Theory of Randomized Search Heuristics in Combinatorial Optimization

Witt, Carsten

Link to article, DOI:
10.1145/2001858.2002135

Publication date:
2011

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
Evolutionary Algorithms and Other Search Heuristics

Most famous search heuristic: Evolutionary Algorithms (EAs)

- a bio-inspired heuristic
- paradigm: evolution in nature, “survival of the fittest”
- actually it’s only an algorithm, a randomized search heuristic (RSH)

Goal: optimization
- Here: discrete search spaces, combinatorial optimization, in particular pseudo-boolean functions

Optimize $f : \{0,1\}^n \rightarrow \mathbb{R}$
Why Do We Consider Randomized Search Heuristics?

- Not enough resources (time, money, knowledge) for a tailored algorithm
- Black Box Scenario rules out problem-specific algorithms
- We like the simplicity, robustness, ... of Randomized Search Heuristics
- They are surprisingly successful.

Point of view

Do not only consider RSHs empirically. We need a solid theory to understand how (and when) they work.

What RSHs Do We Consider?

Theoretically considered RSHs

- (1+1) EA
- (1+\lambda) EA (offspring population)
- (\mu+1) EA (parent population)
- (\mu+1) GA (parent population and crossover)
- GIGA (crossover)
- SEMO, DEMO, FEMO, ... (multi-objective)
- Randomized Local Search (RLS)
- Metropolis Algorithm/Simulated Annealing (MA/SA)
- Ant Colony Optimization (ACO)
- Particle Swarm Optimization (PSO)
- ...

First of all: define the simple ones

The Most Basic RSHs

(1+1) EA, RLS, MA and SA for maximization problems

(1+1) EA

Choose \(x_0 \in \{0,1\}^n\) uniformly at random.

For \(t := 0, \ldots, \infty\)

- Create \(y\) by flipping each bit of \(x_t\) indep. with probab. \(1/n\).
- If \(f(y) \geq f(x_t)\) set \(x_{t+1} := y\) else \(x_{t+1} := x_t\).
The Most Basic RSHs

\((1+1)\) EA, RLS, MA and SA for maximization problems

**RLS**
- Choose \(x_0 \in \{0,1\}^n\) uniformly at random.
- For \(t := 0, \ldots, \infty\):
  - Create \(y\) by flipping one bit of \(x_t\) uniformly.
  - If \(f(y) \geq f(x_t)\) set \(x_{t+1} := y\) else \(x_{t+1} := x_t\).

**MA**
- Choose \(x_0 \in \{0,1\}^n\) uniformly at random.
- For \(t := 0, \ldots, \infty\):
  - Create \(y\) by flipping one bit of \(x_t\) uniformly.
  - If \(f(y) \geq f(x_t)\) set \(x_{t+1} := y\) with probability \(e^{f(x_t) - f(y)} / T\) anyway and \(x_{t+1} := x_t\) otherwise.

\(T\) is fixed over all iterations.

**SA**
- Choose \(x_0 \in \{0,1\}^n\) uniformly at random.
- For \(t := 0, \ldots, \infty\):
  - Create \(y\) by flipping one bit of \(x_t\) uniformly.
  - If \(f(y) \geq f(x_t)\) set \(x_{t+1} := y\) with probability \(e^{f(x_t) - f(y)} / T_t\) anyway and \(x_{t+1} := x_t\) otherwise.

\(T_t\) is dependent on \(t\), typically decreasing.

What Kind of Theory Are We Interested in?
- Not studied here: convergence, local progress, models of EAs (e.g., infinite populations), . . .
- Treat RSHs as randomized algorithm!
- Analyze their “runtime” (computational complexity) on selected problems
What Kind of Theory Are We Interested in?

- Not studied here: convergence, local progress, models of EAs (e.g., infinite populations), ...
- Treat RSHs as randomized algorithm!
- Analyze their "runtime" (computational complexity) on selected problems

Definition

Let RSH $A$ optimize $f$. Each $f$-evaluation is counted as a time step. The runtime $T_{A,f}$ of $A$ is the random first point of time such that $A$ has sampled an optimal search point.

- Often considered: expected runtime, distribution of $T_{A,f}$
- Asymptotical results w.r.t. $n$

How Do We Obtain Results?

We use (rarely in their pure form):
- Coupon Collector’s Theorem
- Principle of Deferred Decisions
- Concentration inequalities:
  - Markov, Chebyshev, Chernoff, Hoeffding, ... bounds
- Markov chain theory: waiting times, first hitting times
- Rapidly Mixing Markov Chains
- Random Walks: Gambler’s Ruin, drift analysis (Wald’s equation), martingale theory, electrical networks
- Random graphs (esp. random trees)
- Identifying typical events and failure events
- Potential functions and amortized analysis
  - ...

Adapt tools from the analysis of randomized algorithms; understanding the stochastic process is often the hardest task.

Early Results

Analysis of RSHs already in the 1980s:
- Sasaki/Hajek (1988): SA and Maximum Matchings
- Sorkin (1991): SA vs. MA
- Jerrum (1992): SA and Cliques
  - ...

These were high-quality results, however, limited to SA/MA (nothing about EAs) and hard to generalize.
Early Results

Analysis of RSHs already in the 1980s:
- Sasaki/Hajek (1988): SA and Maximum Matchings
- Sorkin (1991): SA vs. MA
- Jerrum (1992): SA and Cliques

These were high-quality results, however, limited to SA/MA (nothing about EAs) and hard to generalize.

Since the early 1990s
Systematic approach for the analysis of RSHs, building up a completely new research area

How the Systematic Research Began — Toy Problems

Simple example functions (test functions)
- OneMax($x_1, \ldots, x_n$) = $x_1 + \cdots + x_n$
- LeadingOnes($x_1, \ldots, x_n$) = $\sum_{i=1}^{n} \prod_{j=1}^{i} x_j$
- BinVal($x_1, \ldots, x_n$) = $\sum_{i=1}^{n} 2^{n-i} x_i$
- polynomials of fixed degree

Goal: derive first runtime bounds and methods

This Tutorial

1. The origins: example functions and toy problems
   - A simple toy problem: OneMax for (1+1) EA

2. Combinatorial optimization problems
   - (1+1) EA and minimum spanning trees
   - (1+1) EA and Eulerian cycles
   - (1+1) EA and maximum matchings
   - (1+1) EA and the partition problem
   - SA beats MA in combinatorial optimization

3. End
Agenda

1. The origins: example functions and toy problems
   - A simple toy problem: OneMax for (1+1) EA

2. Combinatorial optimization problems
   - (1+1) EA and minimum spanning trees
   - (1+1) EA and Eulerian cycles
   - (1+1) EA and maximum matchings
   - (1+1) EA and the partition problem
   - SA beats MA in combinatorial optimization

3. End

Example: OneMax

Theorem (e.g., Droste/Jansen/Wegener, 1998)

The expected runtime of the RLS, (1+1) EA, (μ+1) EA, (1+λ) EA on OneMax is \( \Omega(n \log n) \).

Proof by modifications of Coupon Collector’s Theorem.

Example: OneMax

Fitness levels: \( L_i := \{ x \in \{0,1\}^n \mid \text{OneMax}(x) = i \} \)

Proof of the \( O(n \log n) \) bound

Theorem (e.g., Mühlenbein, 1992)

The expected runtime of RLS and the (1+1) EA on OneMax is \( O(n \log n) \).

Holds also for population-based (μ+1) EA and for (1+λ) EA with small populations.
Proof of the $O(n \log n)$ bound

- **Fitness levels:** $L_i := \{x \in \{0, 1\}^n \mid \text{OneMax}(x) = i\}$
- $(1+1)$ EA never decreases its current fitness level.

(1+1) EA never decreases its current fitness level. From $i$ to some higher-level set with prob. at least

$$\left(\frac{n-i}{n}\right) \cdot \left(\frac{1}{n}\right) \cdot \left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{n-i}{en}$$

choose a 0-bit, flip this bit, keep the other bits

Expected time to reach a higher-level set is at most $\frac{en}{n-i}$.

Expected runtime is at most

$$\sum_{i=0}^{n-1} \frac{en}{n-i} = O(n \log n). \quad \Box$$

Later Results Using Toy Problems

- Find the theoretically optimal mutation strength (1/$n$ for OneMax!).
- Bound the optimization time for linear functions ($O(n \log n)$).
- Optimal population size (often 1!)
- Crossover vs. no crossover → Real Royal Road Functions
- Multistarts vs. populations
- Frequent restarts vs. long runs
- Dynamic schedules
- ...
RSHs for Combinatorial Optimization

- Analysis of runtime and approximation quality on well-known combinatorial optimization problems, e.g.,
  - sorting problems (is this an optimization problem?),
  - covering problems,
  - cutting problems,
  - subsequence problems,
  - traveling salesperson problem,
  - Eulerian cycles,
  - minimum spanning trees,
  - maximum matchings,
  - scheduling problems,
  - shortest paths,
  - ...

- What we do not hope: to be better than the best problem-specific algorithms

In the following no fine-tuning of the results
More details in the books (last slide)

Agenda

1. The origins: example functions and toy problems
   - A simple toy problem: OneMax for (1+1) EA

2. Combinatorial optimization problems
   - (1+1) EA and minimum spanning trees
   - (1+1) EA and Eulerian cycles
   - (1+1) EA and maximum matchings
   - (1+1) EA and the partition problem
   - SA beats MA in combinatorial optimization

End

Minimum Spanning Trees

**Problem**

**Given:** Undirected connected graph $G = (V, E)$ with $n$ vertices and $m$ edges with positive integer weights.

**Find:** Edge set $E' \subseteq E$ with minimal weight connecting all vertices.
Minimum Spanning Trees

Problem
Given: Undirected connected graph \( G = (V, E) \) with \( n \) vertices and \( m \) edges with positive integer weights.
Find: Edge set \( E' \subseteq E \) with minimal weight connecting all vertices.

Fitness function
Decrease number of connected components, find minimum spanning tree:
\[
f(s) := (c(s), w(s)).
\]
Minimization of \( f \) with respect to the lexicographic order.

Combinatorial Argument to Approach MSTs
From arbitrary spanning tree \( T \) to MST \( T^* \) (Mayr/Plaxton, 1992):

- \( k := |E(T^*) \setminus E(T)| \)
- Bijection \( \alpha : E(T^*) \setminus E(T) \rightarrow E(T) \setminus E(T^*) \)
- \( \alpha(e_i) \) on the cycle of \( E(T) \cup \{e_i\} \)
- \( w(e_i) \leq w(\alpha(e_i)) \)

\[ k \implies k \text{ accepted 2-bit flips that turn } T \text{ into } T^* \]
Upper Bound

Theorem (Neumann/Wegener, 2007)

The expected time until $(1+1)$ EA constructs a minimum spanning tree is bounded by $O(m^2(\log n + \log w_{\text{max}}))$.

Sketch of proof:
- $w(s)$ weight current solution $s$; assume to be tree
- $w_{\text{opt}}$ weight minimum spanning tree $T^*$

Concentrate on 2-bit flips:
- Expected weight decrease by a factor $1 - 1/n$ (or better)
- Probability $\Theta(n/m^2)$ for a good 2-bit flip
- Expected time until $r$ 2-steps $O(rm^2/n)$

Concentrate on 2-bit flips:
- Expected weight decrease by a factor $1 - 1/n$ (or better)
- Probability $\Theta(n/m^2)$ for a good 2-bit flip
- Expected time until $r$ 2-steps $O(rm^2/n)$

Method expected multiplicative distance decrease:
- Have to bridge distance at most $D := w(s) - w_{\text{opt}} \leq m \cdot w_{\text{max}}$
- Distance after $N$ steps: $\leq (1 - 1/n)^N \cdot D$
- Find $N$ such that $(1 - 1/n)^N \leq 1/(2D)$
  ⇒ choose $N := \lceil n \cdot (\ln D + 1) \rceil$
- In expectation $2N = O(n(\log n + \log w_{\text{max}}))$ 2-steps enough
- Expected time: $O(Nm^2/n) = O(m^2(\log n + \log w_{\text{max}}))$
Further Results

Lower Bound $\Omega(n^4 \log n)$

Related Results
- Experimental investigations (Briest et al., 2004)
- Biased mutation operators (Raidl/Koller/Julstrom, 2006)
- $O(mn^2)$ for a multi-objective approach (Neumann/Wegener, 2006)
- Approximations for multi-objective minimum spanning trees (Neumann, 2007)
- SA/MA and minimum spanning trees (Later!)

Agenda

1. The origins: example functions and toy problems
   - A simple toy problem: OneMax for (1+1) EA
2. Combinatorial optimization problems
   - (1+1) EA and minimum spanning trees
   - (1+1) EA and Eulerian cycles
   - (1+1) EA and maximum matchings
   - (1+1) EA and the partition problem
   - SA beats MA in combinatorial optimization
3. End

Eulerian Cycle Problem

Given: undirected connected Eulerian (degree of each vertex is even) graph $G = (V, E)$ with $n$ vertices and $m$ edges

Find: a cycle (permutation of the edges) such that each edge is used exactly once.
Eulerian Cycle Problem

**Given:** undirected connected Eulerian (degree of each vertex is even) graph \( G = (V, E) \) with \( n \) vertices and \( m \) edges

**Find:** a cycle (permutation of the edges) such that each edge is used exactly once.

Eulerian Cycle (Hierholzer)

Idea: “glue” small cycles together

1. Find a cycle \( C \) in \( G \).
2. Delete the edges of \( C \) from \( G \).
3. If \( G \) is not empty go to step 1; starting from a vertex on \( C \).
4. Construct the Eulerian cycle by running through the cycles produced in Step 1 in the order of construction.

Fitness Function

**Representation:** permutation of edges

**Fitness function**

Consider the edges of the permutation after another and build up a path \( p \) of length \( l \).

\[
\text{path}(\pi) := \text{length of the path} \ p \ \text{implied by} \ \pi
\]

Example: \( \pi = (\{2, 3\}, \{1, 2\}, \{1, 5\}, \{3, 4\}, \{4, 5\}) \implies |p| = 3 \)

The (1+1) EA for the Euler Cycle Problem

**(1+1) EA**

- Choose \( \pi \in S_m \) uniform at random.
- Choose \( s \) from a Poisson distribution with parameter 1. Perform sequentially \( s + 1 \) jump operations to produce \( \pi' \) from \( \pi \).
The (1+1) EA for the Euler Cycle Problem

(1+1) EA

1. Choose \( \pi \in S_m \) uniform at random.
2. Choose \( s \) from a Poisson distribution with parameter 1. Perform sequentially \( s + 1 \) jump operations to produce \( \pi' \) from \( \pi \).

Example: \( \text{jump}(2,4) \) applied to \( \{(2,3),(1,2),(3,4),(1,5),(4,5)\} \) produces \( \{(2,3),(3,4),(1,5),(1,2),(4,5)\} \)

3. Replace \( \pi \) by \( \pi' \) if path(\( \pi' \)) \( \geq \) path(\( \pi \)).
4. Repeat Steps 2 and 3 forever.

Upper Bound, (1+1) EA

Theorem (Neumann, 2007)
The expected time until (1+1) EA working on the fitness function path constructs an Eulerian cycle is bounded by \( O(m^5) \).

