Using non-parametric statistical tests to compare solutions in evolutionary framework for maintenance schedule optimisation

Benjamin Lacroix, John McCall, Jérôme Lonchampt

Robert Gordon University, EDF'Lab

13th November, 2017

Outline

Outline

1. Maintenance schedule optimisation
2. Proposed Method
   - Using statistical tests in EAs
   - Racing Algorithm
3. Experiments
   - Experimental setup
   - Results
4. Going further

Simulation-optimisation for maintenance scheduling

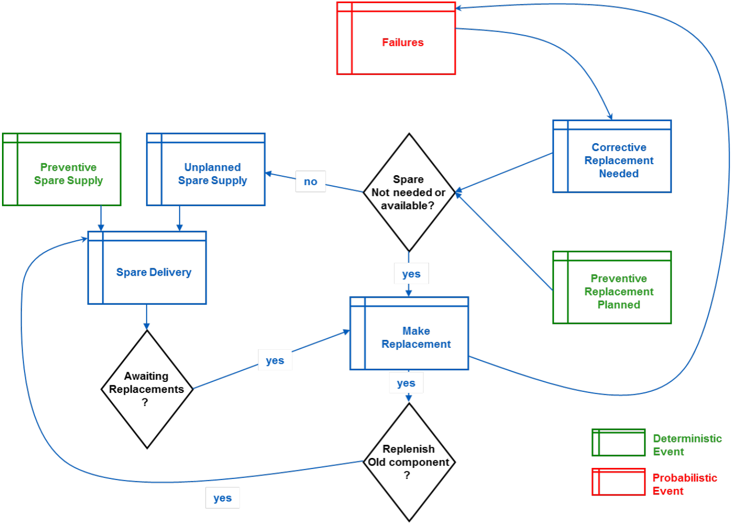
The problem:

- Given a set of c components, find the maintenance dates for each components $X = \{x_1, ...x_c\} \in \mathbb{R}^c$ to maximise the Net Present Value

You need two things:

- A realistic model to simulate the life cycle of the system/asset/components
- An good optimisation algorithm

Event Model



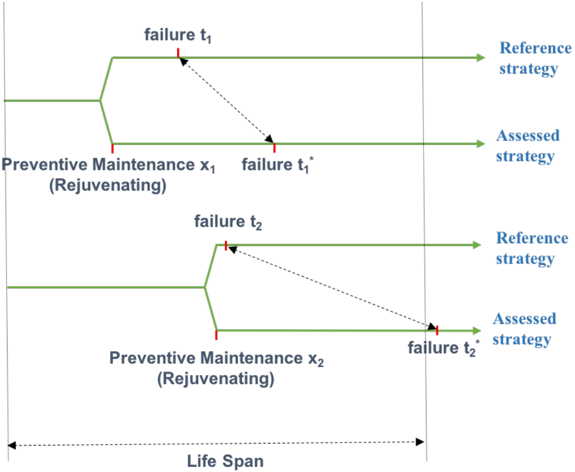
Failures
Preventive Spare Supply
Unplanned Spare Supply
no
Spare Not needed or available?
Corrective Replacement Needed
Spare Delivery
yes
Preventive Replacement Planned
Make Replacement
yes
Awaiting Replacements ?
yes
Replenish Old component ?
yes
Deterministic Event
Probabilistic Event

Instance definition

- Number of components, c.
- Weibull distribution parameters: $\{\lambda, \beta, \gamma\}^c$.
- Stock level.
- Yearly cost of unavailability.
- Failure cost.
- Maintenance cost.
- Spare part cost.
- Storage cost.
- Time to buy a spare part after failure.
- Preventive maintenance time.

Calculating NPV

Given a set of failure times obtained from the failure probabilities of each component



Outline

The situation

What we have:

- Optimisation algorithm based on the comparison of solutions using deterministic cost function.
- A stochastic cost function.

What can be done:

- Use MonteCarlo simulation and use mean or median to evaluate solutions:
  - Problem: how many replications do we need to have a robust comparison?
- Use surrogate models to approximate the stochastic code
  - Problem: To what extent can we trust our surrogate model?

Outline

Using statistical tests to compare solutions

- Since the cost function returns a random sample, we can apply statistical tests.
- Sample size can be augmented until significance is found.
- Eliminates the bias of the arbitrary number of replications.

Non-parametric statistical tests

- Let us consider two of them: Friedman and Wilcoxon test
- Comparison on related samples
- In this case lets consider:
  - a set n failure scenarios $S = \{s_1, ...s_n\}$ each of which is defined by the failure dates of each components.
  - a set of k maintenance schedule $P = \{p_1, ...s_k\}$
  - the NPV of each schedule p_j can be evaluated on each failure scenario s_i by $f(p_j, s_i)$.
- Every comparisons of p_j is performed on the same failure scenarios
- Purposes:
  - Remove the bias of comparing solutions on different failure scenarios
  - Use the only number of simulations necessary to decide whether a solution should be maintained in the population or not.

Friedman's Test

- Answers the question: "in a set of k sample, do at least two of the samples represent populations with different median values?"
- Given a matrix of results $\{f(p_i, s_j)\}_{n \times k}$
- Replace the results by the ranking of each p_i on each s_i, giving a matrix of ranks $\{r_{ij}\}_{n \times k}$
- Apply the Friedman's test to detect statistical difference
- If statistical differences, apply Holm's procedure to identify statistical differences on ranks between the highest ranked p_i and the rest of the population.

Wilcoxon test

Procedure

- Also known as signed-rank test.
- Comparison on each pair of instances (failure scenarios) of two related samples.

