ucc
Coláiste na hOllscoile Corcaigh, Éire
University College Cork, Ireland

CS4618 Artificial Intelligence I

Today: Crossover
Parametrisation:
Offspring Population Size

Thomas Jansen

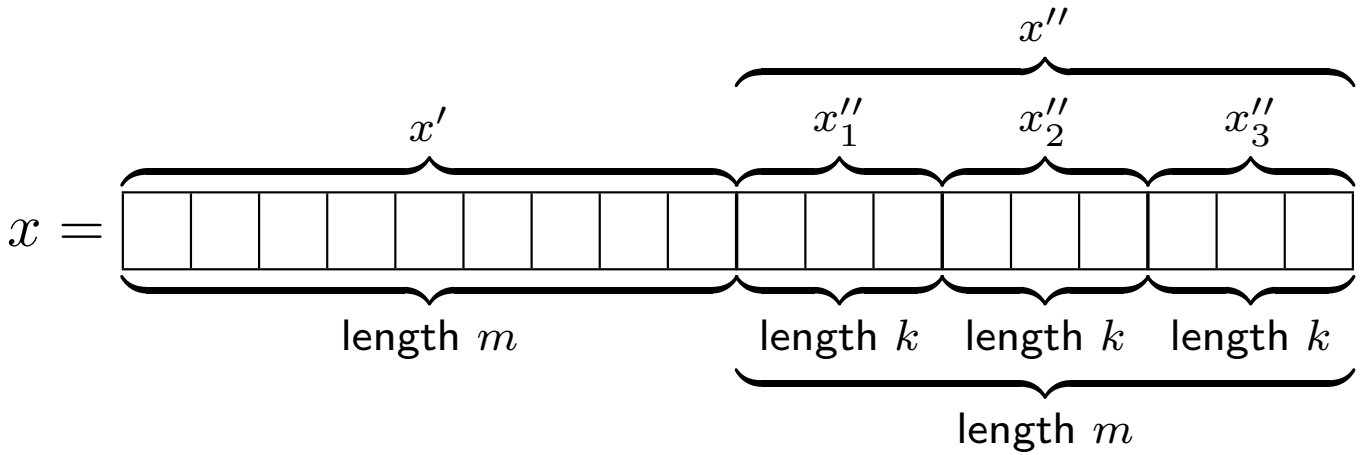
November 28th

Plans for Today

1. Uniform Crossover
   - Idea and Example Function
   - Result
2. Parametrisation
   - Motivation
3. LeadingOnes
   - Function and Intuition
   - Result and Proof
4. Summary
   - Summary & Take Home Message

Preparing an Example Function for Uniform Crossover

Notation partition bit string $x = x'x'' = x'x''_1x''_2x''_3$



$$n = 2m = m + m = m + 3k = m + k + k + k$$

Definition circle $C = \{1^i0^{m-i}, 0^i1^{m-i} \mid i \in \{1, 2, \ldots, m\}\}$

target $T = \{x''_1x''_2x''_3 \mid |x''_1| = |x''_2| = |x''_3| = \lfloor k/2 \rfloor\}$

$\mathsf{H}(x, A) = \min\{\mathsf{H}(x, y) \mid y \in A\}$

Example Function for Uniform Crossover

Definition circle $C = \{1^i 0^{m-i}, 0^i 1^{m-i} \mid i \in \{1, 2, \ldots, m\}\}$
target $T = \{x_1'' x_2'' x_3'' \mid |x_1''| = |x_2''| = |x_3''| = \lfloor k/2 \rfloor\}$
$\mathsf{H}(x, A) = \min\{\mathsf{H}(x, y) \mid y \in A\}$

