Dynamic Resource Allocation for Optimizing Population Diffusion

Shan Xue
xue@eecs.oregonstate.edu
School of EECS
Oregon State University

Alan Fern
afern@eecs.oregonstate.edu
School of EECS
Oregon State University

Daniel Sheldon
sheldon@cs.umass.edu
University of Massachusetts Amherst
Mount Holyoke College

Abstract

This paper addresses adaptive conservation planning, where the objective is to maximize the population spread of a species by allocating limited resources over time to conserve land parcels. This problem is characterized by having highly stochastic exogenous events (population spread), a large action branching factor (number of allocation options) and state space, and the need to reason about numeric resources. Together these characteristics render most existing AI planning techniques ineffective. The main contribution of this paper is to design and evaluate an online planner for this problem based on Hindsight Optimization (HOP), a technique that has shown promise in other stochastic planning problems. Unfortunately, standard implementations of HOP scale linearly with the number of actions in a domain, which is not feasible for conservation problems such as ours. Thus, we develop a new approach for computing HOP policies based on mixed-integer programming and dual decomposition. Our experiments on synthetic and real-world scenarios show that this approach is effective and scalable compared to existing alternatives.

1 INTRODUCTION

This paper addresses adaptive conservation planning, where the goal is to maximize the population growth of a species by purchasing and reserving land parcels to best support population spread. This optimization problem is challenging as it requires reasoning about the stochastic population spread on a large network, uncertain future budgets, and a combinatorial action space (the set of possible investment combinations).

Generic off-the-shelf, model-based planners typically are unable to compactly encode such problems (e.g. due to numeric budgets and/or exogenous events) and scalability is also quite limited. Other simulation-based approaches such as Monte-Carlo tree search have difficulty due to the large branching factors and horizons. In this paper, we develop an online planning algorithm for the adaptive conservation problem, which requires deciding, at each decision epoch, the set of parcels to purchase in order to maximize the long-term population growth. While this work focuses on a particular population diffusion process, it is important to note that the principles are applicable to a wider class of diffusion-control problems that pose similar challenges to existing methods.

Prior work has considered simplified versions of the above problem. Sheldon et al. [2010] studies the non-adaptive, upfront planning problem, where it is assumed that the budget is known and purchases are made only at the first time step. This simplification ignores the reality that budgets often arrives over time and that delaying some allocation decisions until more information is available can be beneficial. In followup work Xue et al. [2012] studies a middle ground between upfront and fully adaptive planning. Their algorithm schedules the purchases of an upfront solution over time in order to get a (non-adaptive) multi-stage conservation plan with maximum flexibility. An approach for two-stage adaptive conservation plans is given by Ahmadizadeh et al. [2010], but scalability to more stages poses many challenges. The only prior work on fully adaptive conservation planning is by Golovin et al. [2011]. They make a crucial assumption that the species does not spread across land parcels and then replan at each time step using a greedy, myopic approach. While approximation bounds are shown under the assumption, the myopic nature of the approach can

Appearing in Proceedings of the 17^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33.

make it short-sighted when populations can spread, such as in our motivating application involving birds. Indeed our experiments confirm this shortsightedness is apparent on realistic problems.

Above we have seen several non-adaptive approaches that reason about population dynamics and an adaptive approach that assumes populations do not spread. Here we fill the research gap in conservation planning with an adaptive approach for highly spreading species. We take both the future population dynamics, the future budget, and the future actions into account. Our approach is based on hindsight optimization (HOP), an online planning approach that has been successfully applied to a variety of difficult stochastic planning problems, e.g. Chang et al. [2011], Chong et al. [2000], Wu et al. [2002], Yoon et al. [2010], Hubbe et al. [2012]. HOP reasons about possible stochastic futures to compute an upper bound on action values and selects the action that has the maximum upper bound. Unfortunately, standard implementations of HOP scale linearly in the number of actions available at any time, which is prohibitive for our exponential action space. Thus, our main contribution is to develop an efficient algorithm for computing HOP policies for such exponentially large, factored action spaces. We accomplish this by representing the HOP policy via a large Mixed Integer Program (MIP) and then applying the Dual Decomposition schema to make its solution more practical. Our experiments show that HOP can significantly outperform more myopic alternatives while also showing scalability to large problems.

2 Problem Setup

Population Model. Typically, a conservation problem is associated with a land map that is divided into many *habitat patches*, which at any time can either be occupied by the species or not. For management convenience, multiple nearby patches are grouped into *land parcels* and assign each parcel p a cost $c(l)$ that can be seen as the expense to conserve all the patches in p. We assume only patches in purchased parcels can be occupied as they are conserved to be accessible and suitable for the species.

The population dynamics are such that at any time step, any patch v becomes occupied via a population spread from patch u with a probability of p_{uv} (here $1-p_{vv}$ is the extinction probability for patch v). Here we use a standard spread model p_{uv} where the spread probability decreases with the distance between u and v. In addition, there is a stochastic budget process, where new funds arrive each year according to some distribution. Parcels can only be purchased when enough funds are available and hence must be purchased incrementally. Since the population can only spread to patches in conserved parcels, the population diffusion is strongly influenced by parcel purchases.

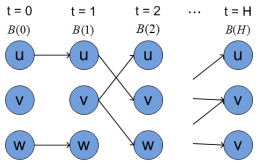

Figure 1: Visualization of An Example Future. Suppose there are totally 3 patches: u, v, and w, then each patch repeats at each time step. The edge indicates the colonization from one patch to the other at a certain time step.

Adaptive Planning Problem. We consider a finite-horizon setting, where the goal is to optimize the total population across patches at the specified horizon H. Decision epochs occur every T_r years and at each a decision is made about the set of parcels to purchase, limited by the current budget. Thus, our planning problem is to produce a policy π that is given the current problem state at a decision epoch and outputs a set of parcels to purchase with cost no more than the current budget $B(t)$. Here the problem state is composed of the time-to-horizon, the species occupancy at each patch, the current budget, and knowledge of previous purchases. We follow a model-based approach, where π can be computed using provided stochastic models of the population dynamics and budget.

It will be useful to define the notion of a future F, which is a random variable encoding the random outcomes that dictate the future population spread and budget over the horizon H. In our setting, a realization f of F deterministically dictates whether a population can spread between two patches at any particular time and the budget at each future year. Such a future can be visualized as a deterministic network as in Figure 1. In the network, all the patches repeat at every time step and an edge between patch u at time t and v at time $t+1$ indicates that if the species is at u at time t then it will spread to v at time $t+1$.

For a fixed realization f, we effectively get a deterministic problem, for which any policy can be evaluated against. We say a policy π is feasible for a future f *iff* at any time step the total cost of parcel purchases of π is within the budget limit of f. For a future f, a feasible policy π, and initial state s_0 we denote the total population/reward achieved at the horizon H by $R(s_0, f, \pi)$. The optimal solution to our problem can then be expressed as finding a policy π^* that maximizes the expected reward, i.e.

$\pi^* = \arg\max_\pi \mathbf{E}[R(s_0, F, \pi)]$.

An exact solution for π^* is beyond the capabilities of existing planners. To address this, prior work Golovin et al. [2011] used a simple policy based on myopic, greedy action selection. The main contribution was to give approximation bounds for the policy under strong assumption of no diffusion across parcels. Rather, here we develop a non-myopic action selection heuristic, hindsight optimization, that has shown success in various other areas of planning. While hindsight optimization also requires strong assumption to provide performance guarantees, our experiments demonstrate that its non-myopic nature can lead to significant improvements over Golovin et al. [2011].

3 Hindsight Optimization for Conservation Planning

Hindsight Optimization. The main idea behind hindsight optimization (HOP) is to drive action selection by computing upper bounds on the values of states. Given a state s, the hindsight value of the state $V_{\text{hs}}(s)$ is an optimistic estimate of the true value obtained by interchanging expectation and maximization, i.e. $V_{\text{hs}}(s) = \mathbf{E}\left[\max_\pi R(s, F, \pi)\right]$. This clearly gives an upper bound on the value since it allows for inconsistent policies to optimize each future.

The key computational advantage of working with V_{hs} rather than directly with the value function V is that it can be approximated by solving a set of deterministic planning problems for individual futures. That is, given a set of sampled futures $\{f_1, f_2, \ldots, f_N\}$ we estimate the hindsight value as:

$$\hat{V}_{\text{hs}}(s) = \sum_k \left[\max_\pi R(s, f_k, \pi)\right]$$

where the internal maximization problem for each f_k can often be solved with existing solvers. The hindsight Q-function $Q_{\text{hs}}(s, a)$ is accordingly defined as the expected hindsight value achieved for states reached by taking action a in state s and is also an upper bound on the true Q-value of a state-action pair. The HOP policy for a state s is then defined as $\arg\max_a Q_{\text{hs}}(s, a)$. Under certain assumptions performance guarantees can be made for the HOP policy (Mercier and Hentenryck [2007], Yoon et al. [2008]). However, in general, no guarantees can be made and examples of arbitrarily poor performance compared to optimal can be constructed. Fortunately such examples are often not reflective of real problems and the HOP policy is often an effective way to select action non-myopically.

The traditional way to compute the HOP policy is to estimate $Q_{\text{hs}}(s, a)$ by sampling a set of states resulting from taking a in s, computing the hindsight value for each state, and averaging. Unfortunately, this traditional approach scales linearly with the size of the action space and hence is not feasible for the combinatorial action space in our conservation problem and many others with factored actions.

HOP for Large Action Spaces. The traditional computation of the HOP policy estimates Q-values for each action for the purpose of maximizing over them. However, this is not strictly necessary if we can directly compute the optimizing HOP action at a state without explicitly estimating each Q-value. Thus, the main idea behind our approach is to encode the problem of computing the optimizing HOP action (Q-values) as a mixed integer program (MIP) and then apply decomposition techniques to solve it, which avoids the explicit enumeration over actions.

Our MIP for HOP is defined relative to a set of sampled futures $\{f_1, f_2, \ldots, f_N\}$ and a current state s. Similar to the formulation of Sheldon et al. [2010], for each future f_k, we can define a MIP, denoted as MIP_k, which encodes the problem of finding a policy that maximizes the reward of f_k. That is, MIP_k solves the problem $\max_{\pi_k} R(s, f_k, \pi_k)$, where π_k is viewed as a binary vector that specifies for each parcel and time, whether to buy the parcel at that time in future f_k. We will denote by π_k^1 the parcel purchases specified for the first time step in MIP_k. We also let OBJ_k and CON_k denote the objective and constraints of MIP_k respectively and let V_k be all variables in MIP_k excluding π_k. Due to space constraints we do not provide the full details of MIP_k, which is similar to that in Sheldon et al. [2010] and not crucial to our main contribution. Since we have separate decision variables for each future, the maximization and summation in the Q-value function can be interchanged, which results in a MIP that encodes the HOP policy:

HOP Policy MIP:

$$\min_{\{\pi_k, V_k\}} -\frac{1}{N}\sum_k \text{OBJ}_k, \text{ s.t.}$$

$$\pi_1^1 = \pi_2^1 = \ldots = \pi_N^1 \text{ and } \bigcup_k \text{CON}_k$$

That is, we can compute the HOP policy by maximizing the sum (minimizing the negative sum) of objectives across the futures, subject to the constraint that the solutions for each future agree on the first action. This MIP can then in concept be given to any MIP solver and then the returned value of π_1^1 can be returned as the HOP action.

While in our experience, it is generally feasible to use existing MIP solvers to solve the individual MIP_k problems for realistic scenarios, when the number of futures increases, solving the combined MIP can become prohibitive. This is an important limitation since the variance of the HOP policy reduces with more fu-

tures. Fortunately, this issue can be largely overcome via the use of dual decomposition techniques as the HOP policy MIP reveals a separable structure.

4 Dual Decomposition for HOP

In the HOP policy MIP, the individual MIP_k problems are only coupled via the policy constraints on the first action. If the constraints are removed then the combined MIP can be solved by solving each MIP_k independently. This structure motivates the application of Lagrangian dual decomposition, which we formulate below and follows a similar structure as prior work on upfront conservation planning by Kumar et al. [2012].

We start by rewriting the coupling constraint $\pi_1^1 = \pi_2^1 = \ldots = \pi_N^1$ of MIP_k as the set of constraints $\{\pi_k^1 = \mathbf{d} \;:\; k = 1, \ldots, N\}$ where $\mathbf{d}$ is a new vector of binary variables that represents the HOP policy action at the first time step. We let $\pi_{k,l}^1$ and d_l denote component l of π_k^1 and d, indicating whether l was purchased or not at time step 1. We can now relax these coupling constraints to get the Lagrangian of the HOP MIP by introducing Lagrangian multipliers $\lambda_{k,l}$ for each constraint.

$$L(\{V_k, \pi_k\}, \mathbf{d}, \boldsymbol{\lambda}) = -\frac{1}{N}\sum_{k=1}^{N} \text{OBJ}_k + \sum_{l,k} \lambda_{k,l}(\pi_{k,l}^1 - d_l)$$
$$\text{s.t.} \bigcup_k \text{CON}_k$$

The dual is then given by

$$\begin{aligned} q(\boldsymbol{\lambda}) &= \min_{\{\mathbf{V_k}, \pi_\mathbf{k}\}, \mathbf{d}} L(\{V_k, \pi_k\}, \mathbf{d}, \boldsymbol{\lambda}) \\ &= \min_{\{\mathbf{V_k}, \pi_\mathbf{k}\}, \mathbf{d}} -\frac{1}{N}\sum_{k=1}^{N} \text{OBJ}_k + \sum_{l,k} \lambda_{k,l}(\pi_{k,l}^1 - d_l) \\ &= \min_{\{\mathbf{V_k}, \pi_\mathbf{k}\}, \mathbf{d}} \sum_{k=1}^{N} -\frac{1}{N}\text{OBJ}_k + \sum_{l,k} \lambda_{k,l}\pi_{k,l}^1 \\ &\quad - \sum_l d_l \sum_k \lambda_{k,l}, \qquad \text{s.t.} \bigcup_k \text{CON}_k \end{aligned}$$

Intuitively, the relaxed constraints in the dual act as a penalty for violating the consistency requirement that all policies across futures agree on the first action. Since the dual minimizes over $\mathbf{d}$, in order to ensure that $q(\boldsymbol{\lambda}) > -\infty$ we require the constraint $\sum_{k=1}^{N} \lambda_{k,l} = 0, \forall l$. To simplify notation, we denote the space of Langrange multipliers that satisfy this constraint as:

$$\Lambda = \{\{\lambda_{k,l}\} \mid \sum_{k=1}^{N} \lambda_{k,l} = 0, \forall l\}$$

