# When is local search both effective and efficient?

STACS 2026

March

Artem Kaznatcheev & Sofia (Vazquez Alferez)

Utrecht University

# Our Goal

Find an expressive class of problem instances where many local search algorithms are both effective and efficient.

# Fitness Landscape

Find an assignment $x$ to $n$ Boolean variables:

$$x = x_1 x_2 x_3 \ldots x_n \qquad x_i \in \{0,1\}$$

$$x \in \{0,1\}^n$$

That maximizes some fitness function

$$f : \{0,1\}^n \longrightarrow \mathbb{Z}$$

Two assignments are adjacent if they differ on exactly one bit.

e.g. 1 1 1
0 1 1

# Fitness Landscape

Find an assignment $x$ to $n$ Boolean variables:

$$x = x_1 x_2 x_3 \ldots x_n \qquad x_i \in \{0,1\}$$

$$x \in \{0,1\}^n$$

That maximizes some fitness function

$$f : \{0,1\}^n \longrightarrow \mathbb{Z}$$

Two assignments are adjacent if they differ on exactly one bit.

# (Strict) Local Search

start at some assignment $x$.

While there is an adjacent $x'$ such that $f(x') > f(x)$:

set $x \leftarrow x'$

Return $x$

fitness function

$x$ is a local peak (local optimum)

# (Strict) Local Search

start at some assignment $x$.

While there is an adjacent $x'$ such that $f(x') > f(x)$:

set $x \leftarrow x'$

Return $x$

Different rules on how to select $x'$ give rise to different local search algorithms

# (Strict) Local Search

relaxed sometimes

start at some assignment $x$.

While there is an adjacent[1] $x'$ such that $f(x') > f(x)$[2]:

set $x \leftarrow x'$

Return $x$

## Effective

Did we find a good solution?

## Efficient

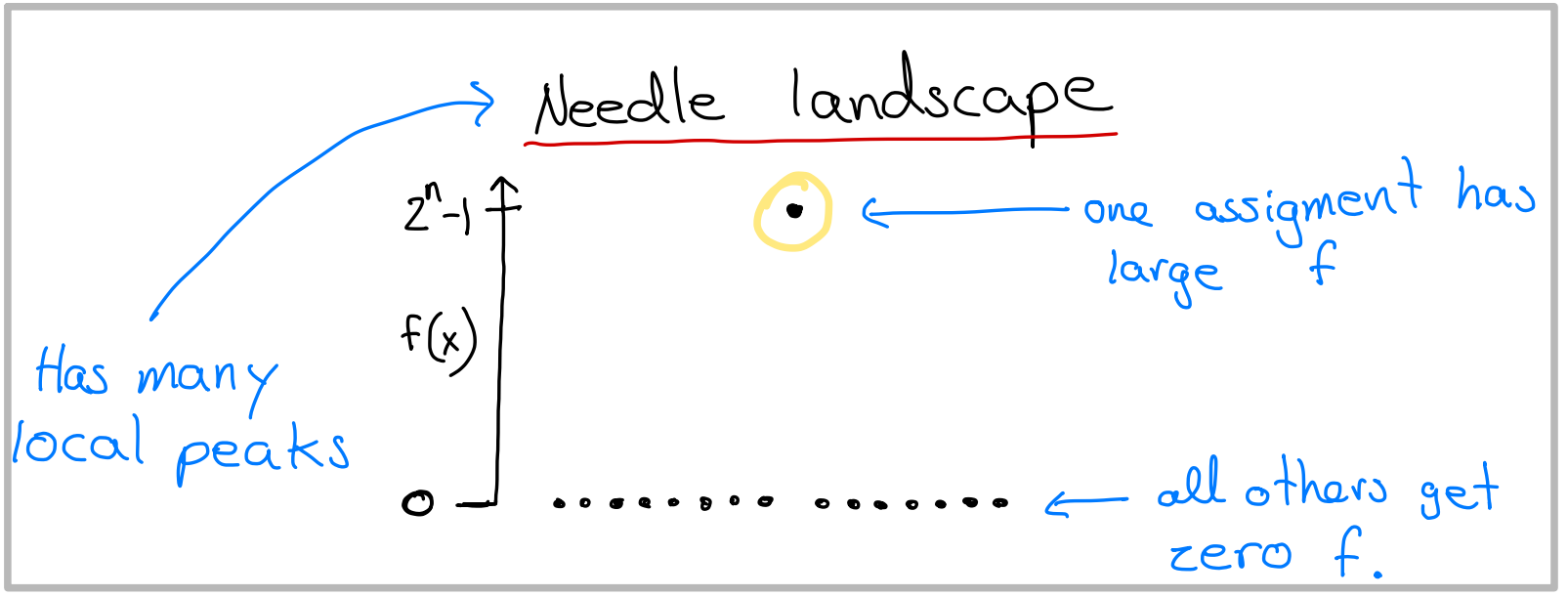
Did we find a solution quickly?

