1 Pseudo-code

(1+1) EA

1. Choose $s \in \{0,1\}^n$ randomly.
2. Produce s' by flipping each bit of s with probability $1/n$.
3. Replace s by s' if $f(s') \geq f(s)$.
4. Repeat Steps 2 and 3 forever.

RLS

1. Choose $s \in \{0,1\}^n$ randomly.
2. Produce s' from s by flipping **one** randomly chosen bit.
3. Replace s by s' if $f(s') \geq f(s)$.
4. Repeat Steps 2 and 3 forever.

2 Chernoff Bound Initialisation

2.1 Expected Ones Initialisation

Given that both algorithms initialise an n-set of binary string at random specifically $X_i = \text{Unif}\{0,1\}$ for $i = 1, 2, \ldots, n$. Therefore the probability of getting 0 or 1 is the same as

$$P(X_i = 1) = P(X_i = 0) = \frac{1}{2}$$

The expected value of each X_i is

$$\mathbb{E}[X_i] = 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{2} = \frac{1}{2}$$

Suppose that $X = X_1 + X_2 + \ldots + X_n$, then the expected value of X by using the *Linearity of Expectation* is

$$\mathbb{E}[X] = \sum_{i=1}^{n} \mathbb{E}[X_i] = \frac{n}{2}$$

If X is the summation of those values inside the binary string, then there will be expected to have $\frac{n}{2}$ ones.

2.2 Chernoff

$$\forall \delta > 0 : P(X > (1+\delta) \cdot \mathbb{E}[X]) < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}} \right)^{\mathbb{E}[X]} \qquad (\bigstar)$$

$$\forall\, 0 < \delta < 1 : P(X < (1-\delta) \cdot \mathbb{E}[X]) < e^{-\mathbb{E}[X]\delta^2/2}$$

From the previous section, it's known that $\mathbb{E}[X] = \frac{n}{2}$, choose inequality $(\bigstar)$ and then substitute.

$$P\left(X > (1+\delta) \cdot \frac{n}{2}\right) < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}} \right)^{\frac{n}{2}}$$

$$\Rightarrow P\left(X > \left(\frac{1}{2} + \frac{\delta}{2}\right) \cdot n\right) < \ldots$$

Let $2\delta' = \delta$, which is always inside the bound of $\delta > 0$. Now choose the domain of δ as $0 < \delta \leq 1$. Therefore, the new domain will be $0 < 2\delta' \leq 1 \Rightarrow 0 < \delta' \leq \frac{1}{2}$, thus

$$P\left(X > \left(\frac{1}{2} + \delta'\right) \cdot n\right) < \left(\frac{e^{2\delta'}}{(1+2\delta')^{1+2\delta'}}\right)^{\frac{n}{2}}$$

Evaluate $\rho = \frac{e^{2\delta'}}{(1+2\delta')^{1+2\delta'}}$

$$\begin{aligned}
\rho &= \frac{e^{2\delta'}}{(1+2\delta')^{\,1+2\delta'}} \\
\ln \rho &= \ln\left(\frac{e^{2\delta'}}{(1+2\delta')^{\,1+2\delta'}}\right) \\
&= \ln\left(e^{2\delta'}\right) \;-\; \ln\left((1+2\delta')^{\,1+2\delta'}\right), \quad \text{Quotient log law} \\
&= 2\delta' \;-\; (1+2\delta')\,\ln(1+2\delta'), \quad \text{Power Log Law} \\
\ln \rho &= 2\delta' - (1+2\delta')\ln(1+2\delta') < 0, \quad \text{Negative constant}
\end{aligned}$$

Therefore,

$$\begin{aligned}
P\left(X > \left(\frac{1}{2} + \delta'\right) \cdot n\right) &< \left(\frac{e^{2\delta'}}{(1+2\delta')^{1+2\delta'}}\right)^{\frac{n}{2}} \\
&= \rho^{\frac{n}{2}} \\
&= e^{\frac{n}{2}\ln\rho} \\
P\left(X > \left(\frac{1}{2} + \delta'\right) \cdot n\right) &< e^{-\Omega(n)}
\end{aligned}$$

Then,

$$\begin{aligned}
P\left(X > \frac{2n}{3}\right) &< e^{-\Omega(n)} \\
\Rightarrow 1 - P\left(X > \frac{2n}{3}\right) &\geq 1 - e^{-\Omega(n)} \\
\Rightarrow P\left(X \leq \frac{2n}{3}\right) &\geq 1 - e^{-\Omega(n)}
\end{aligned}$$

This means that both algorithms with uniform random binary string initialisation will have a very high chance of having at most $2n/3$ one-bits inside the initialisation. It's so high that with large n or large binary space it's almost gauranteed.

3 RLS Lower Bound

Given from the initialisation that there are expected to have $2n/3$ one-bits, thus $m = n/3$ zero-bits needed to be flipped. Each zero-bit has a chance of being flipped is $1/n$. We want to know what is the lower bound Ω that will let flipping all distinct 0-bits at least once. This is a Collector's Coupon Theorem.

3.1 Collector's Coupon Theorem

Suppose that we have n distinct cards and we draw each everytime with *replacement*. How many times do I expect to draw to collect all of those cards. This implies a similarity as our prolem with RLS lower bound.

From Wikipedia, let T be the total number of times to finish flipping all the 0-bits, t_i be the number of times when reaching to the i^{th} bit after $i-1$ bits, therefore the probability for reach. Then $T = t_k + t_{k+1} + \ldots + t_m$. The probability of each 0-bit being picked is,

$$p_i = \frac{m-(i-1)}{n}$$

Here comes the *significant* point, each t_i has a *geometric distribution*, recall the geometric distribution from Wikipedia,

> *The probability distribution of the number X of Bernoulli trials needed to get one success.*

So, each time t_i reflects a geometric distribution since we want to know how many times do we need to be able to pick that 0-bit, by the geometric distribution we can infer the expected value as $\mathbb{E}[t_i] = 1/p_i$, by the *linearity of the Expectation*,

$$\begin{aligned}
\mathbb{E}[T] &= \mathbb{E}[t_k] + \mathbb{E}[t_{k+1}] + \ldots + \mathbb{E}[t_m] \\
&= \frac{1}{p_k} + \frac{1}{p_{k+1}} + \ldots + \frac{1}{p_m} \\
&= \frac{n}{m-(k-1)} + \frac{n}{m-k} + \ldots + \frac{n}{m-(m-1)} \\
&= n\left(\frac{1}{n-(k-1)} + \frac{1}{n-k} + \ldots + \frac{1}{1}\right) \\
&= nH_{n-k+1} \\
&= n\ln(n-k+1) + C \\
\mathbb{E}[T] &= \Omega(n\log n)
\end{aligned}$$

By using the Theorem, we can derive a lower bound for the RLS to be $\Omega(n \log n)$.

4 (1+1)EA Lower Bound

Given the initialisation, as for the RLS the lower bound is $\Omega(n \log n)$, so we are also trying to show that for the EA it's also $\Omega(n \log n)$. We may assume that everytime we flip the 0-bit to 1-bit it's will yield a improvement. We may choose $t = (n-1)\ln n$ iterations until optimum, the reason of choosing such t maybe from more complicated mathmatical proof, since this will give a nice bound for when hitting an optimum.

4.1 Waterfall Probability

$$\begin{aligned}
1 - \frac{1}{n}, &\quad \text{the probability of a specific bit not being flipped.} \\
\left(1 - \frac{1}{n}\right)^t, &\quad \text{the probability of a specific bit not being flipped for all } t \text{ iterations} \\
1 - \left(1 - \frac{1}{n}\right)^t, &\quad \text{the probability of a specific bit being flipped for all } t \text{ iterations} \\
\left[1 - \left(1 - \frac{1}{n}\right)^t\right]^m, &\quad \text{the probability of all } m \text{ bits being flipped for all } t \text{ iterations} \\
1 - \left[1 - \left(1 - \frac{1}{n}\right)^t\right]^m, &\quad \text{the probability of all } m \text{ bits not being flipped for all } t \text{ iterations}
\end{aligned}$$

The last one is called the *failure probability*.

4.2 Some Calculus

As $n \to \infty$ it seems to yields

$$1 - e^{-1/3}$$

We have found that the failure probability is constant which is $\Omega(1)$

Observation

To derive the $\geq$ as the failure probability for $P(T > t)$ can be done by observing the initialisation. As we have known that, there will be $\leq \frac{2n}{3}$ one-bits this mean that there will be $\geq \frac{n}{3}$ zero-bits. We can replace the constant $\frac{1}{3}$ with other scalar in the range of $x \in \left[\frac{1}{3}, 1\right]$, think the failure probability as $1 - e^{-x}$, thus it's increasing therefore we have that

$$P(T > t) \geq 1 - e^{-1/3}$$

Then the probability failing to reach optimum is

$$P(T > t) \geq 1 - e^{-1/3} = \Omega(1)$$

Explanation

The reason of choosing the failure probability instead of successful probability for the lower bound is because we give a candidate of t iterations, the number of iterations we want until it's the iteration of optimum. But then after some maths, it is shown that it's not sufficient enough for t iteration with a probability therefore we need some more iterations, which is the lower bound. It's the buffer of how much more is needed.

