Elementary Landscapes: Theory and Applications

L. Darrell Whitley and Andrew M. Sutton

Department of Computer Science
Colorado State University
Fort Collins, CO, USA

GECCO'13 Companion, July 6-10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

Introduction

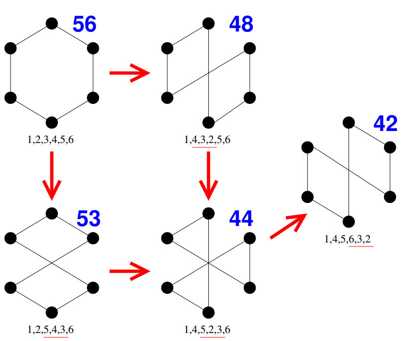
What is a landscape?

- Many different intuitive definitions
- A mathematical formalism of the *search space* of a combinatorial optimization problem

Definition: a landscape is a tuple (X, N, f)

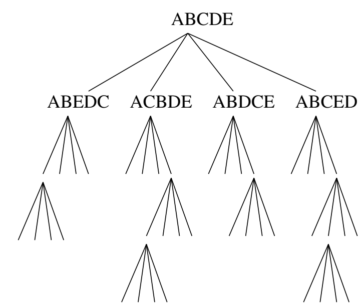
A set of *states*	X
A *neighborhood* operator	$N : X \mapsto \mathcal{P}(X)$
A *fitness* function	$f : X \mapsto \mathbb{R}$

Introduction: a landscape



X	set of *states*
$N : X \mapsto \mathcal{P}(X)$	neighborhood operator
$f : X \mapsto \mathbb{R}$	objective function

Local Search as Tree Search



We often think of local search as greedy depth-first search.
But the neighborhood really induces a connected graph.

Introduction

Landscape: a vertex-weighted graph

- The vertices are points in the search space

What is an "elementary" landscape?

- A fitness function f that has a special relationship with the neighborhood operator N with respect to X
- Elementary = "fundamental component"

Why is this useful?

- Certain "smooth" properties
- Computation of average neighborhood
- Constraints on plateaus, local optima

5

Preliminaries

$$G(X, E)$$

is the underlying graph induced by N.
We assume G is regular with vertices of degree d.

$$\boldsymbol{A} \in \mathbb{R}^{|X| \times |X|}$$

is the *adjacency matrix* of G.
If x_1 and x_2 are neighbors, $A(x_1, x_2) = 1$.

$$\boldsymbol{\Delta} = \boldsymbol{A} - d\boldsymbol{I} \in \mathbb{R}^{|X| \times |X|}$$

is the Laplacian of G.

Note: f is a discrete, finite function, $f \in \mathbb{R}^{|X|}$.

6

The Wave Equation: definition 1

On an arbitrary landscape

- f and N are *unrelated*

On an elementary landscape

The wave equation

$$\boldsymbol{\Delta} f = \lambda f$$

- where λ is a scalar
- In other words, f is an eigenvector of the Laplacian

7

The Wave Equation: definition 1

Average change

$$\boldsymbol{\Delta} f = (\boldsymbol{A} - d\boldsymbol{I})f = k(\bar{f} - f)$$

$$\boldsymbol{\Delta} f(x) = \sum_{y \in N(x)} (f(y) - f(x)) = k(\bar{f} - f(x))$$

Average value

$$\begin{aligned}
\underset{y \in N(x)}{\text{avg}} \{f(y)\} &= \frac{1}{d} \sum_{y \in N(x)} f(y) \\
&= f(x) + \frac{1}{d} \left(\sum_{y \in N(x)} f(y) - f(x) \right) \\
&= f(x) + \frac{1}{d} \Delta f(x) \\
&= f(x) + \frac{k}{d} (\bar{f} - f(x))
\end{aligned}$$

8

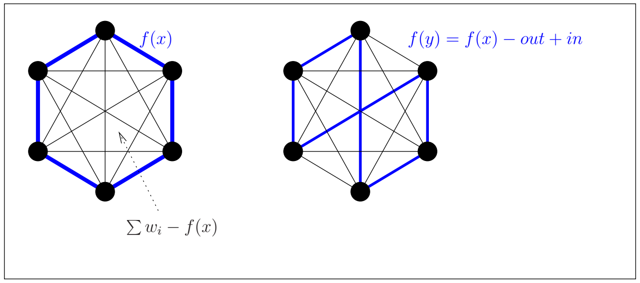
The Wave Equation: definition 2

$$f(x) = \sum \text{ a subset of "components"}$$

Starting from average...

$$\operatorname*{avg}_{y \in N(x)} \{f(y)\} = f(x) + \operatorname*{avg}_{y \in N(x)} \{\text{components in} - \text{components out}\}$$

Example: TSP under 2-opt



- Components: set of edge weights $w_{i,j}$
- $f(x)$ = sum of edge weights induced by tour x
- There are $n(n-1)/2 - n$ weights not in tour x
- Average value of components out: $\frac{2}{n} f(x)$
- Average value of components in: $\frac{2}{n(n-3)/2} \left(\sum w - f(x)\right)$

