Characterising and Improving LVA Behaviour

Advanced aspects of empirical analysis include:

- the analysis of asymptotic and stagnation behaviour,
- the use of functional approximations to mathematically characterise entire RTDs.

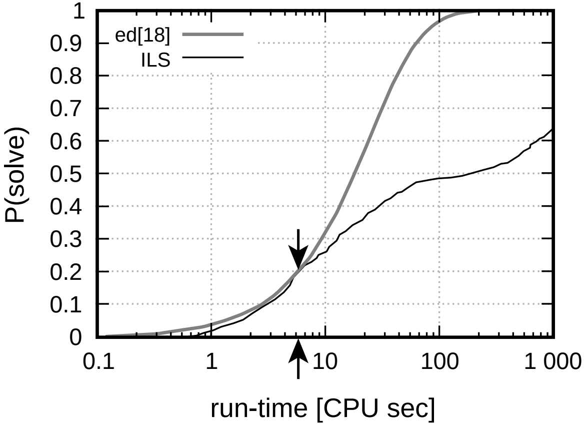
Such advanced analyses can facilitate improvements in the performance and run-time behaviour of a given LVA, *e.g.*, by providing the basis for

- designing or configuring *restart strategies* and other *diversification mechanisms*,
- realising speedups through *multiple independent runs parallelisation*.

LVA efficiency and stagnation

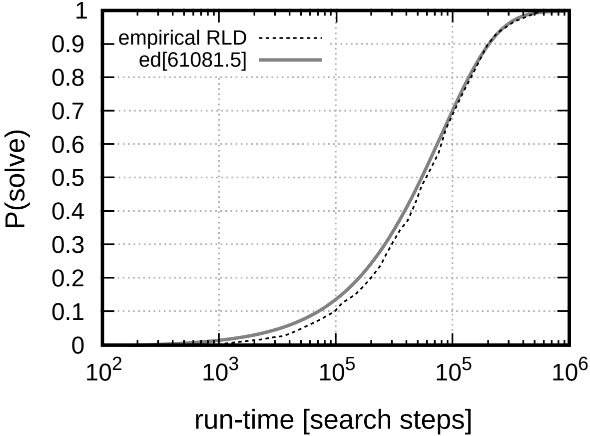
- In practice, the *rate of decrease in the failure probability*, $\lambda_{A,\pi}(t)$, is more relevant than true asymptotic behaviour.
- **Note:** *Exponential RTDs* are characterised by a *constant rate of decrease in failure probability*.
- A drop in $\lambda_{A,\pi}(t)$ indicates *stagnation* of algorithm A's progress towards finding a solution of instance π.
- Stagnation can be detected by comparing the RTD against an exponential distribution.

Evidence of stagnation in an empirical RTD:



'ed[18]' is the CDF of an exponential distribution with median 18; the arrows mark the point at which stagnation behaviour becomes apparent.

Approximation of an empirical RTD with an exponential distribution $ed[m](x) := 1 - 2^{-x/m}$:



The optimal fit exponential distribution obtained from the Marquardt-Levenberg algorithm passes the χ^2 goodness-of-fit test at $\alpha = 0.05$.

Performance improvements based on static restarts (1)

- Detailed RTD analyses can often suggest ways of improving the performance of a given SLS algorithm.
- *Static restarting*, *i.e.*, periodic re-initialisation after all integer multiples of a given cutoff-time t', is one of the simplest methods for overcoming stagnation behaviour.
- A static restart strategy is effective, *i.e.*, leads to increased solution probability for some run-time t'', if the RTD of the given algorithm and problem instance is less steep than an exponential distribution crossing the RTD at some time $t < t''$.

Performance improvements based on static restarts (2)

- To determine the optimal cutoff-time t_{opt} for static restarts, consider the left-most exponential distribution that touches the given empirical RTD and choose t_{opt} to be the smallest t value at which the two respective distribution curves meet.

  (For a formal derivation of t_{opt}, see page 193 of SLS:FA.)

- **Note:** This method for determining optimal cutoff-times only works *a posteriori*, given an empirical RTD.

- Optimal cutoff-times for static restarting typically vary considerably between problem instances; for optimisation algorithms, they also depend on the desired solution quality.

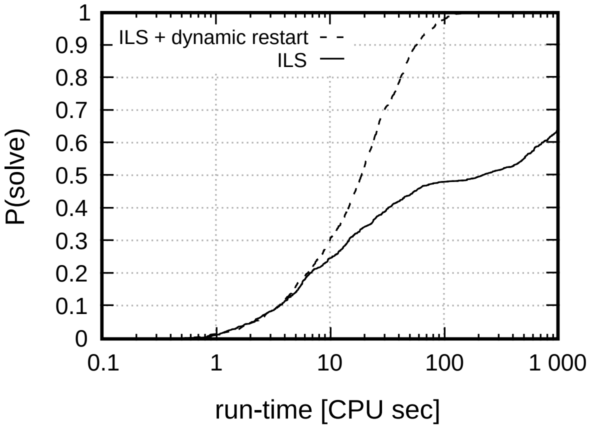
Overcoming stagnation using dynamic restarts

- *Dynamic restart strategies* are based on the idea of re-initialising the search process only when needed, *i.e.*, when stagnation occurs.

- *Simple dynamic restart strategy:* Re-initialise search when the time interval since the last improvement of the incumbent candidate solution exceeds a given threshold θ.
  (Incumbent candidate solutions are not carried over restarts.)

  θ is typically measured in search steps and may be chosen depending on properties of the given problem instance, in particular, instance size.

Example: Effect of simple dynamic restart strategy



Other diversification strategies

- Restart strategies often suffer from the fact that search initialisation can be relatively time-consuming (setup time, time required for reaching promising regions of given search space).

- This problem can be avoided by using other diversification mechanisms for overcoming search stagnation, such as
  - *random walk extensions* that render a given SLS algorithm provably PAC;
  - adaptive modification of parameters controlling the amount of search diversification, such as temperature in SA or tabu tenure in TS.

- Effective techniques for overcoming search stagnation are crucial components of high-performance SLS methods.

Multiple independent runs parallelisation

- Any LVA A can be easily parallelised by performing multiple runs on the same problem instance π in parallel on p processors.

- The effectiveness of this approach depends on the RTD of A on π:

  Optimal parallelisation speedup of p is achieved for an exponential RTD.

- The RTDs of many high-performance SLS algorithms are well approximated by exponential distributions; however, deviations for short run-times (due to the effects of search initialisation) limit the maximal number of processors for which optimal speedup can be achieved in practice.

Speedup achieved by multiple independent runs parallelisation of a high-performance SLS algorithm for SAT: