Efficiently Approximating the Pareto Frontier: Hydropower Dam Placement in the Amazon Basin

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

Jonathan Gomes-Selman
Dept of Computer Science
Stanford University
jgs8@stanford.edu

Qinru Shi
Dept of Computer Science
Cornell University
qs63@cornell.edu

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

Roosevelt García-Villacorta
Dept of Ecology and Evolutionary Biology
Cornell University
rg676@cornell.edu

Elizabeth Anderson
Dept of Earth & Environment
Florida International University
epanders@fiu.edu

Suresh Sethi
U.S. Geological Survey
New York Cooperative Fish and Wildlife Unit
Cornell University
suresh.sethi@cornell.edu

Scott Steinschneider
Dept of Biological & Environ. Engr.
Cornell University
ss3378@cornell.edu

Alexander Flecker
Dept of Ecology & Evolutionary Biology
Cornell University
asf3@cornell.edu

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

Abstract

Real–world problems are often not fully characterized by a single optimal solution, as they frequently involve multiple competing objectives; it is therefore important to identify the so-called Pareto frontier, which captures solution trade-offs. We propose a fully polynomial-time approximation scheme based on Dynamic Programming (DP) for computing a polynomially succinct curve that approximates the Pareto frontier to within an arbitrarily small $\epsilon > 0$ on tree-structured networks. Given a set of objectives, our approximation scheme runs in time polynomial in the size of the instance and $1/\epsilon$. We also propose a Mixed Integer Programming (MIP) scheme to approximate the Pareto frontier. The DP and MIP Pareto frontier approaches have complementary strengths and are surprisingly effective. We provide empirical results showing that our methods outperform other approaches in efficiency and accuracy. Our work is motivated by a problem in computational sustainability concerning the proliferation of hydropower dams throughout the Amazon basin. Our goal is to support decision-makers in evaluating impacted ecosystem services on the full scale of the Amazon basin. Our work is general and can be applied to approximate the Pareto frontier of a variety of multiobjective problems on tree-structured networks.

1 Introduction

In recent years there has been considerable interest in the study of multi-objective optimization problems (see e.g., (Ehrgott and Gandibleux 2000; Qian, Tang, and Zhou 2016; Terra-Neves, Lynce, and Manquinho 2017; Wiecek et al. 2008)). Multi-objective optimization is critical in Computational Sustainability (Gomes 2009), as real-world sustainability problems often involve balancing environmental, economic, and societal objectives. In multi-objective optimization, solutions are evaluated with respect to several, often competing, criteria. We are interested in understanding the trade-offs between the objectives for different solutions: the so-called Pareto frontier.

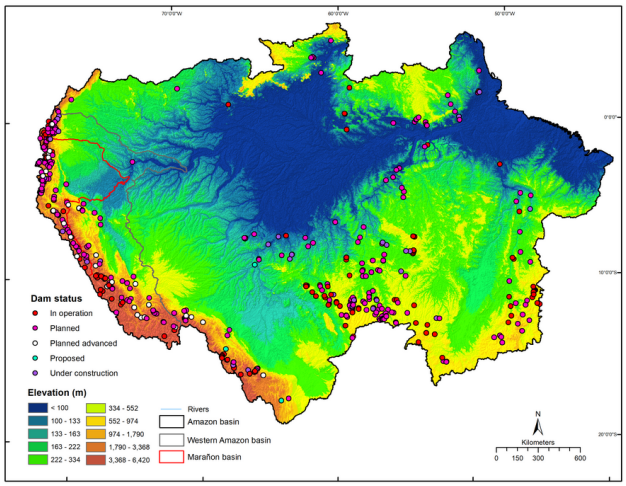


Figure 1: Dams throughout the Amazon basin: around 500 dams are proposed, planned, or under construction.

Our work is motivated by a challenging real-world computational sustainability problem concerning the placement of hydropower dams in a river network. In recent years, there has been a significant proliferation of hydropower dams in the Amazon basin: around 500 dams are proposed, planned, or under construction. Hydropower dams affect many ecosystem services such as, energy production, navigation, biodiversity, sediment export, nutrient cycling, and freshwater fisheries. Therefore, it is imperative to integrate ecosystem service trade-offs into decision-making models for the placement of hydropower dams in the Amazon Basin (Finer and Jenkins 2012; Kareiva 2012;

Winemiller et al. 2016; Zarfl et al. 2015; Ziv et al. 2012).

The Amazon hydropower dam placement problem represents a multiobjective optimization problem naturally captured by a tree-structured network. We are interested in mapping the Pareto frontier, since a single optimal solution does not fully characterize the problem: optimizing one objective often sacrifices the values of other objectives. The Pareto frontier is the set of all Pareto-optimal solutions, where a solution is considered Pareto-optimal if its vector of objective values is not dominated by the corresponding vector of any other feasible solution. Often the Pareto frontier is of exponential size, making it challenging to compute. By introducing an approximation of the exact frontier, we can replace the full frontier by a polynomial number of approximate Pareto points. Our approximate frontier is such that each Pareto solution in the full frontier is within a guaranteed approximation ratio from a point in the approximate frontier. Our approximation therefore provides an effective polynomial representation of the full frontier. Existing approaches are primarily heuristic based and provide no formal guarantees (e.g., (Wiecek et al. 2008; Ehrgott and Gandibleux 2000)). We develop *efficient algorithms for approximating the Pareto frontier for tree-structured networks with a guarantee on the approximation quality.*

Our contributions: 1) We provide an exact algorithm based on dynamic programming (DP) to compute the Pareto frontier on tree-structured networks. **2)** A DP-based fully polynomial-time approximation scheme (FPTAS) for approximating the Pareto frontier for tree-structured networks. **3)** We formulate the hydropower dam placement problem as a multiobjective optimization problem on a tree-structure network and demonstrate how our FPTAS approach can be applied to approximate its Pareto frontier. **4)** We also formulate the problem of optimizing the placement of hydropower dams as a mixed integer programming problem (MIP) and use it to approximate the Pareto frontier following the scheme proposed in (Papadimitriou and Yannakakis 2000). **5)** We provide empirical results on different basins of the Amazon to show interesting trade-offs between the two approaches. In general, our FPTAS captures the Pareto frontier more accurately than the MIP approach and provides a substantially larger set of Pareto frontier solutions. Nevertheless, in some cases, given that the MIP considers a smaller set of Pareto solutions (one solution per cell of the so-called ϵ-hyper-rectangle), it can take less time to approximate the Pareto frontier, despite its worst case exponential complexity. **6)** We also show that our exact DP approach can compute the full (exact) Pareto frontier for several Amazon sub-basins and several criteria, which is infeasible for the MIP approach.

