Level-Based Analysis of Genetic Algorithms and Other Search Processes

Dogan Corus[1], Duc-Cuong Dang[1], Anton V. Eremeev[2], and Per Kristian Lehre[1]

[1] University of Nottingham, United Kingdom
[2] Omsk Branch of Sobolev Institute of Mathematics, Russia

Abstract. The fitness-level technique is a simple and old way to derive upper bounds for the expected runtime of simple *elitist* evolutionary algorithms (EAs). Recently, the technique has been adapted to deduce the runtime of algorithms with *non-elitist* populations and *unary* variation operators [2,8]. In this paper, we show that the restriction to unary variation operators can be removed. This gives rise to a much more general analytical tool which is applicable to a wide range of search processes. As introductory examples, we provide simple runtime analyses of many variants of the Genetic Algorithm on well-known benchmark functions, such as OneMax, LeadingOnes, and the sorting problem.

1 Introduction

The theoretical understanding of Evolutionary Algorithms (EAs) has advanced significantly. A contributing factor for this success may have been the strategy to analyse simple settings before proceeding to more complex scenarios, while at the same time developing appropriate analytical techniques.

The fitness-level technique is one of the oldest techniques for deriving upper bounds on the expected runtime of EAs. In this technique, the solution space is partitioned into disjoint subsets called *fitness-levels* according to ascending values of the fitness function. The expected runtime can be deduced from bounds on the probabilities of escaping the fitness levels. Applications of the technique is widely known in the literature for classical *elitist* EAs [18]. Eremeev used a fitness-level technique to obtain bounds on the expected proportion of the population of a *non-elitist* EA above above a certain fitness level [5]. By generalising results in [11], the first adaptation of the fitness-level technique to run-time analysis of non-elitist population-based EAs was made in [8], and refined in [2]. One limitation of the approaches in [2,8] is that the partition must be fitness-based and only *unary variation operators* are allowed, e.g. Genetic Algorithms (GAs) are excluded. Runtime analysis of GAs has been subjected to increasing interest in the recent years (e.g. see [3,10,12,13,15]).

We show that the above limitations can be removed from [2]. This gives rise to a much more general tool which is applicable to a wide range of search processes involving non-elitist populations. As introductory examples, we analyse the runtime of variants of the Genetic Algorithm (GA) with different selection mechanisms and crossover operators on well-known functions, such as OneMax and LeadingOnes, and on the sorting problem.

T. Bartz-Beielstein et al. (Eds.): PPSN XIII 2014, LNCS 8672, pp. 912–921, 2014.

2 Algorithmic Scheme

We consider population-based algorithms at a very abstract level in which fitness evaluations, selection and variation operations, which depending on the current population P of size λ, are represented by a distribution $D(P)$ over a finite set $\mathcal{X}$. More precisely, D is a mapping from $\mathcal{X}^\lambda$ into the space of probability distributions over $\mathcal{X}$. The next generation is obtained by sampling each new individual independently from $D(P)$. This scheme is summarised below.

Algorithm 1. Population-based algorithm with independent sampling

Require:
Finite state space $\mathcal{X}$, and population size $\lambda \in \mathbb{N}$,
Mapping D from $\mathcal{X}^\lambda$ to the space of probability distributions over $\mathcal{X}$.
1. $P_0 \sim \text{Unif}(\mathcal{X}^\lambda)$
2. **for** $t = 0, 1, 2, \ldots$ until termination condition met **do**
3. Sample $P_{t+1}(i) \sim D(P_t)$ independently for each $i \in [\lambda]$
4. **end for**

A similar scheme was studied in [17], where it was called *Random Heuristic Search* with *an admissible transition rule* (see [17]). Some examples of such algorithms are Simulated Annealing (more generally any algorithm with the population composed of a single individual), Stochastic Beam Search [17], Estimation of Distribution Algorithms such as the Univariate Marginal Distribution Algorithm [1] and the Genetic Algorithm (GA) [6]. The previous studies of the framework were often limited to some restricted settings [12] or mainly focused on infinite populations [17]. In this paper, we are interested in finite populations and develop a general method to deduce the expected runtime of the search processes defined in terms of *number of evaluations*. We illustrate our methods with runtime analysis of GAs under various settings (which are different to [12]).

The term Genetic Algorithm is often applied to EAs that use recombination operators. The GA is Algorithm 1 where the sampling $y \sim D(P_t)$ is the following: (i) $u \sim p_{\text{sel}}(P_t)$, $v \sim p_{\text{sel}}(P_t)$ (selection); (ii) $\{x', x''\} \sim p_{\text{xor}}(u, v)$, $x \sim \text{Unif}(\{x', x''\})$ (crossover); (iii) $y \sim p_{\text{mut}}(x)$ (mutation). The additional part $x \sim \text{Unif}(\{x_1, x_2\})$ at crossover is to match Algorithm 1 that produces only one resulting bitstring. We call this operator the *one-offspring* version of the *standard* crossover. In the rest of this paper, the two operations of the one-offspring version will be denoted simply by $x \sim p_{\text{xor}}(x, y)$. Here the standard operators of GA are formally represented by transition matrices:

