Generalized Jump Functions

GECCO 2021

Henry Bambury, Antoine Bultel
Joint work with Prof. Benjamin Doerr

École polytechnique
Palaiseau, France

July 10-14, 2021

Contents

Motivation: Multimodal Benchmark Functions for Evolutionary Algorithms

Framework of study

- **Pseudo-Boolean optimization:** $f : \{0,1\}^n \to \mathbb{R}$, find (one of) its global maximum.

- **A wide class of algorithms:** Evolutionary Algorithms.
  Algorithms that rely on notions of mutation and selection for optimization purposes.

- $\{0,1\}^n$ is the **searchspace**;
  $x \in \{0,1\}^n$ is an **individual**;
  f is the **fitness function**.

Benchmark functions

Because of the great diversity of pseudo-Boolean functions, there is no hope for obtaining general theoretical results on the problem.

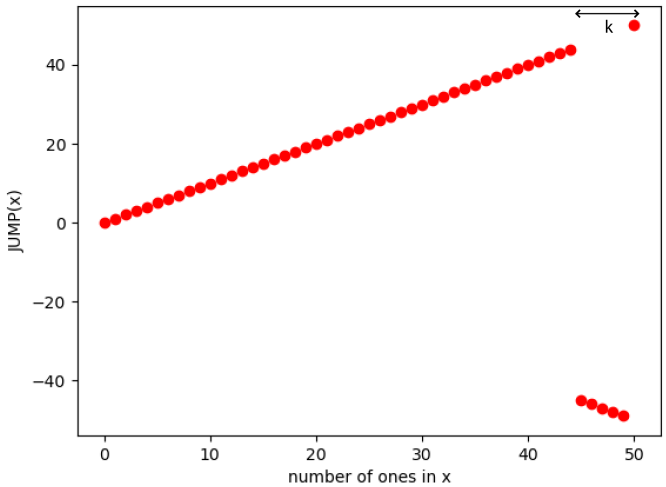
It is standard to focus on a few representative functions to gauge strengths and weaknesses. Famous ones:

- OneMax
- LeadingOnes
- Jump_k
- *etc.*

The choice and design of benchmark functions is a cornerstone of theory of EAs.

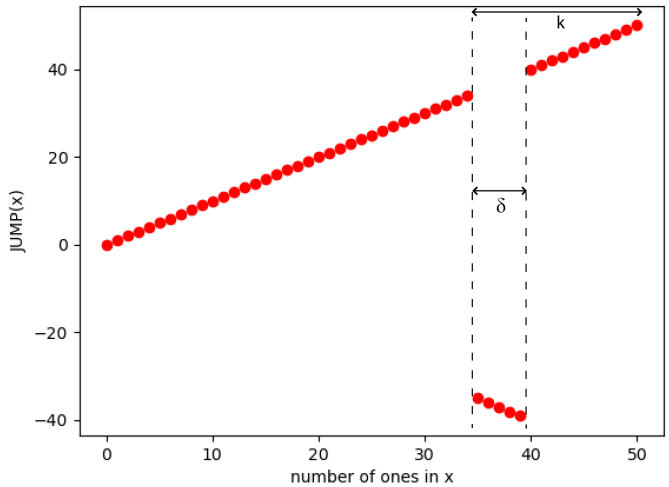
$\textsc{Jump}_{k,\delta}$: a Generalized $\textsc{Jump}_k$

$$\textsc{Jump}_k(x) = \begin{cases} \|x\|_1 & \text{if } \|x\|_1 \in [0..n-k] \cup \{n\}, \\ -\|x\|_1 & \text{otherwise,} \end{cases}$$



$\textsc{Jump}_k$

$$\textsc{Jump}_{k,\delta}(x) = \begin{cases} \|x\|_1 & \text{if } \|x\|_1 \in [0..n-k] \cup \\ & [n-k+\delta..n], \\ -\|x\|_1 & \text{otherwise.} \end{cases}$$



$\textsc{Jump}_{k,\delta}$

Consequences of this generalization

Contrarily to what one could think, $\textsc{Jump}_{k,\delta}$ is **not** equivalent to $\textsc{Jump}_{\delta}$ followed by $\textsc{OneMax}$.

