Next Generation Genetic Algorithms: Efficient Crossover and Local Search and New Results on Crossover Lattices

Darrell Whitley,
GECCO, 2024, Melbourne

GECCO '24 Companion, July 14–18, 2024, Melbourne, VIC, Australia

ACM 979-8-4007-0495-6/24/07
https://doi.org/10.1145/3638530.3648425

1

1

Next Generation Genetic Algorithms

Darrell Whitley

Computer Science
Colorado State University

2

2

Thanks to my PhD Students!

- N. Karunanithi
- Tim Starkweather
- Scott Gordon
- Keith Mathias
- Raja Das
- Soraya Rana Stevens

- Robert Heckendorn
- Cesar Guerra
- Laura Barbulescu
- Jean Paul Watson
- Monte Lunacek
- Rinku Dewri

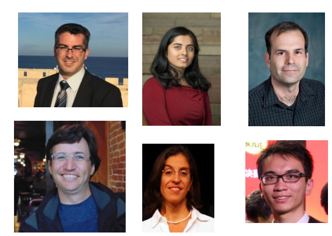
- Andrew Sutton
- Doug Hains
- Elmadhi Omar
- Wenxiang Chen
- Swetha Varadarajan
- Sachini Weerawardhana

3

3

Special Thanks to:

- Francisco Chicano, Swetha Varadarajan, Andrew Sutton, Renato Tinos, Gabriela Ochoa, Wenxiang Chen.



4

4

TUTORIAL

Next Generation Genetic Algorithms:
A User's Guide and Tutorial

Handbook of Metaheuristics
Springer, 2019

Email me:

darrell.whitley@gmail.com

SUBJECT: Tutorial

5

5

INTELLIGENT LOCAL SEARCH:

Intelligent
Iterated Local Search
is a very powerful search strategy for
many combinatorial optimization problems.

Tabu Search

Variable Neighborhood Search

Efficient Simulated Annealing

Generalized Pattern Search

Nelder-Mead

.

6

6

BLIND LOCAL SEARCH:

UNINTELLIGENT LOCAL SEARCH:

BLACK BOX LOCAL SEARCH

7

7

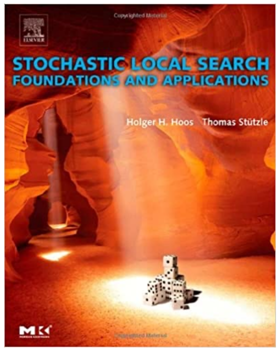
RANDOM MUTATION IS USELESS FOR MANY PROBLEM CLASSES:

MAX-SAT
NK-Landscapes
QUBO
All k-bounded Boolean/Pseudo Boolean functions
Traveling Salesman Problem
Graph Coloring
Many Constraint Satisfaction Problems

8

8

Intelligent Local Search:



Stochastic Local Search

Hoos and Stützle

9

9

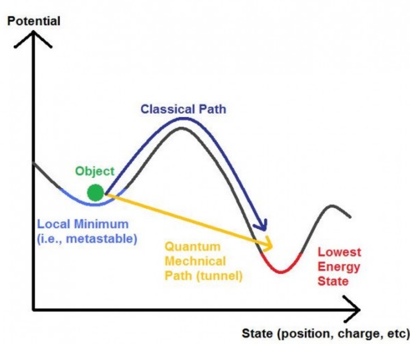
There are also important connections between QUANTUM COMPUTING and EVOLUTIONARY COMPUTATION.

1. The use of k-bounded Boolean functions.
   QUBO, Quadratic Unconstrained Boolean Optimization
2. Tunneling between optima.
3. The use of TRANSFORMS to control nonlinearity.
4. All of the binary results we will discuss are important in Quantum Computing as well.

10

10

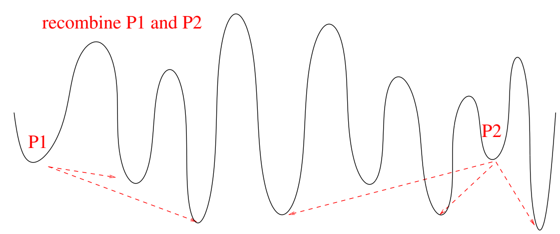
QUANTUM TUNNELING



11

11

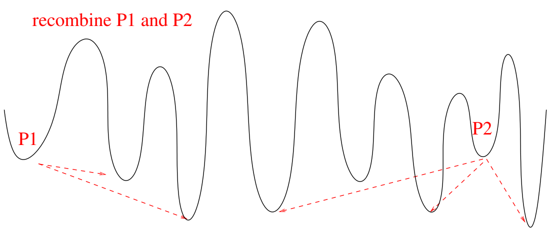
Crossover can *Deterministically* "Tunnel" Between Optima



12

12

Today, I will explain how Local Optima are arranged In Lattices

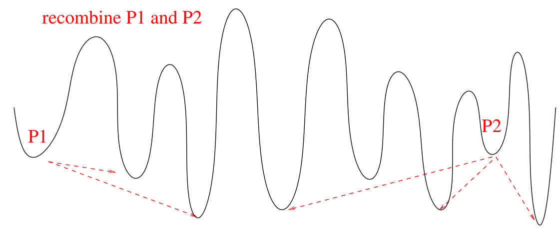


We can often remove randomness from Crossover

13

13

APPLY INTELLIGENT LOCAL SEARCH *BEFORE* CROSSOVER.

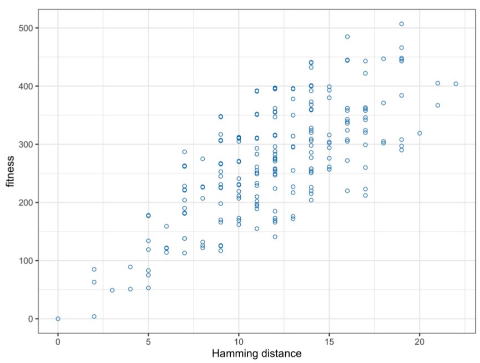


In some cases you can prove that recombination will not be as effective unless you do local search first.

14

14

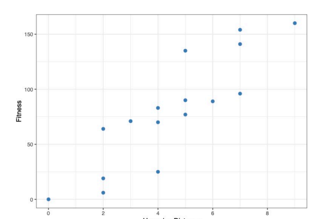
We enumerated N=30 functions. A "BIG VALLEY" Plot



15

15

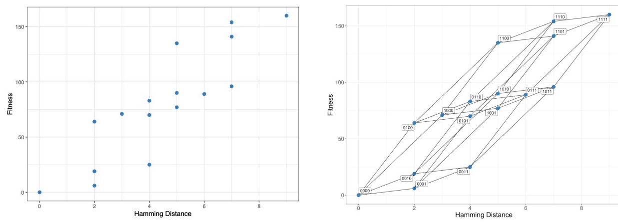
We enumerated N=30 functions. A "BIG VALLEY" Plot



16

16

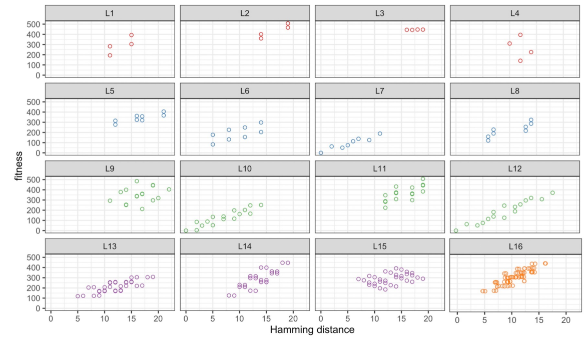
WE ENUMERATED N=30 FUNCTIONS. A "BIG VALLEY" PLOT



17

17

WE ENUMERATED N=30 FUNCTIONS. WE ENUMERATED ALL POSSIBLE CROSSOVERS

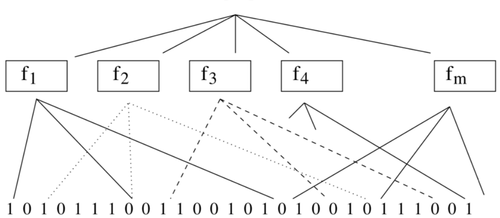


18

18

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS

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



19

19

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS: MAXSAT

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

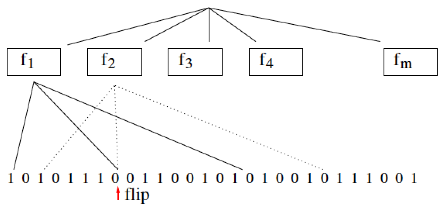
$$(\neg x_2 \vee x_1 \vee x_0) + (x_3 \vee \neg x_2 \vee x_1) + (x_3 \vee \neg x_1 \vee \neg x_0)$$

20

20

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS

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



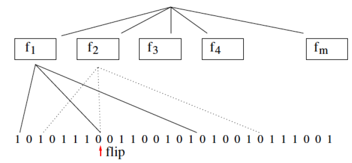
The location of *Improving Moves* can be computed on average in *constant* time. Special versions of this are known from 1992.
A general proof is given by: Whitley et al. 2013 AAAI.

