Extremal Optimization

Arjun Goswami and Zeqian Li
(Dated: December 11, 2019)

This report is for the term project of Phys 598: Statistical Data Analysis and Stochastic Processes in Physics, taught in the Fall 2019 semester at the University of Illinois at Urbana-Champaign by Prof. Jun Song. Here, we briefly review the theoretical foundations of extremal optimization, with an emphasis on the physically- and biologically-inspired origins of this local search heuristic. We also compare the technique and the utility of the algorithm against other local search heuristics over some popular use-cases. Applications across a diversity of domains are discussed. We also apply EO on two NP-hard problems, the minimum feedback arc problem and the tensor CP-Decomposition problem, and discussed EO's performance.

CONTENTS

I. INTRODUCTION

Extremal optimization (EO) [1][2] is a local search heuristic that was designed for problems in combinatorial optimization, involving a discrete search space as the domain. EO is local in the sense that the heuristic searches through the space of candidate solutions by making local changes to a given solution. It is a heuristic in the sense that it is applicable to problems where classical or analytical methods may fail, being too slow or intractable. The method was inspired by a problem from the biological sciences: the Bak-Sneppen model of co-evolution, which exhibits self-organized criticality (SOC). In this way, EO is an important member of the category that consists of other biologically- and physically- inspired search heuristics, such as genetic algorithms (GA), simulated annealing (SA), and particle swarm optimization (PSO). In particular, in much the same way as SA applies to equilibrium statistical physics, EO applies to non-equilibrium processes. How this is the case becomes clear when considering the inherently non-equilibrium nature of self-organized criticality, which can be said to lie at the foundation of the EO method.

I.1. Self-Organized Criticality

In nature (and in general), it is seen that many slowly-driven dynamical systems with a large number of degrees of freedom exhibit fractal-like structure and spatial and temporal scaling laws [3]. Mountainous landscapes, coastal lines along beaches, and sand dunes in the desert are some dramatic examples, but the systems need not be limited to actual physical structures: any system with the necessary ingredients will do (such as biological evolution, as we see next). These dynamical (time-dependent), large systems naturally evolve into a critical state, without the need to tune any external parameter. This is in contrast to equilibrium statistical physics where temperature is the external parameter, tuned according to some cooling schedule, by which the system reaches criticality. Those dynamical systems that naturally evolve into criticality are seen to have a critical point as an attractor, which is the definition of self-organized criticality (SOC). Systems that exhibit SOC share a number of essential features. The dynamics is nonlinear and is characterized by intermittent, barely stable critical periods described by fractal-like self-similarity and scaling power laws, interspersed with periods of rapid change ("avalanches") that can best be described as broadly distributed, long-wavelength fluctuations. The conjunction of these barely stable long periods with short periods of change is known as punctuated equilibrium. Using a minimal model of sandpile evolution [3] [4], authors Bak, Tang, and Wiesenfeld ("BTW") are able to numerically

demonstrate these essential and general features of SOC-exhibiting systems. Briefly, a pile of sand is formed by dropping a single grain of sand at a fixed location at every given timestep (this is the slow driving). The degrees of freedom are the grains of sand that build up over time, and the constitutive law of the system is that a column of sand must collapse into a mound after reaching a certain minimum height with respect to neighboring columns. Analysis of this system reveals literal avalanches of updates, and the stable intermittent periods show criticality. Another simple model we discuss here that exhibts SOC is the Bak-Sneppen model of biological co-evolution.

I.2. The Bak-Sneppen Model of Co-Evolution

In the Bak-Sneppen model, N lattice sites on a line with periodic boundary conditions (a ring) are considered. More generally, any lattice structure and boundary conditions can be considered. Each individual lattice site can be thought of as a given biological species, and the nearest neighbors of a given lattice site correspond to the species that are interrelated to the species on that given site. Each site (species) is assigned a fitness value between 0 and 1, intended to be analogous to biological (Darwinian) fitness. At each time step, the species having the smallest fitness value is updated randomly, a new fitness being chosen from Uniform[0,1]. Biologically speaking, the species are interdependent so the nearest-neighbor sites are also affected, with their fitness also updated randomly, drawn from Uniform[0,1]. The dynamics of this model represent species co-evolution, and the competition leading to the replacement of special fitness is the slow driving. A most interesting emergent model feature that is also seen in actual biological evolution is punctuated equilibrium, where long time periods of relative stasis are interrupted, or punctuated, by much shorter periods of drastic change. Note that only the low fitness of a lattice site's nearest neighbors can cause a random update to be applied to that site's own fitness. Thus, the long time periods can be roughly or qualitatively described as when regions of sites become defined where all the sites of a given region attain maximal fitness values that are all close to each other. However, so-called avalanches, or chain reactions, or large-wavelength fluctuations similar to those found in non-equilibrium statistical physics, can also occur, causing the species lattice to escape its current optimal or critical intermittent configuration. Thus, local optima can be efficiently escaped, the search space is explored without much bias and with a wide reach. The simple model's rich qualitative behavior, reminiscent of some of the features seen in real Darwinian evolution (punctuated equilibrium) and non-equilibrium statistical physics (self-organized criticality) is a brilliant example of how complexity can emerge from a system's simple underlying laws.