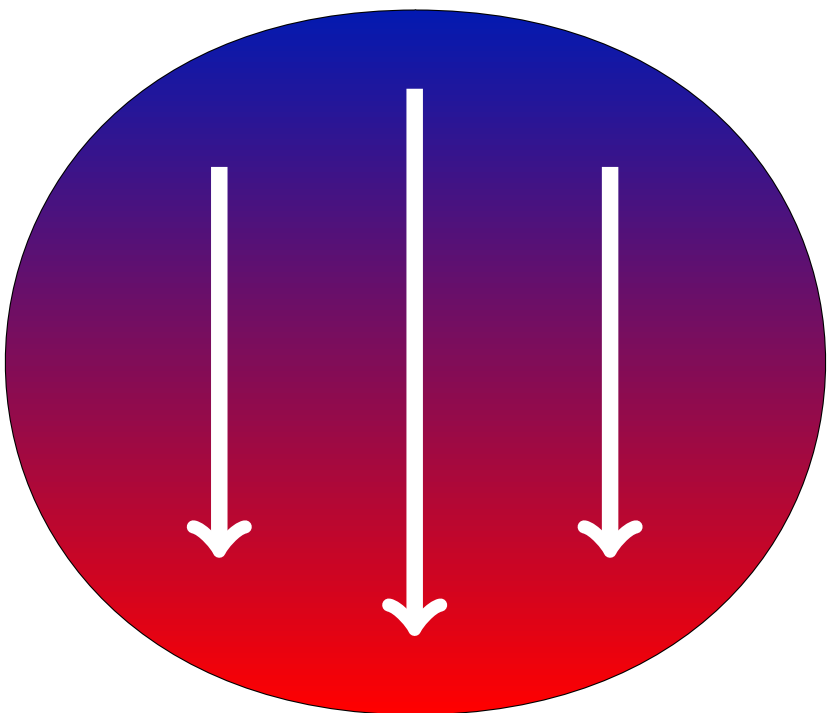
Definition $f_2 \colon \{0,1\}^n \to \mathbb{N}_0$ with

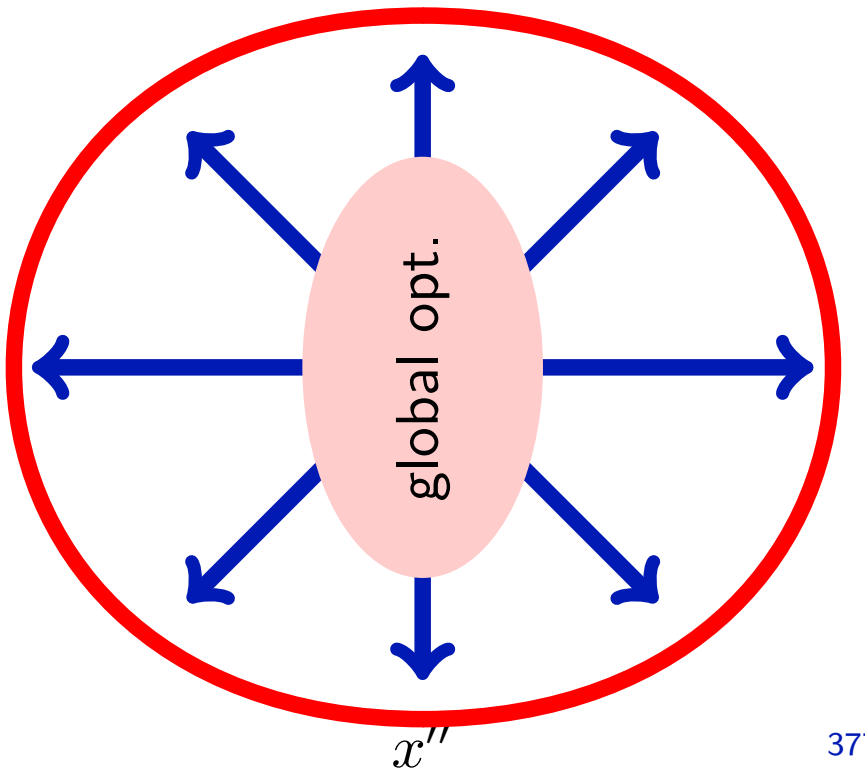
$$f_2(x) = \begin{cases} n - \mathsf{H}(x'', C) & \text{if } x' \neq 0^m \text{ and } x'' \notin C \\ 2n - \mathsf{H}(x', 0^m) & \text{if } x'' \in C \\ 0 & \text{if } x' = 0^m \text{ and } x'' \notin (C \cup T) \\ 3n & \text{if } x' = 0^m \text{ and } x'' \in T \end{cases}$$