## Effective

Did we find a good solution?

## Efficient

Did we find a solution quickly?



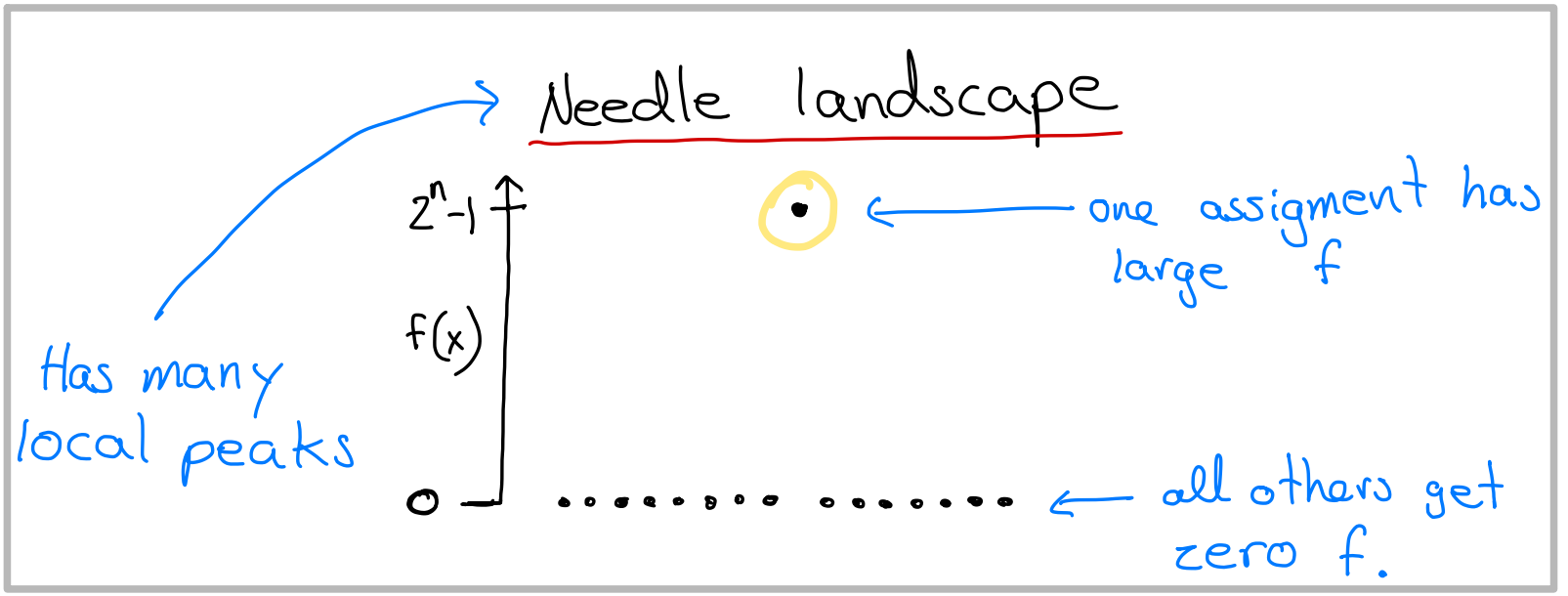
## Effective

Did we find a good solution?