Our DP and MIP Pareto frontier approximations have complementary strengths and are surprisingly effective, scaling up to real-world size problems: the DP approximates the Pareto frontier for the entire Amazon basin, when optimizing for energy, seismic risk, and connectivity (a proxy for e.g., fish migrations, transportation, sediment production), in around 1.7 hours, with a coverage of 19955 non-dominated solutions, with the guarantee that the solutions are within at most 10% of the true optimum ($\epsilon = 0.1$); in around 3 minutes, the DP provides a coverage of 3521 non-dominated solutions ($\epsilon = 0.2$); in around 1 minute, the DP provides a coverage of 1976 non-dominated solutions ($\epsilon = 0.25$); the MIP approach can approximate the Pareto frontier faster (≈ 16 seconds), but with a smaller coverage of 257 non-dominated solutions ($\epsilon = 0.1$).

Our approach will enable policy-makers to make better informed decisions concerning the trade-offs of socio-economic and environmental impacts of hydropower dam placements. We are developing analytical and visualization tools for an interactive exploration of the frontier for policy makers. The tools will allow policy makers to explore the trade-offs among the various solutions.

In the next sections we provide a formal characterization of the concept of Pareto frontier, describe our proposed FPTAS and MIP approaches for computing the Pareto frontier on tree-structured networks, and provide empirical results for different basins of the Amazon, demonstrating the trade-offs between the approaches.

2 Problem Statement

Pareto Frontier: We consider the trade-offs in optimizing *multiple objective functions* $(z^1, z^2, ..., z^m)$, whose values depend on a common solution (referred to as a *policy* π). In our application domain, a policy represents the selection of a particular set of dams to be built. Given such a policy (set of dams), we obtain values for our objective functions such as energy produced, seismic risk, etc. Without loss of generality, we only consider the problem of *maximizing* objective functions. Minimizing objective functions can be treated similarly. A *Pareto optimal* policy is the one that is not dominated by any other policy. Given two policies π and π', we say that π dominates π' if the following two conditions hold: (1) for all i, $z^i(\pi) \geq z^i(\pi')$; (2) at least one strict inequality holds for some i. The goal is to find all Pareto-optimal policies, which forms a set called the *Pareto frontier*.

Example: In our application domain, some objective functions that we consider are Energy (total energy generated by the dams), Connectivity (total river length from root without dams), and Seismic Risk (the overall seismic risk due to the constructed dams). These objective values are a direct function of the choice of dam placement and the structure of the underlying river network. We will define these objectives formally in later sections. For now, consider a very small example with only three possible dam placements P1, P2, and P3, with objective values (10, 5, 35) (i.e., Energy = 10, Connectivity = 5, and Seismic Risk = 35), (10, 3, 35), and (11, 4, 34), respectively. The policy P1 dominates P2 because of the higher Connectivity. P1 and P3 are non-dominated policies and form the Pareto frontier. (Note that P1 does not dominate P3 because of the Energy value, and P3 does not dominate P1 because of the Connectivity and the Seismic Risk.) See figure 2 for an example of a visualization of the Pareto frontier. In the next paragraph we define the problem formally and then we show how our formulation can be applied to the dam placement problem.

Value Function on a Tree Network: The objective functions that we consider are value functions defined on a tree

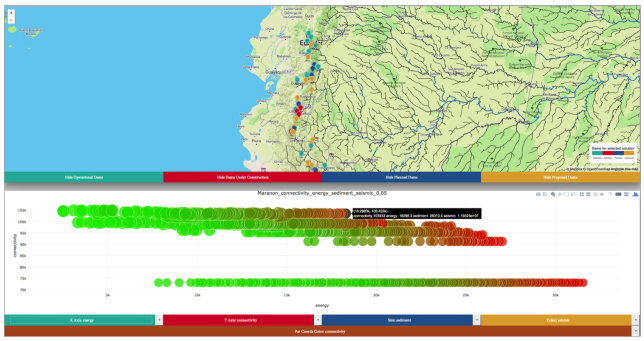


Figure 2: Example of a visualization of the Pareto frontier. (Top panel) Location of the selected dams corresponding to the solution highlighted in black in the bottom panel. (Bottom panel) Visualization of the Pareto frontier for four criteria for the Marañon sub-basin (DP solution; $\epsilon = 0.1$). X axis: energy; Y axis: connectivity; Size of the circle: Sediment; Color: Seismic risk (green - low risk; red - high risk). The solution in the far right of the top row of the circles is highlighted in black. Note that the solutions with the highest energy level have the lowest connectivity and the highest seismic risk.

structured network. A tree structured network $\mathcal{T} = (V, E)$ has a unique *root* node denoted by s. Any node v other than the root s is associated with a unique *parent* node u. v is also referred to as a *child* node of u. There is a directed edge linking the parent u to the child v. All children of u form the children set of u, denoted as $ch(u)$. A node is called a *leaf node* if its children set is empty.

We first give a general definition of the value function on a tree network. Some of the details will become clearer below, when we illustrate this definition in terms of the hydropower dam placement problem.

A value function $z^i(\pi)$ is defined recursively on a tree structured network. We assume there are node rewards $r_v^1, \ldots, r_v^m$ associated with each node v in the tree. The value function defined on a leaf node v is its corresponding reward, i.e., $z_v^i(\pi) = r_v^i$. Each edge has two states. If an edge uv is at its first state, then it has a passage probability of p_{uv}^i; otherwise, q_{uv}^i. In practice, the states of edges are used to model policies that we take on the edges. For example, in the dam placement problem, an edge represents a potential dam location, and the state of an edge represents our decision on whether or not to build the dam. A policy π is the subset of all edges that are in the first state. The value function on a non-leaf node v is defined recursively:

$$z_u^i(\pi) = r_u^i + \sum_{v \in ch(u)} \left(I(uv \in \pi) p_{uv}^i + I(uv \notin \pi) q_{uv}^i \right) z_v^i(\pi). \tag{1}$$

Here, $I(\cdot)$ is an indicator function. The value function for the entire tree network $\mathcal{T}$ is the value function on the root node s, i.e., $z^i(\pi) = z_s^i(\pi)$.

The value function defined in this way is general and captures many interesting families of functions. For example, the value function of a finite binary-state Markov Decision Process can be encoded in this way by unfolding the decision process into a chain, which is in a special tree form. The value function can also be equivalently written in a more concise way. Let $u \rightsquigarrow v$ denote the path from u to v. Define the probability that u is connected to v as

$$p_{u \rightsquigarrow v}^i(\pi) = \prod_{\text{e on } u \rightsquigarrow v} \left(I(e \in \pi) p_e^i + I(e \notin \pi) q_e^i \right),$$

that is, the product of all passage probabilities of edges on the path from u to v depending on their states. Then,

$$z^i(\pi) = \sum_{v \in V} p_{s \rightsquigarrow v}^i(\pi) r_v. \tag{2}$$

Given m objective functions $(z^1, z^2, \ldots, z^m)$, each defined as value functions on a tree network $\mathcal{T}$, our **multi-objective optimization problem on a tree structured network** is to find the Pareto frontier consisting of all non-dominated policies.