The Components and $\bar{f}$

Let C denote the set of components

$0 < p_3 < 1$ is the proportion of the components in C that contribute to the cost function for any randomly chosen solution

$$\bar{f} = p_3 \sum_{c \in C} c$$

For the TSP:

$$\bar{f} = \frac{n}{n(n-1)/2} \sum_{w_{i,j} \in C} w_{i,j}$$

$$\bar{f} = \frac{2}{n-1} \sum_{w_{i,j} \in C} w_{i,j}$$

The Wave Equation: definition 2

$$\begin{aligned}
\operatorname*{avg}_{y \in N(x)} \{f(y)\} &= f(x) + \frac{2}{n(n-3)/2} \left(\sum w - f(x)\right) - \frac{2}{n} f(x) \\
&= f(x) + \frac{2}{n(n-3)/2} \left((n-1)/2\bar{f} - f(x)\right) - \frac{2}{n} f(x) \\
&= f(x) + \frac{(n-1)}{n(n-3)/2} (\bar{f} - f(x)) \\
&= f(x) + \frac{k}{d} (\bar{f} - f(x))
\end{aligned}$$

Properties

One of the following is true.

$$f(x) = \operatorname*{avg}_{y \in N(x)} \{f(y)\} = \bar{f}$$

$$f(x) < \operatorname*{avg}_{y \in N(x)} \{f(y)\} < \bar{f}$$

$$f(x) > \operatorname*{avg}_{y \in N(x)} \{f(y)\} > \bar{f}$$

Properties

Assume $f(x) < \bar{f}$. Note that $0 < k/d < 1$.
Then $(\bar{f} - f(x))$ must be positive. Thus

$$\operatorname*{avg}_{y \in N(x)} \{f(y)\} = f(x) + \frac{k}{d}(\bar{f} - f(x)) > f(x)$$

Properties

Assume $f(x) < \bar{f}$. Note that $0 < k/d < 1$.

$$\operatorname*{avg}_{y \in N(x)} \{f(y)\} - f(x) = \frac{k}{d}(\bar{f} - f(x))$$

$$\frac{k}{d}(\operatorname*{avg}_{y \in N(x)} \{f(y)\} - f(x)) < \operatorname*{avg}_{y \in N(x)} \{f(y)\} + f(x) = \frac{k}{d}(\bar{f} - f(x))$$

$$\frac{k}{d}(\operatorname*{avg}_{y \in N(x)} \{f(y)\} - f(x)) < \frac{k}{d}(\bar{f} - f(x))$$

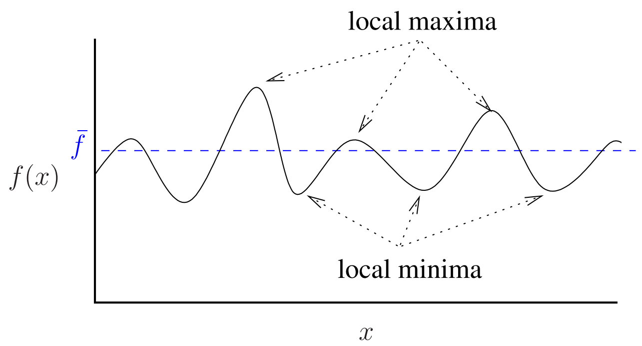
THUS: $f(x) < \operatorname*{avg}_{y \in N(x)} \{f(y)\} < \bar{f}$

Properties

When $f(x) > \bar{f}$ and $0 < k/d < 1$ we can similarly show that:

$$f(x) > \operatorname*{avg}_{y \in N(x)} \{f(y)\} > \bar{f}$$

Properties



A Component Based Model

C is the set of components (e.g. from a cost matrix)

x is a solution (e.g. a subset of the cost matrix)

Let $(C - x)$ denote the set of components, excluding those in x

For a move, we then define:
$0 < p_1 < 1$ is the proportion of components in x that change
$0 < p_2 < 1$ is the proportion of components in $(C - x)$ that change

The Component Theorem

Theorem

If p_1, p_2 and p_3 can be defined for any regular landscape such that the evaluation function can be decomposed into components where $p_1 = \alpha/d$ and $p_2 = \beta/d$ and

$$\bar{f} = p_3 \sum_{c \in C} c = \frac{\beta}{\alpha + \beta} \sum_{c \in C} c$$

then the landscape is elementary.

The Component Theorem

Proof.

$$\begin{aligned}
\operatorname*{avg}_{y \in N(x)} \{f(y)\} &= f(x) - p_1 f(x) + p_2((\sum_{c \in C} c) - f(x)) \\
&= f(x) - p_1 f(x) + p_2((1/p_3 \bar{f}) - f(x)) \\
&= f(x) - (p_1 + p_2) f(x) + (p_2/p_3)\bar{f} \\
&= f(x) - \frac{\alpha + \beta}{d} f(x) + \frac{\beta/d}{\beta/(\alpha+\beta)} \bar{f} \\
&= f(x) + \frac{\alpha + \beta}{d}(\bar{f} - f(x)) \\
&= f(x) + \frac{k}{d}(\bar{f} - f(x))
\end{aligned}$$

□

Note that p_1, p_2 and p_3 must be constants and

$$p_1 + p_2 = p_2/p_3 = k/d$$

where d is the size of the neighborhood and k is a constant.

This computation also can be expressed as a 2-dimensional matrix M with d rows and $|C|$ columns.

