Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Dynamic Optimization of Landscape Connectivity Embedding Spatial-Capture-Recapture Information

Yexiang Xue
Dept. of Computer Science
Cornell University
yexiang@cs.cornell.edu

Xiaojian Wu
Dept. of Computer Science
Cornell University
xw458@cornell.edu

Dana Morin
NY Coop. Fish & Wildlife Res. Unit
Dept. of Natural Resources
Cornell University
djm466@cornell.edu

Bistra Dilkina
College of Computing
Georgia Institute of Technology
bdilkina@cc.gatech.edu

Angela Fuller
U.S. Geological Survey
NY Coop. Fish & Wildlife Res. Unit
Dept. of Natural Resources
Cornell University
akf34@cornell.edu

J. Andrew Royle
U.S. Geological Survey
Patuxent Wildlife Research Center
aroyle@usgs.gov

Carla P. Gomes
Dept. of Computer Science
Cornell University
gomes@cs.cornell.edu

Abstract

Maintaining landscape connectivity is increasingly important in wildlife conservation, especially for species experiencing the effects of habitat loss and fragmentation. We propose a novel approach to dynamically optimize landscape connectivity. Our approach is based on a mixed integer program formulation, embedding a spatial capture-recapture model that estimates the density, space usage, and landscape connectivity for a given species. Our method takes into account the fact that local animal density and connectivity change dynamically and non-linearly with different habitat protection plans. In order to scale up our encoding, we propose a sampling scheme via random partitioning of the search space using parity functions. We show that our method scales to real-world size problems and dramatically outperforms the solution quality of an expectation maximization approach and a sample average approximation approach.

1 Introduction

Conserving our world's declining wildlife is a central problem in computational sustainability (Gomes 2009). Maintaining landscape connectivity allows animals to move among resource patches (Taylor et al. 1993) and is becoming increasingly important in wildlife conservation, especially for species experiencing effects of habitat loss and fragmentation. Conservation plans that optimize landscape connectivity for wildlife, typically with limited budgets, are important to maximize the value of scarce resources devoted to the management and conservation of species. While there has been work in this research area, e.g., (Williams and Snyder 2005; Conrad et al. 2012; Dilkina and Gomes 2010; Sheldon et al. 2010; Wu, Sheldon, and Zilberstein 2014), existing approaches *decouple* the optimization problem of deciding what land parcels to protect *from* the problem of estimating what parcels provide landscape connectivity. In other words, the standard approach in previous work is to incorporate static, pre-computed parameters, derived from a separate model that estimates landscape resistance to movement, into the optimization problem for choosing what parcels to conserve. Further, previous approaches have not considered local abundance or density of the species in conjunction with landscape connectivity, even though protection of a minimum number of individuals and maintaining connectivity are important considerations for species persistence.

In contrast, our approach *dynamically* optimizes landscape connectivity weighted by local animal densities, given constrained budgets, by *simultaneously* estimating animal movement as a function of the conservation plan. This problem is challenging mainly because the spatial proximity of landscape parcels influences parcel specific density and connectivity values for a species, such that a change to any single parcel can result in different density and connectivity values to nearby parcels. We use a well-established ecological model, the spatial capture-recapture (SCR) ecological distance model, to estimate species abundance and landscape connectivity based on species spatial encounter history data (Royle et al. 2013; Sutherland, Fuller, and Royle 2015; Fuller et al. 2016). The SCR ecological distance model estimates where individuals locate their home ranges and how individuals use space within their home range based on where individuals are detected in relation to possible trap locations.

More specifically, our contributions are: **(1)** A novel approach that **encapsulates a spatial capture-recapture model**, that estimates the population density, space use, and landscape connectivity, **into a mixed-integer programming formulation** (MIP) for deciding what parcels to protect, under limited budgets; **(2)** Our approach takes into account the fact that animal density and connectivity change **dynamically and non-linearly** with different protection plans; **(3)** In order to scale up our encoding, we propose **a novel sampling approach via random partitioning of the search space using parity functions**; and **(4)** We show that our method scales to real-world size problems and **dramat-**

ically outperforms the solution quality of an expectation maximization approach and a sample average approach.

2 Spatial Capture Recapture Model

The Spatial Capture-Recapture Model (SCR) (Royle et al. 2013) is a probabilistic model in ecology describing animal movements within a home range based on capture rates in relation to a spatial trap array. The landscape can be modeled as a graph $G = \{V, E\}$. Each node $v \in V$ represents a small land parcel and is associated with an animal population density D_v, which describes the number of animals whose activity center is at v. Each edge e represents a path connecting two neighboring land parcels and is associated with a resistance value r_e, *measuring the cost for animals to travel across the edge*. The resistance is additive, that is, the resistance of a path $\mathcal{P} = \{e_0, e_1, ..., e_k\}$ is $\sum_{e\in\mathcal{P}} r_e$. We treat the resistance of an edge as its length. The *minimal resistance path* between node s and t is the one connecting s and t with minimal resistance value. According to the SCR model, the probability for one animal whose activity center is at land parcel s to use parcel t is determined by the minimal resistance distance between s and t as

$$P(s \rightsquigarrow t) = p_0 \ \exp(-\alpha \cdot d_{s,t}),$$

where $d_{s,t}$ is the length of the minimal resistance path connecting node s and t. Since we mainly study the optimization problem to improve landscape connectivity, the SCR model is used to predict the effects of protection alternatives. Therefore, we assume that p_0 and α are given parameters. One can estimate them with field data following (Royle et al. 2013). Given this definition, the expected number of animals which use parcel t is $\sum_{s\in V} D_s P(s \rightsquigarrow t)$.