4.3 Tying Everything Together

Using the Tail sum formula

$$\begin{aligned}
\mathbb{E}[T] &= \sum_{t=1}^{\infty} P(T > t) \\
&= \sum_{t=1}^{(n-1)\ln n} P(T > t) + \sum_{k=(n-1)\ln(n)+1}^{\infty} P(T > k) \\
&\geq \sum_{t=1}^{(n-1)\ln n} P(T > t) \\
&\geq \sum_{t=1}^{(n-1)\ln n} \Omega(1) \\
&\geq (n-1)\ln(n)\Omega(1) \\
\mathbb{E}[T] &= \Omega(n \log n)
\end{aligned}$$

Therefore the lower bound for (1+1)EA is $\Omega(n \log n)$

Linear Function with Harmonic Weights Problem F3

The F3 problem is intened to maximise value of the linear function, but this can deduce to the OneMax problem, as for the definition of the linear function of $i \cdot x_i + \ldots + n \cdot x_n$,

the value of the function only incease when there is increasing in the number of ones . Given $\mathbf{x} \in \{0,1\}^n$ thus we have that,

$$\arg\max_{\mathbf{x}\in\{0,1\}^n} \mathsf{F}(\mathbf{x}) \text{ where } \mathsf{F}(\mathbf{x}) = \sum_{i=1}^{n} ix_i$$

RLS Setup

RLS

1. Choose $\mathbf{x} \in \{0,1\}^n$ randomly.
2. Produce $\mathbf{x}'$ from $\mathbf{x}$ by flipping **one** randomly chosen bit.
3. Replace $\mathbf{x}$ by $\mathbf{x}'$ if $\mathsf{F}(\mathbf{x}) \geq \mathsf{F}(\mathbf{x})$.
4. Repeat Steps 2 and 3 forever.

Problem

Prove that Random Local Search with fitness evaluations to reach an optimal search point for the function $\mathsf{F}3$ is

$$\mathcal{O}(n \log n) \text{ with probability of } \Omega(1)$$

Proof. From the **Chernoff Bound Initialisation** above note, we know that the tuple $\mathbf{x}$ will expect to have $2n/3$ ones after initialisation with a high probability. Thus now we care for $m = n/3$ 0-bits to be flipped. Let t_i be the number of times to flip each 0-bit where $i = k, k+1, \ldots, m$ $t_i \sim \text{Geo}(p_i)$, where p_i is defined as

$$p_i = \frac{m-(i-1)}{n}$$

And the expected number of times to flipped the i^{th} to 1 is

$$\mathbb{E}[t_i] = \frac{n}{m-(i-1)}$$

Let $T = \sum_{i=k}^{m-1} t_i$ be the total time to get to optimum. Then, using the **Collector's Coupon Theorem** above note.

$$\mathbb{E}[T] = n\ln(n-k+1) + \mathcal{C}, \qquad \text{where } \mathcal{C} > 0$$

Recall that for upper bound,

$$\mathcal{O}(n\log n) = \left\{g(n) : \exists c > 0 : \exists n_0 \in \mathbb{N}^+ : \forall n \geq n_0 : g(n) \leq c \cdot n \log n\right\}$$

Given that $g(n) = n\ln(n-k+1) + \mathcal{C}$, thus:

$$\begin{aligned} n\ln(n-k+1) + \mathcal{C} &\leq cn \\ \Rightarrow c &\geq \ln(n-k+1) + \mathcal{C} \end{aligned}$$

Therefore there exists such $c > 0$, hence

$$\mathbb{E}[T] = \mathcal{O}(n \log n)$$

By Markov's inequality, for any $s > 1$,

$$P(T \geq s \cdot \mathbb{E}[T]) \leq \frac{1}{s}.$$

Choosing $s = 2$, we have

$$P(T \geq 2 \cdot \mathbb{E}[T]) \leq \frac{1}{2},$$

and thus

$$P(T \leq 2 \cdot \mathbb{E}[T]) \geq 1 - \frac{1}{2} = \frac{1}{2} = \Omega(1).$$

Since $2 \cdot \mathbb{E}[T] = \mathcal{O}(n \log n)$, the runtime is $\mathcal{O}(n \log n)$ with probability $\Omega(1)$. ■

(1+1) EA Setup

(1+1) EA

1. Choose $\mathbf{x} \in \{0,1\}^n$ randomly.
2. Produce $\mathbf{x}'$ from flipping each bit of $\mathbf{x}$ with probability $1/n$
3. Replace $\mathbf{x}$ by $\mathbf{x}'$ if $\mathsf{F}(\mathbf{x}') \geq \mathsf{F}(\mathbf{x})$.
4. Repeat Steps 2 and 3 forever.