Example Function for Uniform Crossover

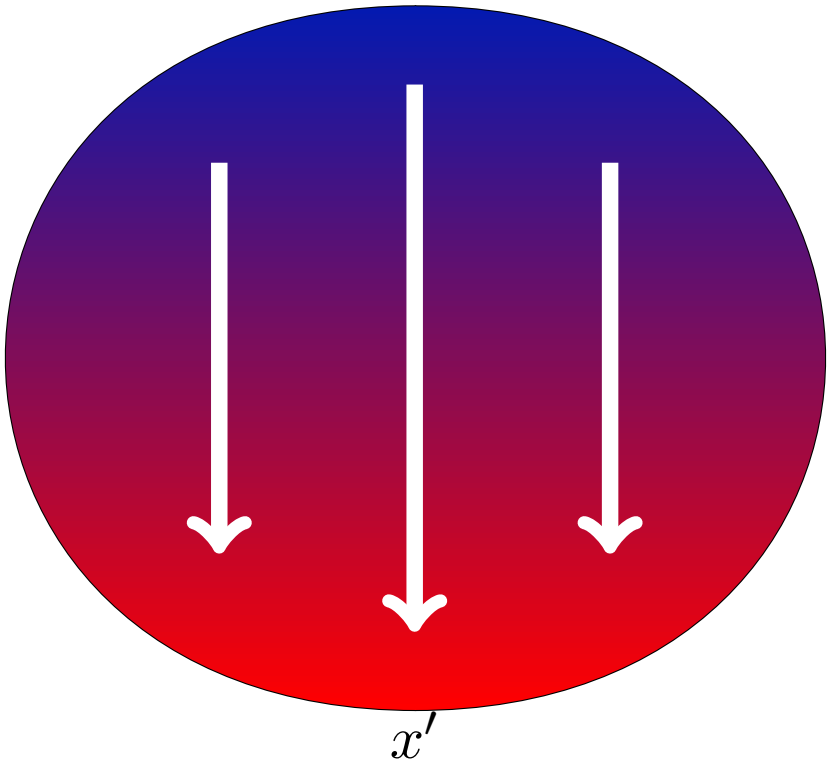
Definition $f_2 : \{0,1\}^n \to \mathbb{N}_0$ with

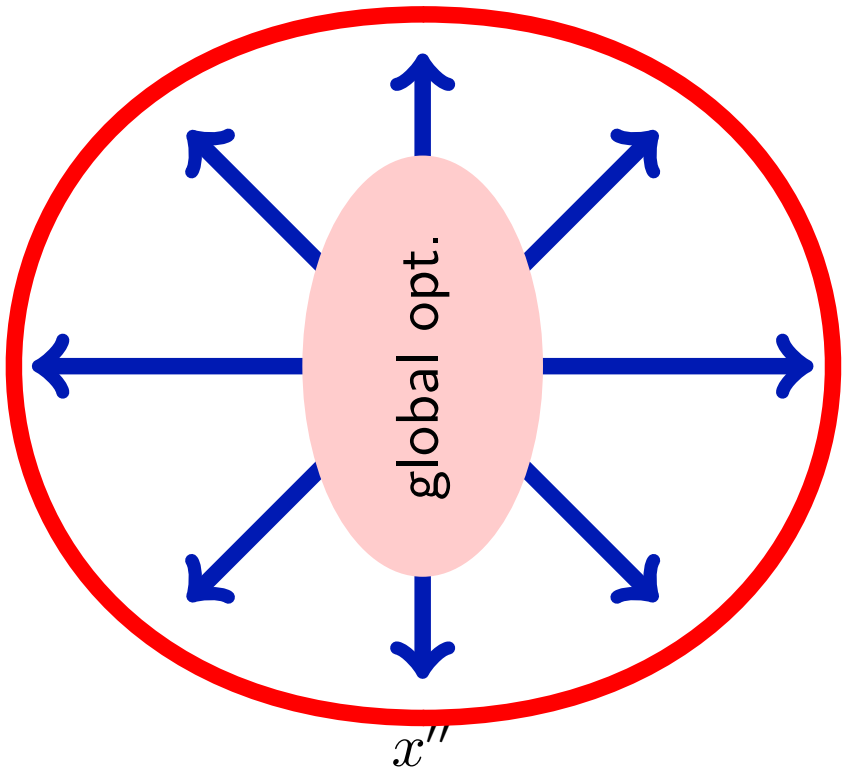
$$f_2(x) = \begin{cases} n - \mathsf{H}(x'', C) & \text{if } x' \neq 0^m \text{ and } x'' \notin C \\ 2n - \mathsf{H}(x', 0^m) & \text{if } x'' \in C \\ 0 & \text{if } x' = 0^m \text{ and } x'' \notin (C \cup T) \\ 3n & \text{if } x' = 0^m \text{ and } x'' \in T \end{cases}$$