Density weighted connectivity (dwc) is the sum of the expected use of each parcel based on estimated cost of movement in relation to individual activity centers, weighted by the estimated density of activity centers in each parcel across the entire landscape. DWC is an important metric capturing the connectivity and density of animals over the landscape:

$$dwc = \sum_{t\in V} \left(\sum_{s\in V} D_s P(s \rightsquigarrow t) \right). \tag{1}$$

Landscape Connectivity Optimization Our problem is to choose a set of edges to protect which best maintain the connectivity of a landscape. We define a binary decision variable x_e for each edge. $x_e = 1$ means that the edge e is *protected*, at which time the cost of movement value is r_e^p. The financial cost to protect an edge is c_e. If $x_e = 0$, edge e is *unprotected*, which increases the resistance value from r_e^p to r_e^u ($r_e^u > r_e^p$). Notice that protecting one edge could decrease the length of the minimal resistance paths between multiple node pairs, which could in turn increase pairwise probabilities and the connectivity of the entire landscape. With slight modification, our constraint programming approach could also apply to problems attempting to select node parcels (instead of edges) to restore or to improve connectivity (McRae et al. 2012). For a given budget limit B, our goal is to find a set of edges to protect to maximize the density weighted connectivity in Equation (1).

The **Budget Constrained Density Weighted Connectivity optimization problem (BCDWC-Opt)** therefore is:

$$\begin{aligned} \max_{\boldsymbol{x}} \quad & dwc = \sum_{t\in V} \left(\sum_{s\in V} D_s P(s \rightsquigarrow t) \right), \\ \text{s.t.} \quad & P(s \rightsquigarrow t) = p_0 \exp(-\alpha \cdot d_{s,t}), \\ & r_e = r_e^u + (r_e^p - r_e^u) x_e, \\ & \sum_{e\in E} x_e c_e \le B. \end{aligned} \tag{2}$$

Theorem 2.1. *The budget constrained density weighted connectivity optimization problem (2) is NP-hard. The density weighted connectivity function is neither submodular nor supermodular.*

The proof of Theorem 2.1 is left to the supplementary materials. Previously, connectivity design problems were studied in (Dilkina and Gomes 2010; Dilkina, Lai, and Gomes 2011; LeBras et al. 2013), in which they optimize for ecological models based on classical concepts in graph theory, such as the length of the shortest path, the number of node disjoint paths, etc. Our connectivity optimization problem is based on SCR – a well-established and more realistic ecological model. (Sheldon et al. 2010) studied the stochastic network design problem, which maximizes the spread of a cascade. Unlike our problem, the probabilities on edges are pre-defined, which is the key that allows them to simulate cascades in advance. The probabilities in our model change as a function of the protection plan, which is a more complex, yet more realistic setting. DWC is also related to a widely used landscape connectivity metric called the probability connectivity (PC) (Saura and Pascual-Hortal 2007). (Wu, Sheldon, and Zilberstein 2014) propose an approach to maximize PC only for tree-structured networks. Our algorithm can optimize PC on a general graph.

3 MIP for All Pairwise DWC

Auto Converge to the Minimal Resistance Path

One key challenge of the BCDWC-Opt Problem (2) is that it embeds a pairwise resistance minimization problem into the global optimization. According to the SCR model, $d_{s,t}$ must be the length of the minimal resistance path connecting s and t, which is nontrivial to incorporate into a MIP encoding, especially when the distance $d_{s,t}$ is an input of an exponential function in the objective function.

A rather interesting observation is that we do not have to explicitly enforce the minimal resistance. The following proposition guarantees that the minimal resistance condition will be automatically enforced in the optimal solution:

Proposition 3.1. *Define a variable $l_{s,t}$ to represent the length of a path connecting node s and node t. Denote the optimal solution of the original problem in (2) as OPT_d. The optimal solution of (2) with all $d_{s,t}$ substituted with $l_{s,t}$ as OPT_l. We have $OPT_d = OPT_l$.*

Intuitively, the objective function we maximize in (2) monotonically increases as the length of the path between (s, t) decreases. Therefore, as we maximize the objective

function (2), the path between each pair of nodes (s,t) will automatically converge to the shortest path, since the objective function further improves with a shorter path.

