Supplementary Materials for 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

1 Proof for Theorem 2.1

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

Proof. First, we prove that the BCDWC-Opt problem is NP-hard. It can be proved by a reduction from the Knapsack problem. In a Knapsack problem, we have n items, each having a cost c_i and a value val_i. Given a budget B, we select a subset of items to maximize their total value with their total cost less than B.

To build an instance of our BCDWC-Opt problem to model the Knapsack problem, we build a graph $G = \{V, E\}$ with $2n$ nodes indexed by $(i_j)_{i=1:n,j=0:1}$ with all $D_{i_1} = 0$. Edges only exist between i_0 and i_1 or $E = \{(i_0, i_1)|\forall i = 1 : n\}$. Define $D_{i_0}, r^u_{i_0,i_1}, r^p_{i_0,i_1}$ arbitrarily as long as $D_{i_0}\left(exp(-r^p_{i_0,i_1}) - exp(-r^u_{i_0,i_1})\right) = val_i$. The cost of protecting edge (i_0, i_1) is c_i. Also, we set $p_0 = 1$ and $\alpha = 1$. That is, in this BCDWC-Opt problem, by protecting an edge with cost c_i, we can increase the objective by and only by val_i. So, with a budget B, the optimal strategy of edge protection corresponds to the optimal solution of the Knapsack problem.

Now, we use an example in Fig. 1 to show that the objective function $DWC(\cdot)$ is not submodular. We have

- $DWC(\phi) = 0$
- $DWC(\{e_1\}) = 100 * 0.9$
- $DWC(\{e_1, e_2\}) = 2(100 * 0.9 + 100 * 0.9 * 0.9)$

where $DWC(\{e_1\})$ means e_1 protected and the same meaning extends to any set of edges.

Let $A = \{e_1\}$, $B = \phi$ with $B \subseteq A$ and $x = e_2$. We have

$$DWC(A \cup \{x\}) - DWC(A) > DWC(B \cup \{x\}) - DWC(B)$$

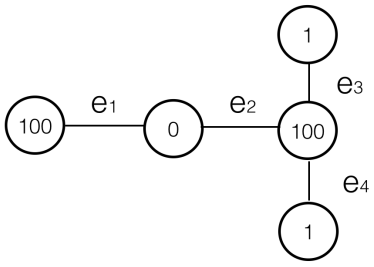

Figure 1: An example. Numbers in nodes represent density values $D(\cdot)$. We assume that all edges have the same resistance r^u and r^p, and r^u is very large and r^p is very small. We let $\exp(-\alpha \cdot r^u)$ be nearly 0 (can be treated as 0), $p_0 \exp(-\alpha \cdot r^p) = 0.9$, and $p_0 = 1$.

which violates the property of submodularity. The objective function is not submodular.

A different example can be built to show that the objective function is not super-modular. Briefly, we build a graph in which there are multiple paths between two nodes with very high densities and all other nodes have 0 density. If one path has been protected and has the highest probability, efforts to protect edges on other paths do not increase the objective effectively. □

2 Proof for Theorem 3.2

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*

$$OPT \geq OPT^* \geq OPT - (\sum_{s,t} D_s)\epsilon.$$

The definition of $\hat{P}$ guarantees that

$$\hat{P}(s \rightsquigarrow t|l) \leq P(s \rightsquigarrow t|l).$$

Because $|\hat{P}(s \rightsquigarrow t|l) - P(s \rightsquigarrow t|l)| < \epsilon$, we must have:

$$P(s \rightsquigarrow t|l) - \epsilon \leq \hat{P}(s \rightsquigarrow t|l) \leq P(s \rightsquigarrow t|l) \quad (1)$$

for any s, t pair, and any l. Suppose $\boldsymbol{x}_0$ is the optimal solution, Equation (1) guarantees that

$$\begin{aligned} OPT - (\sum_{s,t} D_s)\epsilon &= \sum_{s,t} D_s P_{\boldsymbol{x}_0}(s \rightsquigarrow t|l_{st}) - (\sum_{s,t} D_s)\epsilon \\ &\leq \sum_{s,t} D_s \hat{P}_{\boldsymbol{x}_0}(s \rightsquigarrow t|l_{st}) \\ &\leq \sum_{s,t} D_s P_{\boldsymbol{x}_0}(s \rightsquigarrow t|l_{st}). \end{aligned} \quad (2)$$

