HNCO
Empirical runtime analysis of (1+1) EA and RLS on OneMax and LeadingOnes

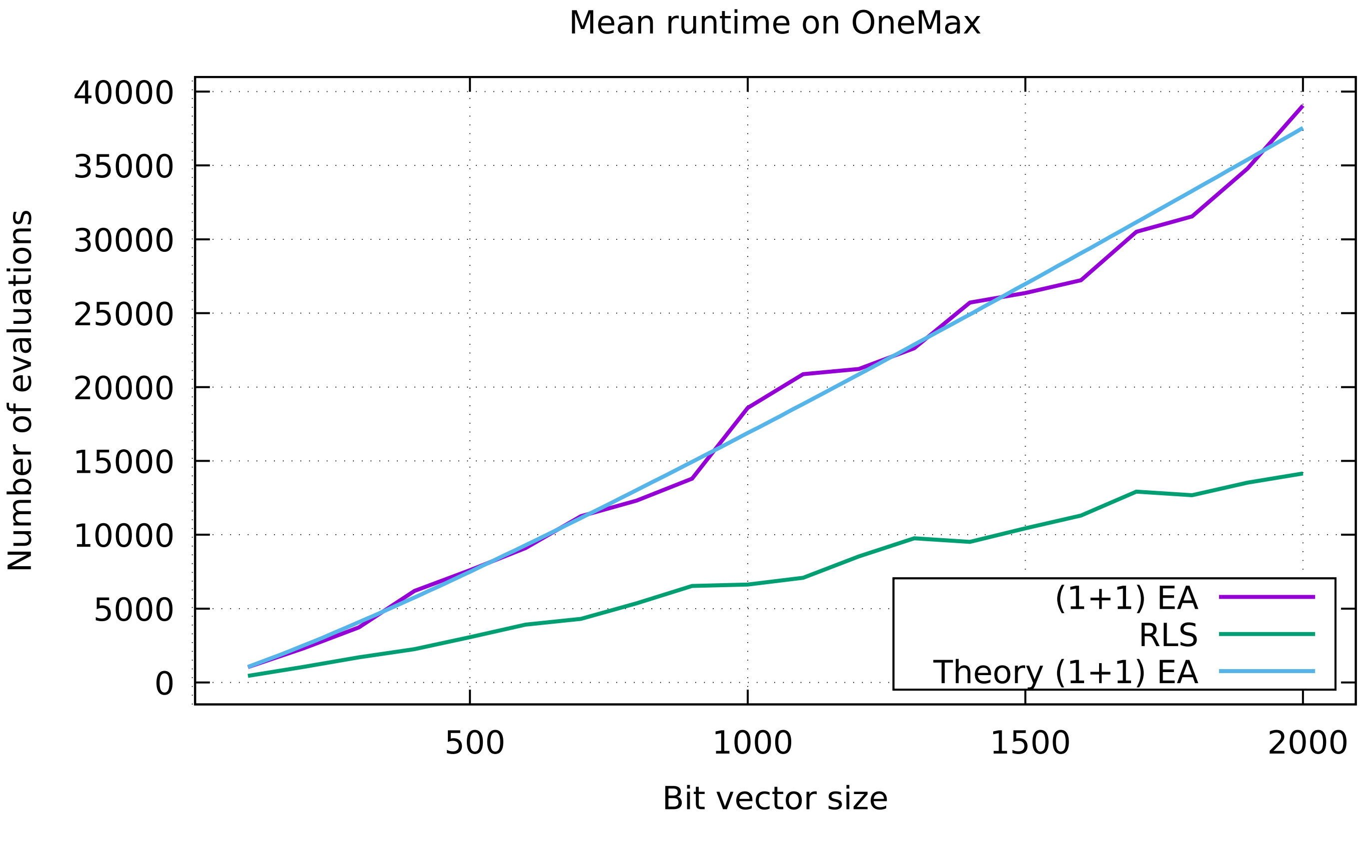
October 12, 2022

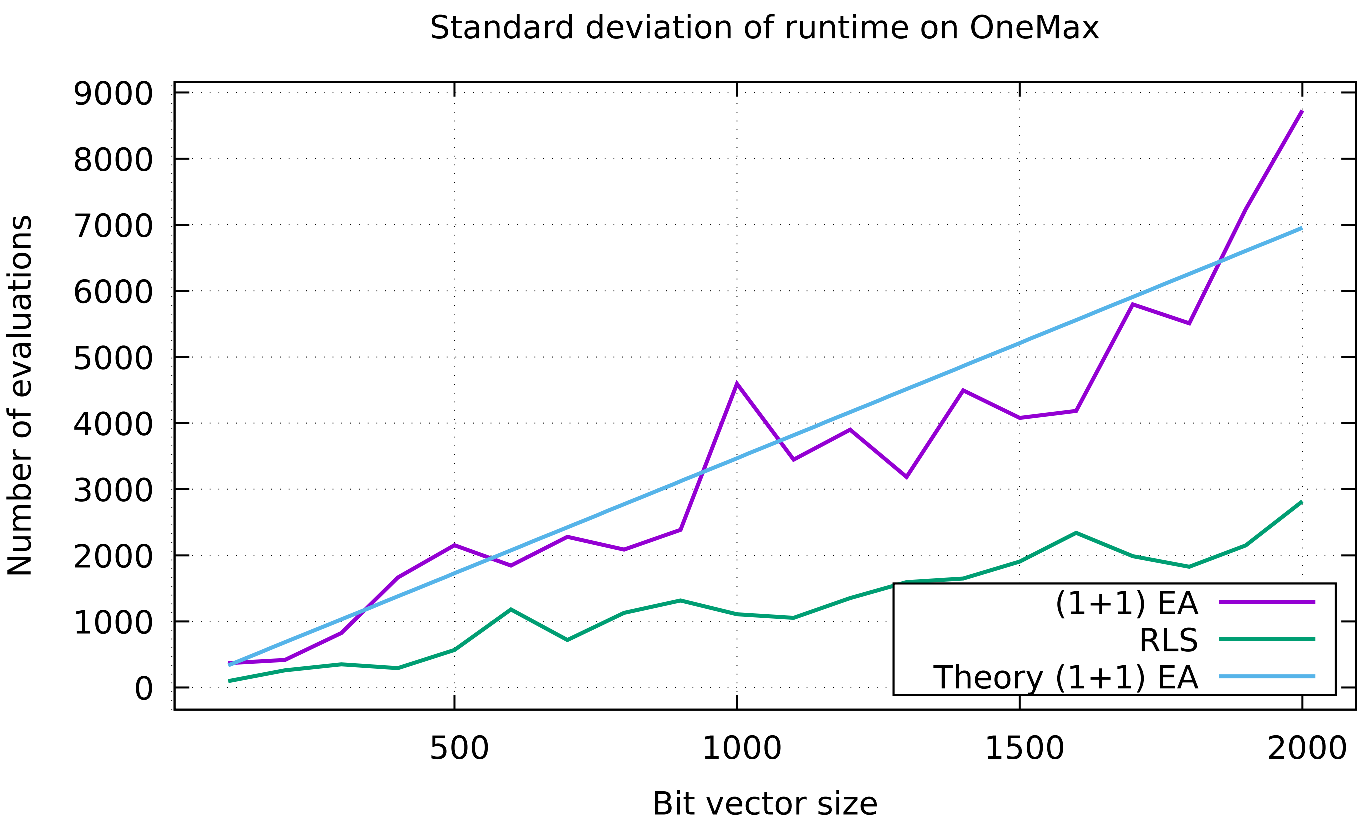
Abstract

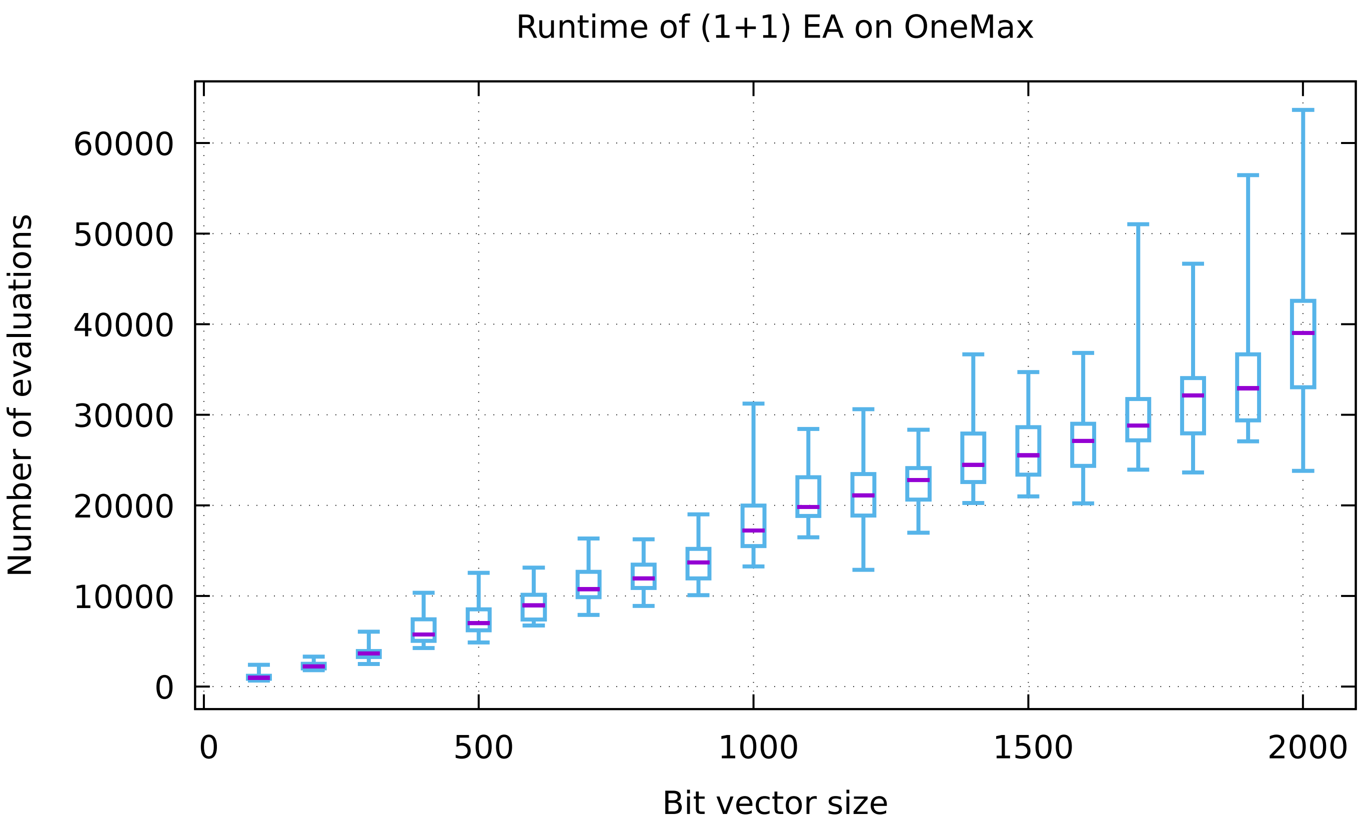
This report contains the results of an empirical runtime analysis of (1+1) EA and RLS on OneMax and LeadingOnes performed with HNCO. The added theoretical results come from [1]. We observe a discrepancy between theory and practice in the case of the standard deviation of the runtime of (1+1) EA applied to LeadingOnes. In this case, the expression is exact only up to a $O(n)$ term, which might explain the discrepancy.