Problem

Prove that (1+1) EA with fitness evaluations to reach an optimal search point for the function F3 is

$$\mathcal{O}(n \log n) \text{ with probability of } \Omega(1)$$

Proof. Given that the $(1+1)$ EA and $(1+1)$ EA_b is similar from the pseudo-code of [1, p.42]. From [1, p.44]

> Assume that we are working in a search space S and consider w.l.o.g. a function $f : S \to \mathbb{R}$ that should be maximized. S is partitioned into disjoint sets $A_1, \ldots, A_m$ such that $A_1 <_f A_2 <_f \cdots <_f A_m$ holds, where $A_i <_f A_j$ means that $f(a) < f(b)$ holds for all $a \in A_i$ and all $b \in A_j$. In addition, A_m contains only optimal search points. An illustration is given in Figure 4.1. We denote for a search point $x \in A_i$ by $p(x)$ the probability that in the next step a solution $x' \in A_{i+1} \cup \cdots \cup A_m$ is produced. Let $p_i = \min_{a \in A_i} p(a)$ be the smallest probability of producing a solution with a higher partition number. [1, p. 44]

There are $n = m$, as similar problem to OneMax, levels. Each level are defined as $A_i = \{1\}^i$ for $i = 1, 2, \ldots, m$. Suppose that the algorithm is currently in A_{m-k} level. There are k 0-bits left.

$$\frac{1}{n}, \quad \text{A specific bit is flipped to 1}$$

$$1 - \frac{1}{n}, \quad \text{A specific bit is not flipped to 1}$$

$$\left(1 - \frac{1}{n}\right)^{(n-1)}, \quad \text{All } n-1 \text{ bits are not flipped to 1}$$

$$\frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{(n-1)}, \quad \text{All } n-1 \text{ bits are not flipped to 1, except for a bit to be flipped to 1}$$

$$\frac{k}{n} \cdot \left(1 - \frac{1}{n}\right)^{(n-1)}, \quad \text{All } n-1 \text{ bits are not flipped to 1, except for the } k \text{ bits to be flipped to 1}$$

We have that,

$$\left(1 - \frac{1}{n}\right)^{(n-1)} \geq \frac{1}{e}$$

Thus,

$$\frac{k}{n} \cdot \left(1 - \frac{1}{n}\right)^{(n-1)} \geq \frac{k}{ne}$$

Then the transition probability to move to the next level for a better result is.

$$p_{n-k} \geq \frac{k}{en}$$

From Lemma 4.1[1, p.45] we have that $A_{n-k} \leq \frac{1}{p_{n-k}} \leq \frac{en}{k}$. The algorithm must move up all the levels until to reach level m for the optimum. Thus we have that,

$$\begin{aligned}
\mathbb{E}[T] &\leq \sum_{k=1}^{m-1} \frac{en}{k} \\
\mathbb{E}[T] &\leq en \sum_{k=1}^{m-1} \frac{1}{k} \\
\mathbb{E}[T] &\leq en \cdot H_{m-1}
\end{aligned}$$

Using the proof from above note of **Collector's Coupon Theorem** we have that

$$en \cdot H_{m-1} = \mathcal{O}(n \log n)$$

Thus

$$\mathbb{E}[T] = \mathcal{O}(n \log n)$$

By Markov's inequality, for any $s > 1$,

$$P(T \geq s \cdot \mathbb{E}[T]) \leq \frac{1}{s}.$$

Choosing $s = 2$, we have

$$P(T \geq 2 \cdot \mathbb{E}[T]) \leq \frac{1}{2},$$

and thus

$$P(T \leq 2 \cdot \mathbb{E}[T]) \geq 1 - \frac{1}{2} = \frac{1}{2} = \Omega(1).$$

Since $2 \cdot \mathbb{E}[T] = \mathcal{O}(n \log n)$, the runtime is $\mathcal{O}(n \log n)$ with probability $\Omega(1)$. ■

References

[1] Frank Neumann and Carsten Witt. *Bioinspired Computation in Combinatorial Optimization: Algorithms and Their Computational Complexity*. Natural Computing Series. Springer, 2010. DOI: 10.1007/978-3-642-16544-3.