Rough Estimate of Complexity (REC)

mandatory performance measure

Note: what follows is based on the concepts defined by the document "General concepts and definitions for BPs", which is part of the information associated to all Benchmark Problem instances. Please refer to that document for clarification.

Introduction

The Rough Estimate of Complexity is a mandatory measure, i.e., its computation is necessary for a Benchmark Solution to be considered valid.

The aim of REC is to provide a basic and non-quantitative estimate of how the running time of an algorithm scales as the quantity of data available to be processed increases. REC is necessarily evaluated by running an implemented version of the algorithm under test (henceforth called **AUT**) on a specific machine, programming environment, etc. To limit the loss of significance associated to this issue, REC does not include any absolute performance indicators.

REC is a set of values, i.e., a discretized curve, depicting the overall running time of the implemented algorithm while proceeding through the processing of the input data from the chosen Benchmark Problem instance. The input data is ideally considered as subdivided into 1-second wide chunks, and the running time of the AUT is sampled each time that the processing of a new chunk begins. The result of this process is a curve with the running time on the y axis and the timestamp of the input data on the x axis, sampled with a 1-second period. To limit the effects of the specific machine used, only the *shape* of the curve should be considered as significant.

Differently from other performance measurements, REC is not based on the use of ground truth.

Preparatory operations

To compute REC, it is necessary to generate a software implementation of the algorithm under test. This program must accept as its input the data files provided by Rawseeds. To correctly evaluate REC, the program - or an auxiliary package external to it - should also be able to monitor the timestamp of the data samples as they are processed, and to record the running time of the implemented algorithm.

More details about the specific operations that need to be performed will be given in the following section. Please note that, due to the way in which REC is used as a performance measure, the impact of the specific techniques used to implement the algorithm (e.g., compiled programming language versus high-level scripting) is expected to be small.

Computation

All samples belonging to one of Rawseeds' data streams are given a *timestamp*. The timestamp identifies the time instant in correspondence of which the sample has been collected, on a real-time time scale generated by the robot used for data collection.

Let us define a non-negative index **k**, which will identify the instants when the running time of the (computer program implementing the) AUT are recorded. Then, let us define $\mathbf{t^{RT}_k}$ as the running time of the AUT in correspondence with the k-th of these instants. We will associate index k = 0 to the instant when the AUT begins processing the first data sample (i.e., the one associated to the first timestamp)[1]. Subsequent values of k are associated, as well, to instants when the AUT starts to process specific data samples. Precisely, the generic value k' > 0 of index k is associated to the time instant when the AUT starts processing the *first* processed[2] data sample which has a timestamp that exceeds the one of the first sample by at least k' seconds.

Given a generic data sample, let $\mathbf{t^{TS}}$ identify the difference between the timestamp of that sample and the timestamp of the first processed data sample of the dataset, in seconds. The value of t^{TS} for the data sample associated to the k-th instant (i.e., for the sample the processing of which started at instant k) is identified by $\mathbf{t^{TS}_k}$. Of course $t^{TS}_0 = 0$ [seconds] by definition.

Please note that, to compute REC, it is not required that the AUT uses a single input data stream (it could, and usually will, use multiple input streams: e.g., omnidirectional vision and odometry). It is also not required that such streams are processed synchronously; nor that all available samples are actually processed; nor that data samples are used in the order given by their timestamps. At any given moment, different - and possibly widely different - timestamps can be associated to the data samples currently being processed, and these can come from different streams. The rule for the selection of the time instant associated to value k' of index k can then be stated as follows: the value k' of index k is associated to the instant when the AUT starts processing the first processed sample for which $t^{TS} \geq k'$ [seconds], whichever data stream such sample belongs to.

The output of the procedure for computing REC is a set of couples $< t^{TS}_k, t^{RT}_k >$, where t^{TS}_k and t^{RT}_k are non-negative real numbers expressing times [in seconds]. Such output describes how the running time of the algorithm grows as the processing of the dataset proceeds, by sampling the

1 If any operation (e.g., memory allocation, file opening and/or reading) is performed by the AUT before the first sample is actually processed, these will occur before instant k = 0: therefore, generally it will be $t^{RT}_k > 0$.

2 Only "processed" samples are taken into consideration, because an algorithm could actually process only a subset of the available data samples.

running time of the AUT each time the processing of a new second-wide chunk of data begins.

Operatively, REC is computed by following these steps:

1. Run the program implementing the AUT on a chosen machine. Set $t^{RT} = 0$ [seconds] at the instant when the program starts.
2. As soon as the AUT actually starts processing the first data sample of the dataset, record the first sample $< t^{TS}_0, t^{RT}_0 >$ of REC[3]. Finally, set $k = 1$.
3. Every time that the AUT starts processing a new data sample, evaluate t^{TS} for that sample. If $t^{TS} \geq k$ [seconds], do as follows:
   - record a new sample $< t^{TS}_k, t^{RT}_k >$ of REC;
   - set $k = k + 1$.
4. Repeat the preceding point until the dataset has been completely processed by the AUT.

The result of the above procedure is a set of samples of REC. These must be recorded as a text file in the so-called *comma-separated values* (csv) format, with one sample per line. The data of each line must be in the form $< t^{TS}_k, t^{RT}_k >$.

Additional information to be provided

Whenever a Benchmark Solution includes the Rough Estimate of Complexity, it is also required to include the following information:

- A description of the computer used for the evaluation of REC, including its main hardware features and the type of operative system installed. In case such computer is a *virtual machine* running within a virtualization system, such information should be provided both for the *guest* and for the *host* machines.
- A high-level description of the implementation of the AUT used for the computation of REC, including what programming languages, compilers and/or interpreters have been used.

3 The first sample of REC is the one corresponding to $k = 0$. The first field of such sample is always equal to zero (in fact $t^{TS}_0 = 0$ by definition).