Nature-inspired Algorithms

Lecture 4

Algorithm Engineering Group
Hasso Plattner Institute, University of Potsdam

29 May 2017

Runtime analysis

Let's consider more interesting cases...

Recall from Project 1:

(1+1) EA

Choose x uniformly at random from $\{0,1\}^n$;
while *stopping criterion not met* **do**
  $y \leftarrow x$;
  foreach $i \in \{1, \ldots, n\}$ **do**
    With probability $1/n$, $y_i \leftarrow (1 - y_i)$;
  end
  if $f(y) \geq f(x)$ **then** $x \leftarrow y$;
end

In each iterations, how many bits flip in expectation?

What is the probability exactly one bit flips?

What is the probability exactly two bits flip?

What is the probability that no bits flip?

Runtime Analysis

Theorem (Droste et al., 2002)

The expected runtime of the (1+1) EA for an arbitrary function $f\colon \{0,1\}^n \to \mathbb{R}$ is $O(n^n)$.

Proof.
Without loss of generality, suppose $x^\star$ is the unique optimum and x is the current solution.

Let $k = |\{i : x_i \neq x_i^\star\}|$.

Each bit flips (resp., does not flip) with probability $1/n$ (resp., with probability $1 - 1/n$).

Runtime Analysis

In order to reach the global optimum **in the next step** the algorithm has to mutate the k bits and leave the $n - k$ bits alone.

The probability to create the global optimum in the next step is

$$\left(\frac{1}{n}\right)^k \left(1 - \frac{1}{n}\right)^{n-k} \geq \left(\frac{1}{n}\right)^n = n^{-n}.$$

Assuming the process has not already generated the optimal solution, in expectation we wait $O(n^n)$ steps until this happens. ∎

Note: we are simply overestimating the time to find the optimal *for any arbitrary pseudo-Boolean function*.

Note: The upper bound is *worse* than for RandomSearch. In fact, there are functions where RandomSearch is *guaranteed* to perform better than the (1+1) EA.

Initialization

Recall from Project 1: $\textsc{OneMax}: \{0,1\}^n \to \mathbb{R}, x \mapsto |x|$;

How good is the initial solution?

Let X count the number of 1-bits in the initial solution. $E(X) = n/2$.

How likely to get exactly $n/2$?

$$\Pr(X = n/2) = \binom{n}{n/2} \frac{1}{2^{n/2}} \left(1 - \frac{1}{2}\right)^{n/2} = \binom{n}{n/2} 2^{-n}.$$

For $n = 100$, $\Pr(X = 50) \approx 0.0796$

Initialization (Tail Inequalities)

How likely is the initial solution no worse than $(3/4)n$?

Markov's Inequality

Let X be a random variable with $P(X < 0) = 0$. For all $a > 0$ we have

$$\Pr(X \geq a) \leq \frac{E(X)}{a}.$$

$$E(X) = n/2; \text{ then } \Pr(X \geq (3/4)n) \leq \frac{E(X)}{(3/4)n} \leq 2/3$$

Initialization (Tail Inequalities)

Let $X_1, X_2, \ldots X_n$ be independent Poisson trials each with probability p_i;
For $X = \sum_{i=1}^{n} X_i$, the expectation is $E(X) = \sum_{i=1}^{p_i}$.

Chernoff Bounds

- for $0 \leq \delta \leq 1$, $\Pr(X \leq (1-\delta)E(X)) \leq e^{\frac{-E(x)\delta^2}{2}}$.
- for $\delta > 0$, $\Pr(X > (1+\delta)E(X)) \leq \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{E(X)}$.

E.g., $p_i = 1/2$, $E(X) = n/2$, fix $\delta = 1/2 \to (1+\delta)E(X) = (3/4)n$,

$$\Pr(X > (3/4)n) \leq \left(\frac{e^{1/2}}{(3/2)^{(3/2)}}\right)^{n/2} = c^{-n/2}.$$

Initialization (Tail Inequalities):

A simple example

Let $n = 100$. How likely is the initial solution no worse than $\textsc{OneMax}(x) = 75$?

$\Pr(X_i) = 1/2$ and $E(X) = 100/2 = 50$.