Under this constraint the last term in the dual vanishes and we finally get the dual which consists of independent subproblems for any fixed $\boldsymbol{\lambda}$:

$$q(\boldsymbol{\lambda}) = \min_{\{\mathbf{V_k}, \pi_\mathbf{k}\}} -\frac{1}{N}\sum_{k=1}^{N} \text{OBJ}_k + \sum_{l,k} \lambda_{k,l}\pi_{k,l}^1$$
$$\text{s.t.} \bigcup_k \text{CON}_k \text{ and } \{\lambda_{k,l}\} \in \Lambda$$

One important characteristic of the dual is that $q(\boldsymbol{\lambda})$ for any feasible $\boldsymbol{\lambda}$ is a lower bound on the optimal primal MIP objective value, which motivates attempting to make the bound as tight as possible by maximizing $q(\boldsymbol{\lambda})$ over $\boldsymbol{\lambda}$. Since $q(\boldsymbol{\lambda})$ is not continuous and the dual includes constraints over $\boldsymbol{\lambda}$ we use projected subgradient descent for this purpose, iterating as follows:

$$\boldsymbol{\lambda}_k^{(i+1)} = [\boldsymbol{\lambda}_k^{(i)} + \alpha_{i+1} g_k(\boldsymbol{\lambda}_k^{(i)})]_\Lambda \tag{1}$$

where i is the iteration number, $g_k(\cdot)$ is a subgradient of $q(\boldsymbol{\lambda})$ with respect to λ_k, α_i is the step size, and $[z]_\Lambda$ is the projection of z onto constraint space Λ.

For our objective, one subgradient of $q(\boldsymbol{\lambda})$ with respect to λ_k is $\bar{\pi}_k^1$ such that

$$\bar{\pi}_k = \arg\min_{\mathbf{V_k}, \pi_\mathbf{k}} -\frac{1}{N}\text{OBJ}_k + \sum_l \lambda_{k,l}\pi_{k,l}^1$$

which can be found by solving a minimization problem involving a single future f_k and hence is much more tractable than the full MIP. Note that this minimization problem is simply the original objective of MIP_k with an added term involving the current Langrange multiplier values that can be viewed as assigning a penalty or reward for purchasing particular parcels at the first time step. Finally, given the subgradient, the projection onto Λ (with Euclidean norm) is well known, requiring only that we subtract from each component of the subgradient the average component value. Letting $\bar{\pi}_k^{1,(i)}$ denote the subgradient at iteration i we get the following:

$$\boldsymbol{\lambda}_k^{(i+1)} = \boldsymbol{\lambda}_k^{(i)} + \alpha_{i+1}\left[\bar{\pi}_k^{1,(i)} - \frac{\sum_{k'=1}^{N} \bar{\pi}_{k'}^{1,(i)}}{N}\right] \tag{2}$$

This shows that the gradient steps for dual optimization can be computed by optimizing independent subproblems for each future (i.e. solving for each $\bar{\pi}_k$), which avoids solving a single MIP involving all futures. Putting everything together, the complete dual optimization algorithm is given in Algorithm 1.

Algorithm 1 Dual Decomposition Algorithm

1: **Given:** initial vector $\boldsymbol{\lambda} \in \Lambda$
2: **while** convergence is not reached **do**
3: *Optimize subproblems independently:*
solve each subproblem and get $\{\bar{\pi}_k^{1,(i)}\}$
4: *Compute average value of* $\bar{\pi}_k^{1,(i)}$ *over N subproblems:*
$\hat{\boldsymbol{d}}^{(i)} = \frac{\sum_{k'=1}^{N} \bar{\pi}_{k'}^{1,(i)}}{N}$
5: *Update* $\boldsymbol{\lambda}$*:*
$\boldsymbol{\lambda}_k^{(i+1)} = \boldsymbol{\lambda}_k^{(i)} + \alpha_{i+1}[\bar{\pi}_k^{1,(i)} - \hat{\boldsymbol{d}}^{(i)}]$
6: **end while**

At a high level, the final algorithm involves optimizing the dual via iterations. Each iteration involves solving multiple modified MIP$_i$ problems, which are different from the originals in that the costs of certain purchases at the first time step are modified in order to encourage the subproblems to agree on the first actions. Specifically costs are increased for parcels that are not currently purchased by most futures and decreased for parcels that are purchased by many futures. The iteration ends when either the subproblems all agree on the first action, in which case we get the optimal HOP action, or the maximum number of iterations is reached, in which case we extract a solution as described below.

Feasible Solution Extraction. The above algorithm optimizes the dual and does not explicitly provide a primal solution, which in our case is the action of the HOP policy. After optimizing the dual it is often the case that the solutions to the independent subproblems (i.e. the π^1_k) are consistent and hence represent a feasible primal solution. In these cases, any of the π^1_k are optimal primal solutions and we output the resulting action as the HOP action. However, in general there can be a duality gap and we are not guaranteed that optimizing the dual will produce feasible primal solutions. Thus, as is typical in Langrangian relaxation techniques we must define a strategy for heuristically selecting a feasible primal solution guided by the information obtained during the optimization.