- On $\textsc{Jump}_5$ with $n = 40$, when stuck on the local optima, there is only 1 point with strictly better fitness.
- On $\textsc{Jump}_{10,5}$ with $n = 40$, there are $\sum_{i=1,2,3,4,5} \binom{10}{i} = 647$.

Intuition

Since crossing the valley is (way) easier on $\textsc{Jump}_{k,\delta}$, algorithms should benefit from an exponential speedup.

Impact on the runtime of algorithms

1. $(1+1)$ **EA with Fixed Mutation Rate**
2. Fast $(1+1)$ EA
3. RLS with Stagnation Detection

The Simplest Evolutionary Algorithm

Algorithm 1: The $(1+1)$ EA with fitness function $f : \{0,1\}^n \to \mathbb{R}$ and static mutation rate p

Initialization;
$x \in \{0,1\}^n \leftarrow$ uniform at random;
Optimization;
repeat
    Sample $y \in \{0,1\}^n$ by flipping each bit in x with probability p;
    if $f(y) \geq f(x)$ **then**
        $x \leftarrow y$
until *Stopping condition*;

Performance on $\textsc{Jump}_k$

Result obtained by Doerr et al. in [DLMN17].

$(1+1)$ EA on $\textsc{Jump}_k$

The best possible expected optimization time of the $(1+1)$ EA on $\textsc{Jump}_k$ is asymptotically achieved in $p = \frac{k}{n}$, and is asymptotically

$$\Theta\left(\left(\frac{k}{n}\right)^{-k}\left(\frac{n}{n-k}\right)^{n-k}\right).$$

Furthermore, any deviation from that value leads to exponential loss in runtime.

It is not obvious whether this generalizes to $\textsc{Jump}_{k,\delta}$.

Performance on $\textsc{Jump}_{k,\delta}$

Definition (Standard regime)

Let the *standard regime* (SR) be the space in which: $k = o(n^{1/3})$

Performance in the standard regime

In the SR, if furthermore $p = o(\frac{1}{\sqrt{n\ell}})$,

$$T_p(k, \delta, n) = (1 + o(1)) \frac{1}{\binom{k}{\delta} p^{\delta} (1-p)^{n-\delta}}.$$

Performance on $\textsc{Jump}_{k,\delta}$ in the Standard regime

Theorem

In the SR, the asymptotic best choice of p is $p = \frac{\delta}{n}$, which gives the runtime

$$T_{\delta/n}(k, \delta, n) = (1 + o(1))\binom{k}{\delta}^{-1}\left(\frac{en}{\delta}\right)^{\delta},$$

and any deviation from that optimal value results in exponential in δ loss on the runtime.

1. $(1+1)$ EA with Fixed Mutation Rate
2. **Fast** $(1+1)$ **EA**
3. RLS with Stagnation Detection

$(1+1)$ FEA$_\beta$

- Introduced in [DLMN17] to improve the simple $(1+1)$ EA on $\textsc{Jump}_k$.
- The mutation rate is chosen randomly at each iteration, using a power-law distribution.

