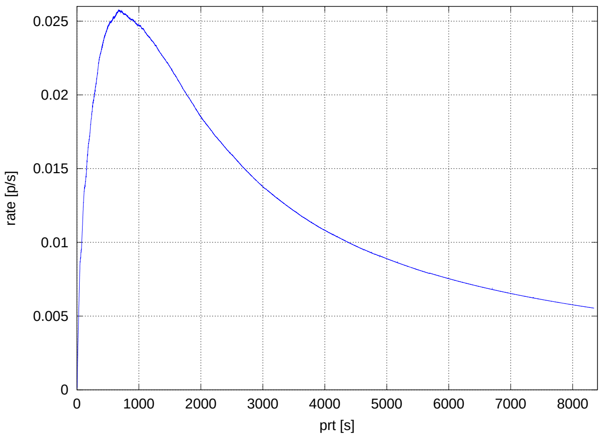


Figure 1: Average gain rate for a fixed patch residence time. Series of 100 patches with initially 50 pucks and a patch switching time of 500 seconds.

The rate rule is an extension of the MVT in that it copes with variance in patch sub-types, but it does not address the two issues mentioned above. None of these rules address the question of how to obtain the magic number on which the decision is based.

To illustrate the difficulty of this task-switching problem we conducted a brief simulation experiment. For this experiment we generated 100 constant size patches, each with initially 50 pucks. Next we had the robot forage in each patch until it was completely exhausted. For each time step we recorded the number of pucks gained from the current patch. From the recorded data we then calculated the average long-term gain rate as a function of patch residence time. In other words we forced the robot to leave each patch in a 100 patch series after a fixed time. By sweeping over patch residence times from 10 to 8000 seconds we obtained Fig. 1. This graph shows the long-term gain rate for a given patch residence time for this particular patch configuration and switching cost. The curve is interesting because it shows how large an error (i.e. reduction on average reward gain rate) a task-switching robot can make if switching too early or too late. It is worth pointing out that a robot is not actually able to measure this curve and exploit a patch optimally at the same time. Fortunately the robot only needs to find the maximum of the long-term gain rate and not determine the function per se.

Having described the optimization problem, in the following we present a new online adaptive solution that is grounded in the robot's perception and achieves foraging results comparable to an idealized forager that bases its decisions on global, unknowable environmental averages.

Marginal Gain Rate Task Switching

To derive the MVT Charnov (1976) argued that an optimal forager should maximize

$$R = \frac{\sum \lambda_j \cdot g_j(t_j) - \tau \cdot E}{\tau + \sum \lambda_j \cdot t_j} \tag{1}$$

where λ_j is the proportion of visited patches that are of type j, $g_j(t_j)$ is the net gain function for a patch of type j, τ is the average inter-patch travel time, E the rate of energy expended while switching patches and t_j is the time spent in a patch of type j. The objective of a forager is to select all patch residence times t_j such that R is maximized.

Without loss of generality we ignore the energetic cost of travel $\tau \cdot E$, since it is independent of the decision variables, so Eq. 1 reduces to

$$R = \frac{\sum \lambda_j \cdot g_j(t_j)}{\tau + \sum \lambda_i \cdot T_j} \tag{2}$$

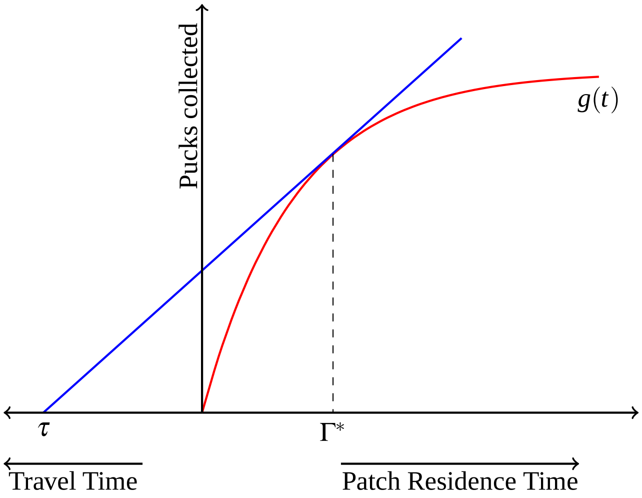


Figure 2: Typical MVT plot with two quantities on the abscissa: travel time increasing to the left, and patch residence time increasing to the right. The optimal patch residence time Γ^* is found by constructing a tangent to the gain function $g(t)$ that begins at the patch switching time τ on the travel time axis.

Charnov showed that R is maximized if $\frac{\partial g_j(t_j)}{\partial t_j} = R$. Graphically this is easy to do. As Fig. 2 shows, the optimal patch residence time T_j is found by constructing a tangent to the gain function that begins at the patch switching time τ on the travel time axis (see Stephens and Krebs (1986) for details).

The gain function $g(t)$ depends on (i) the actual patch quality, which varies from patch type to patch type but can also be variable within a patch type, for example if the pucks are placed randomly and (ii) on the robot environment interaction, e.g. sensor range, search strategy, motor control etc. Thus foraging in two equally sized patches, initially containing the same number of pucks, that is patches with the same puck density, may result in two totally different gain functions and there is no way a forager can predict the gain function of a particular patch before entering the patch. Fig. 3 shows two exemplar gain functions and the average gain function over 100 patches (each patch with initially 50 pucks). Thus as McNamara (1982) argues, the subpatch type variance has to be considered. This immediately raises the question how does the forager determine the type of patch in which she is currently foraging ? In some scenarios the patch type might be detectable by an external cue, but in general it is not and the forager is required to forage in the patch in order to obtain information about the patch. This adds a patch discrimination problem to the decision process.

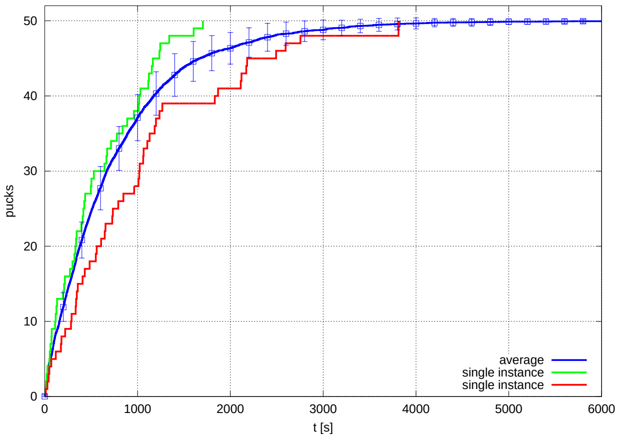


Figure 3: Average gain function (thin line) for random foraging in a 50 puck patch, error bars depict the standard deviation. Two instance of the gain function (thick lines) for patches with the same initial number of pucks.

To overcome these issues, we suggest dropping the notion of patch types and treating each patch as its own type. (In the following we still use the phrase “patch type” to mean patches with the same initial number of pucks (same puck density), but we do not perform any form of rate maximization based on the notion of patch types.) For unique patches the long-term average gain rate is

$$R = \frac{\frac{1}{n}\sum_i^n g_i(t_i)}{\tau + \frac{1}{n}\sum_i^n t_i} \tag{3}$$

We replaced the patch type index j with index i referring to unique patches. The advantage of not having to distinguish patch types and not having to deal with patch subtype variance comes at the disadvantage of having a possibly very large planning horizon of n timesteps. In fact the planning horizon is the lifetime of the robot. Since the robot cannot predict the future, we avoid the large planning horizon by recursively maximizing Eq. 3 based on only past experiences and ignoring possible future changes. Then our approximation of the long-term average gain rate while foraging in patch i, based on observations from previously encountered patches $0..i-1$ is

$$\tilde{R}_i = \frac{g_i(t_i) + G_i}{t_i + \tau + T_i} \tag{4}$$

Where G_i is the sum of collected pucks and T_i the total time (patch residence plus travel time) from all previous patches $0..i-1$. G_0 and T_0 can be used as a prior that provides the robot with an initial estimate of the average patch quality. Both G_i and T_i are a simple model of the average patch quality of the environment. This information (except the prior) is gained by the forager during exploitation. Hence a forager encountering only one patch type will actually maximize Eq. 2. But a forager first encountering a series of only low quality patches and then a series of high quality patches will maximize a very different average gain rate function than an omniscient forager. But an uninformed forager maximizing Eq. 4, will do as well as possible given the limited available information.