Without Crossover on f_2

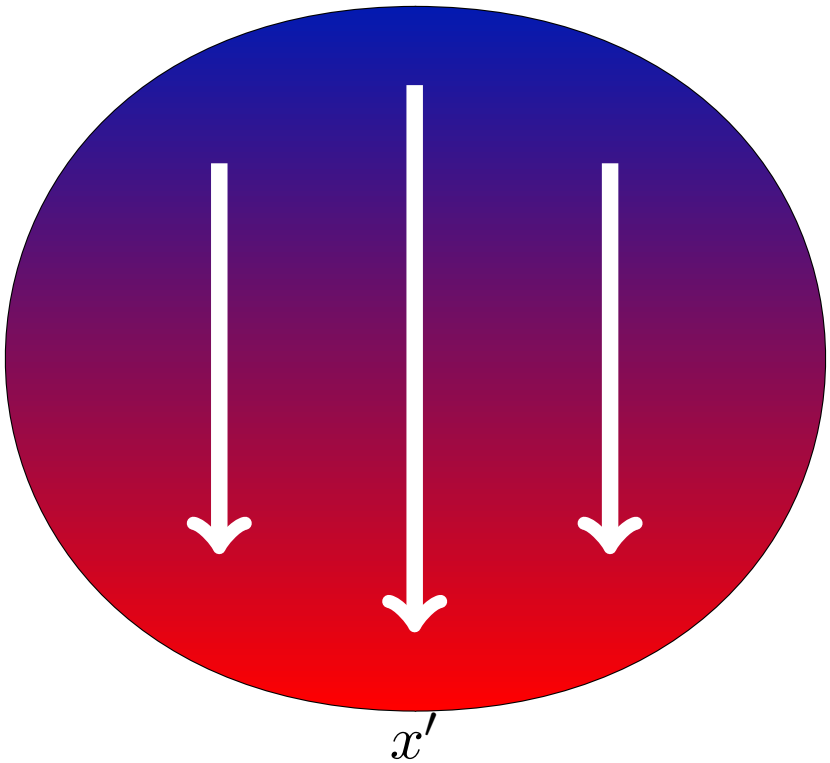


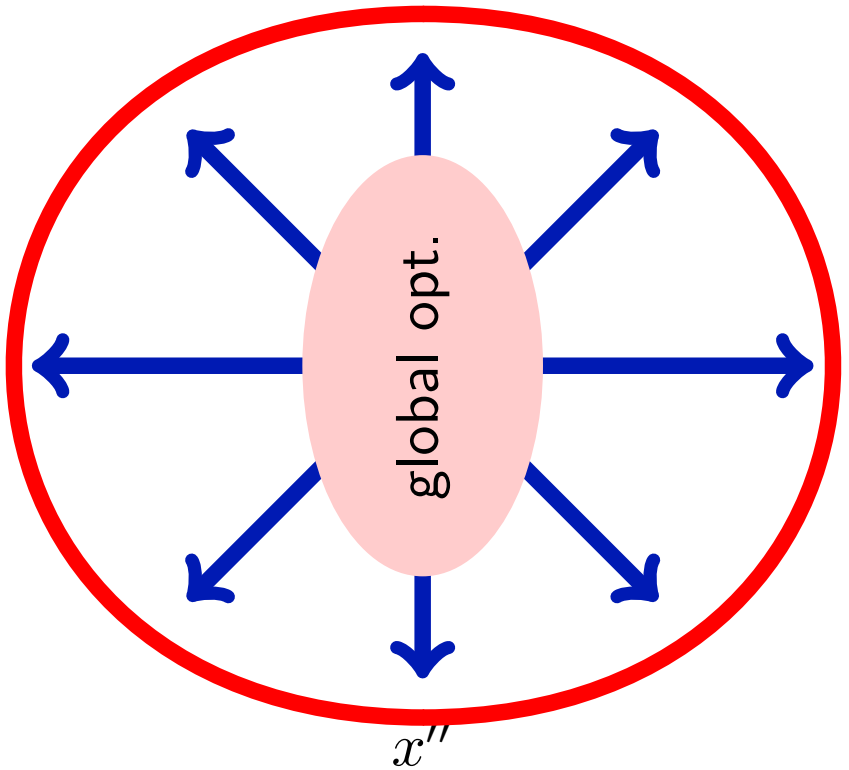


Observation

- quickly $x' = 0^m$ and $x'' \in C$
- gap too large for a mutation
- $\Rightarrow$ exponentially long with overwhelming probability

