Landscapes and Other Art Forms.

Darrell Whitley
Computer Science, Colorado State University

Blind No More = GRAY BOX Optimization

Darrell Whitley
Computer Science, Colorado State University

With Thanks to: Francisco Chicano, Gabriela Ochoa, Andrew Sutton and Renato Tinós

GRAY BOX OPTIMIZATION

We can construct "Gray Box" optimization for pseudo-Boolean optimization problems composed of M subfunctions, where each subfunction accepts at most k variables.

Exploit the *general properties of every* Mk Landscape:

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

Which can be expressed as a Walsh Polynomial

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

Or can be expressed as a sum of k Elementary Landscapes

$$f(x) = \sum_{i=1}^{k} \varphi^{(k)}(W(f(x)))$$

Walsh Example: MAXSAT

Given a logical expression consisting of Boolean variables, determine whether or not there is a setting for the variables that makes the expression TRUE.

Literal: a variable or the negation of a variable

Clause: a disjunct of literals

A 3SAT Example

$$(\neg x_2 \vee x_1 \vee x_0) \wedge (x_3 \vee \neg x_2 \vee x_1) \wedge (x_3 \vee \neg x_1 \vee \neg x_0)$$

recast as a MAX3SAT Example

$$(\neg x_2 \vee x_1 \vee x_0) + (x_3 \vee \neg x_2 \vee x_1) + (x_3 \vee \neg x_1 \vee \neg x_0)$$

Walsh Example

$$\frac{1}{8}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 0.875 \\ -0.125 \\ -0.125 \\ -0.125 \\ 0.125 \\ 0.125 \\ 0.125 \\ 0.125 \end{bmatrix}$$

- All ψ_j's except ψ_0 have 4 1's and 4 -1's.
- ψ_0 has all 1's.
- f for clauses of length 3 will contain one 0

Walsh Example

$$f_1 = (\neg x_2 \vee x_1 \vee x_0)$$
$$f_2 = (x_3 \vee \neg x_2 \vee x_1)$$
$$f_3 = (x_3 \vee \neg x_1 \vee \neg x_0)$$

x	w_i	$W(f_1)$	$W(f_2)$	$W(f_3)$	$W(f(x))$
0000	w_0	0.875	0.875	0.875	2.625
0001	w_1	−0.125	0	0.125	0
0010	w_2	−0.125	−0.125	0.125	−0.125
0011	w_3	−0.125	0	−0.125	−0.250
0100	w_4	0.125	0.125	0	0.250
0101	w_5	0.125	0	0	0.125
0110	w_6	0.125	0.125	0	0.250
0111	w_7	0.125	0	0	0.125
1000	w_8	0	−0.125	−0.125	−0.250
1001	w_9	0	0	0.125	0.125
1010	w_{10}	0	−0.125	0.125	0
1011	w_{11}	0	0	−0.125	−0.125
1100	w_{12}	0	0.125	0	0.125
1101	w_{13}	0	0	0	0
1110	w_{14}	0	0.125	0	0.125

BLACK BOX OPTIMIZATION

Don't wear a blind fold when walking about London if you can help it!

NK-Landscapes

An Adjacent NK Landscape: $n = 6$ and $k = 3$. The subfunctions:

$$
\begin{array}{l}
f_0(x_0, x_1, x_2) \\
\quad f_1(x_1, x_2, x_3) \\
\quad\quad f_2(x_2, x_3, x_4) \\
\quad\quad\quad f_3(x_3, x_4, x_5) \\
\quad\quad\quad\quad f_4(x_4, x_5, x_0) \\
\quad\quad\quad\quad\quad f_5(x_5, x_0, x_1)
\end{array}
$$

These problems can be solved to optimality using Dynamic Programming.

Percent of Offspring that are Local Optima

Using a Very Simple (Stupid) Hybrid GA:

N	k	Model	2-point Xover	Uniform Xover	PX
100	2	Adj	74.2 ±3.9	0.3 ±0.3	100.0 ±0.0
300	4	Adj	30.7 ±2.8	0.0 ±0.0	94.4 ±4.3
500	2	Adj	78.0 ±2.3	0.0 ±0.0	97.9 ±5.0
500	4	Adj	31.0 ±2.5	0.0 ±0.0	93.8 ±4.0
100	2	Rand	0.8 ±0.9	0.5 ±0.5	100.0 ±0.0
300	4	Rand	0.0 ±0.0	0.0 ±0.0	86.4 ±17.1
500	2	Rand	0.0 ±0.0	0.0 ±0.0	98.3 ±4.9
500	4	Rand	0.0 ±0.0	0.0 ±0.0	83.6 ±16.8

Number of partition components discovered

N	k	Model	Paired PX	
			Mean	Max
100	2	Adjacent	3.34 ±0.16	16
300	4	Adjacent	5.24 ±0.10	26
500	2	Adjacent	7.66 ±0.47	55
500	4	Adjacent	7.52 ±0.16	41
100	2	Random	3.22 ±0.16	15
300	4	Random	2.41 ±0.04	13
500	2	Random	6.98 ±0.47	47
500	4	Random	2.46 ±0.05	13

Paired PX uses Tournament Selection. The first parent is selected by fitness. The second parent is selected by Hamming Distance.

