Supplementary Material: A Large-scale Multiple-objective Method for Black-box Attack against Object Detection

Siyuan Liang[1,2], Longkang Li[3], Yanbo Fan[4], Xiaojun Jia[1,2], Jingzhi Li[1,2,⋆], Baoyuan Wu[3,⋆], and Xiaochun Cao[5]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] School of Data Science, Secure Computing Lab of Big Data, The Chinese University of Hong Kong, Shenzhen, China
[4] Tencent AI Lab, Shenzhen, China
[5] School of Cyber Science and Technology, Shenzhen Campus, Sun Yat-sen University, Shenzhen, China
{liangsiyuan, jiaxiaojun, lijingzhi}@iie.ac.cn; {lilongkang, wubaoyuan}@cuhk.edu.cn; fanyanbo0124@gmail.com; caoxiaochun@mail.sysu.edu.cn

Organization of the Supplementary Material

In Section A, we show the algorithm flow of GARSDC, which attacks against object detection. In Section B, we add the specific proof for the lower bound(Eq. (8)) of the P values of the best $\boldsymbol{b}_j$. Section C gives the approximation guarantee of GARSDC(Eq. (9)).

A The Algorithm Flow of GARSDC

Algorithm 1 Adversarial Examples Generation with GARSDC Attack

Require: victim detector H_1, gradient-prior detectors H_2,H_3, clean image $\boldsymbol{x} \in \mathbb{R}^{w\times h\times c}$, ground-truth boxes $\mathcal{O}$, maximum number of iterations T_{max}, the size of a sample patch a, maximum iterations T_{dc} of divide-and-conquer, the flag of divide-and-conquer $flag$, numbers of divide-and-conquer i, crossover rate cr, mutation rate mr.

Ensure: adversarial image $\hat{\boldsymbol{x}} \in \mathbb{R}^{w\times h\times c}$ with $||\boldsymbol{\delta}||_\infty \leq 0.05$, $\hat{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\delta}$

1: $\boldsymbol{\delta}_1^{best}, \boldsymbol{\delta}_2^{best} \leftarrow \text{INIT}(\boldsymbol{x}, H_2, H_3)$(see Alg. 2 for the initial population in Section 3.3)

2: $f_{best} \leftarrow \max\{F(H_1(\boldsymbol{x}), \boldsymbol{x} + \boldsymbol{\delta}_m)\}, m = 1, 2$ in Eq. (3) with ground-truth boxes $\mathcal{O}$