With Uniform Crossover on f_1





Theorem

Consider the steady state GA with crossover probability $p_c \leq 1 - \varepsilon$ (constant $\varepsilon > 0$) and population size $\mu \geq n$.
$\mathrm{E}(T_{\mathsf{GA},f_2}) = O(\mu n^2 + \mu^2 \log(n)/p_c)$

Optimising f_2 with Uniform Crossover

Theorem

Consider the steady state GA with crossover probability $p_c \leq 1 - \varepsilon$ (constant $\varepsilon > 0$) and population size $\mu \geq n$.
$\mathrm{E}\left(T_{\mathsf{GA},f_2}\right) = O\Big(\mu n^2 + \mu^2 n^{3/2}/p_c\Big)$

Proof

Observations

- to $x' = 0^m$ and $x'' \in C$ with all possible points on circle in expecation in $O(\mu n^2)$ steps similar to f_1
  (mutation suffices, no crossover with prob. $\geq \varepsilon = \Omega(1)$)
- probability to create the optimum
  $\geq \mathrm{Prob}\left(\text{pick good parents, crossover and generate opt.}\right) \geq$
  $(1/\mu^2) \cdot p_c \cdot \mathrm{Prob}\left(\text{generate optimum from } 1^0 0^{m-i} \text{ and } 0^i 1^{m-i}\right)$
  $= \Omega\Big((1/\mu^2) \cdot p_c \cdot \Big(1/\sqrt{k}\Big)^3\Big) = \Omega\Big((1/\mu^2) \cdot p_c \cdot 1/n^{3/2}\Big)$
- $\Rightarrow \mathrm{E}\left(T_{\mathsf{GA},f_2}\right) = O(\mu n^2 + \mu^2 \log(n)/p_c)$

Evolutionary Algorithms

Remember

Evolutionary Algorithm

1. Choose $x_1, x_2, \ldots, x_\mu \in S$ uniformly at random.
2. Repeat
3. For $i \in \{1, 2, \ldots, \lambda\}$ do
4. With probability p_c select $z_1, z_2 \in \{x_1, x_2, \ldots, x_\mu\}$.
   $z :=$ crossover(z_1, z_2)
   Else select $z \in \{x_1, x_2, \ldots, x_\mu\}$.
5. $y_i :=$ mutation(z)
6. Select new $x_1, x_2, \ldots, x_\mu$ out of old $x_1, x_2, \ldots, x_\mu$ and $y_1, y_2, \ldots, y_\lambda$.
7. Until 'decide to stop'
8. Output x_i with $f(x_i) = \max\{f(x_j) \mid 1 \leq j \leq \mu\}$.

Design Decisions

Evolutionary Algorithm

1. Choose $x_1, x_2, \ldots, x_\mu \in S$ uniformly at random.
2. Repeat
3. For $i \in \{1, 2, \ldots, \lambda\}$ do
4. With probability p_c select $z_1, z_2 \in \{x_1, x_2, \ldots, x_\mu\}$.
   $z :=$ crossover(z_1, z_2)
   Else select $z \in \{x_1, x_2, \ldots, x_\mu\}$.
5. $y_i :=$ mutation(z)
6. Select new $x_1, x_2, \ldots, x_\mu$ out of old $x_1, x_2, \ldots, x_\mu$ and $y_1, y_2, \ldots, y_\lambda$.
7. Until 'decide to stop'
8. Output x_i with $f(x_i) = \max\{f(x_j) \mid 1 \leq j \leq \mu\}$.