- $p_{\text{sel}} : [\lambda] \to [0, 1]$ represents a selection operator, where $p_{\text{sel}}(i|P_t)$ is the probability of selecting the i-th individual from population P_t.
- $p_{\text{mut}} : \mathcal{X} \times \mathcal{X} \to [0, 1]$, where $p_{\text{mut}}(y|x)$ is the probability of mutating $x \in \mathcal{X}$ into $y \in \mathcal{X}$.
- $p_{\text{xor}} : \mathcal{X} \times \mathcal{X}^2 \to [0, 1]$, where $p_{\text{xor}}(x|u, v)$ is the probability of obtaining x as a result of crossover (or recombination) between $u, v \in \mathcal{X}$

3 Main Theorem

This section states a general technique for obtaining upper bounds on the expected runtime of any process that can be described in the form of Algorithm 1. We use the following notation. For any positive integer n, define $[n] := \{1, 2, ..., n\}$. The natural logarithm is denoted by $\ln(\cdot)$. The complement of an event $\mathcal{E}$ is denoted by $\bar{\mathcal{E}}$. Suppose that for some m there is an ordered partition of $\mathcal{X}$ into subsets $(A_1, \ldots, A_{m+1})$ called *levels*. For $j \in [m]$ we denote by $A_j^+ := \cup_{i=j+1}^{m+1} A_i$, the union of all levels above level j. An example of partition is the *canonical* partition, where each level regroups solutions having the same fitness value (see e.g. [8]). This partition is classified as *fitness-based*, our main theorem is not limited to this particular type of partition.

Lemma 1 (Lemma 5 and 6 in [2]). *Let $X \sim \mathrm{Bin}(\lambda, p)$ with $p \geq (i/\lambda)(1+\delta)$, it holds that $E\left[e^{-\kappa X}\right] \leq e^{-\kappa i}$ for any $\kappa \in (0, \delta)$. For $i \geq 1$, it also holds that $E\left[\ln\left((1+cX)/(1+ci)\right)\right] \geq c\varepsilon$ where $\varepsilon = \min\{1/2, \delta/2\}$ and $c = \varepsilon^4/24$.*

Theorem 1. *Given a partition $(A_1, \ldots, A_{m+1})$ of $\mathcal{X}$, define $T := \min\{t\lambda \mid |P_t \cap A_{m+1}| > 0\}$ to be the first point in time that elements of A_{m+1} appear in P_t of Algorithm 1. If there exist parameters $z_1, \ldots, z_m, z_* \in (0, 1]$, $\delta > 0$, a constant $\gamma_0 \in (0, 1)$ and a function $z_0 : (0, \gamma_0) \to \mathbb{R}$ such that for all $j \in [m]$, $P \in \mathcal{X}^\lambda$, $y \sim D(P)$ and $\gamma \in (0, \gamma_0)$ we have*

(G1) $\Pr\left(y \in A_j^+ \mid |P \cap A_{j-1}^+| \geq \gamma_0\lambda\right) \geq z_j \geq z_*$

(G2) $\Pr\left(y \in A_j^+ \mid |P \cap A_{j-1}^+| \geq \gamma_0\lambda, |P \cap A_j^+| \geq \gamma\lambda\right) \geq z_0(\gamma) \geq (1+\delta)\gamma$

(G3) $\lambda \geq \dfrac{2}{a} \ln\left(\dfrac{16m}{ac\varepsilon z_*}\right)$ *with* $a = \dfrac{\delta^2\gamma_0}{2(1+\delta)}$, $\varepsilon = \min\{\delta/2, 1/2\}$ *and* $c = \varepsilon^4/24$

then $E[T] \leq \frac{2}{c\varepsilon}\left(m\lambda(1+\ln(1+c\lambda)) + \sum_{j=1}^{m} \frac{1}{z_j}\right)$

Informally, the two first conditions require a relationship between P and the distribution $D(P)$: (G1) demands a certain probability z_j of creating an individual at level $j+1$ when some fixed portion of the population is already at level j (or higher); (G2) requires that in the fixed portion, the number of individuals at levels strictly higher than j (if those exist) tends to increase, e.g. by a multiplicative factor of $1+\delta$. Finally, (G3) requires a sufficiently large population size. The proof follows the same ideas as those in [2].

Proof. We use the following notation. The number of individuals in $A_j \cup A_j^+$ at generation t is denoted by X_t^j. The current level of the population at generation t is denoted by Z_t, where $Z_t := \ell$ iff $X_t^\ell \geq \lceil\gamma_0\lambda\rceil$ and $X_t^{\ell+1} < \gamma_0\lambda$. Note that Z_t is uniquely defined, as it is the level of the γ_0-ranked individual at generation t. We also use q_j to denote the probability to generate at least one individual at a level strictly greater than j in the next generation, knowing that there are at least $\lceil\gamma_0\lambda\rceil$ individuals of the population at level j or higher in the current generation. Because of (G1), we have $q_j \geq 1-(1-z_j)^\lambda \geq z_j\lambda/(z_j\lambda+1)$.

