CS4618 Artificial Intelligence I

Today: Parametrisation: Mutation Probability

Thomas Jansen

December 7th

Plans for Today

Setting the Mutation Probability

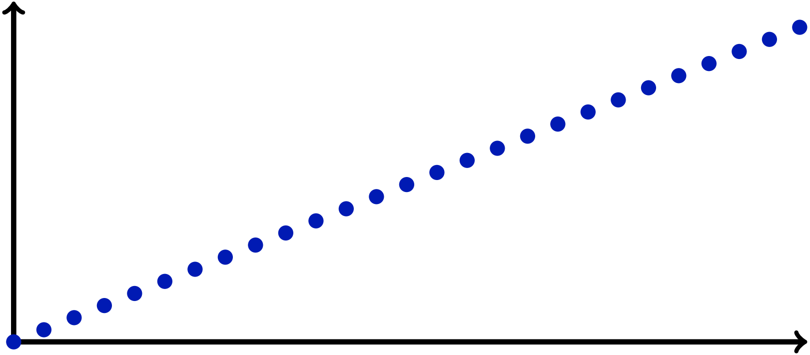
Remember

- $p_m = 1/n$ is by far most recommended choice
- NFL suggests that $p_m = 1/n$ cannot be always good
- silly example ($\textsc{OneMax}$ with function value $n+1$ for 0^n) proves that other mutation probabilities can reduce the expected optimisation time by an exponential factor (for the example from $\Theta(n^n)$ with $p_m = 1/n$ to $\Theta(2^n)$ with $p_m = 1/2$)

Goal for today develop a more detailed understanding of the mutation probability's role using $\textsc{OneMax}$ has a simple and typical example funct as a starting point

OneMax

$$\textsc{OneMax}(x) = \sum_{i=1}^{n} x[i]$$



Goal derive simple upper and lower bounds on $\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right)$ depending on p_m

A Simple Upper Bound

Idea consider only 1-bit mutations

Observe

$$\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) \leq \sum_{i=0}^{n-1} \frac{1}{(n-i)p_m} \cdot (1-p_m)^{-(n-1)}$$

$$= \frac{1}{p_m}(1-p_m)^{-(n-1)} \cdot \sum_{i=1}^{n} \frac{1}{i}$$

$$= \frac{(1-p_m)^{-p_m^{-1} \cdot p_m \cdot (n-1)}}{p_m} \cdot H_n = \Theta\left(\frac{e^{p_m \cdot n}}{p_m} \log n\right)$$

Remember $p_m = \frac{1}{n}$ usual choice

For comparison $p_m = \alpha(n) \cdot \frac{1}{n}$

Conclusion

- $\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = O\left(\frac{e^{\alpha(n)}}{\alpha(n)} \cdot n \log n\right)$
- $\left(\lim\limits_{n\to\infty} \alpha(n) = 0\right)$
  $\Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = O\left(\frac{1}{\alpha(n)} \cdot n \log n\right)$
- $\left(\lim\limits_{n\to\infty} 1/\alpha(n) = 0\right)$
  $\Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = O\left(e^{\alpha(n) - \log \alpha(n)} \cdot n \log n\right)$

A Very Simple Lower Bound

Idea consider only the very last step

Observe
$$\begin{aligned}&\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right)\\ &\geq \min\left\{\left(\frac{1}{p_m}\right)^i (1-p_m)^{-(n-i)} \mid i \in \{1,2,\ldots,n\}\right\}\\ &= \Theta\left(\min\left\{\left(\frac{1}{p_m}\right)^i e^{p_m(n-i)} \mid i \in \{1,2,\ldots,n\}\right\}\right)\end{aligned}$$

For comparison $p_m = \alpha(n) \cdot \frac{1}{n}$

Conclusion

- $\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = \Omega\left(\min\limits_i \left(\frac{n}{\alpha(n)}\right)^i e^{\alpha(n)\cdot(n-i)/n}\right)$
- $(i=1) \Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = \Omega\left(\frac{n}{\alpha(n)} e^{\alpha(n)}\right)$
- $(i=n) \Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = \Omega\left(\left(\frac{n}{\alpha(n)}\right)^n\right)$