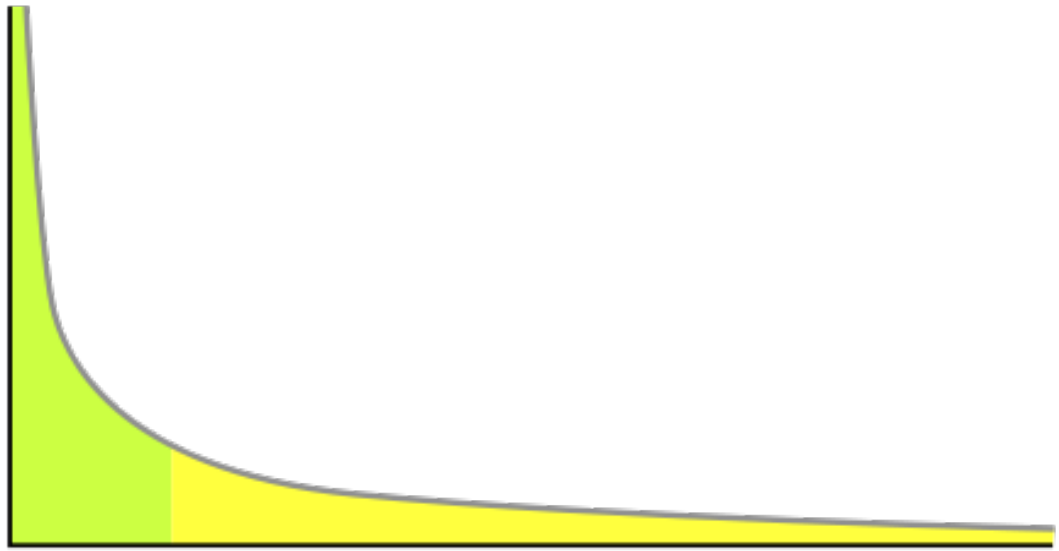


Figure: Plot of a Heavy-tailed power-law distribution.

Pseudocode

Algorithm 2: The $(1+1)$ FEA$_\beta$ with fitness $f : \{0,1\}^n \to \mathbb{R}$

Initialization;
$x \in \{0,1\}^n \leftarrow$ uniform at random;
Optimization;
repeat
    Sample α randomly in $[1..n/2]$ with power-law distribution $D^{\beta}_{n/2}$;
    Sample $y \in \{0,1\}^n$ by flipping each bit in x with probability $\frac{\alpha}{n}$;
    if $f(y) \geq f(x)$ **then**
        $x \leftarrow y$
until *Stopping condition*;

Performance on $\textsc{Jump}_k$

The runtime of $(1+1)$ FEA$_\beta$ is a **small polynomial above** the best runtime with fixed mutation rate.

Theorem [DLMN17]

Let $n \in \mathbb{N}$ and $\beta > 1$. For all $k \in [2..n/2]$, with $m > \beta - 1$, the expected optimization time $T_\beta(k,n)$ of the $(1+1)$ FEA$_\beta$ on $\textsc{Jump}_k$ satisfies

$$T_\beta(k,n) = O\left(C_{n/2}^{\beta} k^{\beta-0.5} T_{opt}(k,n)\right),$$

Where $T_{opt}(k,n)$ is the expected runtime of the simple $(1+1)$ EA with the optimal fixed mutation rate $p = \frac{k}{n}$.

Performance on $\textsc{Jump}_{k,\delta}$

We proved that the result generalizes well.

Theorem

Let $n \in \mathbb{N}$ and $\beta > 1$. For all k, δ in the standard regime, with $\delta > \beta - 1$, the expected optimization time $T_\beta(k, \delta, n)$ of the $(1+1)$ FEA$_\beta$ satisfies

$$T_\beta(k, \delta, n) = O\left(C_{n/2}^{\beta} \delta^{\beta - 0.5} T_{\delta/n}(k, \delta, n)\right).$$

1. $(1+1)$ EA with Fixed Mutation Rate
2. Fast $(1+1)$ EA
3. **RLS with Stagnation Detection**

SD-RLS

- Introduced earlier this year by A.Rajabi and C.Witt in [RW21].
- For each iteration, instead of *standard bit mutation* with $p = \frac{r}{n}$, *randomized local search* of strength r is used. Exactly r bits are chosen uniformly at random and flipped.
- **The strength changes along the run, but not randomly:**
  - Initialized as $r = 1$.
  - If the algorithm stays stuck in the same layer for $\ln(R)\binom{n}{r}$ iterations, then with probability at least $1 - \frac{1}{R}$ there is no improvement at Hamming distance r (R is a control parameter). In this case the strength is increased to $r + 1$.
  - When a strictly better search point is found, return to $r = 1$.

SD-RLS*

Problem: With SD-RLS, termination is not ensured. If the only improvement is at Hamming distance m from the search point, and missed during phase m, the algorithms does not terminate.

Solution: Visit the strengths in a different order.