Design Choices

- selection for reproduction
- mutation operator
- crossover operator
- selection for replacement

Parameters

- population size $\mu \in \mathbb{N}$
- offspring population size $\lambda \in \mathbb{N}$
- crossover probability $p_c \in [0, 1]$

Facts and Goal for Today

Known

- all choices potentially crucial for success
- no universally good choice (clear due to NFL)
- good choices sometimes hard to make
- useful guidelines exist

Goals

- understand guidelines
- learn to develop guidelines
- understand limitations of guidelines

Today offspring population size λ

Method embed in minimal algorithmic framework
consider effects on paradigmatic example functions

The (1+λ) EA

(1+λ) EA

1. $t := 0$; Choose $x_t \in \{0,1\}^n$ uniformly at random.
2. Repeat
3. For $i \in \{1, 2, \ldots, \lambda\}$ do
4. $y_i :=$ standard bit mutation(x_t) with $p_m = 1/n$
5. $m := \max\{f(y_1), f(y_2), \ldots, f(y_\lambda)\}$
6. If $m \geq f(x)$ Then
   Select $y \in \{y' \in \{y_1, y_2, \ldots, y_\lambda\} \mid f(y') = m\}$ u. a. r.
   $x_{t+1} := y$
   Else $x_{t+1} := x_t$
7. $t := t + 1$
8. Until 'decide to stop'
9. Output x.

Only Parameter offspring population size λ

Measuring Performance

Remember $T_{(1+\lambda)\ \mathrm{EA},f} = \#f$ evaluations until optimum found

$$= \underbrace{1}_{x_0} + \underbrace{\lambda}_{y_1,\ldots,y_\lambda} \cdot \underbrace{t}_{\#\text{generations}}$$

Define $G_{(1+\lambda)\ \mathrm{EA},f} = \min\{t \mid f(x_t) = \max\{f(x) \mid x \in \{0,1\}^n\}\}$
number of generations

Observation $T_{(1+\lambda)\ \mathrm{EA},f}$ much closer
to actual computational effort

Why should we care about $G_{(1+\lambda)\ \mathrm{EA},f}$?

Observation $G_{(1+\lambda)\ \mathrm{EA},f}$ measures parallel computation time
$\widehat{=}$ computation time on parallel computer
with $\geq \lambda$ processing units

A First Example

Define $\textsc{LeadingOnes}\colon \{0,1\}^n \to \{0,1,\ldots,n\}$ with

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

counts number of leading 1-bits from left to right

Remember all bits right to the leftmost 0-bit uniformly distributed

Because

- initially true
- bits never involved in selection
- standard bit mutations don't change the uniform distribution

Result

Theorem

$$T_{(1+\lambda)\text{ EA},\textsc{LeadingOnes}} = O\left(n^2 + \lambda \cdot \frac{n}{\log(1+\lambda/n)}\right) \text{ for any } \lambda = n^{O(1)}$$
$$G_{(1+\lambda)\text{ EA},\textsc{LeadingOnes}} = O\left(\frac{n^2}{\lambda} + \frac{n}{\log(1+\lambda/n)}\right) \text{ for any } \lambda = n^{O(1)}$$

Remark
$$T_{(1+\lambda)\text{ EA},\textsc{LeadingOnes}} = \Theta\left(n^2 + \lambda \cdot \frac{n}{\log(1+\lambda/n)}\right) \text{ for any } \lambda = n^{O(1}$$
$$G_{(1+\lambda)\text{ EA},\textsc{LeadingOnes}} = \Theta\left(\frac{n^2}{\lambda} + \frac{n}{\log(1+\lambda/n)}\right) \text{ for any } \lambda = n^{O(1)}$$

Useful Tool — Additive Drift Theorem

algorithm A, Z set of all populations,
$d\colon Z \to \mathbb{R}_0^+$ with $d(P) = 0 \Leftrightarrow P$ contains opt.,
$D_t = d(P_{t-1}) - d(P_t)$,
$\Delta^+ = \max\{\mathrm{E}\,(D_t \mid P_{t-1}, T_{A,f} \geq t)\}$,
$\Delta^- = \min\{\mathrm{E}\,(D_t \mid P_{t-1}, T_{A,f} \geq t)\}$.
$\Delta^+ > 0 \Rightarrow \mathrm{E}\,(T_{A,f}) \geq \mathrm{E}\,(d(P_0))\,/\Delta^+$
$\Delta^- > 0 \Rightarrow \mathrm{E}\,(T_{A,f}) \leq \mathrm{E}\,(d(P_0))\,/\Delta^-$