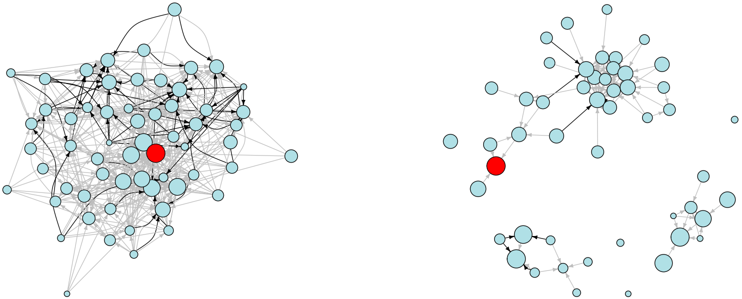
Optimal Solutions for Adjacent NK

		2-point	Uniform	Paired PX
N	k	Found	Found	Found
300	2	18	0	100
300	3	0	0	100
300	4	0	0	98
500	2	0	0	100
500	3	0	0	98
500	4	0	0	70

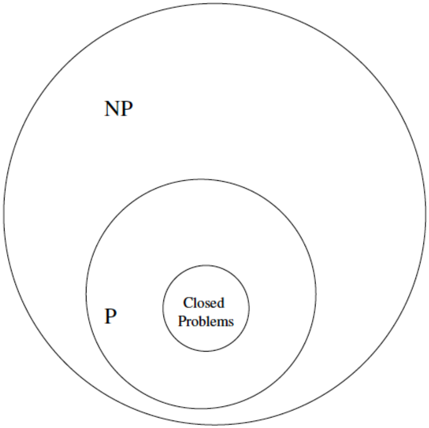
Percentage over 50 runs where the global optimum was Found in the experiments of the hybrid GA with the Adjacent NK Landscape.

Tunnelling Local Optima Networks

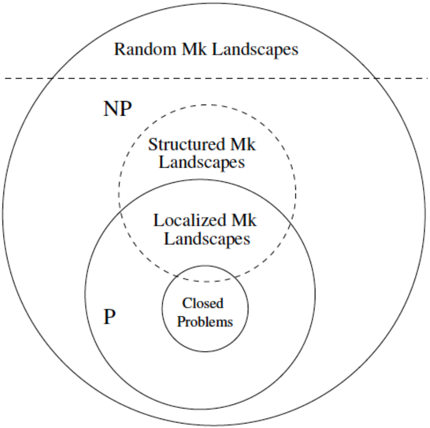
NK Landscapes: Ochoa et al. GECCO 2015



NK and Mk Landscapes, P and NP



NK and Mk Landscapes, P and NP



But a Hybrid Genetic Algorithm is NOT how we should solve these NK Landscape Problems.

We can exactly know the location of improving moves in constant time.
No *enumeration of neighbors* is needed.

Under well behaved conditions,
we can exactly know the location of improving moves
for r steps ahead in constant time.

Walsh Analysis

Every n-bit MAXSAT or NK-landscape or P-spin problem is a sum of m subfunctions, f_i:

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

The Walsh transform of f is a sum of the Walsh transforms of the individual subfunctions.

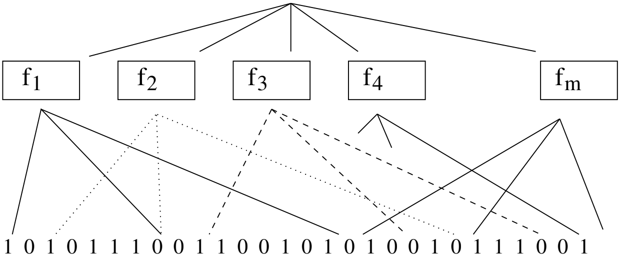
$$W(f(x)) = \sum_{i=1}^{m} W(f_i(x))$$

If m is $O(n)$ then the number of Walsh coefficients is $m\ 2^k = O(n)$.