The Pareto multi-objective optimization problem is NP-hard even though it is defined on a tree. To prove the NP-completeness, consider the following decision problem: `Pareto_Dec`$(t_1, \ldots, t_m)$: *given $t_1, \ldots, t_m$, is there one policy π such that $z^i(\pi) \geq t_i$ for all i?*

Theorem 1. `Pareto_Dec` *is NP-complete even for two objectives.*

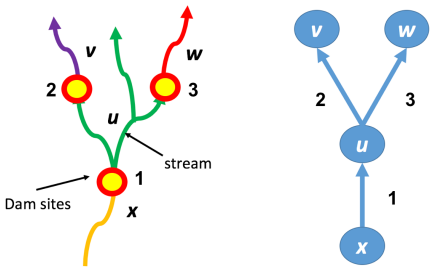
Proof: See the full version of the paper (Wu et al. 2018).

Application to the Hydropower Dam Placement Problem: We refer to the problem of determining the optimal placement of hydropower dams as the hydropower dam placement problem. As discussed earlier, we seek to balance multiple social, economic, and ecological metrics. The hydropower dam placement problem fits naturally into the general framework of the multi-objective optimization problem defined in the previous section.

To encode this problem, we first transform the river network and potential dam locations into a directed tree. A node in the directed tree corresponds to a contiguous region of the river network, that is, all stream segments in a region that are connected without being blocked by potential dam sites. An edge in the directed tree corresponds to a potential dam site. Figure. 3 provides an example of this transformation.

Our policy π is a subset of potential dam sites that we decide to build. Since each potential dam site is represented by an edge, the policy π, which is a subset of potential dam sites, corresponds to a subset of E, i.e., $\pi \subseteq E$. Building dams changes the passage probabilities for the edges on the tree network. If we build a dam, then the corresponding edge in the tree network is in its first state; otherwise it is in its second state. Using this framework, we can encode many important social and economic objectives, as follows:

Longitudinal connectivity: For a given policy π, the connectivity of a river network is measured by the total length of the stream segments that one can travel starting from the root without passing a selected dam site in π. Suppose we use the i^{th} objective to formulate the connectivity. We set r_v^i to be the total lengths of all stream segments in the region represented by node v in the directed tree. We set $p_e^i = 0$ and $q_e^i = 1$ for all $e \in E$, that is, we either acquire all upstream



(a) River Network (b) Dir. Rooted Tree

Figure 3: Converting a river network (left) into a directed rooted tree (right: x is root). One contiguous region of the river network (represented by different color) is converted to a node (also referred to as a hypernode) in the tree network. One potential dam site (represented by a red-yellow circle) is represented by an edge in the directed rooted tree.

segments (when we do not build the dam corresponding to edge e) or lose all of them (when we build the dam). It is easy to show that Equation (1) encodes the connectivity metric.

Energy: The total hydropower that is produced by a set of selected dam sites is $\sum_{e in \pi} h_e$, where h_e is the hydropower for the dam represented by edge e. Therefore, selecting a dam site on edge e adds energy h_e to the total energy. For any dam site (u, v) that can produce hydropower h_{uv}, we set $p^i_{uv} = q^i_{uv} = 1$. We move the energy h_{uv} to its parent u. In other words, the reward at node u, r^i_u now becomes $r^i_u(\pi) = \sum_{v \in ch(u)} I(uv \in \pi) h_{uv}$. Notice that r^i_u depends on whether each edge (u, v) is in policy π. The total energy can be calculated by the same recursion as in (1).

Sediments: This objective captures the total amount of sediment carried downstream. Dams stop a certain fraction of sediments from moving downstream. The formulation of this objective is very similar to connectivity: r^i_v represents the amount of sediment that is produced by the region encoded by a node v; $q^i_e = 1$ and p^i_e is the percentage of sediment that can go through the dam denoted by edge e.

Seismic Risk: Dam locations are associated with three levels of seismic risk: level 1 (no risk) to 2 (medium risk) or 3 (high risk). Let h_e be the hydropower produced by dam e, l_e be the risk level for its location, and $\eta = 10$ be a base risk score. We define the seismic hazard for dam e as zero if the dam is located in a level 1 region ($l_e = 1$); otherwise its risk is $(h_e + C)\eta^{l_e}$, which can be encoded analogously to the energy objective. C is a constant. We set it to 1000.

Many other objectives can be encoded in a similar way.

3 Dynamic Programming and Rounding

We first present a dynamic programming algorithm which computes the Pareto frontier exactly for value functions defined on a tree-structured network. We prove that the algorithm can find all Pareto optimal policies.

In practice, the number of Pareto optimal policies may be exponential even for a fixed number of objective functions. Motivated by the ideas in (Wu, Sheldon, and Zilberstein 2014a; 2014b), we apply a rounding technique to the dynamic programming algorithm. (Wu et al. do not consider the problem of approximating the Pareto frontier.) We provide a fully polynomial-time approximation scheme (FPTAS) that finds a policy set of polynomial size, which approximates the Pareto frontier within an arbitrary small ϵ and runs in time polynomial in the size of the instance and $1/\epsilon$.

Definition 1. *We say that a policy set S ϵ-approximates the Pareto frontier, if and only if for any policy π in the Pareto frontier, there exists a policy π' in set S, such that $z^i(\pi') \geq (1-\epsilon) z^i(\pi)$ for all i.*

Definition 2. *We say an algorithm A is a fully polynomial-time approximation scheme (FPTAS) for a multi-objective optimization problem, if A finds a policy set S that ϵ-approximates the Pareto frontier, and A runs in time polynomial in the size of the instance and $1/\epsilon$.*

As we will show below, our main contribution can be summarized by the following statement:

Suppose there are universal constants c and C such that $c \leq r^i_u \leq C$ for all $u \in V$ and all i. Our algorithm `Pareto_approx`$_{\mathcal{T}}$*, described below, is a FPTAS for the multiple objective optimization problems defined on a tree structured network. The size of policy set found by the algorithm is bounded by $O\left(\left(\frac{n}{\epsilon}\right)^m\right)$, and the running time is bounded by $O\left(\left(\frac{n}{\epsilon}\right)^{2m}\right)$, where n is the number of nodes in $\mathcal{T}$ and m is the number of objective functions.*

We assume that the tree $\mathcal{T}$ is a binary tree. In section 5, we discuss how to transform a general directed rooted tree into an equivalent binary tree, which allows us to apply the algorithm discussed in this section with the additional benefit of increased early pruning.

3.1 Exact Dynamic Programming Algorithm

Our proposed algorithm, `Pareto`$_{\mathcal{T}}$, is an exact dynamic programming algorithm, which computes all policies in the Pareto frontier. One may relate it to the bucket elimination algorithm in probabilistic inference (Dechter 1998). Nevertheless bucket elimination is applied to single objective problems; we note that having multiple objectives makes the problem structure very different from single objective problems. In fact, the complexity of variable elimination is bounded by the tree width, so tractable on trees. However, our problem is NP-hard even for two objectives and on a two-layer tree. In addition, we obtain an FPTAS for our problem, which is not provided by the bucket elimination algorithm, even for single objective problems.