Drift Analysis of the (1+λ) EA on LeadingOnes

Set of Populations $Z = \{0,1\}^n$

Distance Measure $d(x) = n - \text{LeadingOnes}(x)$

E (Initial Distance)

$$\begin{aligned}
\mathrm{E}\left(d(x_0)\right) &= \sum_{i=0}^{n} i \cdot \mathrm{Prob}\left(d(x_0) = i\right) \\
&= \sum_{i=1}^{n} i \cdot 2^{-(n+1-i)} = \sum_{i=1}^{n} \sum_{j=i}^{n} 2^{-(n+1-j)} \\
&= 2^{-(n+1)} \sum_{i=1}^{n} \sum_{j=i}^{n} 2^j \\
&= 2^{-(n+1)} \sum_{i=1}^{n} 2^{n+1} - 1 - \left(2^i - 1\right) \\
&= 2^{-(n+1)} \left(n \cdot 2^{n+1} - \sum_{i=1}^{n} 2^i\right) \\
&= 2^{-(n+1)} \left(n \cdot 2^{n+1} - 2^{n+1} + 2\right) = n - 1 + 2^{-n}
\end{aligned}$$

Drift Analysis of the (1+λ) EA on LeadingOnes

Set of Populations $Z = \{0,1\}^n$

Distance Measure $d(x) = n - \text{LeadingOnes}(x)$

E (Initial Distance) $\mathrm{E}\left(d(x_0)\right) = n - 1 + 2^{-n}$

Drift

$$\begin{aligned}
\mathrm{E}\left(D_t \mid P_{t-1}, T_{A,f} \geq t\right) &= \mathrm{E}\left(d(x_{t-1}) - d(x_t) \mid P_{t-1}, T_{A,f} \geq t\right) \\
&= \sum_{i=1}^{d(x_{t-1})} i \cdot \mathrm{Prob}\left(d(x_t) = d(x_{t-1}) - i\right) \\
&\geq \sum_{i=1}^{d(x_{t-1})} i \cdot \left(1 - \left(1 - \tfrac{1}{n} \cdot 2^{-i} \cdot \left(1 - \tfrac{1}{n}\right)^{n-1}\right)^{\lambda}\right) \\
&\geq \sum_{i=1}^{d(x_{t-1})} i \cdot \left(1 - \left(1 - \tfrac{1}{en2^i}\right)^{\lambda}\right)
\end{aligned}$$

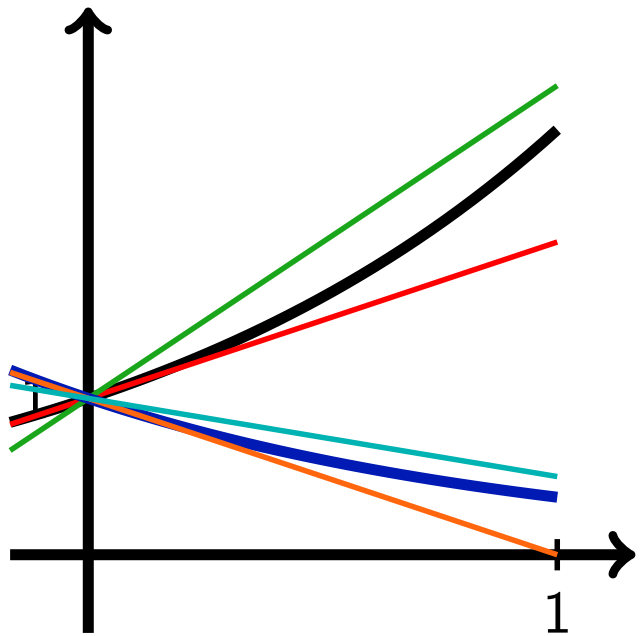
Interlude estimating $\left(1 - \frac{1}{en2^i}\right)^{\lambda}$

Bounding $\left(1-\frac{1}{en2^i}\right)^\lambda$

Remember $\left(1-\frac{1}{en2^i}\right)^\lambda = \left(1-\frac{1}{en2^i}\right)^{en2^i\cdot\lambda/(en2^i)}$
$\approx e^{-\lambda/(en2^i)}$

$\forall x \leq 1\colon 1+x \leq e^x \leq 1+2x$

$\forall x \in [0,1]\colon 1-x \leq e^{-x} \leq 1-x/2$



Thus $\left(1-\frac{1}{en2^i}\right)^\lambda \leq 1-\frac{\lambda}{2en2^i}$ for $\lambda/(en2^i) \leq 1$

