# Gamma Programming Language Benchmark Protocols

# Contents

# Introduction

The Gamma programming language is a *source language* in which to write *source code*. A *translator translates* source code into *object code* written in an *object language*.

This document describes the protocol for performing performance comparisons between translations of Gamma source code or between implementations of algorithms in Gamma and in other languages.

## Interpretation

This document has a prescriptive intent.

# 1 Common Aspects

Benchmarking consists of comparing the *performance* of *executions* of *equivalent algorithms* on the same *inputs*.

## 1.1 Performance & Execution

For the purposes of this document, the performance of an algorithm is the measure of the resources consumed during its execution. The resource consumption of an algorithm is given by its *complexity* $F : I \to U$, where:

- $I$ is the *domain* of the algorithm; and
- $U$ is either one of the following sets of values:
  - *time*, expressed in number of steps, in which case $F$ is the *time* complexity; or
  - *memory*, expressed in bytes, in which case $F$ is the *space* complexity.

The execution model of the algorithm is not specified here.

The complexity $F$ of a given algorithm is $G$ if and only if $F \in O(G)$, where $O(G)$ is the set of all functions asymptotically equal to $G$.

## 1.2 Equivalence

For the purposes of this document, two algorithms are considered equivalent only if:

1. they are extensionally equal; and
2. they have the same complexities.

# 2 Benchmark varieties

All benchmarks here must specify the following:

- the execution model, which includes the machine or target language and environment;
- one or more inputs on which to run the algorithms to benchmark.

## 2.1 Translator benchmarking

Translator benchmarking consists of benchmarking two or more translations of a Gamma module. Translator benchmarks must thus specify:

- a Gamma module; and
- two or more translators to compare.

There are two types of translator benchmarks: *functional benchmarks* and *algorithmic benchmarks*.

### 2.1.1 Functional benchmarking

Functional benchmarks benchmark implementations of Gamma language functions, on which all Gamma source code depends. The modules for these benchmarks will be specified by annex.

### 2.1.2 Algorithmic benchmarking

Algorithmic benchmarks benchmark applications of Gamma code in implementing algorithms of general interest, not necessarily related to the Gamma language.

## 2.2 Language benchmarking

Language benchmarking consists of benchmarking translations of two or more programs, where one of the programs is a Gamma module. Language benchmarks must thus specify:

- for each program not written in Gamma, the language; and
- for each program, the translator used.