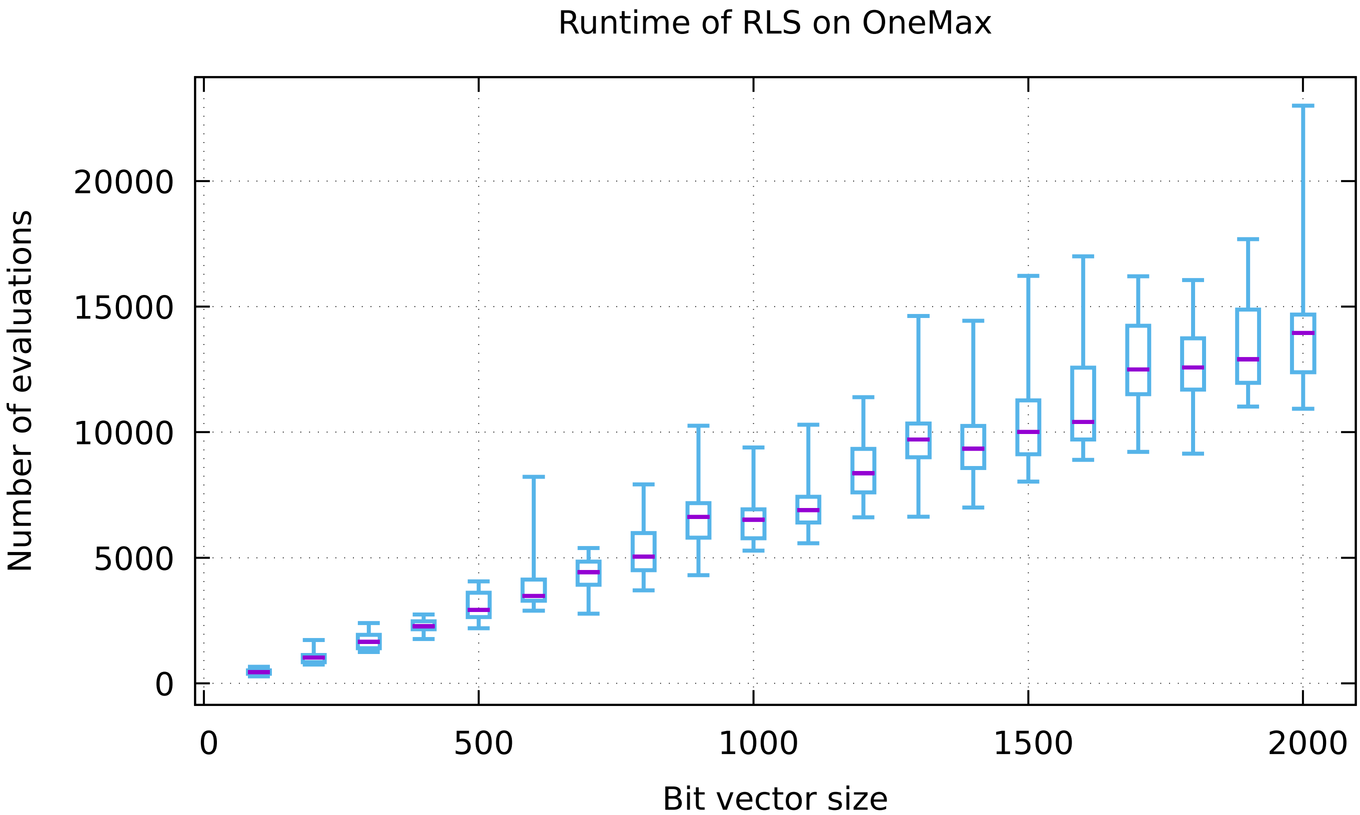
Contents

1 Function OneMax

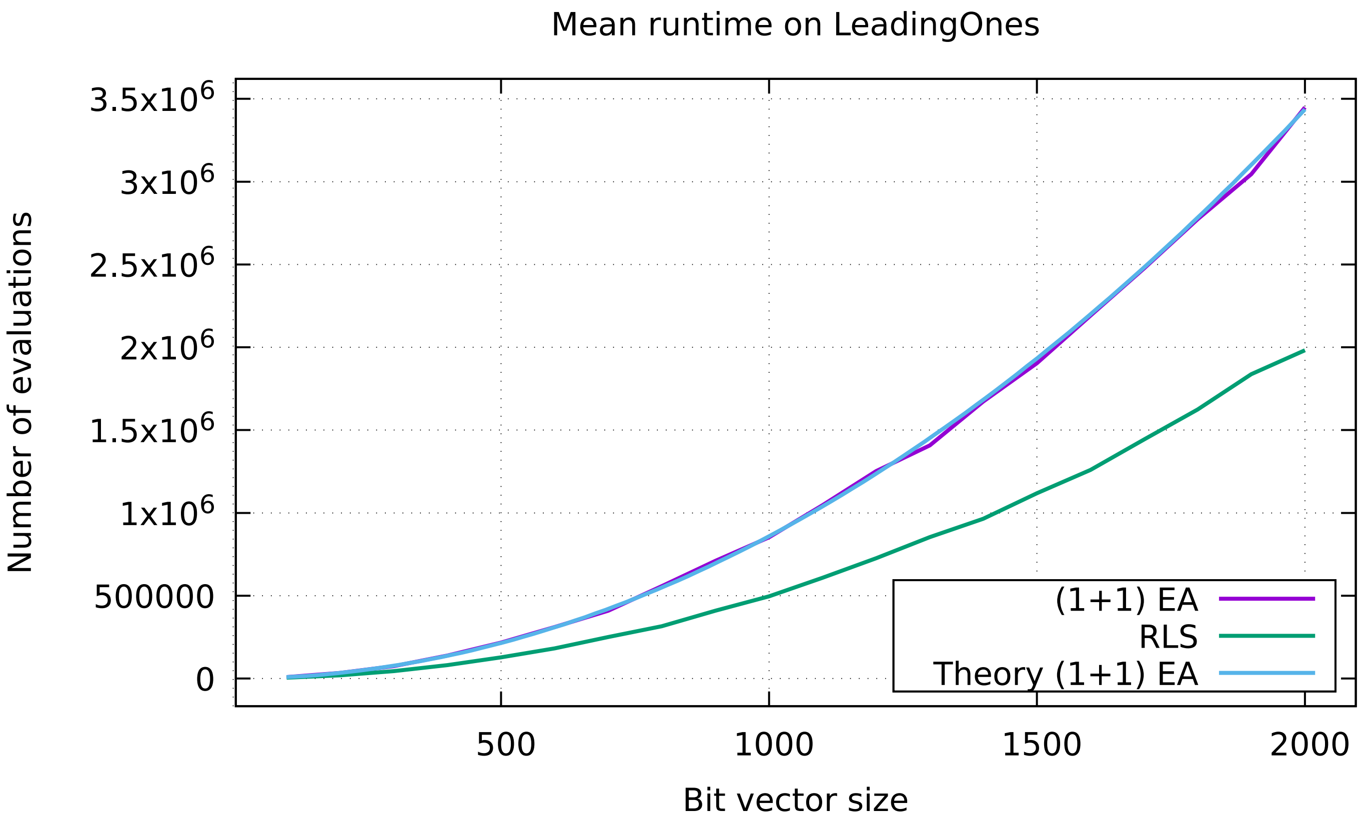


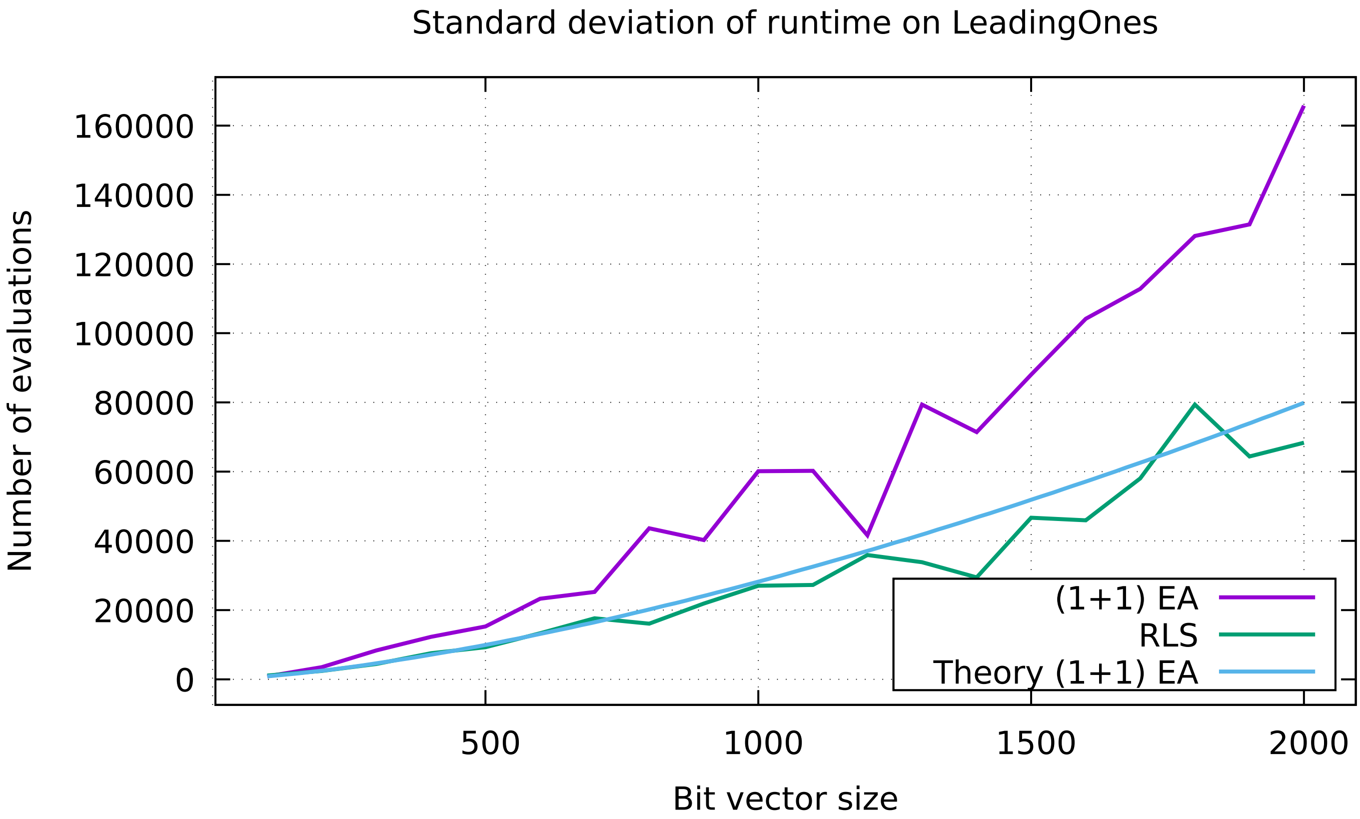

