

Next Generation Genetic Algorithms

Darrell Whitley
Computer Science, Colorado State University

With Thanks to: F. Chicano, G. Ochoa, A. Sutton and R. Tinós

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

GECCO '19 Companion, Genetic and Evolutionary Computation Conference Companion, July 13–17, 2019, Prague, Czech Republic
2019 Copyright is held by the owner/author(s).
ACM ISBN 978-1-4503-6748-6/19/07
<https://doi.org/10.1145/3319619.3323387>

1

Next Generation Genetic Algorithms

There is a book chapter that goes with this tutorial.

D. Whitley.
Next Generation Genetic Algorithms.
The Handbook of Metaheuristics, 2019.

Send an email to whitley@cs.colostate.edu

SUBJECT: TUTORIAL2019

2

Hot Off the Press!!!

We can define a new form of Genetic Algorithm
where Premature Convergence cannot occur!

Furthermore, larger populations converge faster, not slower.

The Mixing Genetic Algorithm

Swetha Vardarajan and Darrell Whitley

3

Next Generation Genetic Algorithms

What do we mean by “Next Generation?”

- ① NOT a Black Box Optimizer.
- ② Uses mathematics to characterize problem structure.
- ③ For many problems: NO MUTATION IS NEEDED

4

Next Generation Genetic Algorithms

What do we mean by “Next Generation?”

- ① Not a blind “population, selection, mutation, crossover” GA.
- ② Uses deterministic move operators and crossover operators
- ③ *Tunnels* between Local Optima.
- ④ Scales to large problems with millions of variables.
- ⑤ Build on our expertise in intelligent ways.

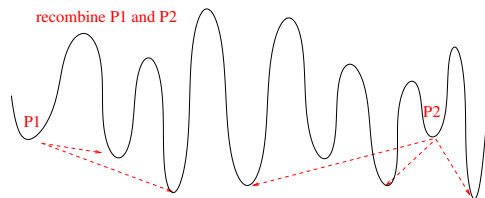
5

Know your Landscape! And Go Downhill!



6

What if you could ...



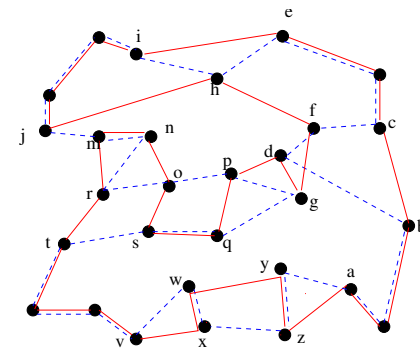
“Tunnel” between local optima on a TSP,
or on an NK Landscape or a MAXSAT problem
and go the BEST reachable local optima!

Tunneling = jump from local optimum to local optimum

7

The Partition Crossover Theorem for TSP

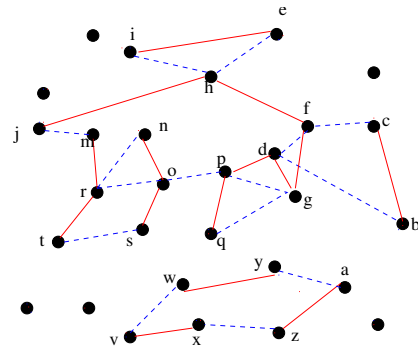
Let G be a graph produced by unioning 2 Hamiltonian Circuits.



8

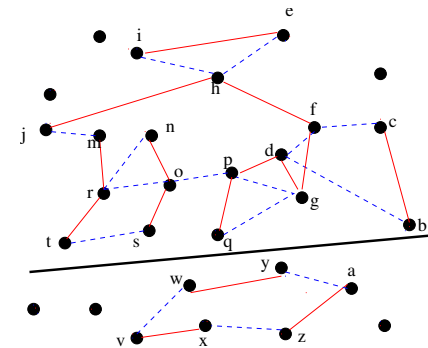
The Partition Crossover Theorem for TSP

Let G' be a reduced graph so that all common subtours are removed.



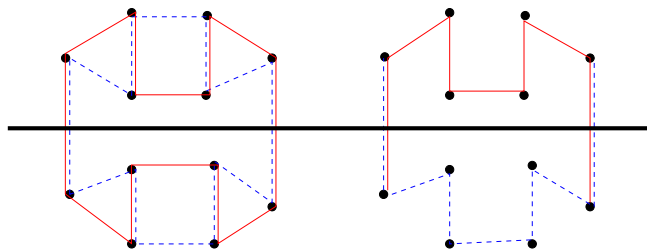
9

Partition Crossover in $O(n)$ time



10

The Partition Crossover for TSP

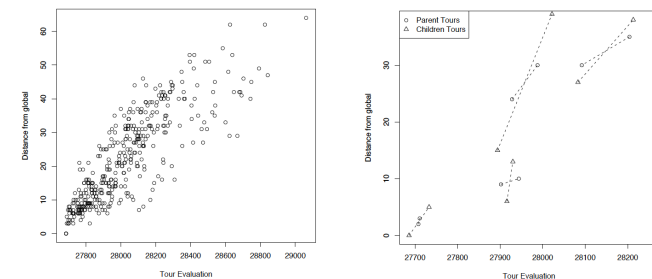


As a side effect: $f(P1) + f(P2) = f(C1) + f(C2)$

11

The Big Valley Hypothesis

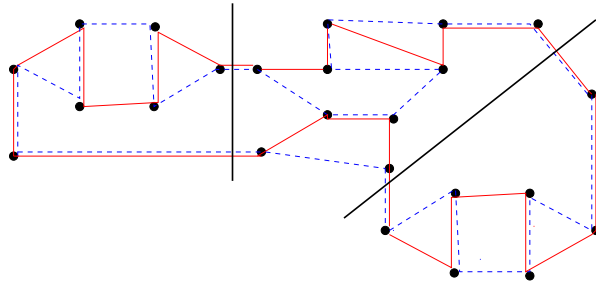
is sometimes used to explain metaheuristic search



Local Optima are "Linked" by Partition Crossover "Tunnels"

12

Generalized Partition Crossover in $O(n)$ time



Generalize Partition Crossover is always feasible if the partitions have 2 exits (same color in and out). If a partition has more than 2 exits, the “colors” must match.

13

How Many Partitions are Discovered?

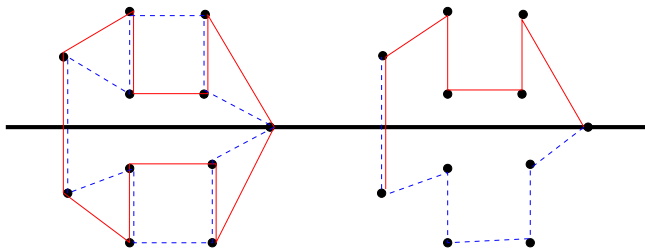
Instance	att532	nrw1379	rand1500	u1817
3-opt	10.5 ± 0.5	11.3 ± 0.5	24.9 ± 0.2	26.2 ± 0.7

Table: Average number of *partition components* used by GPX in 50 recombinations of random local optima found by 3-opt.