The theorem can now be proved using the additive drift theorem with respect to the potential function $g(t) := g_1(t) + g_2(t)$, where

$$g_1(t) := (m - Z_t)\ln(1 + c\lambda) - \ln(1 + cX_t^{Z_t+1})$$
$$\text{and } g_2(t) := \frac{1}{q_{Z_t} e^{\kappa X_t^{Z_t+1}}} + \sum_{j=Z_t+1}^{m-1} \frac{1}{q_j} \text{ with } \kappa \in (0, \delta)$$

The above components originated from the drift analysis of [8], then later improved by [2]. Function g is bounded from above by, $g(t) \leq m\ln(1 + c\lambda) + \sum_{j=1}^{m} \frac{1}{q_j} \leq m(1+\ln(1+c\lambda)) + \frac{1}{\lambda}\sum_{j=1}^{m} \frac{1}{z_j}$. At generation t, we use $R = Z_{t+1} - Z_t$ to denote the random variable describing the next progress in terms of levels. To simplify further writing, let us put $\ell = Z_t$, $i = X_t^\ell$, $X = X_{t+1}^\ell$, then $\Delta = g(t) - g(t + 1) = \Delta_1 + \Delta_2$ with

$$\Delta_1 := g_1(t) - g_1(t + 1) = R\ln(1 + c\lambda) + \ln\left(\frac{1 + X_{t+1}^{\ell+R+1}}{1 + ci}\right)$$
$$\Delta_2 := g_2(t) - g_2(t + 1) = \frac{1}{q_\ell e^{\kappa i}} - \frac{1}{q_{\ell+R} e^{\kappa X_{t+1}^{\ell+R+1}}} + \sum_{j=\ell+1}^{l+R} \frac{1}{q_j}$$

Let us denote by $\mathcal{E}_t$ the event that the population in the next generation does not fall down to a lower level, $\mathcal{E}_t : Z_{t+1} \geq Z_t$. We first compute the *conditional forward drift* $E[\Delta|\mathscr{F}_t, \mathcal{E}_t]$, here $\mathscr{F}_t$ is the filtration induced by P_t. Under $\mathcal{E}_t$, R is a non-negative random variable and Δ is a random variable indexed by R, noted as $\Delta = Y_R$. We can show that $Y_{r\geq 1} \geq Y_0$ for fixed indexes.

$$\begin{aligned}
Y_0 &= \ln\left(\frac{1 + cX}{1 + ci}\right) + \frac{1}{q_\ell e^{\kappa i}} - \frac{1}{q_\ell e^{\kappa X}} \leq \ln\left(\frac{1 + c\lambda}{1 + ci}\right) + \frac{1}{q_\ell e^{\kappa i}} \\
Y_{r\geq 1} &= r\ln(1 + c\lambda) + \ln\left(\frac{1 + X_{t+1}^{\ell+r+1}}{1 + ci}\right) + \frac{1}{q_\ell e^{\kappa i}} - \frac{1}{q_{\ell+r} e^{\kappa X_{t+1}^{\ell+r+1}}} + \sum_{j=\ell+1}^{\ell+r} \frac{1}{q_j} \\
&\geq \ln(1 + c\lambda) + \ln\left(\frac{1}{1 + ci}\right) + \frac{1}{q_\ell e^{\kappa i}} - \frac{1}{q_{\ell+r}} + \sum_{j=\ell+1}^{\ell+r} \frac{1}{q_j} \\
&= \ln\left(\frac{1 + c\lambda}{1 + ci}\right) + \frac{1}{q_\ell e^{\kappa i}} + \sum_{j=\ell+1}^{\ell+r-1} \frac{1}{q_j} \geq \ln\left(\frac{1 + c\lambda}{1 + ci}\right) + \frac{1}{q_\ell e^{\kappa i}} \geq Y_0
\end{aligned}$$

It is then clear (or see Lemma 7 in [2]) that $E[\Delta|\mathscr{F}_t, \mathcal{E}_t] = E[Y_R|\mathscr{F}_t, \mathcal{E}_t] \geq E[Y_0|\mathscr{F}_t, \mathcal{E}_t]$, so we only focus on $r = 0$ to lower bound the drift. We separate two cases, $i = 0$, the event is denoted by $\mathcal{Z}_t$, and $i \geq 1$ (event $\bar{\mathcal{Z}}_t$). Recall that each individual is generated independently from each other, so during $\bar{\mathcal{Z}}_t$ we have that $X \sim \text{Bin}(\lambda, p)$ where $p \geq z_0(i/\lambda)$. From (G2), we also get $z_0(i/\lambda) \geq (i/\lambda)(1+\delta)$. Hence $p \geq (i/\lambda)(1 + \delta)$ and by Lemma 1, it holds for $i \geq 1$ (event $\bar{\mathcal{Z}}_t$) that

$$E\left[\Delta_1|\mathscr{F}_t, \mathcal{E}_t, \bar{\mathcal{Z}}_t\right] \geq E\left[\ln\left(\frac{1 + cX}{1 + ci}\right)|\mathscr{F}_t, \mathcal{E}_t, \bar{\mathcal{Z}}_t\right] \geq c\varepsilon$$

$$E\left[\Delta_2|\mathscr{F}_t,\mathcal{E}_t,\bar{\mathcal{Z}}_t\right] \geq \frac{1}{q_\ell}(e^{-\kappa i} - E\left[e^{-\kappa X}|\mathscr{F}_t,\mathcal{E}_t,\bar{\mathcal{Z}}_t\right]) \geq 0$$