Motivated by Proposition 3.1, we use flow constraints to guarantee that $l_{s,t}$ is the length of an actual path connecting node s with t. More specifically, we push one unit of flow from s to t, and enforce $l_{s,t}$ to be the total length of edges in the flow. Because each edge in our problem has two states – either protected or unprotected, adopting the ideas from (Dilkina, Lai, and Gomes 2011), we replace each edge in the original graph G with two edges, one with the protected resistance as length, the other one with unprotected resistance as length. Let $I^p_{s,t,e}$ be a flow variable indicating whether edge e in its protected state is in the flow from s to t (and $I^u_{s,t,e}$ for the unprotected state). We have the following constraints:

$$l_{s,t} = \sum_{e\in E} r^u_e I^u_{s,t,e} + r^p_e I^p_{s,t,e}. \tag{3}$$

$$\sum_{e\leftarrow v}\left(I^p_{s,t,e}+I^u_{s,t,e}\right) - \sum_{e\rightarrow v}\left(I^p_{s,t,e}+I^u_{s,t,e}\right) = 0, \forall v\neq s,t, \tag{4}$$

$$\sum_{e\leftarrow v}\left(I^p_{s,t,e}+I^u_{s,t,e}\right) - \sum_{e\rightarrow v}\left(I^p_{s,t,e}+I^u_{s,t,e}\right) = 1, \text{ if } v=s, \tag{5}$$

$$\sum_{e\leftarrow v}\left(I^p_{s,t,e}+I^u_{s,t,e}\right) - \sum_{e\rightarrow v}\left(I^p_{s,t,e}+I^u_{s,t,e}\right) = -1, \text{ if } v=t, \tag{6}$$

$$0\le I^p_{s,t,e}\le 1,\ \ 0\le I^u_{s,t,e}\le 1. \tag{7}$$

Constraints (4) to (7) are flow constraints. $e\leftarrow v$ (or $e\rightarrow v$) means that edge e starts (or ends) at node v, respectively. Equation (3) guarantees that $l_{s,t}$ is indeed the length of a path connecting s and t. Finally, we need to enforce the constraint that there cannot be any flow passing $I^p_{i,j,e}$ unless edge e is protected. This can be achieved by enforcing

$$I^p_{s,t,e}\le x_e,\ \ \forall s,t\in V, e\in E. \tag{8}$$

Tradeoff Between Encodings The aforementioned encoding is the best one among a few other encodings that we tried, since it does not introduce extra binary variables to formulate the minimal resistance distance part. However, the tradeoff is a large number of continuous variables, e.g., the number of flow variables is $O(|V|^2|E|)$.

Encoding the Objective Function

We now encode the non-linear objective function:

$$dwc = \sum_{t\in V}\sum_{s\in V} D_s P(s\rightsquigarrow t|l_{s,t}),$$

We write $P(s\rightsquigarrow t|l_{s,t})$, mainly to emphasize that the probability depends on the value of $l_{s,t}$, in which $l_{s,t}$ corresponds to the distance of a path between s and t, and has already been encoded in the previous section. To illustrate the idea, in this section we provide the encoding for one term of dwc: $P(s\rightsquigarrow t|l_{s,t}) = p_0\exp\left(-\alpha\, l_{s,t}\right)$. The non-linear function is convex, but we are maximizing (instead of minimizing) this function. We apply piecewise constant approximation to this objective function. Let $0=m_0<m_1\ldots<m_g$ be $g+1$ points. Let $o_i = p_0\exp(-\alpha\, m_i)$ $(i\in 0,\ldots,g)$. We use extra binary variables to model the following step function:

$$\hat{P}(s\rightsquigarrow t|l_{s,t}) = \begin{cases} o_i & \text{if } m_{i-1}\le l_{s,t}<m_i, i\in\{1,..,g\}\\ 0 & \text{if } l_{s,t}\ge m_g.\end{cases}$$

Algorithm 1: XOR_K($w:\mathcal{X}\times\mathcal{Y}\rightarrow\{0,1\}, k, T$)

Sample T pair-wise independent hash functions $h^{(1)}_k, h^{(2)}_k,\ldots,h^{(T)}_k:\mathcal{Y}\rightarrow\{0,1\}^k$;
Solve the following problem

$$\max_{\boldsymbol{x}\in\mathcal{X},\boldsymbol{y}^{(i)}\in\mathcal{Y}} \sum_{i=1}^{T} w(\boldsymbol{x},\boldsymbol{y}^{(i)}) \quad \text{s.t.}\quad h^{(i)}_k(\boldsymbol{y}^{(i)})=\mathbf{0},\quad i=1,\ldots,T. \tag{11}$$

Return **true** if the max value is larger than $\lceil T/2\rceil$, otherwise return **false**.

Algorithm 2: XOR_MAX_COUNT($w:\mathcal{X}\times\mathcal{Y}\rightarrow\{0,1\}, T$)

```
k = log2 |Y|;
while k > 0 do
    if XOR_K(w, k, T) then
        Return 2^k;
    end
    k ← k − 1;
end
Return 1;
```

We can obtain a good approximation bound of $P(s\rightsquigarrow t|l_{s,t})$ using $\hat{P}(s\rightsquigarrow t|l_{s,t})$, as long as we carefully choose the locations of m_i and a large enough number of segments g. For example, choose $g=\lceil p_0/\epsilon\rceil-1$, and choose correct values for m_i to make $o_i = p_0(g-i+1)/(g+1)$. This ensures that $|\hat{P}(s\rightsquigarrow t|l) - P(s\rightsquigarrow t|l)|<\epsilon$ for any l.