- SD-RLS: $\mathbf{1} \to \mathbf{2} \to \mathbf{3} \to \mathbf{4} \to \dots$
- SD-RLS*: $\mathbf{1} \to \mathbf{2} - 1 \to \mathbf{3} - 2 - 1 \to \mathbf{4} - 3 - 2 - 1 \to \dots$

The full pseudocode can be found in [RW21].

Performance on $\textsc{Jump}_k$

Theorem [RW21]

Let $n \in \mathbb{N}$. Let $T_{SD-RLS^*}(k,n)$ be the expected runtime of the SD-RLS* on $\textsc{Jump}_k$, with $R \geq n^{2+\varepsilon}$ for some constant $\varepsilon > 0$. For all $k \geq 2$,

$$T_{SD-RLS^*}(k,n) = \begin{cases} \binom{n}{k}\left(1 + O\left(\frac{k^2}{n-2k}\ln(n)\right)\right) & \text{if } k < n/2, \\ O(2^n n \ln(n)) & \text{if } k \geq n/2. \end{cases}$$

Better than the $(1+1)$ EA with optimal mutation rate by a factor $\left(\frac{en}{k}\right)^{-k}\binom{n}{k}$, which is at least $1/e$ for small values of k.

Performance on $\textsc{Jump}_{k,\delta}$

Theorem

Let $T_{SD-RLS^*}(k,\delta,n)$ be the runtime of the SD-RLS* on $\textsc{Jump}_{k,\delta}$. Suppose that there exists a constant $\varepsilon > 0$ such that the control parameter is $R \geq n^{2+\varepsilon}$. Then if $k \leq n - \omega(\sqrt{n})$ and $\delta \geq 3$,

$$T_{SD-RLS^*}(k,\delta,n) = (1+o(1))\left[\ln(R)\sum_{i=1}^{\delta-1}\sum_{j=0}^{i}\binom{n}{j} + \binom{n}{\delta}\binom{k}{\delta}^{-1}\right].$$

Comparaison to other algorithms

It is easy to see that this runtime is asymptotically larger than the previous ones. The following lemma puts that difference into perspective.

Theorem

For any integer K, there exists an instance of $\textsc{Jump}_{k,\delta}$, within the standard regime, on which

$$T_{SD-RLS^*}(k,\delta,n) = \Omega\left(n^{K-1} T_{\frac{1}{n}}(k,\delta,n)\right).$$