Upper and Lower Bounds for OneMax

Upper Bounds

- $\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = O\left(\frac{e^{\alpha(n)}}{\alpha(n)} \cdot n \log n\right)$
- $\left(\lim\limits_{n\to\infty} \alpha(n) = 0\right) \Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = O\left(\frac{1}{\alpha(n)} \cdot n \log n\right)$
- $\left(\lim\limits_{n\to\infty} 1/\alpha(n) = 0\right) \Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = O\left(e^{\alpha(n)-\log\alpha(n)} \cdot n \log n\right)$

Lower Bounds

- $\mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = \Omega\left(\min\limits_{i}\left(\frac{n}{\alpha(n)}\right)^{i} e^{\alpha(n)\cdot(n-i)/n}\right)$
- $(i=1) \Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = \Omega\left(\frac{n}{\alpha(n)} e^{\alpha(n)}\right)$
- $(i=n) \Rightarrow \mathrm{E}\left(T_{\text{(1+1) EA},\textsc{OneMax}}\right) = \Omega\left(\left(\frac{n}{\alpha(n)}\right)^{n}\right)$

Observation

- $p_m = \Theta(1/n)$ optimal
- much smaller p_m hurt linearly
- much larger p_m hurt exponentially

Lessons So Far

- $p_m = 1/n$ is useful since 1-bit mutations matter (and are sufficient to get around – see simulated annealing)
- smaller mutation probabilities hurt a bit
- larger mutation probabilities hurt a lot
- with $p_m = \omega(\log(n)/n)$ we cannot optimise $\textsc{OneMax}$-like functions efficiently
- larger mutation probabilities are useful for 'jumping around'

Plans and Ideas

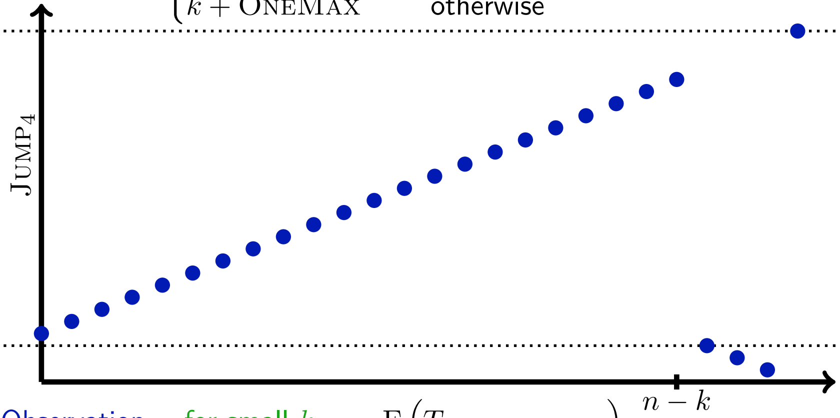
- find out for what kind of jumps what p_m is useful
- consider a function where a 'large jump' is needed

Observe $\left(\frac{k}{n}\right)^d \cdot \left(1 - \frac{k}{n}\right)^{n-d}$ maximal for $k = d$

$\rightsquigarrow$ only jumps of size $O(\log n)$ feasible

A First Example: $\text{J}\textsc{ump}_k$

$$\text{J}\textsc{ump}_k(x) = \begin{cases} n - \text{O}\textsc{ne}\text{M}\textsc{ax}(x) & \text{if } n-k < \text{O}\textsc{ne}\text{M}\textsc{ax}(x) < n \\ k + \text{O}\textsc{ne}\text{M}\textsc{ax} & \text{otherwise} \end{cases}$$



Observation for small k

$$\begin{aligned} & \mathrm{E}\left(T_{\text{(1+1) EA},\text{J}\textsc{ump}_k}\right) \\ &= \mathrm{E}\left(T_{\text{(1+1) EA},\text{O}\textsc{ne}\text{M}\textsc{ax}}\right) + \mathrm{E}\left(\text{time for } k\text{-bit jump}\right) \\ &= O\Big(e^{\alpha(n)-\log\alpha(n)} \cdot n\log n\Big) + \Theta\Big((n/\alpha(n))^k\Big) \end{aligned}$$

