I. Introduction

Why did Benjamin move to *Design and Analysis of Algorithms* to *Heuristics*?

- huge **importance in practice**
- and **very little theoretical work so far**
  - large need
  - many fundamental questions are wide open
  - room for cool results
- some problems **cannot be solved with classic algorithms**

Example (Open questions)

- Do we need *crossover* in EA?
- Is *non-elitism* (forgetting the best-so-far) useful?
- How do I set *parameters* of my favorite heuristic?

II. Runtime Analysis of Simple Heuristics

Definition (Runtime analysis)

Runtime analysis: quantify the time T before the first optimal solution is found.

Reminder from lecture 2 (upper bound for the RLS expected runtime on `OneMax` using fitness-level argument, $(1+1)$-EA on `OneMax` using only 1-bit flips).

Definition ($(1+\lambda)$-EA)

We now still have only one parent, but λ independent offsprings, and we pick the best one (if better than current parent).

Task. How long does it take to find the optimum, and what is the role of λ?

Clever proof. Fitness-level method with expectations.

- $\texttt{OneMax}(x) = n - d$
- $\mathbb{P}(\texttt{OneMax}(y_i) > \texttt{OneMax}(x)) \geq \frac{d}{ne}$
- $\mathbb{E}(T_d) \leq \frac{en}{d} + \lambda$ where T_d is the time to get a better offspring
- $\mathbb{E}(T) = \sum E(T_d) = en \ln n + n\lambda + O(n)$

Proof. Classic fitness-level method.

- $p_d \geq 1 - \left(1 - \frac{d}{en}\right)^\lambda$
- see the slides for tedious details (depends on the regime).

III. Drift Analysis, Linear Functions

Definition ((Pseudo-Boolean) Linear function)

Let $a_1, ..., a_n \in \mathbb{R}$. Then:

$$f : \begin{array}{l} \{0,1\}^n \to \mathbb{R} \\ x \mapsto \sum_{i=1}^{n} a_i x_i \end{array}$$

is called a *(pseudo-boolean) linear function.*

Usual assumptions:
- all weights a_i are different from zero
- all weights a_i are positive

Question. In how many iterations (in expectation) do RLS and $(1 + 1)$-EA optimize such a linear function?

⚠️ The previous proof does not hold anymore. We needed higher fitness $\Rightarrow$ more 1's, which is not valid anymore. For example, think of `BinaryValue`: `BinaryValue`$((0, 1, ...1)) = 2^{n-1} - 1 <$ `BinaryValue`$((1, 0, ..., 0)) = 2^{n-1}$.

This is known as ***low fitness-distance correlation***: the fitness of a search point does not indicate well how close the search point is to the optimum.

Theorem (Multiplicative drift)

Let $X_0, X_1, ...$ be a sequence of random variables taking values in the set $\{0\} \cup [x_{\text{min}}, \infty)$ $(x_{\text{min}} > 0)$.
Let $\delta > 0$. Assume that for all $t \in \mathbb{N}$ and $x \in \mathbb{R}_+$, we have

$$\mathbb{E}[X_{t+1} \mid X_t = x] \leq (1 - \delta)x$$

Let $T := \min\{t \in \mathbb{N} \mid X_t = 0\}$. Then:

$$\mathbb{E}[T \mid X_0 = x] \leq \frac{1 + \ln \frac{x}{x_{\text{min}}}}{\delta}$$

Prop (Corollary of the multiplicative drift)

The $(1 + 1)$-EA optimizes any linear function f with weights in $[a_{\text{min}}, a_{\text{max}}]$ in expected time at most $en\left(1 + \ln \frac{a_{\text{max}} n}{a_{\text{min}}}\right)$.

IV. Metropolis Algorithm

You always take the newly generated solution if it's better. If it's worse, it is picked with probability $\alpha^{f(y) - f(x^{t-1})}$, where α is a given parameter.

Task. Analyze the runtime of the Metropolis algorithm on the `OneMax` problem for reasonable values of α.

- $\mathbb{P}(-1) \leq \frac{1}{\alpha}$
- $\mathbb{P}(+1) = \frac{d}{n} \geq \frac{1}{n}$
- Hopefully we will have: $\mathbb{E}[\Delta f] \geq \frac{1}{n} - \frac{1}{\alpha} \geq \frac{1}{2}n$ (aiming for a multiplicative drift)