For a 5 city TSP

	ab	bc	cd	de	ae	ac	ad	bd	be	ce
ABCDE	1	1	1	1	1	0	0	0	0	0
ABEDC	1	0	1	1	0	1	0	0	1	0
ABCED	1	1	0	1	0	0	1	0	0	1
ABDCE	1	0	1	0	1	0	0	1	0	1
ACBDE	0	1	0	1	1	1	0	1	0	0
ADCBE	0	1	1	0	1	0	1	0	1	0

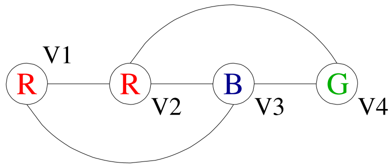
Looking at the neighbors in aggregate.

ab	bc	cd	de	ae	ac	ad	bd	be	ce
1	1	1	1	1	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1

If we can characterize particular types of neighbors, they can be removed.

	ab	bc	cd	de	ae	ac	ad	bd	be	ce
ABCDE	1	1	1	1	1	0	0	0	0	0
ABEDC	1	0	1	1	0	1	0	0	1	0
ABCED	1	1	0	1	0	0	1	0	0	1
ABDCE	1	0	1	0	1	0	0	1	0	1
ACBDE	0	1	0	1	1	1	0	1	0	0
ADCBE	0	1	1	0	1	0	1	0	1	0

Graph Coloring



Graph Coloring

The Components are edges in the graph.
Assume r colors, $|V|$ vertices.

$$p_1 = \frac{2(r-1)}{|V|(r-1)}$$

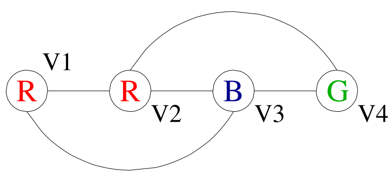
$$p_2 = \frac{2}{|V|(r-1)}$$

$$p_3 = 1/r$$

$$\operatorname*{avg}_{y \in N(x)} \{f(y)\} = f(x) - p_1 f(x) + p_2((1/p_3\bar{f}) - f(x))$$

$$= f(x) + \frac{2r}{|V|(r-1)}(\bar{f} - f(x))$$

Graph Coloring



$$\operatorname*{avg}_{y \in N'(x)} f(y) < f(x) < \operatorname*{avg}_{y \in N(x)} f(y) < \bar{f}$$

Graph Coloring, Partial Neighborhoods

Let Q_x denote vertices involved in a conflict.
E_x denotes the Components (edges that touch $v \in Q_x$)
Let $Degree(v)$ store the degree of vertex v.
The new set of components is given by

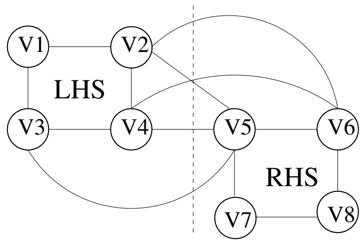
$$|E_x| = \sum_{v \in Q_x} Degree(v) - f(x)$$

$$p_1 = \frac{2(r-1)}{|Q_x|(r-1)}$$

$$p_2 = \frac{1}{|Q_x|(r-1)}$$

$$\operatorname*{avg}_{y \in N'(x)} \{f(y)\} = f(x) + \frac{\sum_{v \in Q_x} Degree(v) - (2r)f(x)}{|Q_x|(r-1)}$$

Min-Cut Graph Partitioning



Min-Cut Graph Partitioning

The Components are edges in the graph.
Assume r colors, $|V|$ vertices.

$$p_3 = \frac{n/2}{n-1} = \frac{n^2/4}{|C|}$$

$$\bar{f} = p_3 \sum_{c \in C} c = \frac{n}{2(n-1)} \sum_{e_{i,j} \in E} w_{i,j}$$

$$p_1 = \frac{2(n/2-1)}{n^2/4} = \frac{n-2}{n^2/4} = \frac{\alpha}{d}$$

$$p_2 = \frac{n}{n^2/4} = \frac{\alpha}{d}$$

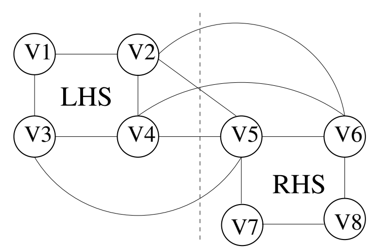
Min-Cut Graph Partitioning

$$\begin{aligned}\underset{y \in N(x)}{\text{avg}}\{f(y)\} &= f(x) - p_1 f(x) + p_2(1/p_3(\bar{f} - f(x))) \\ &= f(x) - \frac{n-2}{n^2/4} f(x) + \frac{n}{n^2/4}\left[\frac{2(n-1)}{n}\bar{f} - f(x)\right] \\ &= f(x) + \frac{2(n-1)}{n^2/4}(\bar{f} - f(x))\end{aligned}$$

where $k = 2(n-1)$ and the neighborhood size is $d = n^2/4$. Grover simplifies this to obtain:

$$\underset{y \in N(x)}{\text{avg}}\{f(y)\} = f(x) + \frac{8(n-1)}{n^2}(\bar{f} - f(x))$$

Min-Cut Graph Partitioning



Partial Neighborhoods for Min-Cut

Let n_L count the number of vertices in the LHS which have no edges that connect to the right hand size.

Let n_R count the number of vertices in the RHS which have no edges that connect to the right hand size.

There are n_L x n_R vertex pairs that cannot be yield an improvement.

Let d' denote the size of the new neighborhood.

Let W' represent the sum of all the weights that are eliminated when these moves are excluded.

Partial Neighborhoods for Min-Cut

Theorem

A partial neighborhood $N'(x)$ exists for the Min-Cut Graph Partitioning problems such that

$$\operatorname*{avg}_{y \in N'(x)}\{f(y)\} = f(x) - \frac{2n-2}{d'} f(x) + \frac{n(\sum_{e_{i,j} \in E} w_{i,j}) - W'}{d'}$$

where $d' = n^2/4 - |n_L||n_r|$

and $W' = |n_L| \sum_{i \in V, x \in n_R} w(i,x) + |n_R| \sum_{i \in V, x \in n_L} w(i,x)$

Partial Neighborhoods for Min-Cut

A vector $Weights(x)$ can be precomputed that stores the sum of the weights associated with edges incident on vertex x.
The information needed to compute W' is found in the vector $Weights(i)$ since

If $x \in n_L$ then $\sum_{i \in V} w(i,x) = Weights(x)$

If $x \in n_R$ then $\sum_{i \in V} w(i,x) = Weights(x)$

A “spectral” approach

Algorithms that use local *move* or *mutation* operators essentially perform a heuristic search on the landscape graph (call it G).