Setting p_m for $\textsc{Jump}_k$

$$\textsc{Jump}_k(x) = \begin{cases} n - \textsc{OneMax}(x) & \text{if } n-k < \textsc{OneMax}(x) < n \\ k + \textsc{OneMax} & \text{otherwise} \end{cases}$$

Observation for small k $\quad \mathrm{E}\left(T_{(1+1)\ \mathrm{EA},\textsc{Jump}_k}\right)$

$$= \mathrm{E}\left(T_{(1+1)\ \mathrm{EA},\textsc{OneMax}}\right) + \mathrm{E}\left(\text{time for } k\text{-bit jump}\right)$$

$$= O\left(e^{\alpha(n)-\log\alpha(n)} \cdot n\log n\right) + \Theta\left((n/\alpha(n))^k\right)$$

Observation with standard choice $p_m = 1/n$

$$\mathrm{E}\left(T_{(1+1)\ \mathrm{EA},\textsc{Jump}_k}\right) = \Theta\left(n^k\right) \text{ for small } k > 1$$

polynomial $\Leftrightarrow k = O(1)$

Observation with optimal choice $p_m = k/n$

$$\mathrm{E}\left(T_{(1+1)\ \mathrm{EA},\textsc{Jump}_k}\right) = O\left(e^{k-\log k} \cdot n\log n\right) + \Theta\left((n/k)^k\right)$$

polynomial $\Leftrightarrow k = O(1)$

Looking Back on JUMP_k

Observation with standard choice $p_m = 1/n$
$\mathrm{E}\left(T_{\text{(1+1) EA},\text{JUMP}_k}\right) = \Theta\left(n^k\right)$ for small $k > 1$
polynomial $\Leftrightarrow k = O(1)$

Observation with optimal choice $p_m = k/n$
$\mathrm{E}\left(T_{\text{(1+1) EA},\text{JUMP}_k}\right) = O\left(e^{k-\log k} \cdot n \log n\right) + \Theta\left((n/k)^k\right)$
polynomial $\Leftrightarrow k = O(1)$

Observation no significant gain for larger p_m

Where is the problem?

Observation 'target' very small

Idea larger p_m advantageous for hitting larger targets in some distance

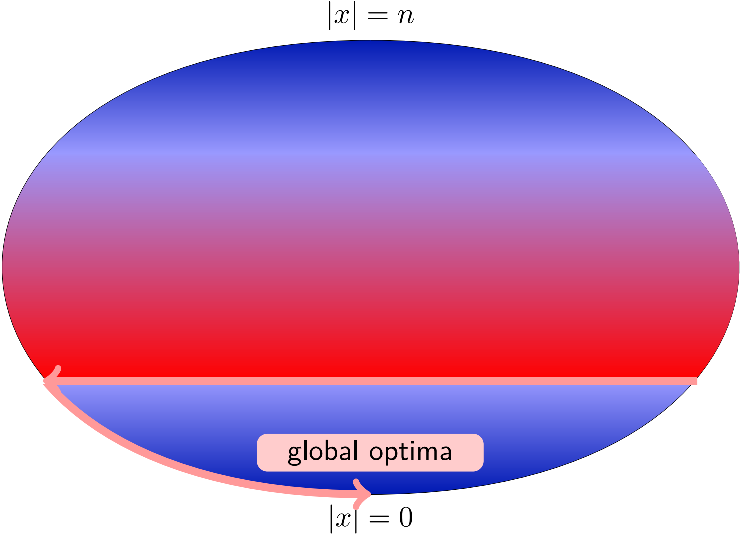
A More Extreme Example

$$f(x) = \begin{cases} n - |x| & \text{if } x \in A \\ (3/4)n + \sum_{i=1}^{n/4} x[i] & \text{if } x \in B \\ 2n - i & \text{if } x \in C \text{ and } x = 1^i 0^{n-i} \\ 2n + 1 & \text{if } x \in D \\ \min\{|x|, n - |x|\} & \text{if } x \in E \end{cases}$$

with

$$\begin{aligned} A &= \{x \in \{0,1\}^n \mid n/4 < |x| < (3/4)n\} \\ B &= \{x \in \{0,1\}^n \mid |x| = n/4\} \\ C &= \{x = 1^i 0^{n-i} \mid i \in \{0, 1, \ldots, (n/4) - 1\}\} \\ D &= \left\{x \in \{0,1\}^n \mid (|x| = \log n) \wedge \left(\sum_{i=1}^{2\log n} x[i] = 0\right)\right\} \\ E &= \{0,1\}^n \setminus (A \cup B \cup C \cup D) \end{aligned}$$