21

21

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS

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



Worst case complexity is O(n) per move when m=O(n).

PROOF SKETCH: Create a function where variable x_j appears in every subfunction.

When x_j is flipped, the number of nonlinear interactions is O(n).

22

22

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS

The location of *Improving Moves* can be computed
on average in *constant* time. Whitley et al. 2013 AAAI.

SKETCH OF PROOF, AVERAGE CASE COMPLEXITY:

Assume m=O(n).
Flip each bit once. The average number of interactions must be O(1).

Nonlinearity is bounded by $m*2^k$ thus on average $(m*2/k)/n = (a*2^k) = O(1)$

23

23

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS

The location of *Improving Moves* can be computed
on average in *constant* time. Whitley et al. 2013 AAAI.

SKETCH OF PROOF, AVERAGE CASE COMPLEXITY:

Pick a constant C.
If a variable appears in less than C subfunctions, no problem.
When that variable is flipped it has O(1) interactions .

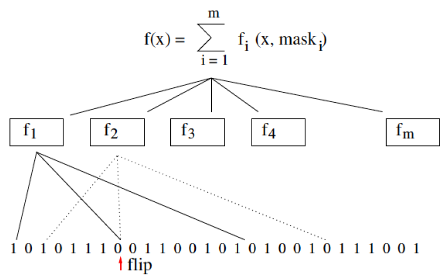
If a variable appears in more than C subfunctions,
the variable becomes Tabu after it is flipped.
You must wait N/C flips before it can be flipped again.

In practice, we never observed repeating (oscillating) high cost bit flips.

24

24

K-Bounded Pseudo-Boolean Functions



The MAXSAT community stopped using random blind local search 30 years ago (1992) but they still call it "Blackbox."

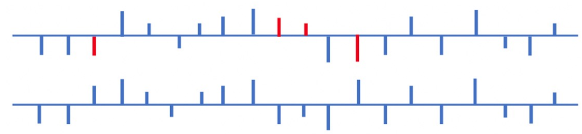
25

25

Improving Moves in Fourier/Walsh Space

$$f(x) = \psi_x w_1 + \psi_x w_2 + \psi_x w_3 + \psi_x w_4 + \psi_x w_5 + \psi_x w_6 + \psi_x w_7 + \psi_x w_8 + \psi_x w_{1,2} + \psi_x w_{2,3} + \psi_x w_{3,4} + \psi_x w_{1,4} + \psi_x w_{3,5} + \psi_x w_{5,6} + \psi_x w_{6,7} + \psi_x w_{5,7} + \psi_x w_{7,8} + \psi_x w_{8,4} + \psi_x w_{5,6,7} + \psi_x w_{4,7,8}$$

(Warning, the notation is compressed.)



26

26

Constant Time Improving Moves

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

$$Score(x, y_p) = f(y_p) - f(x)$$

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

All Walsh coefficients whose signs will be changed by flipping bit p are collected into a single number $Score(x, y_p)$.

See Hoos and Stützle, Stochastic Local Search, 2005

27

27

Constant Time Improving Moves

IMPROVING_MOVE_LIST: y_6, y_5

Flip 6, which interacts with 3 and 8, UPDATE.

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

IMPROVING_MOVE_LIST: y_8, y_5

28

28

Best Improving and Next Improving Moves have the same COST (Almost Always)!

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 speeds up GSAT by 30X

29

29

What about looking 2 or 3 Or 10 moves ahead?

With Thanks to Francisco Chicano

30

30

What about looking 2 or more moves ahead?

$$f(x) = \psi_x w_1 + \psi_x w_2 + \psi_x w_3 + \psi_x w_4 + \psi_x w_5 + \psi_x w_6 + \psi_x w_7 + \psi_x w_8 + \psi_x w_{1,2} + \psi_x w_{2,3} + \psi_x w_{3,4} + \psi_x w_{1,4} + \psi_x w_{3,5} + \psi_x w_{5,6} + \psi_x w_{6,7} + \psi_x w_{5,7} + \psi_x w_{7,8} + \psi_x w_{8,4} + \psi_x w_{5,6,7} + \psi_x w_{4,7,8}$$

Assume you have taken all single bit flip improving moves.

What happens when you flip bits 5 and 8 at the same time?

31

31

What about looking 2 or more moves ahead?

$$f(x) = \psi_x w_1 + \psi_x w_2 + \psi_x w_3 + \psi_x w_4 + \psi_x w_5 + \psi_x w_6 + \psi_x w_7 + \psi_x w_8 + \psi_x w_{1,2} + \psi_x w_{2,3} + \psi_x w_{3,4} + \psi_x w_{1,4} + \psi_x w_{3,5} + \psi_x w_{5,6} + \psi_x w_{6,7} + \psi_x w_{5,7} + \psi_x w_{7,8} + \psi_x w_{8,4} + \psi_x w_{5,6,7} + \psi_x w_{4,7,8}$$

Assume you have taken all single bit flip improving moves.

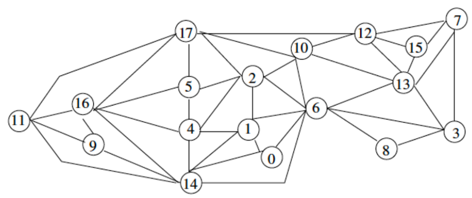
What happens when you flip bits 5 and 8 at the same time?

NOTHING. There are no nonlinear coefficients involving 5 and 8.

32

32

The Variable Interaction Graph

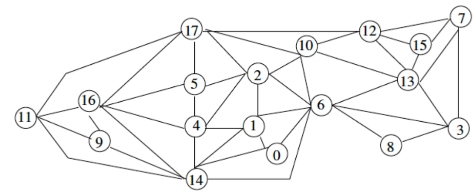


Can be constructed heuristically or exactly.

33

33

The Variable Interaction Graph



If two variables are not connected in the VIG,
there can be no improving move.

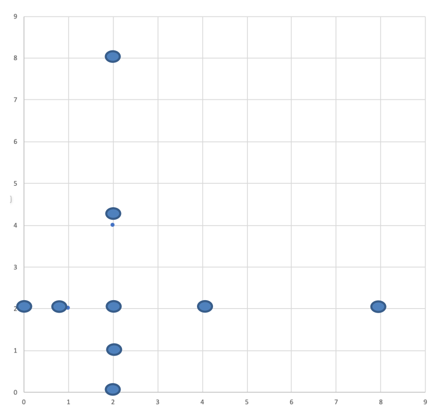
Assume you have taken all of the improving single bit flips.

What happens if you flip 16 and 1 at the same time? NOTHING.

34

34

The problem with 1 Bit Flip



35

35

What about looking 2 or more moves ahead?

$f_a(1,0,6)$
$f_b(2,1,6)$
$f_c(1,2,4)$
$f_d(4,1,14)$
$f_e(5,4,2)$

$f_l(6,10,13)$
$f_m(8,3,6)$
$f_n(7,12,15)$
$f_o(9,11,14)$
$f_p(10,2,17)$

$f_q(11,16,17)$
$f_r(12,10,17)$
$f_s(13,12,15)$
$f_t(14,4,16)$
$f_u(9,14,16)$

$f_v(15,7,13)$
$f_w(16,9,11)$
$f_x(17,5,16)$
$f_y(3,7,13)$
$f_z(0,6,14)$

36

36

What about looking 2 or more moves ahead?