Standard deviation of runtime on OneMax
Number of evaluations
Bit vector size
(1+1) EA
RLS
Theory (1+1) EA
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
500
1000
1500
2000

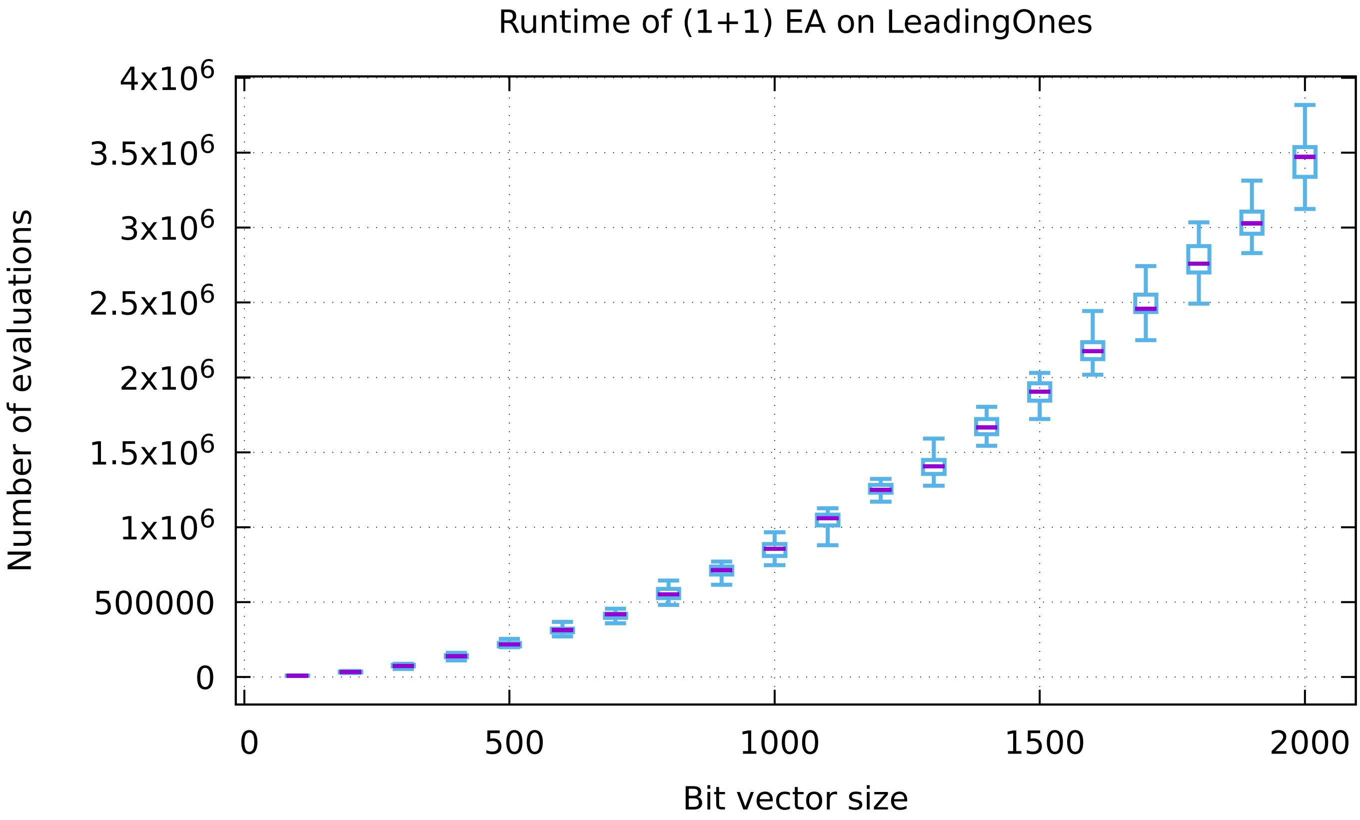

Runtime of (1+1) EA on OneMax
Number of evaluations
Bit vector size
60000
50000
40000
30000
20000
10000
0
0
500
1000
1500
2000