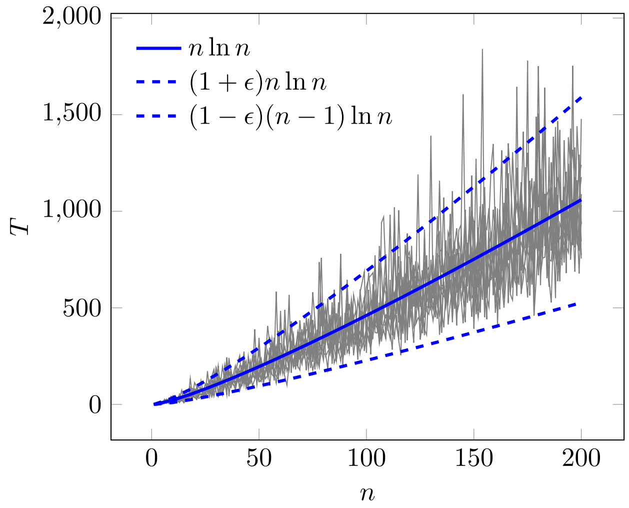
Markov: $\Pr(X \geq 75) \leq \frac{50}{75} = \frac{2}{3}$.

Chernoff: $\Pr(X \geq (1 + 1/2)50) \leq \left(\frac{\sqrt{e}}{(3/2)^{(3/2)}}\right)^{50} < 0.0054$.

In reality, $\Pr(X \geq 75) = \sum_{i=75}^{100} \binom{100}{i} 2^{-100} \approx 0.0000002818141$.

Runtime analysis – RLS on OneMax

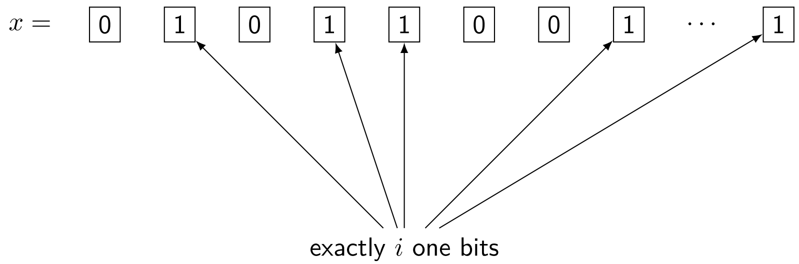
10 trials of $n \in \{1, \dots, 200\}$.



We want to rigorously understand this behavior.

Runtime analysis – RLS on OneMax

Let's suppose: during the execution of RLS the current string x looks like this:



Let's look into

- p_i: probability that RLS makes an improving move from x
- T_i: time until RLS makes an improving move from x

Runtime analysis – RLS on OneMax

0	1	2	3	4	5		
0	0	0	0	0	**0**	$p_0 = \frac{6}{6}$	$E(T_0) = \frac{6}{6}$
0	0	**0**	0	0	1	$p_1 = \frac{5}{6}$	$E(T_1) = \frac{6}{5}$
0	0	1	0	0	1	$p_2 = \frac{4}{6}$	$E(T_2) = \frac{6}{4}$
1	0	1	0	**0**	1	$p_3 = \frac{3}{6}$	$E(T_3) = \frac{6}{3}$
1	**0**	1	0	1	1	$p_4 = \frac{2}{6}$	$E(T_4) = \frac{6}{2}$
1	1	1	**0**	1	1	$p_5 = \frac{1}{6}$	$E(T_5) = \frac{6}{1}$

Runtime analysis – RLS on OneMax

Runtime

T is the random variable that counts the number of steps (function evaluations) taken by RLS until the optimum is generated.

$$\begin{aligned} E(T) &= E(T_0) + E(T_1) + \dots + E(T_5) \\ &= 1/p_0 + 1/p_1 + \dots + 1/p_5 \\ &= \sum_{i=0}^{5} \frac{1}{p_i} = \sum_{i=0}^{5} \frac{6}{i+1} = 6 \sum_{i=1}^{6} \frac{1}{i} = 6 \cdot 2.45 = 14.7 \end{aligned}$$

Runtime analysis – RLS on OneMax

$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	$\boxed{0}$	$\cdots$	$\boxed{0}$		$p_0 = \frac{n}{n}$	$E(T_0) = \frac{n}{n}$
0	1	2	3		n			
$\boxed{0}$	$\boxed{0}$	$\boxed{1}$	$\boxed{0}$	$\cdots$	$\boxed{0}$		$p_1 = \frac{n-1}{n}$	$E(T_1) = \frac{n}{n-1}$
0	1	2	3		n			
$\boxed{1}$	$\boxed{0}$	$\boxed{1}$	$\boxed{0}$	$\cdots$	$\boxed{0}$		$p_2 = \frac{n-2}{n}$	$E(T_2) = \frac{n}{n-2}$
0	1	2	3		n			
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	
$\boxed{1}$	$\boxed{0}$	$\boxed{1}$	$\boxed{1}$	$\cdots$	$\boxed{1}$		$p_{n-1} = \frac{1}{n}$	$E(T_{n-1}) = \frac{n}{1}$
0	1	2	3		n			

remaining zero

Coupon collector process

Suppose there are n different kinds of coupons. We must collect all n coupons during a series of trials.