For $i = 0$, we get $E\left[\Delta_1|\mathscr{F}_t,\mathcal{E}_t,\mathcal{Z}_t\right] \geq E\left[\ln(1)|\mathscr{F}_t,\mathcal{E}_t,\mathcal{Z}_t\right] = 0$ because $X \geq 0$. Recall that $q_\ell = \Pr\left(X \geq 1|\mathscr{F}_t,\mathcal{E}_t,\mathcal{Z}_t\right)$, so

$$\begin{aligned} E\left[\Delta_2|\mathscr{F}_t,\mathcal{E}_t,\mathcal{Z}_t\right] &\geq \Pr\left(X \geq 1|\mathscr{F}_t,\mathcal{E}_t,\mathcal{Z}_t\right) E\left[\frac{1}{q_\ell e^{\kappa i}} - \frac{1}{q_\ell e^{\kappa X}}|\mathscr{F}_t,\mathcal{E}_t,\mathcal{Z}_t, X \geq 1\right] \\ &\geq q_\ell(1/q_\ell)(e^{-\kappa\cdot 0} - e^{-\kappa\cdot 1}) = 1 - e^{-\kappa} \end{aligned}$$

So the conditional forward drift is $E\left[\Delta|\mathscr{F}_t,\mathcal{E}_t\right] \geq \min\{c\varepsilon, 1-e^{-\kappa}\}$. Furthermore, κ can be picked in the non-empty interval $(-\ln(1-c\varepsilon),\delta) \subset (0,\delta)$, so that $1-e^{-\kappa} > c\varepsilon$ and $E\left[\Delta|\mathscr{F}_t,\mathcal{E}_t\right] \geq c\varepsilon$. Next, we compute the *conditional backward drift*, which can be done for the worst case.

$$E\left[\Delta|\mathscr{F}_t,\bar{\mathcal{E}}_t\right] \geq -(m-1)\ln(1+c\lambda) - \ln(1+c\lambda) - \sum_{j=1}^{m} 1/q_j \geq -m\left(c\lambda + 2/z_*\right)$$

The probability that event $\mathcal{E}_t$ does not occur is computed as follows. Recall that $X_t^\ell \geq \lceil\gamma_0\lambda\rceil$ and X_{t+1}^ℓ is binomially distributed random variable with probability at least $z_0(\gamma_0) \geq (1+\delta)\gamma_0$ by condition (G2), so $E\left[X_{t+1}^\ell|\mathscr{F}_t\right] \geq (1+\delta)\gamma_0\lambda$. The event $\bar{\mathcal{E}}_t$ happens when the number of individuals at level ℓ is strictly less than $\lceil\gamma_0\lambda\rceil$ in the next generation. By a Chernoff bound (see [4]), we have

$$\begin{aligned} \Pr\left(\bar{\mathcal{E}}_t|\mathscr{F}_t\right) &= \Pr\left(X_{t+1}^\ell < \lceil\gamma_0\lambda\rceil|\mathscr{F}_t\right) \leq \Pr\left(X_{t+1}^\ell \leq \gamma_0\lambda|\mathscr{F}_t\right) \\ &= \Pr\left(X_{t+1}^\ell \leq \left(1-\delta/(1+\delta)\right)(1+\delta)\gamma_0\lambda|\mathscr{F}_t\right) \\ &\leq \Pr\left(X_{t+1}^\ell \leq \left(1-\delta/(1+\delta)\right)E\left[X_{t+1}^\ell|\mathscr{F}_t\right]|\mathscr{F}_t\right) \\ &\leq \exp\left(-\frac{\delta^2 E\left[X_{t+1}^\ell|\mathscr{F}_t\right]}{2(1+\delta)^2}\right) \leq \exp\left(-\frac{\delta^2(1+\delta)\gamma_0\lambda}{2(1+\delta)^2}\right) = e^{-a\lambda} \end{aligned}$$

Recall condition (G3) that $\lambda \geq (2/a)\ln\left((16m)/(ac\varepsilon z_*)\right)$. This implies that $(8m)/(ac\varepsilon z_*) \leq e^{\frac{a\lambda}{2}}/2 \leq e^{a\lambda}/(a\lambda)$, or $e^{-a\lambda} \leq c\varepsilon z_*/(8m\lambda)$. The drift is therefore