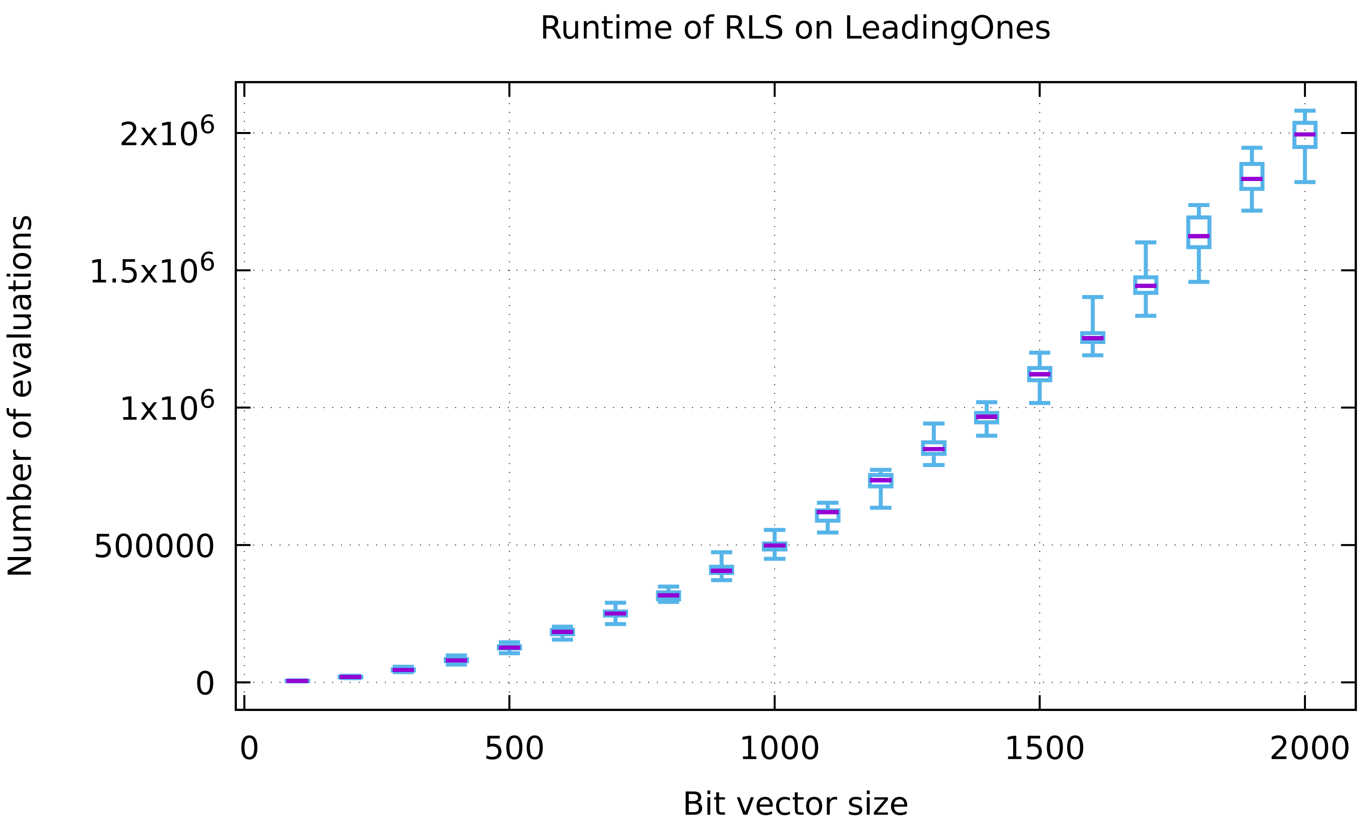


2 Function LeadingOnes




Standard deviation of runtime on LeadingOnes
Number of evaluations
160000
140000
120000
100000
80000
60000
40000
20000
0
500
1000
1500
2000
Bit vector size
(1+1) EA
RLS
Theory (1+1) EA


Runtime of (1+1) EA on LeadingOnes
Number of evaluations
4x10^6
3.5x10^6
3x10^6
2.5x10^6
2x10^6
1.5x10^6
1x10^6
500000
0
0
500
1000
1500
2000
Bit vector size