$f_a(1,0,6)$	$f_l(6,10,13)$	$f_q(11,16,17)$	$f_v(15,7,13)$
$f_b(2,1,6)$	$f_m(8,3,6)$	$f_r(12,10,17)$	$f_w(16,9,11)$
$f_c(1,2,4)$	$f_n(7,12,15)$	$f_s(13,12,15)$	$f_x(17,5,16)$
$f_d(4,1,14)$	$f_o(9,11,14)$	$f_t(14,4,16)$	$f_y(3,7,13)$
$f_e(5,4,2)$	$f_p(10,2,17)$	$f_u(9,14,16)$	$f_z(0,6,14)$

37

37

What about looking 2 or more moves ahead?

$f_a(1,0,6)$	$f_l(6,10,13)$	$f_q(11,16,17)$	$f_v(15,7,13)$
$f_b(2,1,6)$	$f_m(8,3,6)$	$f_r(12,10,17)$	$f_w(16,9,11)$
$f_c(1,2,4)$	$f_n(7,12,15)$	$f_s(13,12,15)$	$f_x(17,5,16)$
$f_d(4,1,14)$	$f_o(9,11,14)$	$f_t(14,4,16)$	$f_y(3,7,13)$
$f_e(5,4,2)$	$f_p(10,2,17)$	$f_u(9,14,16)$	$f_z(0,6,14)$

If the number of subfunctions is $m=O(n)$
The number of pairs must be $O(n)$ and not $O(n^2)$

Thus the number of possible improving moves is $O(n)$
And less than $m*2^k$

38

38

What about looking 2 or more moves ahead?

$f_a(1,0,6)$	$f_l(6,10,13)$	$f_q(11,16,17)$	$f_v(15,7,13)$
$f_b(2,1,6)$	$f_m(8,3,6)$	$f_r(12,10,17)$	$f_w(16,9,11)$
$f_c(1,2,4)$	$f_n(7,12,15)$	$f_s(13,12,15)$	$f_x(17,5,16)$
$f_d(4,1,14)$	$f_o(9,11,14)$	$f_t(14,4,16)$	$f_y(3,7,13)$
$f_e(5,4,2)$	$f_p(10,2,17)$	$f_u(9,14,16)$	$f_z(0,6,14)$

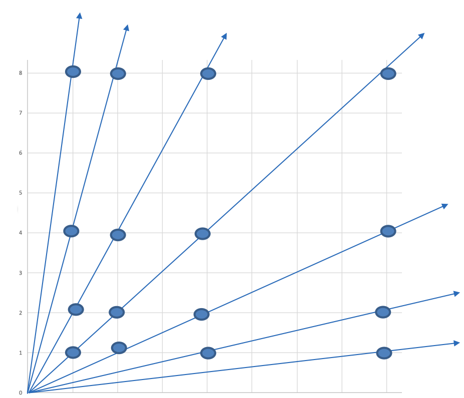
0,6	0,14	1,0	1,2	1,4	1,6	1,14	2,4
2,5	2,6	2,10	2,17	3,6	3,7	3,8	3,13
4,5	4,14	4,16	5,16	5,17	6,8	6,10	6,13
6,14	7,12	7,13	7,15	9,11	9,14	9,16	10,12
10,13	10,17	11,14	11,16	11,17	12,13	12,15	12,17
13,15	14,16	16,17					

The number of pairs must be linear

39

39

What about looking 2 moves ahead?



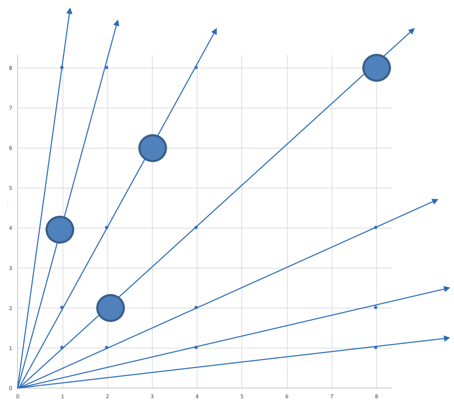
Consider a 2-D Rosenbrock Function

2 moves ahead looks along diagonals

40

40

What about looking 2 moves ahead?



ONLY look at potential improving moves.

On average, we can determine this set in CONSTANT TIME

41

41

What about looking 3 or more moves ahead?

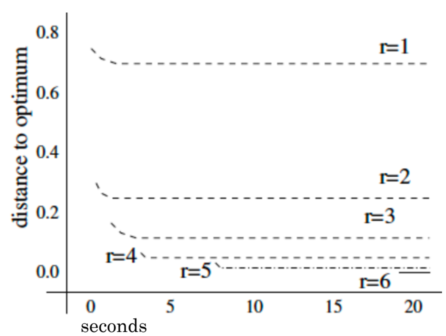
$f_a(1,0,6)$	$f_l(6,10,13)$	$f_q(11,16,17)$	$f_v(15,7,13)$
$f_b(2,1,6)$	$f_m(8,3,6)$	$f_r(12,10,17)$	$f_w(16,9,11)$
$f_c(1,2,4)$	$f_n(7,12,15)$	$f_s(13,12,15)$	$f_x(17,5,16)$
$f_d(4,1,14)$	$f_o(9,11,14)$	$f_t(14,4,16)$	$f_y(3,7,13)$
$f_e(5,4,2)$	$f_p(10,2,17)$	$f_u(9,14,16)$	$f_z(0,6,14)$

There are approximately 3n = 60 potential 3-flips, NOT (n choose 3) = 816.

42

42

Intelligent Local Search looking 2, 3, 4, 5, 6 bits ahead

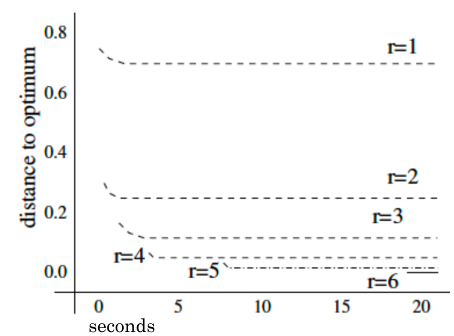


Adjacent NK Landscape N=12,000, K=2 (k=3).

43

43

Intelligent Local Search looking 2, 3, 4, 5, 6 bits ahead



N=12,000, k=3. For radius r = 6, 100% globally optimal.

44

44

K bounded Functions: MAXSAT

a: 1 -0 6	l: -6 10 13	q: -11 16 17	v: -15 -7 -13
b: 2 -1 6	m: 8 -3 6	r: 12 -10 17	w: 16 -9 -11
c: -1 2 4	n: 7 -12 -15	s: -13 -12 15	x: 17 -5 -16
d: -4 1 14	o: 9 11 14	t: 14 -4 16	y: -3 -7 13
e: -5 4 2	p: -10 -2 17	u: -9 14 16	z: 0 6 -14

45

45

What about Recombination?

$f_a(1,0,6)$	$f_l(6,10,13)$	$f_q(11,16,17)$	$f_v(15,7,13)$
$f_b(2,1,6)$	$f_m(8,3,6)$	$f_r(12,10,17)$	$f_w(16,9,11)$
$f_c(1,2,4)$	$f_n(7,12,15)$	$f_s(13,12,15)$	$f_x(17,5,16)$
$f_d(4,1,14)$	$f_o(9,11,14)$	$f_t(14,4,16)$	$f_y(3,7,13)$
$f_e(5,4,2)$	$f_p(10,2,17)$	$f_u(9,14,16)$	$f_z(0,6,14)$

We could consider an NK-Landspace

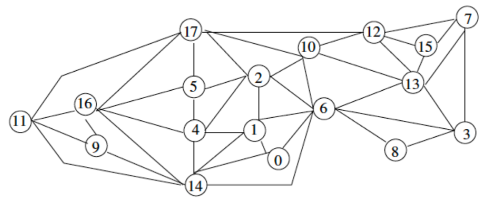
The variables interactions are the same.

Note we have named the subfunctions: a to z.