Theorem 3.2. *Suppose $|\hat{P}(s\rightsquigarrow t|l) - P(s\rightsquigarrow t|l)|<\epsilon$ for any l. Let the optimal solution to the original problem be OPT. Let the optimal solution with $\hat{P}$ as the objective function be OPT^* (measured in the original objective function). We must have $OPT\ge OPT^*\ge OPT-(\sum_{s,t}D_s)\epsilon$.*

The proof of Theorem 3.2 is left to the supplementary materials. We can use the following MIP Encoding to model the piecewise constant approximation function, introducing binary variables $H_{s,t,i}$ for $i\in\{1,\ldots,g\}$. Then the piecewise approximation could be represented as:

$$\hat{P}(s\rightsquigarrow t)(l_{s,t}) = \sum_{i=1}^{g} o_i H_{s,t,i}, \tag{9}$$

$$(H_{s,t,i}=1)\Rightarrow(m_{i-1}\le l_{s,t}<m_i)\,. \tag{10}$$

The indicator constraint (10) can be enforced with the big-M notation. Finally, taking the constant approximation allows us to prune unnecessary variables. If under the best circumstance where every edge is purchased, the length of the shortest path $d_{s,t}$ between node s and t is still beyond m_g, we do not include this pair of nodes into the MIP encoding.

4 XOR Encoding

The number of variables in the previous all-pair DWC encoding prevents it from scaling to large instances. For a network $G=(V,E)$, the number of continuous variables is in

the order of $O(|V|^2|E|)$, and the number of discrete variables is in the order of $O(|V|^2 g)$, where g is the number of segments in the piecewise constant approximation.

One way to circumvent this difficulty is a sampling approach. We sample a small number of (s,t) pairs and find a protection strategy that maximizes the density weighted connectivity of these sampled pairs. We hope that the found strategy is close to the optimal solution, even though it considers only those sampled pairs. However, no principal way is available to decide which pairs of nodes are more important to connect than others before solving the problem.

We therefore leverage a recent progress in hashing based sampling, which uses random XOR constraints to partition the entire space, and takes samples *dynamically* along the progress of the optimization, while enforcing the constraints of our BCDWC-Opt Problem. This drastically differs from the standard approaches that take samples before the optimization starts.

Recently, (Xue et al. 2016) used this idea to develop a randomized algorithm (`XOR_MAX_COUNT` shown in Algorithm 2) with *constant approximation guarantee* to solve the following general Max-Counting Problem:

$$\max_{\boldsymbol{x}} \sum_{\boldsymbol{y}} w(\boldsymbol{x}, \boldsymbol{y}). \tag{12}$$

Here, $\boldsymbol{x} \in \{0,1\}^m$, $\boldsymbol{y} \in \{0,1\}^n$ are binary variables, and w is a general binary function: $w : \{0,1\}^{m+n} \to \{0,1\}$. Here, we call $(\boldsymbol{x}, \boldsymbol{y})$ a *satisfiable solution* if and only if $w(\boldsymbol{x}, \boldsymbol{y}) = 1$. In the max-counting problem, one finds the best configuration $\boldsymbol{x}$ that maximizes the number of satisfiable solutions $(\boldsymbol{x}, \boldsymbol{y})$. Similarly, in the BCDWC-Opt Problem, one finds the best set of edges to protect that maximizes the density weighted connectivity.

The idea behind the `XOR_MAX_COUNT` Algorithm reflects one line of research (Gomes, Sabharwal, and Selman 2006; 2007; Gomes et al. 2007; Ermon et al. 2013; 2014), which approximates the counting problem $\sum_{\boldsymbol{y}} w(\boldsymbol{x}, \boldsymbol{y})$ with a series of decision or optimization problems, each of which enforcing additional parity constraints. The rigorous definition of pairwise independent hash functions in algorithm `XOR_MAX_COUNT` is left to the supplementary materials. To understand the intuitions behind the algorithm, it is sufficient to treat constraint $h_k(\boldsymbol{y}) = \boldsymbol{0}$ as imposing k random XOR constraints on $\boldsymbol{y} \in \mathcal{Y}$. Consider a fixed $\boldsymbol{x}_0$, and the following optimization problem:

$$\max_{\boldsymbol{y} \in \mathcal{Y}} \; w(\boldsymbol{x}_0, \boldsymbol{y}), \quad \text{s.t. } h_k(\boldsymbol{y}) = \boldsymbol{0}. \tag{13}$$

Assume the total number of satisfiable solutions is 2^{k_0}, i.e., $\sum_{\boldsymbol{y}} w(\boldsymbol{x}_0, \boldsymbol{y}) = 2^{k_0}$. Imposing one XOR constraint corresponds to randomly dividing the search space $\mathcal{Y}$ into two halves and discard one of the two halves (those do not satisfy the XOR constraint). This applies equally to satisfying and unsatisfying states. Therefore, half of the 2^{k_0} solutions are discarded, after imposing the first XOR constraint. Imposing k XOR constraints corresponds to repeating this operation of discarding half of the solutions k times. If $k < k_0$, with high probability, there will still be non-zero solutions left after k operations, in which case, problem (13) returns 1. Conversely, when $k > k_0$, with similar arguments, problem (13) returns 0 with high probability. In short, we can obtain a rough count on the number of configurations that makes $w(\boldsymbol{x}_0, \boldsymbol{y}) = 1$ via checking the outcome of problem (13).