With 25 components, 2^{25} represents millions of local optima.

With 1000 components, returns the best of 2^{1000} local optima!!!

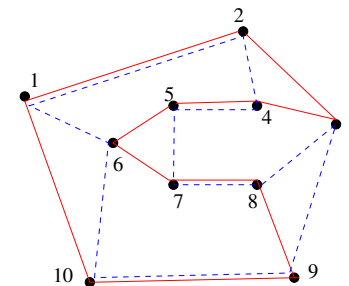
14



With Thanks to Gabriela Ochoa and Renato Tinós

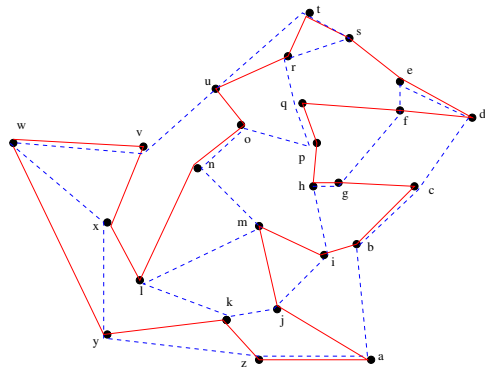
15

GPX, Cuts Crossing 4 Edges (IPT fails here)



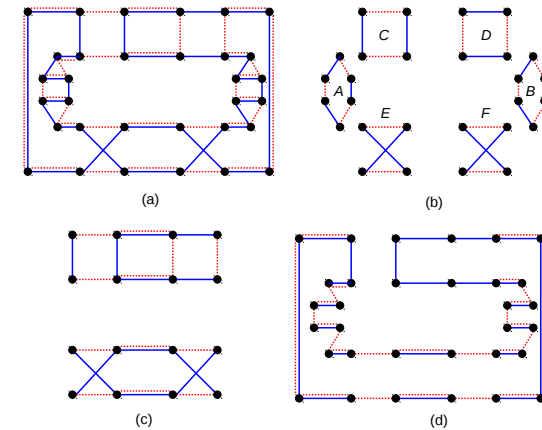
16

GPX, Complex Cuts ... Still $O(n)$ time



17

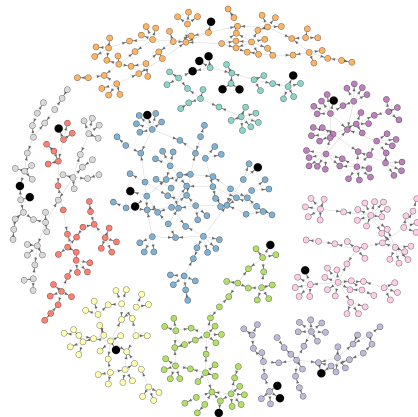
GPX, Complex Cuts ... Still $O(n)$ time



18

Tunneling Between Local Optima

Local Optima are "Linked" by Partition Crossover



Thanks to G. Ochoa and N. Veerapen.

19

The Two Best TSP (solo) Heuristics

Lin Kernighan Helsgaun (LKH 2 with Multi-Starts, and IPT Crossover)
Iterated Local Search

EAX: Edge Assembly Crossover (Nagata et al.)
Genetic Algorithm

Combinations of LKH and EAX
using Automated Algorithm Selection Methods (Hoos et al.)

THE BEST INEXACT "TSP" SOLVERS USE CROSSOVER!

20

Lin-Kernighan-Helsgaun-LKH

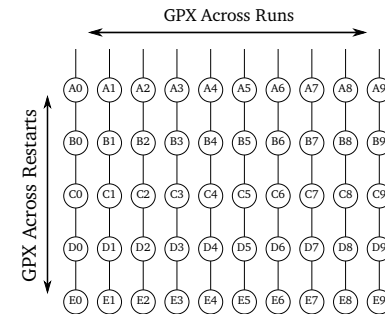
LKH is widely considered the best Local Search algorithm for TSP.

LKH uses deep k-opt moves, clever data structures and a fast implementation.

LKH-2 has found the majority of best known solutions on the TSP benchmarks at the Georgia Tech TSP repository that were not solved by complete solvers: <http://www.tsp.gatech.edu/data/index.html>.

21

GPX Across Runs and Restarts

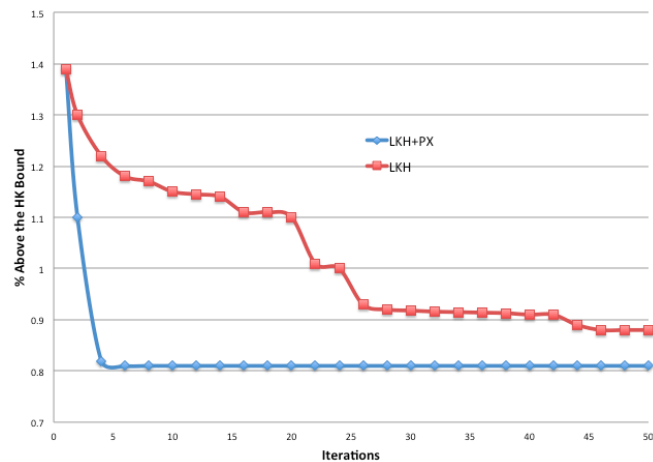


A diagram depicting 10 runs of multi-trial LKH-2 run for 5 iterations per run. The circles represent local optima produced by LKH-2. GPX across runs crosses over solutions with the same letters. GPX across restarts crosses over solutions with the same numbers.

22

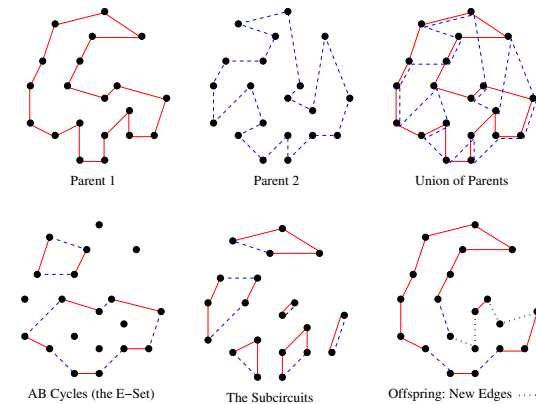
LKH with Partition Crossover

Mult-Start LKH compared to LKH+PX on 31K City Dimacs Cluster Instance



23

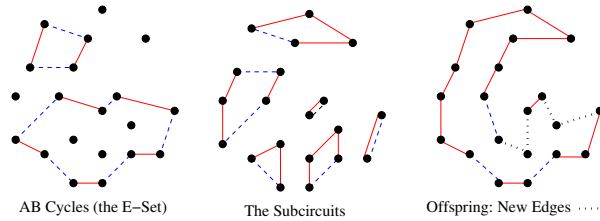
Edge Assembly Crossover



AB-Cycles are extracted from the graph which is the Union of the Parents. The AB-Cycles are used to cut Parent 1 into subcircuits.