The Example Function f_2



Mutation Probabilities for f_2

Theorem

$$\mathrm{E}\left(T_{\text{(1+1) EA},f_2}\right) = n^{O(1)} \Leftrightarrow p_m = \Theta\left(\frac{\log n}{n}\right)$$

$$\mathrm{E}\left(T_{\text{(1+1) EA},f_2}\right) = O\left(\frac{n^{2+c}}{\log n} + n^{c-\log(c)-\log\ln 2}\right) \text{ with } p_m = \frac{c\log n}{n}$$

Proof Ideas

- for $p_m = o(\log(n)/n)$ time for final jump super-polynomial
- for $p_m = \omega(\log(n)/n)$ time on ridge super-polynomial
- $\mathrm{E}\left(\text{time for final jump with } p_m = c\log(n)/n\right)$
  $$= \left(\binom{n-2\log n}{\log n}\left(\frac{c\log n}{n}\right)^{\log n}\left(1-\frac{c\log n}{n}\right)^{n-\log n}\right)^{-1}$$
- calculating fearlessly

Remark $\mathrm{E}\left(T_{\text{(1+1) EA},f_2}\right) = O\left(n^{2.361}\right)$ for $c = 0.361$

But how can we find such a strange good p_m in practice?

A Simple Mutation Schedule

Idea get rid of the choice of p_m
by systematically trying all sensible values
'probably good' values more often,
'probably too large' values less often

Dynamic (1+1) EA

```
1. $p_m := 1/n$; $t := 1$; Choose $x_t \in \{0,1\}^n$ uniformly at random.
2. Repeat
3.     $y :=$ standard bit mutation$(x_t)$ with mutation prob. $p_m$
4.     $t := t+1$; If $f(y) \geq f(x_{t-1})$ then $x_t := y$ else $x_t := x_{t-1}$.
5.     $p_m := 2p_m$; If $p_m > 1/2$ then $p_m := 1/n$
6. Until 'decide to stop'
7. Output $x_t$.
```

About the Dynamic (1+1) EA

Observations

- tries $\log n$ different mutation probabilities
- $p_m \in \left\{\frac{1}{n}, \frac{2}{n}, \frac{4}{n}, \dots, \frac{2^{\lfloor \log n \rfloor - 1}}{n}\right\}$
- $\forall p \in [1/n, 1/2] \colon \exists p_m \in [p, 2p]$

Results

- on many functions (in particular, on simple functions) slower by a factor $\Theta(\log n)$
- $\mathrm{E}\left(T_{\text{Dynamic (1+1) EA},f_2}\right) = O(n^2 \log n)$
- $\mathrm{E}\left(T_{\text{Dynamic (1+1) EA},f_1}\right) < 4^n \log n$
- $\exists f \colon \mathrm{E}\left(T_{\text{Dynamic (1+1) EA},f}\right)$ exponential and $\mathrm{E}\left(T_{\text{(1+1) EA } (p_m = 1/n),f}\right)$ polynomial

Summary & Take Home Message

Things to remember

- (1+λ) EA on OneMax
- super-linear speed-up by parallelisation of EAs
- ideas and example functions
- mutation probability crucial
- fixed mutation probability $p_m = 1/n$ often useful
- larger mutation probabilities more dangerous than small one
- simple dynamic choices exist

Take Home Message

- Linear speed-ups only work up to some cut off point that depends on the objective function.
- More extreme positive and negative effects can occur.
- Be careful with large mutation probabilities.
- Deviate from guidelines only when you have reasons to.
- Dynamic (1+1) EA is robust choice.