After reducing the counting problem into an optimization with additional parity constraints as in Eq. (13), we can embed this optimization problem into the max-counting problem in Eq. (12), so that the entire problem becomes a single optimization over both $\boldsymbol{x}$ and $\boldsymbol{y}$:

$$\max_{\boldsymbol{x} \in \mathcal{X}} \max_{\boldsymbol{y} \in \mathcal{Y}} \; w(\boldsymbol{x}, \boldsymbol{y}), \quad \text{s.t. } h_k(\boldsymbol{y}) = \boldsymbol{0}. \tag{14}$$

The high level idea of our algorithm hence is to find the best $\boldsymbol{x}_0$, such that $w(\boldsymbol{x}_0, \boldsymbol{y})$ can be satisfiable with as many parity constraints added as possible, which indicates that $\sum w(\boldsymbol{x}_0, \boldsymbol{y})$ is large, and therefore $\boldsymbol{x}_0$ is a good solution to the max-counting problem. Algorithm `XOR_MAX_COUNT` takes the aforementioned intuition one step further. In its subprocedure `XOR_K`, it checks whether more than half of T optimization problems of type (13) return 1. All these optimization problems share the common $\boldsymbol{x}$, but have their local copy of variables $\boldsymbol{y}^{(i)}$. This is a necessary variance reduction step to establish the constant approximation result.

Our contribution is a novel way of encoding the BCDWC-Opt problem as a variant of the Max-Counting Problem, with an original mapping of the density weighted connectivity function to an unweighted one, with binary domain and range, taking a 2-approximation. The mapped problem can then be transformed into optimization problems with additional parity constraints, under the `XOR_MAX_COUNT` framework. Interestingly, this mapping can be done by only introducing mixed integer constraints. Our transformation also has very few sum variables, so the whole approach can be implemented with very short XOR constraints, which is highly important for scalability purposes.

Encoding as a Max-Counting Problem We first write the indices of starting and ending points in the density weighted connectivity function using binary representation. Assume $|V| = 2^w$ (it can be easily extended to the general case), denote $\boldsymbol{s} = (s_{w-1}, s_{w-2}, \dots s_0)$ and $\boldsymbol{t} = (t_{w-1}, t_{w-2}, \dots t_0)$ as the binary representation of the indices of s and t, respectively. The objective function of BCDWC-Opt can be written as:

$$\max_{\boldsymbol{x}} \sum_{\boldsymbol{s} \in \{0,1\}^w} \sum_{\boldsymbol{t} \in \{0,1\}^w} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}). \tag{15}$$

Here, $\boldsymbol{x}$ is the vector representing the edge protection plan. $D_{\boldsymbol{s}}$ is the density at the node whose binary index is $\boldsymbol{s}$, and $P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})$ is the transition probability from the node whose binary representation is $\boldsymbol{s}$ to the one whose binary representation is $\boldsymbol{t}$. The total number of variables in this binary representation is small, i.e., $2 \log_2 |V|$ variables. This results in short parity constraints being added in the `XOR_MAX_COUNT` framework, which is beneficial for scaling up the encoding.

Next we transform the weighted objective function (15) into a binary one, taking a 2-approximation. Introduce extra variables $\boldsymbol{u} = (u_1, \dots, u_{l-1})$. Let $D_{max} = \max_{s \in V} D_s$.

We enforce the following constraints:

$$\frac{D_{\boldsymbol{s}}P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})}{D_{max}} \leq \frac{1}{2^{l-j}} \Rightarrow u_j = 0, \forall j \in \{1, \ldots, l-1\}. \quad (16)$$

$$\frac{D_{\boldsymbol{s}}P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})}{D_{max}} > \frac{1}{2^{l}}. \quad (17)$$

Define $w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$ as a binary function that outputs 1 if and only if $(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$ satisfies constraint (16) and (17). The idea is to use the number of different configurations that make $w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) = 1$ to approximate the value of a weighted objective. We prove that the approximation is close by the following theorem:

Theorem 4.1. *If $w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$ is an indicator function that returns one if and only if $(\boldsymbol{x}, \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$ satisfies constraint (16) and (17), we have*

$$\frac{1}{2}\max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}}P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}) - \frac{D_{max}}{2^{l+1-2w}}$$

$$\leq \frac{D_{max}}{2^{l}} \max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t},\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) \leq \max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}}P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}).$$

The proof of Theorem 4.1 is left to supplementary materials. It informs us that the solution obtained by solving the unweighted problem $\max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t},\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$ is almost 2-approximation to the optimal solution of the original BCDWC-Opt problem that has a weighted objective function. For real-world networks, $w = \log_2 |V|$ is usually small. We can choose l to be much larger than $2w$, to make $\frac{D_{max}}{2^{l+1-2w}}$ very small till it can be ignored.