46

46

The Variable Interaction Graph

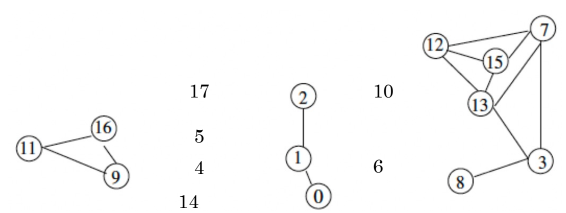


LOCAL OPTIMUM P1: 000000000000000000
LOCAL OPTIMUM P2: 111100011101110110

47

47

The Recombination Graph: Parent 1: 000000000000000000 Parent 2: 111100011101110110



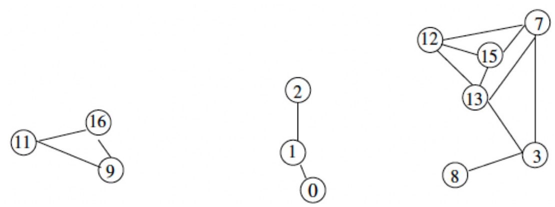
q, o, u, w, x, t   a, b, c, d, e, p, z   l, m, n, r, s, v, y

Delete vertices: 4, 5, 6, 10, 14, 17

48

48

The Recombination Graph: the Decomposed VIG.



q, o, u, w, x, t

a, b, c, d, e, p, z

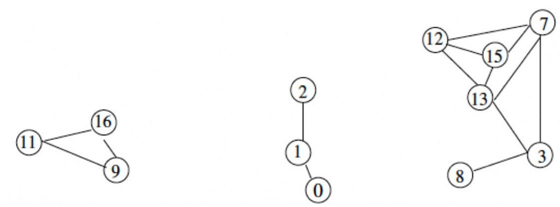
l, m, n, r, s, v, y

This decomposes the variables **and** the subfunctions.

49

49

The Recombination Graph: Be GREEDY



q, o, u, w, x, t

a, b, c, d, e, p, z

l, m, n, r, s, v, y

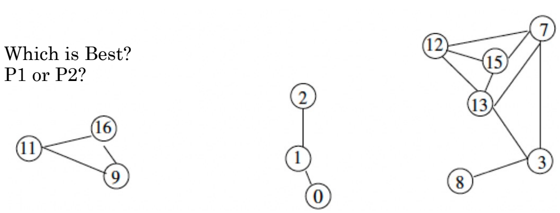
This decomposes the variables **and** the subfunctions.

50

50

The Recombination Graph: Be GREEDY

Which is Best?
P1 or P2?



q, o, u, w, x, t

a, b, c, d, e, p, z

l, m, n, r, s, v, y

This decomposes the variables **and** the subfunctions.

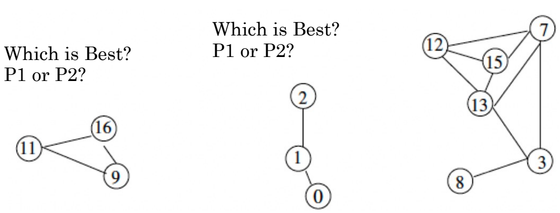
51

51

The Recombination Graph: Be GREEDY

Which is Best?
P1 or P2?

Which is Best?
P1 or P2?



q, o, u, w, x, t

a, b, c, d, e, p, z

l, m, n, r, s, v, y

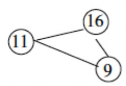
This decomposes the variables **and** the subfunctions.

52

52

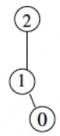
The Recombination Graph: Be GREEDY

Which is Best?
P1 or P2?



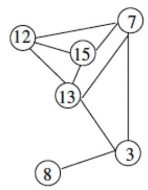
q, o, u, w, x, t

Which is Best?
P1 or P2?



a, b, c, d, e, p, z

Which is Best?
P1 or P2?



l, m, n, r, s, v, y

Partition Crossover deterministically returns the *best* of 2^q offspring.

53

53

Partition Crossover and Local Optima.

The Subspace Optimality Theorem:

For any k-bounded pseudo-Boolean function, *f:*

If the parents are local optima,
then all offspring are local optima
in the smallest hyperplane subspace
that contains the two parents.

54

54

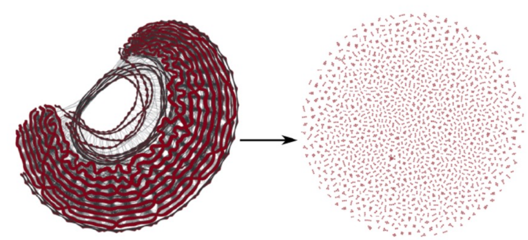
What does the VIG and Recombination Graph look like on Real World Problems?

55

55

The VIG

The Recombination Graph.



atco_enc3_opt1_13_48

Air traffic controller shift scheduling problem: 1087 components.

PX returns the best of 2^{1087} offsprings.

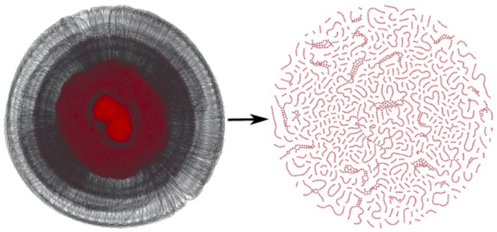
N = 1,067,657

(Thanks to Wenxiang Chen)

56

56

More MAXSAT



LABS_n088_goal008

Finding low autocorrelation binary sequence: 371 components

PX returns the best of 2^{371} offsprings.

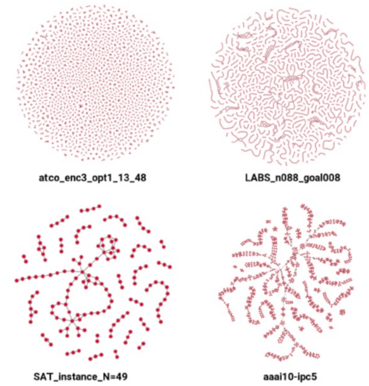
N= 182,015

(Thanks to Wenxiang Chen)

57

57

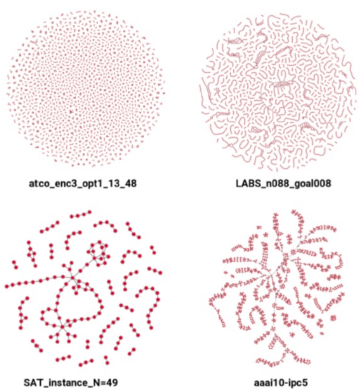
More MAXSAT



58

58

More MAXSAT



These subproblems have a tree decomposition with low width.

Thanks to Francisco Chicano.

These subproblems can be solved by Dynamic Programming!

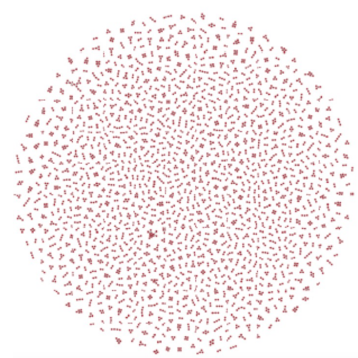
59

59

Decomposed Evaluation for MAXSAT

Crossover returns the Best of 2^{1087} offspring.

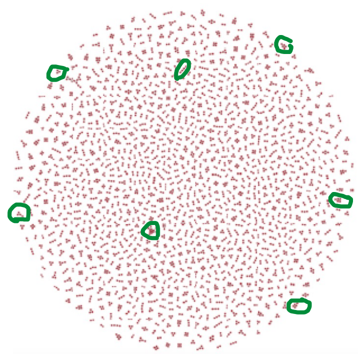
All offspring are Local Optima in this subspace.



60

60

WE ONLY CARE ABOUT COMPONENTS WITH UNSAT CLAUSES.



61

61

PARTITION CROSSOVER AND LOCAL OPTIMA.

The Subspace Optimality Theorem:

For any k-bounded pseudo-Boolean function, *f*:

If the parents are local optima,
then all offspring are local optima
in the smallest hyperplane subspace
that contains the two parents.

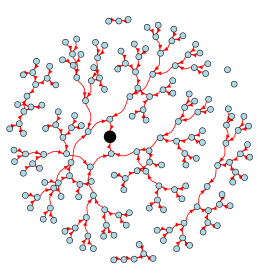
TUNNELING BETWEEN OPTIMA in O(N) time.

62

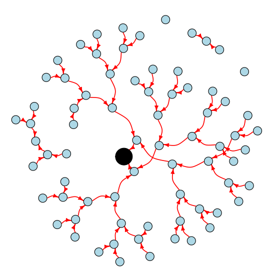
62

TUNNELING BETWEEN LOCAL OPTIMA.

Local Optima Linked by Crossover, Thanks to Gabriela Ochoa.



Adjacent NK Landscape



Random NK Landscape

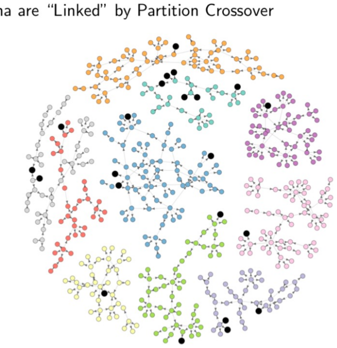
63

63

The Traveling Salesman

Tunneling Between Local Optima

Local Optima are "Linked" by Partition Crossover



These were found using Chained-LK.

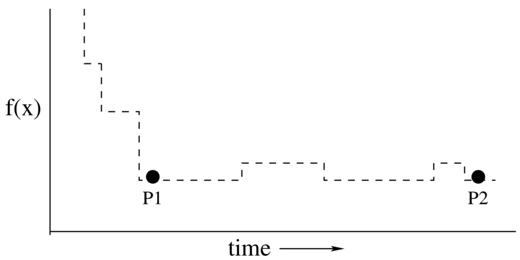
But it could have been Lin Kernighan Helsgaun (LKH).

Thanks to G. Ochoa and N. Veerapen.

64

64

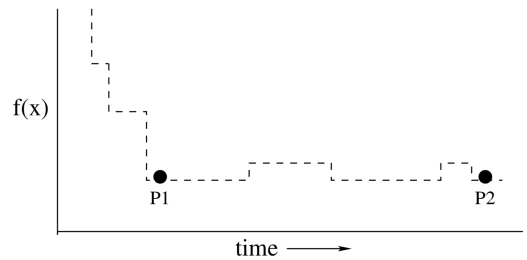
MAX-3SAT AND PLATEAUS



65

65

MAX-3SAT AND PLATEAUS



RUN LOCAL SEARCH FIRST, THEN APPLY CROSSOVER.
There is NO POPULATION.

66

66

Local Search Algorithms for MAXSAT

AdaptG^2WSAT: Best in the 2007 SAT Competition

NEW: AdaptG^2WSAT with Partition Crossover

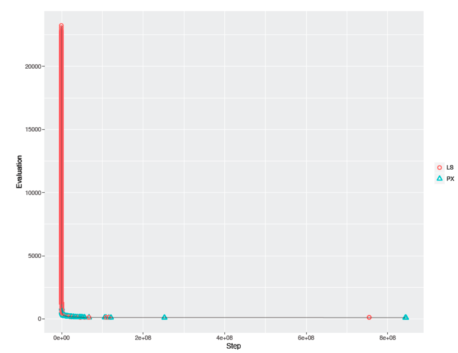
Sparrow: Best among all local search over in "crafted" and "Application" SAT Track in 2014 SAT Competition.

NEW: Sparrow with Partition Crossover

67

67

Early MAXSAT Results

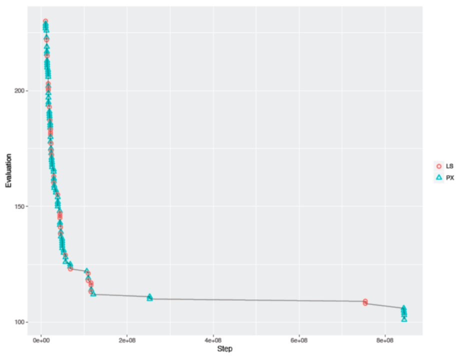


RED is intelligent local search. BLUE is Partition Crossover

68

68

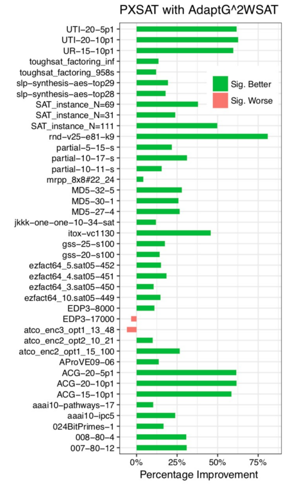
Early MAXSAT Results



RED is intelligent local search. BLUE is Partition Crossover

69

69



MAXSAT Results.

PARTITION CROSSOVER HELPS ON HARD PROBLEMS.

70

70

MAXSAT RESULTS

Theorem

When recombining parents P_1 and P_2:

$$\frac{f(P_1)}{2} + \frac{f(P_2)}{2} = \frac{1}{2^q}\sum_{i=1}^{2^q} f(C_i)$$

Corollary

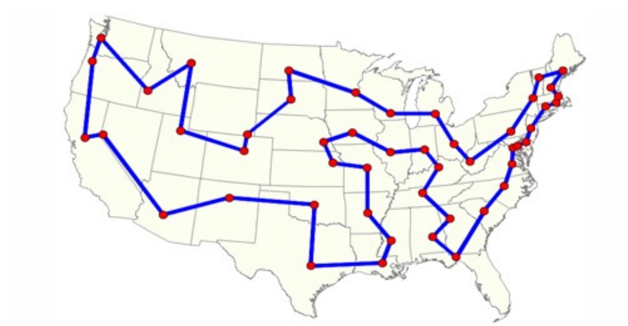
Assume that $f(P1) = f(P2)$.
If **any** *offspring represents a disimproving move, there must also exist an offspring that yields an improving move.*

This makes Partition Crossover very different than local search for MAXSAT. For local search the discovery of a disapproving move says nothing about the existence of an improving move.

71

71

THE TRAVELING SALESMAN PROBLEM

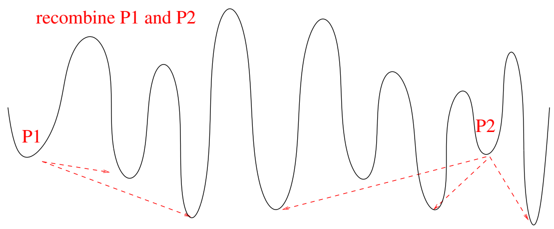


What is the shortest circuit that visits the 50 state capitals?

72

72

Partition Crossover ***Deterministically*** "Tunnels" Between Optima

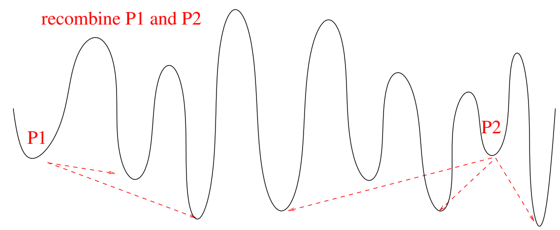


We can remove randomness from Crossover

73

73

First Apply Intelligent Local Search!



74

74

First Apply Intelligent Local Search!

Naïve 2-Opt is O(N^3) in complexity!

Intelligent 2-Opt is O(N).

1) Intelligent evaluation by partial evaluation.
2) Use of Nearest Neighbor moves.
3) Use of "Don't Look Bits"

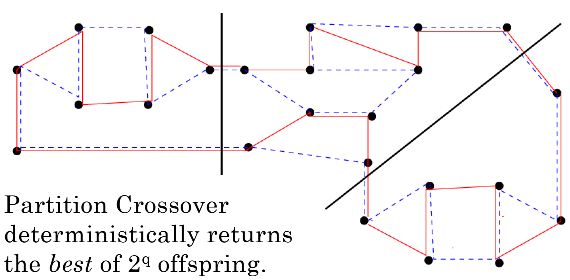
THIS IS NOT BLACK BOX.

75

75

Can We ``Tunnel" between Optima?

Assume the Parents are Local Optima (***under ANY Operator***).

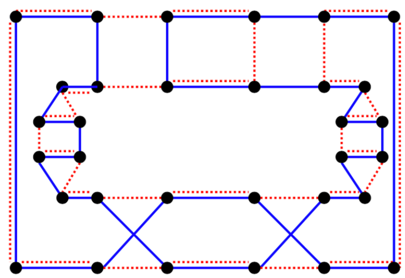


Partition Crossover deterministically returns the *best* of 2^q offspring.