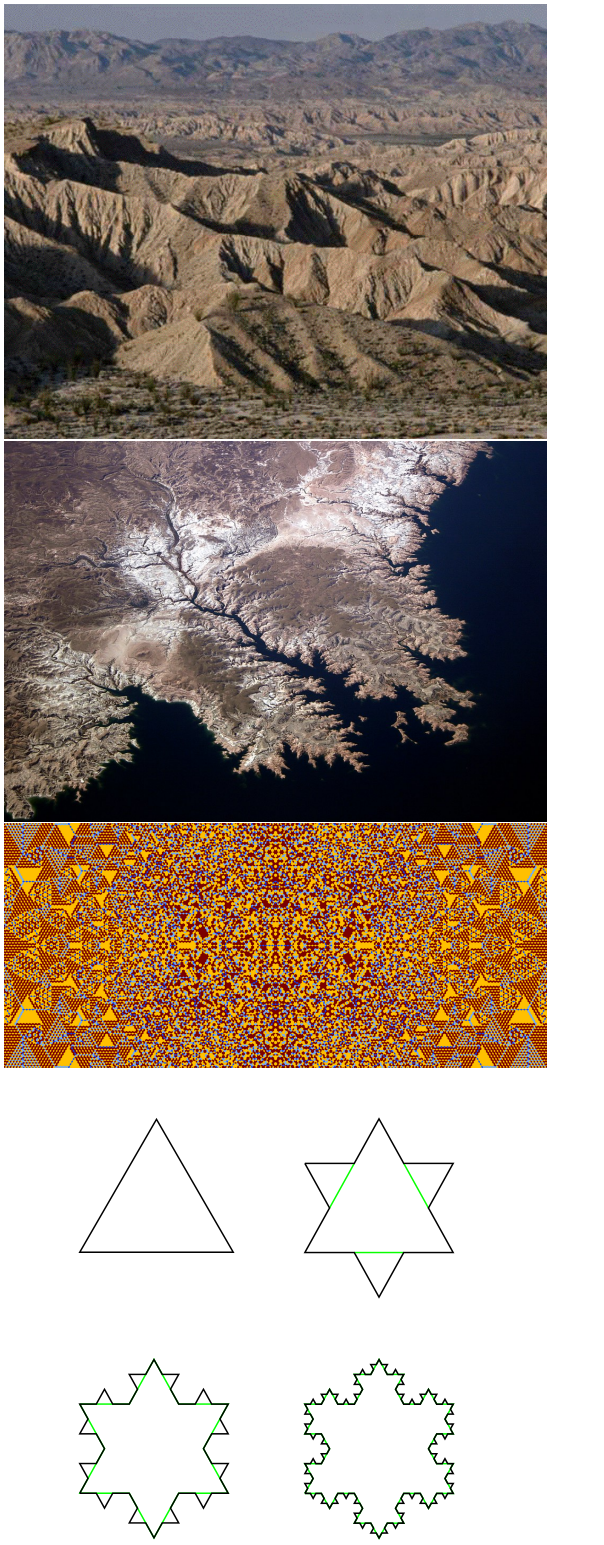

FIG. 1. (Top 3) Examples of slowly-driven dynamical systems with a large number of degrees of freedom: a mountain range subject to deposition and erosion, a coastline subject to water erosion, a and a sandpile fractal. (Lower) Minimal example of a fractal structure and geometric self-similarity depicted a range of four length scales, the famous Koch snowflake.