24

Edge Assembly Crossover



The AB-Cycles are used to cut Parent 1 into subcircuits. These subcircuits are reconnected in a greedy fashion to create an offspring. The offspring is composed of edges from Parent 1, edges from Parent 2, and completely new edges not found in either parent.

25

The EAX Genetic Algorithm Details

- ① EAX is used to generate many (e.g. 30) offspring during every recombination. Only the best offspring is retained (Brood Selection).
- ② There is no selection, just "Brood Selection."
- ③ Typical population size: 300.
- ④ The order of the population is randomized every generation. Parent i is recombined with Parent $i + 1$ and the offspring replaces Parent i . (The population is replaced every generation.)

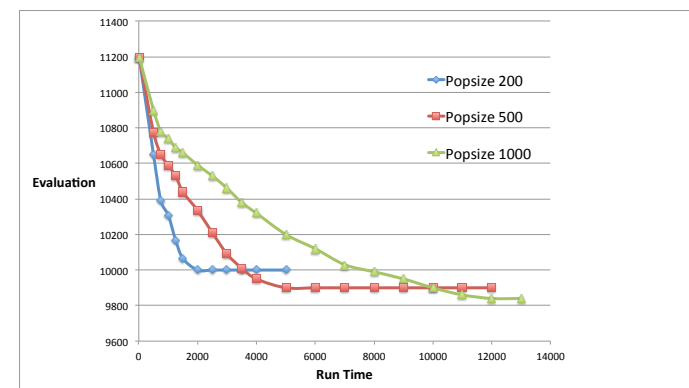
26

The EAX Strategy

- ① EAX can inherit many edges from parents, but also introduces new high quality edges.
- ② EAX disassembles and reassembles, and focuses on finding improvements.
- ③ This gives EAX a "thoroughness" of exploration.
- ④ EAX illustrates the classic trade-off between exploration and exploitation

27

Edge Assembly Crossover: Typical Behavior



28

Combining EAX and Partition Crossover

- ① Partition Crossover can dramatically speed-up exploitation, but it also impact long term search potential.
- ② A Strategy: When EAX generates 30 offspring, recombine all of the offspring using Partition Crossover.
- ③ This can help when EAX gets stuck and cannot find an improvement.

29

EAX and EAX with Partition Crossover

Standard EAX with restarts

Dataset	Pop Size	Evaluation Mean	S. D.	Running Time Mean	S. D.	Number Opt. Sol.
rl5934	200	556090.8	50	1433	34	12/30
rl5915	200	565537.57	29	1221	30	23/30
rl11849	200	923297.7	8	8400	130	1/10
ja9847	800	491930.1	2	37906	618	0/10
pla7397	800	23261065.6	552	12627	344	2/10
usa13509	800	19983194.5	411	81689	1355	0/10

EAX with Partition Crossover

Dataset	Pop Size	Evaluation Mean	S. D.	Running Time Mean	S. D.	Number Opt. Sol.
rl5934	200	556058.63	33	1562	248	21/30
rl5915	200	565537.77	21	1022	73	19/30
rl11849	200	923294.8	8	7484	105	4/10
ja9847	800	491926.33	2	30881	263	4/10
pla7397	800	23260855	222	11647	1235	4/10
usa13509	800	19982987.6	173	66849	818	2/10

30

The Mixing Genetic Algorithm

What is there is no *hard selection*.

Under *HARD selection* poor individuals are removed from the population.

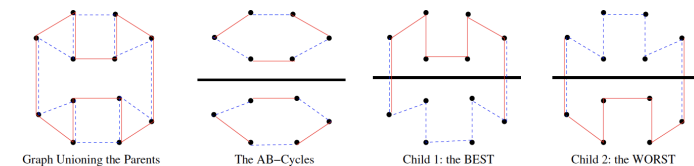
Under *SOFT selection* children replace parents, but individuals are not removed from the population.

Instead, the population becomes stratified, with the best individual at one end of the population, and the worst individuals at the other end of the population.

BEST ————— MEDIAN ————— WORST

31

The Mixing Genetic Algorithm

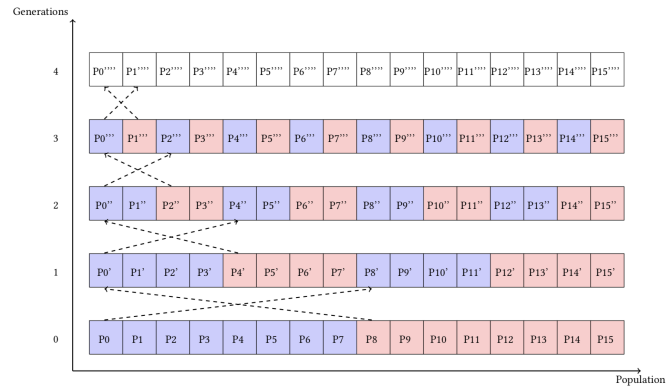


$$eval(P1) + eval(P2) = eval(C_{best}) + eval(C_{worst})$$

If you keep the best and worst child,
the population average never changes.

32

The Mixing Genetic Algorithm



33

The Hypercube Configuration

Assume the population size is 16.

After 4 ($\log_2(16)$) generations, every member of the population can contribute their best edges to location 0 in the population.

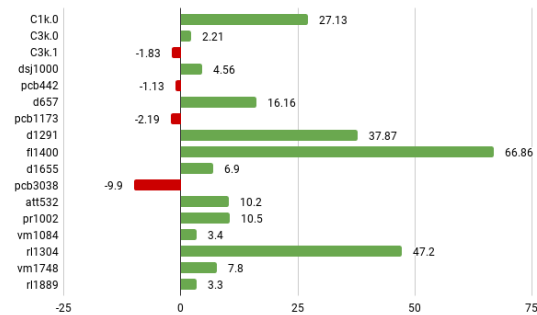
$(([(0 \times 8) \times (4 \times 12)] \times [(2 \times 10) \times (6 \times 14)]) \times ((1 \times 9) \times (5 \times 13)) \times [(3 \times 11) \times (7 \times 15)])$
and in general:

Theorem

(The Mixing Theorem): After each epoch ($\log(\text{popsize})$ generations) of the Mixing GA, all individuals in the population have the potential to contribute their best components to the individual at location 0.