## Efficient

Did we find a solution quickly?

## Needle landscape



☐ Effective

Did we find a good solution?

☐ Efficient

Did we find a solution quickly?

☑ Effective

Did we find a good solution?

☐ Efficient

Did we find a solution quickly?

single peaked landscapes

i.e. only one assignment is a local peak

Effective

Did we find a good solution?

Efficient

Did we find a solution quickly?

single peaked landscapes

☑ Effective

Did we find a good solution?

☒ Efficient

Did we find a solution quickly?

single peaked landscapes

Steepest Ascent

**Greed Is Slow on Sparse Graphs of Oriented Valued Constraints**

Artem Kaznatcheev
Department of Mathematics, and Department of Information and Computing Sciences, Utrecht University, The Netherlands
Sofia Vazquez Alferez
Department of Mathematics, and Department of Information and Computing Sciences, Utrecht University, The Netherlands

GENETICS | INVESTIGATION

**Computational Complexity as an Ultimate Constraint on Evolution**

Artem Kaznatcheev
Department of Computer Science, University of Oxford, Oxford, OX1 3QD, United Kingdom and Department of Translational Hematology and Oncology Research, Cleveland Clinic, Ohio 44195

(2019)

Discrete Applied Mathematics 23 (1989) 285–287
North-Holland


NOTE

**THE WORST CASE BEHAVIOR OF A GREEDY ALGORITHM FOR A CLASS OF PSEUDO-BOOLEAN FUNCTIONS**

M.R. EMAMY-K.
*Departmento de Matematicas, Universidad de Puerto Rico, Recinto de Rio Piedras, Apartado BF Rio Piedras, Puerto Rico 00931*

Received 28 July 1986
Revised 21 September 1988

© DISCRETE MATHEMATICS 4 (1973) 367–377. North-Holland Publishing Company

**THE SIMPLEX ALGORITHM WITH THE PIVOT RULE OF MAXIMIZING CRITERION IMPROVEMENT**

R.G. JEROSLOW
*Carnegie-Mellon University, Pittsburgh, USA*

Received 20 March 1972*

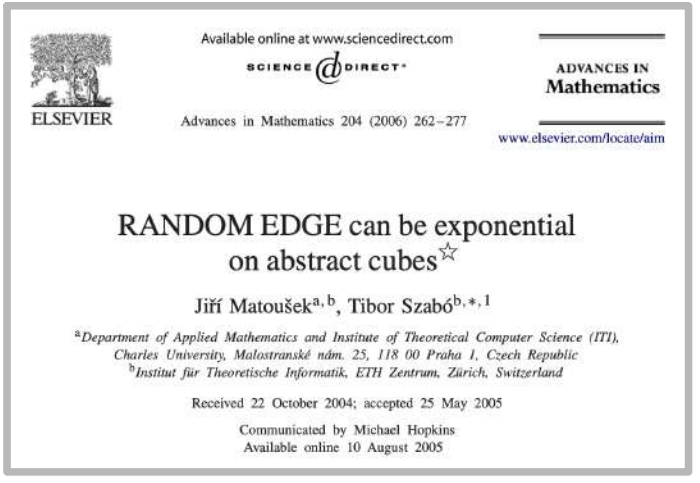
☑ Effective

Did we find a good solution?