Robot Controller

The core of our task switching method is to maximize Eq. 4. This is done by numerically estimating the derivative of R_i at every time step and leaving the patch once the the derivative becomes zero. Since the gain function is assumed to be negatively accelerated, a maximum is found this way.

Algorithm 1 summarizes our task switching method. The robot forages for one time-step, if it collected a puck the local gain function $g(t)$ is incremented (line 10-15). Next we calculate an approximation of the long-term gain rate based on the experience from previous patches (G_i, T_i), an estimate of the travel time $\tilde{\tau}$ and the value of local gain function at the current time. Because of the stochastic and noisy nature of the gain function the estimate of the long-term gain rate has to be smoothed. In our implementation we use a low-pass filter (line 17-21). Other methods maybe substituted, however it performs well enough for our purpose. As mentioned earlier the patch leaving decision is based on checking if the derivative of the long-term gain rate is equal to zero. Again because of the stochasticity of the gain function we might experience a local region of zero or negative gradient, which could be interpreted as a local maximum. A simple counting step helps to overcome those undesired local maxima (line 22-27). As with the low-pass filter, any suitable method may substituted. The actual patch leaving decision is made in line 27. A patch is left once a maximum is found and a minimum amount of time has been spent in the patch. This minimum patch residence time is helpful during the initial time in a patch, since until the first puck is found $g(t) = 0$ would cause the robot to leave the patch immediately.

Once the robot leaves the patch it travels to the next patch. This travel takes τ_i time. Before starting to forage in the new patch the estimates for the environment quality G and T and the estimate of the switching time $\tilde{\tau}$ are updated (line 29-30).

```
1  **Algorithm:**patchMax
2  **init** $G_0$, $T_0$, $\tilde{\tau}$, $k_1$, $k_2$, $k_3$, $k_4$
3  $i = 1$
4  **forall** *patches* **do**
5      enter patch $i$
6      $t = 0$
7      $g(0) = 0$
8      **repeat**
9          $t = t + 1$
10         randomly forage for one time-step
11         **if** *puck collected* **then**
12             $g(t) = g(t-1) + 1$
13         **else**
14             $g(t) = g(t-1)$
15         **end**
16         $r(t) = \frac{g(t)+G_i}{t+\tilde{\tau}+T_i}$
17         **if** $t == 1$ **then**
18             $r_{filt}(t) = r(t)$
19         **else**
20             $r_{filt}(t) = (1-k_3)\, r_{filt}(t-1) + k_3 r(t)$
21         **end**
22         **if** $r_{filt}(t) - r_{filt}(t-1) \leq 0$ **then**
23             $c = c + 1$
24         **else**
25             $c = 0$
26         **end**
27     **until** $c > k_1$ *and* $t > k_2$
28     move to next patch in $\tau_i$ time
29     $G_{i+1} = G_i + g(t)$
30     $T_{i+1} = T_i + t + \tau_i$
31     $\tilde{\tau} = \tilde{\tau} + k_4(\tau_i - \tilde{\tau})$
32     $i = i + 1$
33 **end**
```

