NEXT GENERATION GENETIC ALGORITHMS

JOINT LECTURES ON EVOLUTIONARY ALGORITHMS (JOLEA)

Darrell Whitley

Computer Science
Colorado State University

NOVEMBER 2021

THANKS TO:

CIAI

CASI

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

SPECIAL THANKS TO:

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






TUTORIAL

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

Handbook of Metaheuristics
Springer, 2019

Email me:

darrell.whitley@gmail.com

SUBJECT: Tutorial

INTELLIGENT LOCAL SEARCH:

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

Tabu Search

Variable Neighborhood Search

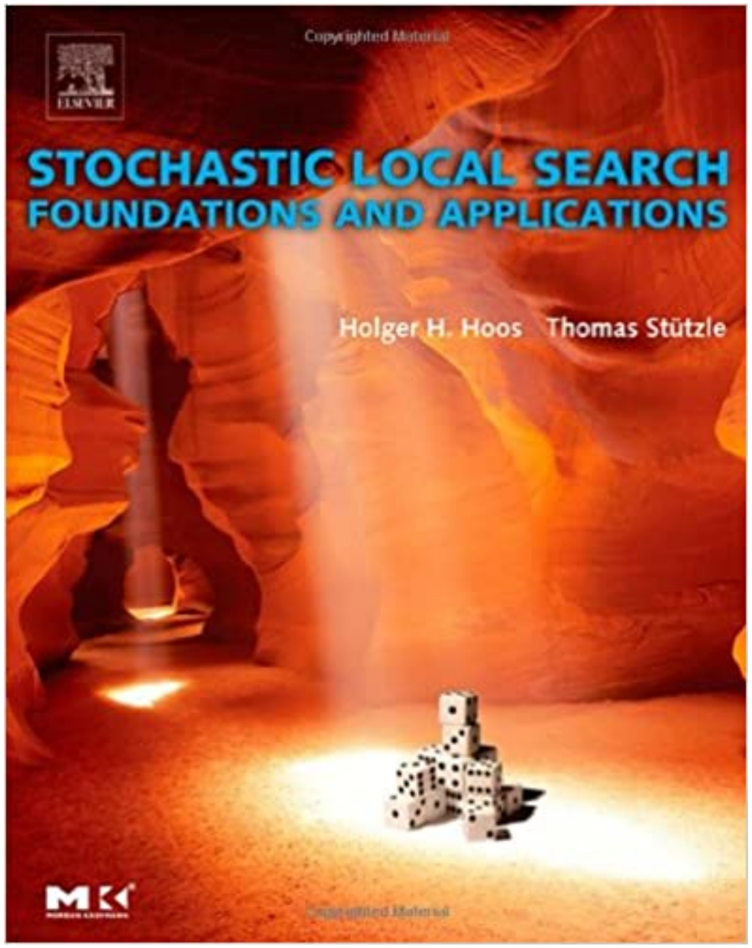
Efficient Annealing

Generalized Pattern Search

Nelder-Mead

.

INTELLIGENT LOCAL SEARCH:



Stochastic Local Search

Hoos and Stützle

Blind Local Search:

unintelligent Local Search:

Blind Random Local Search

is rarely a competitive Search Strategy.

RANDOM MUTATION IS *OBSOLETE AND USELESS* *FOR MANY PROBLEM CLASSES:*

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

What does Evolutionary Computation have to offer to the larger field of ***inexact methods for combinatorial optimization***?

WHAT DOES EVOLUTIONARY COMPUTATION HAVE TO OFFER TO THE LARGER FIELD OF *INEXACT METHODS FOR COMBINATORIAL OPTIMIZATION*?

Recombination

Parallelism

So what can Evolutionary Computation bring to the Table?

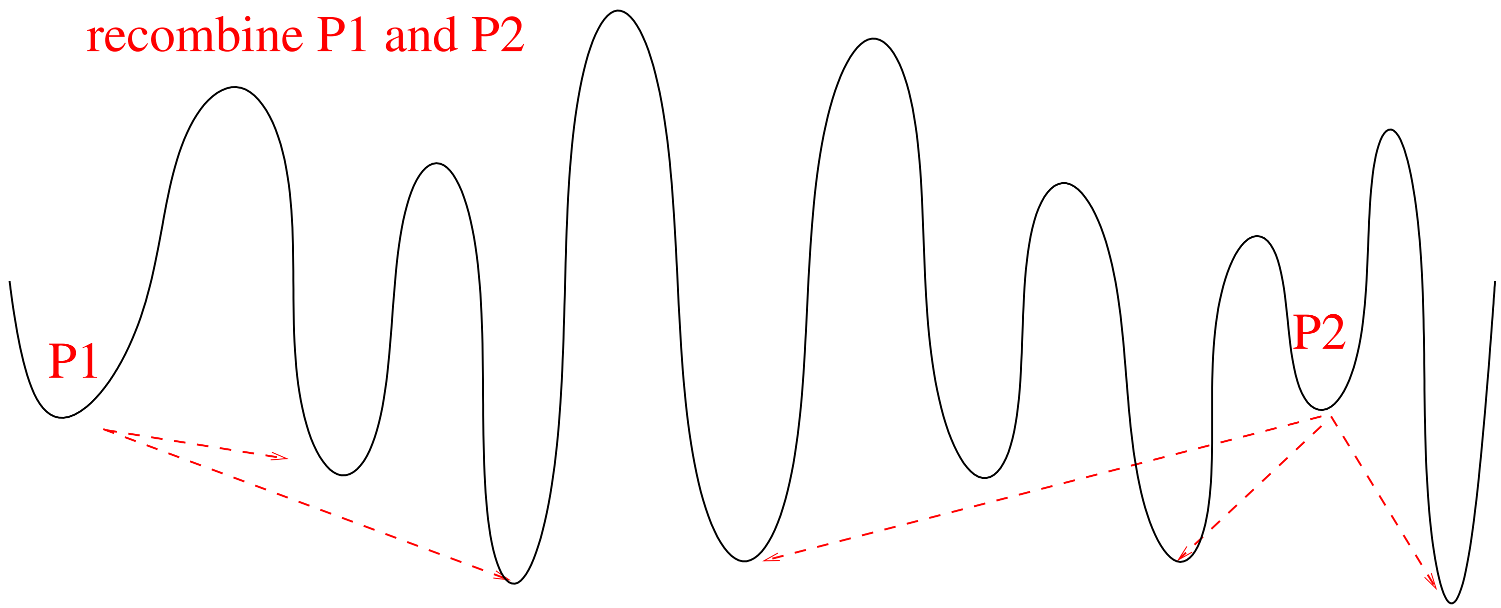
This isn't about

Recombination versus ***Mutation.***

It is really about

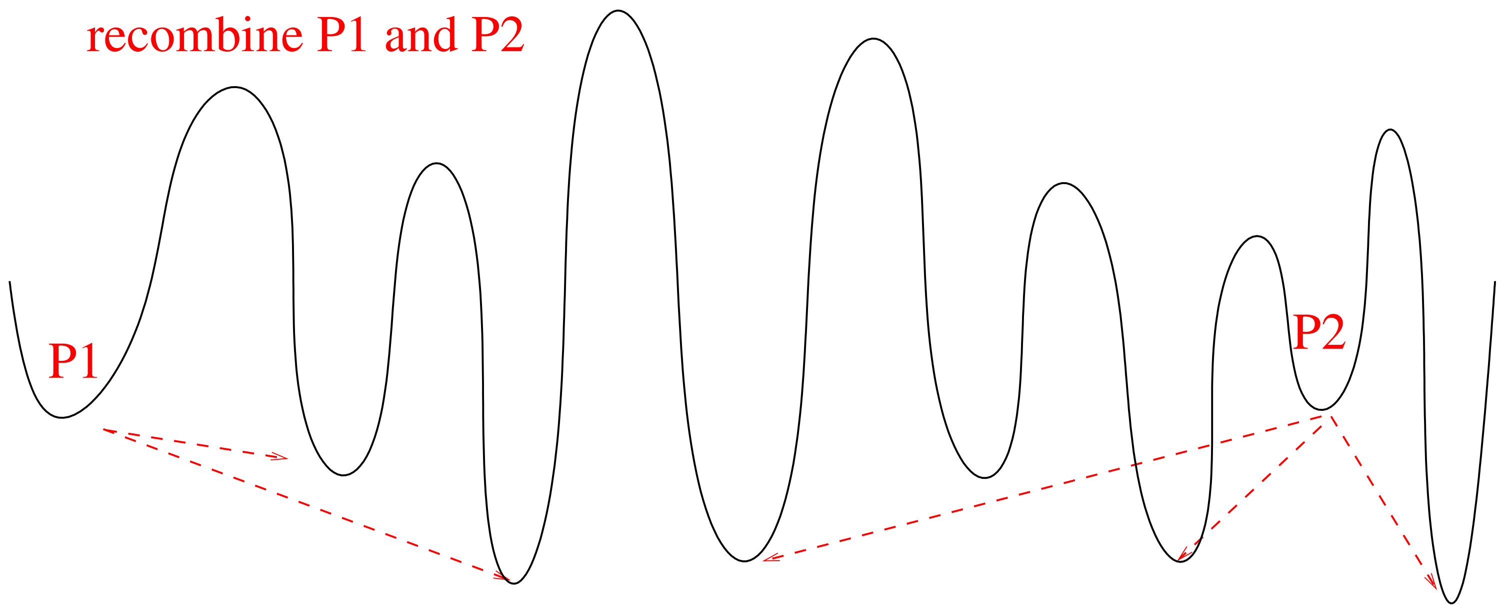
Intelligent Local Search versus
Unintelligent Local Search.

Crossover can *Deterministically* "Tunnel" Between Optima



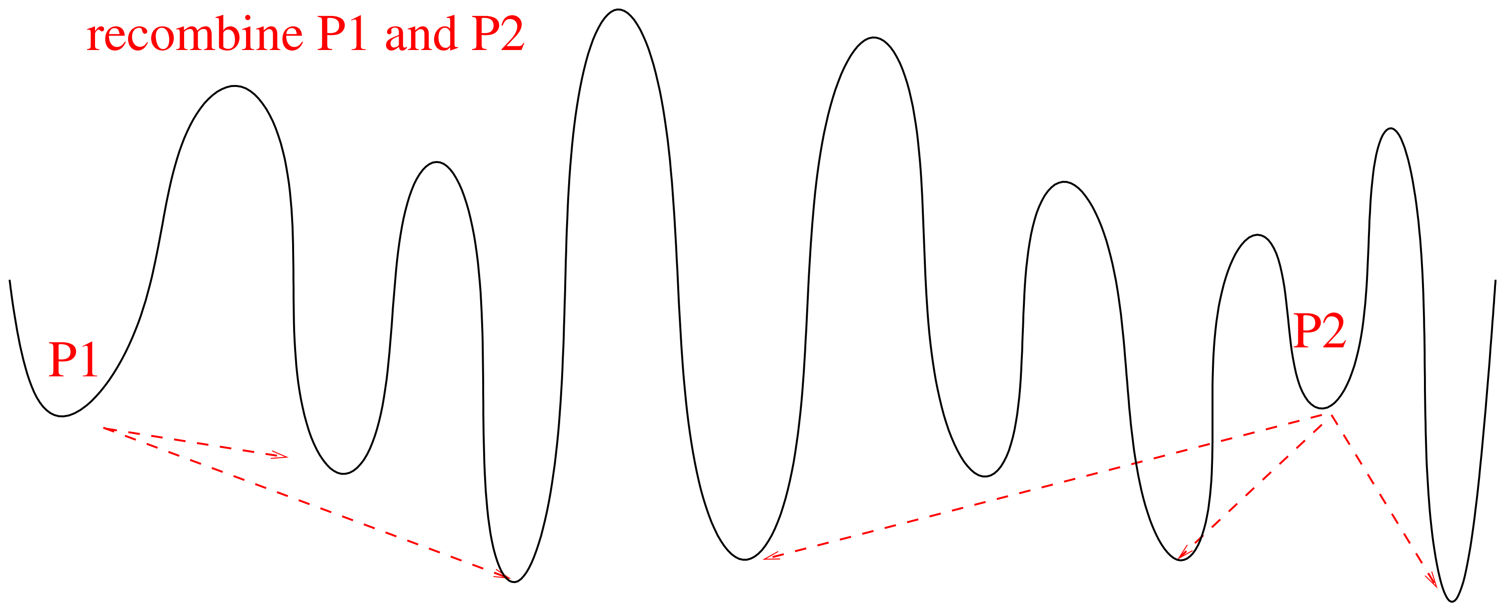
We can often remove randomness from Crossover

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



We can often remove randomness from Crossover

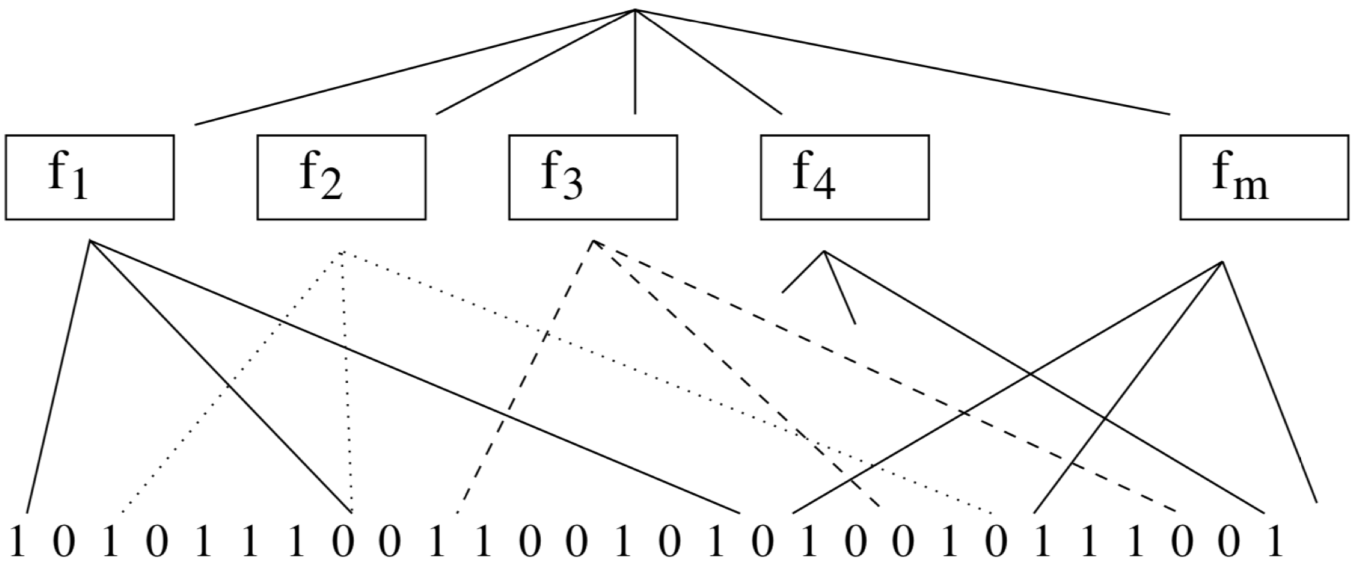
APPLY INTELLIGENT LOCAL SEARCH *BEFORE* CROSSOVER.