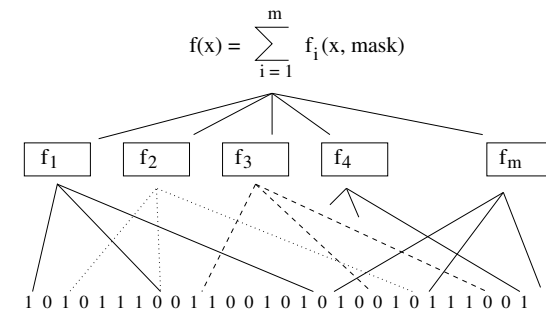
34

The Mixing GA Compared with EAX



35

k-bounded Pseudo-Boolean Functions



36

A General Result over Bit Representations

By Constructive Proof: Every problem with a bit representation and a closed form evaluation function can be expressed as a quadratic (k=2) pseudo-Boolean Optimization problem. (See Boros and Hammer)

$$xy = z \text{ iff } xy - 2xz - 2yz + 3z = 0$$

$$xy \neq z \text{ iff } xy - 2xz - 2yz + 3z > 0$$

Or we can reduce to k=3 instead:

$$f(x_1, x_2, x_3, x_4, x_5, x_6)$$

becomes (depending on the nonlinearity):

$$f1(z_1, z_2, z_3) + f2(z_1, x_1, x_2) + f3(z_2, x_3, x_4) + f4(z_3, x_5, x_6)$$

37

k-bounded Pseudo-Boolean functions

For example: A Random NK Landscape: $n = 10$ and $k = 3$.

The subfunctions:

$$\begin{array}{cccc} f_0(x_0, x_1, x_6) & f_1(x_1, x_4, x_8) & f_2(x_2, x_3, x_5) & f_3(x_3, x_2, x_6) \\ f_4(x_4, x_2, x_1) & f_5(x_5, x_7, x_4) & f_6(x_6, x_8, x_1) & f_7(x_7, x_3, x_5) \\ & f_8(x_8, x_7, x_3) & f_9(x_9, x_7, x_8) & \end{array}$$

But this could also be a MAXSAT Function,
or an arbitrary Spin Glass problem.

38

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

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

39

BLACK BOX OPTIMIZATION

Don't wear a blind fold during search if you can help it!



40

GRAY BOX OPTIMIZATION

THEOREM: All of the following functions are solved in 1 evaluation in $O(n)$ time.

ONEMAX

LEADING-ONES (TRAILING ZEROS)

TRAP functions

Multi-Modal UGLY Deceptive Problems

JUMP functions, ($m \ll n$)

UNITATION functions

All non-deceptive functions

Do we want to solve real problem?

Or just pretend to solve toy problems?

41

GRAY BOX OPTIMIZATION

We can construct “Gray Box” optimization for pseudo-Boolean optimization problems (M subfunctions, k variables per subfunction).

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

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

42

Walsh Example: MAX-3SAT

Walsh Analysis of a Single Clause

Consider the example function consisting of a single clause

$$f(x) = \neg x_2 \vee x_1 \vee x_0$$

$$\begin{aligned} f(000) &= 1 & (\neg x_2 T) \\ f(001) &= 1 & (\neg x_2 T) \\ f(010) &= 1 & (\neg x_2 T) \\ f(011) &= 1 & (\neg x_2 T) \\ f(100) &= 0 & (\neg x_2 F \wedge x_1 F \wedge x_0 F) \\ f(101) &= 1 & (x_0 T) \\ f(110) &= 1 & (x_1 T) \\ f(111) &= 1 & (x_1 T) \end{aligned}$$

43

Walsh Example: MAX-3SAT

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

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

44

Walsh Example: MAX-3SAT

Let $neg(f)$ return a K -bit string with 1 bits indicating which variables in the clause are negated.

$$f(100) = 0 \quad (\neg x_2 F \wedge x_1 F \wedge x_0 F)$$

$$neg(f) = 100$$

Then the Walsh coefficients for f are:

$$w_j = \begin{cases} \frac{2^K - 1}{2^K} & \text{if } j = 0 \\ -\frac{1}{2^K} \psi_j(neg(f)) & \text{if } j \neq 0 \end{cases}$$

45

Walsh Example

$$\begin{aligned} f_1 &= (\neg x_2 \vee x_1 \vee x_0) \\ f_2 &= (x_3 \vee \neg x_2 \vee x_1) \\ f_3 &= (x_3 \vee \neg x_1 \vee \neg x_0) \end{aligned}$$

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
1111	w_{15}	0	0	0	0

46

GRAY BOX OPTIMIZATION

We can construct "Gray Box" optimization for pseudo-Boolean optimization problems (M subfunctions, k variables per subfunction).

Exploit the *general properties of every* Mk Landscape:

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

Which can be expressed as a sum of k Eigenvectors:

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

47

The Eigenvectors of MAX-3SAT

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

$$f1(x) = f1_a(x) + f1_b(x) + f1_c(x)$$

$$f2(x) = f2_a(x) + f2_b(x) + f2_c(x)$$

$$f3(x) = f3_a(x) + f3_b(x) + f3_c(x)$$

$$f4(x) = f4_a(x) + f4_b(x) + f4_c(x)$$

$$\varphi^{(1)}(x) = f1_a(x) + f2_a(x) + f3_a(x) + f4_a(x)$$

$$\varphi^{(2)}(x) = f1_b(x) + f2_b(x) + f3_b(x) + f4_b(x)$$

$$\varphi^{(3)}(x) = f1_c(x) + f2_c(x) + f3_c(x) + f4_c(x)$$

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

48

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

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

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

In almost all cases, Score does not change after a bit flip. Only some Walsh coefficient are affected.

49

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

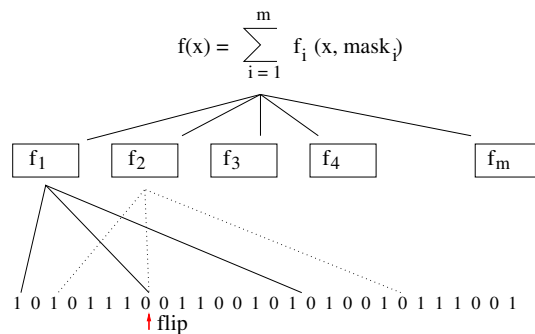
$$Score(x, y_p) = f(y_p) - f(x)$$

Thus, are the scores reflect the increase or decrease relative to $f(x)$ associated with flipping bit p .

In almost all cases, Score does not change after a bit flip. Only some subfunctions are affected.

50

When 1 bit flips what happens?



The improving moves can be identified in $O(1)$ time!
Mutation is not needed, except to diversify the search.