3: $S_{best} = \{\infty\}_{j=1}^{i}$

⋆ Coressponding Author

4: Best individual's index is $m \leftarrow \arg\max_{\boldsymbol{\delta}_m}\{F(H_1(\boldsymbol{x}), \boldsymbol{x}+\boldsymbol{\delta}_m)\}, m = 1, 2$
5: $t \leftarrow 2, flag \leftarrow 0, a^{(2)} \leftarrow 0.05 * min(w, h)$
6: **while** $t < T_{max}$ **and** $\hat{\boldsymbol{x}}$ is not adversarial **do**
7: $a^{(t)}$ ←side length of the sample patch (according to some schedule)
8: $\boldsymbol{\delta}_1^{new}$, $\boldsymbol{\delta}_2^{new}$, $\boldsymbol{s} = \mathrm{RS}(w, h, c, \boldsymbol{\delta}_1^{best}, \boldsymbol{\delta}_2^{best}, a^{(t)})$(see Alg. 3 for Section 3.4)
9: Partition $\boldsymbol{s}$ into i sets $\boldsymbol{s}_1$, $\boldsymbol{s}_2$, ..., $\boldsymbol{s}_i$ evenly so that each $\boldsymbol{s}_j$ is a square with equal sides
10: $f_m^{new}, S_{new}^m \leftarrow F(H_1(\boldsymbol{x}), \boldsymbol{x}+\boldsymbol{\delta}_m^{new}), S(\boldsymbol{\delta}_m^{new}[\boldsymbol{s}])$
11: $t \leftarrow t+1$
12: **if** $\exists s_j \in S_{new}, s_j \neq \infty$ **then** $flag = 1$
13: **end if**
14: **if** $flag == 1$ **then**
15: **while** $j < i$ **do**
16: // Run Alg. 4 with $T = T_{dc} - 1$ on each $\boldsymbol{s}_j$
17: $\boldsymbol{\delta}_1[\boldsymbol{u}_j], \boldsymbol{\delta}_2[\boldsymbol{u}_j], f(\boldsymbol{\delta}_{1j}), f(\boldsymbol{\delta}_{2j}) = \mathrm{DC}(\boldsymbol{\delta}_1^{new}, \boldsymbol{\delta}_2^{new}, T_{dc}-1, H_1, \mathcal{O}, mr, cr, \boldsymbol{s}_j, \boldsymbol{x})$
18: **end while**
19: Merge the i resulting subsets into a set $U = \bigcup_{j=1}^{i} \boldsymbol{u}_j$
20: // Run Alg. 4 with $T = 1$ on each U
21: $\boldsymbol{\delta}_1[\boldsymbol{u}_{i+1}], \boldsymbol{\delta}_2[\boldsymbol{u}_{i+1}], f(\boldsymbol{\delta}_{1(i+1)}), f(\boldsymbol{\delta}_{2(i+1)}) = \mathrm{DC}(\boldsymbol{\delta}_1^{new}, \boldsymbol{\delta}_2^{new}, 1, H_1, \mathcal{O}, mr, cr, U, \boldsymbol{x})$
22: The best sub-component is $\boldsymbol{\delta}[\boldsymbol{u}_{best}] = \arg\max_{\boldsymbol{\delta}[\boldsymbol{u}_j]}\{f(\boldsymbol{\delta}[\boldsymbol{u}_j])\}, j = 1, ..., i+1$
23: Update $\boldsymbol{\delta}_1^{new}, \boldsymbol{\delta}_2^{new}$ with the best sub-component $\boldsymbol{\delta}_1[\boldsymbol{u}_{best}], \boldsymbol{\delta}_2[\boldsymbol{u}_{best}]$
24: Update individual's fitness f_1^{new} and f_2^{new}
25: Update best individual's index $m \leftarrow \arg\max(f_1^{new}, f_2^{new})$
26: $flag \leftarrow 0, t = t + T_{dc}$
27: **end if**
28: **if** $f_m^{new} > f_{best}$ **then** $\boldsymbol{\delta}_m^{best} \leftarrow \boldsymbol{\delta}_m^{new}, f_{best} \leftarrow f_m^{new}$
29: **end if**
30: **end while**
31: return adversarial perturbation $\boldsymbol{\delta}_m^{best}$

We show the overall algorithm of GARSDC in Alg. 1. In line 1, we use gradient-prior detectors consisting of Faster r-cnn H_2 and SSD H_3 to generate skip-based and chain-based perturbations as initial population, which is called mixed initial population in Section 3.3. In line 8, we randomly sample subsets across the full image using a strategy similar to PRFA. However, we do not need a prior-guided dimension reduction, which can circumvent the risk that the prior is terrible. In lines 12-13, we can use the sub-component fitness to help judge whether the sub-components of random search are helpful for optimization and decide search perturbation locally. In line 15, we simultaneously run Alg. 4 on i sets. In the first round (line 17), it evenly distributes the ground set $\boldsymbol{s}$ over i machines, and then each machine runs DC to find a subset $\boldsymbol{s}_j$ in parallel. In the second round, we merge the i resulting subsets are merged on one machine (line 19), and then DS to find another $\boldsymbol{s}_{i+1}$ (line 21). We update $\boldsymbol{\delta}_1^{new}, \boldsymbol{\delta}_2^{new}$ with the best sub-component and record individual fitness f_1^{new}, f_2^{new} at the same time.