Let Φ_u denote the set of all edges in the *subtree* rooted at u and π_u be a partial policy on u that only includes edges in Φ_u; in other words, $\pi_u \subseteq \Phi_u$. A partial policy is Pareto optimal if and only if it is not dominated by any other partial policies on the same node. Our dynamic programming algorithm is based on the following theorem, which states that all Pareto optimal partial policies on a node can be computed based on Pareto optimal partial policies from its children:

Theorem 2. *Let l and r be the children of u. Any Pareto-optimal partial policy at u can be constructed by combining one Pareto-optimal partial policy from node l, one from node r and four different joint states of edge (u, l) and (u, r).*

Proof. Prove by contradiction. See the full version of the paper (Wu et al. 2018). □

Motivated by Theorem 2, we design our exact dynamic programming algorithm, $\texttt{Pareto}_{\mathcal{T}}$, to recursively compute the Pareto optimal partial policies from leaf nodes to the root. Instead of keeping all partial policies, we only keep the Pareto-optimal partial policies at each node. The recursion of computing the Pareto optimal partial policy for node u is as follows. Suppose l and r are the two children of node u. We first compute the Pareto optimal partial policies for l and r. Then we enumerate each pair of partial policies from node l and r and consider four different combinations of whether to include edge (u, l) and (u, r). For each combination, we compute the objective values based on Equation 1, and add them to the policy set P. Finally, we remove all dominated policies from P. Based on Theorem 2, the remaining policies are Pareto optimal for node u. Note that it is straightforward to adapt this algorithm to include the case when the node u only has one child.

3.2 Approximation with Rounding

In practice, the optimal Pareto frontier may have an exponential number of points. In this case, we propose $\texttt{Pareto\_approx}_{\mathcal{T}}$, based on a rounding technique that gives a FPTAS to the multi-objective optimization problem.

We first explain how rounding works for longitudinal connectivity and sediments. The rounding for energy and seismic risk, which will be detailed later, is very similar to the rounding scheme presented here, except for an additional pre-rounding step to take into account the fact that the node reward varies according to policy.

We introduce a hyper-parameter K_v^i for each node and each objective. Define a rounded objective value $\hat{z}_u^i(\pi)$, which is calculated by the following recursion, $\hat{z}_u^i(\pi) =$

$$r_u^i + \left\lfloor \frac{\sum_{v \in ch(u)} \left(I(uv \in \pi) p_{uv}^i + I(uv \notin \pi) q_{uv}^i \right) \hat{z}_v^i(\pi)}{K_u^i} \right\rfloor K_u^i. \tag{3}$$

In Equation. 3, the fraction less than K_u^i is removed so that the number of different rounded objective values is reduced. This idea is similar to rounding different fractional numbers to the same integer value.

The rounded dynamic programming algorithm $\texttt{Pareto\_approx}_{\mathcal{T}}$, is similar to the DP algorithm, $\texttt{Pareto}_{\mathcal{T}}$, except that we use Equation 3 to calculate objective values.

Proposition 1. *$z^i(\pi) \geq \hat{z}^i(\pi)$ for any hyperparameter K_v^i and any policy π.*

This proposition is a direct consequence of taking the floor operator in Equation 3.

Proposition 2. *If we set $K_u^i = \epsilon r_u^i$, for any policy π we have*

$$z^i(\pi) - \hat{z}^i(\pi) \leq \epsilon z^i(\pi).$$

To see why this proposition is true, at each node, we at most lose a value of K_u^i. Based on Equation. 2, the total value we lost due to the floor operation in Equation. 3 is

$$z^i(\pi) - \hat{z}^i(\pi) \leq \sum_{v \in V} p_{s \rightsquigarrow v}(\pi) K_u^i = \epsilon z^i(\pi). \tag{4}$$

Proposition 3. *The approximate algorithm computes all Pareto optimal points for the rounded objective.*

This is because the approximate algorithm $\texttt{Pareto\_approx}_{\mathcal{T}}$ is the same as the exact algorithm $\texttt{Pareto}_{\mathcal{T}}$, except for replacing the exact objective function with a rounded one. The correctness of $\texttt{Pareto}_{\mathcal{T}}$ does not depend on particular objective function. Therefore, this proposition is true.

Theorem 3. *Let $P(s)$ be the set of (partial) Pareto optimal policies for a node s. Let $\overline{P}(s)$ be the set of approximate (partial) Pareto optimal policies computed via $\texttt{Pareto\_approx}_{\mathcal{T}}$ with rounded objective function (Equation 3). We must have $\overline{P}(s)$ ϵ-approximates $P(s)$.*

Proof. Suppose π is in the exact Pareto frontier $P(s)$. If π is in $\overline{P}(s)$, we are done. Otherwise, because of Proposition 3, there must be at least one $\pi' \in \overline{P}(s)$, such that π' dominates π in terms of the rounded objective; i.e.,

$$\hat{z}^i(\pi') \geq \hat{z}^i(\pi), \tag{5}$$

for all i. Because of Proposition 1, we have

$$z^i(\pi') \geq \hat{z}^i(\pi'). \tag{6}$$

Because of Proposition 2, we have

$$\hat{z}^i(\pi) \geq (1 - \epsilon) z^i(\pi). \tag{7}$$

Combining Equation (5) (6) and (7), we have

$$z^i(\pi') \geq (1 - \epsilon) z^i(\pi). \tag{8}$$

This concludes the proof. □

We are also able to show that $\texttt{Pareto\_approx}_{\mathcal{T}}$ runs in polynomial amount of time. To show this, we have to bound the number of partial policies in $\overline{P}(u)$ for each node u. In practice, we can assume that the objective functions are bounded.

Assumption 1. *There are universal constants c and C such that $c \leq r_u^i \leq C$ for all $u \in V$ and all i.*

Theorem 4. *With Assumption 1, the number of different rounded tuples in $\overline{P}(u)$ is $O\left(\left(\frac{n_u}{\epsilon}\right)^m\right)$, where n_u is the number of nodes in the subtree rooted at u.*

Proof. The upper bound of the i^{th} objective value of u is $UB_u^i = \sum_{v \in \mathcal{T}_u} r_v^i$ with $\mathcal{T}_u$ denoting the subtree rooted at u. Due to Assumption 1, the number of different tuples of u is

$$\prod_{i=1}^{m} \frac{UB_u^i}{K_u^i} \leq \prod_{i=1}^{m} \frac{n_u C}{\epsilon c} = O\left(\left(\frac{n_u}{\epsilon}\right)^m\right). \tag{9}$$

□

Theorem 5. *The runtime of the rounded dynamic programming algorithm is* $O\left(\left(\frac{n}{\epsilon}\right)^{2m}\right)$.

