LLMorpheus: Mutation Testing using Large Language Models

Frank Tip
f.tip@northeastern.edu
Northeastern University
Boston, MA, USA

Jonathan Bell
j.bell@northeastern.edu
Northeastern University
Boston, MA, USA

Max Schäfer
max@xbow.com
XBOW
Oxford, UK

ABSTRACT

In mutation testing, the quality of a test suite is evaluated by introducing faults into a program and determining whether the program's tests detect them. Most existing approaches for mutation testing involve the application of a fixed set of mutation operators, e.g., replacing a "+" with a "-", or removing a function's body. However, certain types of real-world bugs cannot easily be simulated by such approaches, limiting their effectiveness. This paper presents a technique for mutation testing where placeholders are introduced at designated locations in a program's source code and where a Large Language Model (LLM) is prompted to ask what they could be replaced with. The technique is implemented in *LLMorpheus*, a mutation testing tool for JavaScript, and evaluated on 13 subject packages, considering several variations on the prompting strategy, and using several LLMs. We find *LLMorpheus* to be capable of producing mutants that resemble existing bugs that cannot be produced by *StrykerJS*, a state-of-the-art mutation testing tool. Moreover, we report on the running time, cost, and number of mutants produced by *LLMorpheus*, demonstrating its practicality.

ACM Reference Format:
Frank Tip, Jonathan Bell, and Max Schäfer. 2025. LLMorpheus: Mutation Testing using Large Language Models. In *Proceedings of XXX*. ACM, New York, NY, USA, 80 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Mutation testing is an approach for evaluating the adequacy of a test suite and is increasingly adopted in industrial settings [1–3]. With mutation testing, an automated tool repeatedly injects a small modification to the system under test and executes the test suite on this mutated code. Mutation testing is premised on the *competent programmer hypothesis*, which posits that most buggy programs are quite close to being correct and that complex faults are *coupled* with simpler faults [4], i.e., a test that is strong enough to detect a simple fault should also be able to detect a more complex one. Hence, mutation analysis tools typically apply a relatively small set of mutation operators: replacing constants, replacing operators, modifying branch conditions, and deleting statements. Studies have shown that, given two test suites for the same system under test, the one that detects more mutants (even using only these limited mutation operators) is likely to also detect more real faults [5, 6].

XXX, XXX, XXX

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

However, not *all* real faults are coupled to mutants due to the limited set of mutation operators. For example, a fault resulting from calling the wrong method on an object is unlikely to be coupled to a mutant, as state-of-the-art mutation tools do not implement a "change method call" operator. While a far wider range of mutation operators has been explored in the literature [7, 8], state-of-the-practice tools like Pitest [9, 10], Major [11] and Stryker [12] typically do not implement them because of the implementation effort required and, especially, the increased cost of mutation analysis. Each additional mutation operator will result in more mutants that must be run and analyzed. Since each mutant must be evaluated in isolation, this may dramatically increase the time needed for developers that use the tool. Furthermore, some mutation operators might not be worthwhile to run, as noted in documentation from the developer of Pitest: "Although pitest provides a number of other operators, they are not enabled by default as they may provide a poorer experience" [13]. An alternative approach for generating mutants is to use a dataset of real faults to train a machine learning model to learn how to inject mutants [14–16]. However, the need for developers to train a model for their project impedes adoption of such techniques.

Our approach, *LLMorpheus*, can be viewed as a generalization of rule-based mutation techniques [9–12] in which the location of mutations is determined using a set of predefined rules and where an LLM is asked to suggest a diversity of mutations that introduce buggy behavior at those locations. To this end, *LLMorpheus* repeatedly prompts an LLM to inject faults at designated locations into a code fragment using prompts that include: (i) general background on mutation testing (ii) (parts of) a source file in which a single code fragment is replaced with the word "PLACEHOLDER", (iii) the original code fragment that was replaced by the placeholder, and (iv) a request to replace the placeholder with a buggy code fragment that has different behavior than the original code. After discarding syntactically invalid suggestions, we use *StrykerJS*, a state-of-the-art mutation testing tool for JavaScript that we modified to apply the mutations suggested by *LLMorpheus* instead of applying its standard mutators, classify mutants as killed, surviving, or timed out, and generate an interactive web site for inspecting the results.

The effectiveness of our approach hinges on the assumption that LLMs can understand the surrounding context of the code fragment represented by a PLACEHOLDER well enough to suggest syntactically valid and realistic buggy code fragments. To determine whether this assumption holds, we evaluate *LLMorpheus* on 13 subject applications written in JavaScript and TypeScript and measure how many mutants are generated and how they are classified (killed, survived, timed-out) using four "open" LLMs for which the training process is documented (Meta's *codellama-34b-instruct*, *codellama-13b-instruct*,

llama-3.3-70b-instruct and Mistral's *mixtral-8x7b-instruct*) and one proprietary LLM (OpenAI's *gpt-4o-mini*). We manually examine a subset of the surviving mutants to determine whether they are equivalent to the original source code or if they represent behavioral changes and contrast the results against mutants generated using *StrykerJS*'s standard mutators. The cost of *LLMorpheus* is assessed by measuring its running time and the number of tokens used for prompts and completions. We also report on experiments with alternative prompts that omit parts of the information encoded in default prompts and with different "temperature" settings of an LLM.

For surviving mutants generated using *codellama-34b-instruct*, we find that the majority (80%) reflect behavioral differences and 20% are equivalent to the original code. Using the *codellama-34b-instruct* and *codellama-13b-instruct* models, results are generally stable at temperature 0.0 when experiments are repeated, but the use of higher temperatures yields more variable results. For *mixtral-8x7b-instruct*, *llama-3.3-70b-instruct*, and *gpt-4o-mini* models, there is already significant variability at temperature 0. The default template generally produces the largest number of mutants and surviving mutants, and removing different fragments of this prompt degrades the results to varying degrees. The *llama-3.3-70b-instruct* and *codellama-34b-instruct* LLMs generally produce the largest number of mutants and surviving mutants, but *LLMorpheus* is still effective when *codellama-13b-instruct*, *mixtral-8x7b-instruct*, and *gpt-4o-mini* are used.

To investigate *LLMorpheus*'s ability to produce mutants that resemble real-world faults, we conducted a detailed case study involving 40 real-world bugs. In this case study, we used *LLMorpheus* to mutate the *fixed* version of a program near the location of the fix, executed the program's tests for each of these mutants and compared the test outcomes against those of the buggy version. For the 40 bugs under consideration in the case study, *LLMorpheus* was able to produce mutants that are syntactically identical to the buggy code fragments in 10 cases, and mutants that produce the same test failures as the original bug in an additional 26 cases. This provides evidence that *LLMorpheus* is capable of generating mutants whose behavior resembles that of real-world bugs, and that this capability is not entirely due to training-set leakage.

In summary, the contributions of this paper are:

(1) A technique for mutation testing in which placeholders are introduced at designated locations in a program's source code, and where an LLM is prompted to suggest what they could be replaced with.
(2) An implementation of this technique in *LLMorpheus*, a practical mutation testing tool for JavaScript.
(3) An empirical evaluation of *LLMorpheus* on 13 subject applications, demonstrating its practicality and comparing it to a standard approach to mutation testing based on mutation operators.
(4) A case study demonstrating *LLMorpheus*' ability to produce mutants with behavior resembling that of real-world bugs.

The remainder of this paper is organized as follows. Section 2 presents motivating examples that illustrate the potential of LLM-based mutation techniques to introduce faults resembling real bugs. In Section 3, an overview of our approach is presented. Section 4 presents an evaluation of *LLMorpheus* and Section 5 covers threats to validity. Related work is discussed in Section 6. Lastly, Section 7 presents conclusions and directions for future work.

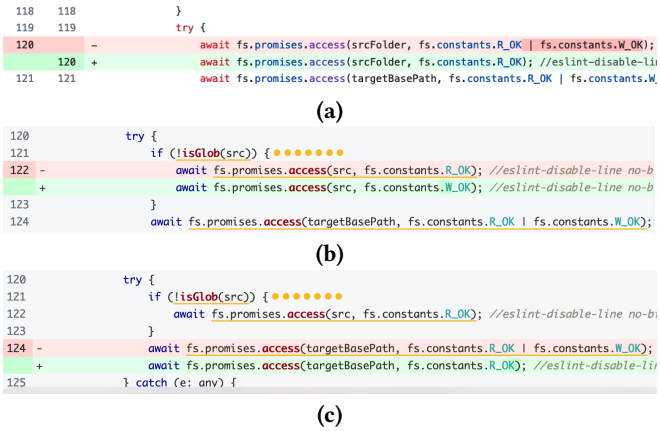

Figure 1: (a) Fix for a bug reported in issue #36 in `zip-a-folder`. (b) A mutation suggested by *LLMorpheus* at the same line that involves replacing read-access with write-access. (c) A mutation suggested by *LLMorpheus* elsewhere in the same file that mirrors the change made by the developer.

2 BACKGROUND AND MOTIVATION

In this section, we study a few bugs that do not correspond to mutation operators supported by state-of-the-art mutation testing tools but that are similar to mutations *LLMorpheus* could suggest.

Example 1. Zip-a-folder [17] is a library for compressing folders. On January 31, 2022, a user observed that the library required write access for source folders unnecessarily and opened issue #36, requesting that this access be removed. The developer applied the fix shown in Figure 1(a) on the same day, which involves replacing a binary bitwise-or expression with one of its operands.

LLMorpheus can suggest mutations that involve *changing or introducing* references to functions, variables, and properties. Figure 1(b) and (c) show two mutations that *LLMorpheus* suggests for this project and that could result in bugs similar to the one described above: part (b) shows a mutation at the same line where the bug was located that involves replacing read access with write access and part (c) shows a mutation at a nearby location that mirrors the change made by the developer.

The state-of-the-art *StrykerJS* tool is unable to suggest either of these mutations because (i) it does not support the mutation of bitwise operator expressions such as `fs.constants.R_OK | fs.constants.W_OK` unless they appear as part of a control-flow predicate, nor (ii) mutations that involve replacing a binary expression with one of its operands. While adding support for mutating bitwise operator expressions would be straightforward, concerns have been expressed that adding more mutation operators to traditional mutation testing tools might result in too many mutants and degraded

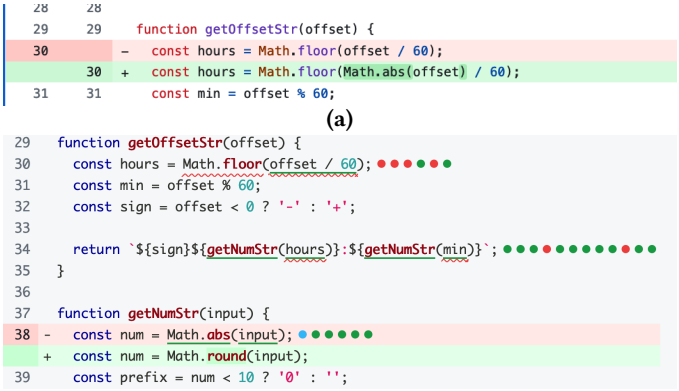

Figure 2: (a) Fix for a bug reported in issue #60 in countries-and-timezones. (b) A mutation suggested by *LLMorpheus* elsewhere in the same file.

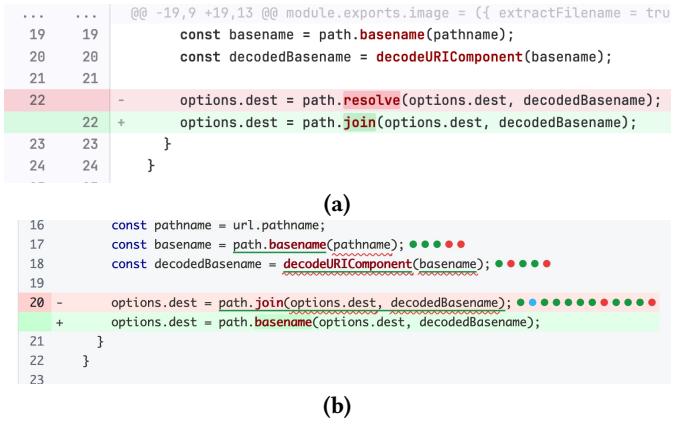

Figure 3: (a) Fix for a bug reported in issue #27 in image-downloader. (b) A mutation suggested by *LLMorpheus* at the same location that similarly involves calling a different function.

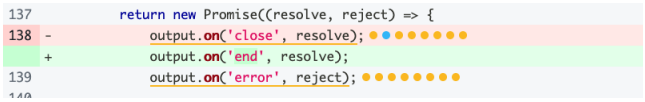

Figure 4: A mutation suggested by *LLMorpheus* that involves associating an event listener with the `end` event instead of with the `close` event.

performance [13, 18, 19]. More significantly, *StrykerJS* does not introduce or modify property access expressions and has very limited support for replacing an expression with a different expression[1].

Example 2. Countries-and-timezones [20] is a library for working with countries and timezones. In October 2023, a user reported a bug in function `getOffsetStr`, stating that it produces incorrect results when invoked with negative values. The developer proposed a simple fix that involves inserting a call to `Math.abs` to convert the argument value to a non-negative number, and a variation on this fix was quickly adopted by the developer, as shown in Figure 2(a).

This bug fix involves the introduction of a function call, so to *introduce* bugs like this one, a mutation testing tool would have to remove function calls or change the function being invoked. *StrykerJS* only supports a very limited set of 20 mutations to function calls[2], such as replacing calls to `String.startsWith` with call to `String.endsWith` and removing a call to `Array.slice`. While one could extend *StrykerJS* with a mutator that removes calls to `Math.abs`, many other function calls could be handled similarly, and adding mutators for all of them would yield an overwhelmingly large number of mutants. Many such candidate functions would not be good choices for mutation, either because the function in question is not a function that a developer inadvertently might have selected or because it would lead to syntactically invalid code.

LLMorpheus suggests mutations that involve introducing and replacing function calls. Figure 2(b) shows a mutation that *LLMorpheus* suggested elsewhere in the same source file that replaces a call to `Math.abs` with a call to `Math.round`, which could, in principle, introduce a bug like the one in Figure 2(a). Moreover, since LLMs are trained to generate code that resembles code written by developers, it is likely that the mutants produced by *LLMorpheus* involve using functions that a developer might have chosen.

Example 3. image-downloader is a module for downloading images. In February 2022, a user opened issue #27, entitled "If the directory name in dest: contains a dot `"."` then the download fails.", providing an example illustrating the problem. The developers soon responded with a fix, shown in Figure 3(a), that involves replacing a call to `path.resolve` with a call to `path.join`. While *LLMorpheus* does not produce a mutant that re-introduces this bug exactly, it does produce several at the same location[3] that similarly replace the invoked function, including the one shown in Figure 3(b). As mentioned, *StrykerJS* has very limited support for mutations that involve calling different functions and so it cannot suggest mutations like the one shown in Figure 3(b).

Example 4. Figure 4 shows another mutant produced by *LLMorpheus* for *zip-a-folder*. Here, the mutation involves changing the name of the event with which an event listener is associated. Such errors often cause "dead listeners", i.e., situations where an event handler is never executed because it is associated with the wrong event. Dead listeners are quite common in JavaScript, where the use of string values to identify events precludes static checking, and previous research has focused on static analysis [21] and statistical methods [22] for detecting such errors.

Discussion. The above examples illustrate just a few of the kinds of mutations that *LLMorpheus* may produce. Other mutations that it may suggest include: replacing a reference to a variable with a reference to a different variable, adding or removing arguments in

[1] In particular, *StrykerJS* only replaces control-flow predicates in `if`-statements and loops with boolean constants, string literals with the value `"Stryker was here"`, and object literals with an empty object literal.

[2] See https://stryker-mutator.io/docs/mutation-testing-elements/supported-mutators/.

[3] The line numbers have shifted slightly as the code has evolved since the bug report.

Figure 5: Overview of approach.

function calls, and modifying object literals by adding or removing property-value pairs.

In practice, the number of such mutations is effectively infinite, so an approach based on exhaustively applying a fixed set of mutation operators is unlikely to be practical. *LLMorpheus*' LLM-based approach leverages the collective wisdom of programmers who wrote the code on which the LLM was trained to develop mutations. As a result, suggested changes are likely to refer only to variables and functions that are in scope and are likely to be type-correct.

3 APPROACH

LLMorpheus is capable of producing interesting mutants without requiring any training on a subject project, which is a key distinction compared to existing work that builds models of real bugs to generate mutants [14–16]. This is accomplished by querying an LLM with a prompt that includes part of an application's source code in which a code fragment is replaced with the text "`<PLACEHOLDER>`". Additional information provided in the prompt includes: (i) general background on mutation testing, (ii) the code fragment that was originally present at the placeholder's location, (iii) a request to apply mutation testing to the code by replacing the placeholder with a buggy code fragment, and (iv) suggestions *how* the code could be mutated. The LLM is asked to provide three possible replacements for the placeholder[4], each accompanied by an explanation how the mutation would change program behavior.

Figure 5 presents a high-level overview of our approach, which involves three components that work in concert: the *prompt generator*, the *mutant generator*, and a version of the *StrykerJS* mutation testing tool that has been modified to apply the mutants created by *LLMorpheus*[5] We now discuss each of these components.

Prompt generator. This component takes as input a package and generates a set of prompts. This involves parsing the source files and identifying locations where mutations will be introduced. For ease of reference during prompting, the source code fragment corresponding to each location is replaced with the text "`<PLACEHOLDER>`". *LLMorpheus* considers the following locations as candidates for mutation: (i) conditions of `if`, `switch`, `while`, and `do-while` statements, (ii) initializers, updaters, and entire headers of loop statements, and (iii) receiver, arguments, and entire sequence of arguments for function calls. For each such location, a separate prompt is created. Figure 6 illustrates where placeholders are introduced into the source code.

`if (x === y){ ... }`	`if (<PLACEHOLDER>){ ... }`
`switch (x === y){ ... }`	`switch (<PLACEHOLDER>){ ... }`
`while (x){ ... }`	`while (<PLACEHOLDER>){ ... }`
`do { ... } while (x)`	`do { ... } while (<PLACEHOLDER>)`

`for (let i=0; i < x; i++){ ... }`	`for (<PLACEHOLDER>; i < x; i++){ ... }`
	`for (let i=0; <PLACEHOLDER>; i++){ ... }`
	`for (let i=0; i < x; <PLACEHOLDER>){ ... }`
	`for (<PLACEHOLDER>){ ... }`

`for (o in obj){ ... }`	`for (<PLACEHOLDER> in obj){ ... }`
	`for (o in <PLACEHOLDER>){ ... }`
	`for (<PLACEHOLDER>){ ... }`

`for (o of obj){ ... }`	`for (<PLACEHOLDER> of obj){ ... }`
	`for (o of <PLACEHOLDER>){ ... }`
	`for (<PLACEHOLDER>){ ... }`

`a.m(x,y)`	`<PLACEHOLDER>(x,y)`
	`a.m(<PLACEHOLDER>,y)`
	`a.m(x,<PLACEHOLDER>)`
	`a.m(<PLACEHOLDER>)`

Figure 6: Illustration of the insertion of placeholders to direct the LLM at source locations that need to be mutated.

The LLM is then given a prompt that is created by instantiating the template shown in Figure 7(a), by replacing `{{{ code }}}` with the original source code in which a placeholder has been inserted, and `{{{ orig }}}` with the code fragment that was replaced by the placeholder. Figure 7(b) shows the system prompt given to the LLM, which provides background on the role the LLM is expected to play in the conversation as a mutation testing expert. As can be seen in Figure 7(a), the prompt provides instructions for applying mutation testing to the specific source code at hand and details the format to which the completion should conform. Specifically, we require that the proposed mutants be provided inside "fenced code blocks" (i.e., code blocks surrounded by three backquote characters).

Mutant generator. This component takes the completions received from the LLM and extracts candidate mutants from the instantiated template by matching a regular expression against the completion to find the fenced code blocks. Candidate mutants identical to the original source code fragment or identical to previously generated mutants are discarded. The candidate mutants are then parsed to check if they are syntactically valid and discarded if this is not the case. The resulting mutants are written to a file `mutants.json` that is read by a customized version of *StrykerJS* that is described below. The mutant generator also saves all experimental data to files, including the generated prompts, completions received from the LLM, and the configuration options (e.g., the LLM's temperature setting).

Custom version of StrykerJS. We modified *StrykerJS* to give it an option `--usePrecomputed` that, if selected, directs it to read its set of mutations from a file `mutants.json` instead. *StrykerJS* then executes all mutants and determines (for each mutant) whether it causes

[4]The mutants produced by *LLMorpheus* always contain exactly one code change; if three valid suggestions are received from the LLM in response to one prompt, then three separate mutants will be generated.