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

K-bounded Pseudo-Boolean Functions

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



K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS: MAXSAT

Literal: a variable or the negation of a variable

Clause: a disjunct of literals

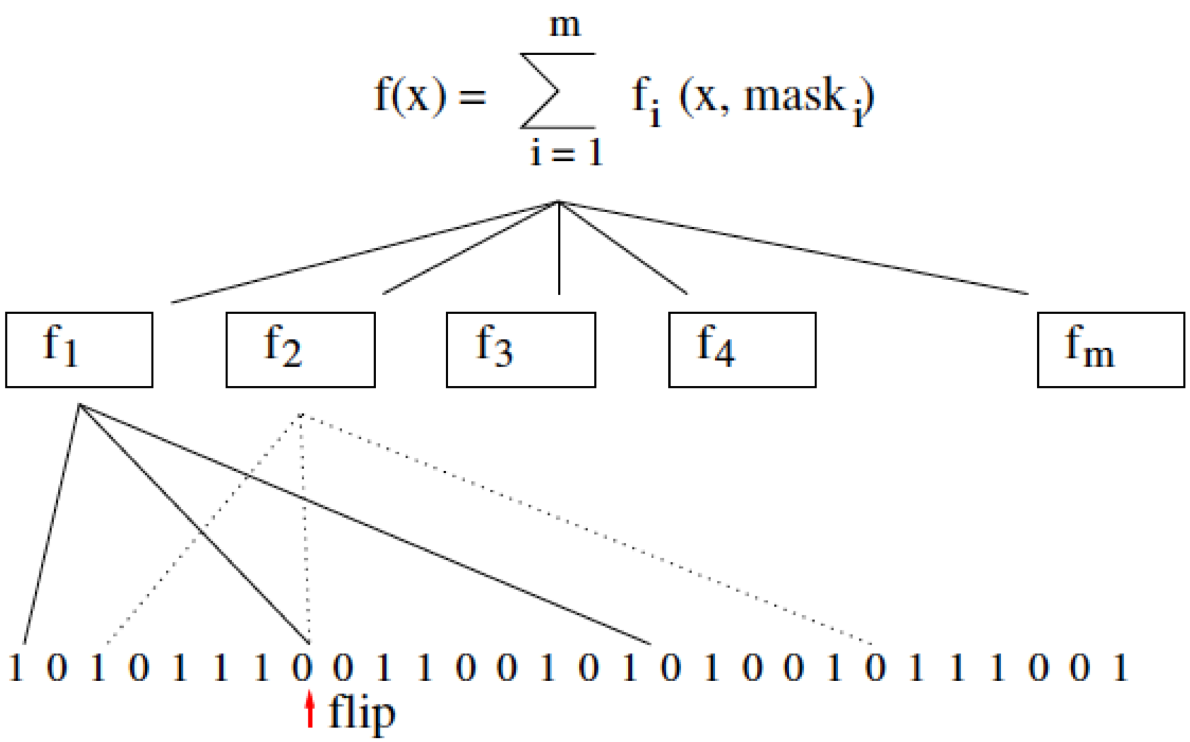
A 3SAT Example

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

recast as a MAX3SAT Example

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

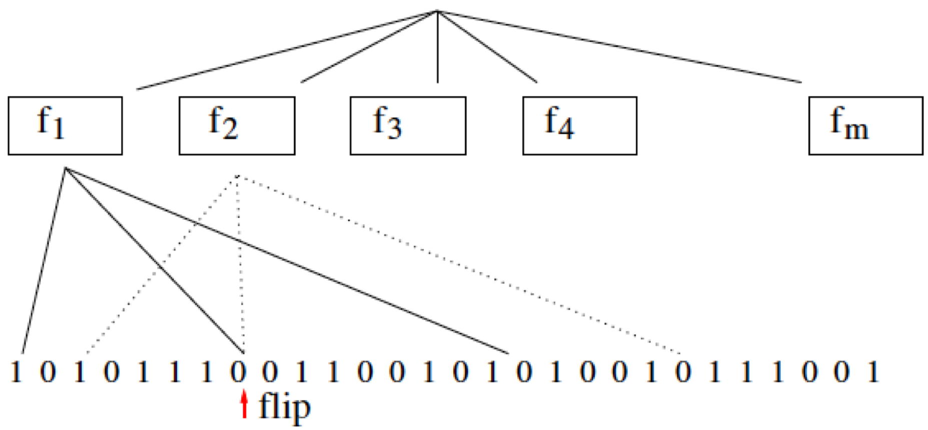
K-bounded Pseudo-Boolean Functions



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.

K-BOUNDED PSEUDO-BOOLEAN FUNCTIONS

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



The 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).

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 $O(1)$.

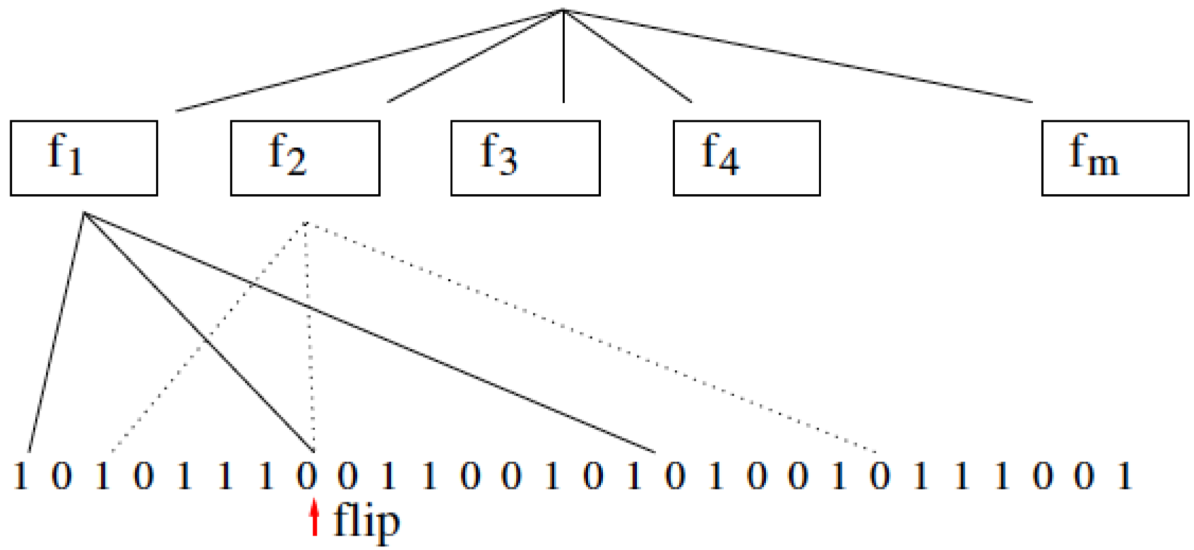
Pick a constant C.
If a variable appears in less than C subfunctions, no problem.
When that variable is flipped it has $O(1)$ interactions .

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.

K-bounded Pseudo-Boolean Functions

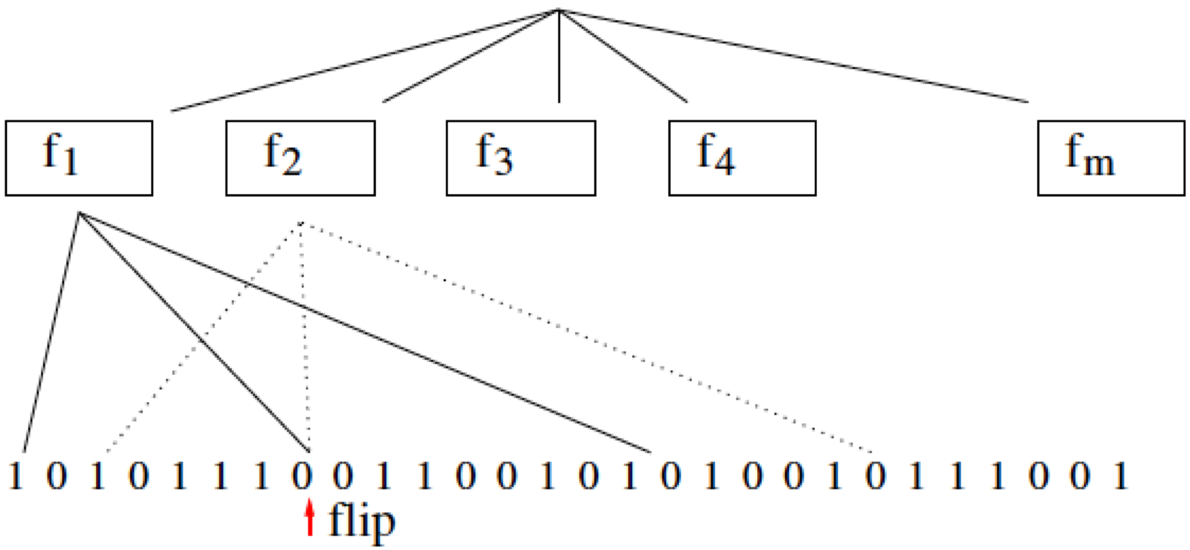
$$f(x) = \sum_{i=1}^{m} f_i\,(x, \text{mask}_i)$$



RANDOM MUTATION IS OBSOLETE.

K-bounded Pseudo-Boolean Functions

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

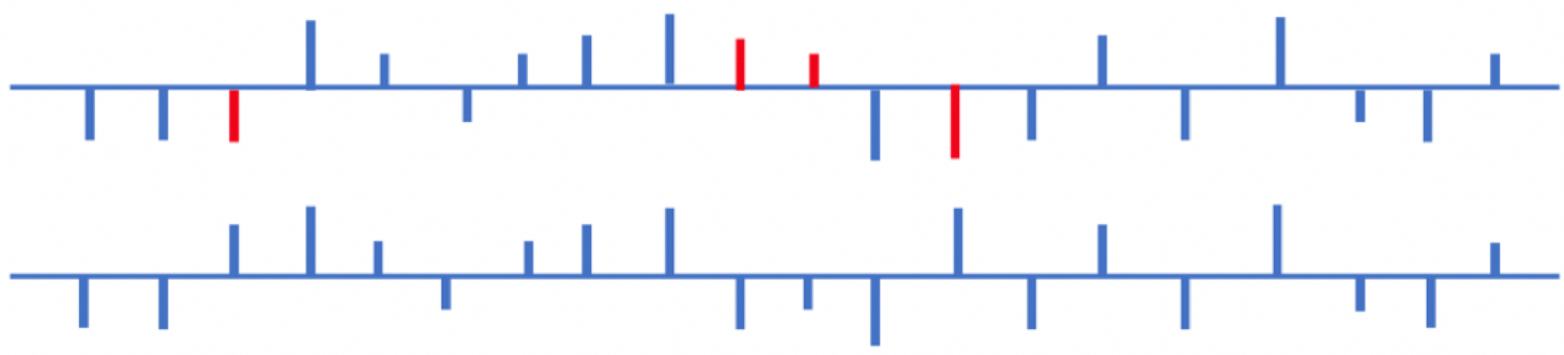


The MAXSAT community stopped using random blind local search 30 years ago (1992) but they still call it "Blackbox."

Improving Moves in Fourier/Walsh Space

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

(Warning, the notation is compressed.)



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

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

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

What about looking 2 or 3 or 10 moves ahead?

With Thanks to Francisco Chicano

WHAT ABOUT LOOKING 2 OR MORE MOVES AHEAD?

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

Assume you have taken all single bit flip improving moves.

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

What about looking 2 or more moves ahead?

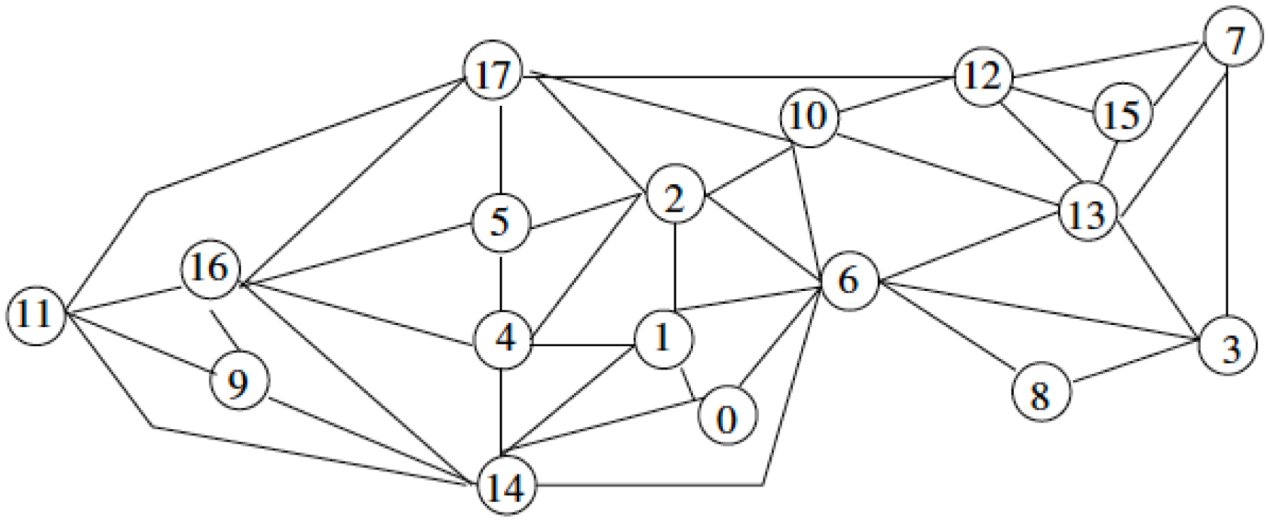
$$
\begin{aligned}
f(x) = {} & \psi_x w_1 + \psi_x w_2 + \psi_x w_3 + \psi_x w_4 + \psi_x w_5 + \psi_x w_6 + \psi_x w_7 + \psi_x w_8 \\
& + \psi_x w_{1,2} + \psi_x w_{2,3} + \psi_x w_{3,4} + \psi_x w_{1,4} + \psi_x w_{3,5} + \psi_x w_{5,6} \\
& + \psi_x w_{6,7} + \psi_x w_{5,7} + \psi_x w_{7,8} + \psi_x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

Assume you have taken all single bit flip improving moves.

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

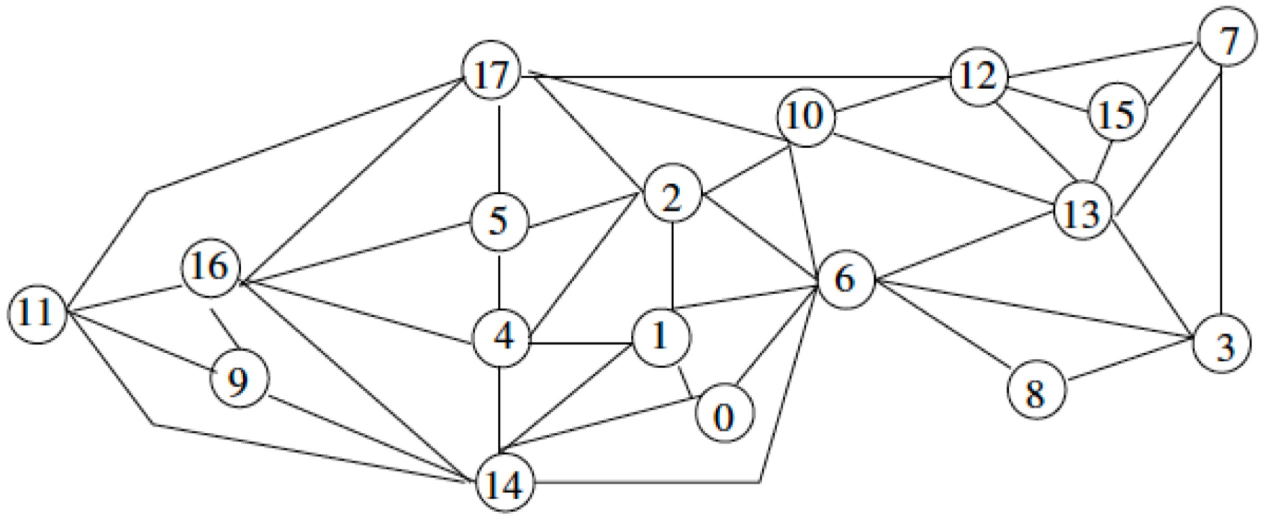
NOTHING. There are no nonlinear coefficients involving 5 and 8.

The Variable Interaction Graph



Can be constructed heuristically or exactly.

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.

WHAT ABOUT LOOKING 2 OR MORE MOVES AHEAD?

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

What about looking 2 or more moves ahead?

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

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					

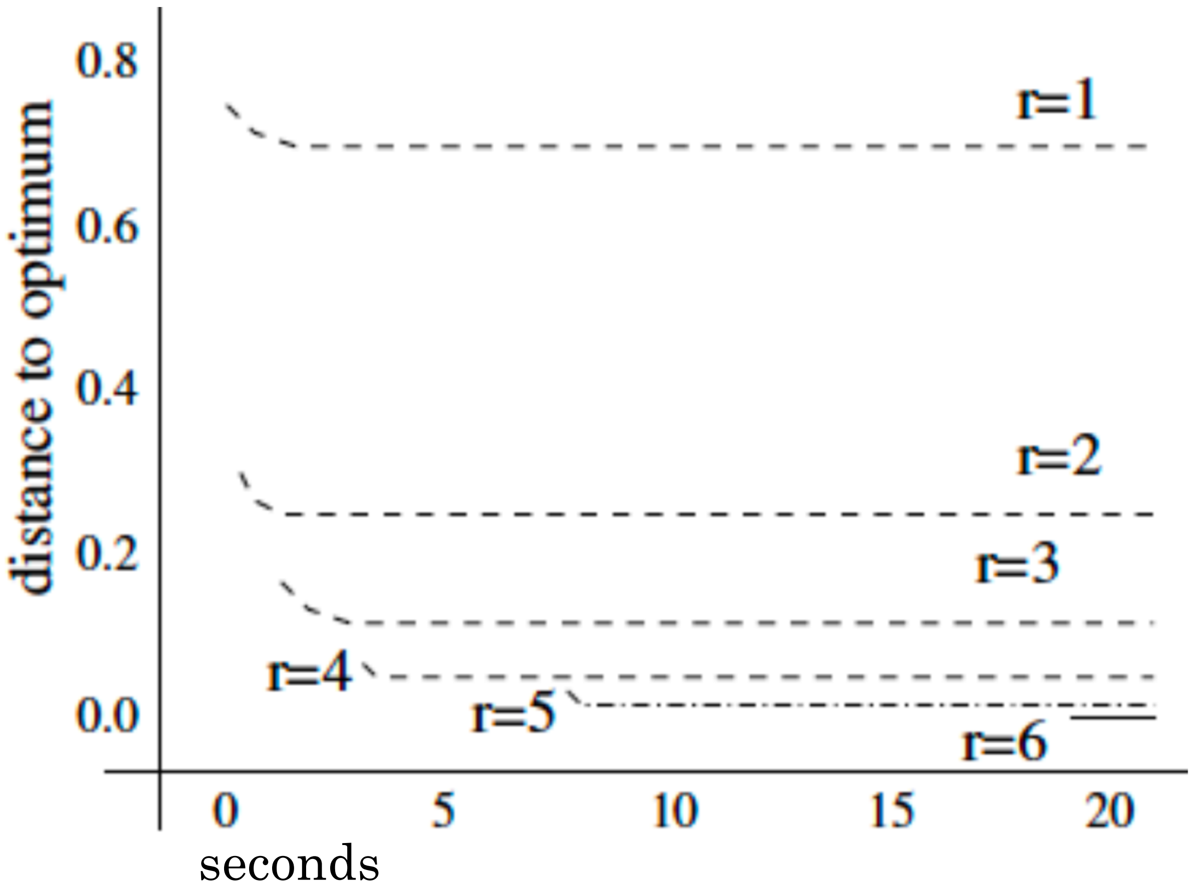
If the number of subfunctions is m=O(N)
The number of pairs must be linear
And less than $m*2^k$

What about looking 3 or more moves ahead?