Algorithm 1: Task switching algorithm

Experiments

To investigate the effectiveness of our approach, we conducted a series of simulation experiments consisting of two phases (i) generate foraging data and (ii) test our task (patch) switching policy on the generated data (see Sec. Experimental Data). To generate the foraging data we used a generic mobile robot model in the well known simulator Stage (Vaughan, 2008). The robot is equipped with a short-range colour blob tracker to sense ‘pucks’, our unit of resources, in its vicinity. The robot knows (or equivalently can detect) the boundaries of a puck patch. Patches are 620 times the size of the robot, and contain initially 10, 30, 50, 100, 200 or 300 pucks placed uniformly at random. A minimum distance between pucks is enforced to avoid overlap. To exploit a patch, the robot randomly forages for pucks, by driving straight until it comes to the patch boundary, where it chooses a new heading that brings it back into the patch, at random. When a puck is detected, the robot servos towards the closest puck and collects it. Collecting a puck takes one simulation time step, so there is virtually no handling time. At each simulation time step we record how many pucks the robot has collected so far in the current patch: this is the gain function.

As mentioned earlier the gain function is not only dependent on the initial number of pucks per patch but also on the robot/environment interaction. To get a good sample of the distribution of gain functions, we randomly generate 100 patches of each of the six patch types and record the gain functions from the robot foraging in those patches. Note that at this point in the experiment no patch leaving decisions are made. The robot simply forages until the patch is exhausted and the simulation is terminated. Testing our approach on this recorded data set rather than during the robot simulation allows us to compare approaches on exactly the same data and it makes it feasible to determine a near-optimal solution by brute force solution search.

As a baseline for comparison we need to find a t_i for each patch such that the long-term gain rate is maximized. No closed form solution is known to this problem, and the gain functions are available as data points only. So we employ a brute force search. Since each patch is unique this technically requires us to solve Eq. 3 for all possible combinations of patch residence times. Because this is computationally prohibitive we resort to calculating the average gain function over all 100 instances of a patch type. Then we find the best patch residence time by solving Eq. 3 for all possible t $(0 \leq t \leq T_{patch\_exhaused})$ and selecting the t that maximizes the average gain rate. In case of multiple patch types we calculate the long-term gain rate for each combination of residence times on the average gain function. This is only feasible since the number of patch types considered is small.

In all of the following experiments we used the obtained long-term average gain rate as a metric for comparison. All algorithm parameters required were set manually and kept constant without any attempt to optimize them. The priors G_0 and T_0 were set to zero. To investigate our task switching method under a wide range of conditions we altered the task (patch) switching time τ from very short 10 seconds to very long 5×10^6 seconds ($\approx$6 days). To put this in perspective we report the mean and standard deviation of observed times required to exhaustively forage patches in Table 1. The spectrum reaches from almost no switching cost to a switching cost about 200 times the average time required to exhaust a patch.

	initial pucks per patch					
	10	30	50	100	200	300
μ [s]	1858	2909	3631	4556	5171	5475
σ	825	1184	1271	1337	1206	1208

Table 1: Mean and standard deviation of the time required to exhaustively forage patches