[5]In particular, we use *StrykerJS*' to (i) determine the impact of each mutant on an application's tests and classify it as "killed", "survived", or "timed-out" and (ii) generate an interactive web page for inspecting results.

```
Your task is to apply mutation testing to the following
code:
```
{{{code}}}
```

by replacing the PLACEHOLDER with a buggy code fragment
that has different behavior than the original code fragment,
which was:
```
{{{orig}}}
```
Please consider changes such as using different operators,
changing constants, referring to different variables, object
properties, functions, or methods.

Provide three answers as fenced code blocks containing a
single line of code, using the following template:

Option 1: The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because
<brief explanation>.

Option 2: The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because
<brief explanation>.

Option 3: The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because
<brief explanation>.

Please conclude your response with "DONE."
```

(a)

```
You are an expert in mutation testing. Your job is to
make small changes to a project's code in order to find
weaknesses in its test suite. If none of the tests fail
after you make a change, that indicates that the tests
may not be as effective as the developers might have
hoped, and provide them with a starting point for
improving their test suite.
```

(b)

Figure 7: Prompt template (a) and system prompt (b) used by *LLMorpheus*.

tests to fail or time out. When this analysis is complete, *StrykerJS* generates a report as an interactive web page, allowing users to inspect the generated mutants. The previously shown Figures 1–4 show screenshots of our custom version of *StrykerJS*.

Pragmatics. While *LLMorpheus* implements a conceptually straightforward technique, considerable engineering effort was required to make it a practical tool. We use BabelJS [23] for parsing source code to identify locations where placeholders should be inserted and to check the syntactic validity of candidate mutants. Handlebars [24] is used for instantiating prompt templates. *StrykerJS* expects mutants to correspond to a single AST node, so for mutants that do not correspond exactly to a single AST node (e.g., loop headers and sequences of arguments passed in function calls), it is necessary to expand the mutation to the nearest enclosing AST node, for which we also rely on BabelJS.

LLMorpheus has command-line arguments for specifying the prompt template and system template to be used. Furthermore, it enables users to specify a number of LLM-specific parameters, such as the maximum length of completions that should be generated, the sampling temperature[6], and number of lines of source code that should be included in prompts (by default, this is limited to 200 lines surrounding the location of the placeholder). Since many LLM installations have limited capacity or explicit rate limits, *LLMorpheus* provides two command-line options to work with such LLMs: `--rateLimit <N>` ensures that at least N milliseconds will have elapsed between successive prompts and `--nrAttempts <N>` will try the same prompt up to N times if a 429 error occurs.

One possible concern with our approach is that *LLMorpheus* relies on a fixed set of locations where it introduces placeholders. The current placeholder scheme aims to balance creating a practical number of mutants and a larger set of mutants where at least one is more likely to result in a different control flow or data flow. Modifying *LLMorpheus* to use a different placeholder scheme would be straightforward. That said, the examples in Section 2 show that mutants produced by *LLMorpheus* (using its current placeholder scheme) involve changing references to variables, properties, and functions that cannot be produced using Stryker's mutation operators and that correspond to real-world bugs.

An open-source release of *LLMorpheus* can be found at https://github.com/neu-se/llmorpheus and the customized version of *StrykerJS* that we used for classifying mutants can be found at https://github.com/neu-se/stryker-js.

4 EVALUATION

4.1 Research Questions

This evaluation aims to answer the following research questions:

- **RQ1** How many mutants does *LLMorpheus* create?
- **RQ2** How many of the surviving mutants produced by *LLMorpheus* are equivalent mutants?
- **RQ3** What is the effect of using different temperature settings?
- **RQ4** What is the effect of variations in the prompting strategy used by *LLMorpheus*?
- **RQ5** How does the effectiveness of *LLMorpheus* depend on the LLM that is being used?
- **RQ6** What is the cost of running *LLMorpheus*?
- **RQ7** Is *LLMorpheus* capable of producing mutants that resemble existing bugs?

4.2 Experimental Setup

Selecting subject applications. Our goal is to evaluate *LLMorpheus* on real-world JavaScript packages that have test suites. Moreover, we want to compare the mutants generated by*LLMorpheus* to those generated using traditional mutation testing techniques, so we decided to focus on projects for which the state-of-the-art StrykerJS mutation testing tool [12] could be applied successfully. As a starting point for benchmark selection, we considered the 25 subject applications that were used to evaluate TestPilot [25], a recent LLM-based unit test generation tool. These applications are written in JavaScript or TypeScript, cover various domains, and have test suites that can be executed successfully.

[6]The sampling temperature is a parameter between 0 and 2 that controls the randomness of the completions generated by the LLM. Roughly speaking, the higher the temperature the more diverse the completions. At temperature zero, the LLM will always generate the most likely completion, which increases the chance that the same prompt will result in the same completion.

application	description	weekly downloads	#LOC	#tests	coverage		*StrykerJS*					
					stmt	branch	#mutants	#killed	#survived	#timeout	mut. score	time (sec)
Complex.js	complex numbers	671K	1,425	216	71.82%	67.54%	1,302	763	539	0	58.60	405.08
countries-and-timezones	accessing countries and timezones data	152K	165	58	100%	92.55%	140	134	6	0	95.71	142.37
crawler-url-parser	URL parser for crawling	495	209	185	96.39%	92.5%	226	143	83	0	63.27	433.53
delta	Format for representing rich text documents and changes	1.76M	806	180	98.99%	95.89%	834	686	88	60	89.45	2747.04
image-downloader	downloading image to disk from a given URL	17.75K	64	11	100%	93.75%	43	28	11	4	74.42	284.20
node-dirty	key value store with append-only disk log	5,604	207	37	83.01%	71.15%	160	78	56	26	65.00	215.93
node-geo-point	calculations involving geographical coordinates	5,618	406	10	85.36%	70.58%	158	98	60	0	62.03	357.70
node-jsonfile	reading/writing JSON files	57.7M	102	43	97.87%	94.11%	61	31	5	25	91.80	188.91
plural	plural forms of nouns	2,271	103	14	95.38%	72.72%	180	143	37	0	79.44	53.66
pull-stream	pipeable pull-stream	57.8K	602	364	90.96%	80.84%	474	318	116	40	75.53	694.33
q	promises	10.1M	2,111	243	89.5%	70.92%	1,058	68	927	63	12.38	7,075.02
spacl-core	path-based access control	3	377	38	100%	100%	259	239	20	0	92.28	1,053.16
zip-a-folder	zip/tar utility	60.1K	156	22	100%	96.87%	74	38	8	28	89.19	513.27

Table 1: Subject applications used to evaluate *LLMorpheus*.

Of these 25 subject applications, 10 could not be used because StrykerJS does not work on them, either because its dependences conflict with those of the subject application itself[7], or because it crashes. On one package, *simple-statistics*, StrykerJS requires approximately 10 hours of running time, which makes using it impractical. We excluded another package, *fs-extra*, a utility library for accessing the file system, because we observed that mutating this application poses a significant security risk, as the mutated code was corrupting our local file system. This left us with 13 subject applications for which Table 1 provides key characteristics. The first set of columns in the table show, from left to right, the name of the package, a short description of its functionality, the number of weekly downloads according to npmjs.com, the number of lines of source code, the number of tests, and the statement and branch coverage achieved by those tests, respectively. The second set of columns shows the results of running *StrykerJS* on the applications: the total set of mutants, the number of mutants that were killed, survived, and that timed out, the *mutation score*[8] reported by *StrykerJS*, and the time required to run *StrykerJS*, respectively.

LLM selection. RQ5 explores how the effectiveness of the proposed technique depends on the LLM being used. We use Meta's *codellama-34b-instruct* model for our main experiments. In addition, we evaluate the technique with Meta's *codellama-13b-instruct* and *llama-3.3-70b-instruct* models, with Mistral's *mixtral-8x7b-instruct* model, and with OpenAI's *gpt-4o-mini* model. The *codellama* models are specifically trained for tasks involving code. *llama-3.3-70b-instruct* is a newer and larger model from Meta that supersedes the smaller, specialized *codellama* models. *mixtral-8x7b-instruct* is a state-of-the-art general-purpose "mixture-of-experts" LLM developed by Mistral. *gpt-4o-mini* is a smaller, faster, and lower-cost variant of OpenAI's popular *gpt-4o* model The *codellama-34b-instruct*, *codellama-13b-instruct*, *llama-3.3-70b-instruct*, and *mixtral-8x7b-instruct* LLMs are "open" in the sense that their training process is documented. We relied on several commercial LLM service providers (https://octo.ai, https://openai.com, and https://openrouter.ai) for the experiments reported on in this paper.

LLM Temperature settings. LLMs have a temperature parameter that reflects the amount of randomness or creativity in their completions. For a task such as mutation testing, randomness and creativity may determine whether generated mutants are killed or survive. Therefore, we conduct experiments using several temperature settings.

Similarity to real-world bugs. Previous work evaluating mutation testing techniques has focused on "coupling" to determine whether mutants resemble real-world bugs [5, 6, 26]. This involves determining whether a test suite that detects particular mutants also detects particular real faults and requires a curated dataset of isolated faults. While many such datasets have been constructed from open-source projects written in Java, we found only one JavaScript dataset, the Bugs.js suite [27]. For each of these bugs, the original faulty version is provided, along with a cleaned patch extracted from the bug fix and instructions on executing the test cases. Unfortunately, we found that most of the Bugs.js subjects could not be used at all due to their reliance on outdated versions of various libraries and because of their incompatibility with modern Node.js versions that *StrykerJS* requires, causing them to be incompatible with *LLMorpheus*. These projects also have flaky tests[9], making it particularly challenging to perform mutation analysis [28].

We therefore constructed a new dataset[10] consisting of 40 real-world bugs, which includes 4 real-world bugs from the Bugs.js suite that we could reproduce reliably and 36 real-world bugs that we manually curated from various Node.js applications that are

[7]Running StrykerJS on an application requires installing it locally among the subject project libraries. Stryker itself depends on various other packages that also need to be installed, and these packages may conflict with packages that the subject application itself depends upon.

[8]The mutation score aims to provide a measure of the quality of a test suite by calculating the fraction of the total number of mutants that are detected (i.e., killed or timed out), see https://stryker-mutator.io/docs/General/faq/.

[9]See https://github.com/BugsJS/bug-dataset/issues/11.

[10]To facilitate further research by the community, our new bug dataset is available from https://github.com/neu-se/mutation-testing-data along with all experimental results associated with this paper.

application	#prompts	LLMorpheus									StrykerJS				
		#candidates	#invalid	#identical	#duplicate	#mutants	#killed	#survived	#timeout	mut. score	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490	1,451	194	13	45	1,199	725	473	1	60.55	1,302	763	539	0	58.60
countries-and-timezones	106	318	89	0	12	217	188	29	0	86.64	140	134	6	0	95.71
crawler-url-parser	176	521	205	14	17	285	157	128	0	55.09	226	143	83	0	63.27
delta	462	1,367	565	10	25	767	634	101	32	86.83	834	686	88	60	89.45
image-downloader	42	124	33	2	0	89	72	17	0	80.90	43	28	11	4	74.42
node-dirty	154	450	153	15	7	275	163	100	12	63.64	160	78	56	26	65.00
node-geo-point	140	408	93	0	13	302	223	79	0	73.84	158	98	60	0	62.03
node-jsonfile	68	199	42	3	0	154	49	48	57	68.83	61	31	5	25	91.80
plural	153	442	101	42	18	281	205	75	1	73.31	180	143	37	0	79.44
pull-stream	351	1,028	238	12	9	769	441	271	57	64.76	474	318	116	40	75.53
q	1,051	3,121	1,000	34	52	2,035	158	1,792	85	11.94	1,058	68	927	63	12.38
spacl-core	134	395	140	10	6	239	199	39	1	83.68	259	239	20	0	92.28
zip-a-folder	49	143	41	1	1	100	23	3	74	97.00	74	38	8	28	89.19
Total	3,376	9,967	2,894	156	205	6,712	3,237	3,155	320	—	4,969	2,767	1,956	246	—

Table 2: Results from LLMorpheus experiment (run #312). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

	temp. 0.0 (run #312)				temp. 0.25 (run #348)				temp. 0.50 (run #318)				temp. 1.0 (run #341)			
	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout
Complex.js	1,199	725	473	1	1,197	730	466	1	1,191	739	452	0	1,028	648	379	1
countries-and-timezones	217	188	29	0	219	181	38	0	224	194	30	0	186	156	30	0
crawler-url-parser	285	157	128	0	260	167	93	0	298	166	108	24	278	202	76	0
delta	767	634	101	32	781	642	111	28	769	642	93	34	698	583	83	32
image-downloader	89	72	17	0	86	71	15	0	89	68	21	0	75	53	22	0
node-dirty	275	163	100	12	279	175	93	11	277	158	107	12	246	150	84	12
node-geo-point	302	223	79	0	293	225	68	0	302	230	72	0	273	213	60	0
node-jsonfile	154	49	48	57	151	52	41	58	153	51	43	59	132	50	22	60
plural	281	205	75	1	273	208	63	2	289	219	69	1	299	229	69	1
pull-stream	769	441	271	57	779	452	270	57	796	465	278	53	743	461	235	47
q	2,035	158	1,792	85	2,050	153	1,813	84	2,073	163	1,823	87	1,899	147	1,671	81
spacl-core	239	199	39	1	223	187	36	0	250	210	39	1	218	180	38	0
zip-a-folder	100	23	3	74	97	24	4	69	87	48	33	6	96	54	38	4
Total	6,712	3,237	3,155	320	6,688	3,267	3,111	310	6,798	3,353	3,168	277	6,171	3,126	2,807	238

Table 3: Number of mutants generated using the *codellama-34b-instruct* LLM at temperatures 0.0, 0.25, 0.5, and 1.0 (showing one run of each)

available from GitHub (including some of the applications listed in Table 1).

We identified these bugs by searching https://www.npmjs.com/ for Node.js applications that cover a variety of domains, studying the issues in the GitHub repositories associated with these applications for messages that indicated the presence of a bug, and finding a subsequent "bug fix" commit in which this bug had been fixed (in most cases these also included the addition of new tests, or changes to existing tests). We then cloned the repository at that commit, reintroduced the bug, and observed if it caused any test failures. We discarded projects in which bugs/issues are not tracked explicitly and we discarded bugs that did not cause any test failures when reintroduced. In addition, we restricted our attention to bugs for which the commit containing the fix involves changing at most three lines of source code. Section 4.9 will report on experiments in which mutants are introduced at these locations, and our goal was to keep the number of such mutants manageable, given that careful manual analysis is involved in comparing the behavior of each mutant to that of the original buggy code.

4.3 RQ1: How many mutants does *LLMorpheus* create?

To answer this question, we ran *LLMorpheus* on the projects listed in Table 1 using the *codellama-34b-instruct* LLM at temperature 0.0 and the prompt templates shown in Figure 7. The results, shown in Table 2, show that *LLMorpheus* produces between 42 and 1,051 prompts for these projects. The following four columns in the table show the number of "candidate mutants", i.e., code fragments obtained by replacing placeholders with code fragments suggested by the LLM. The subcolumn labeled "candidates" shows the total number of candidate mutants, the subcolumn labeled "invalid" shows the number of candidate mutants that were found to be syntactically invalid, the subcolumn labeled "identical" shows the number of candidate mutants that were found to be identical to the original code, and the subcolumn labeled "duplicate" shows the number of candidate mutants that were found to be duplicated. From this data, it can be inferred that, on average, 29.0% (2,894/9,967) of candidate mutants are discarded because they are syntactically invalid, 1.6% (156/9,967) are discarded because they are identical to the original code, and 2.1% (205/9,967) are discarded because they are duplicates. This suggests that LLMs generally do not have too much trouble

with generating syntactically correct code, which is consistent with recent findings by others [25, 29]. The next column, labeled "mutants", shows the number of remaining mutants after discarding the useless candidate mutants. Here, the reader can see that between 89 and 2,035 mutants are generated for the subject packages. Of these mutants, between 23 and 725 are killed, between 3 and 1,792 survive, and between 0 and 85 time out. Aggregating the results over the 13 projects, it can be seen that 48.2% (3,237/6,712) of all mutants are killed, 47.0% (3,155/6,712) of all mutants survive, and 4.8% (320/6,712) of all mutants time out.

The table also shows the *mutation score*[11] as reported by StrykerJS, which aims to provide a measure of the quality of a test suite by calculating the fraction of the total number of detected mutants (i.e., killed or timed out).

To facilitate a quantitative comparison with *StrykerJS*, the last five columns in Table 2 repeat the results of running *StrykerJS* on the subject applications from Table 1. From this data, it can be seen that—in the aggregate for the 13 projects under consideration—*LLMorpheus* produces 3,155 surviving mutants whereas *StrykerJS* produces 1,956 surviving mutants. However, it should be noted that the difference in the number of mutants and surviving mutants varies significantly between subject applications. For example, for *Complex.js StrykerJS* produces more mutants (1,302 vs. 1,199) than *LLMorpheus*, of which more survive (539 vs. 473). On the other hand, for *q*, the situation is reversed with *StrykerJS* producing fewer mutants (1,058 vs. 2,035) and fewer surviving mutants (927 vs.1,792) than *LLMorpheus*. We conjecture that such differences are due to the subject programs' different characteristics, which make them amenable to different types of mutations. Here, *Complex.js* heavily uses arithmetic operators to implement mathematical operations on complex numbers, and such operators are prime candidates for *StrykerJS*'s standard mutation operators. Moreover, *q* makes heavy use of method calls, which are targeted by *LLMorpheus*'s placeholder-based strategy but much less so by *StrykerJS*'s mutation operators.

LLMs are nondeterministic, even at temperature 0.0, so a subsequent experiment may produce results that differ from those shown in Table 2. To determine to what extent this is the case, we repeated the same experiment four more times and measured how often the same mutants occur. We found that, at temperature 0.0, the results of *LLMorpheus* are generally stable across runs, with between 89.29% and 98.89% of all mutants being observed in all 5 experiments[12]. Figure 8 visualizes the stability of *LLMorpheus* when varying the prompt settings, and our supplemental materials include results across all settings of all models.

Using *codellama-34b-instruct* at temperature 0.0, *LLMorpheus* generates between 89 and 2,035 mutants, of which between 3 and 1,792 survive. These results are stable across experiments, with between 89.29% and 98.89% of all mutants being observed in all five experiments.

[11] See https://stryker-mutator.io/docs/General/faq/.

[12] All experimental data associated with this experiment and the other experiments are included with this submission as supplemental materials.

Project	*LLMorpheus* Equiv	*LLMorpheus* Not Equiv	Stryker.js Equiv	Stryker.js Not Equiv
Complex.js	6	44	1	49
countries-and-timezones	18	11	0	6
crawler-url-parser	11	39	5	45
delta	9	41	7	43
image-downloader	4	13	0	8
node-dirty	13	37	0	50
node-geo-point	3	47	5	45
node-jsonfile	12	25	0	3
plural	12	38	0	37
pull-stream	1	49	0	50
q	0	50	0	50
spacl-core	17	24	2	18
zip-a-folder	0	0	0	6
Total	106	418	20	410

Table 4: Number of equivalent surviving mutants generated by *LLMorpheus* and *StrykerJS*.

4.4 RQ2: How many of the surviving mutants are equivalent mutants?

One of the key challenges in mutation testing is the phenomenon of *equivalent mutants*: mutants that have equivalent behavior as the original code [4]. Mutants produced by *LLMorpheus* may involve arbitrary code changes, so the LLM could suggest code that is effectively a refactored version of the code that was originally present. To determine to what extent surviving mutants produced by *LLMorpheus* are equivalent, we conducted a study in which two authors manually examined 50 surviving mutants[13] in each project and classified each mutant as "equivalent" or "not equivalent" by Stryker or by *LLMorpheus* (sampled from run 314).

We labeled a mutant as *equivalent* if we could determine that the change could not cause *any* observable difference in behavior. For example, mutants that added extra parameters to methods (beyond those accepted by the receiver method) are trivially equivalent, as the runtime discards them. Other mutants are far from trivial to evaluate, and we manually wrote test code to attempt to discern the impact of, e.g., changing a condition from `if (!handler)` to `if (handler === undefined)`. Such a mutant is equivalent only if `handler` can never be any other "falsy" value (e.g., `null`, `false`, `NaN`, `0`, or the empty string `''`).

We labeled a mutant as *not equivalent* if it produced a change that could be observed as a behavioral change to a user of the library. By necessity, this definition is conservative: if there could exist any client of the library that would witness a different behavior, then the mutant is not equivalent. Note that this definition also includes changes to output messages printed on the console or to the messages included with errors. For example, a mutant in the statement `const hours=Math.floor(totalMinutes/60)` that changes the call from `Math.floor` to `Math.round` will result in the value of `hours` being incorrect. Of course, if `hours` is never used (or it doesn't matter that it is off-by-one), then the mutant could still be equivalent. Hence, we also found it necessary to trace through code to determine that some mutants were not equivalent.

Coding began with a pilot phase, where each coder labeled 10 surviving mutants in each project as equivalent or not. This process demonstrated strong agreement (Cohen's $\kappa = 0.873$) [30], and the

[13] For projects with fewer than 50 surviving mutants, we used as many as were available.