Algorithm 2 Mixed Initial Population Based on Gradient-prior

Require: gradient-prior detectors H_2,H_3, clean image $\boldsymbol{x} \in \mathbb{R}^{w\times h\times c}$.
Ensure: adversarial image $\hat{\boldsymbol{x}} \in \mathbb{R}^{w\times h\times c}$ with $||\boldsymbol{\delta}||_\infty \leq 0.05$, $\hat{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\delta}$

$\boldsymbol{\delta}_1^{(0)} \leftarrow 0$, $\boldsymbol{\delta}_2^{(0)} \leftarrow 0$, $t \leftarrow 0$
while $t < 20$ **do**
  // skip-based perturbation for Eq. (5)
  Input $\boldsymbol{x}+\boldsymbol{\delta}_1^{(t)}$ to detector H_2 and obtaion the gradient $\boldsymbol{W} * \nabla_{\boldsymbol{x}} H_2(\boldsymbol{x}+\boldsymbol{\delta}_1^{(t)})$

  Update $\boldsymbol{\delta}_1^{(t+1)}$ by $0.05 \cdot \mathrm{sign}(\boldsymbol{W} * \nabla_{\boldsymbol{x}} H_3(\boldsymbol{x} + \boldsymbol{\delta}_1^{(t)}))$
  // chain-based perturbation for Eq. (5)
  Input $\boldsymbol{x}+\boldsymbol{\delta}_2^{(t)}$ to detector H_3 and obtaion the gradient $\boldsymbol{W} * \nabla_{\boldsymbol{x}} H_3(\boldsymbol{x}+\boldsymbol{\delta}_2^{(t)})$
  Update $\boldsymbol{\delta}_2^{(t+1)}$ by $0.05 \cdot \mathrm{sign}(\boldsymbol{W} * \nabla_{\boldsymbol{x}} H_3(\boldsymbol{x} + \boldsymbol{\delta}_2^{(t)}))$
end while
return perturbations $\boldsymbol{\delta}_1^{(19)}, \boldsymbol{\delta}_2^{(19)}$

We take the TIFGSM attack method as an example, and in Alg. 2, we show the process that TIFGSM iterates 20 times on the detectors to generate the initialization perturbation. In Alg. 2, $\boldsymbol{W}$ denotes the kernel matrix in TIFGSM attack. We can return results skip-based $\boldsymbol{\delta}_1$ and chain-based perturbations $\boldsymbol{\delta}_2$ as the mixed initial population with gradient-prior.

Algorithm 3 Random Subset Selection

Require: image width w, height h, channels c, skip-based perturbation $\boldsymbol{\delta}^1$, chain-based perturbation $\boldsymbol{\delta}^2$, random subset's size a

1: $\boldsymbol{s} \leftarrow$ array of zeros of size $w \times h \times c$
2: sample uniformly $r \in \{0, ..., w-a\}, s \in \{0, ..., h-a\}$
3: $\boldsymbol{s}_{r+1:r+a,s+1:s+a} = 1$
4: **while** $i < c$ **do**
5:   $\rho_1 \leftarrow Uniform(\{-1, 1\}), \rho_2 \leftarrow Uniform(\{-1, 1\})$
6:   $\boldsymbol{\delta}^1_{r+1:r+a,s+1:s+a} \leftarrow (\rho_1 \cdot \boldsymbol{\delta}^1)_{r+1:r+a,s+1:s+a}$
7:   $\boldsymbol{\delta}^2_{r+1:r+a,s+1:s+a} \leftarrow (\rho_2 \cdot \boldsymbol{\delta}^2)_{r+1:r+a,s+1:s+a}$
8: **end while**
9: return new perturbations $\boldsymbol{\delta}^1, \boldsymbol{\delta}^2$, and subset $\boldsymbol{s}$

In Alg. 3, we show random subset selection. In the first round, we sample subset in the search space via random search (line 2). In the second round, we record the sample subset (line 3) and alter perturbations $\boldsymbol{\delta}^1, \boldsymbol{\delta}^2$ (lines 6-7).

Algorithm 4 Divide-and-Conquer Algorithm

Require: skip-based perturbation $\boldsymbol{\delta}_1^0$, chain-based perturbation $\boldsymbol{\delta}_2^0$, maximum iterations T, victim detector H_1, ground-truth boxes $\mathcal{O}$, crossover rate cr, mutation rate mr, sub-component $\boldsymbol{s}$, clean image $\boldsymbol{x}$
Ensure: adversarial image $\hat{\boldsymbol{x}} \in \mathbb{R}^{w\times h\times c}$ with $||\boldsymbol{\delta}||_\infty \leq 0.05$, $\hat{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\delta}$