$$\begin{aligned} E\left[\Delta|\mathscr{F}_t\right] &= (1-\Pr\left(\bar{\mathcal{E}}_t|\mathscr{F}_t\right))E\left[\Delta|\mathscr{F}_t,\mathcal{E}_t\right] + \Pr\left(\bar{\mathcal{E}}_t|\mathscr{F}_t\right)E\left[\Delta|\mathscr{F}_t,\bar{\mathcal{E}}_t\right] \\ &= E\left[\Delta|\mathscr{F}_t,\mathcal{E}_t\right] - \Pr\left(\bar{\mathcal{E}}_t|\mathscr{F}_t\right)\left(E\left[\Delta|\mathscr{F}_t,\mathcal{E}_t\right] - E\left[\Delta|\mathscr{F}_t,\bar{\mathcal{E}}_t\right]\right) \\ &\geq c\varepsilon - \frac{c\varepsilon z_*}{8m\lambda}\left(c\varepsilon + m\left(c\lambda + \frac{2}{z_*}\right)\right) \\ &\geq c\varepsilon - \frac{c\varepsilon}{8}\left(\frac{c\varepsilon z_*}{\lambda m} + z_* c + \frac{2}{\lambda}\right) \geq c\varepsilon - \frac{4c\varepsilon}{8} = \frac{c\varepsilon}{2} \end{aligned}$$

By additive drift [7], $E\left[T\right] \leq \frac{2}{c\varepsilon}\left(m\lambda(1+\ln(1+c\lambda)) + \sum_{j=1}^{m}\frac{1}{z_j}\right)$. □

In the special case of unary variation operators Theorem 1 becomes analogous to the main results of [2,8]. It is an open problem whether the upper bound in Theorem 1 is tight. The lack of general tools make this problem hard. The family tree technique [19], the population drift theorem [9], and the fitness level technique in [16], provide lower bounds for population-based algorithms, however only for less general settings than Algorithm 1.

4 Runtime Analysis of Genetic Algorithms

This section provides a version of Theorem 1 tailored to the GAs described in Section 2. The *selective pressure* of a selection mechanism p_{sel} is defined as follows. For any $\gamma \in (0,1)$ and population P of size λ, let $\beta(\gamma, P)$ be the probability of selecting an individual from P that is at least as good as the individual with rank $\lceil \gamma\lambda \rceil$ (see [2] or [8] for a formal definition). We assume that p_{sel} is *monotone* with respect to fitness values [8], ie for all $P \in \mathcal{X}^\lambda$ and pairs $i, j \in [\lambda]$, $p_{\text{sel}}(i \mid P) \geq p_{\text{sel}}(j \mid P)$ if and only if $f(P(i)) \geq f(P(j))$.

Corollary 1. *Given a function* $f : \mathcal{X} \to \mathbb{R}$ *and a partition* $(A_1, \ldots, A_{m+1})$ *of* $\mathcal{X}$*, let* $T := \min\{t\lambda \mid |P_t \cap A_{m+1}| > 0\}$ *be the runtime of the non-elitist Genetic Algorithm, as described in Section 2, on* f*. If there exist parameters* $s_1, \ldots, s_m, s_*, p_0, \varepsilon_1 \in (0,1]$*,* $\delta > 0$*, and a constant* $\gamma_0 \in (0,1)$ *such that for all* $j \in [m]$*,* $P \in \mathcal{X}^\lambda$*, and* $\gamma \in (0, \gamma_0)$

(C1) $p_{\text{mut}}(y \in A_j^+ \mid x \in A_{j-1}^+) \geq s_j \geq s_*$
(C2) $p_{\text{mut}}(y \in A_j^+ \mid x \in A_j^+) \geq p_0$
(C3) $p_{\text{xor}}(x \in A_j^+ \mid u \in A_{j-1}^+, v \in A_j^+) \geq \varepsilon_1$
(C4) $\beta(\gamma, P) \geq \gamma\sqrt{\frac{1+\delta}{p_0\varepsilon_1\gamma_0}}$
(C5) $\lambda \geq \dfrac{2}{a} \ln\left(\dfrac{32mp_0}{(\delta\gamma_0)^2 c s_* \psi}\right)$ *with* $a := \dfrac{\delta^2\gamma_0}{2(1+\delta)}$, $\psi := \min\{\frac{\delta}{2}, \frac{1}{2}\}$ *and* $c := \frac{\psi^4}{24}$

then $E[T] \leq \frac{2}{c\psi}\left(m\lambda(1 + \ln(1 + c\lambda)) + \frac{p_0}{(1+\delta)\gamma_0}\sum_{j=1}^m \frac{1}{s_j}\right)$.

Proof. We show that conditions (C1-5) imply conditions (G1-3) in Theorem 1. We first show that condition (G1) is satisfied for $z_j = \gamma_0(1+\delta)s_j/p_0$. Assume that $|P \cap A_{j-1}^+| \geq \gamma_0\lambda$. Remark that (C3) written for one level below, which is $p_{\text{xor}}(x \in A_{j-1}^+ \mid u \in A_{j-2}^+, v \in A_{j-1}^+) \geq \varepsilon_1$, implies $p_{\text{xor}}(x \in A_{j-1}^+ \mid u \in A_{j-1}^+, v \in A_{j-1}^+) \geq \varepsilon_1$. To sample an individual in A_j^+, it suffices that the selection operator picks two individuals u and v from A_{j-1}^+, that the crossover operator produces an individual x in A_{j-1}^+ from u and v, and the mutation operator produces an individual y in A_j^+ from x. By conditions (C4), (C3) as the remark, and (C1), the probability of this event is at least $\beta(\gamma_0)\beta(\gamma_0)\varepsilon_1 s_j \geq \gamma_0(1+\delta)s_j/p_0 = z_j$.