51

The locations of the updates are obvious

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

52

Some Theoretical Results: k-bounded Boolean

- 1) PROOF: Same runtime for BEST First and NEXT First search.
- 2) Constant time improving move selection under all conditions.
- 3) Constant time improving moves in space of statistical moments.
- 4) Auto-correlation computed in closed form.
- 5) Tunneling between local optima.

53

Best Improving and Next Improving moves

“Best Improving” and “Next Improving” moves cost the same.

GSAT uses a Buffer of best improving moves

$$Buffer(best.improvement) = \langle M_{10}, M_{1919}, M_{9999} \rangle$$

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

Instead uses multiple Buckets to hold improving moves

$$Bucket(best.improvement) = \langle M_{10}, M_{1919}, M_{9999} \rangle$$

$$Bucket(best.improvement - 1) = \langle M_{8371}, M_{4321}, M_{847} \rangle$$

$$Bucket(all.other.improving.moves) = \langle M_{40}, M_{519}, M_{6799} \rangle$$

This improves the runtime of GSAT by a factor of 20X to 30X.

54

Steepest Descent on Moments

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

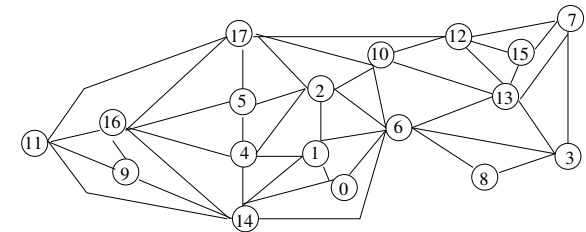
$$f(x) = \sum_{z=0}^3 \varphi^{(z)}(x)$$

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

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

55

The Variable Interaction Graph

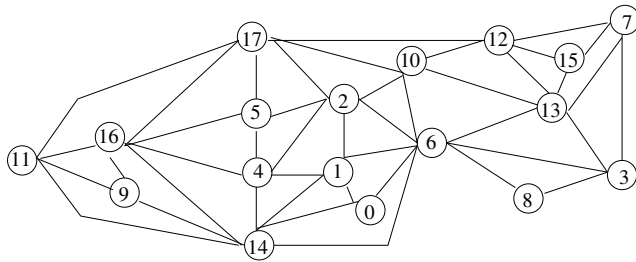


There is a vertex for each variable in the Variable Interaction Graph (VIG). There must be fewer than 2^k $M = O(n)$ Walsh coefficients. There is a connection in the VIG between vertex v_i and v_j if there is a

non-zero Walsh coefficient indexed by i and j , e.g., $w_{i,j}$.

56

What if you want to flip 2 or 3 bits at a time?



Assume all distance 1 moves are taken.

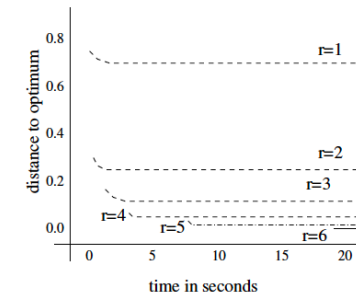
There can never be an improving move flipping bits 2 and 7.

There can never be an improving move flipping bits 4, 6 and 9.

There can never be an improving move over combinations of bits where there are no (non-zero) Walsh coefficients.

57

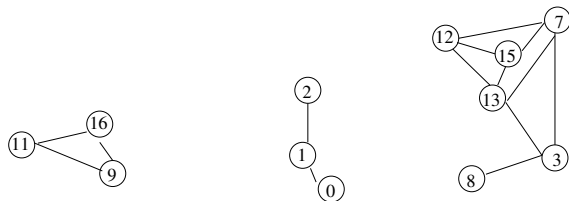
What if you want to flip 2 or 3 bits at a time?



12,000 bit k-bounded functions

58

The Recombination Graph: a reduced VIG



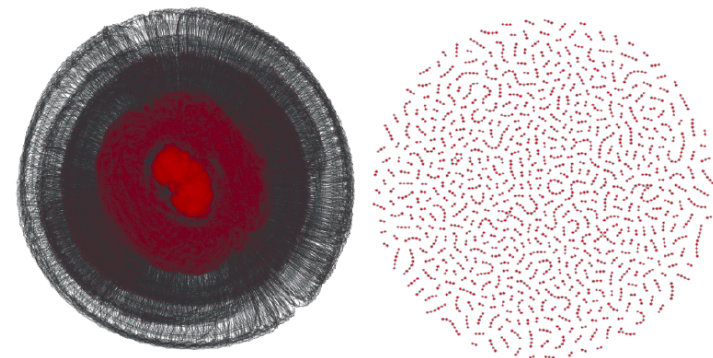
When recombining the solutions $S_{P1} = 0000000000000000$ and $S_{P2} = 111100011101110110$, the vertices and edges associated with shared variables 4, 5, 6, 10, 14 are deleted to yield the **recombination graph**.

Tunneling Crossover Theorem:

If the recombination graph of f contains q connected components, then Partition Crossover returns the best of 2^q solutions.

59

Decomposed Evaluation for MAXSAT



60

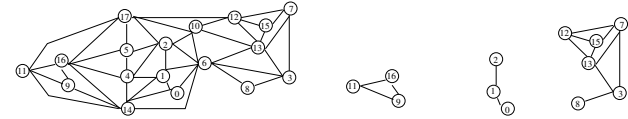
MAXSAT Number of recombining components

Instance	N	Min	Median	Max
aaai10ipc5	308,480	7	20	38
AProVE0906	37,726	11	1373	1620
atcoenc3opt19353	991,419	937	1020	1090
LABSno88goal008	182,015	231	371	2084
SATinstanceN111	72,001	34	55	1218

Tunneling “scans” 2^{1000} local optima and returns the best in $O(n)$ time.

61

Decomposed Evaluation



A new evaluation function can be constructed:

$$g(x) = c + g_1(x_0, x_1, x_2) + g_2(x_9, x_{11}, x_{16}) + g_2(x_3, x_7, x_8, x_{12}, x_{13}, x_{15})$$

where $g(x)$ evaluates any solution (parents or offspring) that resides in the subspace ****000***0***0**.

In general:

$$g(x) = c + \sum_{i=1}^q g_i(x, \text{mask}_i)$$

62

Partition Crossover and Local Optima

The Subspace Optimality Theorem: For any k -bounded pseudo-Boolean function f , if Partition Crossover is used to recombine two parent solutions that are locally optimal, then the offspring must be a local optima in the hyperplane subspace defined by the bits shared in common by the two parents.

Example: if the parents 0000000000 and 1100011101 are locally optimal, then the best offspring is locally optimal in the hyperplane subspace **000***0*.