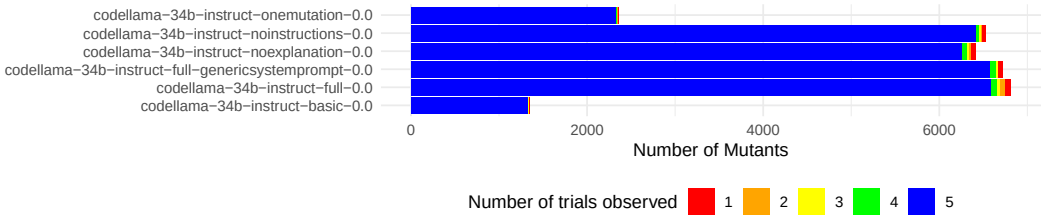

Figure 8: Stability of mutants generated by LLMorpheus with *codellama-34b-instruct* at temperature 0.0. For each replacement generated at each position, we count the number of trials (of 5 total) where that replacement was generated.

coders met briefly to clarify the few disagreements before proceeding with the remainder of the dataset. After independently coding the remainder of the dataset, the two result files were again compared for inter-rater reliability agreement, finding $\kappa = 0.846$, again indicating strong agreement [30]. To finalize the coding, the two authors met to discuss the cases on which they disagreed, reaching a consensus on the coding for all mutants during a single 30-minute session.

The results are shown in Table 4. Of the 524 *LLMorpheus* mutants examined, the majority (418, or 80%) are "not equivalent" and 106 (20%) are "equivalent". To contextualize these results, we also examine the mutants generated by *StrykerJS* and found that of 430 surviving mutants, 410 (95%) are "not equivalent" and 20 (5%) are "equivalent".

We further examined the 106 equivalent *LLMorpheus* mutants and observed several common patterns, including: (i) checking for `null`-ness or `undefined`-ness in different ways (e.g., replacing `x != null` with `!x` or vice versa), (ii) refactoring of calls to the `String.substring` method with one of its near-equivalent counterparts `String.substr` and `String.slice`, (iii) adding modifiers such as `/g` or `/m` to a regular expression in cases where this does not have any effect, (iv) calls to the `Array.slice` method in cases where this does not have any effect, and (v) calling functions with more arguments than are declared. For the 106 equivalent mutants under consideration, approximately 40% fall into one of these categories. We expect that most of these equivalent mutants can be filtered out using an AST-based static analysis. However, further investigation is needed because some mutants that cause behavioral differences are syntactically similar to these patterns. This means that any pattern-matching-based approach should consider the context in which the mutation occurs to determine whether a mutant is likely equivalent. Section 7 will discuss future work to reduce the number of equivalent mutants.

The majority (80%) of the surviving mutants produced by *LLMorpheus* are not equivalent to the original code fragments they replace. *LLMorpheus* produces significantly more "equivalent" mutants than *StrykerJS*. However, the number of "not-equivalent" mutants exceeds the number of equivalent mutants by more than a factor of three, and preliminary analysis reveals good potential for future work on automatically filtering out equivalent mutants using static analysis.

4.5 RQ3: What is the effect of different temperature settings?

To explore the impact of an LLM's temperature setting, we repeated the experiment with the *codellama-34b-instruct* LLM using temperatures 0.25, 0.50, and 1.0. The results of these experiments are summarized in Table 3. As can be seen from the table, the total number of mutants and the number of surviving mutants at temperatures 0.0, 0.25, and 0.50 are generally somewhat similar. However, at temperature 1.0, both the total number of mutants and the number of surviving mutants decline noticeably compared to the results for temperature 0.0. Inspection of the results revealed that this is partly because more of the generated mutants are syntactically invalid.

A secondary question is how temperature affects the variability of results. To answer this question, we repeated the experiment 5 times at each temperature and measured how many distinct mutants occur and how many mutants occur in all five runs. We found that, at higher temperatures, the number of distinct mutants increases rapidly and that the number of mutants common to all runs decreases accordingly. For example, for *Complex.js*, *LLMorpheus* generates 1,217 distinct mutants at temperature 0.0 of which 1,181 (97.04%) are common to all five runs. At temperature 0.25, the number number of distinct mutants increases to 2,354, of which 447 (18.99%) are common to all five runs. At temperature 0.5, there are 3,196 distinct mutants of which 205 (6.41%) are common to all runs. At temperature 1.0, there are 4,200 distinct mutants, of which 17 (0.4%) are common to all runs, meaning that, effectively, at temperature 1.0, each run produces completely different mutants. The results for the other subject applications are similar. The supplemental materials associated with this paper include an analysis showing the overall variability in mutants killed and survived across each of the five runs.

LLMorpheus generally produces similar numbers of mutants at temperatures ≤ 0.5, of which a similar number survives. At temperature 1.0, the number of generated and surviving mutants declines noticeably because more candidate mutants are syntactically invalid. The variability of results is inversely dependent on the temperature, with mostly the same mutants being produced at temperature 0.0 and mostly different mutants at temperature 1.0 in different runs.

4.6 RQ4: What is the effect of variations in the prompting strategy used by *LLMorpheus*?

Thus far, we have evaluated the effectiveness of the prompt template of Figure 7(a) (henceforth referred to as `full`) by measuring how many mutants are generated and classifying them as "killed", "survived", or "timed-out" (see Table 2). To determine what the effect is of each component of this prompt, we experimented with the following variations[14]:

[14] All prompt templates are included with the supplemental materials.

	full (run #312)				onemutation (run #365)				noexplanation (run #372)				noinstructions (run #378)				genericsystemprompt (run #384)				basic (run #390)			
	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout	#mutants	#killed	#survived	#timeout
Complex.js	1,199	725	473	1	406	245	161	0	1,125	676	448	1	1,137	696	440	1	1,199	740	458	1	185	120	65	0
countries-and-timezones	217	188	29	0	79	65	14	0	211	183	28	0	218	174	44	0	217	191	26	0	48	44	4	0
crawler-url-parser	285	157	128	0	86	50	36	0	239	140	99	0	246	134	112	0	246	143	103	0	67	49	18	0
delta	767	634	101	32	266	221	37	8	734	598	110	26	759	612	115	32	790	659	99	32	201	167	28	6
image-downloader	89	72	17	0	34	26	8	0	77	62	15	0	84	69	15	0	88	72	16	0	10	7	3	0
node-dirty	275	163	100	12	99	55	41	3	258	146	99	13	260	146	103	11	277	162	104	11	44	24	18	2
node-geo-point	302	223	79	0	104	74	30	0	297	216	81	0	306	230	76	0	305	229	76	0	62	54	8	0
node-jsonfile	154	49	48	57	57	18	18	21	152	54	45	53	148	45	51	52	150	49	49	52	22	11	3	8
plural	281	205	75	1	100	70	30	0	273	198	74	1	261	189	71	1	272	209	62	1	92	78	14	0
pull-stream	769	441	271	57	280	165	95	20	774	440	278	56	781	467	248	66	763	442	266	55	149	88	54	7
q	2,035	158	1,792	85	703	46	630	27	1,856	138	1,635	83	1,958	138	1,726	94	2,007	145	1,770	92	401	38	350	13
spacl-core	239	199	39	1	80	63	17	0	211	175	35	1	187	155	31	1	214	181	32	1	25	23	2	0
zip-a-folder	100	23	3	74	39	19	17	3	98	27	3	68	97	26	4	67	101	27	3	71	20	5	1	14
Total	6,712	3,237	3,155	320	2,333	1,117	1,134	82	6,305	3,053	2,950	302	6,442	3,081	3,036	325	6,629	3,249	3,064	316	1,326	708	568	50

Table 5: Number of mutants generated using the *codellama-34b-instruct* LLM at temperature 0.0 using templates full, onemutation, noexplanation, noinstructions, gen.system prompt, basic (showing one run of each).

onemutation. This variant requests just one replacement of the placeholder instead of three possible replacements.

noexplanation. This variant omits the phrase "`This would result in different behavior because <brief explanation>.`".

noinstructions. This variant omits the phrase "`Please consider changes such as using different operators, changing constants, referring to different variables, object properties, functions, or methods.`"

genericsystemprompt. In this variant, we replace the system prompt of Figure 7(b) with a generic message "`You are a programming assistant. You are expected to be concise and precise and avoid any unnecessary examples, tests, and verbosity.`"

basic. This minimal template only asks the LLM to provide a code fragment with which the placeholder can be replaced without any additional context.

Table 5 shows, for each template, the total number of mutants and the number that were killed, survived, and timed out, respectively. From these results, it can be seen that:

- *full* and *genericsystemprompt* produced the most mutants and performed similarly, demonstrating that the use of a specialized system prompt has minimal impact,
- *noexplanation* and *noinstructions* produce only slightly fewer mutants and surviving mutants than *full* and *genericsystemprompt*, so including instructions or requesting explanations for suggested mutations has limited impact,
- using *onemutation* dramatically reduces the number of mutants from 6,712 to 2,333, demonstrating that it is helpful to request multiple suggestions, and
- using *basic* reduces the number of mutants to 1,326, suggesting that additional context in prompts is helpful.

We separately analyzed the variability of these results (Table 5 presents the results from a single trial) and found the number of mutants killed and survived to be quite stable across trials (the supplemental materials provide further detail).

	full	onemutation	noexplanation	noinstructions	generic sys. prmpt	basic
Complex.js	4.27	3.37	5.09	4.27	4.17	11.98
countries-and-timezones	11.13	7.75	11.17	10.87	10.85	11.29
crawler-url-parser	9.50	6.41	9.46	9.49	9.30	20.04
delta	9.55	7.38	9.91	9.43	9.14	19.63
image-downloader	12.67	8.82	12.89	11.01	11.48	21.92
node-dirty	7.53	6.90	7.58	7.41	7.51	17.52
node-geo-point	8.86	6.10	8.79	7.75	8.66	15.66
node-jsonfile	9.73	6.98	9.76	7.77	8.91	11.64
plural	8.14	5.21	8.41	7.58	7.80	23.64
pull-stream	6.72	4.57	7.53	7.48	7.30	11.92
q	8.61	7.61	9.21	8.60	8.58	16.18
spacl-core	9.30	5.86	10.44	9.43	9.44	14.27
zip-a-folder	9.85	5.33	9.02	10.05	10.10	24.60

Table 6: Average string similarity of mutants to the original code fragments that they replace, for mutants generated using each of the prompt templates at temperature 0.0 using *codellama-34b-instruct*.

We also investigated how similar mutants produced using the different prompt templates are to the original code fragments they replace. As manually inspecting sufficient samples of mutants from each configuration would be infeasible, we instead rely on an automated measure. We calculate the Levenshtein string edit distance for each mutant between the mutated and original code. Table 6 reports the average string edit distance scores for each of the prompt templates by project.

Interpreting the results across different projects is challenging, as each project uses different code idioms that might lead to different mutations. However, we observe several interesting trends by comparing the mutant similarity across prompts (within the same project). We find the *basic* template to produce the mutants that are *least similar* to the original code. We examined samples of these mutants and found that many were creative changes that injected large code blocks in place of short, simple values. For example, in *crawler-url-parser*, the mutant with the most significant string edit distance (297) involves replacing a constant `TRUE` with an object literal. While the *onemutation* template tended to produce mutants

most similar to the original code, this is likely due to the more limited sample space. We infer that prompting for multiple mutants can result in the LLM suggesting more significant code changes than it would otherwise have.

The *full* template produces the most mutants and surviving mutants overall. Using a specialized system prompt has a marginal effect. Including instructions on performing mutations and requesting explanations for mutations only modestly affects the number of mutants generated and their detection rate. Requesting only one mutation dramatically reduces the number of generated and surviving mutants, and even greater reductions are observed if the LLM is only asked to fill in the placeholder without additional guidance.

4.7 RQ5: How does the effectiveness of *LLMorpheus* depend on the LLM being used?

The results discussed thus far were obtained with the *codellama-34b-instruct* LLM. To determine how the quality of results depends on the particular LLM being used; we also experimented with *codellama-13b-instruct*, *llama-3.3-70b-instruct*, *mixtral-8x7b-instruct*, and *gpt-4o-mini* at temperature 0.0.

Table 7 shows the number of mutant candidates produced using each model (along with a breakdown how many of those candidates are syntactically invalid, identical to the original code, or duplicates), and the number of mutants produced using each model, classified as killed, surviving, and timed-out. Figure 9 shows a visual comparison of the total number of mutant candidates and mutants produced using each of the five LLMs under consideration, aggregated over all 13 subject applications. From these results, it can be seen that:

- The *codellama-34b-instruct* model generates the largest number of mutant candidates (9,967), though the number of mutant candidates produced by *codellama-13b-instruct*, *mixtral-8x7b-instruct*, *llama-3.3-70b-instruct*, and *gpt-4o-mini* are quite similar. *codellama-13b-instruct* produces noticeably fewer mutant candidates (8,088).
- All models produce a significant number of mutant candidates that is syntactically invalid, ranging from 3,703 in the case of *gpt-4o-mini* to 2,540 in the case of *mixtral-8x7b-instruct*.
- *codellama-13b-instruct* is the only model that produces a significant number of mutant candidates that are identical to the original code fragments that they replace (922).
- None of the models produces a significant number of duplicate mutant candidates.
- The number of mutants that remain after discarding the invalid, identical, and duplicate mutant candidates ranges from 5,402 in the case of *mixtral-8x7b-instruct* to 6,823 in the case of *llama-3.3-70b-instruct*, with *codellama-34b-instruct* producing almost as many valid mutants (6,712).
- *llama-3.3-70b-instruct* produces the most surviving mutants (3,423), followed by *codellama-34b-instruct* (3,155).

We also explored the variability of results produced using *codellama-13b-instruct*, *mixtral-8x7b-instruct*, *llama-3.3-70b-instruct* and *gpt-4o-mini* by conducting each experiment 5 times, and determined how many distinct mutants are produced and how many mutants occur in all five runs. We found that, at temperature 0.0, the results obtained with *codellama-13b-instruct* are very stable across runs, with 96.15%–100% of all mutants occurring in each of the five runs. However, with *mixtral-8x7b-instruct*, *llama-3.3-70b-instruct*, and *gpt-4o-mini*, we encountered more variability. With *mixtral-8x7b-instruct*, between 34.22%–50% of mutants occur in all five runs, with *llama-3.3-70b-instruct*, between 28.26%–58.94% occur in all five runs, and with *gpt-4o-mini*, between 28.26%–58.94%. Figure 10 visualizes the variability of *LLMorpheus* across all configurations. The figure shows, in the aggregate, for all 13 subject applications, in how many runs each mutant was observed. We also analyzed the variance of the number of mutants killed and survived, finding that the mutation score was relatively stable despite the diversity of mutants across trials. The supplemental materials include tables showing the average and standard deviation of the number of mutants killed and survived.

We also examined the string similarity of mutants produced by the five LLMs to the original code and found that the *mixtral-8x7b-instruct* model tends to generate mutants with the greatest string edit distance in the most projects. We examined the top 2 mutants with the greatest string edit distance generated by this model for each project, finding several cases of unusual completions. In *q*, mixtral's most dissimilar mutants (distance 219) replaced a string literal that referred to the function `" allResolved "` with a declaration of the same function. In *delta*, mixtral's most dissimilar mutants (distance 155) apply a `reduce` operation to an object before invoking `Object.keys` on it. We saw similar trends for mixtral across all projects, with mutants that tended to include long code declarations. Examining the mutants with the greatest string edit distance for the other four LLMs, we did not find significant trends that held across all targets. Further details can be found in the supplemental materials.

All five LLMs under consideration can successfully generate large numbers of (surviving) mutants. *llama-3.3-70b-instruct* and *codellama-34b-instruct* tend to produce the largest number of surviving mutants, and *codellama-34b-instruct* produces stable results across experiments when temperature 0.0 is used. *llama-3.3-70b-instruct*, *mixtral-8x7b-instruct*, and *gpt-4o-mini* produce highly variable results, even at temperature 0.0.

4.8 RQ6: What is the cost of running *LLMorpheus*?

The primary costs of running *LLMorpheus* are the time required to run experiments and the expenses associated with LLM usage. Regarding the latter, for the experiments reported on in this paper, we have relied on several commercial LLM service providers (octo.ai, openrouter.ai, and openai.com). Such costs tend to vary depending on the provider and the LLM being used and are typically calculated as a function of the number of "tokens" used in the prompt and the completion[15]. The cost of commercial LLM providers also tends to vary over time, and when a newer version of an LLM is released, it often costs the same as the older version that it replaced. In our experiments, the total number of tokens used for running a full experiment with *LLMorpheus* varied by less than 20% for the five LLMs that we used[16], suggesting that token usage is a reasonable proxy for the financial costs incurred. For these reasons, we use the

[15]Depending on the provider, the number of requests may also incur additional costs, though that was not the case for our experiments.

[16]Calculated from the total number of tokens reported in the Supplemental Materials associated with this paper for experiments using the "full' prompt template at temperature 0.0.

	codellama-13b-instruct (run #354)								*mixtral-8x7b-instruct (run #360)*							
	#candidates	**#invalid**	**#identical**	**#duplicate**	**#mutants**	**#killed**	**#survived**	**#timeout**	**#candidates**	**#invalid**	**#identical**	**#duplicate**	**#mutants**	**#killed**	**#survived**	**#timeout**
Complex.js	1,410	339	116	28	955	553	401	1	1,272	310	0	15	962	589	373	0
countries-and-timezones	305	83	15	1	207	177	30	0	272	65	2	5	205	166	39	0
crawler-url-parser	494	186	51	12	247	129	118	0	411	165	0	3	234	130	104	0
delta	1,334	530	92	16	712	583	107	22	1,132	452	0	24	680	516	128	36
image-downloader	122	40	5	2	77	48	29	0	107	38	0	1	69	46	23	0
node-dirty	439	161	33	11	245	142	92	11	300	109	0	10	191	111	72	8
node-geo-point	390	64	21	16	304	237	67	0	341	88	0	11	247	166	81	0
node-jsonfile	191	43	10	7	138	43	45	50	155	23	0	4	132	54	32	46
plural	407	100	99	17	208	154	53	1	299	73	0	8	226	166	60	0
pull-stream	1,002	279	54	13	669	386	237	46	934	255	1	6	678	386	248	44
q	2,993	901	379	55	1,713	122	1,518	73	2,418	772	3	50	1,643	112	1,460	71
spacl-core	377	142	40	7	185	160	25	0	330	152	0	3	157	134	22	1
zip-a-folder	137	43	7	1	87	27	55	5	117	38	0	0	78	24	44	10
Total	9,601	2,911	922	186	5,582	2,761	2,777	209	8,088	2,540	6	140	5,402	2,600	2,686	216

	llama-3.3-70b-instruct (run #23)								*gpt-4o-mini (run #58)*							
	#candidates	**#invalid**	**#identical**	**#duplicate**	**#mutants**	**#killed**	**#survived**	**#timeout**	**#candidates**	**#invalid**	**#identical**	**#duplicate**	**#mutants**	**#killed**	**#survived**	**#timeout**
Complex.js	1,417	279	0	53	1,138	690	447	1	1,432	446	0	38	986	596	390	0
countries-and-timezones	304	64	0	14	240	207	33	0	308	99	0	9	209	171	38	0
crawler-url-parser	506	175	0	22	319	208	111	0	516	227	0	11	275	181	94	0
delta	1,333	539	0	54	794	626	130	38	1,345	636	2	40	707	564	108	35
image-downloader	122	43	0	5	79	54	25	0	123	58	0	3	65	45	20	0
node-dirty	453	121	1	10	331	168	142	21	453	188	0	9	265	154	103	8
node-geo-point	399	39	0	22	358	255	103	0	399	86	0	20	311	225	86	0
node-jsonfile	198	25	0	6	173	64	37	72	198	44	0	6	154	64	26	64
plural	427	96	0	29	331	244	87	0	428	110	5	26	313	257	55	1
pull-stream	1,044	262	0	10	782	465	265	52	1,037	302	0	16	735	420	247	68
q	3,074	855	0	80	2,219	127	2,006	86	3,084	1,287	2	69	1,795	137	1,597	61
spacl-core	383	134	0	18	236	203	32	1	392	158	0	10	215	195	20	0
zip-a-folder	145	24	0	2	121	87	5	29	143	62	0	4	81	14	7	60
Total	9,805	2,656	1	325	6,823	3,398	3,423	300	9,858	3,703	9	261	5,885	3,023	2,791	297

Table 7: Mutants generated with the *codellama-13b-instruct*, *mixtral-8x7b-instruct*, *llama-3.3-70b-instruct*, and *gpt-4o-mini* LLMs, using the following parameters: temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

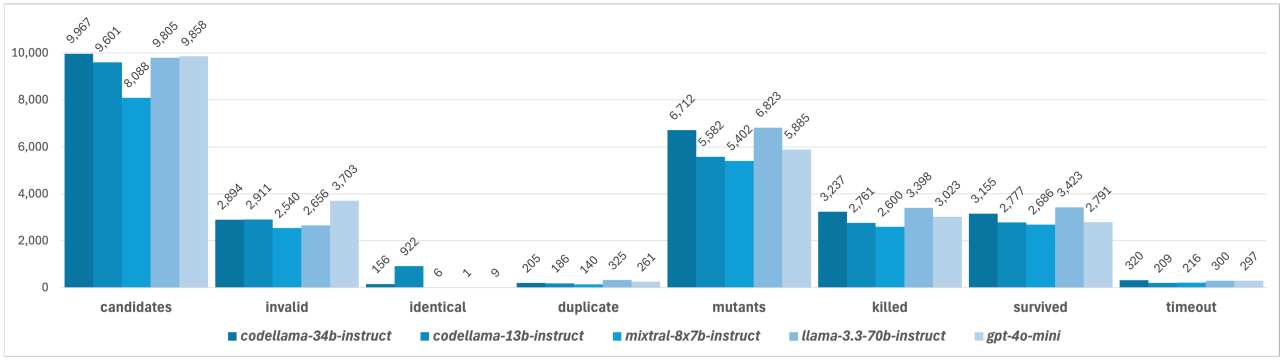

Figure 9: Comparison of the number of mutant candidates and mutants generated with the *codellama-13b-instruct*, *mixtral-8x7b-instruct*, *llama-3.3-70b-instruct*, and *gpt-4o-mini* LLMs at temperature 0.0. This chart was created from the data shown in Tables 2 and 7.

number of input and output tokens used in our experiments as the primary cost metric for evaluating *LLMorpheus*'s LLM usage. For completeness, we also discuss the expense in US dollars at the time of running the experiments below, but the reader should be aware that these costs are likely to vary over time.