Experiments

Experiments

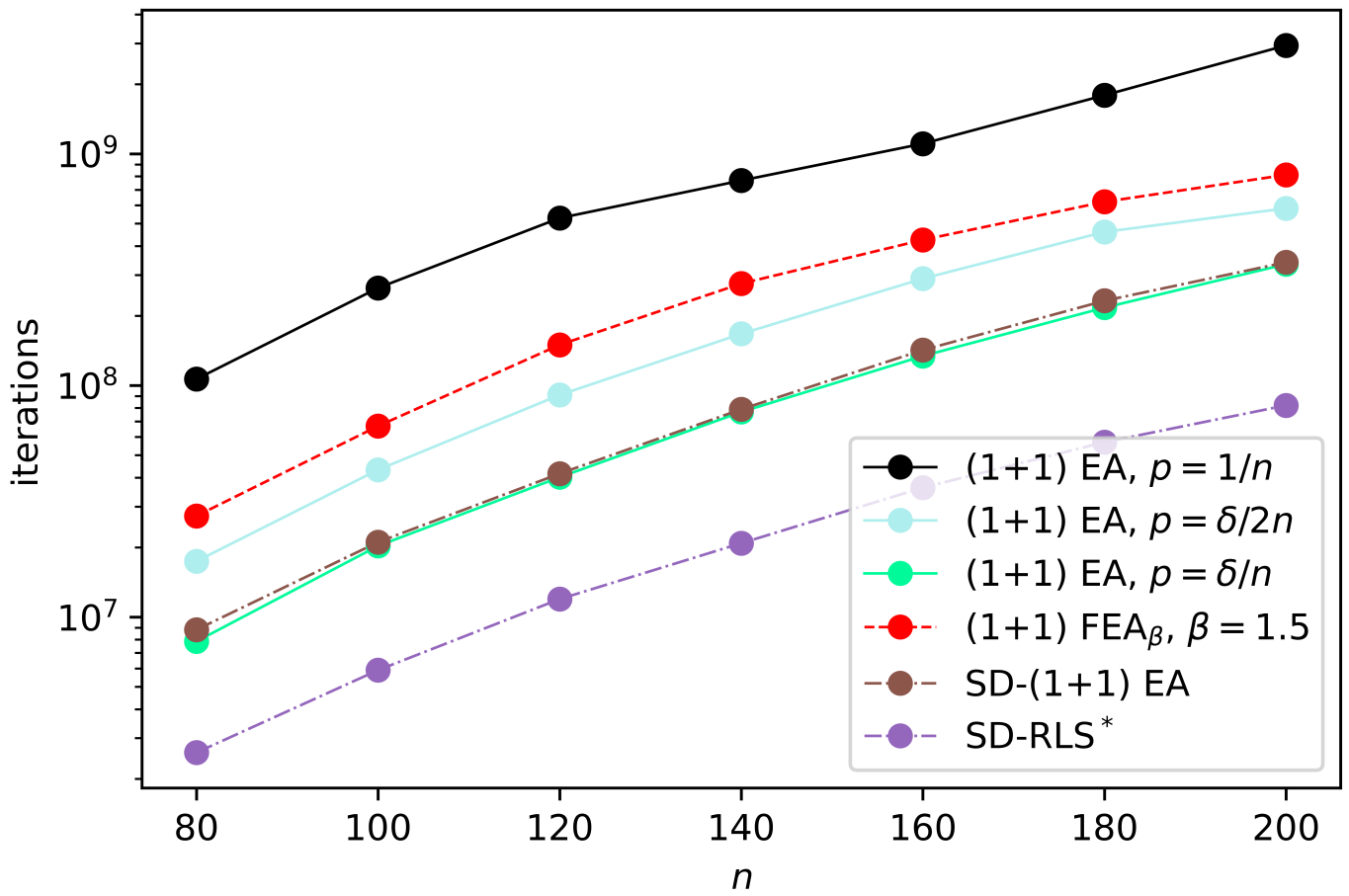


Figure: Optimization times of different algorithms on $\textsc{Jump}_{k,\delta}$ with $k = \delta = 4$.