A Plan

```
{
    "exec": "hnco",
    "opt": "--print-results --stop-on-maximum",
    "budget": 0,
    "num_runs": 20,
    "parallel": true,
    "parameter": {
        "id": "bv-size",
        "label": "Bit vector size",
        "values_perl": "map { 100 + 100 * $_ } (0 .. 19)"
    },
    "graphics": {
        "mean": {
            "title": true
        },
        "stddev": {
            "title": true
        },
        "candlesticks": {
            "title": true,
            "boxwidth": 40
        }
    },
    "functions": [
        {
            "id": "one-max",
            "label": "OneMax",
            "opt": "-F 0 --incremental-evaluation",
            "logscale": false,
            "mean_gnuplot": [
                {
                    "expression": "f(x) = exp(1) * (x + 1/2) * log(x) - 1.8925417883 * x",
```

```
                    "label": "Theory (1+1) EA"
                }
            ],
            "stddev_gnuplot": [
                {
                    "expression": "f(x) = sqrt(pi**2 / 6 * (exp(1) * x)**2 - (2 * exp(1) + 1) * exp(1) * x * log(x))",
                    "label": "Theory (1+1) EA"
                }
            ]
        },
        {
            "id": "leading-ones",
            "label": "LeadingOnes",
            "opt": "-F 10",
            "logscale": false,
            "mean_gnuplot": [
                {
                    "expression": "f(x) = (exp(1) - 1) / 2 * x**2",
                    "label": "Theory (1+1) EA"
                }
            ],
            "stddev_gnuplot": [
                {
                    "expression": "f(x) = sqrt((exp(2) - 1) / 8 * x**3 + (-3 * exp(2) - 8 * exp(1) + 17) / 16 * x**2)",
                    "label": "Theory (1+1) EA"
                }
            ]
        }
    ],
    "algorithms": [
        {
            "id": "ea-1p1",
            "label": "(1+1) EA",
            "opt": "-A 300 --ea-allow-no-mutation"
        },
        {
            "id": "rls",
            "label": "RLS",
            "opt": "-A 100 --rls-patience 0"
        }
    ]
}
```

B Default parameters

```
# algorithm = 100
# bm_mc_reset_strategy = 1
# bm_num_gs_cycles = 1
# bm_num_gs_steps = 100
# bm_sampling = 1
# budget = 10000
# bv_size = 100
# description_path = description.txt
# ea_crossover_bias = 0.5
# ea_crossover_probability = 0.5
# ea_lambda = 100
# ea_mu = 10
# ea_mutation_rate = 1
# ea_mutation_rate_max = 1
# ea_mutation_rate_min = 0.01
```

```
# ea_success_ratio = 4
# ea_tournament_size = 2
# ea_update_strength = 1.01
# expression = x
# fn_name = noname
# fn_num_traps = 10
# fn_prefix_length = 2
# fn_threshold = 10
# fp_expression = (1-x)^2+100*(y-x^2)^2
# fp_lower_bound = -2
# fp_num_bits = 8
# fp_precision = 0.01
# fp_upper_bound = 2
# function = 0
# hea_reset_period = 0
# learning_rate = 0.001
# map = 0
# map_input_size = 100
# map_path = map.txt
# map_ts_length = 10
# map_ts_sampling_mode = 0
# neighborhood = 0
# neighborhood_iterator = 0
# noise_stddev = 1
# num_iterations = 0
# num_threads = 1
# path = function.txt
# pn_mutation_rate = 1
# pn_neighborhood = 0
# pn_radius = 2
# population_size = 10
# pv_log_num_components = 5
# radius = 2
# rep_categorical_representation = 0
# rep_num_additional_bits = 2
# results_path = results.json
# rls_patience = 50
# sa_beta_ratio = 1.2
# sa_initial_acceptance_probability = 0.6
# sa_num_transitions = 50
# sa_num_trials = 100
# seed = 0
# selection_size = 1
# solution_path = solution.txt
# target = 100
# last_parameter
# exec_name = unknown
# version = 0.22
# Generated from hnco.json
```

References

[1] Hsien-Kuei Hwang, Alois Panholzer, Nicolas Rolin, Tsung-Hsi Tsai, and Wei-Mei Chen. Probabilistic analysis of the (1+1)-evolutionary algorithm. *Evolutionary computation*, 26(2):299–345, 2018.