Proof idea:

- \( p \) is not a cycle:
  - 1 improving jump \( \Rightarrow \) expected time for improvement \( O(m^2) \)
- \( p \) is a cycle (with less than \( m \) edges):
  - Show: expected time for an improvement \( O(m^4) \)
- \( O(m) \) improvements \( \Rightarrow \) theorem

Proof Idea: How to Analyze Improvements

Typical run:

- \( k \)-step (accepted mutation with \( k \)-jumps that change \( p \)):
- Only 1-steps: \( O(m^4) \) steps for an improvement
- No \( k \)-step, \( k \geq 4 \), in \( O(m^4) \) steps with prob. \( 1 - o(1) \)
- \( O(1) \) 2- or 3-steps in \( O(m^4) \) steps with prob. \( 1 - o(1) \)
Proof Idea: How to Shift a Cycle

Time $O(m^2)$ to move black vertex
Black vertex performs random walk
Length of cycle at most $m$
Fair random walk
→ $O(m^2)$ movements are enough to reach red vertex
Expected time for an improvement $O(m^4)$

Further Results

Lower bound $\Omega(m^4)$
Restricted jumps (always jump to position 1)
- No random walk, but directed walk
  - Upper bound $O(m^3)$ (Doerr/Hebbinghaus/Neumann, 2007)
Further Results

- Lower bound $\Omega(m^4)$
- Restricted jumps (always jump to position 1)
  - No random walk, but directed walk
  - Upper bound $O(m^3)$ (Doerr/Hebbinghaus/Neumann, 2007)
- Use of more sophisticated representations and mutation operators:
  - $O(m^2 \log m)$ (Doerr/Klein/Storch, 2007)
  - $O(m \log m)$ (Doerr/Johannsen, 2007)

Agenda

1. The origins: example functions and toy problems
   - A simple toy problem: OneMax for (1+1) EA
2. Combinatorial optimization problems
   - (1+1) EA and minimum spanning trees
   - (1+1) EA and Eulerian cycles
   - (1+1) EA and maximum matchings
   - (1+1) EA and the partition problem
   - SA beats MA in combinatorial optimization
3. End

(1+1) EA for the Maximum Matching Problem
The Behavior on Paths

A matching in a graph is a subset of pairwise disjoint edges.
Path: $n + 1$ nodes, $n$ edges: bit string from $\{0, 1\}^n$ selects edges
Fitness function: size of matching/negative for non-matchings

Theorem (Giel/Wegener, 2003)
The expected time until the (1+1) EA finds a maximum matching on a path of $n$ edges is $O(n^4)$. 
Proof idea:
- Consider a second-best matching.
- Is there a free edge? Flip one bit! → probability $\Theta(1/n)$.
- Else 2-bit flips → probability $\Theta(1/n^2)$.

Proof idea:
- Consider a second-best matching.
- Is there a free edge? Flip one bit! → probability $\Theta(1/n)$.
- Else 2-bit flips → probability $\Theta(1/n^2)$.
- Shorten augmenting path
(1+1) EA for the Maximum Matching Problem
The Behavior on Paths (2)

Proof idea:
- Consider a second-best matching.
- Is there a free edge? Flip one bit! → probability $\Theta(1/n)$.
- Else 2-bit flips → probability $\Theta(1/n^2)$.
- Shorten augmenting path
- Then flip the free edge!
Proof idea:
- Consider a second-best matching.
- Is there a free edge? Flip one bit! → probability Θ(1/n).
- Else 2-bit flips → probability Θ(1/n²).
- Shorten augmenting path
- Then flip the free edge!
- Shorten augmenting path
- Then flip the free edge!

(1+1) EA follows the concept of an augmenting path!