63

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 \pm 3.9	0.3 \pm 0.3	100.0 \pm 0.0
300	4	Adj	30.7 \pm 2.8	0.0 \pm 0.0	94.4 \pm 4.3
500	2	Adj	78.0 \pm 2.3	0.0 \pm 0.0	97.9 \pm 5.0
500	4	Adj	31.0 \pm 2.5	0.0 \pm 0.0	93.8 \pm 4.0
100	2	Rand	0.8 \pm 0.9	0.5 \pm 0.5	100.0 \pm 0.0
300	4	Rand	0.0 \pm 0.0	0.0 \pm 0.0	86.4 \pm 17.1
500	2	Rand	0.0 \pm 0.0	0.0 \pm 0.0	98.3 \pm 4.9
500	4	Rand	0.0 \pm 0.0	0.0 \pm 0.0	83.6 \pm 16.8

64

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.

65

Optimal Solutions for Adjacent NK

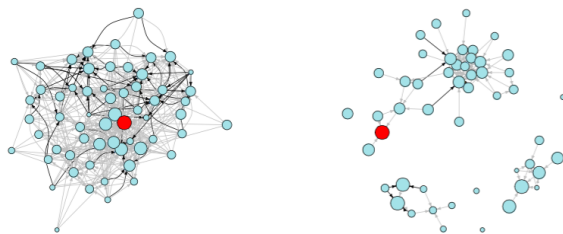
N	k	2-point	Uniform	Paired PX
		Found	Found	Found
300	2	18	0	100
300	3	0	0	100
300	4	0	0	98
500	2	0	0	100
500	3	0	0	98
500	4	0	0	70

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

66

Tunneling Local Optima Networks

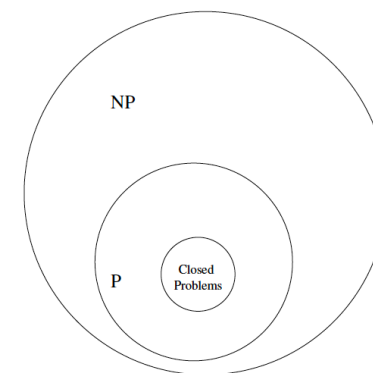
Multimodal problems are not always difficult
NK Landscapes: Ochoa et al. GECCO 2015



Adjacent (easy) NK Landscapes have more optima.
But Random (hard) NK Landscapes have disjunct "funnels."

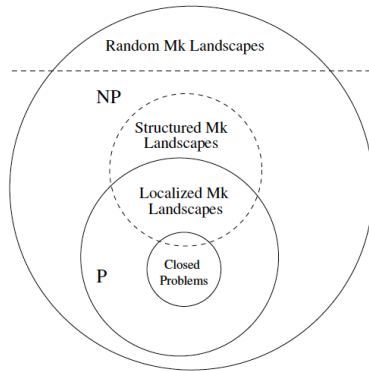
67

NK and Mk Landscapes, P and NP



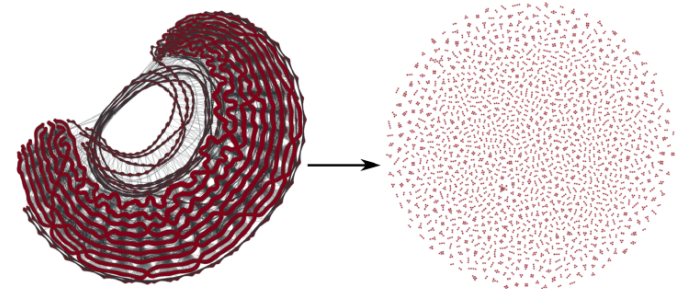
68

NK and Mk Landscapes, P and NP



69

Decomposed Evaluation for MAXSAT



atco_enc3_opt1_13_48

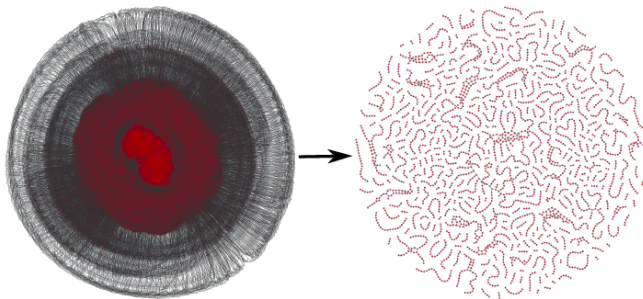
Air traffic controller shift scheduling problem: 1087 components.

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

N= 1,067,657

70

Decomposed Evaluation for MAXSAT



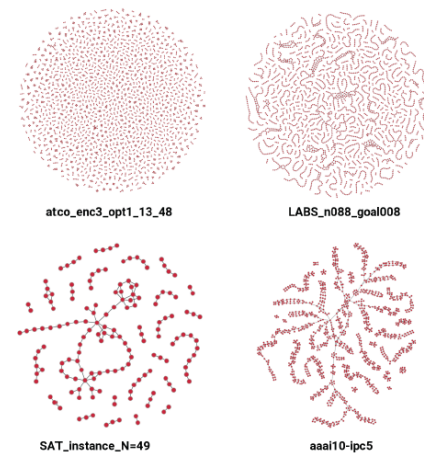
LABS_n088_goal008

Finding low autocorrelation binary sequence: 371 components

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

N= 182,015

71



72

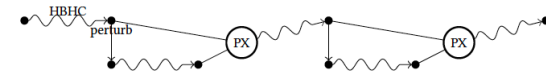
MAXSAT Number of recombining components

Instance	N	Min	Median	Max
aaai10ipc5	308,480	7	20	38
AProVE0906	37,726	11	1373	1620
atcoenc3opt19353	991,419	937	1020	1090
LABSno88goal008	182,015	231	371	2084
SATinstanceN111	72,001	34	55	1218

Imagine:
crossover "scans" 2^{1000} local optima and returns the best in $O(n)$ time

73

What's (Obviously) Next?

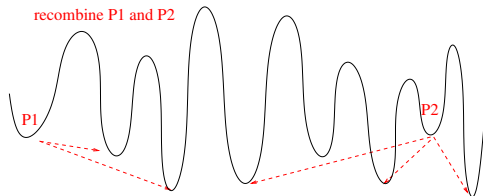


Deterministic Recombination Iterated Local Search (DRILS)

This exploits constant time deterministic improving moves selection and deterministic partition crossover.