Experiments

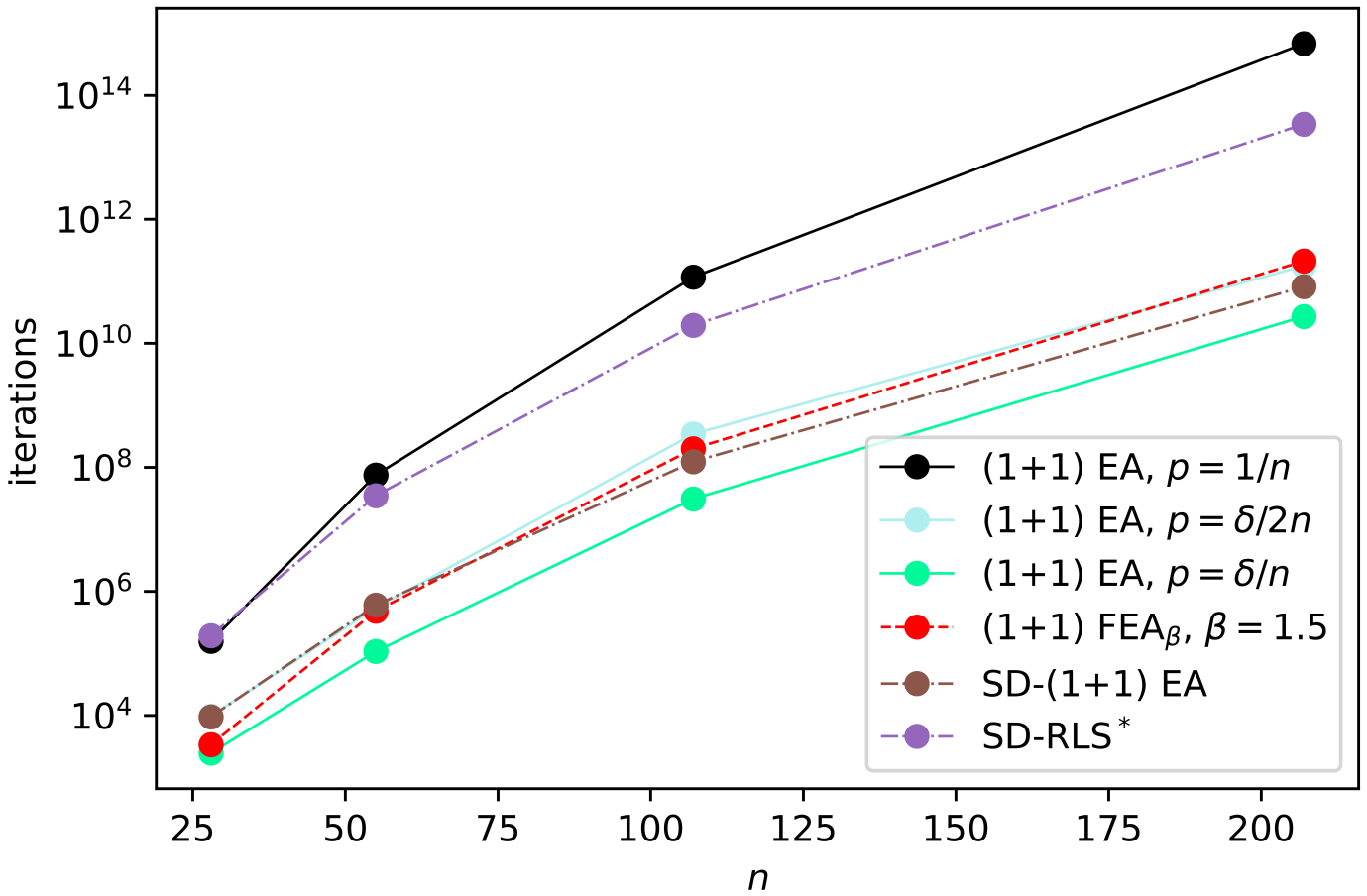
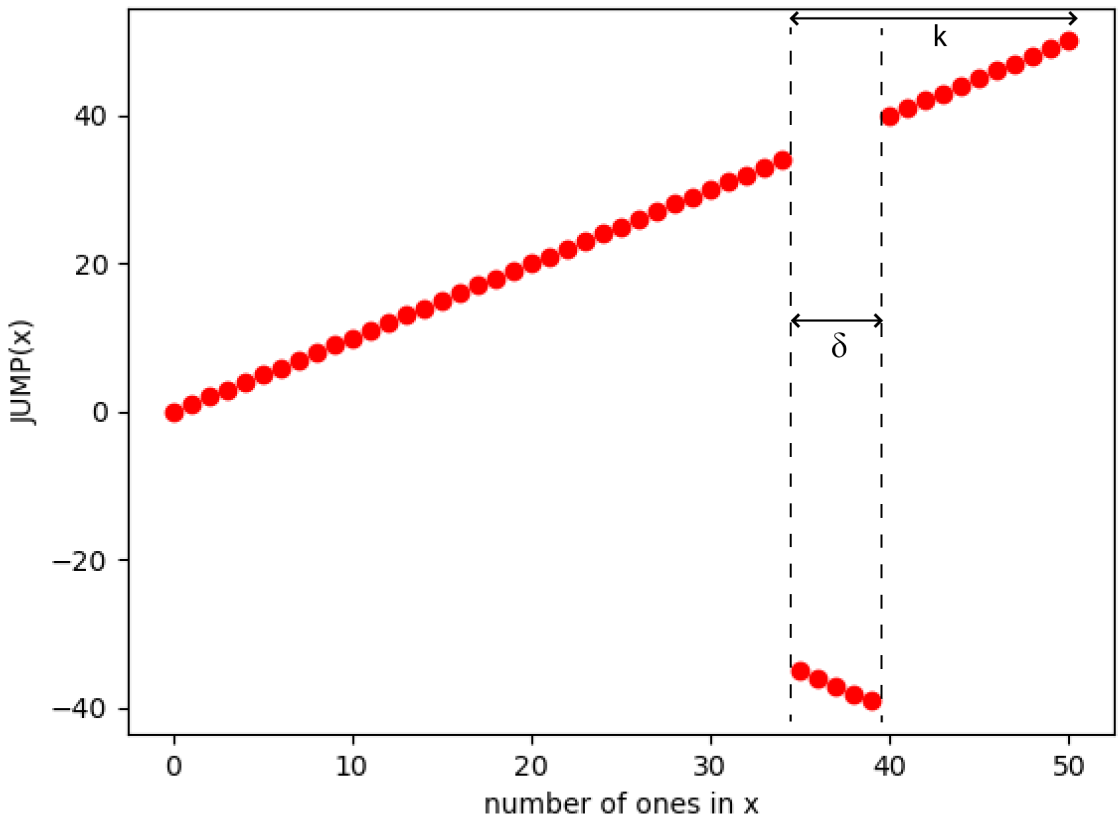