When is such a search successful?

- If there is no relationship between G and f, essentially a random search.
- Local search algorithms: considered the *state of the art* for many combinatorial optimization problems $\Rightarrow$ there must be a relationship between G and f.

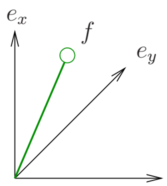
The study of landscapes is really the study of the relationship between the **spectrum**[1] of G and the function f.

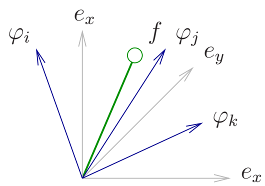
[1] i.e., the eigendecomposition of its adjacency matrix

Working in function space

Any function $f : X \to \mathbb{R}$ can be characterized as a vector in $\mathbb{R}^{|X|}$

Any vector in $\mathbb{R}^{|X|}$ can be characterized as a real function on X



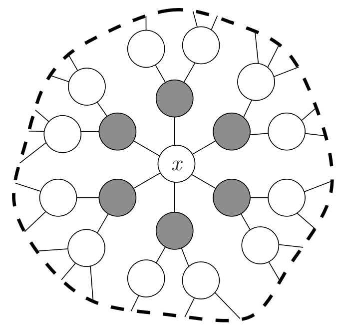


$e_y(x) = \delta_{xy}$

$$f(x) = \sum_y f_y e_y(x)$$

$$f(x) = \sum_i a_i \varphi_i(x)$$

The adjacency matrix



The landscape can be partially represented by an *adjacency matrix* $\boldsymbol{A}$.

$$\boldsymbol{A}_{xy} = \begin{cases} 1 & \text{if } y \in N(x); \\ 0 & \text{otherwise.} \end{cases}$$

Main insight #1: Matrices as operators

Any $|X| \times |X|$ matrix $\boldsymbol{M}$ can be characterized as an operator on the space of real functions over X.

$$\boldsymbol{M}f = g \qquad \boldsymbol{M} : \{f : X \to \mathbb{R}\} \to \{f : X \to \mathbb{R}\}$$

Consider the matrix vector product $\boldsymbol{A}f = g$.

$$x \rightarrow \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \\ 4 \\ 1 \\ 1 \\ 7 \\ 5 \\ 8 \end{bmatrix} = \begin{bmatrix} 7 \\ 11 \\ 9 \\ 14 \\ 15 \\ 11 \\ 13 \\ 13 \end{bmatrix} \leftarrow g(x)$$

Image of f under $\boldsymbol{A}$ is a function $g(x)$ that gives the sum of f values over the neighbors of x (the **sifting property**).

Main insight #2: Eigenfunctions of the adjacency

Want a set of basis functions that allow us to study the relationship between the neighborhood graph and the fitness function.

$$f(x) = \sum_i a_i \varphi_i(x)$$

where a_i is an "amplitude" and $\varphi_i : X \to \mathbb{C}$ is an eigenfunction of $\boldsymbol{A}$.

An *eigenfunction* of $\boldsymbol{A}$ is any function $\varphi : X \to \mathbb{C}$ that satisfies the equation

$$(\boldsymbol{A}\varphi)(x) = \boldsymbol{A}\varphi(x) = \lambda\varphi(x)$$

for a constant λ.

What does this have to do with elementary landscapes?

Recall the "wave" equation

$$\operatorname*{avg}_{y \in N(x)} \{f(y)\} = f(x) + \frac{k}{d}(\bar{f} - f(x)) \qquad (*)$$

$$\operatorname*{avg}_{y \in N(x)} \{f(y)\} = \frac{1}{d} \sum_{y \in N(x)} f(y) = \frac{1}{d} g(x) = \frac{1}{d} \mathbf{A} f(x)$$

(the last equivalence follows by the **sifting property**)

So putting this with (*) above, we get

$$\mathbf{A} f(x) = (d-k) f(x) + (k\bar{f})$$

So if a function obeys the wave equation... it is (up to an additive constant) an eigenfunction of the adjacency matrix of G

Using the analysis

Computing statistics over regions (Sutton, Whitley & Howe 2012)

Approximating the fitness distribution (Sutton, Whitley & Howe 2011)

Finding good mutation rates (Chicano & Alba 2011)

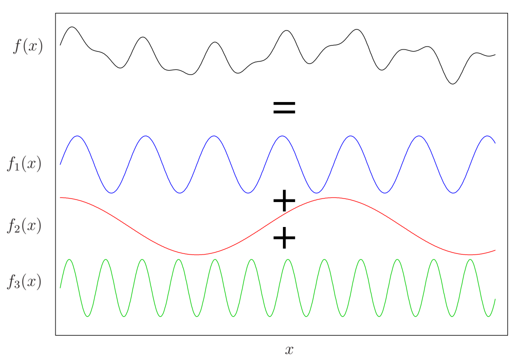
Providing fitness bounds on the existence of certain neighborhood features

Computing the correlation structure

Designing search algorithms and heuristics

Why "elementary"?

Components of more general landscapes



Superpositions of Elementary Landscapes

$$f(x) = f1(x) + f2(x) + f3(x) + f4(x)$$

$$f1(x) = f1_a(x) + f1_b(x) + f1_c(x)$$
$$f2(x) = f2_a(x) + f2_b(x) + f2_c(x)$$
$$f3(x) = f3_a(x) + f3_b(x) + f3_c(x)$$
$$f4(x) = f4_a(x) + f4_b(x) + f4_c(x)$$

$$\varphi^{(1)}(x) = f1_a(x) + f2_a(x) + f3_a(x) + f4_a(x)$$
$$\varphi^{(2)}(x) = f1_b(x) + f2_b(x) + f3_b(x) + f4_a(x)$$
$$\varphi^{(3)}(x) = f1_c(x) + f2_c(x) + f3_c(x) + f4_a(x)$$

$$f(x) = \varphi^{(1)}(x) + \varphi^{(2)}(x) + \varphi^{(3)}(x)$$

MAX-k-SAT

Given: a set of m disjunctive, length-k clauses over a set of n variables

MAX-3-SAT

$$\{(v_2 \vee \neg v_1 \vee v_4), (\neg v_3 \vee v_1 \vee \neg v_2), \ldots\}$$

The set of all assignments is isomorphic to $\{0,1\}^n$.

Fitness function $f : \{0,1\}^n \to \{0, 1, \ldots, m\}$ counts how many clauses are satisfied under an assignment.

Neighborhood operator is *Hamming operator*: flip each bit.

Fitness functions over bitstrings

Spectral decomposition of A

Write $A = WDW^{-1}$ where D is a diagonal matrix
The columns of W are eigenvectors of A

For Hamming operator, A is the hypercube adjacency $\Rightarrow$ W is the well-known Walsh matrix.

Working in function space...
The 2^n columns of W correspond to the Walsh functions

$$\psi_i : \{0,1\}^n \to \mathbb{R}$$

For Hamming adjacency, the Walsh functions obey

$$A\psi_i(x) = \lambda_i \psi_i(x)$$

Fitness functions over bitstrings

W is an orthogonal matrix, so any real function f over $\{0,1\}^n$ can be written as a linear combination of Walsh functions

$$f(x) = \sum_{i=1}^{2^n} w_i \psi_i(x)$$

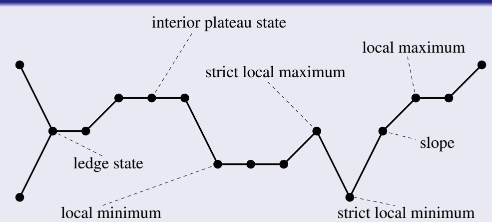
For MAX-k-SAT, most coefficients w_i vanish

- when the length-n binary representation of i has greater than k bits (due to Rana, Heckendorn, Whitley 1998)

Thus there are are $O(2^k)$ nonzero coefficients, and they can be computed in time $O(m2^k)$.

Forbidden structure in MAX-3-SAT search space

Search positions of Hoos & Stützle

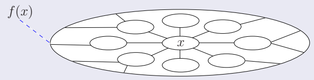


Hoos & Stützle (2004) characterized the search space by empirically sampling and determining the frequency of search positions

On MAX-3-SAT, they could not find any *interior plateau* states.

Forbidden structure in MAX-3-SAT search space

Interior plateau state

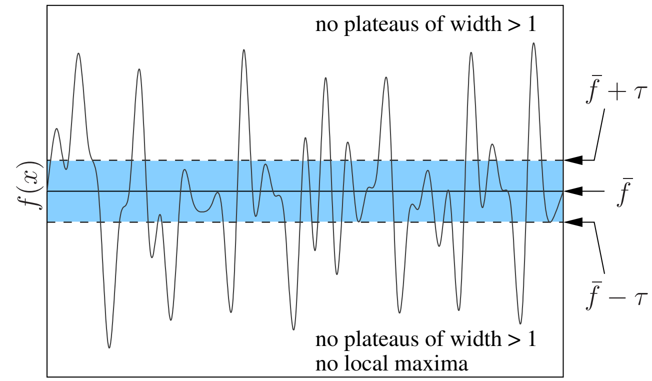


$$f(x) = \frac{1}{n}\sum_{y \in N(x)} f(y) = \frac{1}{n}\mathbf{A}f(x) \qquad \text{(sifting)}$$

$$f(x) = \frac{1}{n}\sum_i w_i \mathbf{A}\psi_i(x) = \frac{1}{n}\sum_i w_i \lambda_i \psi_i(x)$$

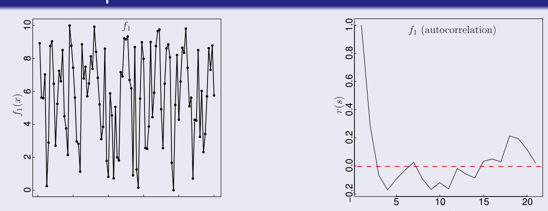
$$\bar{f} - \tau \le f(x) \le \bar{f} + \tau$$

Forbidden structure in MAX-3-SAT search space

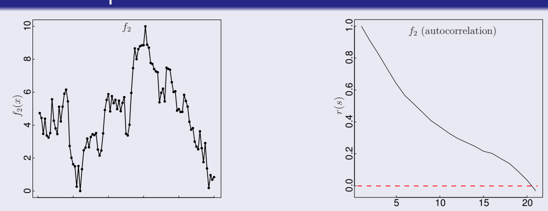


Correlation structure

A rugged landscape



A smooth landscape



Correlation structure

Background

Dynamics of polynucleotide folding landscapes, interest in TSP (Fontana et al., 1989)

TSP (Stadler & Schnabl, 1992)

Graph bipartitioning (Stadler & Happel, 1992)

Idea for problem classification (Angel & Zissimopolous, 2000)

MAX-k-SAT (Sutton, Whitley & Howe, 2009)

Quadratic Assignment Problem (Chicano, Luque & Alba, 2012)

Correlation structure

Random walk transition matrix

$$\boldsymbol{T} = \frac{1}{n}\boldsymbol{A}$$

Random walk process estimates the following equation

$$r(s) = \frac{\langle f, \boldsymbol{T}^s f\rangle - \langle \mathbf{1}, f\rangle^2}{\langle f, f\rangle - \langle \mathbf{1}, f\rangle^2}$$

Replace f with the expansion...

Correlation structure

Lemma

ψ_i is an eigenvector of the random walk transition matrix $\boldsymbol{T}$.

$$\boldsymbol{T}\psi_i = \lambda_i \psi_i$$

where $\lambda_i = \left(1 - \frac{2\langle i,i\rangle}{n}\right)$.

Remember in the Walsh decomposition

$$f(x) = \sum_i w_i \psi_i(x)$$

we are actually writing the fitness function in terms of the eigenbasis of $\boldsymbol{T}$

Correlation structure

Remark. We have the following identities:

$\langle f, f\rangle = \sum_i w_i^2$ $\quad \langle f, \boldsymbol{T}^s f\rangle = \sum_i \lambda_i^s w_i^2$ $\quad \langle \mathbf{1}, f\rangle = w_0$

$$\begin{aligned}\langle f, f\rangle &= \langle \sum_i w_i\psi_i, \sum_j w_j\psi_j\rangle \\ &= \sum_i\sum_j w_i w_j \langle \psi_i, \psi_j\rangle \\ &= \sum_i w_i^2 && \text{since } \{\psi_i\} \text{ is an orthogonal basis}\end{aligned}$$

Correlation structure

Remark. We have the following identities:

$\langle f, f\rangle = \sum_i w_i^2$ $\quad \langle f, \boldsymbol{T}^s f\rangle = \sum_i \lambda_i^s w_i^2$ $\quad \langle \mathbf{1}, f\rangle = w_0$

$$\begin{aligned}\langle f, \boldsymbol{T}^s f\rangle &= \langle \sum_i w_i\psi_i, \boldsymbol{T}^s \sum_j w_j\psi_j\rangle \\ &= \sum_i\sum_j w_i \lambda_j^s w_j \langle \psi_i, \psi_j\rangle && \text{since } \{\psi_i\} \text{ is an eigenbasis} \\ &= \sum_i \lambda_i^s w_i^2 && \text{since } \{\psi_i\} \text{ is an orthogonal basis}\end{aligned}$$

Correlation structure

Remark. We have the following identities:

$\langle f,f\rangle = \sum_i w_i^2 \qquad \langle f, \boldsymbol{T}^s f\rangle = \sum_i \lambda_i^s w_i^2 \qquad \langle \mathbf{1}, f\rangle = w_0$

$$\begin{aligned}\langle \mathbf{1}, f\rangle &= \langle \mathbf{1}, \sum_i w_i \psi_i\rangle \\ &= w_0\end{aligned}$$

Correlation structure

Remark. We have the following identities:

$\langle f,f\rangle = \sum_i w_i^2 \qquad \langle f, \boldsymbol{T}^s f\rangle = \sum_i \lambda_i^s w_i^2 \qquad \langle \mathbf{1}, f\rangle = w_0$

Random walk process estimates the following equation

$$r(s) = \frac{\langle f, \boldsymbol{T}^s f\rangle - \langle \mathbf{1}, f\rangle^2}{\langle f,f\rangle - \langle \mathbf{1}, f\rangle^2}$$

Substitutions...

$$r(s) = \frac{\sum_i \lambda_i^s w_i^2 - w_0^2}{\sum_j w_j^2 - w_0^2} = \frac{\sum_{i\neq 0} \lambda_i^s w_i^2}{\sum_{j\neq 0} w_j^2}$$

Correlation structure

This gives *exact autocorrelation* function

$$r(s) = \frac{\sum_{i\neq 0} \lambda_i^s w_i^2}{\sum_{j\neq 0} w_j^2}$$

where $\lambda_i = \left(1 - \frac{2\langle i,i\rangle}{n}\right)$.

Recall for MAX-k-SAT all nonzero w_i can be computed in $O(m)$ time.

Neighborhoods that result in elementary landscapes

MAX-k-SAT – neighborhood operator is *Hamming operator*, i.e., flip each bit:

$$N\big((0,1,0)\big) = \{(1,1,0),(0,0,0),(0,1,1)\}.$$

Together, f and N **do not** form an elementary landscape, rather we have been expressing f as a linear combination of elementary landscapes.

Questions

- Can we find a new neighborhood operator N' such that f and N' yield an elementary landscape?
- If so, how is this useful?

Max-2-Sat

Consider now the case when the clause size is *exactly* 2...

New neighborhood operator: flip a bit *or* flip all bits at once

$$N'\big((0,1,0)\big) = \{(1,1,0),(0,0,0),(0,1,1)\} \cup \{1,0,1\}.$$

Theorem

$\big(\{0,1\}^n, f, N'\big)$ is elementary

Max-2-Sat

Proof.
Consider an arbitrary assignment x. We study the condition of the i-th clause $(\ell_1 \vee \ell_2)$ under x and its neighbors:

- Case 1: i-th clause is **not** satisfied by x
  Then there are three assignments $y \in N(x)$ that satisfy it.
  (the two distinct Hamming neighbors that negate each variable appearing in the clause, and the element corresponding to the complement of x, which negates both variables in the clause).
- Case 2: exactly one literal evaluates to true under x
  Then there is one element $y \in N(x)$ that does not satisfy it.
  (the negation of the true literal).
- Case 3: both literals evaluate to true
  Then there is one element $y \in N(x)$ that does not satisfy it.
  (when x is complemented).