74

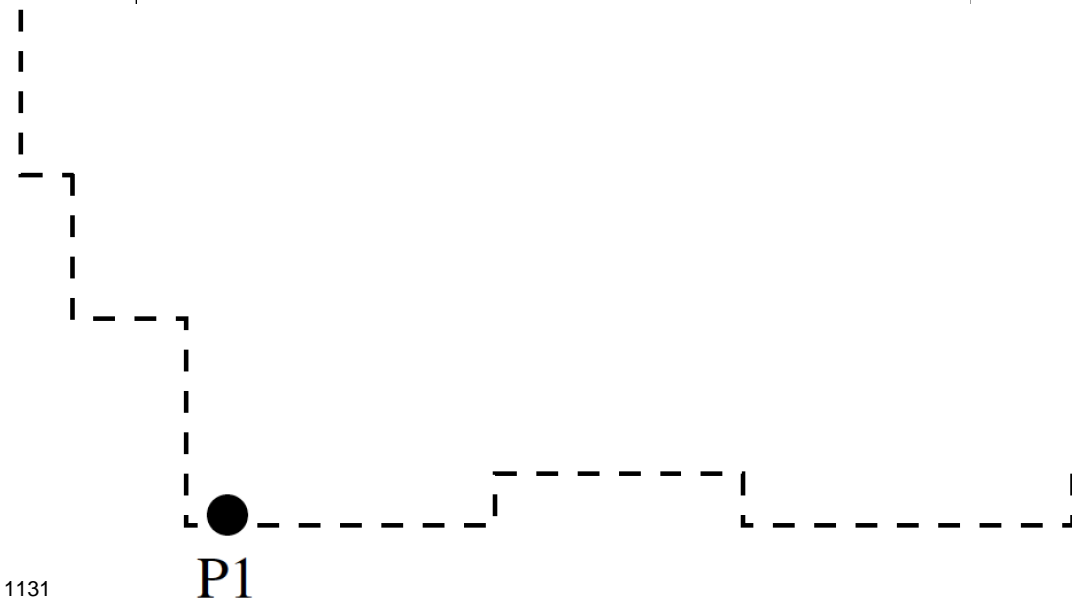
Tunneling Between Local Optima



For Weighted MAXSAT Local Optima are well defined.
Partition Crossover (PX) can "Tunnel" between local optima.

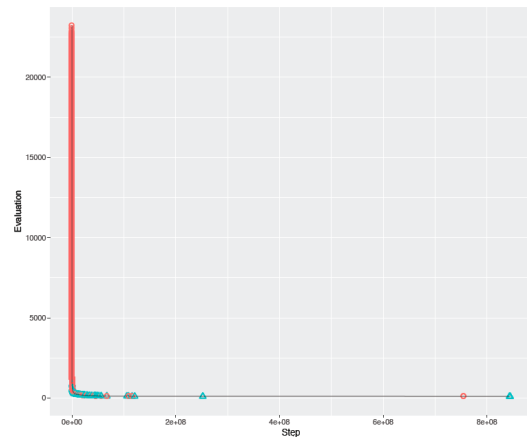
Tunneling = jump from local optimum to local optimum, while finding the best local optima.

Tunneling Between Plateaus for MAXSAT



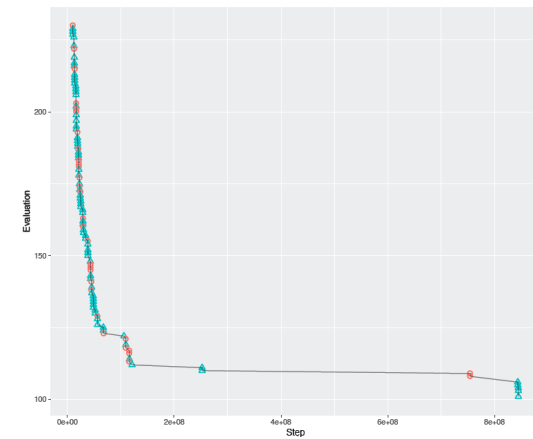
1131

Early MAXSAT Results



77

Early MAXSAT Results



78

New Theoretical 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

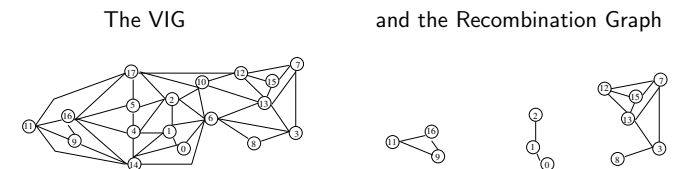
Assume that $f(P_1) = f(P_2)$.

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.

79

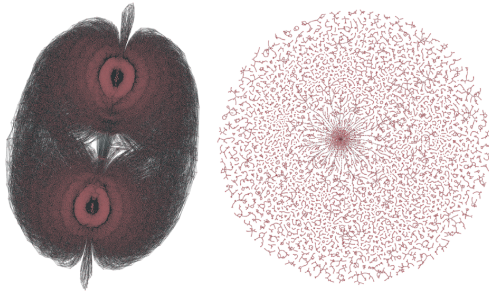
Decomposed Evaluation



80

Local Search plus Recombination

The VIG and the Recombination Graph



For this instance ($N=72,001$), Sparrow got stuck early. One Partition Crossover returned the best of 2^{842} solutions and satisfied 316 additional clauses.

81

Local Search Algorithms for MAXSAT

Adapt G^2 WSAT: Best in the 2007 SAT Competition

NEW: Adapt G^2 WSAT 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

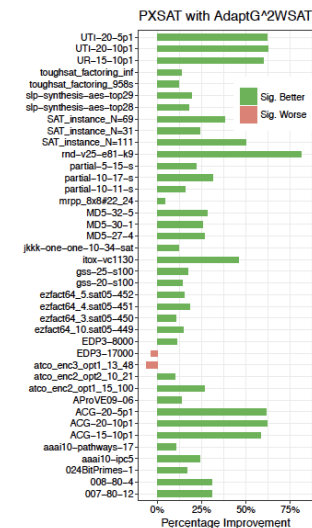
82

The MAXSAT instances

We used 102 instances drawn uniformly from each instance type of the *crafted* and *industrial* problems from the 2014 SAT Competition.

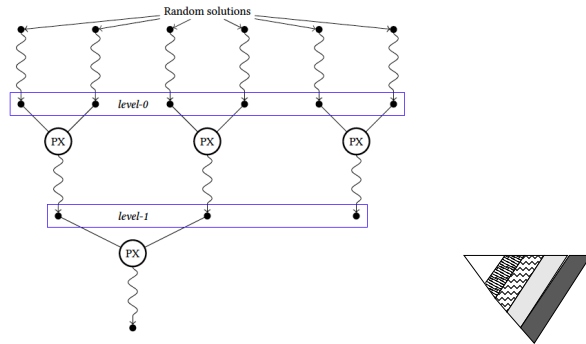
Most problems are $100,000 < n < 1,500,000$

83



84

One Million Variable NK Landscapes

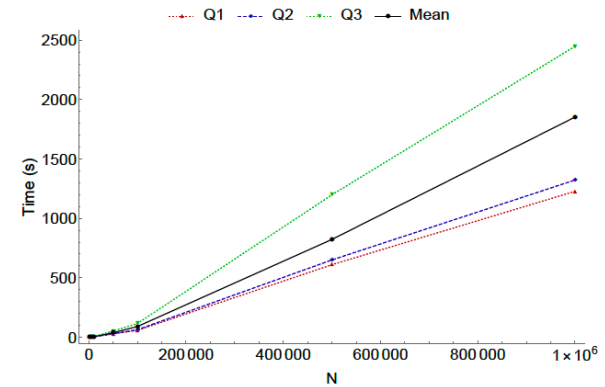


This configuration is best for Adjacent NK Landscapes with low K value.