Length changes according to a fair random walk
→ Expected runtime $O(n^2) \cdot O(n^2) = O(n^4)$. 
(1+1) EA for the Maximum Matching Problem
A Negative Result

Worst-case graph $G_{h,\ell}$ (Sasaki/Hajek, 1988)

$h \geq 3$

$\ell = 2\ell' + 1$

Augmenting path can get shorter but is more likely to get longer.

Theorem

For $h \geq 3$, the (1+1) EA has exponential expected runtime $2^{\Omega(\ell)}$ on $G_{h,\ell}$.

Proof by drift analysis
(1+1) EA for the Maximum Matching Problem

(1+1) EA is a PRAS

**Insight:** do not hope for exact solutions but for approximations

**Theorem (Giel/Wegener, 2003)**

For $\varepsilon > 0$, the (1+1) EA finds a $(1 + \varepsilon)$-approximation of a maximum matching in expected time $O(m^2/[1/\varepsilon])$ and is a polynomial-time randomized approximation scheme (PRAS).

**Proof idea:**

- Look into the analysis of the Hopcroft/Karp algorithm.
- Current solution worse than $(1 + \varepsilon)$-approximate $\rightarrow$ many augmenting paths, in partic. a short one of length $\leq 2[\varepsilon^{-1}]$
- Wait for the (1+1) EA to optimize this short path.

---

**Agenda**

1. The origins: example functions and toy problems
   - A simple toy problem: OneMax for (1+1) EA
2. Combinatorial optimization problems
   - (1+1) EA and minimum spanning trees
   - (1+1) EA and Eulerian cycles
   - (1+1) EA and maximum matchings
   - (1+1) EA and the partition problem
   - SA beats MA in combinatorial optimization
3. End

---

**(1+1) EA for the Maximum Matching Problem**

(1+1) EA is a PRAS

**Insight:** do not hope for exact solutions but for approximations

**Theorem (Giel/Wegener, 2003)**

For $\varepsilon > 0$, the (1+1) EA finds a $(1 + \varepsilon)$-approximation of a maximum matching in expected time $O(m^2/[1/\varepsilon])$ and is a polynomial-time randomized approximation scheme (PRAS).
What about NP-hard problems? → Study approximation quality

For $w_1, \ldots, w_n$, find $I \subseteq \{1, \ldots, n\}$ minimizing

$$\max \left\{ \sum_{i \in I} w_i, \sum_{i \notin I} w_i \right\}.$$ 

This is an “easy” NP-hard problem:
- not strongly NP-hard,
- FPTAS exist,
- ...

Theorem (Witt, 2005)

On any instance for the partition problem, the (1+1) EA reaches a solution with approximation ratio $4/3$ in expected time $O(n^2)$. 

Theorem

There is an instance such that the (1+1) EA needs with prob. $\Omega(1)$ at least $n^{\Omega(n)}$ steps to find a solution with a better ratio than $4/3 - \varepsilon$.

Proof ideas: study effect of local steps and local optima
Theorem

On any instance, the (1+1) EA with prob. \( \geq 2^{-c\frac{1}{\varepsilon}} \ln(1/\varepsilon) \) finds a \((1 + \varepsilon)\)-approximation within \(O(n \ln(1/\varepsilon))\) steps.

Set \( s := \lceil \frac{2}{\varepsilon} \rceil \) and \( w := \sum_{i=1}^{n} w_i \).
Assuming \( w_1 \geq \cdots \geq w_n \), we have \( w_i \leq \varepsilon \frac{w}{2} \) for \( i \geq s \).

Analyze probability of distributing
- large objects in an optimal way,
- small objects greedily \( \Rightarrow \) additive error \( \leq \varepsilon w/2 \).
This is the algorithmic idea by Graham (1969).
(1+1) EA for the Partition Problem
Average-Case Analyses

Models: each weight drawn independently at random, namely

1. uniformly from the interval $[0, 1]$,
2. exponentially distributed with parameter 1 (i.e., \( \text{Prob}(X \geq t) = e^{-t} \) for \( t \geq 0 \)).