Our approach is a heuristic based on the consistency requirement. As in the algorithm, we let $\hat{d}_l$ denote the average value of $\pi^1_{k,l}$ over different futures, which is 1 if parcel l is purchased in all futures and 0 if it is not purchases in any futures, and otherwise $\hat{d}_l \in (0,1)$ indicating the percentage of futures in which parcel l was purchased. To extract a HOP action $\mathbf{d}$ where d_l indicates whether to purchase l, we first set $d_l = 0$ whenever $\hat{d}_l = 0$, since purchasing l was not preferred in any future. Next, we cycle through each patch l with $\hat{d}_l = 1$ and purchase the patch (set $d_l = 1$). If there is remaining budget after processing all parcels with $\hat{d}_l = 1$, we sort all remaining parcels with $\hat{d}_l \in (0,1)$ in descending order. Then for each parcel, if purchasing it does not violate the budget constraint we set $d_l = 1$ and otherwise set $d_l = 0$.

Step Size Control. Correctly controlling the step size α_i can have a large impact on efficiency, since each iteration involves solving N MIP problems (one per future). We follow the same, relatively standard, step size control as Kumar et al. [2012], where the step size is computed according to the gap between the feasible primal solution quality and the dual solution quality. In particular, after extracting a feasible solution for the primal, let APX_i be the sum of rewards on every future and $DUAL_i$ be the dual objective value, we set

$$\alpha_i = \frac{APX_i - DUAL_i}{\sum_{l,t,k}(\bar{\pi}^{t,(i)}_{k,l})^2}$$

5 Dual Decomposition for Baselines

There are multiple alternative heuristics that can be formulated within the same dual decomposition framework described above for HOP. These heuristics differ in terms of the horizon over which they consider future population spread and whether or not they consider the possibility of selecting actions in the future when selecting actions at the current moment. Below we describe two baselines within this framework that are included in our experiments.

GreedyZero Policy. The GreedyZero policy is our most near-sighted baseline as it selects the action that looks best assuming the population growth and available budget after the next decision epoch is zero. In particular, the futures for GreedyZero are simulated for only T_r years instead of until time H. Correspondingly, its MIP is exactly the same as the HOP MIP except that the horizon H is always replaced by T_r and no purchases are allowed after the first year.

HNoop Policy. The HNoop policy is unlike GreedyZero as it considers the population spread until the real horizon H. However, unlike HOP and similar to GreedyZero, it does not consider the possibility of selecting actions after the first time step. Thus, it evaluates purchasing actions according to how much long-term population spread they will facilitate assuming that the noop action is taken thereafter. The computation of HNoop is similar to our approach for HOP, except that we simply remove all "action variables" from each MIP$_k$ after the first time step, which prevents the consideration of future actions. This myopic policy will often work well, when the consideration of future actions is unimportant. However, when this is not the case, we might expect HOP to have an advantage. For example, in some conservation situations it is important to consider building longer term paths for population spread in order to encourage spread to a particularly good habitat. Such paths will often not result from purely myopic reasoning.

Interestingly, the conceptual definition of the HNoop policy corresponds exactly to the myopic policy proposed in the only prior work on adaptive conservation planning in Golovin et al. [2011]. However, their computation of HNoop was carried out via a greedy algorithm that considered greedily adding parcels into the purchased set one at a time until the budget was exhausted. Given certain submodularity assumptions that greedy algorithm came with approximation guarantees. Our framework provides an alternative ap-

proach to computing HNoop that is not purely greedy, instead relying on decomposition for efficiency.

6 Experimental Results

Our evaluation uses a real dataset of the Red-cockaded Woodpecker (RCW) recovery project from Sheldon et al. [2010] along with some hand-designed synthetic maps. The various problems differ in terms of the spatial layout of available parcels, the initial population of birds, and the set of parcels that are already reserved (i.e. free parcels). The real RCW map is from a large land region of the southeastern United States that was of interest to The Conservation Fund. The region was divided into 443 non-overlapping parcels (each with area at least 125 acres) and 2500 patches serving as potential habitat sites. Parcel costs were based on estimated land prices.

Throughout the experiments, we use a reliable MIP solver, IBM CPLEX, to directly solve MIPs if needed. Our experiments are performed on a single core machine with a memory limit of 6GB. We do not set time limit for the computation.

Performance of Dual Decomposition. Here we compare the use of Dual Decomposition (DD) for computing the HOP policy compared to directly applying CPLEX to the full HOP policy MIP, which includes all futures into a single MIP. This provides an indication of the optimality of DD, when CPLEX can solve the problems, and also the efficiency/scalability of the approach. We use the large RCW map and run DD until either the step size $\alpha \leq 0.001$ or a maximum of 50 iterations is reached.

Table 1 shows the HOP value computed by CPLEX and DD as well as the timing results for time horizon $H = 20$. When a method fails to return a solution, no value is shown in the table. From the table, we see that the objective value of DD is very close or equal to that of CPLEX, meaning that DD is providing an extremely close approximation to the HOP policy. When the number of future scenarios is small, the MIP instances seen by CPLEX are relatively small and can be solved quickly by CPLEX. As the number of futures increases, CPLEX takes a much longer time to find solutions compared to DD or even fails to solve the problem within a reasonable amount of time and memory. To make the problem more challenging, we increase the time horizon to $H = 40$ where the problem size is much larger. Table 2 shows the results. We see similar results, but they are more pronounced since the problems seen by CPLEX are now significantly larger. The expensive computation and failures of CPLEX indicate that the full MIP approach is not practical to solve real-world problems.