The **time** column in Table 8 shows the time needed to run *LLMorpheus* and the modified version of *StrykerJS* on each subject application. As can be seen in the table, *LLMorpheus* requires between 430.53 seconds (about 7 minutes) and 5,241.46 seconds

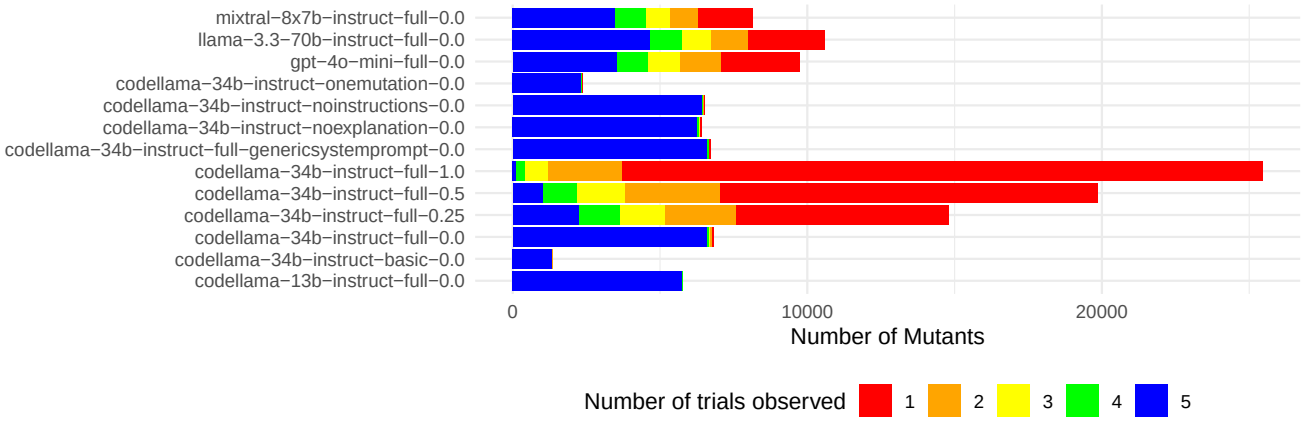

Figure 10: Variability of mutants generated by *LLMorpheus*. For each replacement generated at each position, we count the number of trials (of 5 total) where that replacement was generated.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,050.00	637.85	967,508	102,517	1,070,025
countries-and-timezones	1,070.89	313.86	105,828	23,441	129,269
crawler-url-parser	1,642.70	929.43	386,223	39,175	425,398
delta	2,961.66	3,839.60	890,252	98,974	989,226
image-downloader	430.53	379.25	24,655	9,134	33,789
node-dirty	1,526.20	241.81	246,248	33,070	279,318
node-geo-point	1,411.11	987.17	316,333	30,013	346,346
node-jsonfile	690.61	474.78	57,516	14,797	72,313
plural	1,521.32	155.24	265,602	34,174	299,776
pull-stream	2,492.50	1,608.97	208,130	76,513	284,643
q	5,241.46	14,034.67	2,127,655	220,215	2,347,870
spacl-core	1,351.08	798.96	162,705	29,236	191,941
zip-a-folder	500.57	1,156.11	82,457	10,725	93,182
Total	23,890.64	25,557.70	5,841,112	721,984	6,563,096

Table 8: Results from LLMorpheus experiment (run #312). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

(about 87 minutes) and *StrykerJS* between 155.24 seconds (about 2.5 minutes) and 14,034.67 seconds (about 234 minutes).

The last three columns of Table 8 show the number of tokens used in prompts and completions for each subject application and in the aggregate. From these results, it can be seen that running *LLMorpheus* required between 24,655 and 2,127,655 prompt tokens and between 9,134 and 220,215 completion tokens. Hence, in the aggregate, 5,841,112 prompt tokens and 721,984 completion tokens were required. At the time of conducting the experiments, the cost of the *codellama-34b-instruct* LLM using octo.ai's LLM service was $0.50 per million input tokens and $1.00 per million output tokens, so for running *LLMorpheus* on all 13 applications, a total cost of approximately $3.62 was incurred. Moreover, at the time of conducting our experiments, the *llama-3.3-70b-instruct* model that we used can be accessed from $0.12 per million input tokens and $0.30 per million output tokens at openrouter.ai, and the *gpt-4o-mini* model that we used can be accessed from $0.15 per million input tokens and $0.60 per million output tokens from openai.com. Hence, a full experiment can be run for less than $1 with *llama-3.3-70b-instruct*, and for approximately $1.30 with *gpt-4o-mini*.

It should be pointed out that the cost of the LLMs we used is significantly lower than that of larger state-of-the-art proprietary LLMs such as OpenAI's *gpt-4o*, for which https://openai.com/pricing quotes a cost of $2.50 per million input tokens and $10 per million output tokens at the time of writing. While such models might be even more capable of suggesting useful mutants, it is encouraging to see that lower-cost LLMs can achieve good results.

LLMorpheus requires between 7 and 87 minutes to generate mutants for 13 subject applications. At the time of conducting our experiments, a full experiment with *LLMorpheus* on all 13 applications costs up to $3.62 depending on the LLM being used, suggesting that cost is not a prohibitive limiting factor.

4.9 RQ7: Is *LLMorpheus* capable of producing mutants that resemble existing bugs?

To determine whether *LLMorpheus* is capable of producing mutants that resemble existing bugs, we conducted a case study involving 40 real-world bugs, shown in Table 9. The construction of this dataset was previously discussed in Section 4.2(d). In this study, we applied *LLMorpheus* to the *fixed* version of a program by introducing placeholders near the location of the fix, generating mutants, executing the program's tests for each of these mutants, and checking if the observed test failures were identical to those caused by the original bug. We considered two failures to be the same if the same error message and stack trace were produced. This task involves significant manual effort and time as it involves executing all tests for each mutant and manually comparing the behavior of the test failures caused by mutants against test failures caused by the original bug. Given the potential non-determinism inherent to *LLMorpheus*, we repeat this mutant generation, test execution and manual inspection process a total of five times per-bug. Our artifact contains complete details on each of the bugs examined, showing the buggy code and mutant side-by-side, along with our comments and analysis.

package	description	issue/bug #	SHA	same code change(s)	same test failure(s)	different test failure(s)
css-loader	CSS files	663	d1d8221		✓	
css-loader	CSS files	789	e3bb83a		✓	
css-loader	CSS files	1036	ded2a79		✓	
css-loader	CSS files	1261	729a314			✓
compression	file compression	170	b7d5d77			✓
countries-and-timezones	accessing country/timezone data	60	97a106f	✓		
express.js	web application framework	2	–	✓		
fast-glob	file system	223	05a4c08		✓	
fast-glob	file system	391	eb55d1d		✓	
fast-xml-parser	XML parsing	234	ea5d544		✓	
fast-xml-parser	XML parsing	595	b0ea635		✓	
fs-extra	file system	190	e05c685		✓	
fs-extra	file system	291	2e7f755		✓	
fs-extra	file system	679	7c251d6		✓	
hessian	serialization	4	–		✓	
hexo	blogging framework	12	–	✓		
htmlparser2	HTML parsing	746	214ab08		✓	
htmlparser2	HTML parsing	913	04c411c	✓		
jsdiff	file comparison	94	d76ac52		✓	
jsdiff	file comparison	118	4a899c0		✓	
jsdiff	file comparison	217	6464b29			✓
jsdiff	file comparison	493	f38e47d		✓	
karma	testing framework	4	–	✓		
memfs	in-memory file system	59	b90c016		✓	
memfs	in-memory file system	391	301f2d1		✓	
memfs	in-memory file system	853	8b021b3		✓	
memfs	in-memory file system	870	7c5999c	✓		
memfs	in-memory file system	1024	711c4bd	✓		
memfs	in-memory file system	1093	ede0f4f	✓		
node-jsonfile	reading/writing JSON files	24	c2c8a2c	✓		
node-jsonfile	reading/writing JSON files	25	afaba5d		✓	
normalize-url	URL utilities	38	6078d91		✓	
normalize-url	URL utilities	82	191ad4b		✓	
simple-statistics	statistics	334	522a716		✓	
simple-statistics	statistics	633	6547df7		✓	
yargs	command-line arguments	1364	35d777c		✓	
yargs	command-line arguments	1376	3d26d11	✓		
yargs	command-line arguments	1422	9a42b63			✓
yargs	command-line arguments	1493	63b3dd3		✓	
yargs	command-line arguments	2171	f91d9b3		✓	
Total: 40				10	26	4

Table 9: Results of case study investigating whether *LLMorpheus* can generate mutants that are similar to real bugs. Each row of the table corresponds to one bug, for which the first four columns of the table state the name and description of the package, issue/bug number, and commit id (SHA) containing the bug fix. The last three columns classify each bug into one of the following categories: "same code change(s)" means that *LLMorpheus* generates at least one mutant that contains the same code change(s) as the bug *and* causes the same test failure(s), "same test failure(s)" means that *LLMorpheus* generates at least one mutant that is not syntactically the same as the bug but causes the same test failure(s) (possibly also causing other tests to fail), and "different test failures" means that *LLMorpheus* does not generate at least one mutant that causes the same test failure(s).

For each bug, Table 9 shows the name and description of the package, the issue number associated with the bug in the repository's issue tracker, and the commit ID (SHA) containing the fix. Further details about these bugs, the mutants produced by *LLMorpheus*, and the test results obtained with each mutant are included in our artifact.

The last three columns of the table show, for each bug, to what extent the mutants produced by *LLMorpheus* mimic the original bug. We classify the bugs into the following three categories:

same code change(s). This means that *LLMorpheus* generates at least one mutant that contains the same code change(s) as the bug *and* causes the same test failures,

same test failure(s). This means that *LLMorpheus* generates at least one mutant that is not syntactically the same as the bug but results in the same test failure(s) as the bug (possibly also causing other tests to fail), and

different test failures. This means that *LLMorpheus* does not generate at least one mutant that causes the same test failure as the bug.

As can be seen from the table, for 10 of the 40 bugs, *LLMorpheus* produced mutants consisting of code fragments that are syntactically identical to the original bug and that caused the same test failures. Moreover, in an additional 26 cases, *LLMorpheus* produced mutants that caused the same test failures as the ones caused by the original bug. In only 4 cases did *LLMorpheus* not produce any mutants that cause similar test failures as the original bugs. In addition, it should be noted that, of the 40 bugs under consideration, 35 involved a patch that involved complex changes that do not correspond to the application of a traditional mutation operators (e.g., changing conditions by adding/removing subconditions, referencing different variables, calling different/additional functions, adding arguments in function calls, changing regular expression literals, etc.) This means that a traditional mutation testing tool such as *StrykerJS* would be unable to reproduce these bugs exactly

```
360  360
361       -  if (!trust(this.connection.remoteAddress)) {
     361  +  if (!trust(this.connection.remoteAddress, 0)) {
362  362       return proto;
363  363     }
```

Figure 11: Patch corresponding to bug #2 in *express.js* [27].

```
301  301   proto.writeObject = function (obj) {
302       -  if (is.nullOrUndefined(obj)) {
     302  +  if (is.nullOrUndefined(obj) ||
     303  +    // : { a: { '$class': 'xxx', '$': null } }
     304  +    (is.string(obj.$class) && is.nullOrUndefined(obj.$))) {
303  305       debug('writeObject with a null');
```

Figure 12: Patch corresponding to bug #4 in *hessian.js* [27].

(though it might still be able to create mutants that produce the same failures).

Our artifact contains details regarding all of the bugs that we studied. Below, we report on our findings for four of the bugs in more detail.

Express.js Bug#2. Figure 11 shows the patch for bug #2 in Express, a popular web framework for Node.js. This bug occurs at line 361 in the file `lib/request.js` and involves the invocation of a function `trust` with a single argument `this.connection.remoteAddress`. Here, the fix involved the addition of a second argument, `0`. Reintroducing this bug in the fixed version causes two tests to fail. When applied to the fixed version, *LLMorpheus* creates the following three mutants:

- replacing `!trust(this.connection.remoteAddress, 0)` with `trust(this.connection.remoteAddress, 1)`
- replacing `!trust(this.connection.remoteAddress, 0)` with `!trust(this.connection.remoteAddress)`
- replacing `!trust(this.connection.remoteAddress, 0)` with `trust(this.connection.localAddress, 0)`

The second mutant is identical to the original bug. The other two mutants cause multiple test failures that differ from those caused by the original bug.

Given the possibility of non-determinism impacting this experiment, we conducted five repeated trials[17].We found that in some cases, *LLMorpheus* produces mutants such as `!trust(this.connection.localAddress, 1)` that differ from the original bug but cause the same test failures. Moreover, in one experiment, *LLMorpheus* produced a mutant `!this.app.get('trust proxy')` that reproduces one of the two test failures caused by the original bug.

Hessian.js Bug#4. Figure 12 shows the patch for bug #4 in Hessian, a serialization framework. This bug occurs at line 302 in file `lib/v1/encoder.js` and involves the condition of an `if`-statement. Here, the fix for the bug involves changing the condition from `is.nullOrUndefined(obj)` to `is.nullOrUndefined(obj) || (is.string(obj.$class) && is.nullOrUndefined(obj.$))`. Reintroducing this bug in the fixed version results in a test failure.

When applied to the fixed version, *LLMorpheus* creates the following 3 mutants:

- replacing `is.nullOrUndefined(obj) || (is.string(obj.$class) && is.nullOrUndefined(obj.$))` with `obj === null`,

[17] Data for five experiments with each of the 40 bugs is included with supplemental materials.

```
17  17     dirent.mode = mode;
18      -  dirent.path = link.getPath();
    18  +  dirent.path = link.getParentPath();
19  19
```

Figure 13: Patch corresponding to issue #1024 in *memfs*.

```
104  104
105      -  if (commandKeys.length > 0) {
     105  +  if ((currentContext.commands.length > 0) || (commandKeys.length > 0)) {
106  106       argv._.slice(currentContext.commands.length).forEach((key) => {
```

Figure 14: Patch corresponding to issue #1364 in *yargs*.

- replacing `is.nullOrUndefined(obj) || (is.string(obj.$class) && is.nullOrUndefined(obj.$))` with `is.nullOrUndefined(obj.$)`
- replacing `is.nullOrUndefined(obj) || (is.string(obj.$class) && is.nullOrUndefined(obj.$))` with `!obj`

In this case, none of the generated mutants are identical to the original bug. However, the first and the third mutants *cause exactly the same test failures as the original bug*. The second mutant causes multiple test failures that differ from those produced by the original bug. We repeated the same experiment four more times, and while *LLMorpheus* never reproduced the original bug, it produced mutants with the same behavior as the original bug on multiple occasions.

memfs issue#1024. Figure 13 shows the bug fix for issue #1024 in *memfs*, an in-memory file system for Node.js. Reintroducing the bug in the patched version results in failures in three tests for the `readdirsync` function. Here, the fix involves replacing a method call `link.getPath()` on line 18 in file `src/DirEnt.ts` with a call `link.getParentPath()`.

When applied to this line, *LLMorpheus* creates the following three mutants:

- replacing `link.getParentPath` with `link.getPath`,
- replacing `link.getParentPath` with `link.getName`, and
- replacing `link.getParentPath` with `''`.

The first mutant is identical to the original bug, the second mutant results in code that violates TypeScript's typing rules, and the third mutant causes three test failures that differ from those caused by the original bug. We repeated the experiment four more times and observed that the original bug was reproduced during one of the other runs.

yargs issue#1364. Figure 14 shows the bug fix for issue #1364 in *yargs*, a popular framework for parsing command-line arguments. Reintroducing the bug in the patched version results in a single test failure. Here, the fix involves changing the condition of an `if` statement on line 105 in file `lib/validation.js` from `commandKeys.length > 0)` to `(currentContext.commands.length > 0) || (commandKeys.length > 0))`.

Here, *LLMorpheus* creates three mutants:

- replacing `(currentContext.commands.length > 0) || (commandKeys.length > 0)` with `(currentContext.commands.length > 0) || (commandKeys.length > 0)`,
- replacing `(currentContext.commands.length > 0) || (commandKeys.length > 0)` with `(currentContext.commands.length === 0) && (commandKeys.length === 0)`, and
- replacing `(currentContext.commands.length > 0) || (commandKeys.length > 0)` with `argv._.length > 0`.

The first of these mutants causes two test failures, of which one is identical to the test failure caused by the original bug, the second mutant causes five test failures, of which one is identical to the test failure caused by the original bug, and the third mutant survives, i.e., it does not cause any test failures. We repeated the experiment four more times, and each time, at least one mutant was produced that triggered the same test failure as the original bug, along with a few additional test failures.

For the 40 bugs under consideration in the case study, *LLMorpheus* was able to produce mutants that are syntactically identical to the buggy code fragments in 10 cases, and mutants that produce the same test failures as the original bug in an additional 26 cases. This provides evidence that *LLMorpheus* is capable of generating mutants whose behavior resembles that of real-world bugs and that this capability is not entirely due to training-set leakage.

4.10 Experimental Data

All experimental data associated with the experiments reported on in this paper can be found at https://github.com/neu-se/mutation-testing-data.

5 THREATS TO VALIDITY

The projects used to evaluate *LLMorpheus* may not be representative of the entire ecosystem of JavaScript packages. To mitigate this risk, we select popular packages used in prior JavaScript testing tool evaluations and report results per project, discussing the full range of behaviors we witness. As in many evaluations of LLM-based tools, the validity of our conclusions may be threatened by including our evaluation subjects in the training data for the models. If the model were trained on bugs in some of the programs we asked it to create bugs in, one would expect its performance on those programs to vary significantly from those on which it was not pre-trained. We mitigate this risk by conducting experiments with five LLMs, four of which are "open" in the sense that the training process is documented, thus enabling reproducibility and detailed analysis of experimental results.

Truly determining if a mutant is equivalent requires significant effort and despite the best efforts of two authors to evaluate them rigorously, there may be errors in categorizing mutants. We interpret the high degree of inter-rater reliability ($\kappa = 0.846$) as a reasonable assurance of the reliability of this process.

One of the key evaluation criteria used in previous work on mutation testing is "coupling", i.e., determining whether a test suite that detects particular mutants also detects particular real faults [5, 6, 26]. We investigated the feasibility of conducting such a study using the Bugs.js suite [27], but we found that most of these subjects could not be used at all due to their reliance on outdated versions of various libraries and because of their incompatibility with modern Node.js versions that *StrykerJS* requires, causing them to be incompatible with *LLMorpheus*. These projects also have flaky tests, making it particularly challenging to perform mutation analysis [28]. We, therefore, opted for conducting a case study involving 40 real-world bugs, including 4 real-world bugs from the Bugs.js suite that we were able to reproduce reliably and an additional 36 bugs taken from a variety of real-world Node.js applications for which we manually identified an issue in the project's issue tracker that reported the problem and a subsequent bug-fix commit. For these 40 bugs, *LLMorpheus* produced mutants that replicated the code changes from the original bug in 10 cases, and it produced mutants that replicated the test failures caused by the original bug in an additional 26 cases, suggesting that *LLMorpheus* can produce mutants that behave similarly to existing bugs in most cases. The results of this case study may be skewed because the code for previous buggy versions of the applications may have been included in the training set of the LLM that we used. However, the fact that *LLMorpheus* frequently produced mutants in the case study that *differed from the original bug but caused the same test failures* suggests that *LLMorpheus*'s ability to produce mutants that resemble real-world bugs is not entirely due to training-set leakage. The results of the case study may also have been skewed by the selection of the subject applications in the case study and by our focus, for pragmatic reasons, on bugs for which the fix involves a small number of lines of code.