We can now solve 1 million variable NK-Landscapes to optimality in approximately linear time. This exploits constant time deterministic improving moves selection and deterministic partition crossover.

85

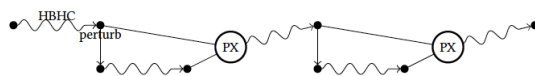
One Million Variable NK Landscapes



Scaling for runtime, Adjacent NK Landscapes with $K = 2$ ($k = 3$).

86

One Million Variable NK Landscapes



This DRILS configuration is best for Random NK Landscapes, and in general problems with higher values of K.

This exploits constant time deterministic improving moves selection and deterministic partition crossover.

87

NK Landscapes and MAXSAT

Black Box Optimization is HOPELESSLY inefficient.



In expectation, for $N = 1,000,000$, with 1 improving move available:

In the worst case,
for the 1 improving move made by a Black Box Optimizer
a Gray Box Optimizer can make 1,000,000 improving moves.

88

Cast Scheduling: K. Deb and C. Myburgh.

A foundry casts objects of various sizes and numbers by melting metal on a crucible of capacity W . Each melt is called a *heat*.

Assume there N total objects to be cast, with r_j copies of the j^{th} object. Each object has a fixed weight w_i , thereby requiring $M = \sum_{j=1}^N r_j w_j$ units of metal.

DEMAND: Number of copies of the j^{th} object.

CAPACITY of the crucible, W .

89

Casts: Multiple Objects, Multiple Copies



90

Cast Scheduling: Deterministic Recombination

Parent 1:										Metal	
Weight	154	136	57	55	67	83	187	20	123	Used	
	0	1	0	0	0	0	0	1	1	0	343
	2	0	0	0	0	1	0	1	0	2	808
	1	1	0	1	1	3	1	0	0	0	629
	0	0	2	1	0	2	0	2	1	4	698
Demand	3	2	2	2	4	3	2	3	4	0.953 Infeasible	
Fitness:										-63.6	

Parent 2:										Metal	
Weight	154	136	57	55	67	83	187	20	123	Used	
	1	0	1	2	2	0	0	1	0	3	625
	1	1	0	0	0	1	1	1	2	0	667
	1	0	0	0	0	1	0	0	3	0	606
	0	1	1	0	1	1	1	0	0	1	580
Demand	3	2	2	2	4	3	2	3	4	0.953 Infeasible	
Fitness:										0.269	

Offspring:										Metal	
Weight	154	136	57	55	67	83	187	20	123	Used	
	1	0	1	2	2	0	0	1	0	3	625
	1	1	0	0	1	1	1	2	0	0	667
	1	1	0	1	3	1	0	0	0	0	629
	0	1	1	0	1	1	1	0	0	1	580
Demand	3	2	2	2	4	3	2	3	4	0.962 Infeasible	

Recombination is illustrated for a small problem with $N = 10$, $H = 4$, with capacity $W = 650$. Demand (r_j) is shown in the final row.

91

Cast Scheduling: Deterministic Recombination

Parent 1 :										Metal
Weight	154	136	57	55	67	83	187	20	123	Used
	0	1	0	0	0	0	1	1	0	343
	2	0	0	0	1	0	1	0	2	808
	1	1	0	1	3	1	0	0	0	629
	0	0	2	1	0	2	0	2	1	698
Demand	3	2	2	2	4	3	2	3	4	0.953 Infeasible
Fitness:										-63.6

Parent 2 :										Metal
Weight	154	136	57	55	67	83	187	20	123	Used
	1	0	1	2	2	0	0	1	0	625
	1	1	0	0	1	1	1	2	0	667
	1	0	0	0	0	1	0	0	3	606
	0	1	1	0	1	1	1	0	0	580
Demand	3	2	2	2	4	3	2	3	4	0.953 Infeasible
Fitness:										0.269

Offspring:										Metal	
Weight	154	136	57	55	67	83	187	20	123	Used	
	1	0	1	2	2	0	0	1	0	3	625
	1	1	0	0	1	1	1	2	0	667	
	1	1	0	1	3	1	0	0	0	629	
	0	1	1	0	1	1	1	0	0	580	
Demand	3	2	2	2	4	3	2	3	4	0.962 Infeasible	
Fitness:										0.278	

Parent 2 has a better metal utilization for rows 1, 2 and 4. Row 3 is taken from Parent 1. Recombination is greedy.

92

Cast Scheduling: Deterministic Recombination

Repair 1:												Metal
Weight	154	136	57	55	67	83	187	20	123	50		Used
	1	0	1	2	1	0	0	1	0	3		558
	1	0	0	0	1	1	1	2	1	0		654
	1	1	0	0	2	1	0	0	1	0		630
	0	1	1	0	0	1	1	0	1	1		636
Demand	3	2	2	2	4	3	2	3	3	4		0.953
Fitness:												0.915

Repair operators are applied to offspring solution.

Repair 1: The respective variables are increased (green) or decreased (blue) to meet Demand.

93

Cast Scheduling: Deterministic Recombination

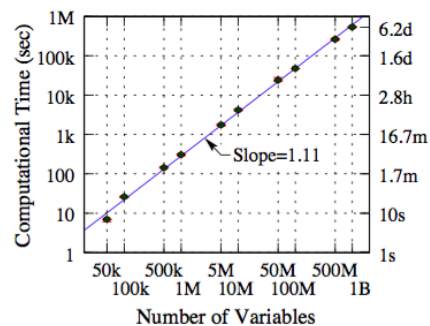
Repair 2:												Metal
Weight	154	136	57	55	67	83	187	20	123	50		Used
	1	0	1	2	1	0	0	2	0	3		578
	1	0	0	0	1	1	1	1	1	0		634
	1	1	0	0	2	1	0	0	1	0		630
	0	1	1	0	0	1	1	0	1	1		636
Demand	3	2	2	2	4	3	2	3	3	4		0.953
Fitness:												0.953

Repair operators are applied to offspring solution.

Repair 2: Objects are moved to different heats within the individual columns to reduce or minimize infeasibility.

94

One Billion Variables



Breaking the Billion-Variable Barrier in Real World Optimization Using a Customized Genetic Algorithm. K. Deb and C. Myburgh. GECCO 2016.

95

What's (Obviously) Next?



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

96