In each trial, exactly one of the n coupons is drawn, each one equally likely. We must keep drawing in each trial until we have collected each coupon at least once.

Starting with zero coupons, what is the exact number of trials needed before we have all n coupons?

Theorem (Coupon collector theorem)

Let T be the number of trials until all n coupons are collected. Then

$$E(T) = \sum_{i=0}^{n-1} \frac{1}{p_{i+1}} = \sum_{i=0}^{n-1} \frac{n}{n-i} = n \sum_{i=0}^{n-1} \frac{1}{i}$$
$$= n \cdot H_n = n(\log n + \Theta(1)) = n \log n + O(n)$$

Coupon collector process: concentration bounds

What is the probability that $T > n \ln n + O(n)$?

Theorem (Coupon collector upper bound)

Let T be the number of trials until all n coupons are collected. Then

$$\Pr(T \geq (1+\epsilon) n \ln n) \leq n^{-\epsilon}$$

Proof.

Probability of choosing a specific coupon: $1/n$.

Probability of not choosing a specific coupon: $1 - 1/n$.

Probability of not choosing a specific coupon for t rounds: $(1-1/n)^t$

Probability that one of the n coupons is not chosen in t rounds: $n \cdot (1-1/n)^t$ *(union bound)*

Let $t = cn \ln n$,

$$\Pr(T \geq cn \ln n) \leq n(1-1/n)^{cn \ln n} \leq ne^{-c \ln n} = n \cdot n^{-c} = n^{-c+1}$$

■

Coupon collector process: concentration bounds

Theorem (Coupon collector lower bound) (Doerr, 2011)

Let T be the number of trials until all n coupons are collected. Then

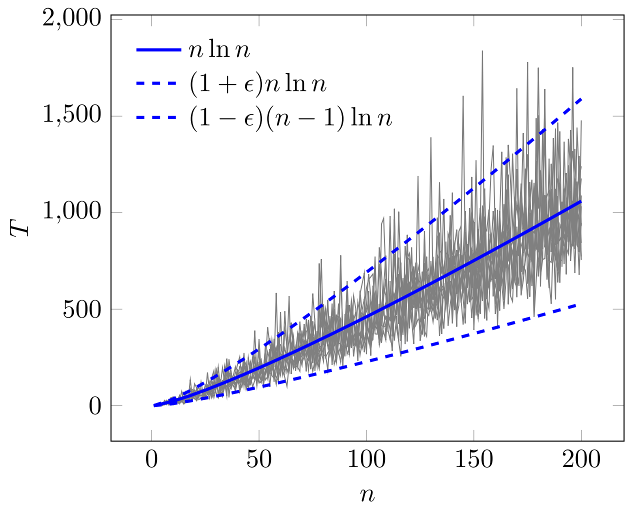
$$\Pr(T < (1-\epsilon)(n-1) \ln n) \leq e^{-n^{\epsilon}}$$

Corollary

Let T be the time for RLS to optimize OneMax. Then,

$$E(T) = \Theta(n \log n)$$
$$\Pr(T \geq (1+\epsilon) n \ln n) \leq n^{-\epsilon}$$
$$\Pr(T < (1-\epsilon)(n-1) \ln n) \leq e^{-n^{-\epsilon}}$$

Runtime analysis – RLS on OneMax



What about **(1+1) EA**? **Can we use Coupon Collector? Why/why not?**

Fitness levels

Observation: fitness during optimization is always monotone increasing

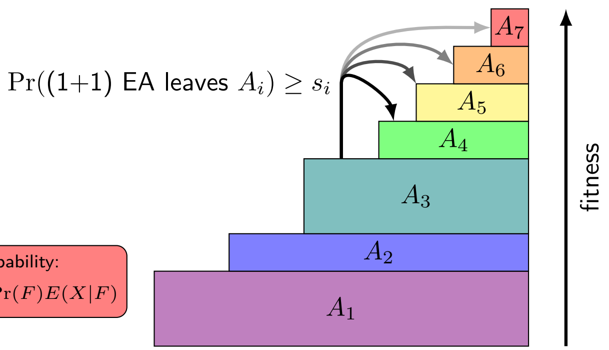
Idea: partition the search space $\{0,1\}^n$ into m sets $A_1, \dots A_m$ such that

1. $\forall i \neq j: \quad A_i \cap A_j = \emptyset$
2. $\bigcup_{i=1}^{m} A_i = \{0,1\}^n$
3. for all points $a \in A_i$ and $b \in A_j$, $f(a) < f(b)$ if $i < j$

We require A_m to contain *only* optimal search points

Procedure: for each *level* A_i, bound the probability of leaving a level A_i for a higher level A_j, $j > i$.