76

76

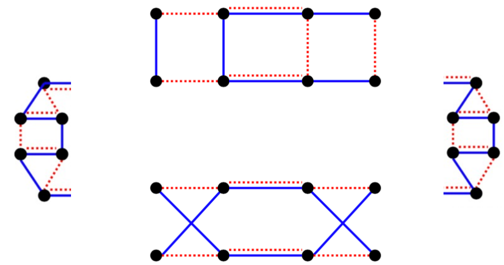
Partition Crossover and TSP



77

77

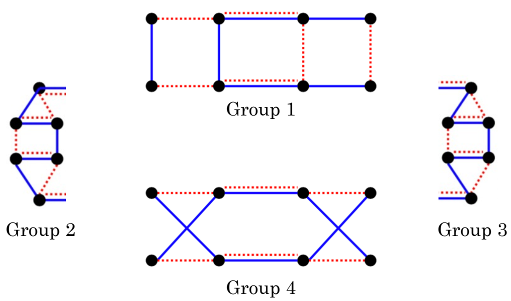
Partition Crossover and TSP



78

78

Partition Crossover and TSP



79

79

The quasi-local optima form a lattice in hyperspace:

Assume you have these connected groups of variables during recombination.

Group 1: v1, v2, v4, v5, v7, v9

Group 2: v11, v13, v14, v15, v17, v18

Group 3: v20, v21, v23, v26, v27, v28

Group 4: v32, v33, v34, v35, v36, v39

80

80

The quasi-local optima form a lattice in hyperspace:

Assume you have these connected groups of variables during recombination.

Group 1:	v1, v2, v4, v5, v7, v9	Parent 1 or Parent 2?
Group 2:	v11, v13, v14, v15, v17, v18	Parent 1 or Parent 2?
Group 3:	v20, v21, v23, v26, v27, v28	Parent 1 or Parent 2?
Group 4:	v32, v33, v34, v35, v36, v39	Parent 1 or Parent 2?

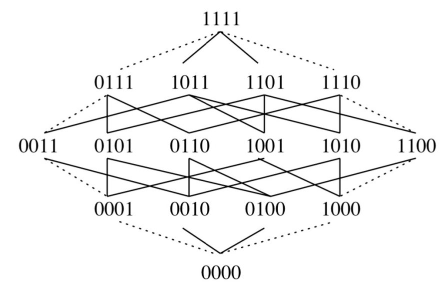
Partition Crossover returns the best of 2^4 = 16 solutions.

81

81

The quasi-local optima form a lattice in hyperspace:

Group 1. Group 2. Group 3. Group 4.

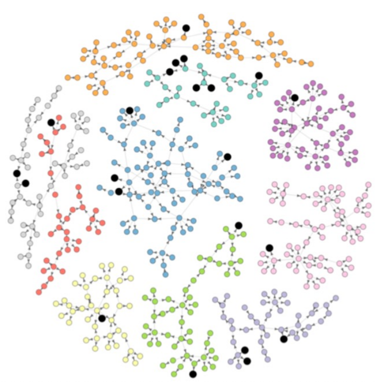


ALL of the 16 solutions are LOCAL OPTIMA In the Hyperplane Subspace.

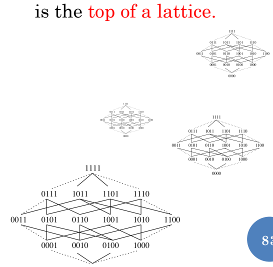
82

82

These tunnels are just the **TOPS** of lattices of quasi local optima.



Each tunnel is one recombination, and each recombination is the top of a lattice.



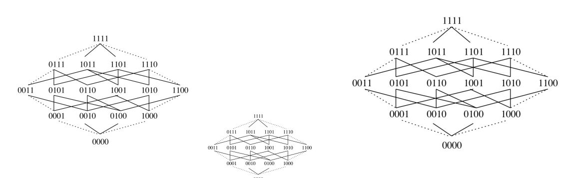
83

83

THEOREM: A lattice of quasi-local optima can be exponentially Large:

PROOF BY CONSTRUCTION: Construct a traveling salesman problem (or MAXSAT instance) over N vertices such that it has two local optima, and these two local optima decompose into N/c recombining components for some constant c.

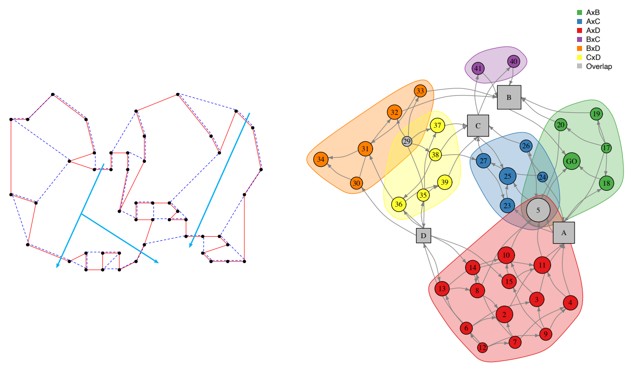
This results in a lattice of size $2^{N/c}$



84

84

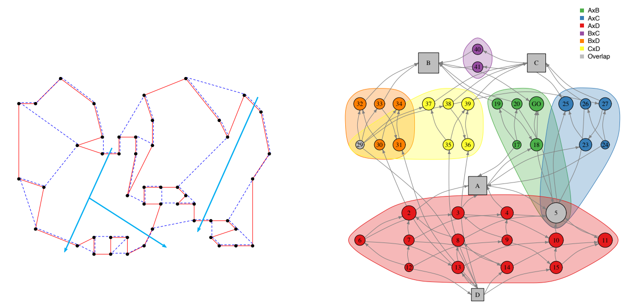
LOCAL OPTIMA ORGANIZE INTO LATTICES UNDER RECOMBINATION



85

85

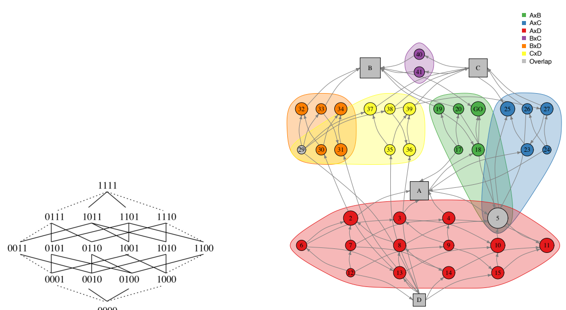
LOCAL OPTIMA ORGANIZE INTO LATTICES UNDER RECOMBINATION



86

86

LOCAL OPTIMA ORGANIZE INTO LATTICES UNDER RECOMBINATION



87

87

PARTITION CROSSOVER INDUCES A LINEAR EQUATION THAT EVALUATES CHILDREN (WHICH INCLUDES LOCAL OPTMA).

THEOREM 3.1. *Under Partition Crossover, all of the 2^q children of parents P1 and P2 can be evaluated using the following linear equation and the auxiliary bit function b.*

$$f(x) = g(x') + c = \alpha_0 + \sum_{i=1}^{q} \alpha_i b_i \qquad (3)$$

where $\alpha_0 = f(P2)$.

Where string b is a crossover mask over "components"

88

88

Partition Crossover Induces a Linear Equation that evaluates Children (which includes Local Optma).