The XOR_K procedure with an unweighted objective function now consists of the following optimization:

$$\max_{\boldsymbol{x},\boldsymbol{s}^{(i)},\boldsymbol{t}^{(i)},\boldsymbol{u}^{(i)}} \sum_{i=1}^{T} w(\boldsymbol{x}; \boldsymbol{s}^{(i)}, \boldsymbol{t}^{(i)}, \boldsymbol{u}^{(i)}),$$

$$\text{s.t.} \sum_{e \in E} c_e x_e \leq B, \quad (18)$$

$$h_k^{(i)}(\boldsymbol{s}^{(i)}, \boldsymbol{t}^{(i)}, \boldsymbol{u}^{(i)}) = \boldsymbol{0}, \forall i = 0, \ldots, T.$$

Here, $(\boldsymbol{s}^{(i)}, \boldsymbol{t}^{(i)}, \boldsymbol{u}^{(i)})$ are replication variables of $(\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$. $h_k^{(i)}$ is a randomly sampled pairwise independent function for the i-th replication. The MIP encoding of problem (18) is left to the supplementary materials. The main challenge for the MIP encoding comes from enforcing the constraint that the flow for replication (i) is between the node whose binary representation is $\boldsymbol{s}^{(i)}$, and the node whose binary representation is $\boldsymbol{t}^{(i)}$. Unlike from the previous all-pair DWC encoding (section 3), $\boldsymbol{s}^{(i)}$ and $\boldsymbol{t}^{(i)}$ are not pre-defined. They are dynamically updated in the XOR_K procedure.

5 Competing Encodings

Expectation Maximization

The BCDWC-Opt Problem in (2) can be expressed as a parameter learning problem with hidden variables. Define two random variables S and T, both with domain V, and a random variable $\boldsymbol{z}$ taking value 0 or 1. Define $P(S = s, T = t) = \frac{D_s}{|V|\sum_{s \in V} D_s}$ as the prior distribution and $P(z = 1|S = s, T = t, \boldsymbol{x}) = p_0 \exp(-\alpha d_{s,t}(\boldsymbol{x}))$ as the conditional distribution of z given S and T. We wrote $P(z|S = s, T = t, \boldsymbol{x})$ to emphasize that it depends on the edge protection plan $\boldsymbol{x}$. Then, maximizing the objective of the BCDWC-Opt problem is equivalent to:

$$\max_{\boldsymbol{x}} \sum_{s,t} P(S = s, T = t)P(z = 1|S = s, T = t, \boldsymbol{x})$$

Assuming that we have the observation $z = 1$, in machine learning, this optimization problem is to find the value of parameters $\boldsymbol{x}$ to maximize the marginal probability of $z = 1$ while S and T are treated as hidden variables. This learning problem can be solved by the expectation maximization (EM) algorithm (Bishop 2007), which interleaves between the following two steps.

E step The posterior probability distribution of the hidden variables S and T given our observation is calculated by

$$P(S, T|z = 1, \boldsymbol{x}^{old})$$

$$= \frac{P(S, T)P(z = 1|S, T, \boldsymbol{x}^{old})}{\sum_{s,t \in V} P(s, t)P(z = 1|S = s, T = t, \boldsymbol{x}^{old})}.$$

$\boldsymbol{x}^{old}$ is the edge protection plan in the previous iteration.

M step We maxmize $Q(\boldsymbol{x}, \boldsymbol{x}^{old}) =$

$$\sum_{S,T} P(S, T|z = 1, \boldsymbol{x}^{old}) \ln P(z = 1, S, T|\boldsymbol{x}) \quad (19)$$

$$= \sum_{S,T} P(S, T|z = 1, \boldsymbol{x}^{old})(-\alpha\, d_{S,T}(\boldsymbol{x})) + const. \quad (20)$$

$$\text{s.t.} \sum_{e \in E} x_e c_e \leq B.$$

In this case, the M-step in (20) does not involve the exponential operator, so the M-step can be solved by maximizing the weighted sum of the shortest path. We can use classical techniques in EM algorithm (Bishop 2007) to show that the objective function is guaranteed non-decreasing with the E-step and the M-step.

Greedy

We also tried the greedy approach, which selects one edge to protect at a time until we run out of budget. At each step, the edge with the largest marginal gain, which is defined as the ratio between the increase of density weighted connectivity and its cost, is added. Interestingly, even Greedy is an expensive algorithm, since we have to maintain all pairs of shortest path during each iteration. When the resistance of one edge is updated (reduced), the best algorithm to update the all-pair shortest path takes time $O(|V|^2)$. Besides, since in each iteration we have to check which edge is the best one to add (then for each one updating the shortest path), the complexity of one greedy iteration is $O(|V|^2|E|)$, which is already too large for the problem size that we consider.

Sample Average Approximation

We also tried the Sample Average Approximation approach (Shapiro 2003), which randomly samples a small number of

(s,t) pairs and finds a protection strategy that maximizes the density weighted connectivity, only taking into account of these sampled (s,t) pairs. We are not aware of a good strategy to select samples intelligently. Currently, the samples are taken randomly, which could be improved as future work.