☒ Efficient

Did we find a solution quickly?

single peaked landscapes



ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT®

ADVANCES IN Mathematics

Advances in Mathematics 204 (2006) 262–277

www.elsevier.com/locate/aim

RANDOM EDGE can be exponential on abstract cubes☆

Jiří Matoušek[a,b], Tibor Szabó[b,*,1]

[a]Department of Applied Mathematics and Institute of Theoretical Computer Science (ITI), Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic
[b]Institut für Theoretische Informatik, ETH Zentrum, Zürich, Switzerland

Received 22 October 2004; accepted 25 May 2005

Communicated by Michael Hopkins
Available online 10 August 2005

Steepest Ascent
Random Ascent

**Random-Edge Is Slower Than Random-Facet on Abstract Cubes**

Thomas Dueholm Hansen*[1] and Uri Zwick†[2]

1 Department of Computer Science, Aarhus University, Aarhus, Denmark
tdh@cs.au.dk
2 Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel
zwick@tau.ac.il

(2016)

☑ Effective

Did we find a good solution?

☒ Efficient

Did we find a solution quickly?

single peaked landscapes

The Random-Facet Simplex Algorithm on Combinatorial Cubes *

Bernd Gärtner, ETH Zürich, Switzerland

January 11, 2002

Steepest Ascent
Random Ascent
Random Facet

Effective

Did we find a good solution?

Efficient

Did we find a solution quickly?

single peaked landscapes

**Jumping Doesn't Help in Abstract Cubes**

Ingo Schurr* and Tibor Szabó**

Theoretical Computer Science, ETH Zürich,
CH-8092 Zürich, Switzerland
{schurr, szabo}@inf.ethz.ch

(2005)

Steepest Ascent
Random Ascent
Random Facet
Jumping rules

# Our Goal

Find an expressive class of problem instances where many local search algorithms are both effective and efficient.

can guarantee by single peak

# Our Goal

Find an expressive subclass of single peaked landscapes where many local search algorithms are efficient.

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

( Recall $x_i \in \{0,1\}$ and we are maximizing )

# Building up to our class

$$f(x_1 x_2 x_3) = 10 x_1 + 30 x_2 + 30 x_3$$

should we set $x_1 = 1$ ?

# Building up to our class

$$f(x_1 x_2 x_3) = 10 x_1 + 30 x_2 + 30 x_3$$

should we set $x_1 = 1$ ? YES

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

should we set $x_1 = 1$ ? YES

$x_2 = 1$ ? YES

$x_3 = 1$ ? YES

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

should we set $x_1 = 1$ ? YES

$x_2 = 1$ ? YES

$x_3 = 1$ ? YES

unique peak: 111

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

setting any variable to 1 improves $f$ independent of the assignment to other variables

# Building up to our class

$$f(x_1 x_2 x_3) = 10 x_1 + 30 x_2 + 30 x_3$$

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

Should we set $x_1 = 1$ ?

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

Should we set $x_1 = 1$ ?

$x_2 = 1$ ?

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

Should we set $x_1 = 1$ ?

$x_2 = 1$ ?

$x_3 = 1$ ? YES !

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

$$\hookrightarrow f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$$

should we set $x_1 = 1$? YES

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

↳ $f(1 x_2 1) = -10x_2 + 60$

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

↳ $f(1 x_2 1) = -10x_2 + 60$ ← Easy again!

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

↳ $f(1 x_2 1) = -10x_2 + 60$ ← Easy again!

Should we set $x_2 = 1$? NO

# Building up to our class

$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$

$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

↳ $f(1 x_2 1) = -10x_2 + 60$

Optimizing is easy if we first ask about $x_3$, then ask about $x_1$, and then ask about $x_2$

# Building up to our class

$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$

$x_3$ unconditionally prefers 1

$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$

$x_1$ prefers 1 conditioned on $x_3 = 1$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

$x_2$ prefers 0 conditioned on $x_1 = 1$ and $x_3 = 1$

↳ $f(1 x_2 1) = -10x_2 + 60$

Optimizing is easy if we first ask about $x_3$, then ask about $x_1$, and then ask about $x_2$