// genetic algorithm

2: $f_1^{(0)}, f_2^{(0)}, s_1^{(0)}, s_2^{(0)} \leftarrow F(H_1(\boldsymbol{x}), \boldsymbol{x}+\boldsymbol{\delta}_1), F(H_1(\boldsymbol{x}), \boldsymbol{x}+\boldsymbol{\delta}_2), S(\boldsymbol{\delta}_1[\boldsymbol{s}]), S(\boldsymbol{\delta}_2[\boldsymbol{s}])$ with ground-truth boxes $\mathcal{O}$ in Eq. (3) and Eq. (7).
3: $f_1^{bset}, f_2^{best}, \boldsymbol{\delta}_1^{best}[\boldsymbol{s}], \boldsymbol{\delta}_1^{best}[\boldsymbol{s}] \leftarrow f_1^0, f_2^0, \boldsymbol{\delta}_1^0[\boldsymbol{s}], \boldsymbol{\delta}_1^0[\boldsymbol{s}]$
4: $t \leftarrow 0$
5: **while** $t < T$ **do**
6:   **if** $\max\{s_1^{(t)}, s_2^{(t)}\} = \infty$ **then**
7:     break
8:   **end if**
9:   $\mathrm{loser}, \mathrm{winner} = \mathrm{sort\_by\_fitness}(\boldsymbol{\delta}_1^{(\mathrm{t})}[\mathrm{s}], \boldsymbol{\delta}_2^{(\mathrm{t})}[\mathrm{s}], \mathrm{s}_1^{(\mathrm{t})}, \mathrm{s}_2^{(\mathrm{t})})$
10:   $\boldsymbol{\delta}_1^{(t)}[\boldsymbol{s}], \boldsymbol{\delta}_2^{(t)}[\boldsymbol{s}] = \mathrm{crossover}(\mathrm{cr}, \mathrm{loser}, \mathrm{winner})$
11:   $\boldsymbol{\delta}_1^{(t)}[\boldsymbol{s}], \boldsymbol{\delta}_2^{(t)}[\boldsymbol{s}] = \mathrm{mutation}(\mathrm{mr}, \mathrm{loser}, \mathrm{winner})$
12:   $t \leftarrow t+1$
13:   $f_1^{(t)}, f_2^{(t)}, s_1^{(t)}, s_2^{(t)} \leftarrow F(H_1(\boldsymbol{x}), \boldsymbol{x}+\boldsymbol{\delta}_1^{(t)}), F(H_1(\boldsymbol{x}), \boldsymbol{x}+\boldsymbol{\delta}_2^{(t)}), S(\boldsymbol{\delta}_1^{(t)}[\boldsymbol{s}]), S(\boldsymbol{\delta}_2^{(t)}[\boldsymbol{s}])$ with ground-truth boxes $\mathcal{O}$ in Eq. (3) and Eq. (7).
14:   **if** $f_1^{(t)} > f_1^{(t-1)}$ **then**
15:     $f_1^{best} \leftarrow f_1^t, \boldsymbol{\delta}_1^{best}[\boldsymbol{s}] \leftarrow \boldsymbol{\delta}_1^t[\boldsymbol{s}]$
16:   **end if**
17:   **if** $f_2^{(t)} > f_2^{(t-1)}$ **then**
18:     $f_2^{best} \leftarrow f_2^t, \boldsymbol{\delta}_2^{best}[\boldsymbol{s}] \leftarrow \boldsymbol{\delta}_2^t[\boldsymbol{s}]$
19:   **end if**
20: **end while**
21: return best sub-components $\boldsymbol{\delta}_1^{best}[\boldsymbol{s}], \boldsymbol{\delta}_2^{best}[\boldsymbol{s}]$ and individual's fitness f_1^{best}, f_2^{best}

In Alg. 4, we show the genetic algorithm based on the divide-and-conquer algorithm. In line 6, it denotes there are no true positive or false positive objects in $\boldsymbol{s}$. Thus, the sub-component do not need to be improved. We sort the fitness of sub-components in line 9 and alter sub-components by crossover and mutation in lines 10-11. Since object detection is holistic, we return the best fitness of individuals and corresponding sub-components over the entire process.