Bounding the Drift

We have $\mathrm{E}\left(D_t \mid P_{t-1}, T_{A,f} \geq t\right) \geq \sum_{i=1}^{d(x_{t-1})} i \cdot \left(1 - \left(1 - \frac{1}{en2^i}\right)^\lambda\right)$

$\left(1 - \frac{1}{en2^i}\right)^\lambda \leq 1 - \frac{\lambda}{2en2^i}$ for $\lambda/(en2^i) \leq 1$

Case 1: $\lambda \leq n$ $\mathrm{E}\left(D_t \mid P_{t-1}, T_{A,f} \geq t\right) \geq \sum_{i=1}^{d(x_{t-1})} i \cdot \frac{\lambda}{2en2^i} \geq \frac{\lambda}{4en}$

Case 2: $\lambda > n$ for $i = \log(\lambda/n)$

$\left(1 - \frac{1}{en2^i}\right)^\lambda \leq e^{-\lambda/(en2^{\log(\lambda/n)})} = e^{-\lambda/(en\lambda/n)} = e^{-1/e}$

$\Rightarrow \mathrm{E}\left(D_t \mid P_{t-1}, T_{A,f} \geq t\right) \geq \log(\lambda/n) \cdot e^{-1/e}$

Combining Findings

Remember $\mathrm{E}\left(G_{(1+\lambda)\ \mathsf{EA},\textsc{LeadingOnes}}\right) \leq \mathrm{E}\left(d(x_0)\right)/\Delta^-$

$\mathrm{E}\left(d(x_0)\right) = n - 1 + 2^{-n}$

$$\Delta^- \geq \begin{cases} \lambda/(4en) & \text{if } \lambda \leq n \\ e^{-1/e}\log(\lambda/n) & \text{otherwise} \end{cases}$$

$$\Rightarrow \mathrm{E}\left(G_{(1+\lambda)\ \mathsf{EA},\textsc{LeadingOnes}}\right) \leq \frac{n-1+2^{-n}}{\lambda/(4en)} + \frac{n-1+2^{-n}}{e^{-1/e}\log(\lambda/n)}$$

$$= O\left(\frac{n^2}{\lambda} + \frac{n}{\log(\lambda/n)}\right)$$

Remember $\mathrm{E}\left(T_{(1+\lambda)\ \mathsf{EA},\textsc{LeadingOnes}}\right) = \mathrm{E}\left(G_{(1+\lambda)\ \mathsf{EA},\textsc{LeadingOnes}}\right) \cdot \lambda$

$$\Rightarrow \mathrm{E}\left(T_{(1+\lambda)\ \mathsf{EA},\textsc{LeadingOnes}}\right) = O\left(n^2 + \frac{n\lambda}{\log(\lambda/n)}\right)$$

Discussion

Theorem

$T_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = \Theta\left(n^2 + \lambda \cdot \frac{n}{\log(1+\lambda/n)}\right)$ for any $\lambda = n^{O(1)}$

$G_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = \Theta\left(\frac{n^2}{\lambda} + \frac{n}{\log(1+\lambda/n)}\right)$ for any $\lambda = n^{O(1)}$

Observations

- $G_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = \Omega(n/\log n)$ for all λ
- $T_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = \Omega(n^2)$ for all λ
- $T_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = O(n^2)$ for $\lambda = O(n)$
- $G_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = \Theta(n)$ $\lambda = \Theta(n)$
- $T_{(1+\lambda)\text{ EA},\text{LeadingOnes}} = \Theta(n^2)$ $\lambda = \Theta(n)$
- with $\lambda = \Theta(n)$ linear speed-up without increased workload

Observation $\text{Prob}\,(\text{single mutation is improving}) = \Theta(1/n)$ always
$\widehat{=}$ reciprocal of $\Theta(n)$

Summary & Take Home Message

Things to remember

- insights and design of appropriate example functions
- immense potential speed-up by crossover
- $(1+\lambda)$ EA
- generations vs. function evaluations
- success probability and offspring population size

Take Home Message

- Crossover can speed-up search dramatically.
- Problem structure needs to be appropriate for crossover to work.
- Encoding issues become more important when using crossover.
- Larger offspring population sizes allow for simple utilisation of parallel computing platforms.