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

↳ $f(1 x_2 1) = -10x_2 + 60$

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

These easy landscapes where the "preferred" assignment of a variable is independent of the assignment to any other variable are called smooth fitness landscapes

# Building up to our class

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3$$

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

↳ $f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$

↳ $f(1 x_2 1) = -10x_2 + 60$

# Building up to our class

We will call these conditionally-smooth

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

$$\hookrightarrow f(x_1 x_2 1) = 30x_1 + 10x_2 - 20x_1x_2 + 30$$

$$\hookrightarrow f(1 x_2 1) = -10x_2 + 60$$

# Conditionally smooth landscapes

**Definition 8.**

A fitness landscape is conditionally-smooth if there exist an ordering* of the variables such that the preferred assignment of a variable is independent of variables later in the ordering conditioned on all earlier variables being set to their assignment at the peak.

(* can be a partial order)

# Conditionally-smooth landscapes are expressive

Can express instances where some popular local search algorithms are inefficient

# Conditionally-smooth landscapes are expressive

Can express instances where some popular local search algorithms are inefficient

*Complex Systems* **2** (1988) 191–196

**Steepest Descent Can Take Exponential Time for Symmetric Connection Networks***

**Armin Haken**
**Michael Luby**
*Department of Computer Science, University of Toronto,*
*10 King's College Road, Toronto, M5S 1A4, Canada*

# Conditionally-smooth landscapes are expressive

Can express instances where some popular local search algorithms are inefficient

**Theorem 19.** There exist conditionally-smooth fitness landscapes where greedy local search takes an exponential number of steps to the peak.

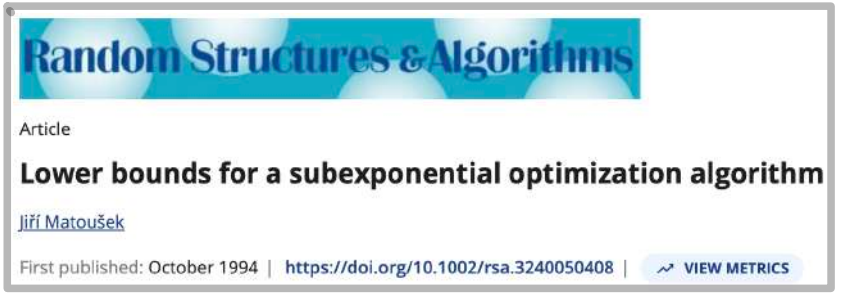
# Conditionally-smooth landscapes are expressive

Can express instances where some popular local search algorithms are inefficient

The Random-Facet Simplex Algorithm on Combinatorial Cubes *

Bernd Gärtner, ETH Zürich, Switzerland

January 11, 2002



# Conditionally-smooth landscapes are expressive

Can express instances where some popular local search algorithms are inefficient

Theorem 20. There exist conditionally smooth fitness landscapes where RandomFacet takes an expected $e^{\Theta(\sqrt{n})}$ steps to the peak.

# Conditionally-smooth landscapes are expressive

Both Steepest Ascent and Random Facet lack a property that many other local search algorithms have ...

# Poly - Bypass

**Definition 10 (intuition):**

An algorithm $A$ is a poly-bypass algorithm if no improving flip of a variable is by-passed by $A$ for more than a polynomial number of steps.

# Intuition for poly-bypass

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

# Intuition for poly-bypass

$$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

| f(x) | 0 | 10 | 20 | 30 | 40 |
|---|---|---|---|---|---|
| x | 000 | 100 | 110 | 010 | 011 |

# Intuition for poly-bypass

$$f(x_1\, x_2\, x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$$

recall $x_3$ unconditionally prefers 1.

| $f(x)$ | 0 | 10 | 20 | 30 | 40 |
|---|---|---|---|---|---|
| $x$ | 000 → | 100 → | 110 → | 010 → | 011 |

# Intuition for poly-bypass

$f(x_1 x_2 x_3) = 10x_1 + 30x_2 + 30x_3 - 20x_1x_2 + 20x_1x_3 - 20x_2x_3$

recall $x_3$ unconditionally prefers 1.