Table 1: Solution Quality and Run Time(H=20)

N	CPLEX-OBJ	DD-OBJ	CPLEX-Time(s)	DD-Time(s)
5	-393.0	-392.4	26.8	112.0
10	-389.6	-388.0	41.9	100.3
15	-354.5	-353.2	71.2	79.0
20	-382.6	-381.4	102.5	210.8
25	-383.2	-381.2	368.7	159.8
30	-378.6	-375.0	650.8	187.1
35	-380.7	-377.9	430.1	205.6
40	-394.2	-392.8	1201.3	247.6
45		-391.3		280.3

Table 2: Solution Quality and Run Time(H=40)

N	CPLEX-OBJ	DD-OBJ	CPLEX-Time(s)	DD-Time(s)
5	-502.4	-502.4	61.8	94.4
10	-480.8	-480.4	196.9	181.4
15	-505.7	-501.7	1039.7	294.2
20	-504.6	-504.6	1417.2	358.2
25	-500.8	-499.7	4091.1	487.0
30		-502.3		568.5

It is important to note that DD can be easily parallelized to achieve significant speedup in terms of the number of processors. In particular, if we use one processor per future, the wall clock time per iteration would be equal to the maximum time required to solve an individual future. If those times are nearly uniform then this would result in nearly linear speedup.

Adaptive Planning Results on Synthetic Maps. We now evaluate the quality of the HOP policies and compare it with the GreedyZero and HNoop policies. The results reported for each algorithm are averaged over 10 runs to account for randomness of the environment and sample futures. We note that we have attempted to apply other planning formalisms to this problem, such as Monte-Carlo Tree Search, but without success due to the extreme action and stochastic branching factors of our problems.

We created two simple grid maps (Figure 2) to illustrate the advantage of the non-myopic HOP policy over the more myopic baselines. In the maps, each grid represents one parcel and the patches are marked using their indexes. The cost of each parcel is 1 and the annual budget is also 1. In other words, only one parcel can be purchased each year. In the first grid map, most parcels contain only one patch, but there are many parcels with two patches in the south of the initial population. Generally speaking, there are two possible directions of purchasing: to either the Northwest or the Southeast. Presumably, purchasing the Northwest part would lead the population to the most promising free area as long as the time horizon is large enough for the population to spread there, while the Southeast provides more instant benefit as each year two patches would be available instead of only one. It is obvious that the optimal policy would follows the first strategy in order to maximize the long-term re-

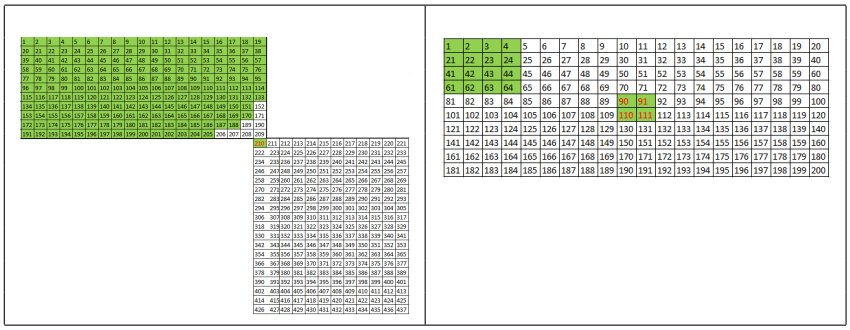

Figure 2: (Left) Grid Map 1. (Right) Grid Map 2. For both maps, the number marks the index of patches and each grid represents one parcel. The initially occupied patches are numbered in red and the free parcels are shaded in green.

ward. The results shown in Table 3 illustrate that HOP recognizes the potential benefit of the free parcels and purchases parcels towards the correct direction, while GreedyZero and HNoop are easily distracted by the Southeast purchasing due to their myopia.

Grid map 2 is similar to grid map 1, but the purchasing is not limited to only two directions so a planner would have more choices of purchasing, which also means more distractions from the far-away free area. Again, HOP recognizes the potential benefit of the free parcels and purchases parcels towards that direction, while GreedyZero and HNoop expand the reserve around the initial population uniformly. This leads HOP to achieve a higher reward as shown in Table 3.

Table 3: Rewards on Different Maps

Map	HOP	HNoop	GreedyZero
Grid map 1	85.2	35.0	70.2
Grid map 2	32	25.1	23.2
Real map	248.75	220.6	198.8

Adaptive Planning Results on Real Map. The real map (Figure 3) shows, via red + marks, where the initial bird population is, and free parcels are shaded in dark gray. Parcels shaded in pink are expensive yet affordable ones. The right map gives the natural population spread that would result if all parcel were conserved. We see that the free area in the Northeast corner is promising for optimal reward, therefore the optimal solution prefers to build a path from the initial population to it as long as there is enough budget and time for diffusion. However, many parcels on such a path are comparatively more expensive, adding more distractions for myopic decision makers.