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

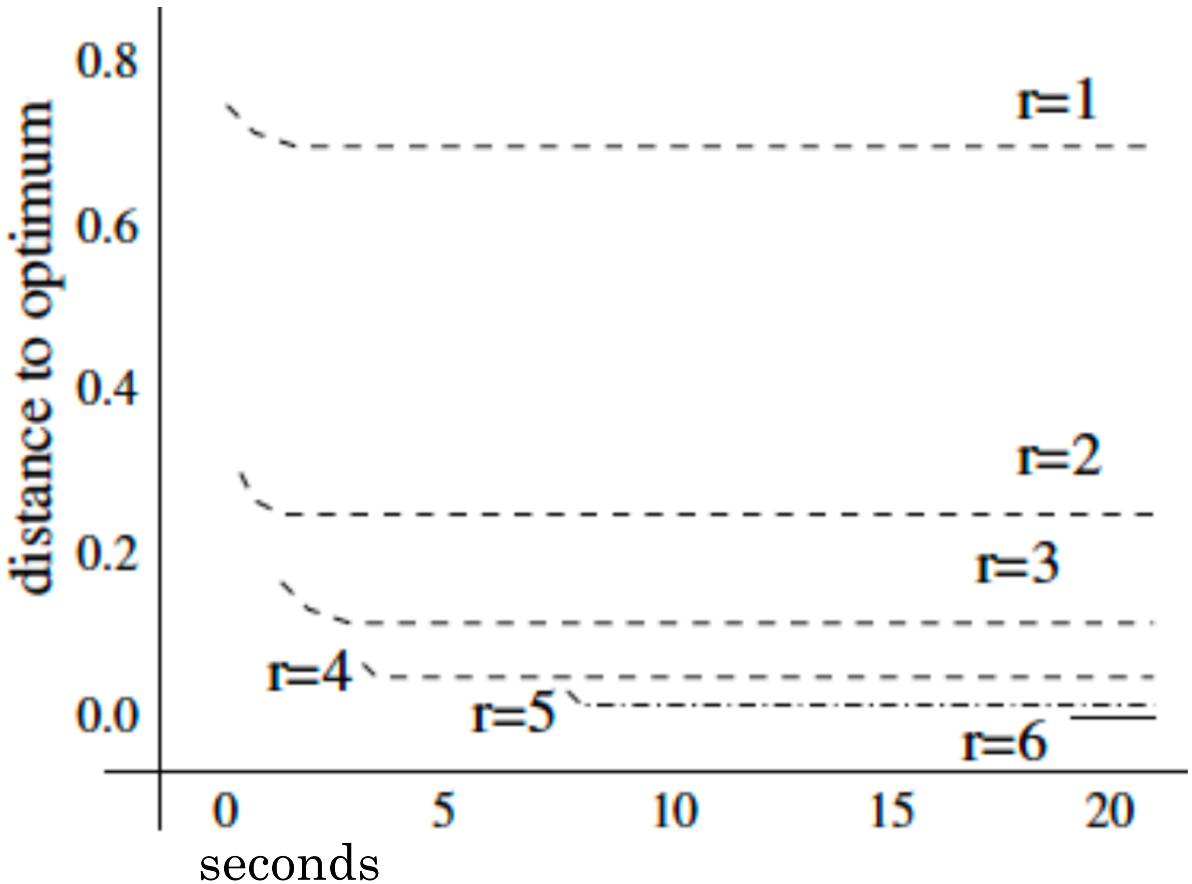
There are approximately 3n moves (60)
NOT (n choose 3) = 816.

INTELLIGENT LOCAL SEARCH LOOKING 2, 3, 4, 5, 6 BITS AHEAD



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

INTELLIGENT LOCAL SEARCH LOOKING 2, 3, 4, 5, 6 BITS AHEAD



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

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

What about Recombination?

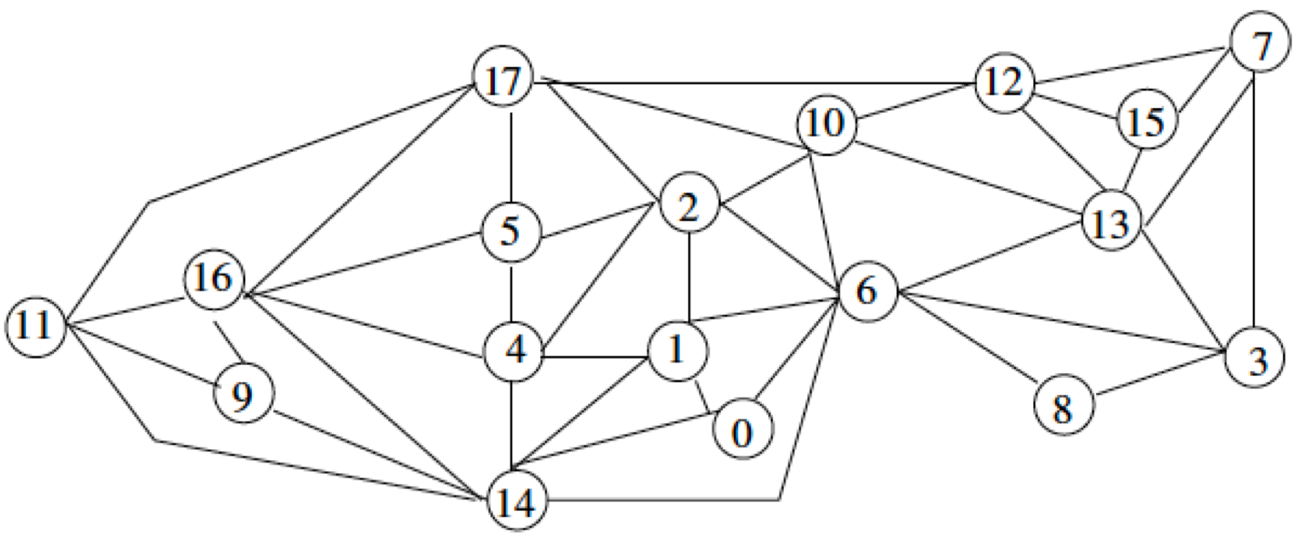
$f_a(1,0,6)$	$f_l(6,10,13)$	$f_q(11,16,17)$	$f_v(15,7,13)$
$f_b(2,1,6)$	$f_m(8,3,6)$	$f_r(12,10,17)$	$f_w(16,9,11)$
$f_c(1,2,4)$	$f_n(7,12,15)$	$f_s(13,12,15)$	$f_x(17,5,16)$
$f_d(4,1,14)$	$f_o(9,11,14)$	$f_t(14,4,16)$	$f_y(3,7,13)$
$f_e(5,4,2)$	$f_p(10,2,17)$	$f_u(9,14,16)$	$f_z(0,6,14)$

We could consider an NK-Landspace

The variables interactions are the same.

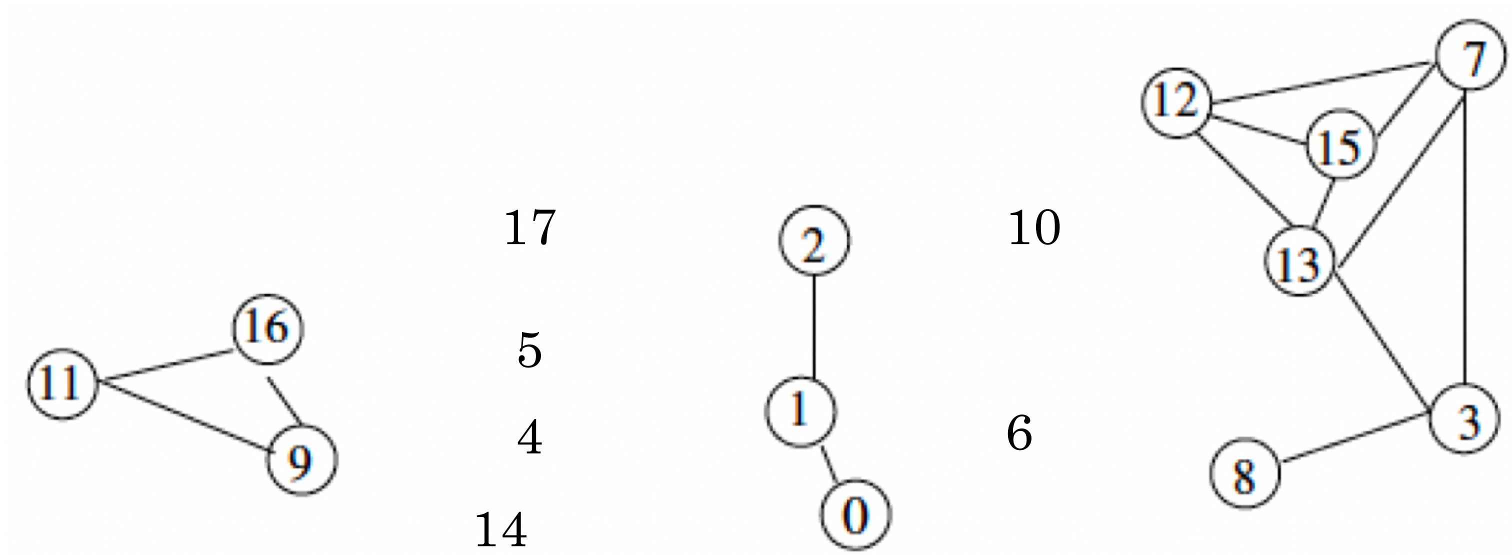
Note we have named the subfunctions: a to z.

The Variable Interaction Graph



LOCAL OPTIMUM P1: 000000000000000000
LOCAL OPTIMUM P2: 111100011101110110

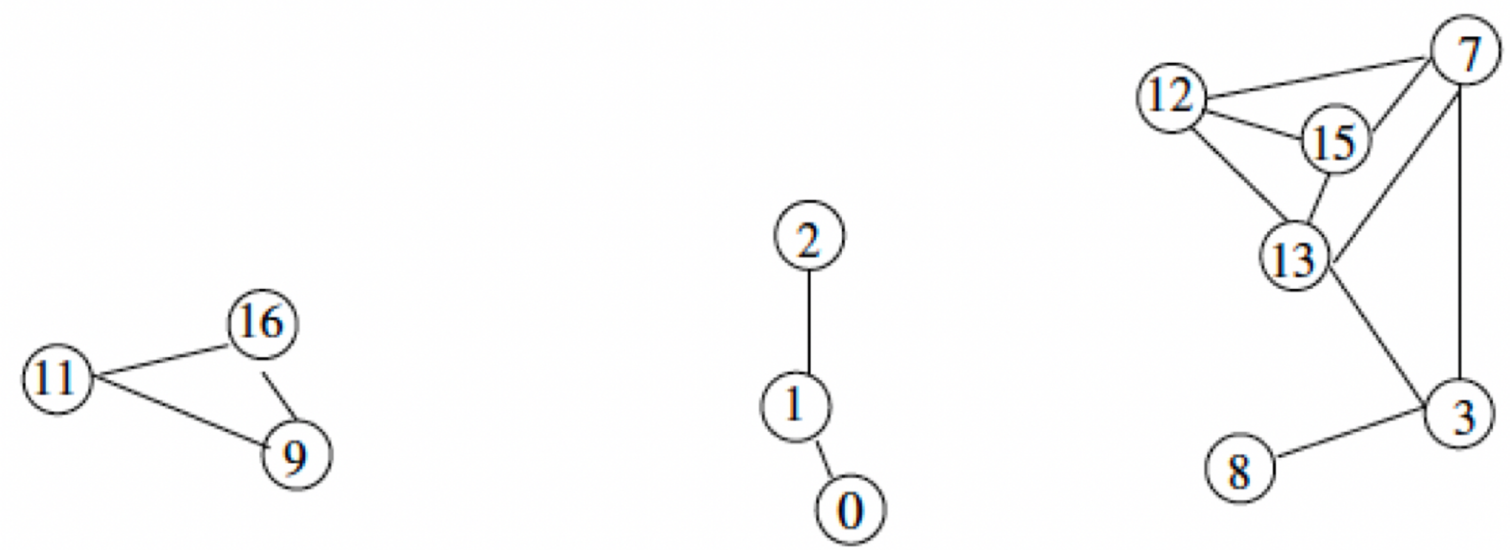
The Recombination Graph:
Parent 1: 00000000000000000
Parent 2: 11110001110111011 0



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

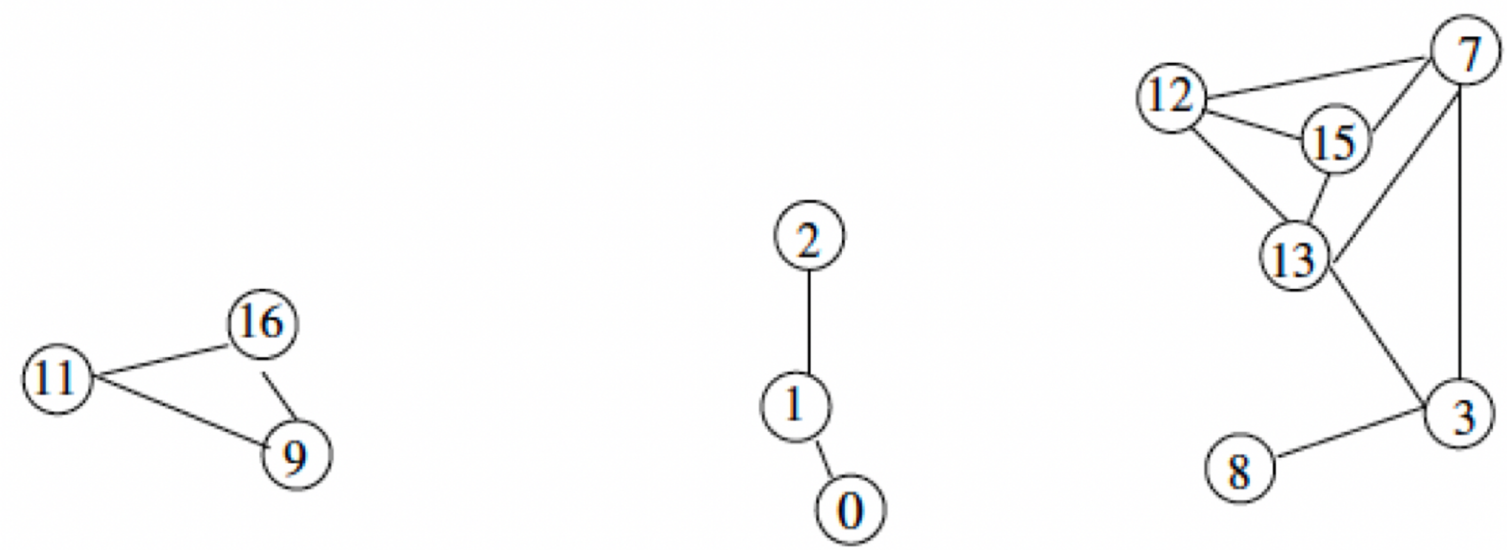
The Recombination Graph: the Decomposed VIG.



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.