Proof. As we proved, the number of different rounded tuples $\overline{P}(u)$ at node u is $O\left(\left(\frac{n_u}{\epsilon}\right)^m\right)$. Let T_u be the runtime to compute all these tuples at u. We have the recursion

$$T_u = O(\overline{P}_l \overline{P}_r) + T_l + T_r. \tag{10}$$

Solving the recursion, we have $T_u = O\left(\left(\frac{n_u}{\epsilon}\right)^{2m}\right)$. Therefore, the total runtime is $T_s = O\left(\left(\frac{n}{\epsilon}\right)^{2m}\right)$ where n is the number of nodes in the tree. □

For energy and seismic risk: We apply a two-step rounding scheme for energy and seismic risk. Notice it can be applied to connectivity and sediments to obtain the same approximation guarantee. For presentation purposes, we use energy as an example. In the first step, we round upfront the energy value h_{uv} to

$$\lfloor \frac{h_{uv}}{K_1} \rfloor K_1.$$

in which $K_1 = \frac{\epsilon}{2} h_{min}$, where h_{min} is the minimal energy value among all dams. This rounding is similar to the one used in the FPTAS to solve the knapsack problem (see page 67 of (Williamson and Shmoys 2011)[1]). In fact, one can prove that we already get an FPTAS, if we only apply the first-step rounding. The proof is similar to that of the knapsack rounding.

The rounding in the first step is often too conservative. To obtain additional pruning, we apply a second rounding step in our DP approach when combining partial solutions from the children for the parent node. The rounding looks exactly like the one we use for connectivity and sediments in (3)[2], except that we do not round when $r_u^i(\pi) = 0$. The second rounding is more aggressive. As more partial solutions are rounded to the same value, the algorithm is able to prune more solutions, therefore scaling to larger instances.

To prove the approximation guarantee of the two-step rounding, we show that the first and the second rounding each incurs at most a $\epsilon/2$-approximation. Therefore, the entire algorithm has an ϵ-approximation guarantee. Let $z_u^i(\pi)$ be the true objective function without any rounding. $\tilde{z}_u^i(\pi)$ is the objective value after applying the first rounding. $\hat{z}_u^i(\pi)$ is the objective value after applying both the first and the second rounding. We are able to prove that

$$z_u^i(\pi) - \tilde{z}_u^i(\pi) \leq \frac{\epsilon}{2} z_u^i(\pi), \tag{11}$$

$$\tilde{z}_u^i(\pi) - \hat{z}_u^i(\pi) \leq \frac{\epsilon}{2} \tilde{z}_u^i(\pi). \tag{12}$$

[1]One difference: the rounding depends on the minimal energy value, rather than the maximal value as in the knapsack rounding. This is because the value of the most valuable object is naturally a lower bound to the knapsack problem, but it is not the case in our multi-objective optimization problem.

[2]K_v^i is set to $\lfloor \frac{r_u^i}{h_{min}} \rfloor \frac{\epsilon}{2} h_{min}$.

Adding these two inequalities together, we get Proposition 2, which the proof of the approximation guarantee mainly depends on. The proof to Equation 11 is similar to that for the knapsack algorithm. The proof to Equation 12 is similar to the proof of Proposition 2 with connectivity and sediments as objectives.

To prove that the DP algorithm runs in polynomial amount of time, we notice that during the execution of the algorithm, all intermediary energy values are in the form of kK_1, where k is an integer and K_1 is the rounding factor of the first step. As a consequence, the maximal number of different energy values at node u are bounded by

$$\frac{n_u h_{max}}{K_1} = \frac{2 n_u h_{max}}{\epsilon h_{min}} = O\left(\frac{n_u}{\epsilon}\right)$$

where n_u is the number of nodes for the subtree rooted at u. h_{max} is the maximal energy value.

Theorem 6. `Pareto_approx`$_{\mathcal{T}}$ *is an FPTAS for the multi-objective optimization problem on a tree structured networks.*

We provide a detailed proof of FPTAS with the two-step rounding scheme in the full version of the paper (Wu et al. 2018).

4 Mixed Integer Programming

We also formulate the problem of optimizing the placement of hydropower dams as a mixed integer programming problem (MIP) and use it to approximate the Pareto frontier following the scheme proposed in (Papadimitriou and Yannakakis 2000). While the resulting method has exponential worst case complexity because of the cost of solving the MIP, we find that in practice it is complementary to the DP approximation.

The scheme proposed by (Papadimitriou and Yannakakis 2000), which finds a policy set that ϵ-approximates the Pareto frontier, can be summarized as follows: for a multi-objective optimization problem, we divide the space of objectives into small rectangular cells and we query whether there exists a solution in each cell. We form a set S that includes one solution for each rectangular cell where there is at least one solution. It can be proved that if the boundaries of the rectangular cells are carefully chosen, the set of non-dominated solutions from S form a ϵ-approximate Pareto frontier.

Specifically, the rectangular cells are designed to satisfy the condition that, for each dimension, the upper bound is $(1+\epsilon)$ from the lower bound (assuming the objectives are always positive values). For our problem, we are considering four objectives: C (longitudinal connectivity), S (Sediment), H (Hydropower) and L (Seismic Risk). The four objectives are always positive. Assume that $C_{\min}$ and $C_{\max}$ are the minimum and maximum possible connectivity values for any dam building scheme. Let $\mathcal{C}$ be the set

$$\{C_{\min}(1+\epsilon)^k \mid k = 0, 1, 2, ...K_C, C_{\min}(1+\epsilon)^{K_C+1} \geq C_{\max}\}$$

$\mathcal{L}$, $\mathcal{H}$ and $\mathcal{S}$ are defined similarly. Then, each rectangular cell is of the form

$$[\hat{C}, (1+\epsilon)\hat{C}] \times [\hat{S}, (1+\epsilon)\hat{S}] \times [\hat{H}, (1+\epsilon)\hat{H}] \times [\hat{L}, (1+\epsilon)\hat{L}]$$

where $\hat{C}$ ranges in $\mathcal{C}$, $\hat{S}$ ranges in $\mathcal{S}$, $\hat{H}$ ranges in $\mathcal{H}$, and $\hat{L}$ ranges in $\mathcal{L}$. The decision problem that the MIP solves is to decide whether there is at least one solution for each rectangular cell.

We construct a MIP formulation of the decision problem. We use the following notations:

- V: the set of all nodes.
- E: the set of all edges (dams).
- s: the root of the tree (also the mouth of the river network).
- $e = (u, v)$: u is the node downstream of edge e and v is the node upstream of e.
- c_v: the total connectivity of river segments at node v.
- s_v: the total sediment produced at node v.
- h_e: hydropower of dam e.
- l_e: seismic risk level of dam e.
- p_e: percentage of sediment trapped by dam e if the dam is built.
- π: the set of dams we plan to build.
- π_e: indicator variable of MIP. It evaluates to 1 iff $e \in \pi$.
- n_v: indicator variable of MIP. It evaluates to 1 iff node v can be reached from the river mouth without passing a dam.
- y_v: continuous variable in MIP. The percentage of the sediment produced at the node v not trapped by dams.