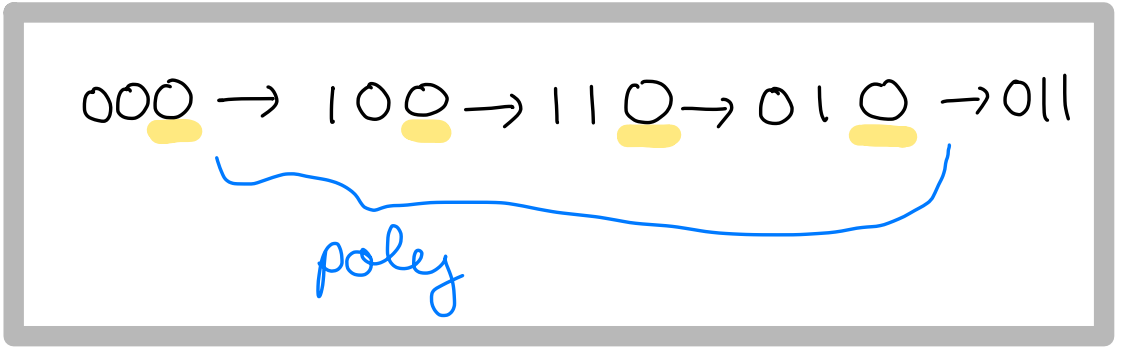
| $f(x)$ | 0 | 10 | 20 | 30 | 40 |
|---|---|---|---|---|---|
| $x$ | 000 → | 100 → | 110 → | 010 → | 011 |

In all these steps the improving flip of $x_3$ is bypassed

# Poly - Bypass

## Definition 10 (intuition):

An algorithm $A$ is a poly-bypass algorithm if no improving flip of a variable is by-passed by $A$ for more than a polynomial number of steps.



# Poly - Bypass

Theorem 11* On a conditionally-smooth landscape on $n$ bits a poly-bypass algorithm takes at most $n \cdot \text{poly}(n)$ steps to find the peak.

# Poly - Bypass

**Theorem 11*** On a conditionally-smooth landscape on $n$ bits a poly-bypass algorithm takes at most $n \cdot \text{poly}(n)$ steps to find the peak.

**Proof (sketch):** Suppose w.l.o.g that the natural order $(<)$ certifies conditional smoothness.

Let the current assignment $x^{\text{current}}$ agree with $x^*$ on the first $m$ bits. $(m < n)$.

In at most poly$(n)$ steps $x^{\text{current}}$ agrees with $x^*$ on at least the first $m+1$ bits.

Two assignments differ in at most $n$ bits. □

# Poly-Bypass

**Theorem 11*** On a conditionally-smooth landscape on $n$ bits a poly-bypass algorithm takes at most $n \cdot \text{poly}(n)$ steps to find the peak.

Many popular local search algorithms are poly-bypass!

# Poly - Bypass

Theorem 11* On a conditionally-smooth landscape on $n$ bits a poly-bypass algorithm takes at most $n \cdot \text{poly}(n)$ steps to find the peak.

This bound is loose and we can tighten it

Many popular local search algorithms are poly-bypass!

# Poly - Bypass

Theorem 11* On a conditionally-smooth landscape on $n$ bits a poly-bypass algorithm takes at most $n \cdot \text{poly}(n)$ steps to find the peak.