The Recombination Graph: Be GREEDY

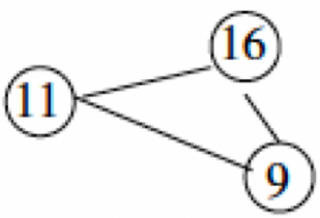


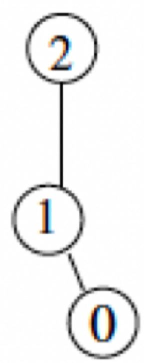
q, o, u, w, x, t　　a, b, c, d, e, p, z　　l, m, n, r, s, v, y

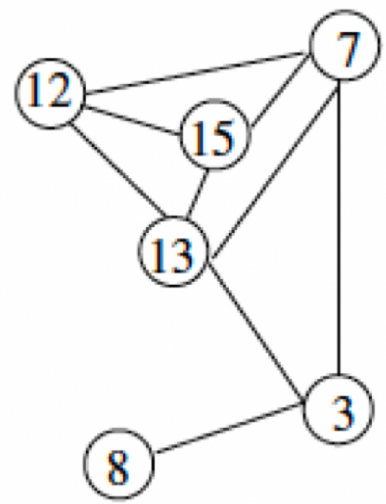
This decomposes the variables **and** the subfunctions.

The Recombination Graph: Be GREEDY

Which is Best?
P1 or P2?







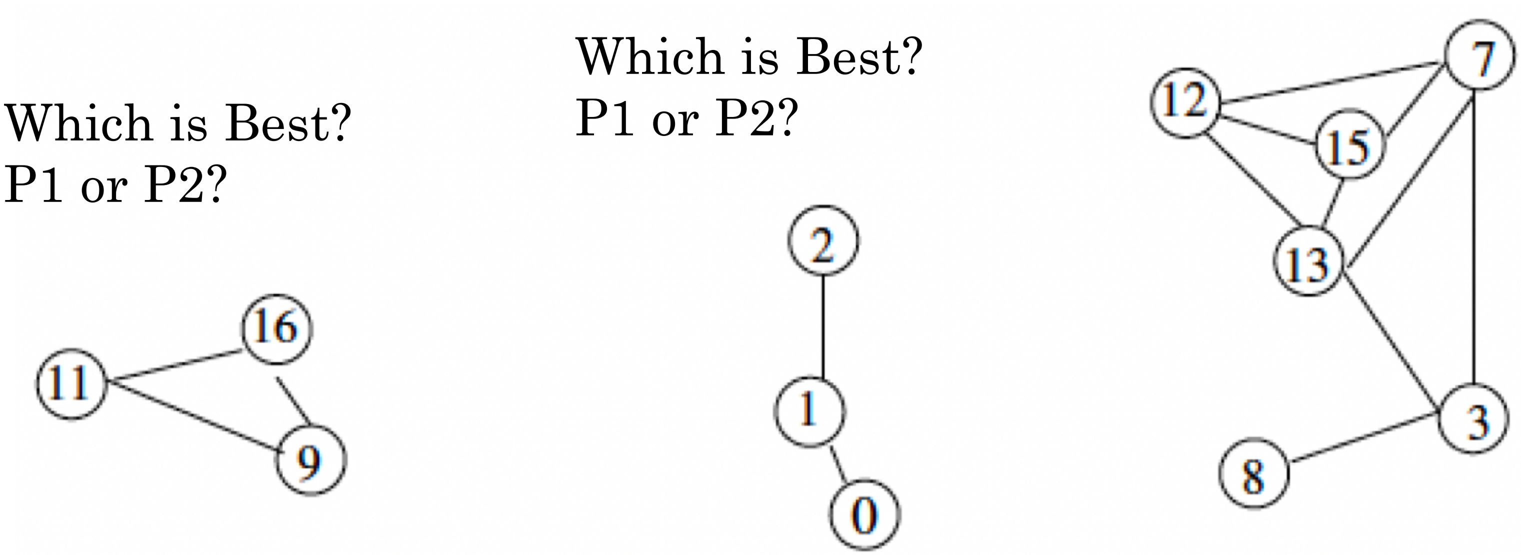
q, o, u, w, x, t

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

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

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

The Recombination Graph:
Be GREEDY

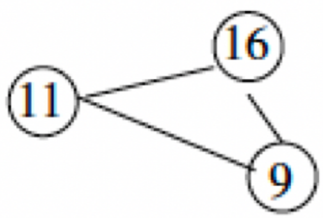


q, o, u, w, x, t   a, b, c, d, e, p, z   l, m, n, r, s, v, y

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

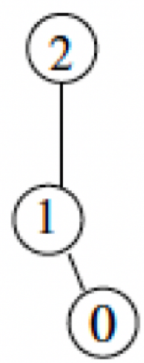
The Recombination Graph:
Be GREEDY

Which is Best?
P1 or P2?



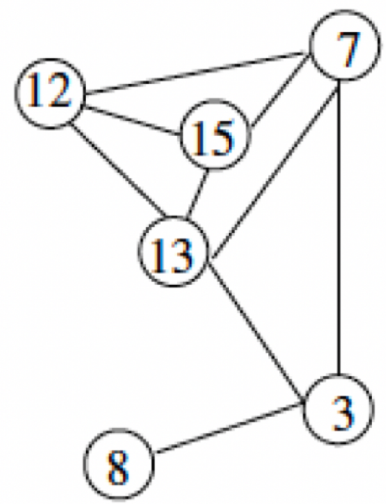
q, o, u, w, x, t

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.

Partition Crossover and Local Optima.

The Subspace Optimality Theorem:

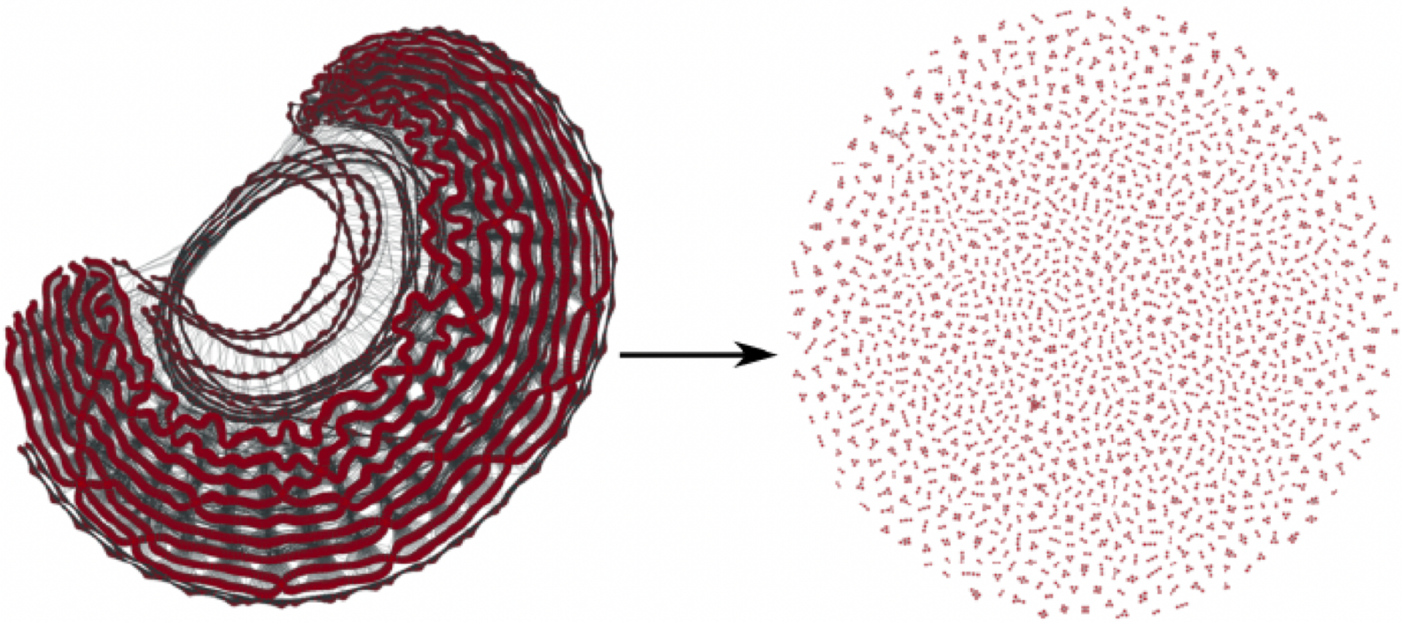
For any k-bounded pseudo-Boolean function, *f:*

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

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

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.

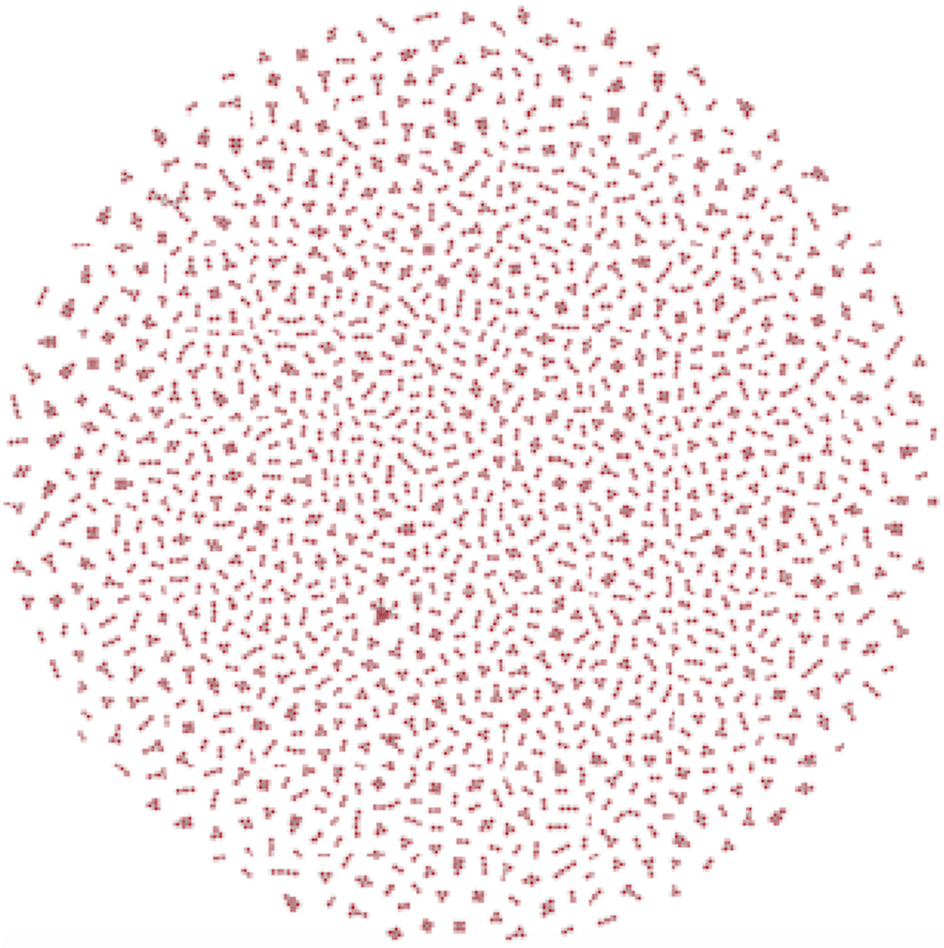
N = 1,067,657

(Thanks to Wenxiang Chen)

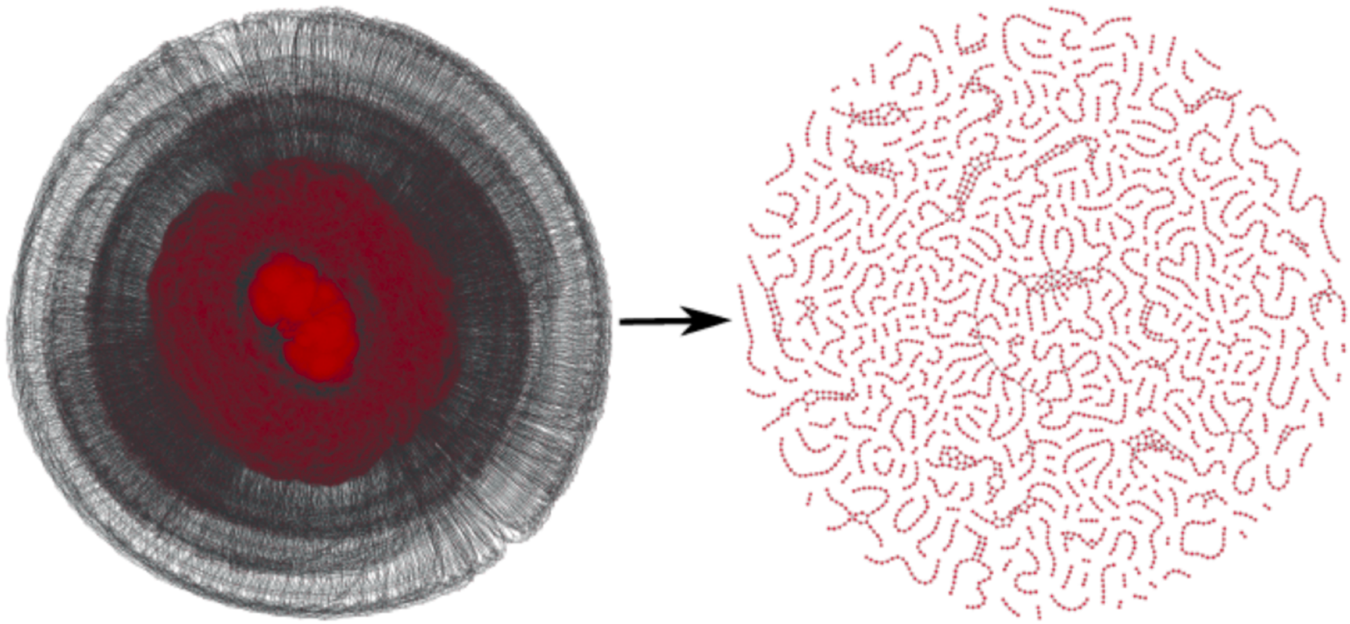
Decomposed Evaluation for MAXSAT

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

All offspring are Local Optima in this subspace.



More MAXSAT



LABS_n088_goal008

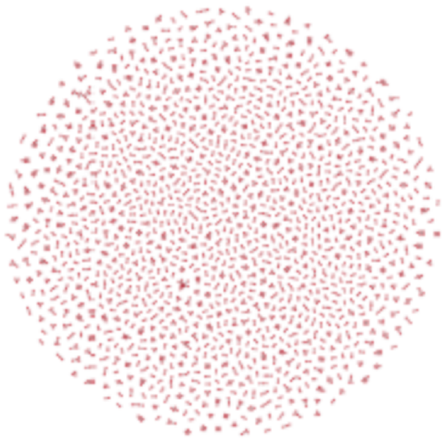
Finding low autocorrelation binary sequence: 371 components

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

N= 182,015

(Thanks to Wenxiang Chen)

More MAXSAT



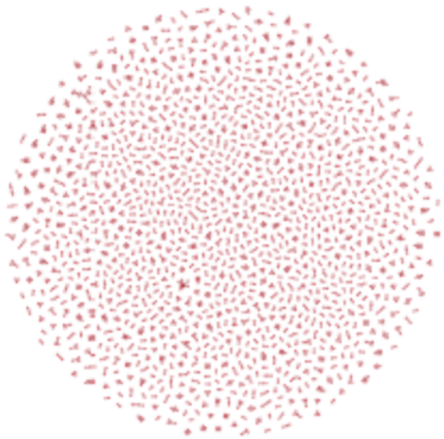
atco_enc3_opt1_13_48

LABS_n088_goal008

SAT_instance_N=49

aaai10-ipc5

More MAXSAT



atco_enc3_opt1_13_48

LABS_n088_goal008

SAT_instance_N=49

aaai10-ipc5

These subproblems have a tree decomposition with low width.

Thanks to Francisco Chicano.

These subproblems can be solved by Dynamic Programming!

Partition Crossover and Local Optima.