Here we show the MIP encoding considering the following objectives:

Longitudinal Connectivity (C). To define connectivity we assign a binary variable $n_v \in \{0, 1\}$ to each node v, which represents whether a node can be reached from the river mouth without passing a built dam. Therefore, the longitudinal connectivity can be defined as

$$C = \sum_{v \in V} n_v c_v.$$

Suppose s is the river mouth. For each node $v \neq s$, $n_v = 1$ if and only if at least one downstream node u satisfies $n_u = 1$ and the dam $e = (u, v)$ is not built ($\pi_e = 0$). This fact can be encoded using the following constraints:

$$n_s = 1; n_u \geq n_v, \forall (u, v) \in E$$
$$n_v \leq 1 - \pi_{u,v}, \forall (u, v) \in E$$
$$n_v \geq n_u - \pi_{u,v}, \forall (u, v) \in E$$

Sediment (S). For each node v, we define a continuous variable $y_v \in [0, 1]$ that represents the percentage of the sediment produced at the node that is not trapped by downstream dams. At river mouth s, $y_s = 1$. Also, $y_v = \begin{cases} y_u & \text{if } \pi(i) = 0, \\ (1 - p_e)y_u & \text{otherwise.} \end{cases}$ This condition can be linearized using the big M method. Since $y_v \in [0, 1]$ for all $v \in V$, we can replace the condition with

$$y_v \leq y_u \text{ and } y_v \geq (1 - p_e)y_u, \forall (u, v) \in E$$
$$y_v \leq (1 - p_e)y_u + (1 - \pi_e), \forall e = (u, v) \in E$$
$$y_v \geq y_u - \pi_e, \forall e = (u, v) \in E.$$

Minimize: d

subject to: $d = 0$ (d is a dummy variable)

$$\hat{C} \leq C \leq (1 + \epsilon)\hat{C};\ \hat{S} \leq S \leq (1 + \epsilon)\hat{S}$$
$$\hat{H} \leq H \leq (1 + \epsilon)\hat{H}; \hat{L} \leq L \leq (1 + \epsilon)\hat{L}$$
$$C = \sum_{v \in V} n_v c_v;\ L = \sum_{e \in E, l_e > 1} (h_e + F)\eta^{l_e}$$
$$H = \sum_{e \in E} \pi_e h_e;\ S = \sum_{v \in V} y_v s_v$$
$$n_v \in \{0, 1\}, \forall v \in V; \pi_e \in \{0, 1\}, \forall e \in E$$
$$n_s = 1; n_u \geq n_v, \forall (u, v) \in E$$
$$n_v \leq 1 - \pi_{u,v}, \forall (u, v) \in E$$
$$n_v \geq n_u - \pi_{u,v}, \forall (u, v) \in E$$
$$y_s = 1$$
$$y_v \leq y_u \text{ and } y_v \geq (1 - p_e)y_u, \forall (u, v) \in E$$
$$y_v \leq (1 - p_e)y_u + (1 - \pi_e), \forall e = (u, v) \in E$$
$$y_v \geq y_u - \pi_e, \forall e = (u, v) \in E$$

Figure 4: MIP encoding to decide if there is a valid solution in the rectangular region $[\hat{C}, (1 + \epsilon)\hat{C}] \times [\hat{S}, (1 + \epsilon)\hat{S}] \times [\hat{H}, (1 + \epsilon)\hat{H}] \times [\hat{L}, (1 + \epsilon)\hat{L}]$.

MIP solvers like CPLEX can also linearize the constraint automatically. The total sediment S is $\sum_{v \in V} y_v s_v$.

Hydropower (H). The energy output is encoded as $H = \sum_{e \in E} \pi_e h_e$.

Seismic Risk (L). L can be encoded as $L = \sum_{e \in E, l_e > 1} (h_e + F)\eta^{l_e}$. (F is a constant. See earlier description of seismic risk.)

Summarizing the constraints above, we obtain a compact MIP encoding of the problem shown in Figure 4. Because the MIP solves a decision problem, we minimize a dummy variable d. We run the MIP to see if there is a solution in every rectangular cell and we remove dominated solutions.

5 Experimental Results

Data: In order to test our methodological approach at different spatial scales, we developed three sub-sets of data that correspond to different geographical areas within the Amazon region: the western Amazon basin, the Marañon basin, and the entire Amazon basin (Finer and Jenkins 2012; Shedlock et al. 2000; Venticinque et al. 2016; Winemiller et al. 2016; Zarfl et al. 2015). See figure 1. We use the Amazon river network for the tree-structured graph and corresponding values for connectivity, sediment, and seismic risk, as described in section 2. The number of dams (proposed, planned or built) for the entire Amazon basin is 467; 108 for the Marañon basin; and 217 for the western Amazon basin. The entire Amazon basin has more than four million (4083059) river segments , whereas the western Amazon,

ϵ	Orig. Tree (sec.)	Bin. Tree (sec.)	Orig. Tree: Num. Policies	Bin. Tree: Num. Policies
0	$1day+$	2.75	N/A	363081
0.001	$1day+$	0.08	N/A	39633
0.0025	17500.32	0.04	3910567437	22872
0.005	7817.56	0.03	1725983274	15943
0.01	1546.82	0.02	320738784	10594
0.025	83.97	0.01	17710130	6119
0.05	5.09	0.01	1146922	3921
0.1	2.91	0.01	609127	8255
0.25	0.23	0.01	50288	4130

Table 1: The equivalent DP binary tree (Western Amazon) leads to considerable speed-ups: it considers many fewer policies than the original tree due to early pruning.

B	Criteria	ϵ	DP (sec)	MIP (sec)	DP #Sol	MIP #Sol
M	S_dE	0	1313	N/A	24575	N/A
M	S_dE	0.001	1.63	$2d+$	297	–
M	S_dE	0.1	0.01	14.81	8	20
M	CES_dS_s	0.02	46798	$2d+$	28251	–
M	CES_dS_s	0.25	6.26	14.9	276	147
WA	S_sE	0.001	804.3	$2d+$	12640	–
WA	S_sE	0.0025	284.17	2138	7887	1198
A	CE	0	17170	N/A	38459	N/A
A	CE	0.001	72.97	6432.9	3095	342
A	CS_sE	0.01	$2d+$	13856	–	21959
A	CS_sE	0.1	6109	16.8	19955	257
A	CS_sE	0.2	186.67	3.24	3521	77
A	CS_sE	0.25	58.93	1.96	1976	52

Table 2: Examples of runtimes and number of solutions: Marañon (M; 109 nodes), Western Amazon (WA; 219 nodes) and Amazon (A; 468 nodes) showing the trade-offs between the different methods. (S_d - sediment; S_s seismic)

and Marañon basin have 455156, and 128801 river segments respectively. As described in section 2, we generate a directed rooted tree that collapses contiguous regions of the network without dams into a hypernode and associates with each node the corresponding values of connectivity, sediment, and seismic risk. The resulting collapsed trees are as follows: 1) Amazon basin: 468 hypernodes and 467 edges (dams). 2) Western Amazon: 219 hypernodes and 218 edges. 3) Marañon: 109 hypernodes and 108 edges.