We then show that condition (G2) is satisfied. Assume that $|P \cap A_{j-1}^+| \geq \gamma_0\lambda$ and $|P \cap A_j^+| \geq \gamma\lambda$. To produce an individual y in A_j^+, it suffices that the selection operator picks an individual u in A_{j-1}^+ and an individual v in A_j^+, that the crossover operator produces an individual x in A_j^+ from u and v, and the mutation operator produces an individual y in A_j^+ from x. By conditions (C4), (C3), and (C2), the probability of this event is at least $\beta(\gamma_0)\beta(\gamma)\varepsilon_1 p_0 \geq (1+\delta)\gamma$.

Finally, to see that condition (G3) is satisfied, it suffices to note that $az_* = (\delta\gamma_0)^2 s_*/(2p_0)$. Hence, the statement now follows from Theorem 1. $\square$

4.1 Runtime of GAs on Simple pseudo-Boolean Functions

We apply Corollary 1 to bound the expected runtime of the non-elitist GA on the functions $\textsc{OneMax}(x) := \sum_{i=1}^{n} x_i$ (also written shortly as $|x|_1$ or $\textsc{Om}$) and $\textsc{LeadingOnes}(x) := \sum_{i=1}^{n} \prod_{j=1}^{i} x_j$ (shortly as $\textsc{Lo}$).

We first show how to parameterise three standard selection mechanisms such that condition (C4) is satisfied. In *k-tournament selection*, k individuals are sampled uniformly at random with replacement from the population, and the fittest of these individuals is returned. In (μ, λ)-*selection*, parents are sampled uniformly at random among the fittest μ individuals in the population. A function $\alpha : \mathbb{R} \to \mathbb{R}$ is a ranking function [6] if $\alpha(x) \geq 0$ for all $x \in [0, 1]$, and $\int_0^1 \alpha(x)dx = 1$. In ranking selection with ranking function α, the probability of selecting individuals ranked γ or better is $\int_0^\gamma \alpha(x)dx$. We define *exponential ranking* parameterised by $\eta > 0$ as $\alpha(\gamma) := \eta e^{\eta(1-\gamma)}/(e^\eta - 1)$. [1]

Lemma 2. *For any constant $\delta > 0$, there exists a constant $\gamma_0 \in (0, 1)$ such that*

1. *k-tournament selection with $k \geq 4(1 + \delta)/(\varepsilon_1 p_0)$ satisfies (C4)*
2. *(μ, λ)-selection with $\lambda/\mu \geq (1 + \delta)/(\varepsilon_1 p_0)$ satisfies (C4)*
3. *exponential ranking selection with $\eta \geq 4(1 + \delta)/(\varepsilon_1 p_0)$, satisfies (C4).*

Lemma 3 shows that two standard crossover operators satisfy (C3) for $\varepsilon_1 = \frac{1}{2}$.

Lemma 3. *If $x \sim p_{\text{xor}}(u, v)$, where p_{xor} is one-point or uniform crossover, then*

1. *If $\textsc{Lo}(u) = \textsc{Lo}(v) = j$, then $\Pr(\textsc{Lo}(x) \geq j) = 1$.*
2. *If $\textsc{Lo}(u) \neq \textsc{Lo}(v)$, then $\Pr(\textsc{Lo}(x) > \min\{\textsc{Lo}(u), \textsc{Lo}(v)\}) \geq 1/2$.*
3. *$\Pr(\textsc{Om}(x) \geq \lceil(\textsc{Om}(u) + \textsc{Om}(v))/2\rceil) \geq 1/2$.*

Theorem 2. *Assume that the GA with one-point or uniform crossover, bitwise mutation with mutation rate χ/n for a constant $\chi > 0$, and either k-tournament selection with $k \geq 8(1 + \delta)e^\chi$, or (μ, λ) selection with $\lambda/\mu \geq 2(1 + \delta)e^\chi$ or the exponential ranking selection with $\eta \geq 8(1 + \delta)e^\chi$, for a constant $\delta > 0$. Then there exists a constant $c > 0$, such that the GA with population size $\lambda \geq c \ln n$, has expected runtime $O(n\lambda \ln \lambda + n^2)$ on* $\textsc{LeadingOnes}$*, and expected runtime $O(n\lambda \ln \lambda)$ on* $\textsc{OneMax}$.

Proof. Let f be either $\textsc{Om}$ or $\textsc{Lo}$. We apply Corollary 1 with the canonical partition of the search space into $n + 1$ levels $A_j := \{x \mid f(x) = j - 1\}$, for $j \in [n + 1]$. We use $p_0 := (1 - \chi/n)^n$ the probability of not flipping any bit position by mutation, and for all $j \in [n]$, define

$$s_j := \begin{cases} (\chi/n)(1 - \chi/n)^{n-1} & \text{if } f = \textsc{Lo}, \text{ and} \\ (n - j + 1)(\chi/n)(1 - \chi/n)^{n-1}p_0 & \text{if } f = \textsc{Om}. \end{cases}$$

Considering condition (C1), when $x \in A_j$ it suffices to upgrade x to a higher level, the probability of such an event is at least s_j for $\textsc{Lo}$ and $s_j/p_0 > s_j$ for $\textsc{Om}$.