Theorem 3.1. *Under Partition Crossover, all of the 2^q children of parents P1 and P2 can be evaluated using the following linear equation and the auxiliary bit function b.*

$$f(x) = g(x') + c = \alpha_0 + \sum_{i=1}^{q} \alpha_i b_i \quad (3)$$

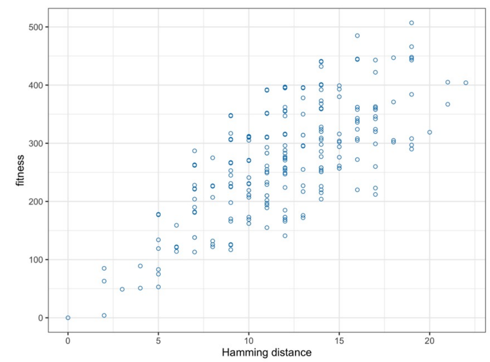
where $\alpha_0 = f(P2)$.

$$f_r(x_1, x_2, x_4, x_8) = w_0 + \psi_1(x)w_1 + \psi_2(x)w_2 + \psi_4(x)w_4 + \psi_8(x)w_8$$

89

89

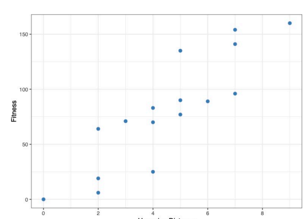
We enumerated N=30 functions. A "BIG VALLEY" Plot



90

90

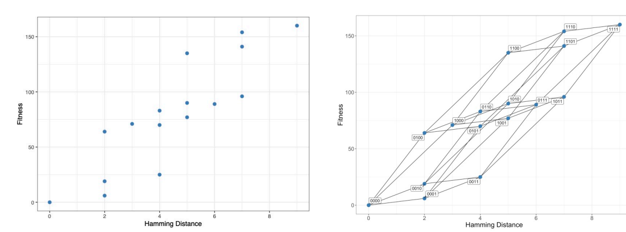
We enumerated N=30 functions. A "BIG VALLEY" Plot



91

91

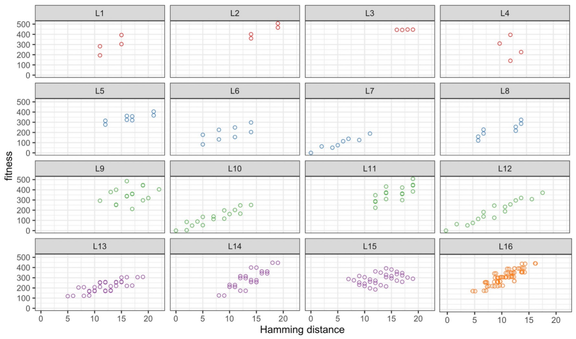
We enumerated N=30 functions. A "BIG VALLEY" Plot



92

92

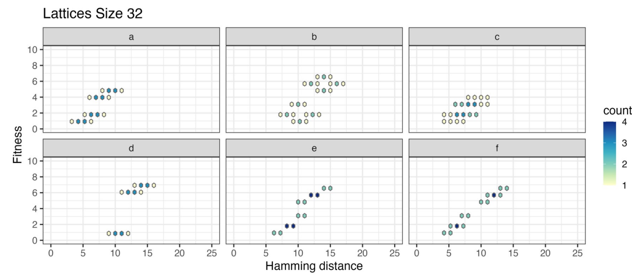
We enumerated N=30 functions. We enumerated all possible Crossovers



93

93

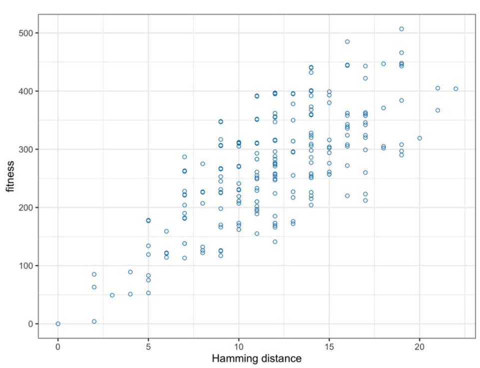
We enumerated N=30 functions. MAXSAT Lattices of Size 32



94

94

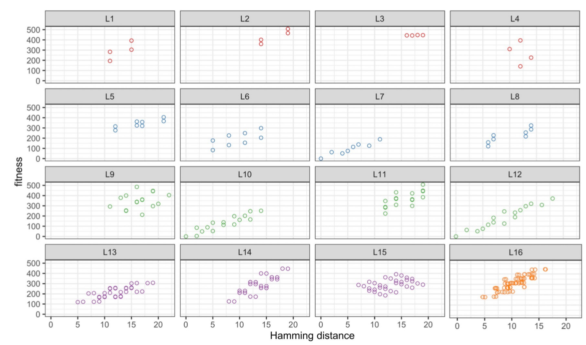
We enumerated N=30 functions. A "BIG VALLEY" Plot



95

95

We enumerated N=30 functions. We enumerated all possible Crossovers



96

96

Partition Crossover Induces a Linear Equation that evaluates Children (which includes Local optma).

Theorem 3.1. *Under Partition Crossover, all of the 2^q children of parents P1 and P2 can be evaluated using the following linear equation and the auxiliary bit function b.*

$$f(x) = g(x') + c = \alpha_0 + \sum_{i=1}^{q} \alpha_i b_i \qquad (3)$$

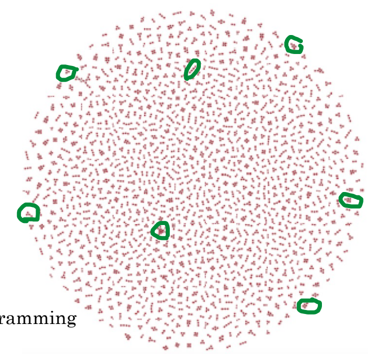
where $\alpha_0 = f(P2)$.

Where string b is a crossover mask over "components"

97

97

We only care about COMPONENTS with UNSAT clauses.



Optimal Crossover
With Dynamic Programming

98

98

Transforms

- SAT to MAXSAT
  - For decades, SAT problems have been converted into MAX-3SAT instances. Modern SAT solvers expect a MAXSAT form.
  - TRANSFORMS may also serve as REDUCTIONS used to prove NP-Completeness.

99

99

Transforms

- Transforms exist for all Pseudo-Boolean Functions

"All pseudo-Boolean optimization problems can be *reduced* to the quadratic case." Boros and Hammer (2002):186

This assumes a polynomial evaluation function.

The transformed function is polynomial in size relative to the original function.

100

100

Transforms

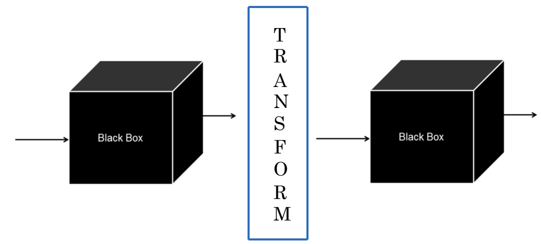
A PROJECTION INTO A HIGHER DIMENSION
WITH LOWER NON-LINEARITY

TRANSFORMS are regularly used to
Covert Boolean Problems into
QUBO
For Quantum Computing.

101

101

Transforms can be Quasi-Black Box (But Not Really).



The quadratic function is recovered by sampling in $O(n^2)$ time.

102

102

Transforms

For example, you could convert a NK landscape where

N = 10,000, K = 9 (k=10)

Into an NK landscape where

N= 50,000, K=1 (k=2),

A PROJECTION INTO A HIGHER DIMENSION
WITH LOWER NON-LINEARITY

103

103

The Multilinear form

$w_0 = f(000)$
$w_1 = f(001) - w_0$
$w_2 = f(010) - w_0$
$w_3 = f(011) - w_0 - w_1 - w_2$
$w_4 = f(100) - w_0$
$w_5 = f(101) - w_0 - w_4 - w_1$
$w_6 = f(110) - w_0 - w_4 - w_2$

$f(x) = w_0 + w_1x_1 + w_2x_2 + w_4x_4 + w_3x_1x_2 + w_5x_1x_4 + w_6x_2x_4$

104

104

Transforms

- Assume our function is written in Multilinear Form (helpful, but not necessary)

$$f(x_1, \ldots, x_n) = \sum_{S \subseteq V} c_S \prod_{j \in S} x_j$$

Where
x_i is a Boolean Variable, c_S is a weight (constant)
S is a single variable
or a subset of variables with nonlinear interactions
And j indexes a variable in S.

105

105

Leading Ones as a Multilinear Form

- The n=4 case.

$$f(x_1, x_2, x_3, x_4) = x_1 + x_1x_2 + x_1x_2x_3 + x_1x_2x_3x_4$$

- A general example

f(01111111111111111) = 0
f(10011001100011111) = 1
f(11111000011000111) = 5

106

106

Leading Ones as a Multilinear Form

- The n=4 case.

$$f(x_1, x_2, x_3, x_4) = x_1 + x_1x_2 + x_1x_2x_3 + x_1x_2x_3x_4$$

- The Walsh Polynomial is EXPONENTIAL in Size!