As a deliberate design choice, *LLMorpheus* employs a fixed strategy for introducing placeholders, as illustrated in Figure 6, which precludes the creation of mutants at certain locations, thus potentially limiting its effectiveness. However, traditional mutation testing tools such as *StrykerJS* are similarly limited by applying mutations only in selected locations and are *additionally limited* by restricting mutations to a fixed repertoire of mutation operators. Moreover, our current placeholder strategy has been shown to be effective at producing large numbers of (surviving) mutants and at producing mutants that resemble real-world bugs. Exploring mechanisms that allow users to specify placeholder schemes, e.g., as a predicate on AST nodes, is a topic for future work.

Evaluating tools that rely on LLMs face significant reproducibility challenges. We mitigate these risks by (i) evaluating *LLMorpheus* using four open LLMs that are version-controlled and permanently archived (in addition to one popular proprietary LLM), (ii) repeating each experiment 5 times and (iii) making all experimental data available as supplemental materials, and (iv) making *LLMorpheus*, our evaluation scripts and results publicly available. Including all results for all experiments in the main body of this paper would significantly decrease the readability of the work. Where we observed significant variability in results, we include data regarding that distribution in the paper directly. In all cases, the supplemental materials associated with this paper include results for all trials of all experiments and summary tables that describe the observed variability for each configuration.

Lastly, a possible concern is that *LLMorpheus* only supports JavaScript and TypeScript, and its applicability beyond these languages may be unclear. Implementing the same approach for a different language would involve various steps (parsing ASTs, executing tests, etc.) that are language-specific and would involve significant engineering effort, but should otherwise be straightforward.

6 RELATED WORK

Mutation testing, first introduced in the 1970's [4], has a long history [31]. The era of "big code" and software repository mining

has enabled the large-scale evaluation of the core hypothesis behind mutation testing: mutants are coupled to real faults. Just *et al*. mined real faults from Java applications and found a statistically significant correlation between mutation detection and real fault detection [5]. This finding has since been replicated on newer, larger datasets of faults from even more Java programs [6]. Gay and Salahirad extended this methodology to examine the extent to which individual mutation operators are most coupled to real faults [26]. While this has demonstrated that test suites that detect more mutants are also likely to detect more bugs, it also underscores the need for new mutation approaches that can generate faults coupled to more real bugs.

ML for Mutation Testing: Several recent projects have considered using LLMs and other AI-based techniques for mutation testing. μBert [32, 33] resembles *LLMorpheus* in that both techniques select some designated code fragments, and query a model what they could be replaced with. μBert masks one token at a time, so its mutations involve changes to a single variable or operator. By contrast, *LLMorpheus*' placeholders correspond to (sequences of) AST nodes, so it may suggest mutations involving more significant changes to complex expressions. A crucial difference between the techniques is that *LLMorpheus* utilizes prompts that provide an LLM with additional guidance, whereas μBert provides no way of guiding the mutations at all and is, therefore, completely at the mercy of what the model thinks masked tokens should be replaced with. In our experiments with different prompts (Section 4.6), the *basic* prompt is analogous to μBert in that it merely asks the LLM what placeholders should be replaced with. Our results show this to be much less effective at producing interesting mutants, thus demonstrating the usefulness of including additional information in prompts. Our work also differs from [32] by considering several LLMs and different temperatures and targeting a different language.

In recent work, Garg et al. [34] explore the coupling between mutants generated using μBert and 45 reproducible vulnerabilities from the Vul4J dataset. They distinguish between *strongly coupled mutants* that fail the same tests for the same reasons as the vulnerabilities and *test coupled mutants* that fail the same tests but for different reasons. While they find the majority (32 of 45) of μBert-generated mutants to be strongly coupled, they also find that strongly coupled mutants are scarce, representing just 1.17% of killable mutants. It would be interesting to explore whether the use of more elaborate prompting strategies, such as those employed by *LLMorpheus* could be used to increase the ratio of strongly coupled mutants.

Tian et al. [35] consider the use of LLMs for determining whether mutants are equivalent and compare their effectiveness to that of traditional techniques for mutant equivalence detection. Their study considers the detection of equivalent mutants in 19 Java programs from the MutantBench suite [36], from which mutants were derived using standard mutation operators from μJava [37]. Tian et al. experimented with 10 LLMs. They consider 10 state-of-the-art LLMs and several strategies for fine-tuning and prompting, and consider three widely used traditional techniques (compiler-based, ML-based, and Tree-Based Neural NetWork) as the baseline for comparison. Their results indicate that LLMs are significantly better than traditional techniques at equivalent mutant detection, with the fine-tuned code embedding strategy being the most effective. It would be interesting to explore to what extent these results carry over to detecting mutants that were produced using LLMs using tools such as *LLMorpheus*.

Similar to our interests, Wang et al. [38] perform an exploratory study on using large language models to generate mutants. Unlike our prompting strategy that generates up to three mutants per-AST node, Wang et al. explore a strategy that generates mutants at the granularity of entire methods. We demonstrate the nuances of prompt engineering in this context by exploring performance under different prompts (RQ4). These complementary works demonstrate the potential of using LLMs for mutation testing.

Several projects [39, 40] have considered the use of LLMs as mutation operators in the context of Genetic and Search-Based techniques to improve the efficiency of the search. Brownlee et al. [40] consider the generation of alternate implementations for methods and experimented with prompts exhibiting different levels of detail, similar to our experiments reported on in Section 4.6, finding that more detailed prompting generally improves the number of successful patches.

Several other works rely on LLMs to validate the results of mutation testing tools. Li and Shin [41] use 4 syntactic mutation operators and then observe the change to the natural language description that an LLM generates of the mutated code. MuTAP [42] uses an off-the-shelf syntactic mutation tool to generate mutants for a Python program and then prompts an LLM to generate a test that can detect those mutants.

Equivalent Mutants: Kushigian et al. [43] study the types and prevalence of equivalent mutants in Java programs, considering why they are equivalent and how challenging it is to detect that they are. Their study considers 19 Java open-source programs from which mutants are derived using Major [11], a rule-based mutation-testing framework for Java that supports similar mutation operators as *StrykerJS*. Their findings indicate that around 3% of mutants are equivalent, and these equivalent mutants are further classified according to criteria that reflect *why* a mutant is equivalent, and *how* this could be determined. Based on these findings, Kushigian et al. propose Equivalent Mutant Suppression (EMS), a collection of simple static checks for detecting equivalent mutants.

Improving Mutation operators: Other approaches for mutation testing aim to generate mutants that represent a wider variety of faults. "Higher-order mutation" combines multiple mutations concurrently, creating more complex faults, but still limited by the set of operators implemented [8, 44]. More recently, Brown *et al*. improve mutation by mining patches for new idioms to use as mutation operators [45]. Beller *et al*. design a similar tool and evaluate it at Facebook, with the goal of increasing adoption of mutation testing [46] Taking this idea further, Tufano *et al*. create *DeepMutation*, an approach that learns models for performing mutation from real bugs [14]. This idea was refined by Tian *et al*.'s *LEAM*, which improves the search process by leveraging program grammars [15]. Patra and Pradel's *SemSeed* learns to generate mutants from fixes of real-world identifier and literal semantic bugs [16]. Unlike these approaches, *LLMorpheus* uses a *pre-trained LLM*, requiring no training to apply it to a new project.

Mutation Testing Applications and Tools: Belén Sánchez et al. [47] report on a results of a qualitative study among open-source developers on the use of mutation testing. Their findings indicate

that developers find mutation testing useful for improving test suite quality, detecting bugs, and improving code maintainability and performance considerations are the biggest impediments to adoption. Much of the research advancing the state of mutation testing tooling has targeted Java, such as MuJava [37], Javalanche [48], Jumble [49], Judy [50] and Major [11]. Gopinath *et al.* empirically compared two of these research-oriented tools tools (Judy [50], Major [11]) with an industry-oriented tool (Pit [9]), finding that despite the stated similarities between the tools, each produced a somewhat different set of mutants [51]. Pit is actively maintained, and the open-source tool is also available packaged with professional plugins under the name 'ArcMutate' [10]. Also aimed at practitioners, the *Stryker* mutation tool is a framework that supports code written in JavaScript, TypeScript, C#, and Scala [12]. We build *LLMorpheus* atop Stryker. Deb *et al.* examine a new, language-agnostic approach to generating mutants using regular expressions [52]. Future work may examine the feasibility of implementing *LLMorpheus* using this approach.

Mutation and Test Generation: There is a long line of research on test-generation techniques that specifically target mutated code. DeMillo and Offutt [53] presented a technique that relies on solving systems of algebraic constraints to derive test cases that target mutated code. Fraser and Zeller [54] present μTEST, an approach that automatically generates unit tests for object-oriented classes based on mutation analysis. Their test generation technique uses mutations as the coverage criterion that it aims to maximize and creates tests containing oracles that test the mutated value. Chekam et al. [55] present a test generation technique based on symbolic execution that systematically searches for situations where program behaviors of the original program diverges from that of mutated versions. Lee et al. [56] present a grey-box fuzzing technique that involves executing both the original and the mutated code in the same fuzzing driver to direct the generation of test inputs toward those that kill mutants. Adapting LLM-based test generation techniques [25, 57, 58] to target mutated code would be an interesting topic for future work.

LLMs and Testing: Beyond mutation testing, LLMs have also been used for test generation. Bareiß et al. [58] present an approach for test generation that follows a few-shot learning paradigm, outperforming traditional feedback-directed test generation [59]. Tufano et al. [57] present an approach for test generation using a BART transformer model [60] that is fine-tuned on a training set of functions and corresponding tests. Lemieux et al. [29] present an approach where tests generated by Codex are used to assist search-based testing techniques [61] in situations where such techniques get "stuck" because the generated test cases diverge too far from the expected uses of the code under test. TestPilot [25] produces unit tests for JavaScript programs by prompting an LLM with the start of a test for an API function, with information about that function (signature, body, and usage examples mined from project documentation) embedded in code comments. In response, the LLM will produce a candidate test, which it executes to determine whether it passes or fails. In case of failure, TestPilot attempts to fix the failing test by re-prompting the LLM with the error message. In principle, *LLMorpheus* can be used to evaluate such test generation techniques by providing a means to assess the quality of the generated tests.

7 CONCLUSIONS AND FUTURE WORK

We have presented *LLMorpheus*, an LLM-based technique for mutation testing. In this approach, code fragments at designated locations in the program's source code are replaced with the word "PLACEHOLDER", and an LLM is given a prompt that includes: general background on mutation testing, the original code fragment, and instructions directing the LLM to replace the placeholder with a buggy piece of code. The mutants produced by *LLMorpheus* are passed to a modified version of the popular *StrykerJS* mutation testing tool, which runs the tests, classifies mutants, and creates an interactive web page for inspecting the results.

An empirical evaluation on 13 subject applications demonstrates that *LLMorpheus* can produce mutants that resemble real bugs that cannot be produced using standard mutation operators. We found that the majority (80%) of surviving mutants produced by *LLMorpheus* are behavioral changes and that 20% of them are equivalent mutants. Experiments with variations on the prompt template reveal that the "full" template that includes all information performs best and that omitting parts of the information from this template matters to varying degrees. From experiments with five LLMs, we found that *llama-3.3-70b-instruct* and *codellama-34b-instruct* generally produced the largest number of mutants and surviving mutants. Moreover, in a case study involving 40 real-world bugs, we found that *LLMorpheus* produced mutants that are syntactically identical to the buggy code fragments in 10 cases and mutants that produce the same test failures as the original bug in an additional 26 cases. These results provide strong evidence that *LLMorpheus* is capable of generating mutants whose behavior resembles that of real-world bugs and that this capability is not entirely due to training-set leakage.

The number of mutants produced by *LLMorpheus* can become quite large, and executing them can take considerable time. In future work, we plan to explore techniques for pruning and prioritizing mutants, focusing particularly on reducing the number of equivalent mutants. From a manual investigation of 105 equivalent mutants, we observed several common patterns, such as replacing an condition `!x` with `x === null` or `x === undefined` or replacing call to `String.substring` with calls to `String.substr` and `String.slice`, two methods with similar semantics. We expect that most of these equivalent mutants can be filtered out using simple AST-based analysis. However, further investigation is needed because some mutants that cause behavioral differences are syntactically similar to these patterns. This means that any pattern-matching-based approach should consider the context of the mutation to determine whether a mutant is likely to be equivalent. To deal with more challenging cases, future work could also explore the use of symbolic execution or efficient formal reasoning techniques for automatically identifying mutants that are likely to be equivalent.

LLMorpheus currently employs a fixed strategy for introducing placeholders, as illustrated in Figure 6. While this strategy has been shown to be effective at producing large numbers of (surviving) mutants and at producing mutants that resemble real-world bugs, it precludes the creation of mutants at locations that do not match this strategy. As future work, we plan to explore mechanisms that allow users to specify a placeholder scheme, e.g., as a predicate on AST nodes.

In our research, we used LLMs in their default configuration without any fine-tuning. The strong results obtained with the relatively small *codellama-34b-instruct* LLM that is trained for code-related tasks suggests that fine-tuning an LLM for the specific task of mutation testing might be worthwhile, particularly to optimize the number of mutants that are not equivalent.

ACKNOWLEDGEMENTS

This work was supported in part by the US National Science Foundation under grants CCF-2100037, CCF-2307742 and CNS-2100015.

REFERENCES

[1] G. Petrović, M. Ivanković, G. Fraser, and R. Just, "Does mutation testing improve testing practices?," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pp. 910–921, 2021.

[2] G. Petrović, M. Ivanković, G. Fraser, and R. Just, "Practical mutation testing at scale: A view from google," *IEEE Transactions on Software Engineering*, vol. 48, no. 10, pp. 3900–3912, 2022.

[3] G. Petrović, M. Ivanković, G. Fraser, and R. Just, "Please fix this mutant: How do developers resolve mutants surfaced during code review?," in *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pp. 150–161, 2023.

[4] R. DeMillo, R. Lipton, and F. Sayward, "Hints on test data selection: Help for the practicing programmer," *Computer*, vol. 11, no. 4, pp. 34–41, 1978.

[5] R. Just, D. Jalali, L. Inozemtseva, M. D. Ernst, R. Holmes, and G. Fraser, "Are mutants a valid substitute for real faults in software testing?," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014)*, pp. 654–665, 2014.

[6] T. Laurent, S. Gaffney, and A. Ventresque, "Re-visiting the coupling between mutants and real faults with defects4j 2.0," in *2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 189–198, 2022.

[7] Y. Jia and M. Harman, "An analysis and survey of the development of mutation testing," *IEEE Transactions on Software Engineering*, vol. 37, no. 5, pp. 649–678, 2011.

[8] A. S. Ghiduk, M. R. Girgis, and M. H. Shehata, "Higher order mutation testing: A systematic literature review," *Computer Science Review*, vol. 25, pp. 29–48, 2017.

[9] H. Coles, T. Laurent, C. Henard, M. Papadakis, and A. Ventresque, "Pit: a practical mutation testing tool for java (demo)," in *Proceedings of the 25th International Symposium on Software Testing and Analysis*, ISSTA 2016, (New York, NY, USA), p. 449–452, Association for Computing Machinery, 2016.

[10] H. Coles, "Arcmutate: Advanced mutation testing for java and kotlin." https://www.arcmutate.com, 2024.

[11] R. Just, "The Major mutation framework: efficient and scalable mutation analysis for java," in *Proceedings of the 2014 International Symposium on Software Testing and Analysis*, ISSTA 2014, (New York, NY, USA), p. 433–436, Association for Computing Machinery, 2014.

[12] Stryker Team, "Stryker mutator." https://stryker-mutator.io, 2025. Accessed 1/8/2025.

[13] H. Coles, "Arcmutate: Extended mutation operators." https://docs.arcmutate.com/docs/extended-operators.html, 2024.

[14] M. Tufano, C. Watson, G. Bavota, M. Di Penta, M. White, and D. Poshyvanyk, "Learning how to mutate source code from bug-fixes," in *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 301–312, 2019.

[15] Z. Tian, J. Chen, Q. Zhu, J. Yang, and L. Zhang, "Learning to construct better mutation faults," in *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, ASE '22, (New York, NY, USA), Association for Computing Machinery, 2023.

[16] J. Patra and M. Pradel, "Semantic bug seeding: a learning-based approach for creating realistic bugs," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, p. 906–918, 2021.

[17] "zip-a-folder." https://github.com/maugenst/zip-a-folder, 2025. Accessed 1/8/2025.

[18] R. Just and F. Schweiggert, "Higher accuracy and lower run time: Efficient mutation analysis using non-redundant mutation operators," *Software Testing, Verification and Reliability (JSTVR)*, vol. 25, pp. 490–507, Jan. 2015.

[19] R. Just, B. Kurtz, and P. Ammann, "Inferring mutant utility from program context," in *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*, (Santa Barbara, CA, USA), pp. 284–294, July 10–12 2017.

[20] "countries-and-timezones." https://github.com/manuelmhtr/countries-and-timezones, 2025. Accessed 1/8/2025.

[21] M. Madsen, F. Tip, and O. Lhoták, "Static analysis of event-driven Node.js JavaScript applications," in *Proceedings of the 2015 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2015, Pittsburgh, PA, USA, October 25-30, 2015* (J. Aldrich and P. Eugster, eds.), pp. 505–519, ACM, 2015.

[22] E. Arteca, M. Schäfer, and F. Tip, "Learning how to listen: Automatically finding bug patterns in event-driven JavaScript APIs," *IEEE Trans. Software Eng.*, vol. 49, no. 1, pp. 166–184, 2023.

[23] "Babel." https://babeljs.io/, 2025. Accessed 1/8/2025.

[24] "Handlebars." https://handlebarsjs.com/, 2025. Accessed 1/8/2025.

[25] M. Schäfer, S. Nadi, A. Eghbali, and F. Tip, "An empirical evaluation of using Large Language Models for automated unit test generation," *IEEE Trans. Software Eng.*, vol. 50, no. 1, pp. 85–105, 2024.

[26] G. Gay and A. Salahirad, "How closely are common mutation operators coupled to real faults?," in *2023 IEEE Conference on Software Testing, Verification and Validation (ICST)*, pp. 129–140, 2023.

[27] P. Gyimesi, B. Vancsics, A. Stocco, D. Mazinanian, Árpád Beszédes, R. Ferenc, and A. Mesbah, "BugJS: A benchmark of javascript bugs," in *Proceedings of 12th IEEE International Conference on Software Testing, Verification and Validation (ICST)*, 2019.

[28] A. Shi, J. Bell, and D. Marinov, "Mitigating the effects of flaky tests on mutation testing," in *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2019, p. 112–122, 2019.

[29] C. Lemieux, J. P. Inala, S. K. Lahiri, and S. Sen, "CodaMOSA: Escaping coverage plateaus in test generation with pre-trained large language models," in *45th International Conference on Software Engineering*, ICSE, 2023.

[30] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia Medica*, vol. 22, no. 3, pp. 276–282, 2012.

[31] M. Papadakis, M. Kintis, J. Zhang, Y. Jia, Y. L. Traon, and M. Harman, "Mutation testing advances: An analysis and survey," *Adv. Comput.*, vol. 112, pp. 275–378, 2019.

[32] R. Degiovanni and M. Papadakis, "μbert: Mutation testing using pre-trained language models," in *15th IEEE International Conference on Software Testing, Verification and Validation Workshops ICST Workshops 2022, Valencia, Spain, April 4-13, 2022*, pp. 160–169, IEEE, 2022.

[33] A. Khanfir, R. Degiovanni, M. Papadakis, and Y. L. Traon, "Efficient mutation testing via pre-trained language models," *CoRR*, vol. abs/2301.03543, 2023.

[34] A. Garg, R. Degiovanni, M. Papadakis, and Y. L. Traon, "On the coupling between vulnerabilities and LLM-generated mutants: A study on Vul4J dataset," in *IEEE Conference on Software Testing, Verification and Validation, ICST 2024, Toronto, ON, Canada, May 27-31, 2024*, pp. 305–316, IEEE, 2024.

[35] Z. Tian, H. Shu, D. Wang, X. Cao, Y. Kamei, and J. Chen, "Large language models for equivalent mutant detection: How far are we?," in *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2024, Vienna, Austria, September 16-20, 2024* (M. Christakis and M. Pradel, eds.), pp. 1733–1745, 2024.

[36] L. van Hijfte and A. Oprescu, "Mutantbench: an equivalent mutant problem comparison framework," in *14th IEEE International Conference on Software Testing, Verification and Validation Workshops, ICST Workshops 2021, Porto de Galinhas, Brazil, April 12-16, 2021*, pp. 7–12, IEEE, 2021.

[37] Y.-S. Ma, J. Offutt, and Y.-R. Kwon, "MuJava: a mutation system for Java," in *Proceedings of the 28th International Conference on Software Engineering*, ICSE '06, (New York, NY, USA), p. 827–830, Association for Computing Machinery, 2006.

[38] B. Wang, M. Chen, Y. Lin, M. Papadakis, and J. M. Zhang, "An exploratory study on using large language models for mutation testing," 2024.