[1] The proofs of Lemmas 2 and 3 are omitted due to space restrictions.

When $x \in A_j^+$, it suffices to not modify x, the probability of such an event is at least $p_0 \geq s_j$ with sufficiently large n for Lo and $p_0 > (n-j+1)(\chi/n)(1-\chi/n)^{n-1}p_0 = s_j$ for Om. So, condition (C1) is satisfied for both functions with the given s_j. In addition, condition (C2) is trivially satisfied for the given p_0 and condition (C3) is satisfied for the parameter $\varepsilon_1 := 1/2$ by Lemma 3.

We now look at condition (C4), and remark that $p_0 = (1-\chi/n)^{(n/\chi-1)\chi}(1-\chi/n)^\chi \geq e^{-\chi}(1-\chi/n)^\chi$. So $e^\chi \geq (1-\chi/n)^\chi/p_0$ and with the given condition for k-tournament, we get $k \geq 8(1+\delta)e^\chi = 4(1+\delta)e^\chi/(1/2) \geq 4(1+\delta)(1-\chi/n)^\chi/(\varepsilon_1 p_0)$. Then for any constant $\delta' \in (0,\delta)$ and sufficiently large n, literally $n \geq \chi/(1-((1+\delta')/(1+\delta))^{1/\chi})$, it holds that $k \geq 4(1+\delta')/(\varepsilon_1 p_0)$. So condition (C4) is satisfied with the constant δ' for k-tournament by Lemma 2. The same reasoning can be applied so that (C4) is also satisfied for the other selection mechanisms.

Finally, $s_* := \min_{j\in[n]} s_j = \Omega(1/n)$. So assuming $\lambda \geq c\ln n$ for a sufficiently large constant c, condition (C5) is satisfied as well. Note that the p_0 part in s_j of Lo only removes the p_0 from $p_0/(1+\delta)\gamma_0$ in the runtime of Corollary 1. Therefore, the upper bounds $O(n\lambda\ln\lambda + n^2)$ and $O(n\lambda\ln\lambda)$ on the expected runtime are proven for Om and Lo respectively. □

Note that the upper bounds in Theorem 2 match the upper bounds obtained in [2] for EAs without crossover.

4.2 Runtime of GAs on the Sorting Problem

Given n distinct elements from a totally ordered set, we consider the problem of finding an ordering of them so that some *measure of sortedness* is maximised. Scharnow et al. [14] considered several sortedness measures in the context of analysing the (1+1) EA. One of those is $INV(\pi)$ which is defined to be the number of pairs (i,j) such that $1 \leq i < j \leq n$, $\pi(i) < \pi(j)$ (i.e. pairs in correct order). We show that with the methods introduced in this paper, analysing GAs on $INV(\pi)$ is not much harder than analysing the (1+1) EA.

As mutation operator, we consider the $Exchange(\pi)$ operator, which consecutively applies N pairwise exchanges between uniformly selected pairs of indices, where N is a random number drawn from a Poisson distribution with parameter 1. We consider a crossover operator, denoted by $p_{\mathrm{xor}(p_c)}$, which returns one of the parents unchanged with probability $1-p_c$. For example, $p_{\mathrm{xor}(p_c)}$ is built up from any standard crossover operator so that with probability p_c the standard operator is applied and the offspring is returned, otherwise with probability $1-p_c$ one of the parents is returned in place of the offspring. This construction corresponds to a typical setting (see [6]) where there is some *crossover probability* p_c of applying the crossover before the mutation. As selection mechanism, we consider k-tournament selection, (μ,λ)-selection, and exponential ranking selection.

Theorem 3. *If the GA uses a $p_{\mathrm{xor}(p_c)}$ crossover operator with p_c being any constant in $[0,1)$, the Exchange mutation operator where the number of exchanges N is drawn from a Poisson distribution with parameter 1, k-tournament selection with $k \geq 8e(1+\delta)/(1-p_c)$, or (μ,λ)-selection with $\lambda/\mu \geq 2e(1+\delta)/(1-p_c)$,* *or exponential ranking selection with* $\eta \geq 8e(1+\delta)/(1-p_c)$*, then there exists a constant* $c > 0$ *such that if the population size is* $\lambda \geq c \ln n$*, the expected time to obtain the optimum of INV is* $O(n^2 \lambda \log \lambda)$*.*

Proof. Define $m := \binom{n}{2}$. We apply Corollary 1 with the canonical partition, $A_j := \{\pi \mid INV(\pi) = j\}$ for $j \in [m]$. The probability that the exchange operator exchanges 0 pairs is $1/e$. Hence, condition (C2) is trivially satisfied for $p_0 := 1/e$.