Max-2-Sat

Proof (continued).
Clause indicator function $c_i : \{0,1\}^n \to \{0,1\}$.

$$\sum_{y \in N'(x)} c_i(y) = 3(1 - c_i(x)) + (|N'(x)| - 1)c_i(x) = 3 + (n-3)c_i(x).$$

Since $f(x) = \sum_{i=1}^{m} c_i(x)$, we have

$$\sum_{y \in N'(x)} f(y) = \sum_{i=1}^{m} \big(3 + (n-3)c_i(x)\big) = 3m + (n-3)f(x).$$

Thus N' and f satisfy the "wave equation".

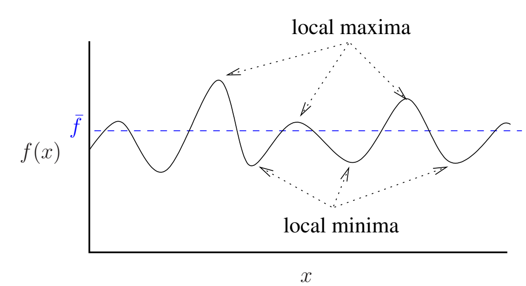
Max-2-Sat

Corollary

Suppose $\hat{x}$ has no improving neighbors in N'. Then

$$f(\hat{x}) \geq \frac{3}{4} m$$

where m is the number of clauses.



MAX-2-SAT

Define $A_{i,j}$ be the 3-set of clauses defined on two variables v_i and v_j

$$A_{i,j} = \{(\neg v_i \vee \neg v_j), (\neg v_i \vee v_j), (v_i \vee \neg v_j)\}.$$

Construct a MAX-2-SAT instance on $2q$ variables by taking the union of q 3-sets of clauses

$$A_{1,2} \cup A_{3,4} \cup \cdots \cup A_{2q-1,2q}.$$

Thus for this instance, $m = 3q$.

Consider $\hat{x} = (111 \cdots 1)$ (no improving Hamming neighbors)

Since $\hat{x}$ satisfies 2 clauses in each set $A_{i,j}$, we have

$$f(\hat{x}) = 2q = \frac{2}{3}m < \frac{3}{4}m.$$

MAX-2-SAT

It follows that, when using the Hamming (flip) operator on MAX-2-SAT there can be local optima with inferior fitness to **all local optima** on the landscape induced by the new operator.

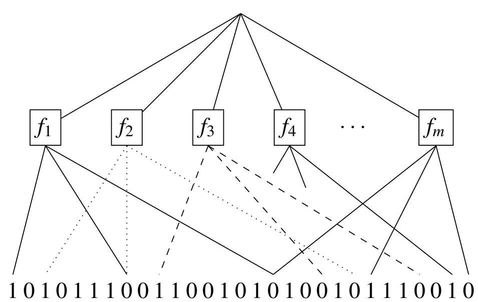
Local search using N' is a polynomial-time $3/4$-approximation algorithm for MAX-2-SAT.

This result was also used to show that the (1+1) EA is a randomized fixed-parameter tractable algorithm for the standard parameterization of MAX-2-SAT (Sutton, Day, & Neumann, GECCO 2012)

Constant Time Steepest Descent

A model for all bounded pseudo-Boolean functions:

$$f(x) = \sum_{i=1}^{m} f_i(x; mask)$$



Constant Time Steepest Descent

Let vector w' store the Walsh coefficients including the sign relative to solution x.

$$w'_i(x) = w_i \psi_i(x)$$

Flip bit p such that $y_p \in N(x)$. Then

if $p \subset i$ then $w'_i(y_p) = -w'_i(x)$

otherwise $w'_i(y_p) = w'_i(x)$

For MAX-kSAT and NK-Landscapes flipping one bit changes the sign of only a constant number of Walsh coefficients.

Constant Time Steepest Descent

Construct a vector S such that

$$S_p(x) = \sum_{\forall b,\ p \subset b} w'_b(x)$$

In this way, all of the Walsh coefficients whose signs will be changed by flipping bit p are collected into a single number $S_p(x)$.

Constant Time Steepest Descent

Lemma 1.
Let $y_p \in N(x)$ be the neighbor of string x generated by flipping bit p.
Then $f(y_p) = f(x) - 2(S_p(x))$.

If $p \subset b$ then $\psi_b(y_p) = -1(\psi_b(x))$ and otherwise $\psi_b(y_p) = \psi_b(x)$. For each Walsh coefficient that changes, the change is $-2(w'_b(x))$.

Corollary: For all bit flips j, $f(y_j) = f(x) - 2(S_j(x))$.
Thus, $S_j(x)$ can be used as a proxy for $f(y_j)$; $f(x)$ is constant as j is varied. Maximizing $S_j(x)$ minimizes the neighborhood of $f(x)$.

Constant Time Steepest Descent

To make this easy, assume we have an NK-Landscape or MAX-kSAT problem such that **every variable occcurs exactly the same number of times.**

This case is easy to analysis, but also exactly corresponds to the average complexity case (with mild restrictions on the frequence of bit flips).

Assume each variable appears kc time.