“Mk Landscapes”

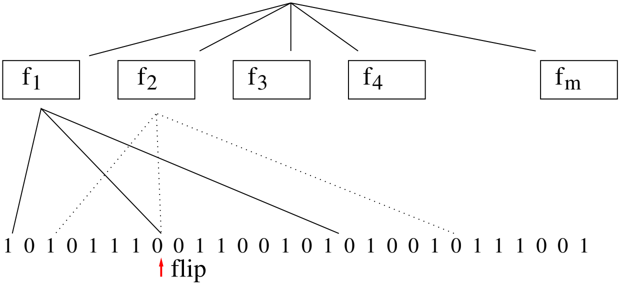
A General Model for all bounded Pseudo-Boolean Problems

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



When 1 bit flips what happens?

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



Constant Time Steepest Descent

Assume we flip bit p to move from x to $y_p \in N(x)$.
Construct a vector $Score$ such that

$$Score(x, y_p) = -2 \left\{ \sum_{\forall b,\ p \subset b} -1^{b^T x} w_b(x) \right\}$$

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

The GSAT algorithm has done this for 23 years (Thanks to H. Hoos).

NOTE: Hoos and Stützle have claimed a constant time result, but without proof. An average case complexity proof is required to obtain general constant time complexity results (Whitley 2013, AAAI). Also it does not matter if the problems are uniform random or not.

Best Improving and Next Improving moves

"Best Improving" and "Next Improving" moves cost the same.

GSAT uses a Buffer of best improving moves

$$Buffer(best.improvement) =< M_{10}, M_{1919}, M_{9999} >$$

But the Buffer does not empty monotonically: this leads to thrashing.

Instead uses multiple Buckets to hold improving moves

$$Bucket(best.improvement) =< M_{10}, M_{1919}, M_{9999} >$$

$$Bucket(best.improvement - 1) =< M_{8371}, M_{4321}, M_{847} >$$

$$Bucket(all.other.improving.moves) =< M_{40}, M_{519}, M_{6799} >$$

This improves the runtime of GSAT by a factor of 20X to 30X.
The solution for NK Landscapes is only slightly more complicated.

The locations of the updates are obvious

$$
\begin{aligned}
Score(y_p, y_1) &= Score(x, y_1) \\
Score(y_p, y_2) &= Score(x, y_2) \\
Score(y_p, y_3) &= Score(x, y_3) - 2\Big(\sum_{\forall b,\ (p \wedge 3) \subset b} w'_b(x)\Big) \\
Score(y_p, y_4) &= Score(x, y_4) \\
Score(y_p, y_5) &= Score(x, y_5) \\
Score(y_p, y_6) &= Score(x, y_6) \\
Score(y_p, y_7) &= Score(x, y_7) \\
Score(y_p, y_8) &= Score(x, y_8) - 2\Big(\sum_{\forall b,\ (p \wedge 8) \subset b} w'_b(x)\Big) \\
Score(y_p, y_9) &= Score(x, y_9)
\end{aligned}
$$

The locations of the updates are obvious

$$S_1(y_p) = S_1(x) - 2 \sum_{\forall b,\ (1 \wedge p) \subset b} w'_b(x)$$

$$S_2(y_p) = S_2(x)$$

$$S_3(y_p) = S_3(x)$$

$$S_4(y_p) = S_4(x)$$

$$S_5(y_p) = S_5(x) - 2 \sum_{\forall b,\ (5 \wedge p) \subset b} w'_b(x)$$

$$S_6(y_p) = S_6(x)$$

$$S_7(y_p) = S_7(x)$$

What if we could look R Moves Lookahead?

Consider R=3

Let $Score(3, x, y_{i,j,k})$ indicate we move from x to $y_{i,j,k}$ by flipping the 3 bits i, j, k. In general, we compute $Score(r, x, y_p)$ when flipping r bits.

$$f(y_i) = f(x) + Score(1, x, y_i)$$

$$\begin{aligned} f(y_{i,j}) &= f(y_i) + Score(1, y_i, y_j) \\ f(y_{i,j}) &= f(x) + Score(2, x, y_{i,j}) \end{aligned}$$

$$\begin{aligned} f(y_{i,j,k}) &= f(y_{i,j}) + Score(1, y_{i,j}, y_k) \\ f(y_{i,j,k}) &= f(x) + Score(3, x, y_{i,j,k}) \end{aligned}$$

With thanks to Francisco Chicano!