▶ **Lemma 13.** *Given a $\prec$-smooth fitness landscape $f$ on $n$ bits and any assignment $x$ with $l := height_f(x)$, let $Y : \Omega \times \mathbb{N} \to \{0,1\}^n$ be a stochastic process such that $Y_t \sim \mathsf{A}_f^t(x)$ is the random variable of outcomes of $t$ applications of the local search step algorithm $\mathsf{A}_f^1$ starting from $x$ and let the random variable $\tau_{<l}(\omega) := \inf\{t \mid \overline{\phi^{\ominus}(Y_t(\omega))} \in S_{<l}\}$ be the number of steps to decrement height. If the expected number of steps to decrement height is:*

$$\mathbb{E}\{\tau_{<l}(\omega)\} \leq p(n,l) \leq q(n) \tag{2}$$

*then the expected total number of steps taken by $\mathsf{A}$ to find the peak from an initial assignment $x^0$ is $\leq \sum_{l=1}^{height_f(x^0)} p(n,l) \leq height_f(x_0)q(n) \leq height(\prec)q(n)$.*

# Poly - Bypass

Proposition 14. On a conditionally-smooth landscapes on $n$ bits, the expected number of steps of Random Ascent is

$$\leq n + \text{width}(\prec)\left(1+\log\left(\text{width}(\prec)\right)\right)\binom{\text{height}(\prec)-1}{2}$$

the partial order certifying conditional-smoothness

# Poly - Bypass

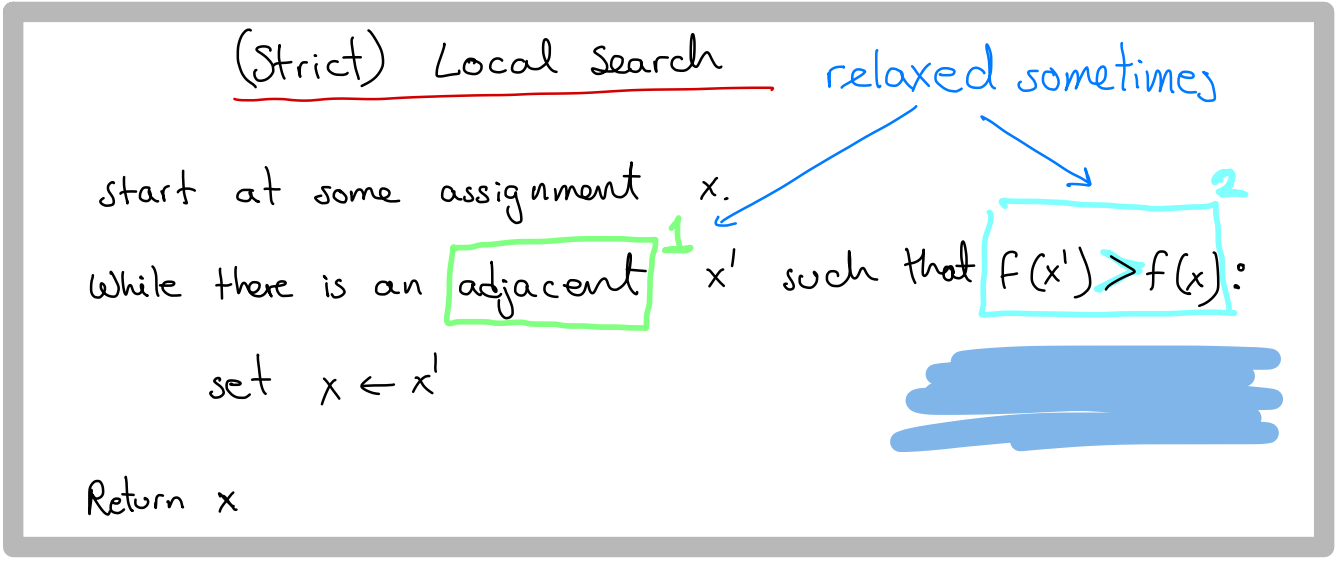
Proposition 15. On a conditionally-smooth landscapes on $n$ bits, the expected number of steps of Simulated Annealing is

$$\leq \tau^{\alpha} + n^2 \frac{(e^{\alpha} - 1)}{\alpha}$$

some constants specific to Simulated Annealing

# Poly - Bypass

Proposition 15. On a conditionally-smooth landscapes on n bits, the expected number of steps of Simulated Annealing is