107

107

Leading Ones as a Multilinear Form

- The n=4 case.

$$f(x_1, x_2, x_3, x_4) = x_1 + x_1x_2 + x_1x_2x_3 + x_1x_2x_3x_4$$

- In general for maximization.

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$$

- In general for minimization.

$$f(x_1, \ldots, x_n) = n + \sum_{i=1}^{n} -1 \prod_{j=1}^{i} x_j$$

108

108

The Basic Substitution: 1 variable replaces 2 Variables

$$x_1x_2 = z \text{ iff } x_1x_2 - 2x_1z - 2x_2z + 3z = 0$$

$$x_1x_2 \neq z \text{ iff } x_1x_2 - 2x_1z - 2x_2z + 3z > 0$$

Substitution with Penalty Constraints

$$P(x_1x_2 - 2x_1z - 2x_2z + 3z)$$

P is the sum of $|C_S|$

109

109

For Leading Ones n=4, p=5

$$f(x_1, x_2, x_3, x_4) = 4 - x_1 - x_1x_2 - x_1x_2x_3 - x_1x_2x_3x_4$$

$$f(x_1, x_2, x_3, x_4, z_1) = 4 - x_1 - x_1x_2 - z_1x_3 - z_1x_3x_4 + 5x_1x_2 - 10x_1z_1 - 10x_2z_1 + 15z_1$$

$$f(x_1, x_2, x_3, x_4, z_1, z_2) = 4 - x_1 - x_1x_2 - z_1x_3 - z_2x_4 + 5x_1x_2 - 10x_1z_1 - 10x_2z_1 + 15z_1 + 5z_1x_3 - 10z_1z_2 - 10x_3z_2 + 15z_2$$

110

110

Leading Ones in General:

$$f(x, z) = n - x_1 - z_{n-2}x_n + \sum_{i=1}^{n-2} f_i(z_i, z_{i-1}, x_{i+1})$$

$$f_i(z_i, z_{i-1}, x_{i+1}) = -z_{i-1}x_{i+1} + P(z_{i-1}x_{i+1} - 2z_iz_{i-1} - 2z_ix_{i+1} + 3z_i)$$

The Walsh Polynomial is LINEAR in size.

111

111

Leading Ones in General:

$$f(x, z) = n - x_1 - z_{n-2}x_n + \sum_{i=1}^{n-2} f_i(z_i, z_{i-1}, x_{i+1})$$

$$f_i(z_i, z_{i-1}, x_{i+1}) = -z_{i-1}x_{i+1} + P(z_{i-1}x_{i+1} - 2z_iz_{i-1} - 2z_ix_{i+1} + 3z_i)$$

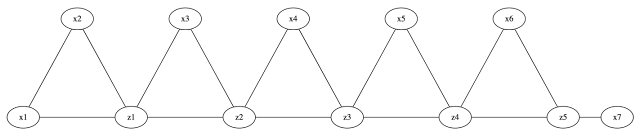
One can optimize each subfunction z_i=1, z_{i-1}=1, x_{i+1}=1 to obtain the global optimum in O(n) time.

You can use **Local Search with Lookahead** to solve the problem in O(n) time.

112

112

Leading Ones:
The New Variable Interaction Graph



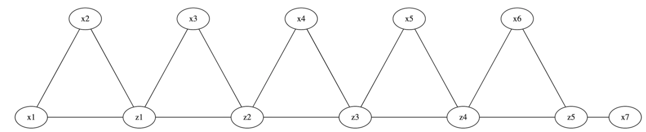
RESULT:

Quadratic Leading Ones problem has a bounded tree width w < 4 and is solved by Dynamic Programming in O(n) time.

113

113

Leading Ones:
The New Variable Interaction Graph



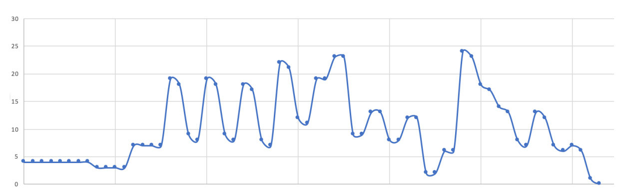
The same **Variable Interaction Graph** holds for all **Hidden Permutations** versions of Leading Ones.

x(i) is just replaced with x(π(i))

114

114

For Leading Ones
N=4



Every penalized point in f(x,z) is greater that every point in f(x).

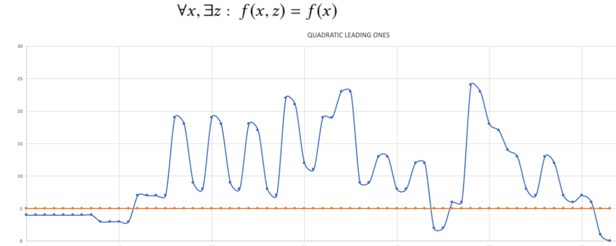
115

115

GENERAL RESULT

THEOREM 1:. *Let $f(x)$ be a multilinear pseudo-Boolean function, and let $f(x,z)$ be a k-bounded pseudo-Boolean function produced by replacing variables in $f(x)$ with auxiliary variables in vector z. Index the variables in z and let z^j denote the first j variables in z. If z_{j+1} only replaces variables in x and z^j then:*

$$\forall x, \exists z : f(x,z) = f(x)$$



116

116

A More Recent Transform:

x1 + x1x2 + x1x2x3 + x1x2x3x4 + x1x2x3x4x5

Transforms to:

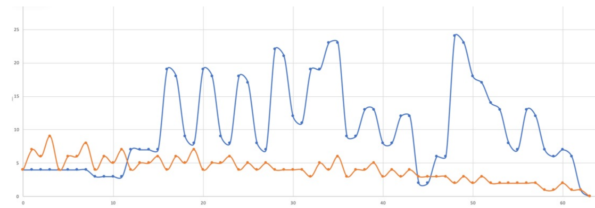
x1 + x1x2
x1y1 + x2y1 + x3y1 -2y1
x1y2 + x2y2 + x3y2 + x4y2 -3y2
x1y3 + x2y3 + x3y3 + x4y3 + x5y3 -4y3

This transform can reveal submodular functions:
Leading Ones and Hidden Permutations are **submodular**

117

117

A More Recent Transform:



This transform can reveal submodular functions:
Leading Ones and Hidden Permutations are submodular

118

118

Transforms can change the "Query" (Runtime) Complexity

Consider: LeadingOnes(x) + ONEMAX ($x_{n/2} \ldots x_n$)

LeadingOnes($x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}$)

\+ ONEMAX($x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}$)

This function is solved in Θ(n^2) queries by Random Local Search.

It can be solved in O(n lg n) queries after
it is transformed into Quadratic form.

It is also still ***Submodular***, of course.

119

119

One Last Thought:

What if DNA is K-bounded?

E.g., the fitness landscape is an NK-Landscape

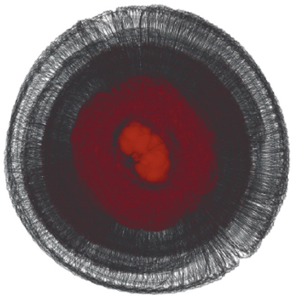
120

120

One Last Thought:

What if DNA is K-bounded?

What if "gene interaction" looks like this?



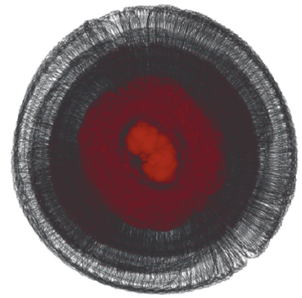
121

121

One Last Thought:

What if DNA is K-bounded?

99.9% of DNA is identical in all humans



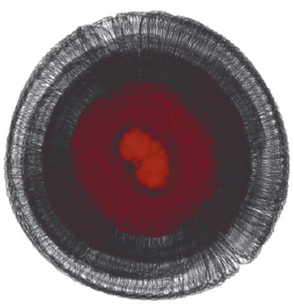
122

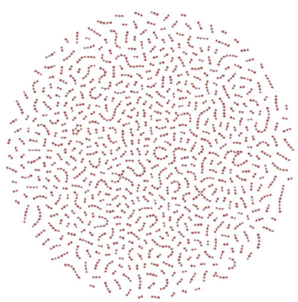
122

One Last Thought:

What if DNA is K-bounded?

99.9% of DNA is identical in all humans





123

123

Questions?

124

124