[39] S. Kang and S. Yoo, "Towards objective-tailored genetic improvement through Large Language Models," in *IEEE/ACM International Workshop on Genetic Improvement, GI@ICSE 2023, Melbourne, Australia, May 20, 2023*, pp. 19–20, IEEE, 2023.

[40] A. E. I. Brownlee, J. Callan, K. Even-Mendoza, A. Geiger, C. Hanna, J. Petke, F. Sarro, and D. Sobania, "Enhancing genetic improvement mutations using Large Language Models," in *Search-Based Software Engineering - 15th International Symposium, SSBSE 2023, San Francisco, CA, USA, December 8, 2023, Proceedings* (P. Arcaini, T. Yue, and E. M. Fredericks, eds.), vol. 14415 of *Lecture Notes in Computer Science*, pp. 153–159, Springer, 2023.

[41] Z. Li and D. Shin, "Mutation-based consistency testing for evaluating the code understanding capability of LLMs," in *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI, CAIN 2024, Lisbon, Portugal, April 14-15, 2024* (J. Cleland-Huang, J. Bosch, H. Muccini, and G. A. Lewis, eds.), pp. 150–159, ACM, 2024.

[42] A. M. Dakhel, A. Nikanjam, V. Majdinasab, F. Khomh, and M. C. Desmarais, "Effective test generation using pre-trained Large Language Models and mutation testing," *Inf. Softw. Technol.*, vol. 171, p. 107468, 2024.

[43] B. Kushigian, S. J. Kaufman, R. Featherman, H. Potter, A. Madadi, and R. Just, "Equivalent mutants in the wild: Identifying and efficiently suppressing equivalent mutants for Java programs," in *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2024, Vienna, Austria,*

September 16-20, 2024 (M. Christakis and M. Pradel, eds.), pp. 654–665, ACM, 2024.

[44] Y. Jia and M. Harman, "Higher order mutation testing," *Inf. Softw. Technol.*, vol. 51, p. 1379–1393, oct 2009.

[45] D. B. Brown, M. Vaughn, B. Liblit, and T. Reps, "The care and feeding of wild-caught mutants," in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2017, (New York, NY, USA), p. 511–522, Association for Computing Machinery, 2017.

[46] M. Beller, C.-P. Wong, J. Bader, A. Scott, M. Machalica, S. Chandra, and E. Meijer, "What it would take to use mutation testing in industry: a study at facebook," in *Proceedings of the 43rd International Conference on Software Engineering: Software Engineering in Practice*, ICSE-SEIP '21, p. 268–277, IEEE Press, 2021.

[47] A. Belén Sánchez, J. A. Parejo, S. Segura, A. D. Toro, and M. Papadakis, "Mutation testing in practice: Insights from open-source software developers," *IEEE Trans. Software Eng.*, vol. 50, no. 5, pp. 1130–1143, 2024.

[48] D. Schuler and A. Zeller, "Javalanche: efficient mutation testing for Java," in *Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, ESEC/FSE '09, (New York, NY, USA), p. 297–298, Association for Computing Machinery, 2009.

[49] S. A. Irvine, T. Pavlinic, L. Trigg, J. G. Cleary, S. Inglis, and M. Utting, "Jumble java byte code to measure the effectiveness of unit tests," in *Proceedings of the Testing: Academic and Industrial Conference Practice and Research Techniques - MUTATION*, TAICPART-MUTATION '07, (USA), p. 169–175, IEEE Computer Society, 2007.

[50] L. Madeyski, "Judy – a mutation testing tool for java," *IET Software*, vol. 4, pp. 32–42(10), February 2010.

[51] R. Gopinath, I. Ahmed, M. A. Alipour, C. Jensen, and A. Groce, "Does choice of mutation tool matter?," *Software Quality Journal*, vol. 25, no. 3, pp. 871–920, 2017.

[52] S. Deb, K. Jain, R. Van Tonder, C. Le Goues, and A. Groce, "Syntax is all you need: A universal-language approach to mutant generation," in *Proceedings of the ACM on Software Engineering*, FSE 2024, 2024.

[53] R. A. DeMillo and A. J. Offutt, "Constraint-based automatic test data generation," *IEEE Trans. Software Eng.*, vol. 17, no. 9, pp. 900–910, 1991.

[54] G. Fraser and A. Zeller, "Mutation-driven generation of unit tests and oracles," in *Proceedings of the Nineteenth International Symposium on Software Testing and Analysis, ISSTA 2010, Trento, Italy, July 12-16, 2010* (P. Tonella and A. Orso, eds.), pp. 147–158, ACM, 2010.

[55] T. T. Chekam, M. Papadakis, M. Cordy, and Y. L. Traon, "Killing stubborn mutants with symbolic execution," *ACM Trans. Softw. Eng. Methodol.*, vol. 30, no. 2, pp. 19:1–19:23, 2021.

[56] J. Lee, E. Viganò, O. Cornejo, F. Pastore, and L. C. Briand, "Fuzzing for CPS mutation testing," in *38th IEEE/ACM International Conference on Automated Software Engineering, ASE 2023, Luxembourg, September 11-15, 2023*, pp. 1377–1389, IEEE, 2023.

[57] M. Tufano, D. Drain, A. Svyatkovskiy, S. K. Deng, and N. Sundaresan, "Unit test case generation with transformers and focal context." arXiv, May 2021.

[58] P. Bareiß, B. Souza, M. d'Amorim, and M. Pradel, "Code Generation Tools (Almost) for Free? A Study of Few-Shot, Pre-Trained Language Models on Code," *CoRR*, vol. abs/2206.01335, 2022.

[59] C. Pacheco and M. D. Ernst, "Randoop: Feedback-directed random testing for java," in *Companion to the 22Nd ACM SIGPLAN Conference on Object-oriented Programming Systems and Applications Companion*, OOPSLA '07, (New York, NY, USA), pp. 815–816, ACM, 2007.

[60] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[61] A. Panichella, F. M. Kifetew, and P. Tonella, "Automated test case generation as a many-objective optimisation problem with dynamic selection of the targets," *IEEE Transactions on Software Engineering*, vol. 44, no. 2, pp. 122–158, 2018.

LLMorpheus: Mutation Testing using Large Language Models

SUPPLEMENTAL MATERIALS

Frank Tip, Jonathan Bell, Max Schäfer

A APPENDIX: EXPERIMENTAL DATA

This appendix contains the following experimental results:

- Section A.1 expands on the results reported in Sections 4.3 and 4.8 by including the results for 5 experiments using the *codellama-34b-instruct* LLM at temperature 0.0.
- Sections A.2, A.3, and 1.4 expand on the results reported in Section 4.5 by including the results for 5 experiments using the *codellama-34b-instruct* LLM at temperatures 0.25, 0.50, and 1.0, respectively.
- Section 1.5 expands on the discussion in Sections 4.3 and 4.8 about the variability of the mutants observed in 5 experiments using the *codellama-34b-instruct* LLM, at temperatures 0.25, 0.50, and 1.0, respectively.
- Section 1.6 expands on the results reported in Section 4.7 by including results for 5 experiments using the *codellama-13b-instruct* LLM at temperature 0.0.
- Section 1.7 expands on the results reported in Section 4.7 by including results for 5 experiments using the *mixtral-8x7b-instruct* LLM at temperature 0.0.
- Section 1.8 expands on the results reported in Section 4.7 by including results for 5 experiments using the *gpt-4o-mini* LLM at temperature 0.0.
- Section 1.9 expands on the results reported in Section 4.7 by including results for 5 experiments using the *llama-3.3-70b-instruct* LLM at temperature 0.0.
- Section 1.10 expands on the results reported in Section 4.6 by including results for 5 experiments using the `onemutation` template.
- Section 1.11 expands on the results reported in Section 4.6 by including results for 5 experiments using the `noexplanation` template.
- Section 1.12 expands on the results reported in Section 4.6 by including results for 5 experiments using the `noinstructions` template.
- Section 1.13 expands on the results reported in Section 4.6 by including results for 5 experiments using a generic system prompt.
- Section 1.14 expands on the results reported in Section 4.6 by including results for 5 experiments using the `basic` template.
- Section 1.16 includes the results of running the standard mutation operators of *StrykerJS* on the subject applications.
- Section 1.17 reports measurements of the average string edit distance observed when using different LLMs.

All experimental data associated with the experiments reported on in this paper can be found at https://github.com/neu-se/mutation-testing-data.

A.1 Results for *codellama-34b-instruct*, `full` template, temperature 0.0

Tables 10–19 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the default prompt and system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,451	194	13	45	1,199	725	473	1	60.55
countries-and-timezones	106	318	89	0	12	217	188	29	0	86.64
crawler-url-parser	176	521	205	14	17	285	157	128	0	55.09
delta	462	1,367	565	10	25	767	634	101	32	86.83
image-downloader	42	124	33	2	0	89	72	17	0	80.90
node-dirty	154	450	153	15	7	275	163	100	12	63.64
node-geo-point	140	408	93	0	13	302	223	79	0	73.84
node-jsonfile	68	199	42	3	0	154	49	48	57	68.83
plural	153	442	101	42	18	281	205	75	1	73.31
pull-stream	351	1,028	238	12	9	769	441	271	57	64.76
q	1,051	3,121	1,000	34	52	2,035	158	1,792	85	11.94
spacl-core	134	395	140	10	6	239	199	39	1	83.68
zip-a-folder	49	143	41	1	1	100	23	3	74	97.00
Total	3,376	9,967	2,894	156	205	6,712	3,237	3,155	320	—

Table 10: Results from LLMorpheus experiment (run #312). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,450	194	13	45	1,198	726	471	1	60.68
countries-and-timezones	106	318	89	0	12	217	188	29	0	86.64
crawler-url-parser	176	521	208	14	17	282	156	126	0	55.32
delta	462	1,367	565	10	24	768	636	100	32	86.98
image-downloader	42	124	34	0	0	90	73	17	0	81.11
node-dirty	154	450	154	15	7	274	161	101	12	63.14
node-geo-point	140	410	95	0	13	302	223	79	0	73.84
node-jsonfile	68	199	43	3	0	153	49	47	57	69.28
plural	153	441	101	42	18	280	204	75	1	73.21
pull-stream	351	1,028	236	12	9	771	441	273	57	64.59
q	1,051	3,123	1,004	34	54	2,031	159	1,788	84	11.96
spacl-core	134	394	138	10	7	239	197	41	1	82.85
zip-a-folder	49	143	41	1	1	100	23	3	74	97.00
Total	3,376	9,968	2,902	154	207	6,705	3,236	3,150	319	—

Table 11: Results from LLMorpheus experiment (run #314). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,452	195	13	45	1,199	724	474	1	60.47
countries-and-timezones	106	318	89	0	12	217	188	29	0	86.64
crawler-url-parser	176	522	208	14	17	283	157	126	0	55.48
delta	462	1,367	566	10	26	765	633	100	32	86.93
image-downloader	42	124	35	0	0	89	72	17	0	80.90
node-dirty	154	450	154	14	7	275	161	102	12	62.91
node-geo-point	140	408	93	0	13	302	223	79	0	73.84
node-jsonfile	68	199	42	3	0	154	49	48	57	68.83
plural	153	442	101	42	18	281	205	75	1	73.31
pull-stream	351	1,028	237	12	9	770	441	272	57	64.68
q	1,051	3,121	1,000	34	52	2,035	159	1,793	83	11.89
spacl-core	134	393	137	10	7	239	197	41	1	82.85
zip-a-folder	49	143	41	1	1	100	23	3	74	97.00
Total	3,376	9,967	2,898	153	207	6,709	3,232	3,159	318	—

Table 12: Results from LLMorpheus experiment (run #315). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,452	195	13	45	1,199	725	473	1	60.55
countries-and-timezones	106	318	89	0	12	217	188	29	0	86.64
crawler-url-parser	176	521	207	14	17	283	157	125	1	55.83
delta	462	1,367	565	10	26	766	634	100	32	86.95
image-downloader	42	124	35	0	0	89	72	17	0	80.90
node-dirty	154	450	153	15	8	274	160	102	12	62.77
node-geo-point	140	410	95	0	13	302	222	80	0	73.51
node-jsonfile	68	200	43	3	0	154	49	48	57	68.83
plural	153	441	101	43	18	279	203	75	1	73.12
pull-stream	351	1,028	237	13	9	769	444	268	57	65.15
q	1,051	3,121	1,001	34	54	2,032	158	1,790	84	11.91
spacl-core	134	394	139	10	7	238	197	40	1	83.19
zip-a-folder	49	143	41	1	1	100	23	3	74	97.00
Total	3,376	9,969	2,901	156	210	6,702	3,232	3,150	320	—

Table 13: Results from LLMorpheus experiment (run #316). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,450	194	13	44	1,199	726	472	1	60.63
countries-and-timezones	106	317	89	0	12	216	187	29	0	86.57
crawler-url-parser	176	521	205	14	17	285	157	128	0	55.09
delta	462	1,367	565	10	25	767	636	100	31	86.96
image-downloader	42	124	32	2	0	90	73	17	0	81.11
node-dirty	154	450	153	15	7	275	161	102	12	62.91
node-geo-point	140	409	94	0	13	302	223	79	0	73.84
node-jsonfile	68	199	42	3	0	154	49	48	57	68.83
plural	153	442	101	43	18	280	204	75	1	73.21
pull-stream	351	1,028	236	12	9	771	441	273	57	64.59
q	1,051	3,122	1,003	34	54	2,031	159	1,788	84	11.96
spacl-core	134	394	138	10	7	239	198	40	1	83.26
zip-a-folder	49	143	41	1	1	100	23	3	74	97.00
Total	3,376	9,966	2,893	157	207	6,709	3,237	3,154	318	—

Table 14: Results from LLMorpheus experiment (run #317). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,050.00	637.85	967,508	102,517	1,070,025
countries-and-timezones	1,070.89	313.86	105,828	23,441	129,269
crawler-url-parser	1,642.70	929.43	386,223	39,175	425,398
delta	2,961.66	3,839.60	890,252	98,974	989,226
image-downloader	430.53	379.25	24,655	9,134	33,789
node-dirty	1,526.20	241.81	246,248	33,070	279,318
node-geo-point	1,411.11	987.17	316,333	30,013	346,346
node-jsonfile	690.61	474.78	57,516	14,797	72,313
plural	1,521.32	155.24	265,602	34,174	299,776
pull-stream	2,492.50	1,608.97	208,130	76,513	284,643
q	5,241.46	14,034.67	2,127,655	220,215	2,347,870
spacl-core	1,351.08	798.96	162,705	29,236	191,941
zip-a-folder	500.57	1,156.11	82,457	10,725	93,182
Total	23,890.64	25,557.70	5,841,112	721,984	6,563,096

Table 15: Results from LLMorpheus experiment (run #312). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,087.58	637.10	967,508	102,428	1,069,936
countries-and-timezones	1,070.89	313.12	105,828	23,427	129,255
crawler-url-parser	1,645.11	679.89	386,223	39,210	425,433
delta	2,941.55	3,838.23	890,252	98,951	989,203
image-downloader	430.54	377.12	24,655	9,175	33,830
node-dirty	1,526.11	248.57	246,248	33,038	279,286
node-geo-point	1,411.06	999.59	316,333	29,959	346,292
node-jsonfile	690.69	478.32	57,516	14,829	72,345
plural	1,521.04	147.85	265,602	34,164	299,766
pull-stream	2,477.99	1,400.76	208,130	76,398	284,528
q	5,231.88	14,004.86	2,127,655	220,252	2,347,907
spacl-core	1,351.05	805.30	162,705	29,283	191,988
zip-a-folder	500.57	1,152.62	82,457	10,705	93,162
Total	23,886.05	25,083.32	5,841,112	721,819	6,562,931

Table 16: Results from LLMorpheus experiment (run #314). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,000.46	630.00	967,508	102,314	1,069,822
countries-and-timezones	1,070.90	314.35	105,828	23,438	129,266
crawler-url-parser	1,644.58	1,051.08	386,223	39,105	425,328
delta	3,006.51	3,795.29	890,252	98,978	989,230
image-downloader	430.54	376.08	24,655	9,186	33,841
node-dirty	1,526.69	247.49	246,248	33,089	279,337
node-geo-point	1,411.05	1,003.93	316,333	30,010	346,343
node-jsonfile	690.67	478.93	57,516	14,803	72,319
plural	1,521.23	148.33	265,602	34,082	299,684
pull-stream	2,492.02	1,392.68	208,130	76,599	284,729
q	5,296.20	14,072.49	2,127,655	220,395	2,348,050
spacl-core	1,351.09	802.30	162,705	29,334	192,039
zip-a-folder	500.56	1,155.04	82,457	10,764	93,221
Total	23,942.51	25,467.99	5,841,112	722,097	6,563,209

Table 17: Results from LLMorpheus experiment (run #315). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,034.16	631.23	967,508	102,497	1,070,005
countries-and-timezones	1,070.91	309.40	105,828	23,444	129,272
crawler-url-parser	1,644.09	1,022.89	386,223	39,174	425,397
delta	2,983.03	3,969.11	890,252	99,003	989,255
image-downloader	430.51	374.45	24,655	9,148	33,803
node-dirty	1,563.30	251.17	246,248	33,068	279,316
node-geo-point	1,411.00	1,001.75	316,333	30,041	346,374
node-jsonfile	690.69	474.83	57,516	14,750	72,266
plural	1,521.09	151.19	265,602	34,132	299,734
pull-stream	2,541.67	1,398.89	208,130	76,567	284,697
q	5,399.09	13,959.40	2,127,655	220,191	2,347,846
spacl-core	1,351.08	959.37	162,705	29,287	191,992
zip-a-folder	510.58	1,154.14	82,457	10,725	93,182
Total	24,151.21	25,657.83	5,841,112	722,027	6,563,139

Table 18: Results from LLMorpheus experiment (run #316). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,019.69	669.52	967,508	102,524	1,070,032
countries-and-timezones	1,070.82	313.99	105,828	23,425	129,253
crawler-url-parser	1,641.20	958.26	386,223	39,160	425,383
delta	3,013.74	3,867.52	890,252	99,031	989,283
image-downloader	430.56	378.47	24,655	9,117	33,772
node-dirty	1,527.49	251.95	246,248	33,113	279,361
node-geo-point	1,451.19	1,042.75	316,333	29,894	346,227
node-jsonfile	690.66	479.95	57,516	14,803	72,319
plural	1,521.18	150.87	265,602	34,163	299,765
pull-stream	2,486.78	1,384.03	208,130	76,520	284,650
q	5,250.40	14,006.76	2,127,655	220,193	2,347,848
spacl-core	1,350.99	808.62	162,705	29,297	192,002
zip-a-folder	500.63	1,171.49	82,457	10,705	93,162
Total	23,955.33	25,484.18	5,841,112	721,945	6,563,057

Table 19: Results from LLMorpheus experiment (run #317). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

A.2 Results for *codellama-34b-instruct*, `full` template, temperature 0.25

Tables 20–29 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.25 using the default prompt and system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,460	210	9	44	1,197	730	466	1	61.07
countries-and-timezones	106	315	86	0	10	219	181	38	0	82.65
crawler-url-parser	176	515	216	7	12	260	167	93	0	64.23
delta	462	1,366	550	10	25	781	642	111	28	85.79
image-downloader	42	125	38	1	0	86	71	15	0	82.56
node-dirty	154	457	157	9	12	279	175	93	11	66.67
node-geo-point	140	381	76	1	8	293	225	68	0	76.79
node-jsonfile	68	202	45	4	2	151	52	41	58	72.85
plural	153	440	104	44	19	273	208	63	2	76.92
pull-stream	351	1,030	236	7	8	779	452	270	57	65.34
q	1,051	3,126	1,011	23	42	2,050	153	1,813	84	11.56
spacl-core	134	396	140	11	4	223	187	36	0	83.86
zip-a-folder	49	141	41	1	1	97	24	4	69	95.88
Total	3,376	9,954	2,910	127	187	6,688	3,267	3,111	310	—

Table 20: Results from LLMorpheus experiment (run #348). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,460	202	13	45	1,200	729	471	0	60.75
countries-and-timezones	106	316	87	1	5	223	198	25	0	88.79
crawler-url-parser	176	518	206	13	15	269	163	106	0	60.59
delta	462	1,362	576	5	29	752	608	110	34	85.37
image-downloader	42	123	35	3	0	85	67	18	0	78.82
node-dirty	154	457	165	17	4	271	163	96	12	64.58
node-geo-point	140	412	85	0	13	311	243	68	0	78.14
node-jsonfile	68	204	42	4	0	158	50	54	54	65.82
plural	153	438	101	35	19	283	213	68	2	75.97
pull-stream	351	1,027	241	9	5	772	450	265	57	65.67
q	1,051	3,125	1,037	45	63	1,980	144	1,756	80	11.31
spacl-core	134	393	134	10	6	225	194	30	1	86.67
zip-a-folder	49	140	41	0	1	95	36	53	6	44.21
Total	3,376	9,975	2,952	155	205	6,624	3,258	3,120	246	—

Table 21: Results from LLMorpheus experiment (run #350). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,451	199	12	45	1,195	737	457	1	61.76
countries-and-timezones	106	315	88	0	12	215	189	26	0	87.91
crawler-url-parser	176	512	211	16	19	247	144	103	0	58.30
delta	462	1,362	554	8	24	776	647	100	29	87.11
image-downloader	42	123	36	0	0	87	71	16	0	81.61
node-dirty	154	454	156	14	10	274	162	100	12	63.50
node-geo-point	140	413	103	0	13	294	218	76	0	74.15
node-jsonfile	68	201	43	4	0	154	55	43	56	72.08
plural	153	439	98	36	19	286	204	81	1	71.68
pull-stream	351	1,029	239	12	5	773	442	280	51	63.78
q	1,051	3,118	1,038	32	46	2,002	146	1,780	76	11.09
spacl-core	134	396	142	9	3	228	193	35	0	84.65
zip-a-folder	49	145	43	2	2	98	24	3	71	96.94
Total	3,376	9,958	2,950	145	198	6,629	3,232	3,100	297	—

Table 22: Results from LLMorpheus experiment (run #351). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,457	222	7	36	1,192	721	471	0	60.49
countries-and-timezones	106	317	86	2	2	227	200	27	0	88.11
crawler-url-parser	176	513	213	15	12	260	151	109	0	58.08
delta	462	1,370	576	7	22	765	619	109	37	85.75
image-downloader	42	122	37	0	3	82	62	20	0	75.61
node-dirty	154	454	160	12	15	267	158	97	12	63.67
node-geo-point	140	414	92	2	9	308	226	82	0	73.38
node-jsonfile	68	201	48	1	1	151	48	47	56	68.87
plural	153	440	101	36	24	279	212	66	1	76.34
pull-stream	351	1,033	243	8	9	773	443	277	53	64.17
q	1,051	3,121	1,009	17	51	2,044	141	1,822	81	10.86
spacl-core	134	397	145	14	8	214	183	30	1	85.98
zip-a-folder	49	142	35	1	1	105	27	2	76	98.10
Total	3,376	9,981	2,967	122	193	6,667	3,191	3,159	317	—

Table 23: Results from LLMorpheus experiment (run #352). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,451	226	11	46	1,168	717	451	0	61.39
countries-and-timezones	106	313	80	0	8	225	199	26	0	88.44
crawler-url-parser	176	523	204	15	19	268	161	107	0	60.07
delta	462	1,368	574	11	23	760	620	106	34	86.05
image-downloader	42	126	34	3	1	85	71	14	0	83.53
node-dirty	154	450	161	17	7	265	167	86	12	67.55
node-geo-point	140	412	93	1	19	296	220	76	0	74.32
node-jsonfile	68	203	41	3	1	158	55	49	54	68.99
plural	153	441	100	30	20	291	215	75	1	74.23
pull-stream	351	1,029	237	9	8	775	448	276	51	64.39
q	1,051	3,132	1,018	20	53	2,041	151	1,803	87	11.66
spacl-core	134	395	132	10	9	226	189	36	1	84.07
zip-a-folder	49	140	35	0	1	104	48	50	6	51.92
Total	3,376	9,983	2,935	130	215	6,662	3,261	3,155	246	—

Table 24: Results from LLMorpheus experiment (run #353). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,044.19	631.79	967,508	101,588	1,069,096
countries-and-timezones	1,070.90	327.25	105,828	23,471	129,299
crawler-url-parser	1,646.30	818.05	386,223	39,000	425,223
delta	2,954.37	3,844.00	890,252	99,341	989,593
image-downloader	430.54	348.86	24,655	9,217	33,872
node-dirty	1,532.06	232.35	246,248	32,400	278,648
node-geo-point	1,708.63	961.35	291,061	26,301	317,362
node-jsonfile	740.73	502.52	57,516	14,400	71,916
plural	1,537.89	149.38	265,602	33,182	298,784
pull-stream	2,522.38	1,395.14	208,130	76,091	284,221
q	5,294.80	14,085.10	2,127,655	218,620	2,346,275
spacl-core	1,351.05	728.32	162,705	29,167	191,872
zip-a-folder	500.56	1,086.06	82,457	10,557	93,014
Total	24,334.42	25,110.17	5,815,840	713,335	6,529,175

Table 25: Results from LLMorpheus experiment (run #348). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,029.62	629.66	967,508	100,540	1,068,048
countries-and-timezones	1,070.85	314.98	105,828	23,186	129,014
crawler-url-parser	1,777.23	867.78	386,223	38,916	425,139
delta	2,978.88	3,758.68	890,252	99,176	989,428
image-downloader	430.55	366.05	24,655	9,223	33,878
node-dirty	1,528.73	237.16	246,248	32,776	279,024
node-geo-point	1,411.08	1,021.61	316,333	29,301	345,634
node-jsonfile	690.69	485.24	57,516	14,071	71,587
plural	1,521.09	154.20	265,602	33,560	299,162
pull-stream	2,503.60	1,383.12	208,130	76,551	284,681
q	5,379.31	13,584.78	2,127,655	217,699	2,345,354
spacl-core	1,350.98	739.03	162,705	29,184	191,889
zip-a-folder	500.57	531.10	82,457	10,753	93,210
Total	24,173.19	24,073.41	5,841,112	714,936	6,556,048

Table 26: Results from LLMorpheus experiment (run #350). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,026.34	650.01	967,508	101,118	1,068,626
countries-and-timezones	1,070.91	309.33	105,828	23,331	129,159
crawler-url-parser	1,644.86	811.88	386,223	39,215	425,438
delta	2,962.02	3,896.15	890,252	99,274	989,526
image-downloader	430.54	373.38	24,655	9,163	33,818
node-dirty	1,526.45	247.44	246,248	32,894	279,142
node-geo-point	1,421.09	935.39	316,333	29,830	346,163
node-jsonfile	690.68	504.57	57,516	14,702	72,218
plural	1,521.32	151.42	265,602	33,298	298,900
pull-stream	2,497.56	1,351.06	208,130	76,100	284,230
q	5,341.32	13,704.14	2,127,655	218,805	2,346,460
spacl-core	1,351.03	756.66	162,705	28,939	191,644
zip-a-folder	500.57	1,119.70	82,457	10,786	93,243
Total	23,984.67	24,811.13	5,841,112	717,455	6,558,567

Table 27: Results from LLMorpheus experiment (run #351). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,031.72	631.91	967,508	102,075	1,069,583
countries-and-timezones	1,070.83	324.27	105,828	23,502	129,330
crawler-url-parser	1,642.15	835.28	386,223	39,240	425,463
delta	2,969.01	3,912.89	890,252	99,383	989,635
image-downloader	430.57	486.28	24,655	9,228	33,883
node-dirty	1,527.16	237.65	246,248	32,850	279,098
node-geo-point	1,411.10	1,015.42	316,333	28,895	345,228
node-jsonfile	690.64	496.41	57,516	14,557	72,073
plural	1,521.03	148.10	265,602	33,162	298,764
pull-stream	2,509.05	1,370.63	208,130	75,917	284,047
q	5,300.14	14,131.12	2,127,655	218,921	2,346,576
spacl-core	1,351.01	712.33	162,705	28,809	191,514
zip-a-folder	500.63	1,266.61	82,457	10,707	93,164
Total	23,955.04	25,568.93	5,841,112	717,246	6,558,358

Table 28: Results from LLMorpheus experiment (run #352). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,082.97	613.65	967,508	101,316	1,068,824
countries-and-timezones	1,070.86	326.66	105,828	22,979	128,807
crawler-url-parser	1,641.17	853.01	386,223	38,790	425,013
delta	2,952.50	3,773.99	890,252	99,524	989,776
image-downloader	470.58	359.76	24,655	8,898	33,553
node-dirty	1,526.98	239.71	246,248	32,476	278,724
node-geo-point	1,411.01	1,001.71	316,333	29,427	345,760
node-jsonfile	690.74	500.76	57,516	14,495	72,011
plural	1,521.11	158.00	265,602	33,838	299,440
pull-stream	2,510.21	1,355.46	208,130	75,432	283,562
q	5,390.06	14,133.05	2,127,655	217,855	2,345,510
spacl-core	1,350.98	752.72	162,705	29,399	192,104
zip-a-folder	520.57	564.11	82,457	10,749	93,206
Total	24,139.74	24,632.57	5,841,112	715,178	6,556,290

Table 29: Results from LLMorpheus experiment (run #353). Model: *codellama-34b-instruct*, temperature: 0.25, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

A.3 Results for *codellama-34b-instruct*, `full` template, temperature 0.5

Tables 30–39 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.5 using the default prompt and system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,448	211	7	39	1,191	739	452	0	62.05
countries-and-timezones	106	311	81	3	3	224	194	30	0	86.61
crawler-url-parser	176	510	185	14	13	298	166	108	24	63.76
delta	462	1,369	560	13	27	769	642	93	34	87.91
image-downloader	42	126	35	1	1	89	68	21	0	76.40
node-dirty	154	454	162	8	7	277	158	107	12	61.37
node-geo-point	140	409	87	2	18	302	230	72	0	76.16
node-jsonfile	68	201	41	3	4	153	51	43	59	71.90
plural	153	438	105	23	21	289	219	69	1	76.12
pull-stream	351	1,037	225	10	6	796	465	278	53	65.08
q	1,051	3,114	977	25	39	2,073	163	1,823	87	12.06
spacl-core	134	393	132	3	8	250	210	39	1	84.40
zip-a-folder	49	122	33	1	1	87	48	33	6	62.07
Total	3,376	9,932	2,834	113	187	6,798	3,353	3,168	277	—

Table 30: Results from LLMorpheus experiment (run #318). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,456	250	7	33	1,166	714	452	0	61.23
countries-and-timezones	106	316	81	4	10	221	192	29	0	86.88
crawler-url-parser	176	515	176	11	18	310	173	137	0	55.81
delta	462	1,366	548	8	18	792	668	89	35	88.76
image-downloader	42	126	36	3	5	82	54	28	0	65.85
node-dirty	154	456	147	8	13	288	173	104	11	63.89
node-geo-point	140	414	91	2	12	309	230	79	0	74.43
node-jsonfile	68	200	44	3	1	152	53	38	61	75.00
plural	153	440	102	30	23	285	215	70	0	75.44
pull-stream	351	1,028	218	9	17	784	463	270	51	65.56
q	1,051	3,125	1,017	36	52	2,020	166	1,774	80	12.18
spacl-core	134	392	120	10	5	257	221	35	1	86.38
zip-a-folder	49	143	36	3	1	103	29	5	69	95.15
Total	3,376	9,977	2,866	134	208	6,769	3,351	3,110	308	—

Table 31: Results from LLMorpheus experiment (run #319). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,459	232	10	35	1,182	730	452	0	61.76
countries-and-timezones	106	315	86	4	3	222	199	23	0	89.64
crawler-url-parser	176	516	201	15	16	284	166	117	1	58.80
delta	462	1,357	567	9	21	760	621	104	35	86.32
image-downloader	42	125	33	3	4	85	60	25	0	70.59
node-dirty	154	457	156	9	13	279	166	101	12	63.80
node-geo-point	140	371	88	3	11	269	211	58	0	78.44
node-jsonfile	68	200	46	4	0	150	54	39	57	74.00
plural	153	417	88	22	14	293	208	84	1	71.33
pull-stream	351	1,034	205	5	11	813	476	278	59	65.81
q	1,051	3,127	987	30	53	2,057	143	1,838	76	10.65
spacl-core	134	395	124	7	4	260	217	42	1	83.85
zip-a-folder	49	143	27	0	2	114	31	4	79	96.49
Total	3,376	9,916	2,840	121	187	6,768	3,282	3,165	321	—

Table 32: Results from LLMorpheus experiment (run #320). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,456	216	9	41	1,190	716	474	0	60.17
countries-and-timezones	106	306	70	6	9	221	191	29	1	86.88
crawler-url-parser	176	512	202	18	13	279	158	121	0	56.63
delta	462	1,358	554	12	17	775	634	105	36	86.45
image-downloader	42	125	36	2	2	85	68	17	0	80.00
node-dirty	154	454	166	12	7	269	152	105	12	60.97
node-geo-point	140	409	81	3	13	312	229	83	0	73.40
node-jsonfile	68	201	40	1	0	160	58	36	66	77.50
plural	153	435	96	30	13	296	232	63	1	78.72
pull-stream	351	1,034	232	5	8	789	466	268	55	66.03
q	1,051	3,126	1,015	23	55	2,033	152	1,784	97	12.25
spacl-core	134	391	118	5	6	262	223	38	1	85.50
zip-a-folder	49	144	41	1	2	100	45	50	5	50.00
Total	3,376	9,951	2,867	127	186	6,771	3,324	3,173	274	—

Table 33: Results from LLMorpheus experiment (run #321). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,451	219	7	39	1,186	725	461	0	61.13
countries-and-timezones	106	311	78	1	8	224	197	27	0	87.95
crawler-url-parser	176	518	200	8	13	297	173	124	0	58.25
delta	462	1,362	535	8	28	791	660	92	39	88.37
image-downloader	42	125	40	1	4	80	68	12	0	85.00
node-dirty	154	455	159	8	5	283	161	110	12	61.13
node-geo-point	140	413	98	0	10	305	239	66	0	78.36
node-jsonfile	68	204	41	2	1	160	51	42	67	73.75
plural	153	439	97	30	23	289	225	64	0	77.85
pull-stream	351	1,032	216	3	7	806	490	255	61	68.36
q	1,051	3,120	1,002	31	59	2,028	141	1,801	86	11.19
spacl-core	134	397	133	8	4	252	212	39	1	84.52
zip-a-folder	49	144	39	1	1	103	27	5	71	95.15
Total	3,376	9,971	2,857	108	202	6,804	3,369	3,098	337	—

Table 34: Results from LLMorpheus experiment (run #322). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,016.85	592.07	967,508	100,627	1,068,135
countries-and-timezones	1,070.93	319.44	105,828	22,932	128,760
crawler-url-parser	1,668.66	1,244.64	386,223	38,920	425,143
delta	2,980.99	3,869.42	890,252	98,347	988,599
image-downloader	430.55	365.90	24,655	8,925	33,580
node-dirty	1,532.57	247.29	246,248	32,995	279,243
node-geo-point	1,411.06	1,006.90	316,333	29,454	345,787
node-jsonfile	720.71	494.61	57,516	15,051	72,567
plural	1,522.56	152.03	265,602	33,550	299,152
pull-stream	2,506.81	1,378.63	208,130	75,239	283,369
q	5,332.46	14,311.02	2,127,655	217,886	2,345,541
spacl-core	1,351.01	850.39	162,705	28,919	191,624
zip-a-folder	900.98	471.62	72,362	9,438	81,800
Total	24,446.14	25,303.97	5,831,017	712,283	6,543,300

Table 35: Results from LLMorpheus experiment (run #318). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,065.41	611.71	967,508	100,190	1,067,698
countries-and-timezones	1,070.87	313.43	105,828	22,897	128,725
crawler-url-parser	1,643.05	1,045.88	386,223	39,148	425,371
delta	3,028.61	4,054.65	890,252	99,121	989,373
image-downloader	430.56	496.08	24,655	8,793	33,448
node-dirty	1,527.72	252.26	246,248	33,054	279,302
node-geo-point	1,411.09	1,060.50	316,333	28,836	345,169
node-jsonfile	690.68	520.06	57,516	14,997	72,513
plural	1,522.17	146.72	265,602	33,944	299,546
pull-stream	2,517.82	1,365.08	208,130	75,400	283,530
q	5,232.53	13,865.94	2,127,655	217,305	2,344,960
spacl-core	1,351.05	848.71	162,705	28,593	191,298
zip-a-folder	500.59	1,105.81	82,457	10,544	93,001
Total	23,992.16	25,686.82	5,841,112	712,822	6,553,934

Table 36: Results from LLMorpheus experiment (run #319). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,080.04	618.08	967,508	102,242	1,069,750
countries-and-timezones	1,070.88	311.67	105,828	22,556	128,384
crawler-url-parser	1,645.90	965.22	386,223	38,423	424,646
delta	2,992.70	3,820.99	890,252	99,184	989,436
image-downloader	430.53	500.27	24,655	9,240	33,895
node-dirty	1,527.92	252.60	246,248	32,911	279,159
node-geo-point	1,740.11	890.54	289,389	26,285	315,674
node-jsonfile	690.70	487.19	57,516	14,355	71,871
plural	1,677.14	158.69	249,979	30,944	280,923
pull-stream	2,510.30	1,455.39	208,130	75,369	283,499
q	5,453.21	14,010.80	2,127,655	217,999	2,345,654
spacl-core	1,351.08	861.29	162,705	28,654	191,359
zip-a-folder	500.56	1,278.26	82,457	10,677	93,134
Total	24,671.06	25,610.98	5,798,545	708,839	6,507,384

Table 37: Results from LLMorpheus experiment (run #320). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,025.70	617.91	967,508	101,251	1,068,759
countries-and-timezones	1,070.86	333.07	105,828	23,224	129,052
crawler-url-parser	1,686.57	939.88	386,223	39,014	425,237
delta	3,020.99	3,992.69	890,252	99,683	989,935
image-downloader	430.56	347.98	24,655	9,059	33,714
node-dirty	1,526.16	227.18	246,248	32,693	278,941
node-geo-point	1,411.04	1,057.16	316,333	29,723	346,056
node-jsonfile	690.68	565.63	57,516	14,528	72,044
plural	1,521.10	156.75	265,602	34,049	299,651
pull-stream	2,516.94	1,385.89	208,130	75,071	283,201
q	5,280.14	14,131.09	2,127,655	217,911	2,345,566
spacl-core	1,350.99	879.83	162,705	28,767	191,472
zip-a-folder	510.57	542.42	82,457	10,709	93,166
Total	24,042.30	25,177.49	5,841,112	715,682	6,556,794

Table 38: Results from LLMorpheus experiment (run #321). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,058.33	648.14	967,508	100,962	1,068,470
countries-and-timezones	1,070.88	328.85	105,828	23,165	128,993
crawler-url-parser	1,645.45	966.36	386,223	38,649	424,872
delta	2,954.29	4,010.15	890,252	97,995	988,247
image-downloader	430.57	330.02	24,655	9,182	33,837
node-dirty	1,528.13	239.77	246,248	32,568	278,816
node-geo-point	1,411.08	1,024.88	316,333	29,330	345,663
node-jsonfile	690.67	543.04	57,516	14,833	72,349
plural	1,522.09	148.56	265,602	33,906	299,508
pull-stream	2,509.37	1,433.54	208,130	75,574	283,704
q	5,254.98	13,947.69	2,127,655	216,579	2,344,234
spacl-core	1,351.03	834.35	162,705	29,103	191,808
zip-a-folder	500.60	1,144.50	82,457	10,347	92,804
Total	23,927.46	25,599.86	5,841,112	712,193	6,553,305

Table 39: Results from LLMorpheus experiment (run #322). Model: *codellama-34b-instruct*, temperature: 0.5, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

1.4 Results for *codellama-34b-instruct*, `full` template, temperature 1.0

Tables 40–49 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 1.0 using the default prompt and system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,405	345	5	27	1,028	648	379	1	63.13
countries-and-timezones	106	300	108	2	4	186	156	30	0	83.87
crawler-url-parser	176	490	202	4	6	278	202	76	0	72.66
delta	462	1,250	519	11	22	698	583	83	32	88.11
image-downloader	42	118	42	1	0	75	53	22	0	70.67
node-dirty	154	438	175	9	8	246	150	84	12	65.85
node-geo-point	140	405	118	4	10	273	213	60	0	78.02
node-jsonfile	68	187	52	1	2	132	50	22	60	83.33
plural	153	417	96	8	14	299	229	69	1	76.92
pull-stream	351	999	239	8	9	743	461	235	47	68.37
q	1,051	3,012	1,057	23	33	1,899	147	1,671	81	12.01
spacl-core	134	381	147	9	7	218	180	38	0	82.57
zip-a-folder	49	142	45	0	1	96	54	38	4	60.42
Total	3,376	9,544	3,145	85	143	6,171	3,126	2,807	238	—

Table 40: Results from LLMorpheus experiment (run #341). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,382	363	9	23	987	611	376	0	61.90
countries-and-timezones	106	254	79	7	4	164	145	19	0	88.41
crawler-url-parser	176	497	228	6	6	257	177	80	0	68.87
delta	462	1,307	580	7	19	701	596	75	30	89.30
image-downloader	42	115	52	1	3	59	46	13	0	77.97
node-dirty	154	438	159	4	7	268	164	96	8	64.18
node-geo-point	140	404	123	1	10	270	209	61	0	77.41
node-jsonfile	68	199	42	4	3	150	67	16	67	89.33
plural	153	442	114	11	12	304	245	58	1	80.92
pull-stream	351	1,012	255	11	5	741	460	238	43	67.88
q	1,051	3,025	1,080	16	35	1,894	169	1,652	73	12.78
spacl-core	134	372	162	0	5	205	173	32	0	84.39
zip-a-folder	49	130	44	2	0	84	23	4	57	95.24
Total	3,376	9,577	3,281	79	132	6,084	3,085	2,720	279	—