For MAX-kSAT c is the clause variable ratio. For NK-landscapes $c = 1$.

Constant Time Steepest Descent

When one bit flips, it impacts kc subfunctions. There are $k(k-1)$ pairings of bits in each subfunction. Thus there are $ck(k-1)$ total bits affected by a bit flip.

Also at most $ck(k-1)$ terms in vector S change.

When one bit flips, it impacts at most $2^{k-1} - 1$ Walsh coefficients in any subfunction. If a bit appears in exactly kc functions, then at most $ck(2^{k-1} - 1)$ nonlinear Walsh coefficients change. **Thus, the update take $O(1)$ time.**

The locations of the updates are obvious

$$
\begin{aligned}
S_1(y_p) &= S_1(x) \\
S_2(y_p) &= S_2(x) \\
S_3(y_p) &= S_3(x) + \sum_{\forall b,\ (p \wedge 3) \subset b} w'_b(x) \\
S_4(y_p) &= S_4(x) \\
S_5(y_p) &= S_5(x) \\
S_6(y_p) &= S_6(x) \\
S_7(y_p) &= S_7(x) \\
S_8(y_p) &= S_8(x) + \sum_{\forall b,\ (p \wedge 8) \subset b} w'_b(x) \\
S_9(y_p) &= S_9(x)
\end{aligned}
$$

70

"Old" and "New" improving moves

A "new" improving move must be a new updated locations in S. Checking these takes $O(1)$ time on average.

There can be previously discovered “old" moves stored in a buffer. Here we approximate steepest descent.

If there are less than $ck(k-1)$ old moves items in the buffer we check them all. If there are more than $ck(k-1)$ old moves in the buffer, we sample $ck(k-1)$ moves and select the best old move.

We then select either the best new move or the (approximate) best old move. **Total cost: at most** $2ck(k-1)+1$ **comparisons, which is** $O(1)$

71

Next Ascent

If we want to do Next Ascent instead of Steepest Ascent, we just all of the improving moves into a buffer and pick one. Again, this takes $O(1)$ time.

72

Identifying Local Optima

If there are no improving moves, the point is a local optimum. The point is automatically identified: there are no "old" improving moves and no update is an improving move.

73

Speed Results for MAXSAT Solvers

	AdaptG2WSAT	GSAT	Walsh
UR-1000000	698.86	32.13	1.80
UR-2000000	3458.06	140.37	3.88
UR-3000000	8157.01	319.95	6.05
mem-ctrl2	4120.52	54.11	4.17
wb_4m8s-48	7339.77	83.16	6.06

Table: Time in seconds require to reach a Local Optima for several stochastic local search algorithms for MAX-kSAT problems.

Steepest Descent over Neighborhood Means

We have the vector S such that

$$S_p(x) = \sum_{\forall b,\ p \subset b} w'_b(x)$$

Also construct the vector Z such that

$$Z_p(x) = \sum_{\forall b,\ p \subset b} order(b)\ w'_b(x)$$

Note that S and Z and U all update at exactly the same locations.

Lemma 2.

$$Avg(N(y_p)) = Avg(N(x)) - 2(S_p(x)) + \frac{4}{N} Z_p(x)$$

Steepest Descent over Neighborhood Means

Let $U_p(x) = -2(S_p(x)) + \frac{4}{N} Z_p(x)$

$$Avg(N(y_p)) = Avg(N(x)) + U_p(x)$$

The vector $U(x)$ can now be used as a proxy for $Avg(N(x))$
Maximizing $U_p(x)$ minimizes the neighborhood of $Avg(N(y_p))$.

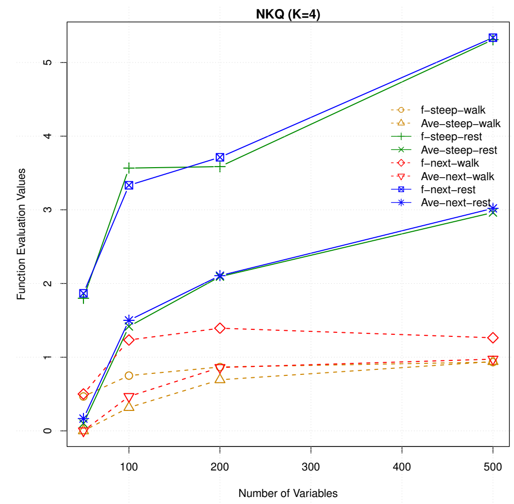
The locations of the updates are obvious

$$\begin{aligned}
U_1(y_p) &= U_1(x)\\
U_2(y_p) &= U_2(x)\\
U_3(y_p) &= U_3(x) + Update\\
U_4(y_p) &= U_4(x)\\
U_5(y_p) &= U_5(x)\\
U_6(y_p) &= U_6(x)\\
U_7(y_p) &= U_7(x)\\
U_8(y_p) &= U_8(x) + Update\\
U_9(y_p) &= U_9(x)
\end{aligned}$$

Search on an NKq-Landscape

And NKq-Landscape generates subfunctions using only q values. For $q = 2$ there are many plateaus and equal moves.
1. $f(x)$ versus $Avg(N(x))$
2. Steepest Ascent versus Next Ascent
3. Random Walk Restart (with $O(1)$ cost) versus Hard Random Restart (with $O(N)$ cost)

Search on an NKq-Landscape



Conclusions

- Elementary landscapes provide an interesting tool for analyzing search in combinatorial optimization
- Linear algebraic approach to formalizing “landscape” concept for discrete problems
- Ongoing research to connect search space topology to algorithm dynamics