To show that condition (C1) is satisfied, define first $s_j := (m-j)p_0/(em)$. In the case that $x \in A_j$, then the probability that the exchange operator exchanges exactly one pair is $1/e$, and the probability that this pair is incorrectly ordered in x, is $(m-j)/m$. In the other case that, $x \in A_j^+$, it is sufficient that the exchange operator exchanges 0 pairs, which by condition (C2) occurs with probability at least $p_0 \geq s_j$. Hence, in both cases, $y \in A_j^+$ with probability at least s_j, and condition (C1) is satisfied.

Condition (C3) is trivially satisfied for $\varepsilon_1 := (1-p_c)/2$, because the crossover operator returns one of the parents unchanged with probability $1-p_c$, and with probability $1/2$, this parent is v. Condition (C4) is satisfied for some constant $\gamma_0 \in (0,1)$ by Lemma 2. Finally, since γ_0, δ, and p_0 are constants, there exists a constant $c > 0$ such that condition (C5) is satisfied for any $\lambda \geq c \ln(n)$.

It therefore follows that the expected runtime of the GA on INV is upper bounded by $O(n^2 \lambda \log \lambda)$. □

5 Conclusion

Most results in runtime analysis of evolutionary algorithms concern relatively simple algorithms, e.g. the (1+1) EA, which do not employ populations and higher-arity variation operators, such as crossover. This paper introduces a new tool, akin to the fitness-level technique, that easily yields upper bounds on the expected runtime of complex, non-elitist search processes. The tool is illustrated on Genetic Algorithms. Given an appropriate balance between selection and variation operators, we have shown that GAs optimise standard benchmark functions, as well as combinatorial optimisation problems, efficiently. Future applications of the theorem might consider the impact of the crossover operator more in detail, e.g. by a more precise analysis of the population diversity.

Acknowledgements. This research received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no 618091 (SAGE), from EPSRC grant no EP/F033214/1 (LANCS), and from Russian Foundation for Basic Research grant 12-01-00122. Early ideas where discussed at Dagstuhl Seminar 13271 Theory of Evolutionary Algorithms.

References

1. Chen, T., Lehre, P., Tang, K., Yao, X.: When is an estimation of distribution algorithm better than an evolutionary algorithm? In: Proceedings of IEEE Congress on Evolutionary Computation (CEC 2009), pp. 1470–1477 (2009)

2. Dang, D.C., Lehre, P.K.: Refined upper bounds on the expected runtime of non-elitist populations from fitness levels. In: Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2014) (to appear, 2014)
3. Doerr, B., Doerr, C., Ebel, F.: Lessons from the black-box: Fast crossover-based genetic algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2013), pp. 781–788 (2013)
4. Dubhashi, D., Panconesi, A.: Concentration of Measure for the Analysis of Randomized Algorithms. Cambridge University Press, NY (2009)
5. Eremeev, A.V.: Modeling and analysis of genetic algorithm with tournament selection. In: Fonlupt, C., Hao, J.-K., Lutton, E., Schoenauer, M., Ronald, E. (eds.) AE 1999. LNCS, vol. 1829, pp. 84–95. Springer, Heidelberg (2000)
6. Goldberg, D.E.: Genetic Algorithms in search, optimization and machine learning. Addison-Wesley, MA (1989)
7. He, J., Yao, X.: Drift analysis and average time complexity of evolutionary algorithms. Artificial Intelligence 127(1), 57–85 (2001)
8. Lehre, P.K.: Fitness-levels for non-elitist populations. In: Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2011), pp. 2075–2082 (2011)
9. Lehre, P.K.: Negative drift in populations. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 244–253. Springer, Heidelberg (2010)
10. Lehre, P.K., Yao, X.: Crossover can be constructive when computing unique input-output sequences. Soft Computing 15(9), 1675–1687 (2011)
11. Lehre, P.K., Yao, X.: On the impact of mutation-selection balance on the runtime of evolutionary algorithms. IEEE Transactions on Evolutionary Computation 16(2), 225–241 (2012)
12. Oliveto, P.S., Witt, C.: Improved runtime analysis of the simple genetic algorithm. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2013), pp. 1621–1628 (2013)
13. Qian, C., Yu, Y., Zhou, Z.H.: An analysis on recombination in multi-objective evolutionary optimization. Artificial Intelligence 204, 99–119 (2013)
14. Scharnow, J., Tinnefeld, K., Wegener, I.: The analysis of evolutionary algorithms on sorting and shortest paths problems. Journal of Mathematical Modelling and Algorithms 3(4), 349–366 (2004)
15. Sudholt, D.: Crossover speeds up building-block assembly. In: Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2012), pp. 689–702 (2012)
16. Sudholt, D.: A new method for lower bounds on the running time of evolutionary algorithms. IEEE Transactions on Evolutionary Computation 17(3), 418–435 (2013)
17. Vose, M.D.: The Simple Genetic Algorithm: Foundations and Theory. MIT Press, Cambridge (1999)
18. Wegener, I.: Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. Evolutionary Optimization 48, 349–369 (2002)
19. Witt, C.: Runtime analysis of the (μ+1) EA on simple pseudo-boolean functions. Evolutionary Computation 14(1), 65–86 (2006)