Approximation ratio no longer meaningful, we investigate: discrepancy = absolute difference between weights of bins.

How close to discrepancy 0 do we come?

Deterministic, problem-specific heuristic LPT
Sort weights decreasingly, put every object into currently emptier bin.

Analysis in both random models:
After LPT has been run, additive error is \( O((\log n)/n) \) (Frenk/Rinnooy Kan, 1986).

Can RLS or the (1+1) EA reach a discrepancy of \( o(1) \)?
(1+1) EA for the Partition Problem

New Result

Theorem

In both models, the (1+1) EA reaches discrepancy $O((\log n)/n)$ after $O(n^{1+c} \log^2 n)$ steps with probability $1 - O(1/n^c)$. Almost the same result as for LPT!

Proof exploits order statistics:

\[
W. h. p. \quad X(i) - X(i+1) = O((\log n)/n) \quad \text{for } i = \Omega(n).
\]

Simulated Annealing Beats Metropolis in Combinatorial Optimization

Jerrum/Sinclair (1996)

"It remains an outstanding open problem to exhibit a natural example in which simulated annealing with any non-trivial cooling schedule provably outperforms the Metropolis algorithm at a carefully chosen fixed value" of the temperature.

Agenda

1. The origins: example functions and toy problems
   • A simple toy problem: OneMax for (1+1) EA
2. Combinatorial optimization problems
   • (1+1) EA and minimum spanning trees
   • (1+1) EA and Eulerian cycles
   • (1+1) EA and maximum matchings
   • (1+1) EA and the partition problem
   • SA beats MA in combinatorial optimization
3. End
Simulated Annealing Beats Metropolis in Combinatorial Optimization

Jerrum/Sinclair (1996)

“It remains an outstanding open problem to exhibit a natural example in which simulated annealing with any non-trivial cooling schedule provably outperforms the Metropolis algorithm at a carefully chosen fixed value” of the temperature.

Solution (Wegener, 2005): MSTs are such an example.

A bad instance for MA

\[
\begin{array}{cccc}
1 & 1 & 1 & 1 \\
\vdots & \vdots & \vdots & \vdots \\
 n & n & n & n \\
\end{array}
\]

light triangles

\[
\begin{array}{cccc}
1 & 1 & 1 & 1 \\
\vdots & \vdots & \vdots & \vdots \\
 n & n & n & n \\
\end{array}
\]

heavy triangles

Theorem (Wegener, 2005)

The MA with arbitrary temperature computes the MST for this instance only with probability \(e^{-\Omega(n)}\) in polynomial time. SA with temperature \(T_t := n^3(1 - \Theta(1/n))t\) computes the MST in \(O(n \log n)\) steps with probability \(1 - O(1/poly(n))\).

Proof idea: need different temperatures to optimize all triangles.

Simulated Annealing Beats Metropolis in Combinatorial Optimization

Results

Concentrate on wrong triangles: one heavy, one light edge chosen
Simulated Annealing Beats Metropolis in Combinatorial Optimization
Proof Idea

Concentrate on wrong triangles:
one heavy, one light edge chosen

- Soon after initialization $\Omega(n)$ wrong triangles,
both in heavy and light part of the graph
- To correct such triangle, light edge must be flipped in.

- Light edges of heavy triangles still much heavier than heavy
edges of light triangles $\rightarrow$ at temperature $T^*$ almost random
search on light triangles $\rightarrow$ many light triangles remain wrong.
Summary and Conclusions

- Analysis of RSHs in combinatorial optimization
- Starting from toy problems to real problems
- Surprising results
- Interesting techniques
- Analysis of new approaches

→ Altogether, an exciting research direction.

Suggested Reading

Books
Anne Auger, Benjamin Doerr:

Frank Neumann, Carsten Witt:
Book homepage: www.bioinspiredcomputation.com

Thank you!