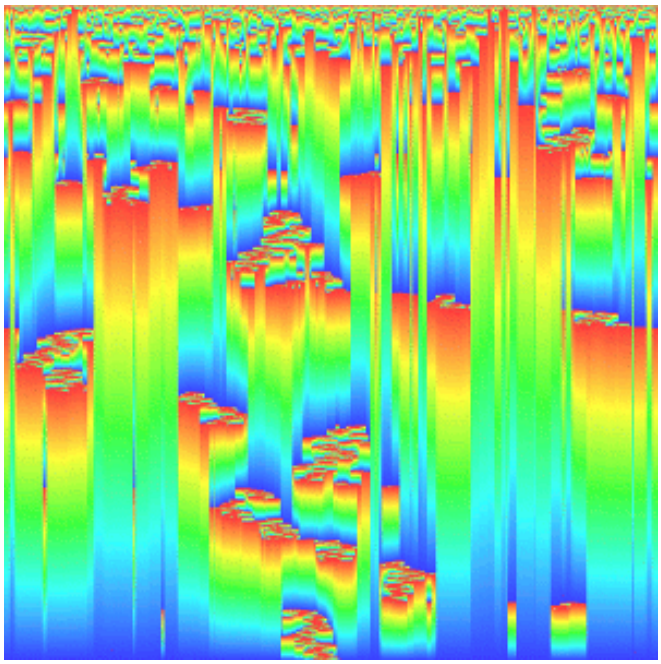

FIG. 2. Time-evolution of the Bak-Sneppen model. The x-axis is the population status (1D lattice of species). The y-axis is time. Each discontinuity is an evolution (replacement of a species, or random update of a species' fitness value), and the color is the species age for a given species, apparently normalized to be red at birth and blue at death. Punctuated equilibrium is clearly evident.

II. THE EXTREMAL OPTIMIZATION ALGORITHM

II.1. Motivation

The intuition behind EO, similar to the Bak-Sneppen model, is to update only those variables that most (extremally) contribute to the cost function, leaving the others unchanged. Moreover, these values are replaced at random without explicitly attempting to improve the cost function. The behavior leads to punctuated-equilibrium-like evolution and allows efficient exploration of the search space and overcoming of local minima.

The idea is to express the cost function $C(S)$ of a given configuration S as a function of the local degrees of freedom x_i that make up the configuration:

$$C(S) = -\sum_{i=1}^{n} \lambda_i \tag{1}$$

Here λ_i is the so-called "fitness" of each local degree of freedom x_i. The smaller or more negative each λ_i is, the more unfit that local degree of freedom is, and the more prone it is to be replaced by a random value under the EO algorithm. Let $N(S)$ denote the set of all configurations neighboring S (such that S and any $S' \in N(S)$ differ only by local change). We can write the most general pseudo-code for the algorithm as follows:

II.2. Steps

1. Initialize configuration S. Set $S^* = S$.
2. For the current configuration S:
   (a) Evaluate λ_i for each x_i
   (b) Find $j = \arg\min_i \lambda_i$
   (c) Choose $S' \in N(S)$ such that x_j changes
   (d) Set $S = S'$
   (e) If $C(S) < C(S^*)$ set $S^* = S$.
3. Repeat above for as long as desired
4. Return S^* and $C(S^*)$

II.3. τ-EO

There is one noticeable class of problems in the above pseudo-code, involving the possibility of getting stuck in the loop. There are situations where always picking the worst degree of freedom (with the least fitness) and updating S to S' leads the algorithm to deterministic or dead-end behavior. To avoid getting stuck in these local minima or closed or deterministic paths in the search space, stochasticity can be implemented in the algorithm by introducing a new parameter τ. Instead of choosing the worst x_i we consider a probability distribution over the x_i as follows. Rank the x_i according to their fitness λ_i. The rank of the x_i with the highest λ_i is n, the rank k of the x_i with the lowest λ_i is 1, and so on with the rest of the x_i in between. Define $P_k \propto k^{-\tau}$ for some value of the parameter τ. In the pseudo-code, change step 2(c) so that a neighboring configuration is chosen with local degree of freedom changed according to this probability distribution. The higher τ is, the more unlikely it is to stochastically choose the degrees of the better fitness to alter. In the $\tau \to \infty$ limit, τ-EO just becomes regular EO since the probability of choosing the high-fitness degrees of freedom will be completely damped out. In the $\tau \to 0$ limit, the probability distribution is uniform and extremal optimization just becomes optimization by randomly locally searching through the search space (effectively just a local random walk) without any regard to which degrees of freedom contribute the least to the cost function. The asymptotic choice of $\tau = 1 + 1/\log n$ optimizes the performance of the τ-EO algorithm [1].

II.4. Comparison to other search heuristics

It should now be clear that extremal optimization (EO) is directly inspired by self-organized criticality (SOC).

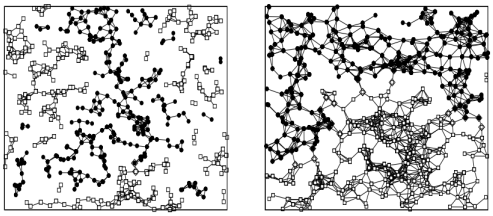

Figure 1. Two random geometric graphs, $n = 500$, with connectivity 4 (top) and connectivity 8 (bottom) in an optimized configuration found by EO. At $\alpha = 4$ the graph barely "percolates," with merely one "bad" edge (between points of opposite sets, masked by diamonds) connecting a set of 250 round points with a set of 250 square points. For the denser graph on the bottom, EO reduced the cutsize to 13. A 1-*exchange* will turn a square vertex into a round one, and a round vertex into a square one.