Here, $P_{\boldsymbol{x}_0}$ presents probabilities for a protection plan $\boldsymbol{x}_0$. Now suppose there is an algorithm which optimizes for objective function $\hat{P}$ and produces the optimal solution $\boldsymbol{x}_1$. By the definition of the optimal solution, we have:

$$\sum_{s,t} D_s \hat{P}_{\boldsymbol{x}_0}(s \rightsquigarrow t|l_{st}) \leq \sum_{s,t} D_s \hat{P}_{\boldsymbol{x}_1}(s \rightsquigarrow t|l_{st}). \quad (3)$$

By Equation (1),

$$\sum_{s,t} D_s \hat{P}_{\boldsymbol{x}_1}(s \rightsquigarrow t|l_{st}) \leq \sum_{s,t} D_s P_{\boldsymbol{x}_1}(s \rightsquigarrow t|l_{st}). \quad (4)$$

By the Definition that $\boldsymbol{x}_0$ is the optimal solution for objective function P, we have

$$\sum_{s,t} D_s P_{\boldsymbol{x}_1}(s \rightsquigarrow t|l_{st}) \leq \sum_{s,t} D_s P_{\boldsymbol{x}_0}(s \rightsquigarrow t|l_{st}) = OPT. \quad (5)$$

This is one side of the Inequality to prove. Combining the Equations (3), (4) and (2), we can have the other side of the Inequality.

3 Proof of Theorem 4.2

Theorem 4.2 *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 (15) and (16) in the main text, we have*

$$\begin{aligned} &\frac{1}{2}\max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}) - \frac{D_{max}}{2^{l+1-2w}} \\ \leq &\frac{D_{max}}{2^l} \max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t},\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) \leq \max_{\boldsymbol{x}} \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}). \end{aligned}$$

To prove Theorem 4.2, we need the following two lemmas:

Lemma 3.1. *For fixed $\boldsymbol{x}$, $\boldsymbol{s}$, $\boldsymbol{t}$, we have*

$$\begin{aligned} \frac{1}{2} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}) - \frac{D_{max}}{2^{l+1}} &\leq \frac{D_{max}}{2^l} \sum_{\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) \\ &\leq D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}). \end{aligned} \quad (6)$$

Proof. For $j = 1, \ldots, l-1$, when $\frac{1}{2^{l-j+1}} < \frac{D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})}{D_{max}} \leq \frac{1}{2^{l-j}}$, $u_j, u_{j+1}, ..., u_{l-1}$ are all fixed to 0 (Equation (13) in main text). $u_1, \ldots, u_{j-1}$ are not constrained, so $\frac{1}{2^l} \sum_{\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) = \frac{1}{2^{l-j+1}}$. We have

$$\frac{1}{2} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}) - \frac{D_{max}}{2^{l+1}} \leq \frac{D_{max}}{2^{l-j+1}}$$

Equation (6) holds.

When $\frac{D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})}{D_{max}} > \frac{1}{2}$, all $u_1, \ldots u_{l-1}$ are free, so $\frac{1}{2^l} \sum_{\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) = \frac{1}{2}$ (note $\frac{D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})}{D_{max}} < 1$). Equation (6) holds as well.

When $\frac{D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})}{D_{max}} < \frac{1}{2^l}$, Equation (14) in the main text guarantees that $\frac{1}{2^l} \sum_{\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) = 0$. Equation (6) holds as well. □

Lemma 3.2. *For fixed $\boldsymbol{x}$, we have*

$$\begin{aligned} \frac{1}{2} \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}) - \frac{D_{max}}{2^{l+1-2w}} &\leq \frac{D_{max}}{2^l} \sum_{\boldsymbol{s},\boldsymbol{t},\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) \\ &\leq \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}). \end{aligned} \quad (7)$$

Proof. Add the left and right hand side of Equation (6) together for different s and t. Note: $w = \log_2 |V|$. □

Proof. (Theorem 4.2) From Lemma 3.2,

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

holds for all fixed $\boldsymbol{x}$. Note that the probability $P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t})$ depends on $\boldsymbol{x}$. Let $\boldsymbol{x}_0$ be the one that attains maximal value of the objective function $\frac{D_{max}}{2^l} \sum_{\boldsymbol{s},\boldsymbol{t},\boldsymbol{u}} w(\boldsymbol{x}; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u})$, then we have $\frac{D_{max}}{2^l} \sum_{\boldsymbol{s},\boldsymbol{t},\boldsymbol{u}} w(\boldsymbol{x}_0; \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}) \leq \sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}; \boldsymbol{x}_0)$. Further because $\sum_{\boldsymbol{s},\boldsymbol{t}} D_{\boldsymbol{s}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}; \boldsymbol{x}_0) \leq \max_{\boldsymbol{x}} P(\boldsymbol{s} \rightsquigarrow \boldsymbol{t}; \boldsymbol{x})$, we have the second inequality of Theorem 4.2. The first inequality is similar. □