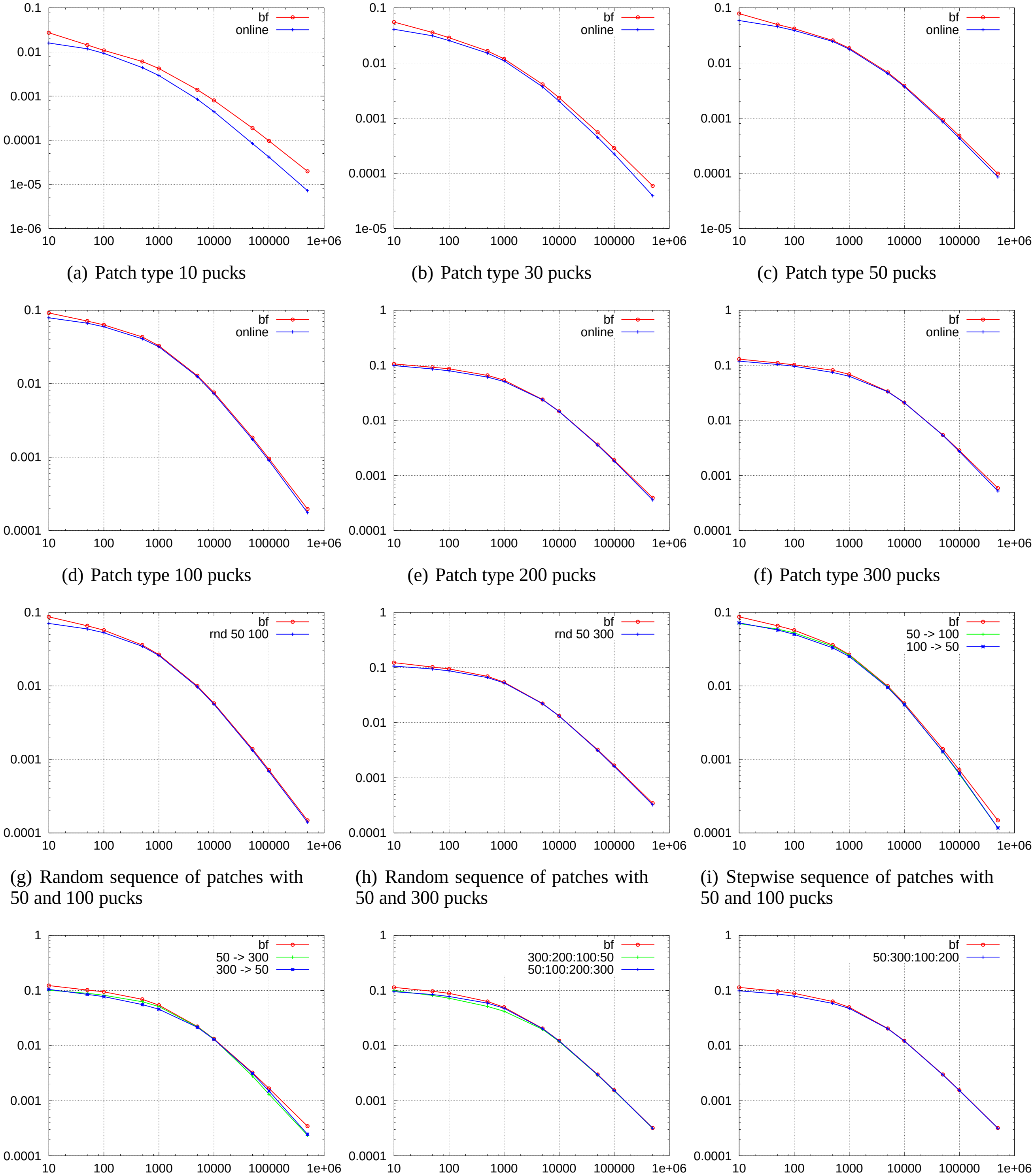
Single Patch Type

In a first experiment we had the robot forage in a series of 100 patches with the same initial number of pucks. Figure 4(a)-(f) shows the achieved long-term average gain rate for each patch type over a variety of switching times compared to the brute force solution. From the graphs we can draw three conclusions. (i) If the task switching times are short (i.e. much lower than the patch residence times) the performance of our method is in general lower than that of the near-optimal brute force method. The MVT predicts short patch residence times in situations where patch switching is cheap. But because of the various filters (filter parameters kept constant for all experiments over all conditions) our method's responsiveness is too slow in these short residence time situations. We say the performance is lower, but it is still above 78% (except in the 10 puck patches, where the performance drops to 50%). (ii) Under low patch quality situations (10 pucks, 30 pucks) our method performs less well than the brute force method. Again the reason is in the choice of parameters. The filters are too slow for the optimal, short patch residence time. (iii) The method described in this paper achieves similar long-term rates as the brute force method in all other cases examined. Recall that it uses only locally obtained information, in contrast to the omniscient brute force method.