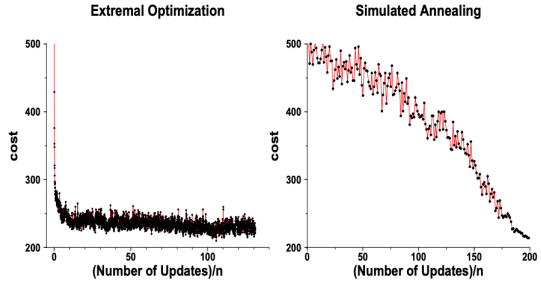

Figure 2. Evolution of the cost function $C(S)$ during a typical run of EO (left) and SA (right) for the bipartitioning of an $n = 500$-vertex graph G_{500} introduced in Ref. [38]. The best cost ever found for G_{500} is 206. In contrast to SA, which has large fluctuations in early stages of the run and then converges much later, extremal optimization quickly approaches a stage where broadly distributed fluctuations allow it to probe and escape many local minima.

FIG. 3. Figure taken from the original paper [1]

The heuristic mimics the convergence of real systems to optimal configurations by driving the extremal contributors to the cost function to be updated randomly. The walk through configuration space resembles the dynamical behavior of SOC-exhibiting systems: as we see in step 2(d) of the pseudo-code, the configuration is updated unconditionally, without any criterion for acceptance or rejection as in the case of other local search heuristics.

Among the other physically-inspired search heuristics, genetic algorithms (GAs)are also motivated by biological evolution, but in reality the methods of EO and GA are very different, with deceptively similar terminology [1]. GA search is not local, but cross-over operators "breed" a pair of candidate configurations in a global manner.

As stated previously, SA mimics physical behavior by approaching thermal equilibrium via a cooling schedule, where fluctuations that are designed to escape local minima decrease at late runtimes. In contrast, random fluctuations for EO can continue at late runtimes. EO selects against the worst fitnesses, but SA functions through a Metropolis walk [1]. The physical intutions are quite distinct but both algorithms are effective, and in some domains EO is more effective (see Fig. 3).

III. SOME RESULTS FOR WELL-KNOWN PROBLEMS

The EO algorithm has been applied by [1] to problems in graph bipartitioning, graph coloring, and spin glasses,

Table 2. EO approximations to the average ground-state energy per spin $e(n)$ of the $\pm J$ spin glass in $d = 3$, compared with GA results from Refs. [50, 32]. For each size $n = L^3$ we have studied a large number I of instances. Also shown is the average time t (in seconds) needed for EO to find the presumed ground state on a 450MHz Pentium. (As for a normal distribution, for increasing n fewer instances are needed to obtain similar error bars.)

L	I	$e(n)$	t	Ref. [50]	Ref. [32]
3	40100	-1.6712(6)	0.0006	-1.67171(9)	-1.6731(19)
4	40100	-1.7377(3)	0.0071	-1.73749(8)	-1.7370(9)
5	28354	-1.7609(2)	0.0653	-1.76090(12)	-1.7603(8)
6	12937	-1.7712(2)	0.524	-1.77130(12)	-1.7723(7)
7	5936	-1.7764(3)	3.87	-1.77706(17)	
8	1380	-1.7796(5)	22.1	-1.77991(22)	-1.7802(5)
9	837	-1.7822(5)	100.		
10	777	-1.7832(5)	424.	-1.78339(27)	-1.7840(4)
12	30	-1.7857(16)	9720.	-1.78407(121)	-1.7851(4)

FIG. 4. Results for the ground-state energy of the spin-glass, taken from the original paper [1]

where the cost function $C(S)$ can be expressed as a sum of fitnesses of local degrees of freedom. In particular, for a spin glass, we have the Hamiltonian

$$C(S) = -\frac{1}{2}\sum_{<ij>} J_{ij}x_i x_j - \sum_i h_i x_i \tag{2}$$

With the spins $x_i \in -1, 1$. If we define the fitness as

$$\lambda_i = x_i(\frac{1}{2}\sum_j J_{ij}x_j + h_i) \tag{3}$$

Then we can apply EO to find the ground-state energy (see figure 4).

IV. EO IMPLEMENTATION AND RESULTS

We implement EO for two NP-hard problems: the graph minimum feedback arc set problem and the tensor CP-decomposition problem. Note that both problems use modified versions of EO. We compare EO with a simulated annealing (SA) algorithms on the minimum feedback arc set problem, and discuss advantages and disadvantages of EO. All codes are available at [5].

IV.1. Minimum feedback arc set

The directed graph minimum feedback arc set (FAS) problem is defined as follow [6]: given a directed graph $G = (V, E)$, where V and E are sets of vertices and edges, find a minimum set of edges (arc) such that by deleting these edges, the remaining graph is a feed-forward graph. Minimum FAS problem is a NP-hard problem [6], and many exact or approximate algorithms have been proposed [7]. Minimum FAS problem is useful in understanding structures of complex directed network data, such as feedback in gene regulation networks [8].