4 Definition of Pairwise Independent Functions

A family of functions $\mathcal{H} = \{h : \{0,1\}^n \rightarrow \{0,1\}^k\}$ is said to be *pairwise independent* if any function h randomly chosen from $\mathcal{H}$ satisfies: (i) $\forall \boldsymbol{x} \in \{0,1\}^n$, $h(\boldsymbol{x})$ is uniformly randomly distributed in $\{0,1\}^k$; (ii) $\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \{0,1\}^n$, $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$, the random variables $h(\boldsymbol{x}_1)$ and $h(\boldsymbol{x}_2)$ are independent. As a special case, when matrix $A \in \{0,1\}^{k \times n}$ and vector $\boldsymbol{b} \in \{0,1\}^k$ are randomly uniformly sampled, $\mathcal{H}_{A,b} = \{h_{A,b} : h_{A,b}(\boldsymbol{x}) = A\boldsymbol{x} + \boldsymbol{b} \bmod 2\}$ forms a family of pairwise independent functions. Notice that in this case $h_{A,b}(\boldsymbol{x})$ also correpsonds to a binary function formed with k parity constraints.

5 MIP encoding for XOR_k

For replication (i) in XOR_k algorithm, we need establish the following flow constraints:

$$\sum_{e \leftarrow v} \left(I_e^{p,(i)} + I_e^{u,(i)} \right) - \sum_{e \rightarrow v} \left(I_e^{p,(i)} + I_e^{u,(i)} \right) = \mathbf{1}(bin(v) = \boldsymbol{s}^{(i)}) - \mathbf{1}(bin(v) = \boldsymbol{t}^{(i)}), \forall v \in V. \quad (8)$$

Here, $I_e^{p,(i)}$ and $I_e^{u,(i)}$ are flow variables for replication (i). $\mathbf{1}(bin(v) = \boldsymbol{s}^{(i)})$ is an indicator variable that is one iff the binary representation of the index of node v is $\boldsymbol{s}^{(i)} = (s_{w-1}^{(i)}, \ldots, s_0^{(i)})$. We introduce $s_{w-1}^{(i)}, \ldots, s_0^{(i)}$ as binary variables in the MIP encoding. We use a continous variable $\beta_v^{(i)}$ ($0 \leq \beta_v^{(i)} \leq 1$) to represent this indicator variable. For example, when $v = 1$, we can enforce the following contraints:

$$\beta_v^{(i)} \leq 1 - s_k^{(i)}, \quad \forall k = 1, \ldots, w-1$$

and

$$\beta_v^{(i)} \leq s_0^{(i)},$$

and

$$\beta_v^{(i)} \geq s_0^{(i)} + \sum_{k=1}^{w-1} (1 - s_k^{(i)}) - w + 1.$$

This guarantees that $\beta_v^{(i)}$ is the indicator variable for $\mathbf{1}(bin(v) = \boldsymbol{s}^{(i)})$. Similarily, we introduce $\gamma_v^{(i)}$ as the indicator for $\mathbf{1}(bin(v) = \boldsymbol{t}^{(i)})$, and we use similar constraints to enforce it. Finally, we can enforce the flow constraint as:

$$\sum_{e \leftarrow v} \left(I_e^{p,(i)} + I_e^{u,(i)} \right) - \sum_{e \rightarrow v} \left(I_e^{p,(i)} + I_e^{u,(i)} \right) = \beta_v^{(i)} - \gamma_v^{(i)}. \quad (9)$$

Equation (15) (16) in the main text are essentially indicator constraints, which can be encoded using big-M notation.

6 Further Details about the Dataset

Synthetic Instances The density matrices are designed, with population centers located far away from each other. The edges are defined to connect adjacent parcels from top, bottom, left and right. The resistance matrix is a randomly located subregion of the resistance matrix for the realistic instance (see below). The price of purchasing an edge c_e is always 1. r_e^p – the resistance of an edge under protection is the average of the resistances of the two parcels it connects, and r_e^u is set to a large value.

Realistic Instances In order to generate realistic size instance we use a a kriging algorithm to create a 32 x 32 pixel spatially correlated surface of covariate values (resistance), simulating differential cost of movement among landscape pixels in a patchy environment. We scaled the covariate values between 0 and 1 to allow for only positive estimates of cost.