B Proof A Lower Bound on The P Value

In Section 3.2, we set the p-th individual fitness $P(\boldsymbol{x}+\boldsymbol{\delta}_p) = F(\boldsymbol{x}+\boldsymbol{\delta}_p)$. Thus, proof a lower bound on the P value is same as on the F value. We firstly give two notions of 'approximate submodularity', which measure to the extent of a set function F has the submodular property. The γ- and α-submodularity as follows:

Definition 1 γ-Submodularity Ratio [1]. The submodularity ratio of a set function $F : 2^{\boldsymbol{s}} \rightarrow \mathbb{R}$ with respect to a set $\boldsymbol{u} \subseteq \boldsymbol{s}$ and a paremeter $l \geq 1$ is

$$\gamma_{\boldsymbol{u},l}(F) = \min_{\boldsymbol{l} \subseteq \boldsymbol{u}, \boldsymbol{m}: |\boldsymbol{m}| \leq l, \boldsymbol{m} \cap \boldsymbol{l} = \emptyset} \frac{\sum_{v \in \boldsymbol{m}} (F(\boldsymbol{l} \cup v) - F(\boldsymbol{l}))}{F(\boldsymbol{l} \cup \boldsymbol{m}) - F(\boldsymbol{l})} \tag{10}$$

Definition 2 α-Submodularity Ratio [2]. The submodularity ratio of a set function $F : 2^{\boldsymbol{s}} \to \mathbb{R}$ is

$$\alpha_F = \min_{\boldsymbol{u}\subseteq\boldsymbol{m}\subseteq\boldsymbol{s}, v\notin\boldsymbol{m}} \frac{F(\boldsymbol{s}\cup v) - F(\boldsymbol{s})}{F(\boldsymbol{m}\cup v) - F(\boldsymbol{m})} \tag{11}$$

The subset selection problem as follows:

Definition 3 Subset Selection. Given all items $\boldsymbol{s} = \{s_1, ..., s_n\}$, an objective function F and a budget z. we aim to find a subset of most z items maximizing F, i.e.,

$$\arg\max_{\boldsymbol{u}\subseteq\boldsymbol{s}} F(\boldsymbol{u}), s.t. |\boldsymbol{u}| \leq z \tag{12}$$

where $|\cdot|$ denotes the size of a set.

Now, we will proof the Eq. (8). We first prove that $\max\{P(\boldsymbol{\delta}[\boldsymbol{b}_j])|1 \leq j \leq i\} \geq \frac{\alpha}{i}OPT$. The $\boldsymbol{b}$ denotes an optimal subset of $\boldsymbol{s}$, i.e., $P(\boldsymbol{\delta}[\boldsymbol{b}]) = OPT$. For $1 \leq j \leq i$, Let $\boldsymbol{a}_j = \boldsymbol{b} \cap \boldsymbol{s}_j$. Thus, $\cup_{j=1}^{i}\boldsymbol{a}_j = \boldsymbol{b}$ and for any $j \neq m$, $\boldsymbol{a}_j \cap \boldsymbol{a}_m = \emptyset$. Then, we have

$$P(\boldsymbol{\delta}[\boldsymbol{b}]) = P(\boldsymbol{\delta}[\cup_{j=1}^{i}\boldsymbol{a}_j]) = \sum_{j=1}^{i} P(\boldsymbol{\delta}[\cup_{m=1}^{j}\boldsymbol{a}_m]) - P(\boldsymbol{\delta}[\cup_{m=1}^{j-1}\boldsymbol{a}_m]) \tag{13}$$

The set $\{s_1^j, ..., s_{|\boldsymbol{a}_j|}^j\}$ denotes the items in $\boldsymbol{a}_j$. Then, for any $1 \leq j \leq i$, it holds that

$$\begin{aligned} &P(\boldsymbol{\delta}[\cup_{m=1}^{j}\boldsymbol{a}_m]) - P(\boldsymbol{\delta}[\cup_{m=1}^{j-1}\boldsymbol{a}_m]) \\ &\quad= \sum_{l=1}^{|\boldsymbol{a}_j|} P(\boldsymbol{\delta}[\cup_{m=1}^{j-1}\boldsymbol{a}_m \cup \{s_1^j, ..., s_l^j\}]) - P(\boldsymbol{\delta}[\cup_{m=1}^{j-1}\boldsymbol{a}_m \cup \{s_1^j, ..., s_{l-1}^j\}]) \\ &\quad\leq \frac{1}{\alpha}\sum_{l=1}^{|\boldsymbol{a}_j|} P(\boldsymbol{\delta}[\{s_1^j, ..., s_l^j\}]) - P(\boldsymbol{\delta}[\{s_1^j, ..., s_{l-1}^j\}]) = \frac{P(\boldsymbol{\delta}[\boldsymbol{a}_j])}{\alpha}, \end{aligned} \tag{14}$$