We apply a generalized extremal optimization (GEO) algorithm on the minimum FAS problem, inspired by [9] and [10]. For comparison, we also implement the SA algorithm proposed in [9]. Both algorithms follow a similar recipe:

For a directed graph $G(V, E)$, label nodes as $(1, 2, ..., n)$. Define a hierarchy $h \in S(n)$ as a permutation of nodes (figure 5). Then h uniquely defines a choice of feedback arc set: for each directed edge $(i \to j)$, if $h(i) > h(j)$, include the edge in the feedback arc set. By deleting all such edges and regarding the hierarchy as the feed-forward direction, the remaining edges are all feed-forward. So, there is a many-to-one map between the space of h and the space possible feedback arc sets.

Both the generalized EO and the SA algorithms search in the space of h through local steps (local permutations) to find an optimal solution minimizing the energy, defined by the number of feedback arcs. Local steps are illustrated in (figure 6): given a feedback arc, one can move the parent node on top of the children node or move the children node under the parent node to eliminate the feedback arc. It may decrease or increase the global energy and the recalculation of energy is efficient due to the locality of the move.

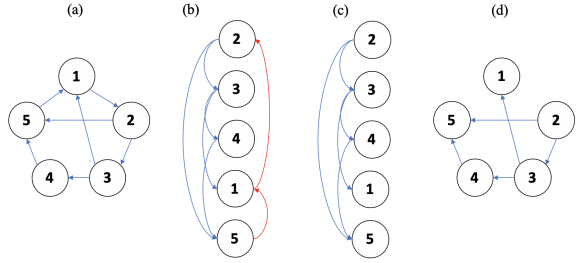

FIG. 5. FAS can be represented by a permutation h. (a) A directed graph. It contains cycles and thus is not feed-forward. (b) A possible h. Red arrows are feedback arcs. (c) Delete feedback arcs. (d) The remaining graph is feed-forward (acyclic).

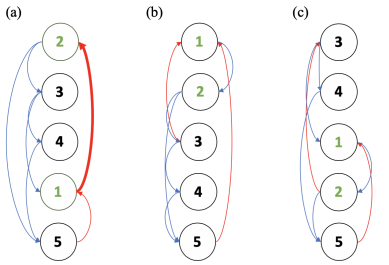

FIG. 6. Local moves are local permutations on h. (a) One select a feedback arc (thick red arrow and green nodes). There are two options: move parent (b) or move child (c). (b) Move parent. In this case $\Delta E = +1$. (c) Move child. In this case $\Delta E = +1$. In general, moving parents and moving children have different energy change.

The generalized EO (GEO) algorithm [10] defines "fitness" of each feedback edge $(i \to j)$ as the energy change after moving i to top of j (or, moving j under i). This steps out of the original EO, which defines individual fitness as it contribution to the global energy. The reason for using GEO is that the original recipe makes little sense for the problem's optimization goal and also shows poor performance (appendix VI.1). The rest of GEO are similar to EO. The recipe is:

1. Initialize h randomly.
2. Select a subset from the feedback arcs set defined by h. Calculated each arc's "fitness", i.e, the energy change caused by moving its parent (or children). Rank k the energy changes from large to small ($k = 1$ for the largest energy change). Select an arc from the subset by probability $p \propto k^{-\tau}$.
3. Move the parent (or child) (in actual implementation, we move parents and children alternatively to diversify moves). Update energy and the feedback arc set. Store h if it's a better solution that the current optima.
4. Repeat 2-3 until a maximum number of iteration is reached.

For comparison, we implement the SA algorithm in [9]:

1. Initialize h randomly.
2. Select an arc randomly from the feedback arc set. Calculated the energy change ΔE caused by moving this arc (move parent or children). If $\Delta E < 0$, apply the move; if $\Delta E > 0$, apply the move by probability $e^{-\beta \Delta E}$. (Similarly, we move parents and children alternatively in each iteration to diversify moves.)
3. Update energy and the feedback arc set. Store h if it's a better solution that the current optima.
4. Increase β following a cooling procedure.
5. Repeat 2-4 until a maximum number of iteration is reached.

We test GEO and SA on two data sets: a Erdős–Rényi random graph ($|V| = 100, |E| = 481$) and the Florida food web network (($|V| = 128, |E| = 2044$) [11] (figure 8). Note that we remove double edges $(i \leftrightarrow j)$ from the data set as they does not affect the FAS results. (appendix IV.1).

For the ER random network, GEO performed significant worse than SA, with $E_{\min} = 124$ and $E_{\min} = 95$. As shown in 8a/b, GEO was worse at exploring the complicated energy landscape compared to SA. For the food web network, where the energy landscape is less complicated, both GEO and SA found an optimal solution $E_{\min} = 6$. Therefore, in certain cases, EO/GEO gets stuck in local minima and cannot fully explore the energy landscape.