Fitness levels



Law of total probability:
$E(X) = \sum_F \Pr(F) E(X|F)$

- $p(A_i)$ be the probability that a random chosen point belongs to A_i
- s_i be the probability to leave level A_i for level A_j with $j > i$

$$E(T) \leq \sum_{i=1}^{m-1} p(A_i) \cdot \left(\frac{1}{s_i} + \dots + \frac{1}{s_{m-1}}\right) \leq \left(\frac{1}{s_1} + \dots + \frac{1}{s_{m-1}}\right) = \sum_{i=1}^{m-1} \frac{1}{s_i}$$

Figure adapted from D. Sudholt, Tutorial 2011

Runtime analysis – (1+1) EA on OneMax

Theorem

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

Proof

We partition $\{0,1\}^n$ into disjoint sets $A_0, A_1, \dots, A_n$ where x is in A_i if and only if it has i zeros ($n - i$ ones).

To escape A_i, it suffices to flip a single zero and leave all other bits unchanged.

Thus, $s_i \geq \frac{i}{n}\left(1 - \frac{1}{n}\right)^{n-1} \geq \frac{i}{en}$, and $\frac{1}{s_i} \leq \frac{en}{i}$.

We conclude

$$E(T) \leq \sum_{i=1}^{m-1} \frac{1}{s_i} \leq \sum_{i=1}^{n} \frac{en}{i} = en \cdot H_n = O(n \log n).$$

■

Runtime analysis – (1+1) EA on OneMax

This gives only an upper bound. Maybe the (1+1) EA can be much quicker. For example it could be $O(n)$ or even something like $O(n \log \log n)$.

Runtime analysis – (1+1) EA on OneMax

Theorem (Droste, Jansen, Wegener 2002)

The expected runtime of the (1+1) EA on OneMax is $\Omega(n \log n)$.

Lemma

The probability that the (1+1) EA needs at least $(n-1)\ln n$ steps is at least a constant c.

Runtime analysis – (1+1) EA on OneMax

Proof of Lemma.

The initial solution has **at most** $n/2$ one bits with probability **at least** $1/2$.

There is a constant probability that in $(n-1)\ln n$ steps one of the remaining zero bits does not flip:

- Probability a particular bit doesn't flip in t steps: $(1-1/n)^t$
- Probability it flips **at least once** in t steps: $1-(1-1/n)^t$
- Probability $n/2$ bits flip at least once in t steps: $(1-(1-1/n)^t)^{n/2}$
- Probability at least one of the $n/2$ bits does not flip in t steps: $1-[1-(1-1/n)^t]^{n/2}$.

Set $t=(n-1)\ln n$. Then

$$\begin{aligned}1-[1-(1-1/n)^t]^{n/2} &= 1-[1-(1-1/n)^{(n-1)\ln n}]^{n/2}\\ &\geq 1-[1-(1/e)^{\ln n}]^{n/2}\\ &= 1-[1-1/n]^{n/2}\\ &= 1-[1-1/n]^{n\cdot 1/2} \geq (1-(2e))^{-1/2} = c.\end{aligned}$$

■

Runtime analysis – (1+1) EA on OneMax

Theorem (Droste, Jansen, Wegener 2002)

The expected runtime of the (1+1) EA on OneMax is $\Omega(n \log n)$.

Proof

Expected runtime:

$$\begin{aligned}E(T) = \sum_{t=1}^{\infty} t \Pr(T=t) &\geq (n-1)\ln n \cdot \Pr(T \geq (n-1)\ln n)\\ &\geq (n-1)\ln n \cdot c = \Omega(n \log n).\end{aligned}$$

by previous lemma

■

Upper bound given by fitness levels is tight.

Fitness levels

There are several more advanced results that use the fitness levels technique:

Expected runtime of the $(1+\lambda)$ EA on $\textsc{LeadingOnes}$ is $O(\lambda n + n^2)$ (Jansen et al., 2005)

Expected runtime of the $(\mu+1)$ EA on $\textsc{LeadingOnes}$ is $O(\mu n \log n + n^2)$ (Witt, 2006)

Fitness levels for proving lower bounds (Sudholt, 2010).

Non-elitist populations (Lehre, 2011).