Transforming DP original tree into binary tree: To improve the efficiency of the DP algorithm, we first convert the original directed rooted tree into an equivalent binary tree. Given a directed rooted tree, we transform each node u with more than two children in to a binary subtree equivalent, by creating additional intermediary nodes and link the children of u to these intermediary nodes (and u). The newly added edges between the u and intermediate nodes are treated as

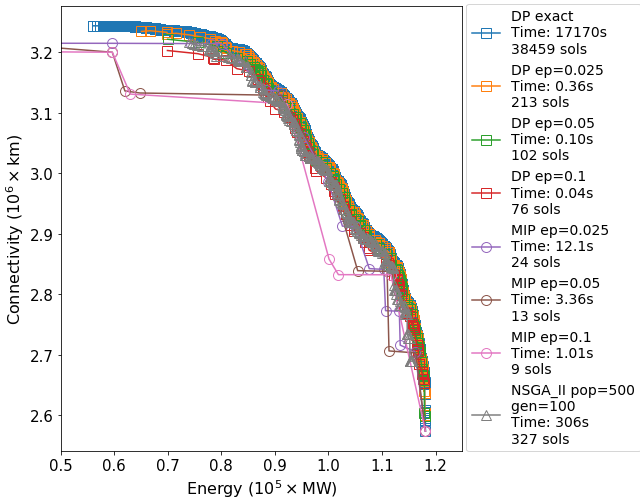


Figure 5: Approximation quality: The DP approximations (for all considered ϵ) nearly overlap the exact Pareto curve (blue). The MIP approximations lie farther from the exact Pareto-Frontier than the corresponding DP approximations do. The DP approximations also better capture the shape and coverage of the exact Pareto frontier.

special non-dam edges. The algorithm never builds a dam on these edges. We also propose a scheme to maintain the objective value information of the original tree. Node rewards such as connectivity and sediments are split evenly among the original root u and intermediary nodes. Edge rewards such as energy and seismic risk are moved to their parent node in the binary tree. The details of this transformation can be found in the full version of the paper (Wu et al. 2018). Overall, our construction is such that dam placements and resulting objective values are in a one-to-one correspondents with those of the original tree.

Transforming to binary tree representation allows us to apply early pruning and scale to large instances. If we apply the DP approach to the original non-binary tree the calculation can quickly become infeasible. Consider a node u with k children, where each child has l Pareto optimal partial policies. In the worst case, when enumerating each group of partial policies, we would have to consider $l^k 2^k$ new partial policies, when considering building or not building each dam. For example, the root node in the full Amazon basin has $k = 44$; therefore, even for very small values of l, running DP becomes infeasible. Converting the original tree into a binary tree ensures that at each node the partial policies of at most two children are considered for pruning. As a result, the binary tree leads to a dramatic increase in efficiency by allowing for earlier and more frequent pruning of partial solutions, which is critical to try to ensure that the set of Pareto optimal partial policies remains of reasonable size throughout the computation. See table 1.

Exact Pareto frontier: The DP approach can compute the exact Pareto Frontier (time permitting), which is infeasible for the MIP approach. We were surprised to see that the DP approach can compute the *complete exact* Pareto frontier, for

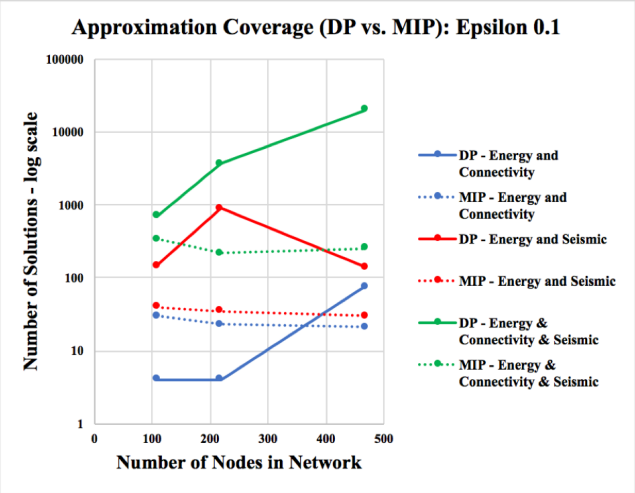


Figure 6: Compared to the MIP approximations (dashed lines), the larger number of solutions produced by the DP approximations (full lines) produce a more complete coverage of the exact Pareto frontier.

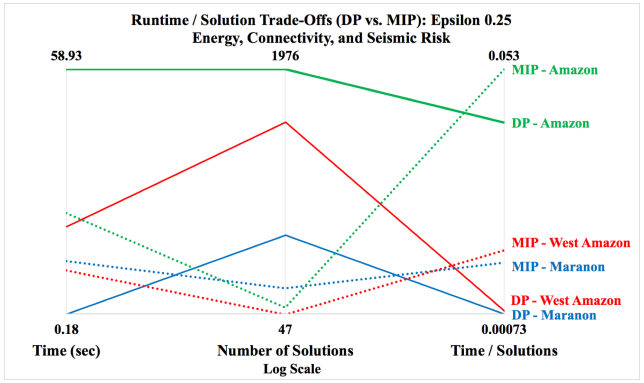


Figure 7: DP and MIP Complementary Strengths: The DP samples the true frontier much more completely than the MIP does (see num. solutions). As a result, the DP runtime depends greatly on the number of true Pareto optimal solutions and network size; in contrast, the MIP shows comparable runtimes and number of solutions across different networks, despite its worst case complexity. The average solution time nicely captures this trade-off: each approach takes different paths to produce a similar avg. time per solution.

connectivity and energy, for the entire Amazon basin, which contains 38459 solutions, in around 5 hours (17170 secs). (See figure 5.)

Quality of the Pareto Frontier approximation: We observe that, for the same $\epsilon > 0$, the accuracy of the DP approximation is in general better than the MIP approximation, in practice. This is illustrated in figure 5.

Approximation Coverage of the Pareto Frontier: The DP approximation has a substantially better coverage of the exact Pareto frontier, compared to the MIP approximation, which is reflected in the larger number of solutions that it produces. See figures 5 and 6.

Evolutionary algorithms have been widely used to approximate (without approximation guarantees) Pareto frontiers (see e.g, (Neumann 2007; Qian, Yu, and Zhou 2013; 2015; Qian, Tang, and Zhou 2016; Wiecek et al. 2008; Märtens and Izzo 2013)). Here we include a comparison to the Non-dominated Sorting Genetic Algorithm (NSGA-II). From figure 5, we see that NSGA-II is comparable in accuracy to the DP approximations with $\epsilon = 0.1$ but takes longer to compute (and without approximation guarantees).