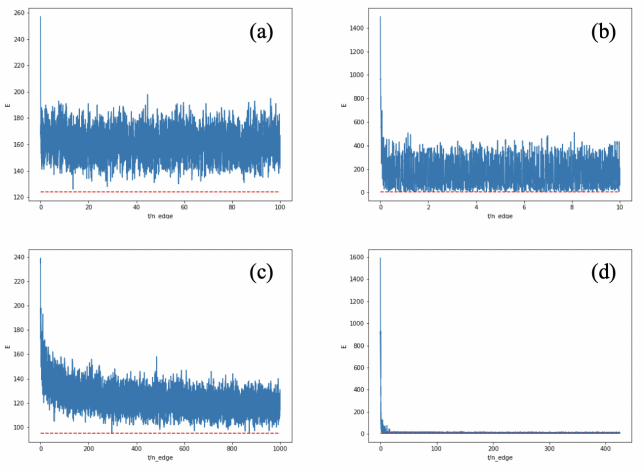

FIG. 7. GEO and SA on the ER graph and the food web network. (a) GEO on the ER network. $E_{\min} = 124$. Parameters: $\tau = 1.4$. (b) GEO on the food web network. $E_{\min} = 6$. Parameters: $\tau = 1.4$. (c) SA on the ER network. $E_{\min} = 95$. Parameters: cooling schedule $\beta = 0.1\log(t)$. (d) SA on the food web network. $E_{\min} = 6$. Parameters: cooling schedule $\beta = 0.1\log(t)$.

IV.2. CP-Decomposition

The tensor rank decomposition problem, or CP-decomposition, is a NP-hard problem[12]. It is significant in understanding mathematical structures of tensors as well as real-world applications [13].

Given a tensor T with order p and dimensions $(I_1, ..., I_p)$ and the maximum rank number K, one search for a set of rank-1 tensors $\{U_k\}$ such that $\sum_{k=1}^{K} U_k \approx T$, i.e., find

$$\begin{aligned}
&\underset{\{U_k\}}{\operatorname{argmin}} E \\
=&\underset{\{U_k\}}{\operatorname{argmin}} |T - \sum_k U_k|_F^2 \\
=&\underset{\substack{\{u_{1k}\}\in R^{I_1} \\ \cdots \\ \{u_{pk}\}\in R^{I_p}}}{\operatorname{argmin}} |T - \sum_k u_{1k} \otimes ... \otimes u_{pk}|_F^2
\end{aligned}$$

where $u_{nk} \in \mathbb{R}^{I_n}$. Rewrite

$$\begin{aligned}
E =&\langle T - \sum_k U_k, T - \sum_k U_k\rangle^2 \\
=&T^2 - 2\sum_k \langle U_k, T\rangle + \sum_{k,l}\langle U_k, U_l\rangle \\
=&T^2 + \sum_k \langle U_k, -2T + \sum_l U_l\rangle
\end{aligned}$$

Define the energy (negative "fitness") of each rank $E_k = \langle U_k, -2T + \sum_l U_l\rangle$, and the original EO recipe follows through. Because the search space is continuous, we apply a modified EO inspired by [14], which restrict the search space from $\mathbb{R}^{K(I_1+...+I_p)}$ to the space of local minima. The full algorithm is:

1. Initialize $\{U_k\}$ randomly. In implementation, we sample vectors from a Gaussian distribution with zero mean.

2. Converge to a local minimum through gradient descent: calculate gradient $g_{k,n,i_n} = \nabla_{u_k^{n,i_n}} E$ (appendix VI.3); concatenate vectors $(\{u_{1k}, ..., u_{pk}\})$ and gradients to 1d vectors v and g; in each iteration, update $v \leftarrow v - \alpha g$, where α is the learning rate; stop until a convergence criteria is satisfied.

3. Calculate $E_k = \langle U_k, -2T + \sum_l U_l\rangle$ for each rank. Rank E_k from large to small, with $r = 1$ for the largest E_k. Select a rank k by probability $p \propto r^{-\tau}$.

4. For the selected k, re-sample $u_{1k}, ..., u_{pk}$. In implementation, we random walk these vectors by steps sampling from a Gaussian distribution with zero mean.

5. Repeat 2-4 until a maximum number of iteration is reached.

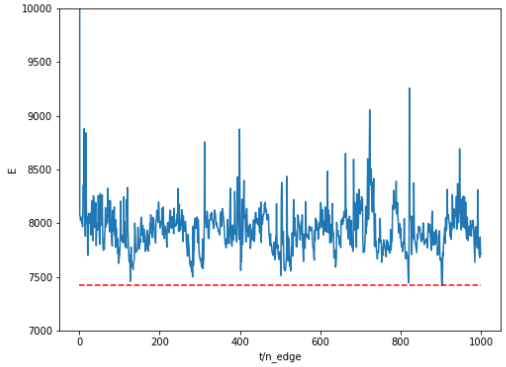

FIG. 8. EO for CP-decomposition. We run EO on a random tensor of dimensions (10,10,10) sampled from Uniform[0, 1]. Parameters: $K = 3$, $\tau = 1.0$.

We test the EO algorithm on a random $(10, 10, 10)$ tensor. EO managed to reduce the energy function and find a rank-3 approximation of the tensor. A more thorough analysis was not completed due to time limit and we will improve it in the future.

V. DISCUSSIONS

In conclusion, we present the theory and physical intuition behind the extremal optimization (EO) algorithm, which originates from the notion of self-organized criticality (SOC). A pseudo-code is also presented and some the results for some popular use cases are shown. The

algorithm is compared with genetic algorithms (GA) and simulated annealing (SA). We implemented EO and SA on two NP-hard problems: the minimum feedback arc set problem and the tensor CP-decomposition problem. We explored EO's performance and compared with simulated annealing. We showed that in certain cases EO gets stuck in local minima

Due to the limited time of the course project, many tasks are left open or unfinished in this paper, and the full potential of EO was not fully explored. If given more time, we will improve this paper in the following ways.

- EO for CP-Decomposition was unfinished due to time limit. We will further explore by better parameter tuning and testing on real-world data.
- A similar SA algorithm for CP-Decomposition is possible yet we are unable to explore due to time restriction.
- Both minimum FAS and CP-decomposition need a more comprehensive statistical analysis to test the variation of EO/SA algorithms. They also need better parameter tuning, testing on larger datasets.
- EO algorithms codes are slow compared to SA, which is better optimized. We will improve computer engineering aspects of the codes and test the algorithms' speed.
- Both EO and SA are easily applicable to many other combinatorial problems, which we wanted but didn't explore due to time limit. Two examples are the traveling salesman problem the minimum feedback vertex problem.

Minimum FAS: need a more careful statistical analysis; need better parameter tuning; need to use on larger dataset; need better parameter tuning; want to study dependence on parameters. Tensor: apply to real work data; study on parameters;

VI. APPENDIX

VI.1. EO and GEO in minimum FAS

Here we explain why the original EO recipe was abandoned for the minimum feedback arc set problem and we adopted GEO.

Original EO requires writing the total energy as a sum of individual components. For minimum FAS problem, we can write

$$E(h) = \sum_{\substack{i,j \\ (i\to j)\in FAS(h)}} 1 = \sum_i \sum_{\substack{j \\ (i\to j)\in FAS(h)}} 1 = \sum_i E_i,$$

i.e, we can define the total energy as sum of individual energy of each node, defined by the number of out-FAS-edges connect by that code (similarly, we can also define individual energy as the number of in-FAS-edges connect to that code). The original recipe makes the following algorithm:

- Initialize h randomly. Find the FAS defined by h.
- Calculate energy of each node, defined by the number of feedback arcs originated (or pointed to) the node. Rank energies from large to small, with $k = 1$ for the largest energy.
- Pick a node by probability $p \propto k^{-\tau}$.
- Reset the node's position and recalculate energy and FAS. One may either
  1. Move it on top of all its children nodes (or under all of its parent nodes), or
  2. Move it on top of a random children node (or under a random parent node)
- Repeat until a maximum number of iteration is reached.

However both method 1 and 2 showed poor performance in the actual implementation. Possible reasons are: method 1 greatly limits the choice of possible local moves; and method 2 does not strongly drive the searching process to smaller energy due to strong randomness.

We designed a third method with the following consideration: in figure 9, moving arc $D \to B$ and arc $G \to E$ should be considered equally good, as they eliminate the same number of feedback arcs. So, we define energy of each arc as: the number of arcs in FAS (not considering feed forward edges) eliminated by moving the arc. However, method 3 also shows poor performance in implementation.

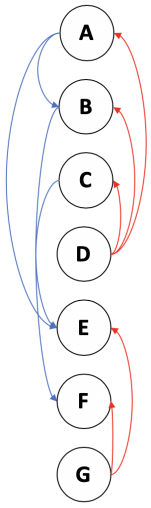

FIG. 9.

The failure of original EO is because defining fitness as a sum of individual fitness makes little sense for the

minimum FAS problem. By adopting GEO and defining the fitness of an arc as the negative global energy change by moving the arc, algorithm performance is significantly improved.

VI.2. Double edges in minimum FAS

Before applying GEO/SA algorithm, we remove double edges from the graph (if both $(i \to j)$ and $(j \to i)$ are present in the graph, we say there is a double edge between i, j, and we remove both $(i \to j)$ and $(j \to i)$).

The reason is, for any h and its FAS, one can always add the double edges back, with half of them are feed-forward arcs and half of them are feedback arcs. So, there is one-to-one correspondence between the solutions for the graph without double edges and the solutions for the graph with double edges. So we remove double edges for efficiency and numerical stability. All results are reported after double edges are removed.

VI.3. Gradient in CP-Decomposition

Let T be a tensor of order p and dimensions $(I_1, ..., I_p)$. To find the K-rank approximation, one minimize

$$E = |T - \sum_{k=1}^{K} u_{1k} \otimes ... \otimes u_{pk}|_F^2$$

where $u_{nk} \in \mathbb{R}^{I_n}$. Then,

$$\begin{aligned} E =& T^2 - 2 \sum_k \sum_{i_1 ... i_p} T^{i_1 ... i_p} u_{1k}^{i_1} ... u_{pk}^{i_p} \\ &+ \sum_k \sum_{i_1 ... i_p} (u_{1k}^{i_1} ... u_{pk}^{i_p})^2 \\ &+ \sum_l \sum_{l \neq k} \sum_{i_1 ... i_p} u_{1k}^{i_1} ... u_{pk}^{i_p} u_{1l}^{i_1} ... u_{pl}^{i_p} \end{aligned}$$

Calculate gradient:

$$\begin{aligned} \nabla_{u_{nk}^{i_n}} E =& -2 \sum_{\backslash i_n} T^{i_1 ... i_p} u_{1k}^{i_1} ... \backslash_n ... u_{pk}^{i_p} \\ &+ 2 \sum_{\backslash i_n} (u_{1k}^{i_1} ... \backslash_n ... u_{pk}^{i_p})^2 u_{nk}^{i_n} \\ &+ 2 \sum_{\backslash i_n} \sum_{l \neq k} u_{1k}^{i_1} ... \backslash_n ... u_{pk}^{i_p} u_{1l}^{i_1} ... u_{pl}^{i_p} \\ =& -2 \sum_{\backslash i_n} u_{1k}^{i_1} ... \backslash_n ... u_{pk}^{i_p} (T - T_{\text{approx}})^{i_1 ... i_p} \\ =& -\frac{2}{u_{ik}^{i_n}} \sum_{\backslash i_n} U_k^{i_1 ... i_p} (T - T_{\text{approx}})^{i_1 ... i_p} \end{aligned}$$

where $\sum_{\backslash i_n} = \sum_{i_1 ... i_{n-1} i_{n+1} ... i_p}$ and $u_{1k}^{i_1} ... \backslash_n ... u_{pk}^{i_p} = u_{1k}^{i_1} ... u_{n-1,k}^{i_{n-1}} u_{n+1,k}^{i_{n+1}} ... u_{pk}^{i_p}$. Using gradient descent can reach a local minimum in the energy function.

VII. CODE AVAILABILITY

All codes are available at `https://github.com/zeqianli/ExtremalOptimization` [5].

VIII. AUTHORS' CONTRIBUTIONS

Arjun Goswami and Zeqian Li contributed equally. AG discusses the theory, motivation, and previously-obtained results for the EO algorithm. ZL implemented EO and SA for the two problems. Both AG and ZL contributed to discussions and wrote the manuscript.

IX. ACKNOWLEDGMENTS

We thank Prof. Jun Song for discussions about the class project.

[1] S. Boettcher and A. G. Percus, Computational Modeling and Problem Solving in the Networked World **21**, 61 (2002).
[2] S. Boettcher and A. G. Percus, Physical Review Letters **86**, 5211 (2001).
[3] T. C. Bak, Per and K. Wiesenfeld, Physical Review Letters **59**, 381 (1987).
[4] V. A. Golyk (2012).
[5] Z. Li, https://github.com/zeqianli/ExtremalOptimization (2019).
[6] R. M. Karp *et al.*, Reducibility among combinatorial problems **23**, 85 (1972).
[7] Y.-Z. Xu and H.-J. Zhou, Journal of Statistical Physics **169**, 187 (2017).
[8] I. Ispolatov and S. Maslov, BMC Bioinformatics **9**, 424 (2008).
[9] J.-H. Zhao and H.-J. Zhou, Chinese Physics B **26**, 078901 (2017).
[10] F. L. De Sousa and F. M. Ramos, in *International Conference on Inverse Problems in Engineering Rio de Janeiro, Brazil* (Citeseer, 2002).
[11] V. Batagelj and A. Mrvar, hup://vladojmJuni-lj.si/pub/networks/data (2006).
[12] C. J. Hillar and L.-H. Lim, Journal of the ACM (JACM) **60**, 45 (2013).
[13] M. Wang, J. Fischer, Y. S. Song, *et al.*, The Annals of Applied Statistics **13**, 1103 (2019).
[14] T. Zhou, W.-J. Bai, L.-J. Cheng, and B.-H. Wang, Physical Review E **72**, 016702 (2005).