Figure: Optimization times on $\textsc{Jump}_{k,\delta}$ for $k = 3\ln(n)$, $\delta = \frac{k}{2}$.

Conclusion

Conclusion: The $\textsc{Jump}_{k,\delta}$ function

$$\textsc{Jump}_{k,\delta}(x) = \begin{cases} \|x\|_1 & \text{if } \|x\|_1 \in [0..n-k] \cup [n-k+\delta..n], \\ -\|x\|_1 & \text{otherwise.} \end{cases}$$



A more realistic version of the well-known $\textsc{Jump}_k$ function.

Conclusion: Performance of Algorithms

Algorithm	**Jump$_k$**	**Jump$_{k,\delta}$ in the SR**
$(1+1)$ EA with optimal MR	$\Theta((\frac{k}{n})^{-k}(\frac{n}{n-k})^{n-k})$ [DLMN17]	$(1+o(1))(\frac{en}{\delta})^{\delta}\binom{k}{\delta}^{-1}$
$(1+1)$ FEA$_\beta$	$O(C^{\beta}_{n/2}k^{\beta-0.5}(\frac{k}{n})^{-k}(\frac{n}{n-k})^{n-k})$ [DLMN17]	$O(C^{\beta}_{n/2}\delta^{\beta-0.5}(\frac{en}{\delta})^{\delta}\binom{k}{\delta}^{-1})$
SD-RLS*	$\binom{n}{k}(1+O(\frac{k^2}{n-2k}\ln(n)))$ [RW21]	$(1+o(1))[\ln(R)\sum_{i=1}^{\delta-1}\sum_{0}^{i}\binom{n}{j}+\binom{n}{\delta}\binom{k}{\delta}^{-1}]$

Conclusion: Performance of Algorithms

Algorithm	**Jump$_k$**	**Jump$_{k,\delta}$ in the SR**
$(1+1)$ EA with optimal MR	$\Theta((\frac{k}{n})^{-k}(\frac{n}{n-k})^{n-k})$	$(1+o(1))(\frac{en}{\delta})^{\delta}\binom{k}{\delta}^{-1}$
$(1+1)$ FEA$_\beta$	$k^{\beta-0.5}$	$\delta^{\beta-0.5}$
SD-RLS*	$\left(\frac{en}{k}\right)^{k}\binom{n}{k}^{-1}$	$\Omega(n^K), \forall K > 0$

References I


Benjamin Doerr, Huu Phuoc Le, Régis Makhmara, and Ta Duy Nguyen, *Fast genetic algorithms*, Genetic and Evolutionary Computation Conference, GECCO 2017, ACM, 2017, pp. 777–784.


Amirhossein Rajabi and Carsten Witt, *Stagnation detection with randomized local search*, Evolutionary Computation in Combinatorial Optimization, EvoCOP 2021, Springer, 2021, pp. 152–168.