where the inequality is by the definition of α-submodularity ratio since $\{s_1^j, ..., s_{l-1}^j\} \subseteq \cup_{m=1}^{j-1}\boldsymbol{a}_m \cup \{s_1^j, ..., s_{l-1}^j\}$. Note that for any $1 \leq j \leq i, P(\boldsymbol{\delta}[\boldsymbol{b}_j]) \geq P(\boldsymbol{\delta}[\boldsymbol{a}_j])$, since $\boldsymbol{a}_j \subseteq \boldsymbol{s}_j$ and $|\boldsymbol{a}_j| \leq |\boldsymbol{b}| \leq z$. Thus we get

$$OPT = P(\boldsymbol{a}[\boldsymbol{b}]) \leq \frac{1}{\alpha}\sum_{j=1}^{i} P(\boldsymbol{a}_j) \leq \frac{1}{\alpha}\sum_{j=1}^{i} P(\boldsymbol{b}_j) \tag{15}$$

which leads to $\max\{P(\boldsymbol{\delta}[\boldsymbol{b}_j])|1 \leq j \leq i\} \geq \frac{\alpha}{i}OPT$.

We then prove that $\max\{P(\boldsymbol{\delta}[\boldsymbol{b}_j])|1 \leq j \leq i\} \geq \frac{\gamma_{\emptyset,z}}{z}OPT$. By the definition of γ-submodularity ratio, $P(\boldsymbol{\delta}[\boldsymbol{b}]) \leq \sum_{s\in\boldsymbol{b}} F(\boldsymbol{\delta}[s])/\gamma_{\emptyset,z}$. Let $s^* \in \arg\max_{s\in\boldsymbol{b}} P(\boldsymbol{\delta}[s])$, and $\{\boldsymbol{s}_1, ..., \boldsymbol{s}_i\}$ is a partition set of $\boldsymbol{s}$, s^* must belong to one of the j-th sets. Thus $\max\{P(\boldsymbol{\delta}[\boldsymbol{b}_j])|1 \leq j \leq i\} \geq \frac{\gamma_{\emptyset,z}}{z}OPT$. We put z into the above equation and Eq. (15) and proof the Eq. (8)

C Proof Approximation Guarantee

For any $\boldsymbol{u} \subseteq \boldsymbol{s}_j (1 \le j \le i)$, there exits one item $s \in \boldsymbol{s}_j/\boldsymbol{u}$ such that [1]

$$P(\boldsymbol{\delta}[\boldsymbol{u} \cup s]) - P(\boldsymbol{\delta}[s]) \ge (\gamma_{\boldsymbol{u},z}/z) * (P(\boldsymbol{\delta}[\boldsymbol{b}_j]) - P(\boldsymbol{\delta}[s])) \tag{16}$$

In above section, we analyze the maximum number of iterations on each machine until $P(\boldsymbol{\delta}[\boldsymbol{u}]) \ge (1 - e^{-\gamma_{min}}) \cdot P(\boldsymbol{\delta}[\boldsymbol{o}_j])$. For the machine running DS on $\boldsymbol{s}_j$, let J^j_{max} denote the maximum value of $m \in \{1, ..., z\}$ such that in the archive Ω, there exists a solution $\boldsymbol{u}$ with $\boldsymbol{u} \le j$ and $P(\boldsymbol{\delta}[\boldsymbol{u}]) \ge (1-(1-\frac{\gamma_{min}}{z})) \cdot P(\boldsymbol{\delta}[\boldsymbol{o}_j])$. That is,

$$J^j_{max} = max\{m \in \{1, ..., k\} | \exists \boldsymbol{u} \in R, |\boldsymbol{u}| < k \wedge P(\boldsymbol{\delta}[\boldsymbol{u}]) \ge (1-(1-\frac{\gamma_{min}}{z})) \cdot P(\boldsymbol{\delta}[\boldsymbol{o}_j])\} \tag{17}$$