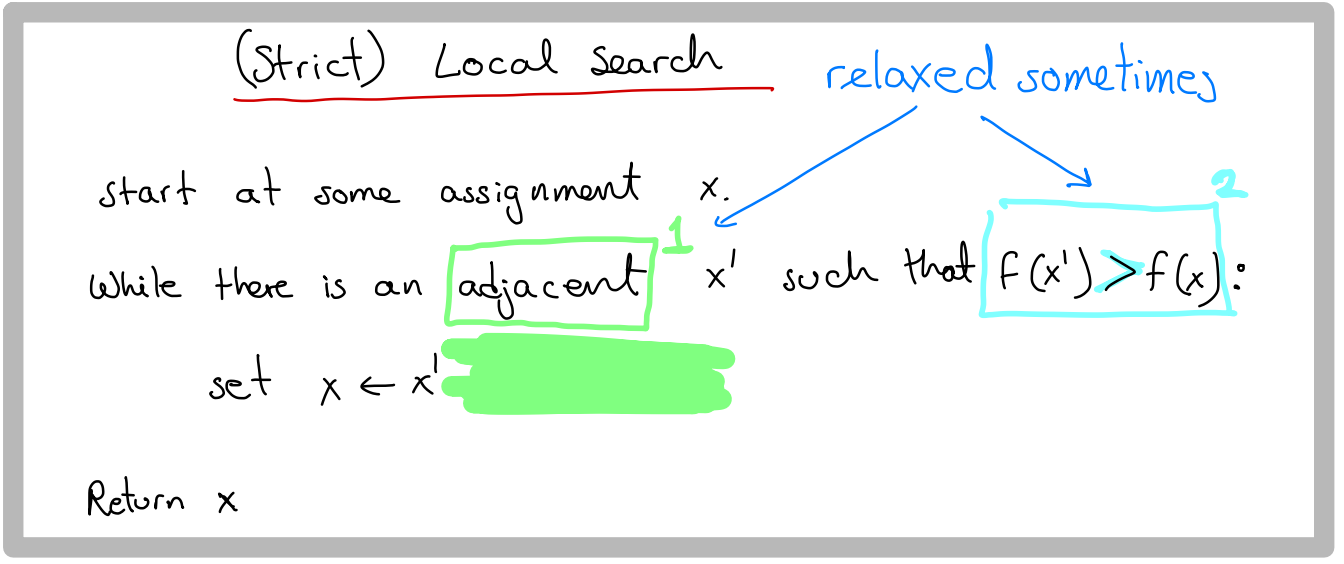
# Poly - Bypass

Proposition 16. On a conditionally-smooth landscapes on $n$ bits, the expected number of steps of RandomJump is

$$\leq \log(\text{width}(\mathcal{L})+2)\ \text{height}(\mathcal{L})$$

# Poly - Bypass

Proposition 16. On a conditionally-smooth landscapes on $n$ bits, the expected number of steps of RandomJump is



# Conclusion

Conditionally-smooth fitness landscapes are an expressive class of problem instances on which many (but not all) local search algorithms are effective an efficient.

# Conclusion

Conditionally-smooth fitness landscapes are an expressive class of problem instances on which many (but not all) local search algorithms are effective an efficient.

## Not covered today

Connection to constraint graphs of valued CSPs that are "directed and acyclic"
(Sections 4 & 5)

# Future work

1) Generalize conditionally-smooth from Boolean to higher-valence domains.
2) Are block-wise conditionally-smooth landscapes still efficient for many local search algorithms?
   2a) FPT parameterized by size of largest block?

# THANKS!

Artem Kaznatcheev
a.kaznatcheev@uu.nl

& Sofia (Vazquez Alferez)
s.vazquezalferez@uu.nl

Utrecht University