We present the reward data for the three policies in Table 3, showing that HOP gains more reward than others. To further check their strategies, we plot the purchased parcels and corresponding population spread of the HOP and HNoop policies in Figures 4 and 5. While not shown, the GreedyZero policy gradually purchases parcels around the initial population. Compared to GreedyZero, HNoop finds the Southwest part more beneficial for longer term population spread, so more parcels are bought in that direction. However, HNoop fails to recognize the most promising free area in the Northeast, which requires consideration of future actions. HOP rather recognizes the Northeast area and purchases expensive parcels to build a path to the free parcels. In addition, HOP also buys parcels around the initial population if the reward gain is justified.

7 Summary and Future Work

We presented an action selection approach for adaptive conservation planning where it is necessary to dynamically reason about an extremely large set of resource management decisions and how they will impact the stochastic population spread. Our algorithm, based on hindsight optimization (HOP), is the first non-myopic approach for this problem and was shown to be effective compared to natural baselines. The main technical contribution was to show how to compute HOP policies for huge factor action spaces, for which prior HOP algorithms were inapplicable.

In future work, we plan to consider more broadly defined problem classes that involve the adaptive control of stochastic diffusion processes and other types of stochastic exogenous events. Such problems expose serious limitations of most AI planning techniques, making them an interesting sub-class from an AI research perspective. It is also of interest to understand additional conditions under which HOP policies can provide theoretical guarantees, and also tractable modifications to HOP that might support such guarantees.

Acknowledgements

This work is supported by NSF under grants IIS-0964705.

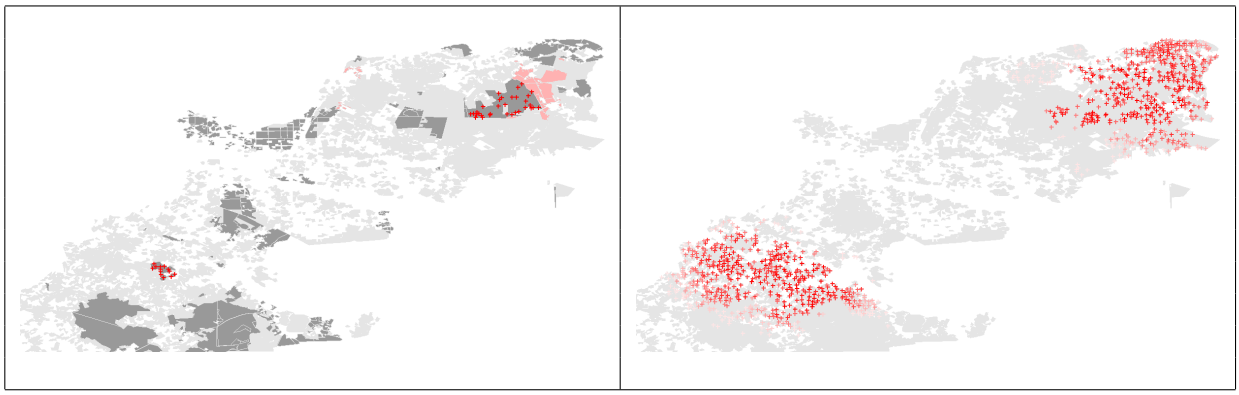

Figure 3: (Left) Initial state of the real map. (Right) The population distribution after 20 years on the real map.

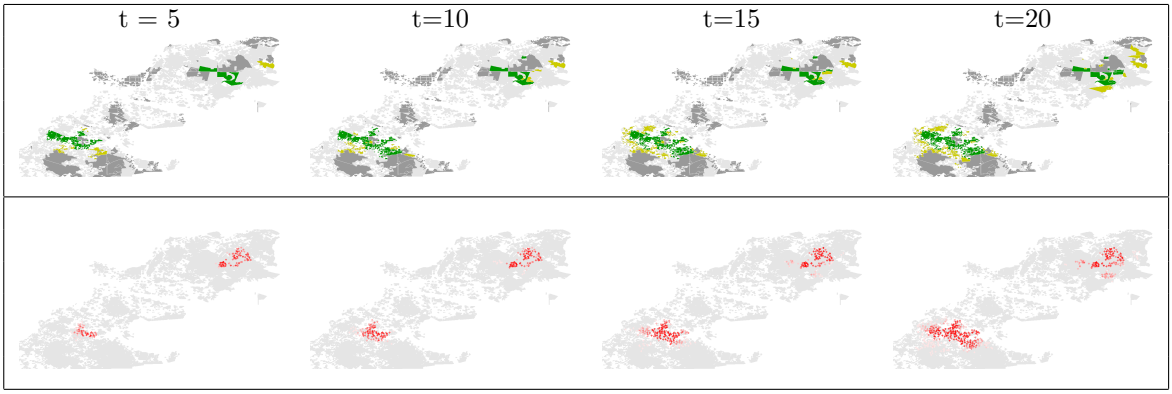

Figure 4: Purchases and population spread of HNoop. Parcels shaded in green are purchased with a probability of ≥ 0.5. Yellow is used for parcels with purchase probability of < 0.5. Patches are colored by the probability of being occupied (lighter color indicates lower probability).