Multiple Patch Types

A more challenging problem is the case where patches of very different quality are encountered. As the MVT predicts the patch leaving decision is not only dependent on the quality of a given patch but on the global quality. To illustrate the difficulty of this decision we give a brief example. Let t_h be the optimal patch residence time if a forager only encounters patches of a fixed, high quality. If the same forager now encounters a mixture of high and low quality patches, t_h is no longer the optimal patch residence time for the high quality patches. The reason is that the cost of lost opportunity has increased due to the patches of low quality. As a consequence the forager should increase t_h under these circumstances.

To investigate our system under these conditions we conducted a series of experiments. In a first experiment we had the robot encounter 100 patches of type A and 100 patches of type B in a random order. Figure 4(g) and 4(h) show the averaged results over 20 trials for patch configurations 50:100



(a) Patch type 10 pucks

(b) Patch type 30 pucks

(c) Patch type 50 pucks

(d) Patch type 100 pucks

(e) Patch type 200 pucks

(f) Patch type 300 pucks

(g) Random sequence of patches with 50 and 100 pucks

(h) Random sequence of patches with 50 and 300 pucks

(i) Stepwise sequence of patches with 50 and 100 pucks

(j) Stepwise sequence of patches with 50 and 300 pucks

(k) Stepwise sequence of patches with 50, 100, 200 and 300 pucks

(l) Stepwise sequence of patches with 50, 300, 100 and 200 pucks

Figure 4: Long-term average gain rates achived by the bruteforce method (red line with circle) and our online method (green line with cross, blue with asterix). Inter patch travel time τ in seconds on the x-axis and long-term gain rate in pucks per seconds on the y-axis. More details in the text.

pucks and 50:300 pucks respectively. Errorbars were omitted because of the small standard deviation. As in the single patch type experiments and for the same reasons, the performance is somewhat lower under short switching time conditions, but in the general the graphs show that our method copes well with randomly encountered patches of different qualities.

An even harder problem is to encounter a longer series of patches of type A followed by a series of patches of type B, where the forager does not know anything about type B patches while it forages in type A patches. On encountering type B patches, the robot has built a strong prior expecting type A patches. In this experiment the robot was faced with a series of 100 patch of one type followed by 100 patch of a different type. The results for 50:100 and 50:300 patches with a stepwise change in both directions is shown in Fig. 4(i) and 4(j) respectively. Here the brute force method is at a significant advantage because the patch leaving decisions are derived with full knowledge of the future patch change. Our method does not/can not anticipate the patch quality change and thus for the first 100 patches acts under the “assumption” of a constant environment. The error resulting from this “assumption” grows with the difference in patch qualities. That is why the performance difference in the 50:300 scenario (Fig. 4(j)) is larger than in the 50:100 case (Fig. 4(i)).

Figure 4(k) shows the results for a stepwise sequence of 50:100:200:300 puck patches and the reverse ordering. The results are qualitatively very similar to those discussed previously. In one last experiment of this type we choose step wise patch encounter with larger step sizes. The ordering chosen was 50:300:100:200. Results are shown in Fig. 4(l). The performance results are again qualitatively similar, suggesting the our method handles this type of variance well.

Variable Switching Cost

So far we tested different switching costs but kept them constant in the single patch type as well as multi patch type experiments. To investigate varying inter-patch travel time, we conducted an experiment in which the travel time between patches was drawn from a normal distribution with mean 1000 seconds and standard deviation 100, 500 and 700 seconds respectively. Table 2 shows the results in percent compared to the long-term gain rate of the brute force solutions. Because of the computational complexity the brute force solution was only calculated using the mean and not the actual randomly drawn travel times. As in the previous experiments we see generally good performance and the usual drop in situations with low patch quality.