Given two maintenance schedule p_1 and p_2 simulated on a n failure scenarios giving two samples $\{f(p_1, s_1), ...f(p_1, s_n)\}$ and $\{f(p_2, s_1), ...f(p_2, s_n)\}$

1. Calculate each paired difference, $d_i = f(p_1, s_i) - f(p_2, s_i)$
2. Rank the d_i ignoring the signs: assign rank 1 to the smallest $|d_i|$, rank 2 to the next etc. using fractional ranking.
3. Label each rank with its sign, according to the sign of d_i.
4. Calculate R^+, the sum of the ranks of the positive d_i and R^-, the sum of the ranks of the negative d_i
5. Choose $R = min(R^+, R^-)$
6. Calculate $p - value$ using normal approximation

Outline

Racing Algorithm

Origins:

- Initially proposed in machine learning to test on data
- Further used in parameter tuning (IRACE)

Idea:

- Perform selection over a population
- Iteratively test solutions on instances until statistical difference is reached

Racing example

S	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	T
s_1	●	●	●	●	●	●	●	●	=
s_2	●	●	●	●	●	●	●	●	=
s_3	●	●	●	●	●	●	●	●	=
s_4	●	●	●	●	●	●	●	●	=
s_5	●	●	●	●	●	●	●	●	-
s_6	●	●	●	●	●				-
s_7	●	●	●	●					=
s_8	●	●	●	●					=
s_9	●	●	●	●					-
s_{10}	●	●	●						=
s_{11}	●	●	●						=
s_{12}	●	●	●						-
s_{13}	●	●							=
s_{14}	●	●							

Racing pseudo-code

```
Generate initial population at random P = {p_1, ...p_k}
Generate set of failure scenarios S = {s_1, ...s_n}
λ = μk
while !termination criterion not reached do
    while k > t do
        i = i + 1
        for j in 1 to k do
            Evaluate x_ij = f(p_j, s_i)
        end for
        Perform Holm's procedure on {x_ij}_ixk
        Remove from P all individuals p_j if significantly worse than the "best" individual in P
        k = |P|
    end while
    Generate new population from P
end while
```

Outline

Outline

Instance generation

- Number of components $c = \{4, 8, 12, 16, 20, 40\}$
- Weibull distribution parameters:
  - $\Lambda = \{\lambda_1, ..., \lambda_c\} \in [0.01, 0.1]^c$
  - $B = \{\beta_1, ..., \beta_c\} \in [1, 4]^c$
  - $\Gamma = \{\gamma_1, ..., \gamma_c\}$ always equal to 0
- Components age at $t = 0$: 0
- stock level = 0.25c
- Yearly cost of unavailable = 10000
- Failure cost $C_{failure} = 1000$
- Maintenance cost $C_{maintenance} = 100$
- Spare part cost $C_{spare} = 10$
- Storage cost $C_{storage} = 1$
- Time to buy a spare part after failure $t_{failure} = 1$
- Preventive maintenance time $t_{maintenance} = 1$

Experiments

Benchmark:

- 25 instances in each dimensions, generating random failure probabilities.
- Each instance is ran 25 times with different seeds (using the same failure scenarios in each run)
- A budget of 500K simulations

Racing algorithm tested:

- Univariate EDA:
  - Generate new solutions from the $x_i = \mathcal{N}(\mu_i, \sigma_i^2)$
- Statistical tests:
  - Friedman and Wilcoxon
  - Significance level: $\alpha = \{0.05, 0.1\}$
- Population size : 50
- Elitism : $\lambda = 0.5$
- Each instance is ran 25 times with different seeds (using the same failure scenarios in each run)

Outline

Comparison of the statistical tests

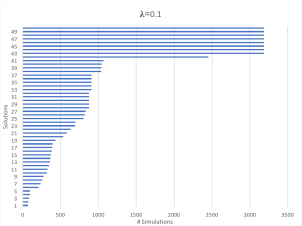
Table: Mean ranking of each statistical tests over Friedman's $p - value =$ 1.48E-10

Test	Ranking
wilcoxon-0.05	2.66
wilcoxon-0.1	3.89
friedman-0.05	1.71
friedman-0.1	1.72

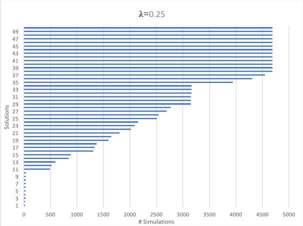
Table: Holm / Hochberg Table for $\alpha = 0.05$

i	algorithm	$z = (R_0 - R_i)/SE$	p	Holm/Hochberg/Hommel
3	wilcoxon-0.1	14.62	1.98E-48	0.016
2	wilcoxon-0.05	6.39	1.60E-10	0.025
1	friedman-0.1	0.089	0.93	0.05

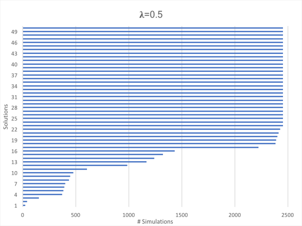
Racing effect



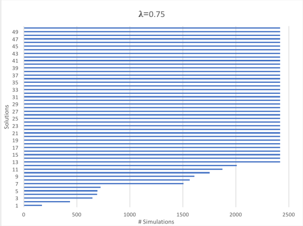
(a) $\lambda = 0.1$, gain $= 0.325$



(b) $\lambda = 0.25$, gain $= 0.555$



(c) $\lambda = 0.5$, gain $= 0.755$



(d) $\lambda = 0.75$, gain $= 0.891$

Outline

Statistical tests in Steady-State EA

- One on one comparison: using Wilcoxon test
- Risk of comparing two similar solutions with no statistical different
  - Waste of simulations to compare equivalent solutions
  - When to stop in the case of equivalent solution? (back to the original problem
- Which replacement strategy do you use?

Thanks!
Questions?
Code?
b.m.e.lacroix@rgu.ac.uk