The Subspace Optimality Theorem:

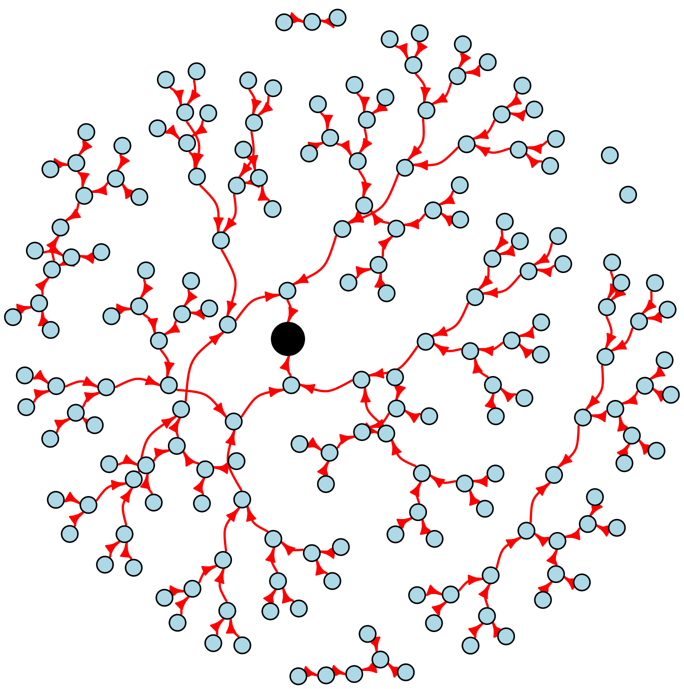
For any k-bounded pseudo-Boolean function, *f:*

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

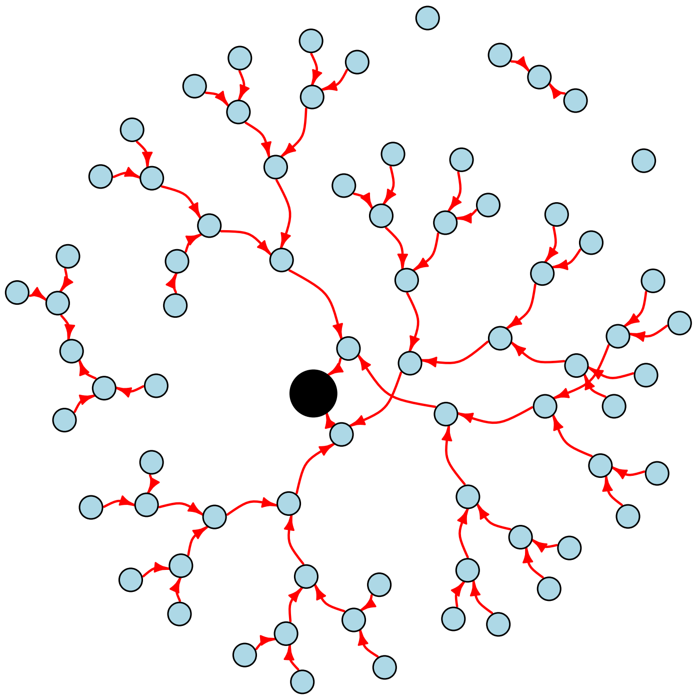
TUNNELING BETWEEN OPTIMA in O(N) time.

Tunneling Between Local Optima.

Local Optima Linked by Crossover, Thanks to Gabriela Ochoa.



Adjacent NK Landscape

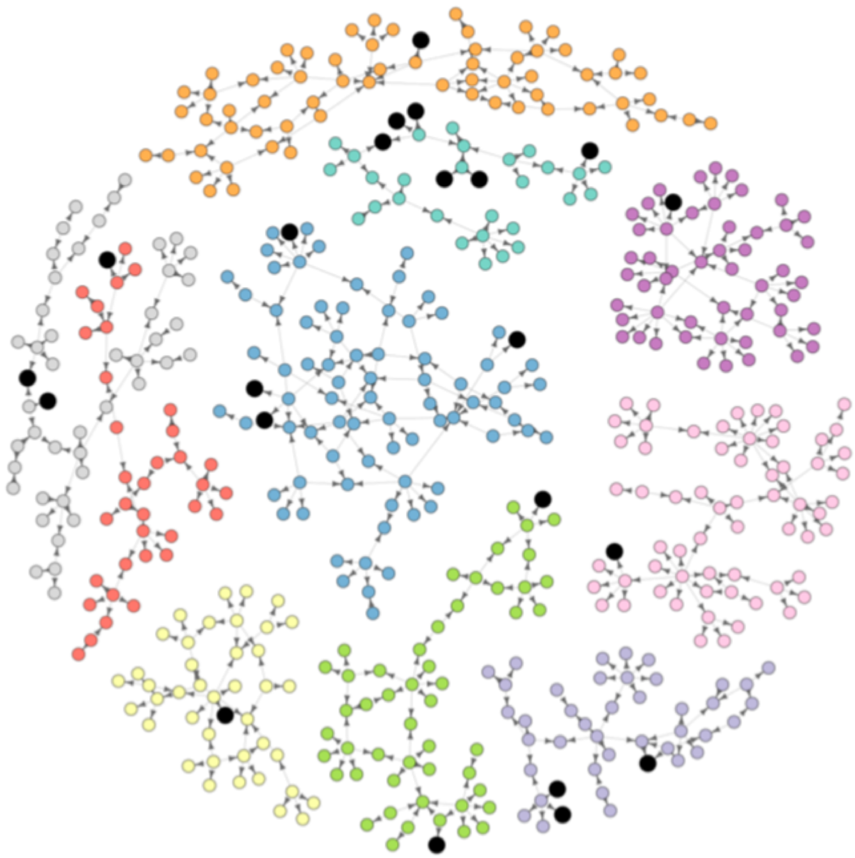


Random NK Landscape

The Traveling Salesman

Tunneling Between Local Optima

Local Optima are "Linked" by Partition Crossover

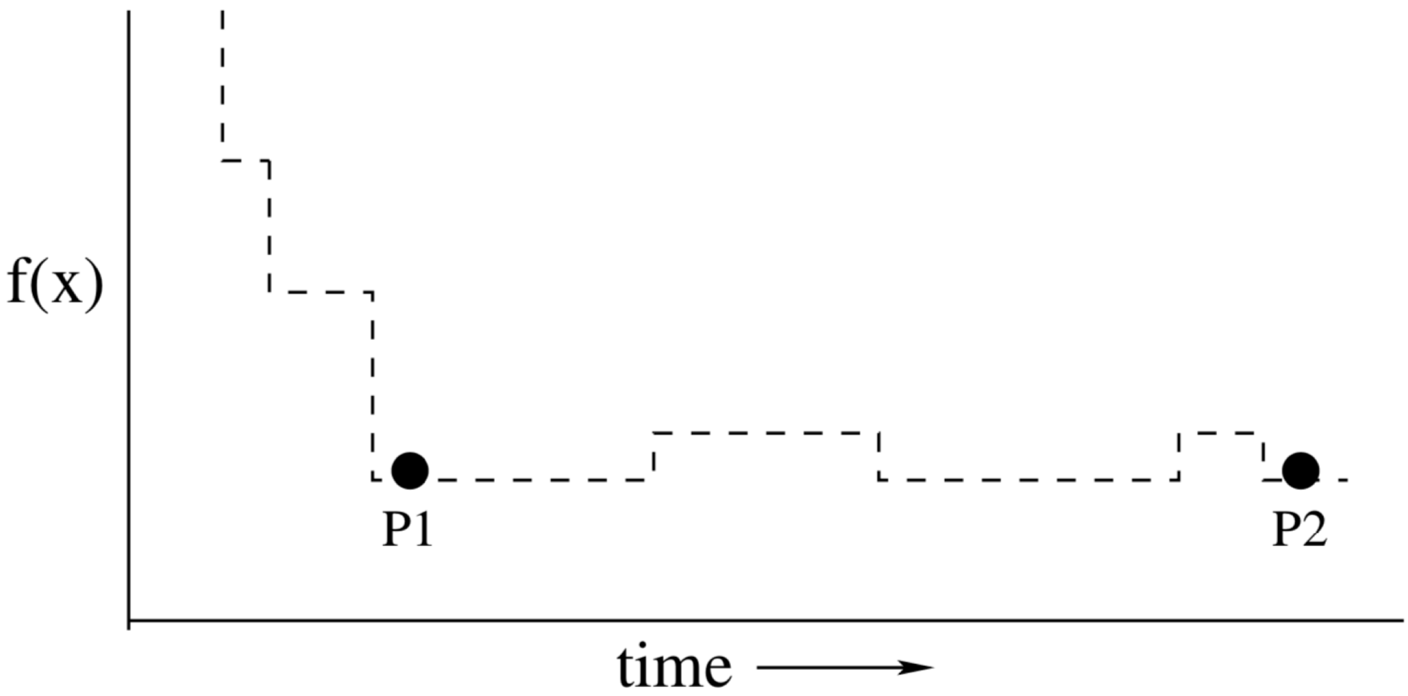


Thanks to G. Ochoa and N. Veerapen.

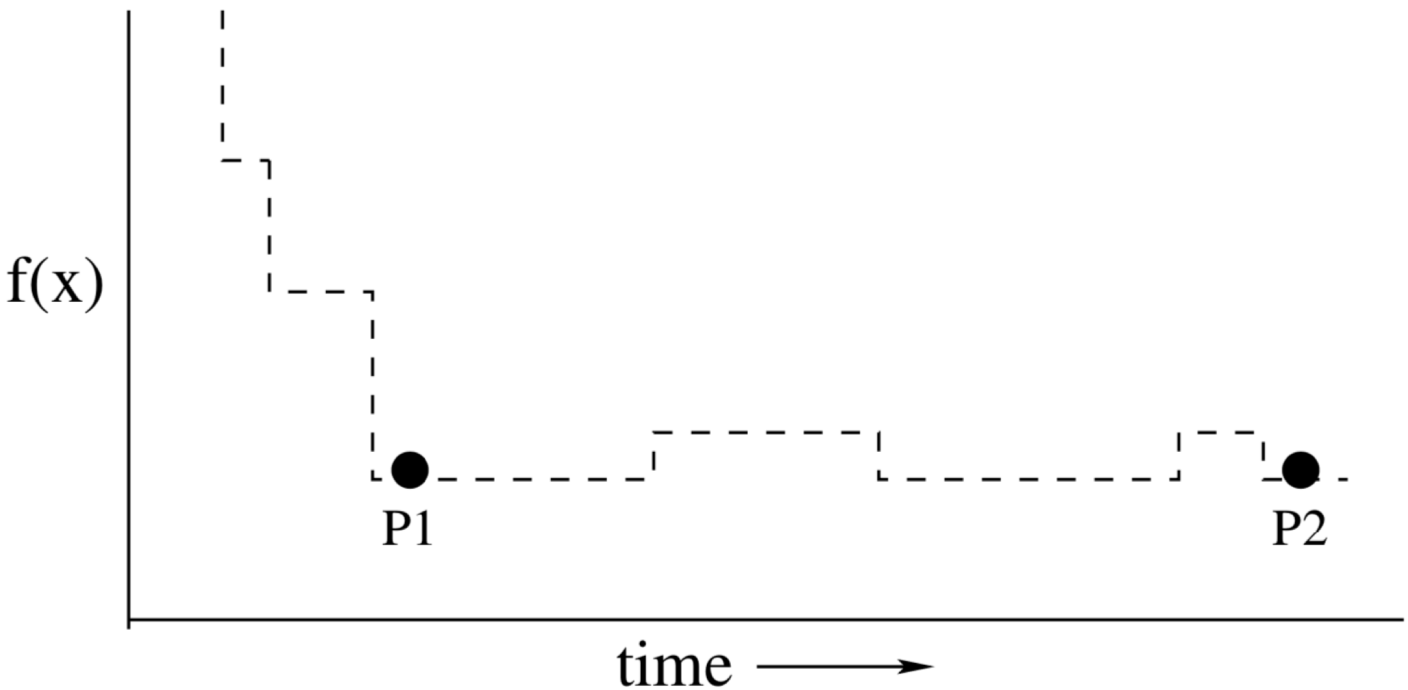
These were found using Chained-LK.

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

Max-3SAT and Plateaus



MAX-3SAT AND PLATEAUS



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

Local Search Algorithms for MAXSAT

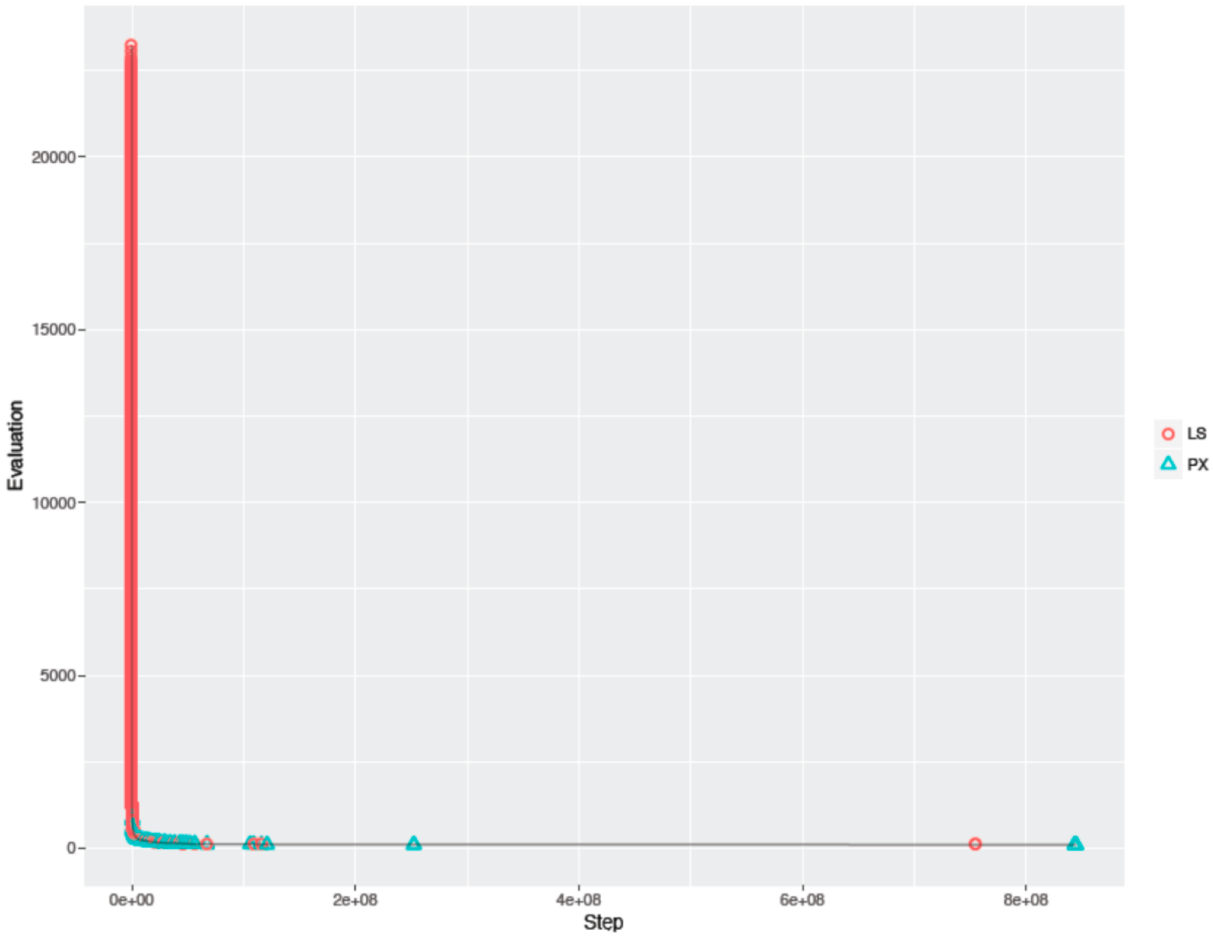
AdaptG^2WSAT: Best in the 2007 SAT Competition