Then, we only need to analyze the maximum iterations until $\min\{J^j_{max} | 1 \le j \le i\} = z$. For GA running on $\boldsymbol{s}_j$, let $\boldsymbol{z}_j$ be a corresponding solution with the value J^j_{max}. Because $\boldsymbol{z}_j$ is weakly dominated by the newly generated solution, the J^j_{max} can not decrease. In Eq. (16), we know that for any $1 \le j \le i$, change specific bit of $\boldsymbol{z}_j$ can generate a new solution $\boldsymbol{s}^{new}_j$ and satisfy that $P(\boldsymbol{\delta}[\boldsymbol{s}^{new}_j]) - P(\boldsymbol{\delta}[\boldsymbol{z}_j]) \ge \frac{\gamma_{\boldsymbol{z}_j,z}}{z}(P(\boldsymbol{\delta}[\boldsymbol{b}_j]) - P(\boldsymbol{\delta}[\boldsymbol{z}_j]))$. Then, if $J^j_{max} < z$, we have

$$P(\boldsymbol{\delta}[\boldsymbol{s}^{new}_j]) \ge (1 - (1 - \gamma_{min}/z)^{J^j_{max}+1}) \cdot P(\boldsymbol{\delta}[\boldsymbol{b}_j]) \tag{17}$$

where $\gamma_{\boldsymbol{z}_j,z} > \gamma_{min}$. Since $|\boldsymbol{s}^{new}_j| = |\boldsymbol{z}_j| + 1 \le J^j_{max} + 1$, $\boldsymbol{s}^{new}_j$ will be included into Ω. If $\boldsymbol{s}^{new}_j$ may be dominated by one solution in Ω, which are contradictory with the J^j_{max}. Due to that $\boldsymbol{s}^{new}_j$, J^j_{max} increases by at least 1 and the $\Omega_{\max}$ denotes the largest size of Ω in the Alg. 4, the J^j_{max} can increase by at least 1 in one iteration with probability $\frac{1}{en_j\Omega_{\max}}$. The $\frac{1}{\Omega_{\max}}$ is a lower bound on the crossover probability of selecting $\boldsymbol{z}_j$ and n_j is the mutation rate. Because that solution limitation, we have $\Omega_{\max} \le$ 2z. We the get that after one iteration in the first around DC, l can decrease by at least 1 with probability as least

$$1 - \prod_{j:J^j_{max}=m} (1 - \frac{1}{2ezn_m}) \ge 1 - (1 - \frac{1}{2ezn_{max}})^l \tag{18}$$

since it is sufficient that at least one of those $J^j_{max} = m$ increases. Thus, the expected number of iterations until m increases

$$\sum_{l=1}^{i} \frac{1}{1-(1-\frac{1}{2ezn_{max}})^l} \le i + (2ezn_{max} - 1)H_i \tag{19}$$

where H_i is the j-th harmonic number. Then the complexity is $O(z^2 n_{max}(1 + \log i))$. Since $\min\{J^j_{max} | 1 \le j \le i\} = z$ implies that $P(\boldsymbol{\delta}[\boldsymbol{u}]) \ge (1 - e^{-\gamma_{min}}) \cdot P(\boldsymbol{\delta}[\boldsymbol{o}_j])$ for any $1 \le j \le i$, the P value of the final output subset satisfies that $\max\{P(\boldsymbol{\delta}[\boldsymbol{u}_j]) | 1 \le j \le i\} \ge (1 - e^{-\gamma_{min}}) * \max\{P(\boldsymbol{\delta}[\boldsymbol{b}_j]) | 1 \le j \le i\}$. By Lemma 1, Eq. (9) holds.

References

1. Das, A., Kempe, D.: Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. arXiv preprint arXiv:1102.3975 (2011)
2. Qian, C., Shi, J.C., Yu, Y., Tang, K., Zhou, Z.H.: Parallel pareto optimization for subset selection. In: IJCAI. pp. 1939–1945 (2016)