Discussion

Task switching under diminishing returns is daily routine for many animals and important for many conceivable autonomous robots. Maximizing the long-term average gain or reward rate under these conditions requires the robot to have knowledge of future gain functions. This is not achievable by a robot relying solely on information obtained by its own actions. To the best of our knowledge no solution to this problem is known. In this paper we have argued that the MVT is not implementable because an instantaneous gain rate is meaningless in the case of rewards obtained in chunks. It also requires a continuous exploration phase in order to find the global maximum rate, but the MVT itself does not explore the action space.

	initial pucks per patch					
σ	10	30	50	100	200	300
100	74.0	92.2	96.0	96.4	95.3	92.2
500	76.3	90.2	93.9	94.5	89.7	92.3
700	67.8	89.5	96.9	92.6	88.7	90.1

Table 2: Percent performance for variable patch switching time with mean 1000 sec. and standard deviation $\sigma = \{100, 500, 700\}$

Instead we proposed a task switching method that bases its decision only on previously obtained information, well aware that we therefore maximize a different function. Thus we may make suboptimal task switching decisions, but these decisions are as good as possible given no information about the future.

An important issue to discuss is how large the time window of past experiences should be, that are considered in the task-switching decision. In this paper we simply included all past foraging experiences when modelling the global patch quality. This is reasonable as long as the past is a good predictor for the future. On the other hand in situations where the future strongly deviates from the past, forgetting or a short memory can be beneficial. The memory size is also interesting from a behavioural ecology point of view, because it might explain why animals often appear to maximize the short-term and not the long-term intake rate (Real et al., 1990). In future it would be interesting to investigate what influence the memory size has on the rate maximization of a robot and what the optimal size is.

We draw a lot of insight from behavioural ecology, but we make no claims about mechanisms employed by animals.

Acknowledgements

The authors thank Yaroslav Litus for very valuable discussions on the work that led to this paper. This work was supported by NSERC and DRDC in Canada.

Experimental Data

In accordance with the Autonomy Lab’s policy on code publication, the foraging data and the implementation of the experiments are made available online at `git://github.com/jwawerla/tsw_` `experiment.git`. The exact data that led to the presented results can be accessed via the commit hash `4f84a82d09f2c181df57ab5d7faa2e53cc3348f3`.

References

Andrews, B. W., Passino, K. M., and Waite, T. A. (2007). Foraging theory for autonomous vehicle decision-making system design. *Journal of Intelligent and Robotic Systems*, 49:39–65.

Cao, Y. U., Fukunaga, A. S., and Kahng, A. B. (1997). Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:226–234.

Charnov, E. L. (1976). Optimal foraging: Attack strategy of a mantid. *The American Naturalist*, 110:141–151.

Charnov, E. L. and Orians, G. H. (1973). Optimal foraging: Some theoretical explorations. Unpublished manuscript `http://hdl.handle.net/1928/1649`.

Danchin, É., Giraldeau, L.-A., and Cézilly, F., editors (2008). *Behavioural Ecology*. Oxford University Press.

Gibb, J. A. (1958). Predation by tits and squirrels on the eucosmid *Ernarmonia conicolana*. *Anim. Ecol.*, 27:375–396.

Green, R. F. (1984). Stopping rules for optimal foragers. *The American Naturalist*, 123(1):30–43.

Krebs, J., Ryan, J., and Charnov, E. (1974). Hunting by expectation or optimal foraging: A study of patch use by chickadees. *Animal Behaviour*, 22:953–964.

Krebs, J. R. (1973). *Perspectives in Ethology*, volume 1, chapter Behavioral aspects of predation, pages 73–111. Plenum New York.

McNamara, J. M. (1982). Optimal patch use in a stochastic environment. *Theoretical Population Biology*, 21(2):269–288.

Real, L., Ellner, S., and Harder, L. D. (1990). Short-term energy maximization and risk-aversion in bumble bees: A reply to Possingham et al. *Ecology*, 71(4):1625–1628.

Stephens, D. W., Brown, J. S., and Ydenberg, R. C., editors (2007). *Foraging - Behavior and Ecology*. University of Chicago Press.

Stephens, D. W. and Krebs, J. R. (1986). *Foraging Theory*. Princeton University Press.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: an introduction*. MIT Press.

Vaughan, R. T. (2008). Massively multi-robot simulations in Stage. *Swarm Intelligence*, 2(2-4):189–208.

Wawerla, J. and Vaughan, R. T. (2009). Robot task switching under diminishing returns. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5033–5038.