NEW: AdaptG^2WSAT with Partition Crossover

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

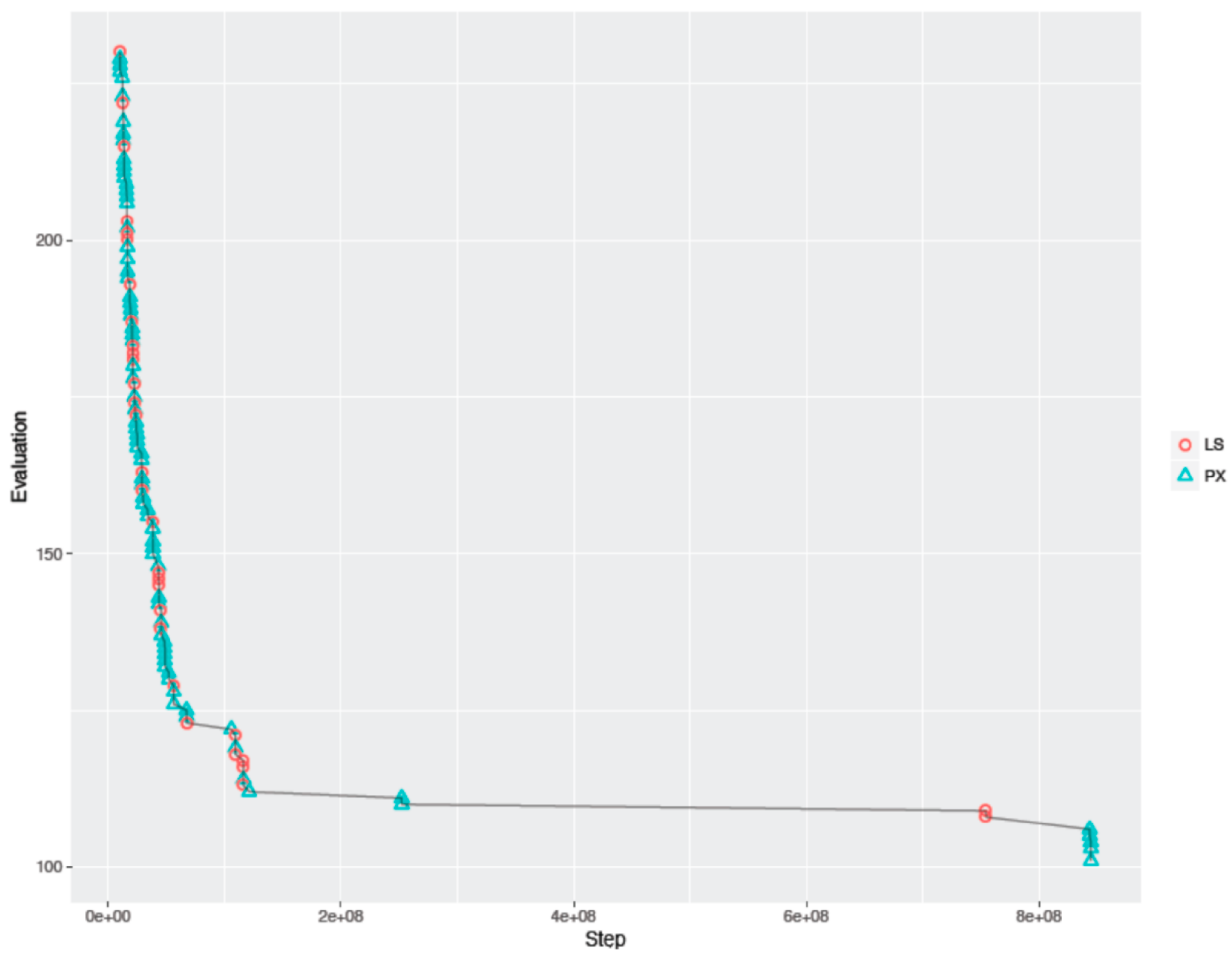
NEW: Sparrow with Partition Crossover

Early MAXSAT Results

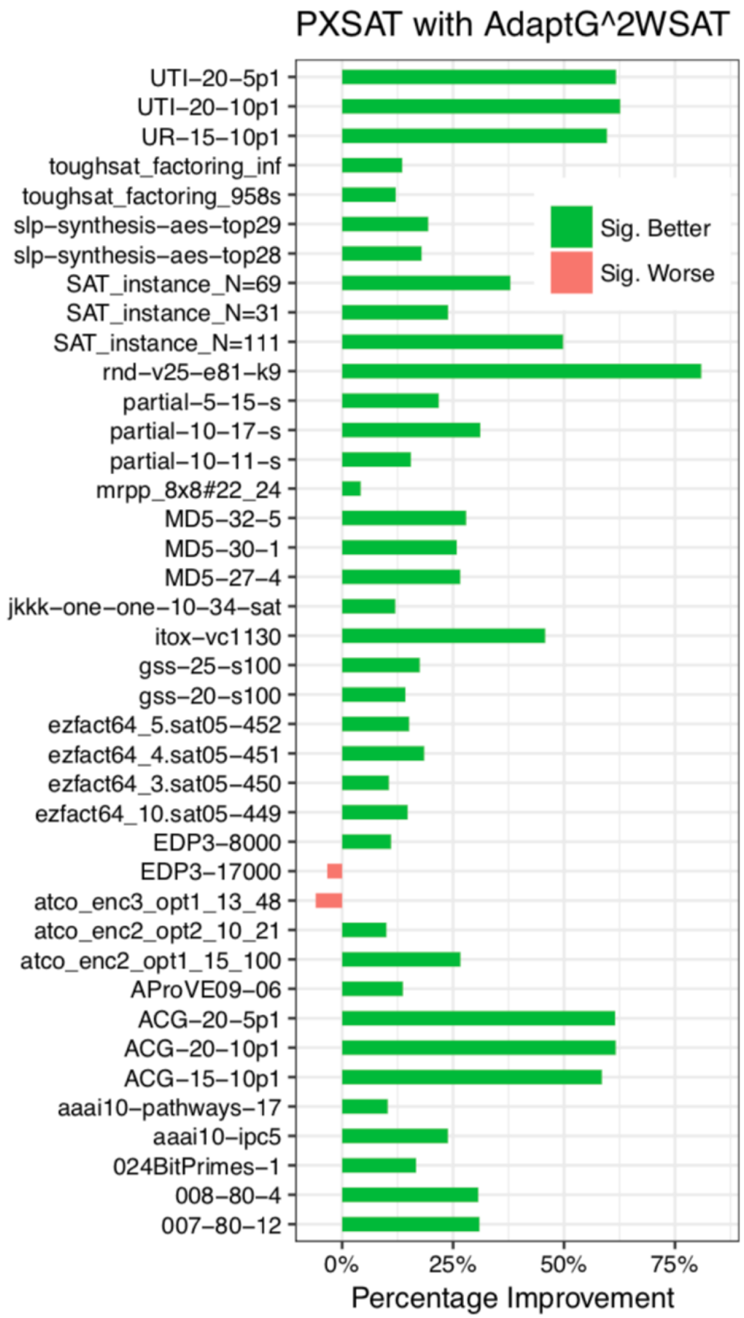


RED is intelligent local search. BLUE is Partition Crossover

Early MAXSAT Results



RED is intelligent local search.   BLUE is Partition Crossover



MAXSAT Results.

Partition Crossover helps on Hard Problems.

MAXSAT RESULTS

Theorem

When recombining parents P_1 and P_2:

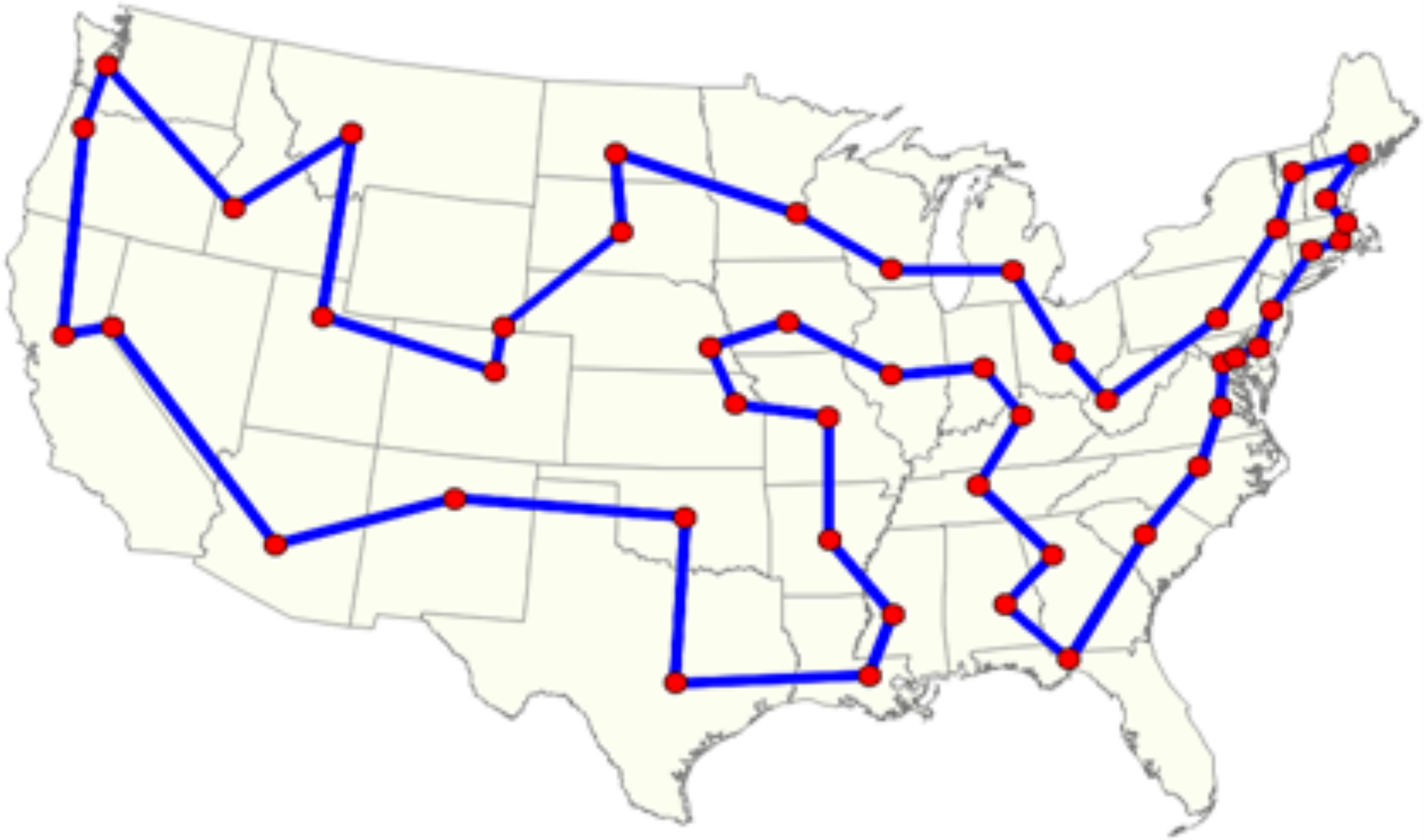
$$\frac{f(P_1)}{2} + \frac{f(P_2)}{2} = \frac{1}{2^q} \sum_{i=1}^{2^q} f(C_i)$$

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.*

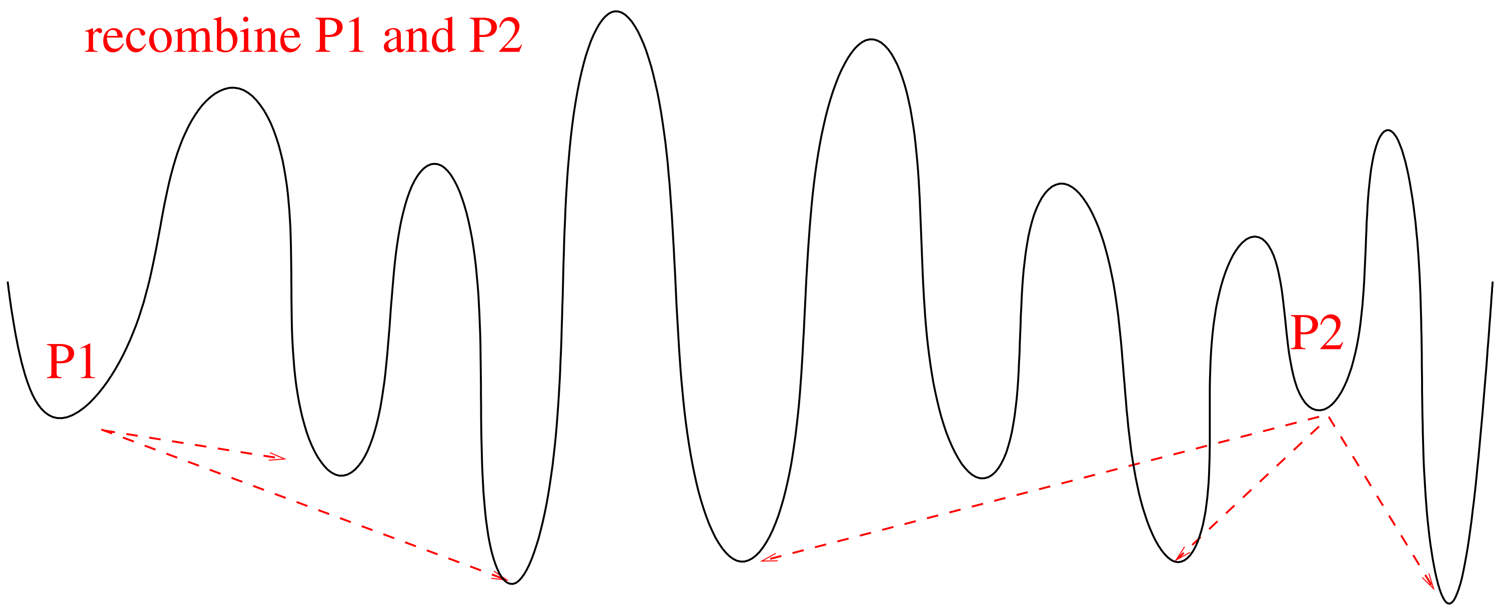
This makes Partition Crossover very different than local search for MAXSAT. For local search the discovery of a disapproving move says nothing about the existence of an improving move.

The Traveling Salesman Problem



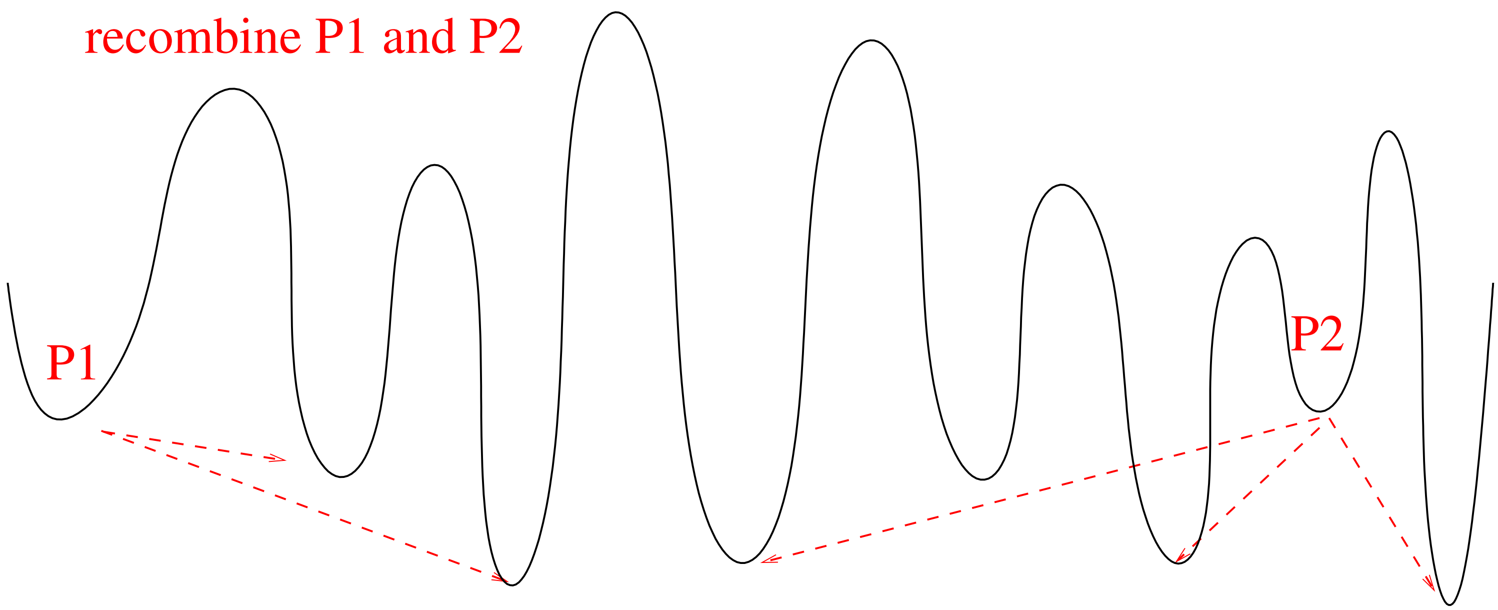
What is the shortest circuit that visits the 50 state capitals?

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



We can remove randomness from Crossover

First Apply Intelligent Local Search!



First Apply Intelligent Local Search!

Naïve 2-Opt is $O(N^3)$ in complexity!