6 Experiments

We compare our algorithm against various algorithms, including the Expectation Maximization (EM) approach with 4 different initialization strategies, the Sample Average Approximation (SAA) and the greedy algorithm. Our approach and SAA are encoded and run as MIP without warm starting from any initial solutions. We first run our experiments on a set of small synthetic instances (25 parcels), where our all-pair DWC encoding (discussed in Section 3) scales, providing a nice lower bound on the objective function. We run experiments with different budgets. For each budget, we randomly generate 10 instances. We give all algorithms 4-hour time limit and 4GB of memory. After 4 hours, the solver returns the best feasible solution it has found so far. If no feasible solution is found, we treat it as failure, and report the result as if it has the worst objective value. The number of segments g is set to 3 for all-pair DWC algorithm ($o_g = \frac{1}{3}$), mainly due to scaling limitations. The Sample Average Approximation algorithm randomly samples 500 (s,t) pairs in DWC objective as its objective function, which is the largest number without exceeding the memory limit. The number of replications T in the XOR encoding is set to 11, based on empirical performance. Selecting a large T would improve solution quality, however at a cost of computational complexity. The number of XOR constraints we need to add is $O(\log_2 |V| + l)$. In our application, since $\log_2 |V|$ is small, we run `XOR_K` with different number of parity constraints k in parallel, and choose the best solution among the ones returned with different k. Empirically, the solutions of our XOR encoding keep improving until approximately one hour, at which time they become relatively stable. We compare the solutions returned by each algorithm by evaluating the density weighted connectivity objective function value. A better algorithm should return a solution which achieves a larger value in this metric. Because the objective value could vary by orders of magnitude across different instances, they are further normalized by the best solution among all solvers for each instance, which rescales the value to between 0 and 1. We call this value the fraction w.r.t. the best solution.

The upper panel of Figure 1 shows the median fraction w.r.t. the best solution for various solvers, averaged over 10 instances per budget on small instances. As we can see, the algorithm that always perform the best is the XOR encoding. It is the best in all budget levels except two. The suboptimality of the two cases is due to the randomness as well as the 2-approximation we take in the XOR encoding. It even outperforms the all-pair DWC encoding, which almost models the entire problem exactly, except for an approximation in the objective function. The solution values obtained by all-pair DWC encoding are pretty good, except for a few losses on certain budget levels. This is due to the coarse objective function approximation (small g chosen). We cannot enforce a more refined approximation while keeping the solver making sufficient progress within the time limit. The EM algorithm is sensitive to initialization (which is a known issue in machine learning). The initialization that works best for this set of benchmarks involves solving a related MIP problem upfront, which could take hours. Even EM with the best initialization performs worse than XOR encoding on small budgeted instances. The greedy and SAA algorithms perform poorly. We further compare the performance of the solvers on larger instances (225 parcels), for which the all-pair DWC encoding cannot scale. As shown in the lower panel of Figure 1, our XOR encoding still outperforms all other approaches.

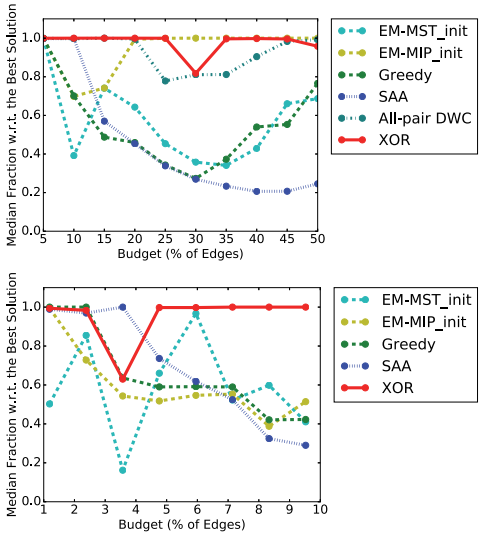

Figure 1: The XOR encoding outperforms the all-pair DWC encoding, EM with 4 initializations, SAA, and the greedy algorithm. (Upper) on small instances of 25 parcels. (Lower) on large instances of 255 parcels, where all-pair DWC encoding does not scale. The metric compares the dwc objective function of solutions obtained by each algorithm, normalized by the best solution among solvers. Higher is better. For clarity, we only plot the 2 best performing EM variants.

Finally, we run our XOR solver on large scale (1024 parcel) realistic instances (see details in the supplementary materials). The size of the problem is challenging. Even greedy algorithm cannot scale to problems with this size in reasonable amount of time. We run XOR solver with a 10-hour time limit. Figure 2 shows the edges purchased with our XOR encoding with a 200-edge budget, with different SCR model parameterizations. Here, α is the parameter in the exponential distribution, that tunes the expected distance that animals travel. As we can see, when α is small (0.02) in the left figure, in expectation animals travel far, so it becomes more important to connect long range density centers. As a result, the protection plan tend to purchase edges that form a connected component. In contrast, when α is large (0.45) in the right figure, animals stay mostly locally. In this case, many edges are bought to enforce local connectivity.

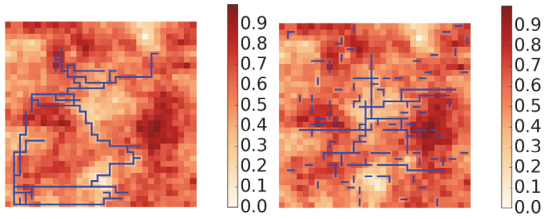

Figure 2: The solutions of the XOR encoding on large-scale simulated dataset (budget: 200 edges). The edge purchased are shown in blue. When α is small (Left $\alpha = 0.02$), the solutions found by the XOR encoding tend to form one connected component. When α is big, local connectivity is emphasized (Right $\alpha = 0.45$). The resistance matrix is in the background.