Scalability: The complementary strengths of our approximation paradigms are captured in Figure 7 and Table2, which includes a small sample of the runtimes and number of solutions for the DP and MIP approximations, for the three different networks that we considered: Marañon; Western Amazon; and entire Amazon basin. The MIP runtime is largely determined by the number of cells in the hypercube that need to be considered for feasibility, which is determined by ϵ, the size of network, and number of criteria. Somewhat surprisingly, the size of the network does not appear to affect significantly the MIP feasibility checks. This phenomenon may be explained by the fact that most feasibility checks are either in under or over constrained areas, which are relatively easy to solve. Only a few cells are critically constrained. Our DP approximation, provides a good coverage of the solutions on the Pareto frontier. In contrast to the MIP, the DP approximation is largely dependent on the number of true Pareto optimal solutions, which generally scales with the size of the network and of course the number of criteria. Interestingly, the DP scales better than the MIP for the sediment objective, while the seismic objective is more challenging for DP than for MIP.

Visualization of the Pareto Frontier: For policy makers to make informed decisions based on the approximate Pareto frontier, we are developing a variety of analytical and visualization tools that allow for an interactive exploration of the frontier. The tools allow policy makers to explore the trade-offs among the various solutions. See figure 2.

6 Conclusions

We introduced a DP-based fully polynomial-time approximation scheme for computing a polynomially succinct curve that approximates the Pareto frontier to within an arbitrarily small $\epsilon > 0$ on tree-structured networks. Given a set of objectives, our fully polynomial approximation scheme runs in time polynomial in the size of the instance and $1/\epsilon$. We also introduced a MIP-based scheme to approximate the Pareto frontier. Our work is motivated by a problem in computational sustainability concerning the placement of hydropower dams throughout the Amazon basin. The DP and MIP Pareto frontier approximations have complementary strengths and are surprisingly effective, scaling up to the entire Amazon basin (500 potential dam sites). For example, our DP can generate an approximate Pareto frontier that is within 10% of optimal and contains 19955 non-dominated solutions in roughly 1.7 hours, when optimizing for energy, connectivity, and seismic risk. A within 25% from optimal frontier can be generated in around 1 min-

utes and contains 1976 solutions. The MIP approach is even faster (16 seconds, within 10% from optimal) but provides less coverage of the frontier in terms of number of solutions (257 non-dominated solutions). Our overall goal is to support decision-makers in evaluating impacted ecosystem services on the full scale of the Amazon basin, which is unreachable for other methods. Moreover, our work is general and can be applied to approximate the Pareto frontier of a variety of multiobjective problems, e.g., those concerning finite Markov Decision Processes.

7 Acknowledgements

The authors thank the anonymous reviewers. This research was partially supported by Cornell University's David R. Atkinson Center for a Sustainable Future (ACSF) and by the National Science Foundation (CCF- 1522054 and CNS-1059284).

References

Dechter, R. 1998. Bucket elimination: A unifying framework for probabilistic inference. In *Learning in Graphical Models*, 75–104.

Ehrgott, M., and Gandibleux, X. 2000. A survey and annotated bibliography of multiobjective combinatorial optimization. *Or Spectrum* 22(4):425–460.

Finer, M., and Jenkins, C. N. 2012. Proliferation of hydroelectric dams in the Andean Amazon and implications for Andes-Amazon connectivity. *Plos one* 7(4):e35126.

Gomes, C. P. 2009. Computational sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge* 39(4):5–13.

Kareiva, P. M. 2012. Dam choices: analyses for multiple needs. *Proceedings of the National Academy of Sciences* 109(15):5553–5554.

Märtens, M., and Izzo, D. 2013. The asynchronous island model and nsga-ii: study of a new migration operator and its performance. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, GECCO'13, 1173–1180.

Neumann, F. 2007. Expected runtimes of a simple evolutionary algorithm for the multi-objective m inimum spanning tree problem. *European Journal of Operational Research* 181(3):1620 – 1629.

Papadimitriou, C. H., and Yannakakis, M. 2000. On the approximability of trade-offs and optimal access of web sources. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, FOCS '00.

Qian, C.; Tang, K.; and Zhou, Z.-H. 2016. Selection hyperheuristics can provably be helpful in evolutionary multi-objective optimization. In *International Conference on Parallel Problem Solving from Nature*, 835–846. Springer.

Qian, C.; Yu, Y.; and Zhou, Z.-H. 2013. An analysis on recombination in multi-objective evolutionary optimization. *Artificial Intelligence* 204(Supplement C):99 – 119.

Qian, C.; Yu, Y.; and Zhou, Z.-H. 2015. Pareto ensemble pruning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligenc e*, AAAI'15, 2935–2941.

Shedlock, K. M.; Giardini, D.; Grunthal, G.; and Zhang, P. 2000. The gshap global seismic hazard map. *Seismological Research Letters* 71(6):679–686.

Terra-Neves, M.; Lynce, I.; and Manquinho, V. 2017. Introducing Pareto minimal correction subsets. In *International Conference on Theory and Applications of Satisfiability Testing*, 195–211. Springer.

Venticinque, E.; Forsberg, B.; Barthem, R.; Petry, P.; Hess, L.; Mercado, A.; Cañas, C.; Montoya, M.; Durigan, C.; and Goulding, M. 2016. An explicit gis-based river basin framework for aquatic ecosystem conservation in the amazon. *Earth System Science Data* 8(2):651.

Wiecek, M. M.; Ehrgott, M.; Fadel, G.; and Figueira, J. R. 2008. Multiple criteria decision making for engineering.

Williamson, D. P., and Shmoys, D. B. 2011. *The design of approximation algorithms*. Cambridge University Press.

Winemiller, K.; McIntyre, P.; Castello, L.; Fluet-Chouinard, E.; Giarrizzo, T.; Nam, S.; Baird, I.; Darwall, W.; Lujan, N.; Harrison, I.; et al. 2016. Balancing hydropower and biodiversity in the amazon, congo, and mekong. *Science* 351(6269):128–129.

Wu, X.; Gomes-Selman, J.; Shi, Q.; Xue, Y.; García-Villacorta, R.; Anderson, E.; Sethi, S.; Steinschneider, S.; Flecker, A.; and Gomes, C. P. 2018. Efficiently approximating the Pareto frontier: Hydropower dam placement in the Amazon basin (full version, forthcoming).

Wu, X.; Sheldon, D.; and Zilberstein, S. 2014a. Rounded dynamic programming for tree-structured stochastic network design. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, 479–485.

Wu, X.; Sheldon, D.; and Zilberstein, S. 2014b. Stochastic network design in bidirected trees. In *Advances in Neural Information Processing Systems*, 882–890.

Zarfl, C.; Lumsdon, A. E.; Berlekamp, J.; Tydecks, L.; and Tockner, K. 2015. A global boom in hydropower dam construction. *Aquatic Sciences* 77(1):161–170.

Ziv, G.; Baran, E.; Nam, S.; Rodríguez-Iturbe, I.; and Levin, S. A. 2012. Trading-off fish biodiversity, food security, and hydropower in the Mekong river basin. *Proceedings of the National Academy of Sciences* 109(15):5609–5614.