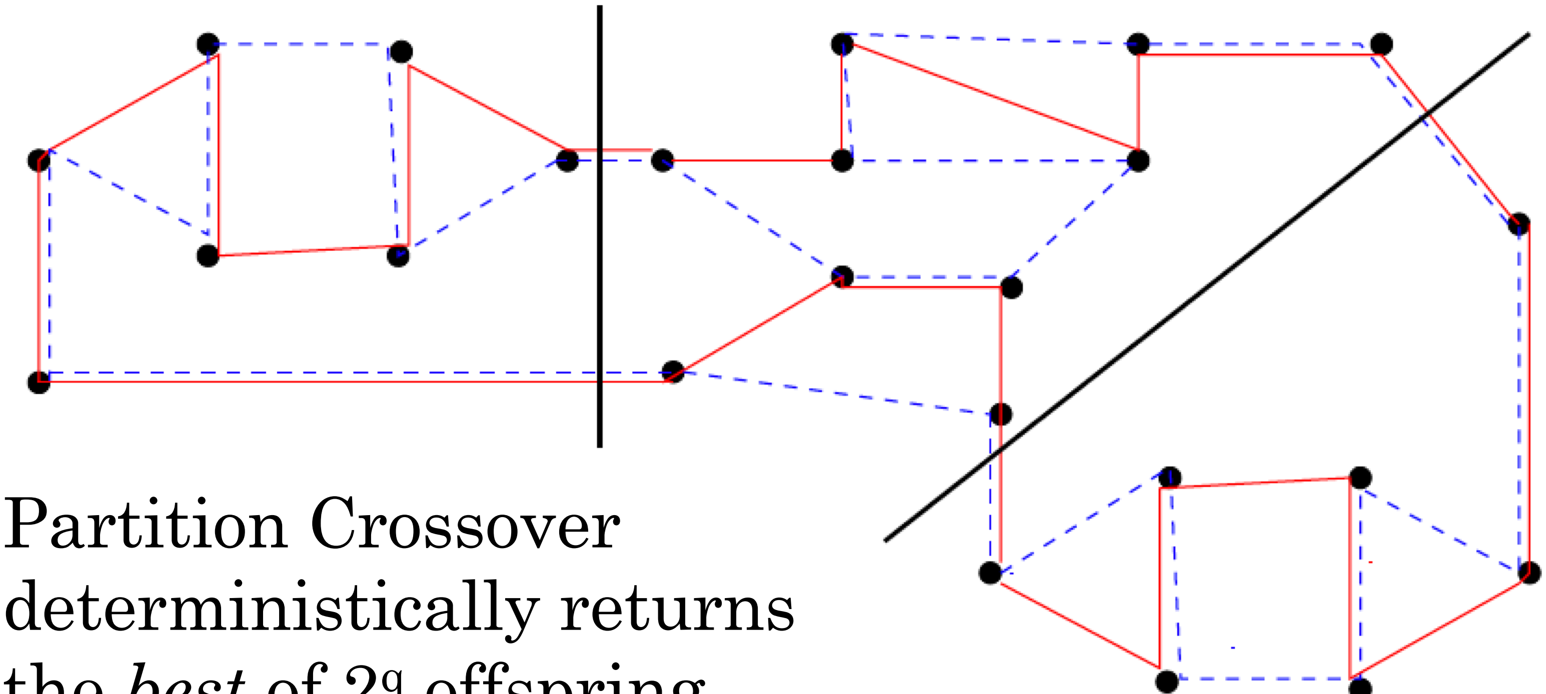
Intelligent 2-Opt is $O(N)$.

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

THIS IS NOT BLACK BOX.

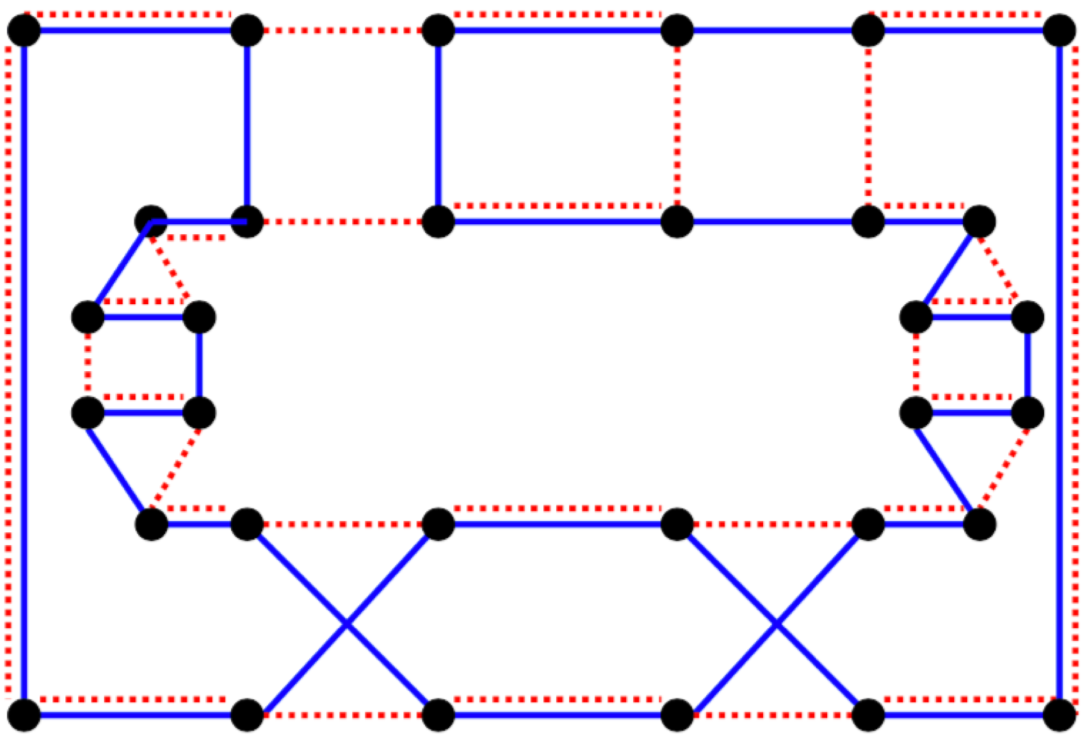
Can We ``Tunnel'' between Optima?

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

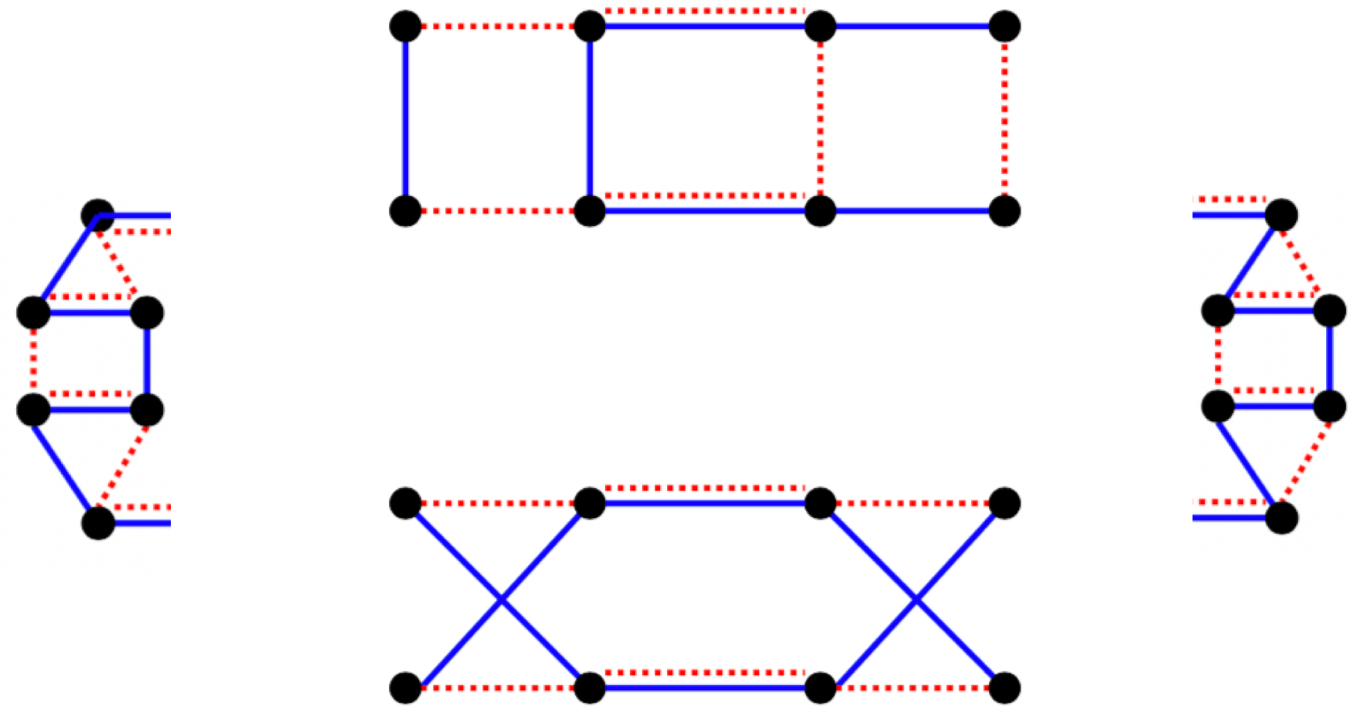


Partition Crossover deterministically returns the *best* of 2^q offspring.

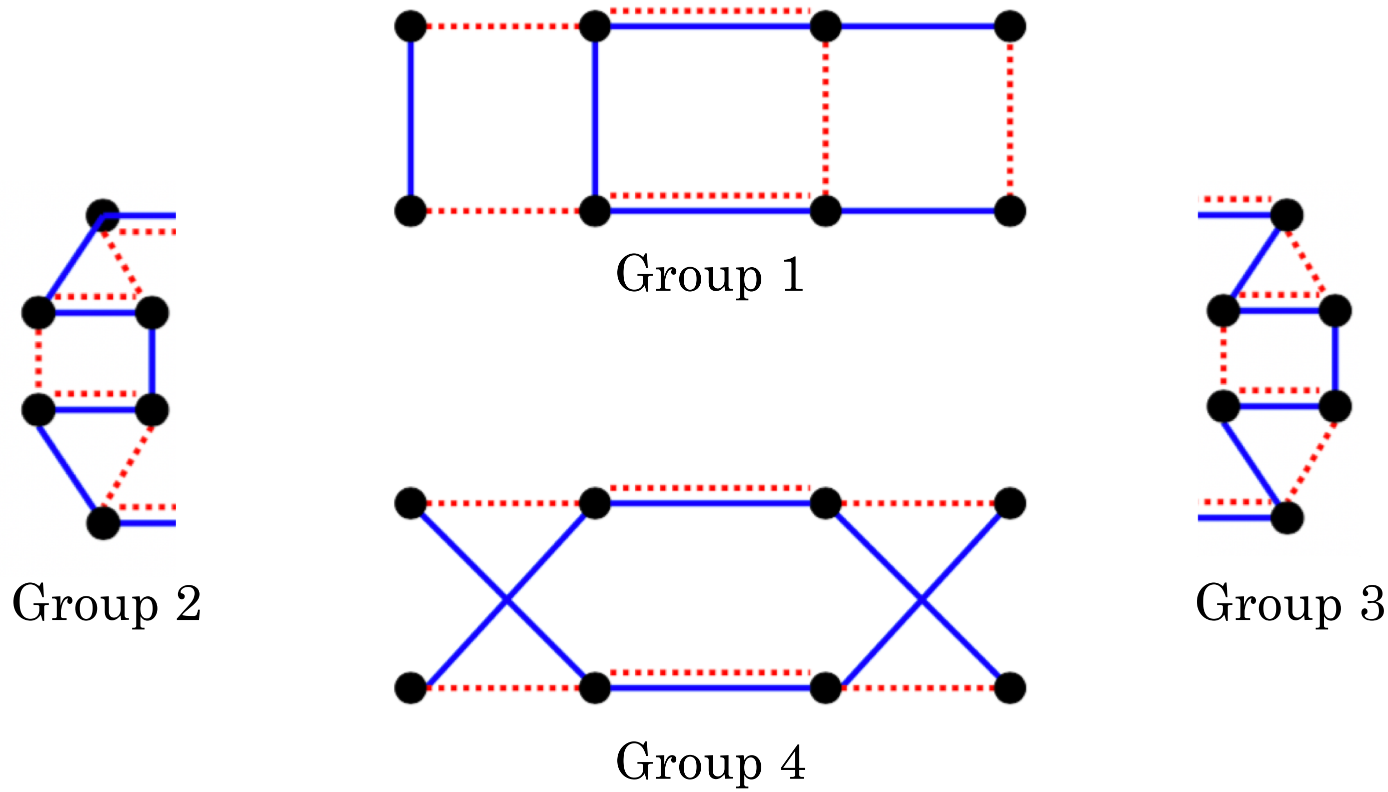
Partition Crossover and TSP



Partition Crossover and TSP



Partition Crossover and TSP



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

THE QUASI-LOCAL OPTIMA FORM A LATTICE IN HYPERSPACE:

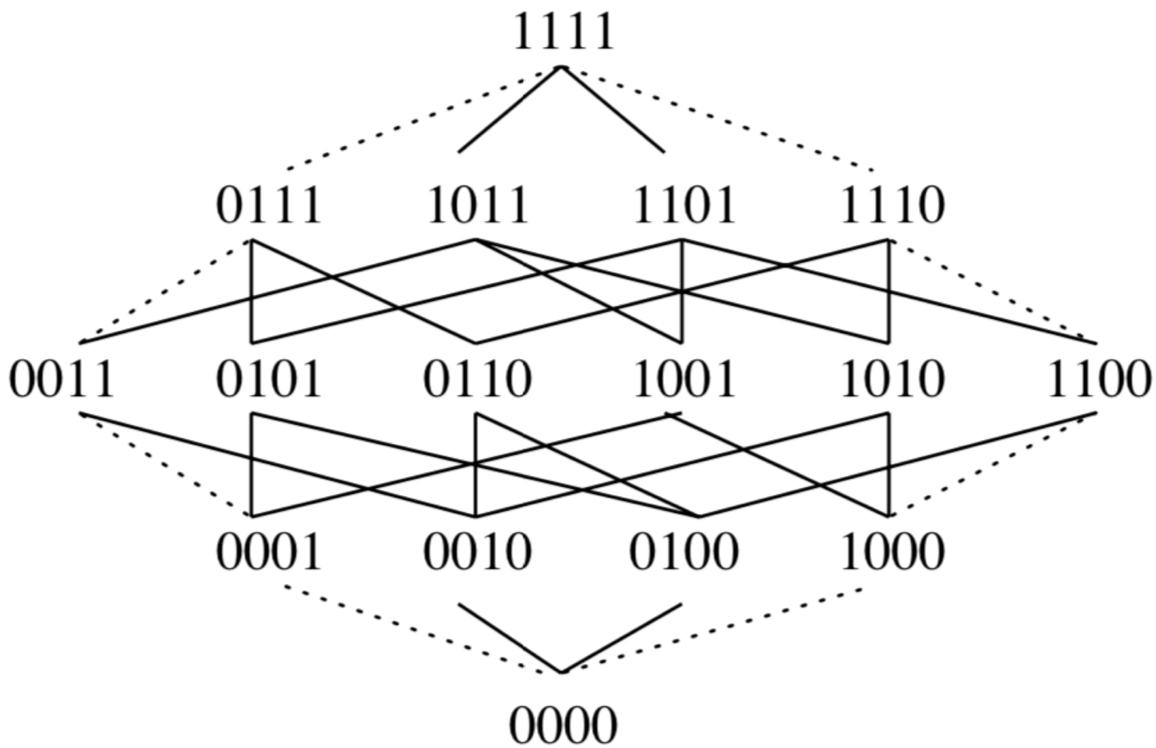
Assume you have these connected groups of variables during recombination.

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.

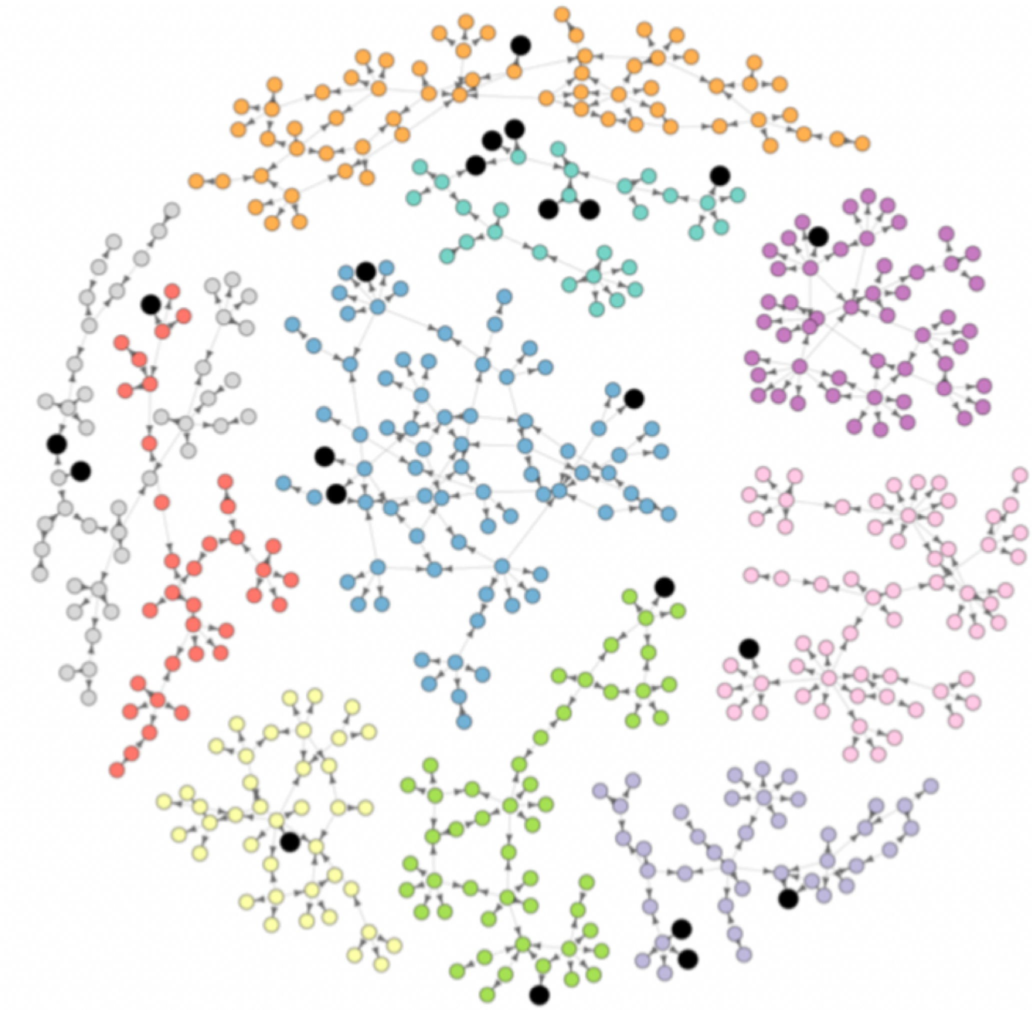
THE QUASI-LOCAL OPTIMA FORM A LATTICE IN HYPERSPACE:

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

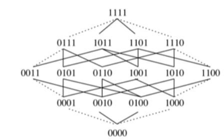


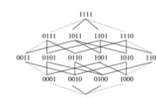
ALL of the 16 solutions are LOCAL OPTIMA In the Hyperplane Subspace.