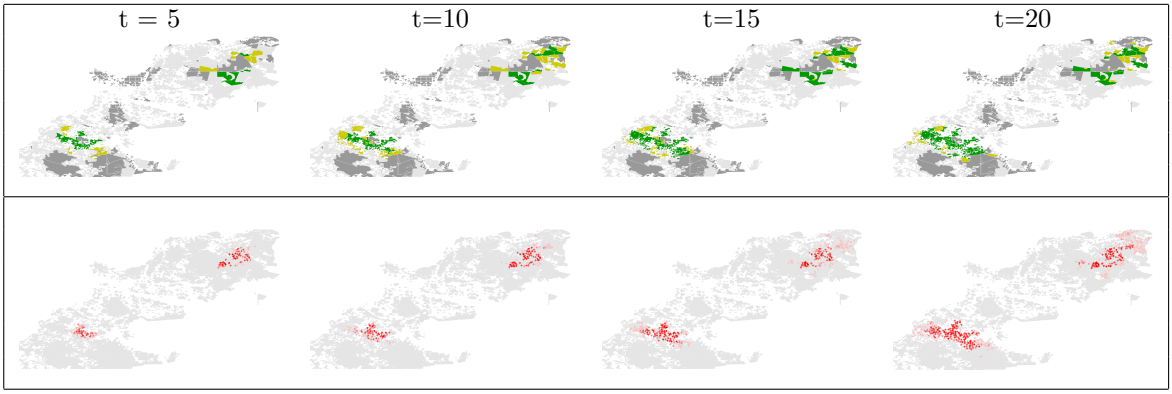

Figure 5: Purchases and population spread of the HOP policy. The color setting is the same as Figure 4.

References

K. Ahmadizadeh, C. Dilkina, C. P. Gomes, and A. Sabharwal. An empirical study of optimization for maximizing diffusion in network. In *CP '10: 16th International Conference on Principles and Practice of Constraint Programming*, 2010.

R. Bent and P. V. Hentenryck. Regret only! online stochastic optimization under time constraints. In *AAAI '04: Proceedings of the Nineteenth Conference on Artificial Intelligence*, 2004.

D. P. Bertsekas. *Nonlinear Programming, 2nd edition.* Athena Scientific, 1999.

H. S. Chang, R. L. Givan, and Chong E. K.P. On-line scheduling via sampling. In *AIPS '00: Artificial Intelligence Planning and Scheduling*, 2011.

E. K.P. Chong, R. L. Givan, and H. S. Chang. A framework for simulation-based network control via hindsight optimization. In *IEEE CDC '00: IEEE Conference on Decision and Control*, 2000.

D. Golovin, A. Krause, B. Gardner, S. J. Converse, and S. Morey. Dynamic resource allocation in conservation planning. In *AAAI '11: Proceedings of the Twenty-Fifth Conference on Artificial Intelligence*, 2011.

A. Hubbe, W. Ruml, S. Yoon, J. Benton, and M. B. Do. On-line anticipatory planning. In *ICAPS '12: International Conference on Automated Planning and Scheduling*, 2012.

A. Kumar, X. Wu, and S. Zilberstein. Lagrangian relaxation techniques for scalable spatial conservation planning. In *AAAI '12: Proceedings of the Twenty-sixth Conference on Artificial Intelligence*, 2012.

L. Mercier and P. V. Hentenryck. Performance analysis of online anticipatory algorithms for large multistage stochastic integer programs. In *IJCAI '07: International Joint Conference on Artificial Intelligence*, 2007.

D. Sheldon, B. Dilkina, A. Elmachtoub, R. Finseth, A. Sabharwal, J. Conrad, C. Gomes, D. Shmoys, W. Allen, O. Amundsen, and B. Vaughan. Maximizing the spread of cascades using network design. In *UAI '10: Uncertainty in Artificial Intelligence*, 2010.

G. Wu, E. Chong, and R. Givan. Burst-level congestion control using hindsight optimization. *IEEE Transactions on Automatic Control*, 47:979–991, 2002.

S. Xue, A. Fern, and D. Sheldon. Scheduling conservation designs via network cascade optimization. In *AAAI '12: Proceedings of the Twenty-sixth Conference on Artificial Intelligence*, 2012.

S. Yoon, A. Fern, R. L. Givan, and S. Kambhampati. Probabilistic planning via determinization in hindsight. In *AAAI '08: Proceedings of the Twenty-third Conference on Artificial Intelligence*, 2008.

S. Yoon, W. Ruml, J. Benton, and M. B. Do. Improving determinization in hindsight for online probabilistic planning. In *ICAPS '10: International Conference on Automated Planning and Scheduling*, 2010.