7 Conclusion

We propose a novel approach to dynamically optimize density weighted landscape connectivity, *simultaneously estimating animal movement* as a function of the conservation plan. Our approach is based on a mixed integer program, embedding the well-known spatial capture-recapture ecological model. Our approach dynamically adapts to the change of connectivity values of the entire landscape, as a result of the change of spatial proximity of several landscape parcels due to the selected conservation plan. In order to scale up our encoding, we propose a novel sampling scheme via random partitioning of the search space using parity functions. We show that our method scales to real-world size problems and dramatically outperforms the solution quality of an expectation maximization approach and a sample average approximation approach. This is a novel and scalable way of solving an important dynamic optimization problem in computational sustainability, with the potential of being generalized to many other domains.

Acknowledgements

This research was supported by National Science Foundation (0832782, 1522054, 1059284), ARO grant W911-NF-14-1-0498, and the Atkinson Center for a Sustainable Future.

References

Bishop, C. 2007. Pattern recognition and machine learning.

Conrad, J.; Gomes, C. P.; van Hoeve, W.-J.; Sabharwal, A.; and Suter, J. F. 2012. Wildlife corridors as a connected subgraph problem. *Journal of Environmental Economics and Management* 63(1).

Dilkina, B., and Gomes, C. P. 2010. Solving connected subgraph problems in wildlife conservation. In *CPAIOR*, 102–116.

Dilkina, B.; Lai, K. J.; and Gomes, C. P. 2011. Upgrading shortest paths in networks. CPAIOR.

Ermon, S.; Gomes, C. P.; Sabharwal, A.; and Selman, B. 2013. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proceedings of the 30th International Conference on Machine Learning, ICML*.

Ermon, S.; Gomes, C. P.; Sabharwal, A.; and Selman, B. 2014. Low-density parity constraints for hashing-based discrete integration. In *Proceedings of the 31th International Conference on Machine Learning, ICML*.

Fuller, A. K.; Sutherland, C. S.; Royle, J. A.; and Hare, M. P. 2016. Estimating population density and connectivity of american mink using spatial capturerecapture. *Ecological Applications* 26(4).

Gomes, C. P.; Van Hoeve, W.-J.; Sabharwal, A.; and Selman, B. 2007. Counting csp solutions using generalized xor constraints. AAAI.

Gomes, C. P.; Sabharwal, A.; and Selman, B. 2006. Model counting: A new strategy for obtaining good bounds. AAAI.

Gomes, C. P.; Sabharwal, A.; and Selman, B. 2007. Near-uniform sampling of combinatorial spaces using xor constraints. In *NIPS*.

Gomes, C. P. 2009. Computational Sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge, National Academy of Engineering* 39(4).

LeBras, R.; Dilkina, B. N.; Xue, Y.; Gomes, C. P.; McKelvey, K. S.; Schwartz, M. K.; and Montgomery, C. A. 2013. Robust network design for multispecies conservation. In *AAAI*.

McRae, B. H.; Hall, S. A.; Beier, P.; and Theobald, D. M. 2012. Where to restore ecological connectivity? detecting barriers and quantifying restoration benefits. *PLoS ONE*.

Royle, J. A.; Chandler, R. B.; Sollmann, R.; and Gardner, B. 2013. *Spatial capture-recapture*. Academic Press.

Saura, S., and Pascual-Hortal, L. 2007. A new habitat availability index to integrate connectivity in landscape conservation planning: comparison with existing indices and application to a case study. *Landscape and Urban Planning* 83(2):91–103.

Shapiro, A. 2003. Monte carlo sampling methods. *Handbooks in operations research and management science* 10:353–425.

Sheldon, D.; Dilkina, B.; Elmachtoub, A.; Finseth, R.; Sabharwal, A.; Conrad, J.; Gomes, C.; Shmoys, D.; Allen, W.; Amundsen, O.; and Vaughan, W. 2010. Maximizing the spread of cascades using network design. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 517–526.

Sutherland, C.; Fuller, A. K.; and Royle, J. A. 2015. Modelling non-euclidean movement and landscape connectivity in highly structured ecological networks. *Methods in Ecology and Evolution* 6(2).

Taylor, P. D.; Fahrig, L.; Henein, K.; and Merriam, G. 1993. Connectivity is a vital element of landscape structure. *Oikos* 73:43–48.

Williams, J. C., and Snyder, S. A. 2005. Restoring habitat corridors in fragmented landscapes using optimization and percolation models. *Environmental Modeling and Assessment* 10(3):239–250.

Wu, X.; Sheldon, D.; and Zilberstein, S. 2014. Stochastic network design in bidirected trees. In *Advances in Neural Information Processing Systems*.

Xue, Y.; Li, Z.; Ermon, S.; Gomes, C. P.; and Selman, B. 2016. Solving marginal map problems with np oracles and parity constraints. NIPS.