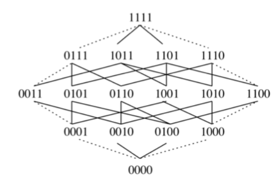
THESE TUNNELS ARE JUST THE **TOPS** OF LATTICES OF QUASI LOCAL OPTIMA.

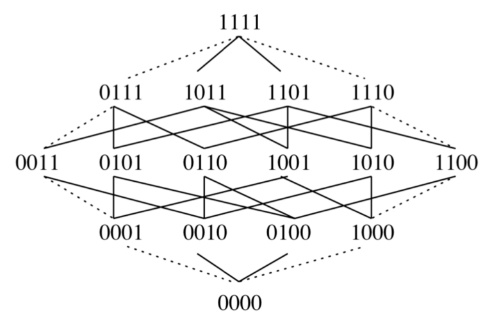


Each tunnel is one recombination,
and each recombination is the top of a lattice.





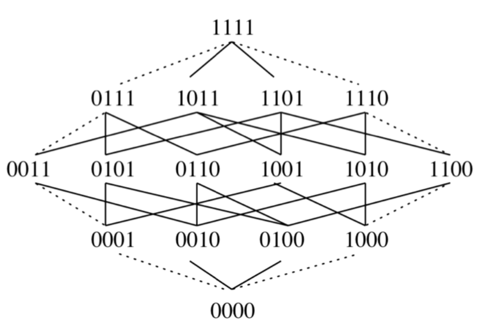


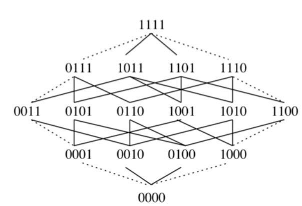


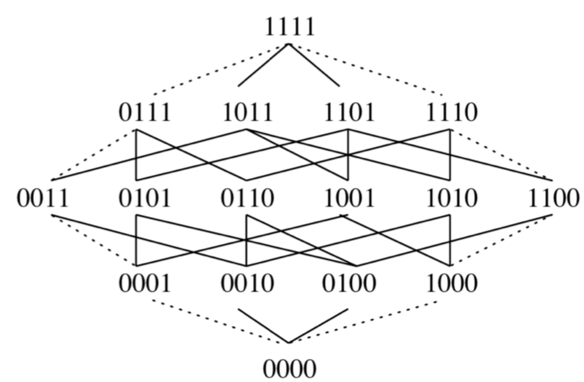
THEOREM: A LATTICE OF QUASI-LOCAL OPTIMA CAN BE EXPONENTIALLY LARGE:

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.

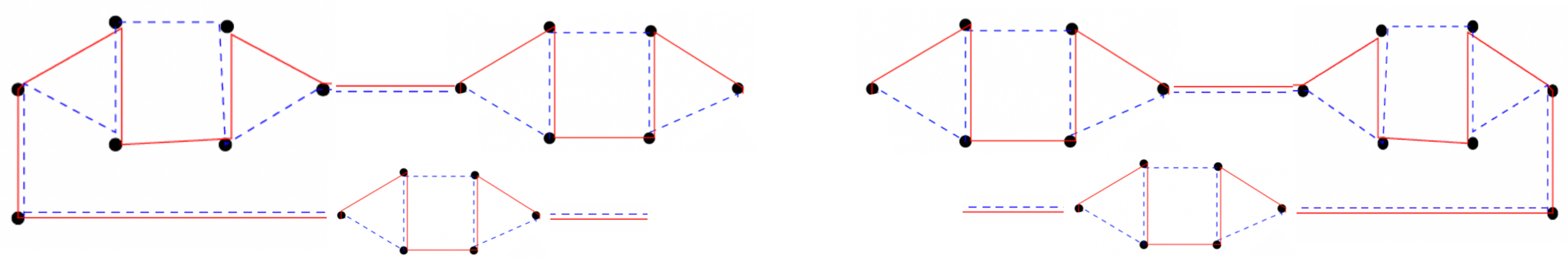
This results in a lattice of size $2^{N/c}$







THEOREM: A LATTICE OF QUASI-LOCAL OPTIMA CAN BE EXPONENTIALLY LARGE:



The construction builds a chain of recombining components.

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.

Transforms

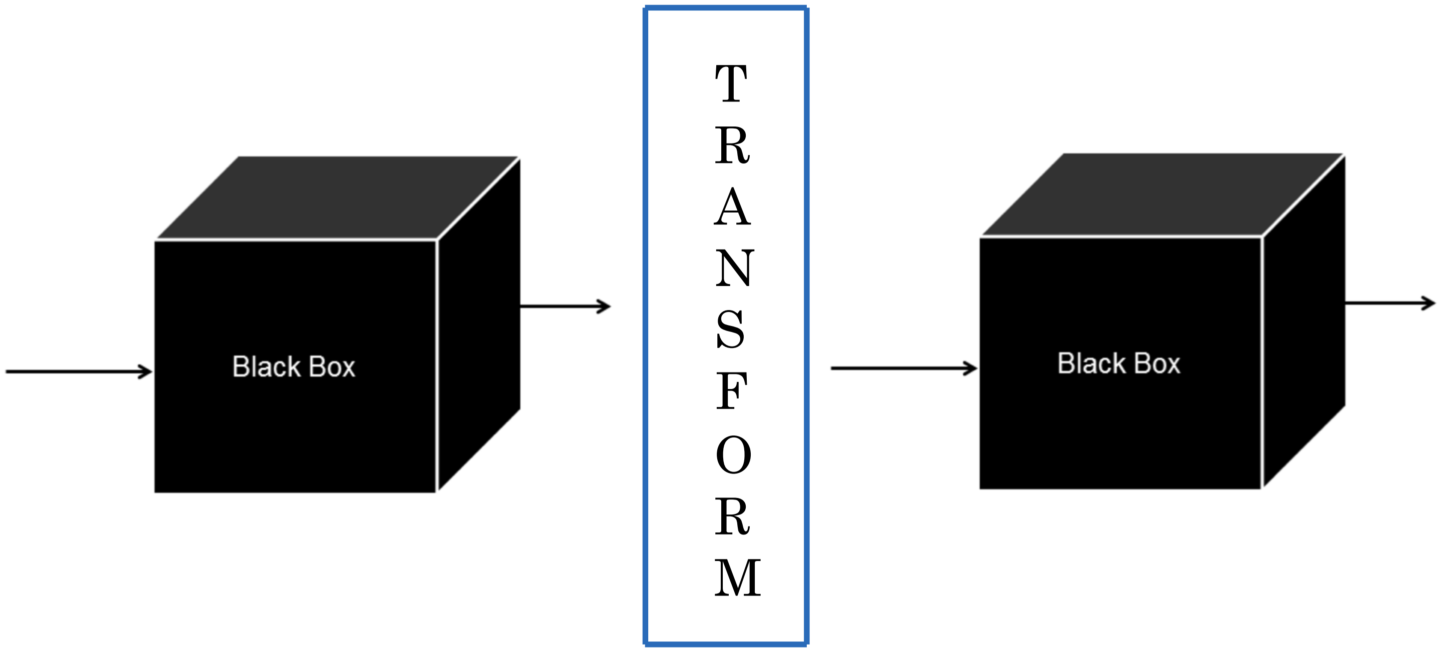
- Transforms exist for all Pseudo-Boolean Functions

"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.

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



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

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

One Last Thought:

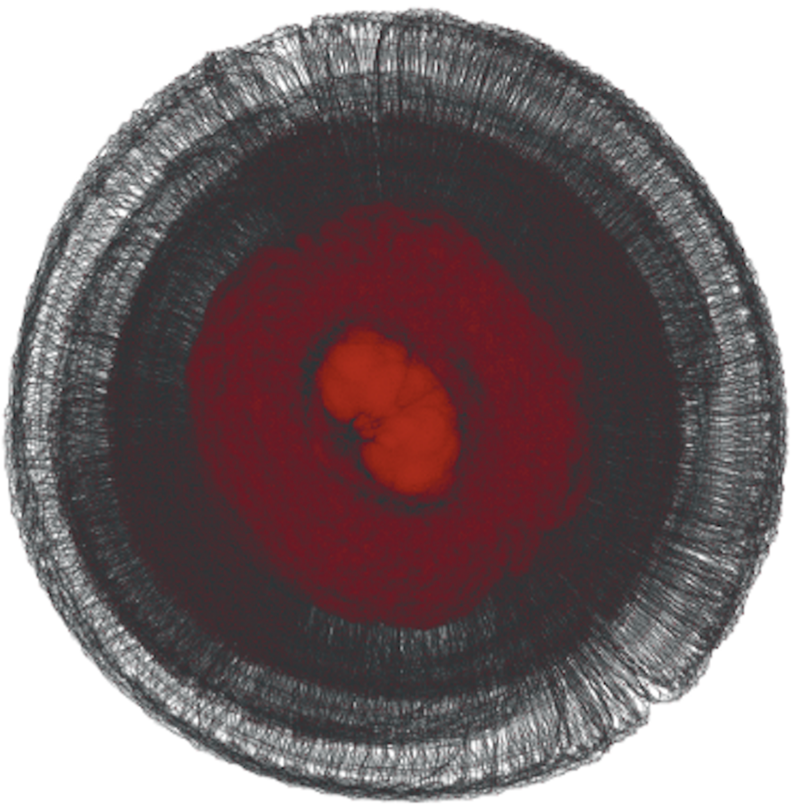
What if DNA is K-bounded?

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

One Last Thought:

What if DNA is K-bounded?

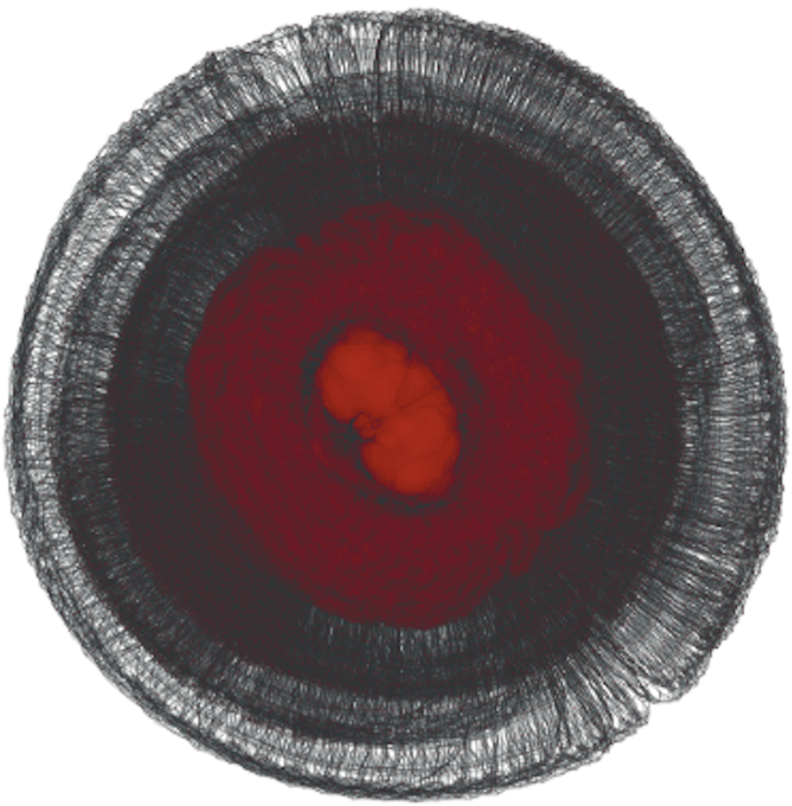
What if “gene interaction” looks like this?



One Last Thought:

What if DNA is K-bounded?

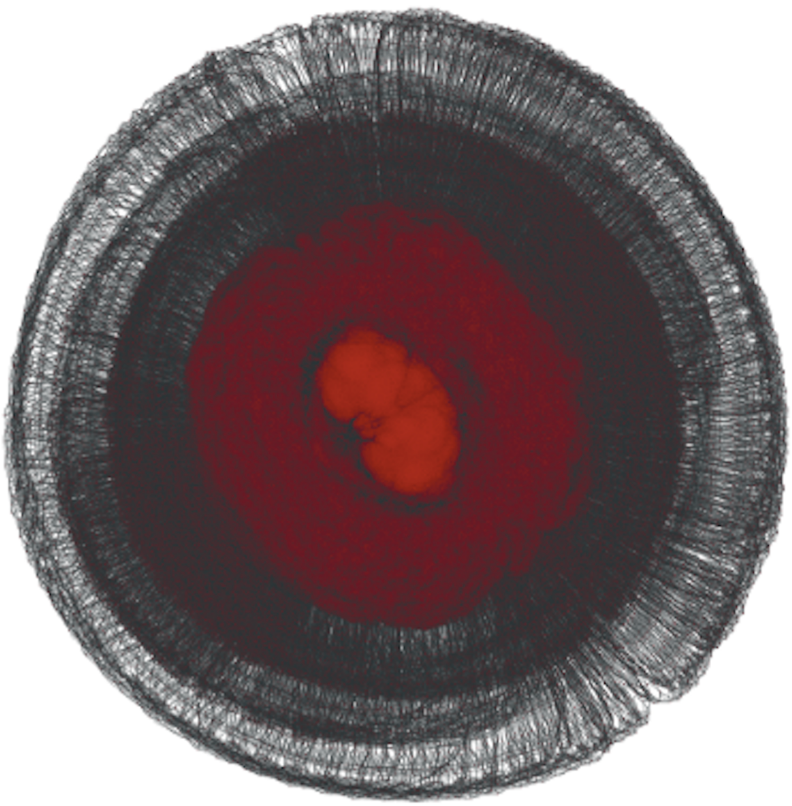
99.9% of DNA is identical in all humans

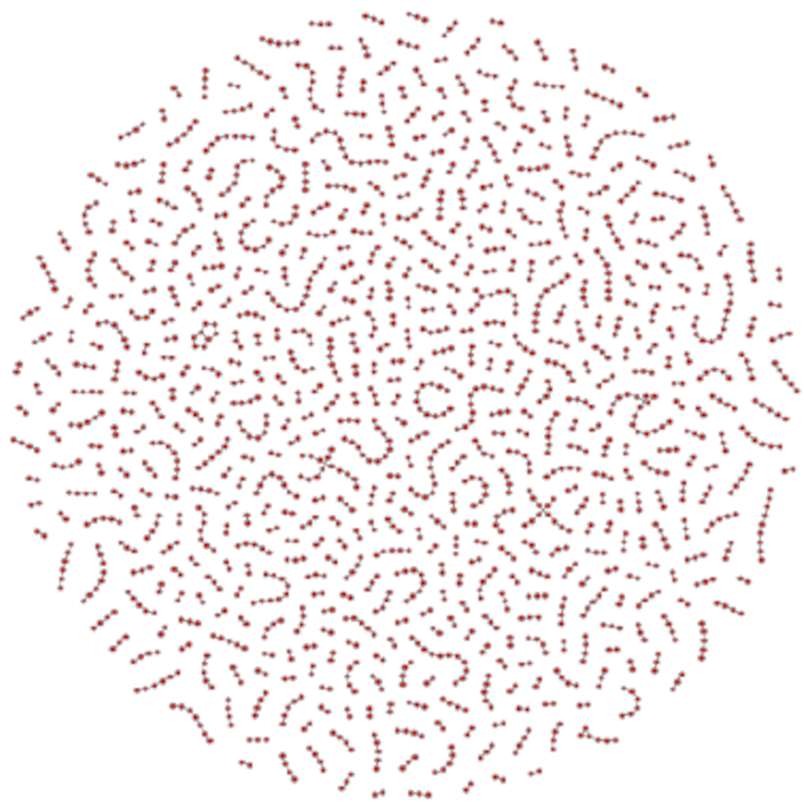


One Last Thought:

What if DNA is K-bounded?

99.9% of DNA is identical in all humans





QUESTIONS?