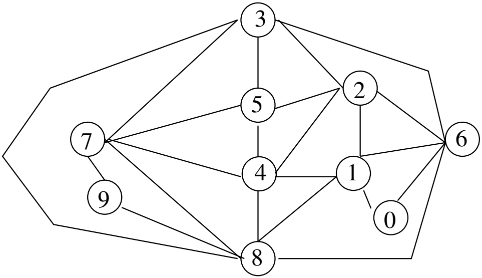
Why Doesn't this exponentially EXPLODE???

$$
\begin{aligned}
f(y_{i,j,k}) &= ((f(x) + Score(1, x, y_i)) + Score(1, y_i, y_j)) + Score(1, y_{i,j}, y_k) \\
Score(3, x, y_{i,j,k}) &= Score(2, x, y_{i,j}) + Score(1, y_{i,j}, y_{i,j,k})
\end{aligned}
$$

If there is no Walsh Coefficient $w_{i,j}$ then $Score(1, y_i, y_{i,j}) = 0$.

Assume we have already moves of length shorter than 3.
If there are no Walsh Coefficients "linking" i, j, k then
$Score(3, x, y_{i,j,k}) = 0$.

The Variable Interaction Graph

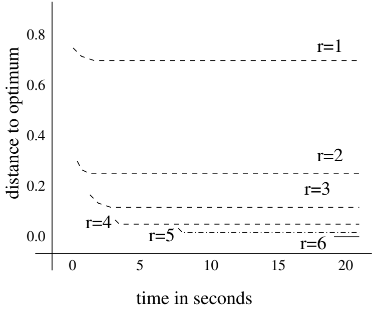


The Variable Interaction Graph

Assume all distance 1 moves are taken.

There cannot be a move flipping bits 4, 6, 9 that yields an improving move because there are no interactions and no Walsh coefficients.

Multiple Step Lookahead Local Search



In this figure, N = 12,000, k=3, and q=4. The radius is 1, 2, 3, 4, 5, 6. At r=6 the global optimum is found.

Steepest Descent on Moments

Both $f(x)$ and $Avg(N(x))$ can be computed with Walsh Spans.

$$f(x) = \sum_{z=0}^{3} \varphi^{(z)}(x)$$

$$Avg(N(x)) = f(x) - 1/d \sum_{z=0}^{3} 2z\varphi^{(p)}(x)$$

$$Avg(N(x)) = \sum_{z=0}^{3} \varphi^{(z)}(x) - 1/N \sum_{z=0}^{3} 2z\varphi^{(z)}(x)$$

$$Avg(N(x)) = \sum_{z=0}^{3} \varphi^{(z)}(x) - 2/N \sum_{z=0}^{3} z\varphi^{(z)}(x)$$

Steepest Descent on Moments

Assume the vectors $S_p = Score_p$ and Z_p just map the change in the Walsh coefficients. Z_p is computed using changes in $z\varphi^{(z)}(x)$

$$
\begin{aligned}
ScoreAvg(y_p, y_1) &= ScoreAvg(x, y_1) \\
ScoreAvg(y_p, y_2) &= ScoreAvg(x, y_2) \\
ScoreAvg(y_p, y_3) &= ScoreAvg(x, y_3) + WeightedWalshUpdate \\
ScoreAvg(y_p, y_4) &= ScoreAvg(x, y_4) \\
ScoreAvg(y_p, y_5) &= ScoreAvg(x, y_5) \\
ScoreAvg(y_p, y_6) &= ScoreAvg(x, y_6) \\
ScoreAvg(y_p, y_7) &= ScoreAvg(x, y_7) \\
ScoreAvg(y_p, y_8) &= ScoreAvg(x, y_8) + WeightedWalshUpdate \\
ScoreAvg(y_p, y_9) &= ScoreAvg(x, y_9)
\end{aligned}
$$

$\Theta(1)$ Steepest Descent on Moments

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

What's (Obviously) Next?

- Local Search with **r** Move Lookahead PLUS Partition Crossover.
- Apply **r** Move Lookahead and Partition Crossover to MAX-kSAT.
- Use Deterministic Improving Moves.
- Use Deterministic Recombination.

What's (Obviously) Next?

We can now solve 1 million variable NK-Landscapes to optimality in approximately linear time. (Paper submitted.)

What's (Obviously) Next?

- Put an End to the domination of Black Box Optimization.
- Wait for Tonight and Try to Take over the World.

THANK YOU

Take Home Message:

PROBLEM STRUCTURE MATTERS.

Black Box Optimizers can never match the performance of an algorithm that efficiently exploits problem structure.

But we need only a small amount of information: **Gray Box Optimization**.

For Mk Landscapes , we can use Deterministic Moves and Deterministic Crossover.