Table 41: Results from LLMorpheus experiment (run #342). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,378	346	7	23	1,002	637	365	0	63.57
countries-and-timezones	106	297	94	1	4	198	177	21	0	89.39
crawler-url-parser	176	491	207	6	9	269	173	96	0	64.31
delta	462	1,296	535	11	23	727	611	85	31	88.31
image-downloader	42	118	39	1	0	78	70	8	0	89.74
node-dirty	154	437	180	4	8	245	136	97	12	60.41
node-geo-point	140	399	132	3	5	259	194	65	0	74.90
node-jsonfile	68	190	41	2	1	146	68	14	64	90.41
plural	153	429	109	9	16	295	236	58	1	80.34
pull-stream	351	1,008	249	12	7	740	443	248	49	66.49
q	1,051	3,048	1,040	19	47	1,941	147	1,707	87	12.06
spacl-core	134	378	143	5	5	225	196	29	0	87.11
zip-a-folder	49	140	44	2	0	94	28	4	62	95.74
Total	3,376	9,609	3,159	82	148	6,219	3,116	2,797	306	—

Table 42: Results from LLMorpheus experiment (run #343). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,385	318	3	22	1,042	667	374	1	64.11
countries-and-timezones	106	290	84	1	3	202	183	19	0	90.59
crawler-url-parser	176	504	221	4	3	276	200	76	0	72.46
delta	462	1,296	531	5	16	744	639	71	34	90.46
image-downloader	42	116	38	2	1	75	49	26	0	65.33
node-dirty	154	412	142	7	3	260	145	100	15	61.54
node-geo-point	140	398	127	2	10	259	205	54	0	79.15
node-jsonfile	68	192	46	1	2	143	56	24	63	83.22
plural	153	439	120	5	11	303	242	60	1	80.20
pull-stream	351	1,006	237	9	8	752	456	238	58	68.35
q	1,051	3,030	1,109	18	24	1,879	130	1,659	90	11.71
spacl-core	134	377	129	3	5	240	221	19	0	92.08
zip-a-folder	49	139	40	1	2	96	38	3	55	96.88
Total	3,376	9,584	3,142	61	110	6,271	3,231	2,723	317	—

Table 43: Results from LLMorpheus experiment (run #345). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,366	318	6	24	1,018	640	378	0	62.87
countries-and-timezones	106	298	88	1	2	207	177	30	0	85.51
crawler-url-parser	176	494	232	4	6	252	178	74	0	70.63
delta	462	1,315	617	4	13	679	564	83	32	87.78
image-downloader	42	122	41	0	6	75	56	17	2	77.33
node-dirty	154	435	165	2	1	267	155	103	9	61.42
node-geo-point	140	400	123	3	11	263	198	65	0	75.29
node-jsonfile	68	195	46	1	7	141	60	25	56	82.27
plural	153	438	109	11	10	308	245	60	3	80.52
pull-stream	351	999	246	10	2	741	457	237	47	68.02
q	1,051	3,024	1,106	14	36	1,868	130	1,652	86	11.56
spacl-core	134	379	149	3	3	224	202	21	1	90.63
zip-a-folder	49	145	40	1	2	102	28	1	73	99.02
Total	3,376	9,610	3,280	60	123	6,145	3,090	2,746	309	—

Table 44: Results from LLMorpheus experiment (run #347). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,090.47	545.31	967,508	101,516	1,069,024
countries-and-timezones	1,070.88	268.04	105,828	23,041	128,869
crawler-url-parser	1,786.52	902.86	377,644	36,825	414,469
delta	3,134.68	3,531.25	877,266	93,814	971,080
image-downloader	430.55	457.54	24,655	9,002	33,657
node-dirty	1,526.84	215.86	246,248	32,721	278,969
node-geo-point	1,411.05	897.39	316,333	29,494	345,827
node-jsonfile	710.73	477.34	57,516	14,447	71,963
plural	1,533.27	157.43	265,602	31,993	297,595
pull-stream	2,504.20	1,268.62	208,130	73,625	281,755
q	5,355.10	13,120.09	2,127,655	213,824	2,341,479
spacl-core	1,351.04	679.82	162,705	28,574	191,279
zip-a-folder	500.58	493.11	82,457	10,747	93,204
Total	24,405.89	23,014.65	5,819,547	699,623	6,519,170

Table 45: Results from LLMorpheus experiment (run #341). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,099.95	515.00	967,508	99,831	1,067,339
countries-and-timezones	1,537.50	234.01	96,352	19,743	116,095
crawler-url-parser	1,643.69	843.29	386,223	36,746	422,969
delta	3,119.28	3,448.69	890,252	95,930	986,182
image-downloader	430.53	351.02	24,655	8,785	33,440
node-dirty	1,526.21	216.70	246,248	31,856	278,104
node-geo-point	1,422.53	877.31	316,333	30,037	346,370
node-jsonfile	690.70	537.75	57,516	14,738	72,254
plural	1,521.12	159.03	265,602	32,288	297,890
pull-stream	2,659.93	1,222.06	208,130	73,902	282,032
q	5,264.59	12,818.11	2,127,655	212,677	2,340,332
spacl-core	1,361.03	682.26	162,705	27,970	190,675
zip-a-folder	500.58	908.74	82,457	9,890	92,347
Total	24,777.63	22,813.96	5,831,636	694,393	6,526,029

Table 46: Results from LLMorpheus experiment (run #342). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,024.06	525.74	967,508	98,687	1,066,195
countries-and-timezones	1,070.86	293.27	105,828	23,318	129,146
crawler-url-parser	1,644.85	845.43	386,223	36,638	422,861
delta	2,948.43	3,663.87	890,252	96,381	986,633
image-downloader	430.52	274.68	24,655	9,228	33,883
node-dirty	1,526.11	216.05	246,248	31,882	278,130
node-geo-point	1,411.07	852.26	316,333	29,152	345,485
node-jsonfile	690.73	549.73	57,516	13,670	71,186
plural	1,521.17	153.33	265,602	31,946	297,548
pull-stream	2,506.18	1,294.61	208,130	72,667	280,797
q	5,178.82	13,300.53	2,127,655	213,258	2,340,913
spacl-core	1,351.08	743.18	162,705	27,865	190,570
zip-a-folder	500.55	1,038.21	82,457	10,518	92,975
Total	23,804.42	23,750.90	5,841,112	695,210	6,536,322

Table 47: Results from LLMorpheus experiment (run #343). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,070.12	559.72	967,508	100,998	1,068,506
countries-and-timezones	1,070.86	285.32	105,828	22,109	127,937
crawler-url-parser	1,644.31	872.80	386,223	38,267	424,490
delta	2,974.80	3,703.16	890,252	96,594	986,846
image-downloader	430.54	438.80	24,655	8,660	33,315
node-dirty	1,527.36	248.76	246,248	31,479	277,727
node-geo-point	1,411.07	843.77	316,333	29,660	345,993
node-jsonfile	690.69	512.58	57,516	14,276	71,792
plural	1,521.47	156.91	265,602	32,805	298,407
pull-stream	2,495.10	1,354.27	208,130	73,802	281,932
q	5,350.94	13,107.50	2,127,655	213,504	2,341,159
spacl-core	1,351.11	811.62	162,705	28,193	190,898
zip-a-folder	500.61	968.74	82,457	10,354	92,811
Total	24,038.96	23,863.96	5,841,112	700,701	6,541,813

Table 48: Results from LLMorpheus experiment (run #345). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,021.09	531.51	967,508	100,183	1,067,691
countries-and-timezones	1,070.89	297.61	105,828	23,229	129,057
crawler-url-parser	1,642.94	776.31	386,223	36,920	423,143
delta	2,984.22	3,374.88	890,252	96,698	986,950
image-downloader	430.54	435.23	24,655	9,024	33,679
node-dirty	1,528.77	227.23	246,248	32,731	278,979
node-geo-point	1,411.05	858.36	316,333	29,774	346,107
node-jsonfile	690.66	494.30	57,516	14,127	71,643
plural	1,521.30	172.76	265,602	32,844	298,446
pull-stream	2,481.70	1,251.69	208,130	73,022	281,152
q	5,205.03	13,013.51	2,127,655	214,495	2,342,150
spacl-core	1,351.07	731.40	162,705	27,723	190,428
zip-a-folder	500.58	1,168.69	82,457	10,656	93,113
Total	23,839.84	23,333.49	5,841,112	701,426	6,542,538

Table 49: Results from LLMorpheus experiment (run #347). Model: *codellama-34b-instruct*, temperature: 1.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

1.5 Results for variability analysis for *codellama-34b-instruct*

Tables 50–53 show the variability of the mutants observed in 5 runs using *codellama-34b-instruct*, for each of the temperature settings that were previously discussed in Sections A.1–1.4.

application	#min	#max	#distinct	#common
Complex.js	1,198	1,199	1,217	1,181 (97.04%)
countries-and-timezones	216	217	218	215 (98.62%)
crawler-url-parser	282	285	289	278 (96.19%)
delta	765	768	787	746 (94.79%)
image-downloader	89	90	90	89 (98.89%)
node-dirty	274	275	283	266 (93.99%)
node-geo-point	302	302	307	297 (96.74%)
node-jsonfile	153	154	158	149 (94.30%)
plural	279	281	288	274 (95.14%)
pull-stream	758	760	770	746 (96.88%)
q	2,031	2,035	2,053	2,014 (98.10%)
spacl-core	238	239	252	225 (89.29%)
zip-a-folder	100	100	102	98 (96.08%)

Table 50: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *codellama-34b-instruct* LLM at temperature 0.0 (run #312,#314,#315,#316,#317). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

application	#min	#max	#distinct	#common
Complex.js	1,168	1,200	2,354	447 (18.99%)
countries-and-timezones	215	227	551	53 (9.62%)
crawler-url-parser	266	285	700	67 (9.57%)
delta	752	781	1,706	247 (14.48%)
image-downloader	82	88	202	27 (13.37%)
node-dirty	265	279	600	82 (13.67%)
node-geo-point	296	314	660	99 (15.00%)
node-jsonfile	151	158	369	42 (11.38%)
plural	273	291	678	79 (11.65%)
pull-stream	764	770	1,664	280 (16.83%)
q	1,980	2,050	4,491	714 (15.90%)
spacl-core	230	244	613	58 (9.46%)
zip-a-folder	98	105	233	36 (15.45%)

Table 51: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *codellama-34b-instruct* LLM at temperature 0.25 (run #348,#350,#351,#352,#353). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

application	#min	#max	#distinct	#common
Complex.js	1,166	1,191	3,196	205 (6.41%)
countries-and-timezones	221	224	735	13 (1.77%)
crawler-url-parser	279	310	966	31 (3.21%)
delta	760	792	2,322	94 (4.05%)
image-downloader	80	89	273	13 (4.76%)
node-dirty	269	288	831	33 (3.97%)
node-geo-point	269	312	905	44 (4.86%)
node-jsonfile	150	160	487	15 (3.08%)
plural	285	296	893	30 (3.36%)
pull-stream	779	809	2,183	161 (7.38%)
q	2,020	2,073	5,964	327 (5.48%)
spacl-core	250	262	782	23 (2.94%)
zip-a-folder	87	114	313	17 (5.43%)

Table 52: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *codellama-34b-instruct* LLM at temperature 0.5 (run #318,#319,#320,#321,#322). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

application	#min	#max	#distinct	#common
Complex.js	987	1,042	4,200	17 (0.40%)
countries-and-timezones	164	207	835	2 (0.24%)
crawler-url-parser	252	278	1,171	1 (0.09%)
delta	681	744	3,013	2 (0.07%)
image-downloader	59	78	298	1 (0.34%)
node-dirty	245	268	1,075	2 (0.19%)
node-geo-point	259	273	1,111	2 (0.18%)
node-jsonfile	132	150	606	1 (0.17%)
plural	295	308	1,316	5 (0.38%)
pull-stream	736	749	2,814	30 (1.07%)
q	1,868	1,942	7,695	43 (0.56%)
spacl-core	205	240	958	2 (0.21%)
zip-a-folder	84	102	366	6 (1.64%)

Table 53: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *codellama-34b-instruct* LLM at temperature 1.0 (run #341,#342,#343,#345,#347). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

1.6 Results for *codellama-13b-instruct*

Tables 54–63 show the results for 5 experiments with the *codellama-13b-instruct* model at temperature 0.0 using the default prompt and system prompt shown in Figure 7. Table 64 shows the variability of the mutants observed in 5 runs using *codellama-13b-instruct* at temperature 0.0.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,438	339	116	28	955	553	401	1	58.01
countries-and-timezones	106	306	83	15	1	207	177	30	0	85.51
crawler-url-parser	176	506	186	51	12	247	129	118	0	52.23
delta	462	1,350	530	92	16	712	583	107	22	84.97
image-downloader	42	124	40	5	2	77	48	29	0	62.34
node-dirty	154	450	161	33	11	245	142	92	11	62.45
node-geo-point	140	406	64	21	16	304	237	67	0	77.96
node-jsonfile	68	198	43	10	7	138	43	45	50	67.39
plural	153	424	100	99	17	208	154	53	1	74.52
pull-stream	351	1,015	279	54	13	669	386	237	46	64.57
q	1,051	3,048	901	379	55	1,713	122	1,518	73	11.38
spacl-core	134	384	142	40	7	185	160	25	0	86.49
zip-a-folder	49	138	43	7	1	87	27	55	5	36.78
Total	3,376	9,787	2,911	922	186	5,747	2,761	2,777	209	—

Table 54: Results from LLMorpheus experiment (run #354). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,439	340	116	28	955	553	401	1	58.01
countries-and-timezones	106	306	83	15	1	207	177	30	0	85.51
crawler-url-parser	176	506	186	51	12	247	129	118	0	52.23
delta	462	1,350	530	92	16	712	583	107	22	84.97
image-downloader	42	124	40	5	2	77	48	29	0	62.34
node-dirty	154	450	161	33	11	245	142	92	11	62.45
node-geo-point	140	406	64	21	16	304	237	67	0	77.96
node-jsonfile	68	198	45	10	6	137	43	45	49	67.15
plural	153	410	98	96	16	200	148	51	1	74.50
pull-stream	351	1,015	279	54	13	669	386	237	46	64.57
q	1,051	3,047	899	379	55	1,714	122	1,519	73	11.38
spacl-core	134	384	142	40	7	185	160	25	0	86.49
zip-a-folder	49	138	43	7	1	87	27	55	5	36.78
Total	3,376	9,773	2,910	919	184	5,739	2,755	2,776	208	—

Table 55: Results from LLMorpheus experiment (run #355). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,439	340	116	28	955	553	401	1	58.01
countries-and-timezones	106	306	83	15	1	207	177	30	0	85.51
crawler-url-parser	176	506	186	51	12	247	129	118	0	52.23
delta	462	1,350	530	92	16	712	583	107	22	84.97
image-downloader	42	124	40	5	2	77	48	29	0	62.34
node-dirty	154	450	161	33	11	245	142	92	11	62.45
node-geo-point	140	406	64	21	16	304	237	67	0	77.96
node-jsonfile	68	198	45	10	6	137	43	45	49	67.15
plural	153	424	100	99	17	208	154	53	1	74.52
pull-stream	351	1,015	280	54	13	668	386	236	46	64.67
q	1,051	3,046	899	379	55	1,713	122	1,518	73	11.38
spacl-core	134	384	142	40	7	185	160	25	0	86.49
zip-a-folder	49	138	43	7	1	87	27	55	5	36.78
Total	3,376	9,786	2,913	922	185	5,745	2,761	2,776	208	—

Table 56: Results from LLMorpheus experiment (run #356). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,439	340	116	28	955	553	401	1	58.01
countries-and-timezones	106	306	83	15	1	207	177	30	0	85.51
crawler-url-parser	176	506	186	51	12	247	129	118	0	52.23
delta	462	1,350	530	92	16	712	583	107	22	84.97
image-downloader	42	124	40	5	2	77	48	29	0	62.34
node-dirty	154	450	161	33	11	245	142	92	11	62.45
node-geo-point	140	406	64	21	16	304	237	67	0	77.96
node-jsonfile	68	198	45	10	6	137	43	45	49	67.15
plural	153	424	100	99	17	208	154	53	1	74.52
pull-stream	351	1,015	279	54	13	669	387	236	46	64.72
q	1,051	3,047	898	379	55	1,715	122	1,520	73	11.37
spacl-core	134	384	142	40	7	185	160	25	0	86.49
zip-a-folder	49	138	43	7	1	87	27	55	5	36.78
Total	3,376	9,787	2,911	922	185	5,748	2,762	2,778	208	—

Table 57: Results from LLMorpheus experiment (run #358). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,439	340	116	28	955	553	401	1	58.01
countries-and-timezones	106	306	83	15	1	207	177	30	0	85.51
crawler-url-parser	176	506	186	51	12	247	124	118	5	52.23
delta	462	1,349	530	91	16	712	583	107	22	84.97
image-downloader	42	124	40	5	2	77	48	29	0	62.34
node-dirty	154	450	161	33	11	245	142	92	11	62.45
node-geo-point	140	406	64	21	16	304	237	67	0	77.96
node-jsonfile	68	198	45	10	6	137	43	45	49	67.15
plural	153	424	100	99	17	208	154	53	1	74.52
pull-stream	351	1,015	279	54	13	669	387	236	46	64.72
q	1,051	3,046	899	379	55	1,713	123	1,518	72	11.38
spacl-core	134	384	142	40	7	185	159	26	0	85.95
zip-a-folder	49	138	43	7	1	87	27	55	5	36.78
Total	3,376	9,785	2,912	921	185	5,746	2,757	2,777	212	—

Table 58: Results from LLMorpheus experiment (run #359). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,041.69	525.25	967,508	104,246	1,071,754
countries-and-timezones	1,070.79	310.80	105,828	23,971	129,799
crawler-url-parser	1,637.98	803.17	386,223	39,906	426,129
delta	2,993.70	3,529.05	890,252	103,085	993,337
image-downloader	430.56	460.64	24,655	9,339	33,994
node-dirty	1,526.96	211.68	246,248	34,892	281,140
node-geo-point	1,411.04	1,000.74	316,333	30,715	347,048
node-jsonfile	690.65	425.39	57,516	15,398	72,914
plural	1,521.00	111.98	265,602	34,926	300,528
pull-stream	2,489.93	1,188.59	208,130	77,308	285,438
q	5,187.67	11,850.22	2,127,655	231,175	2,358,830
spacl-core	1,350.97	616.43	162,705	30,694	193,399
zip-a-folder	500.51	496.32	82,457	11,494	93,951
Total	23,853.45	21,530.25	5,841,112	747,149	6,588,261

Table 59: Results from LLMorpheus experiment (run #354). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,065.29	504.22	967,508	104,246	1,071,754
countries-and-timezones	1,070.80	296.35	105,828	23,971	129,799
crawler-url-parser	1,646.62	799.72	386,223	39,938	426,161
delta	2,972.99	3,511.68	890,252	103,085	993,337
image-downloader	430.51	461.29	24,655	9,339	33,994
node-dirty	1,527.57	208.98	246,248	34,892	281,140
node-geo-point	1,411.07	1,010.61	316,333	30,715	347,048
node-jsonfile	690.63	424.92	57,516	15,398	72,914
plural	1,696.97	108.86	255,187	33,552	288,739
pull-stream	2,488.59	1,182.03	208,130	77,307	285,437
q	5,141.59	11,793.14	2,127,655	231,269	2,358,924
spacl-core	1,350.98	624.29	162,705	30,694	193,399
zip-a-folder	500.55	488.31	82,457	11,494	93,951
Total	23,994.15	21,414.40	5,830,697	745,900	6,576,597

Table 60: Results from LLMorpheus experiment (run #355). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,041.58	506.22	967,508	104,246	1,071,754
countries-and-timezones	1,070.84	308.67	105,828	23,971	129,799
crawler-url-parser	1,639.49	858.45	386,223	39,915	426,138
delta	2,978.12	3,447.64	890,252	103,085	993,337
image-downloader	430.52	459.07	24,655	9,339	33,994
node-dirty	1,527.40	208.98	246,248	34,892	281,140
node-geo-point	1,411.02	1,011.58	316,333	30,715	347,048
node-jsonfile	690.68	420.13	57,516	15,398	72,914
plural	1,521.00	112.81	265,602	34,926	300,528
pull-stream	2,481.30	1,179.99	208,130	77,302	285,432
q	5,249.72	11,806.54	2,127,655	231,355	2,359,010
spacl-core	1,351.04	617.86	162,705	30,694	193,399
zip-a-folder	500.58	495.97	82,457	11,494	93,951
Total	23,893.29	21,433.91	5,841,112	747,332	6,588,444

Table 61: Results from LLMorpheus experiment (run #356). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,045.49	511.92	967,508	104,246	1,071,754
countries-and-timezones	1,070.84	308.11	105,828	23,971	129,799
crawler-url-parser	1,639.50	877.27	386,223	39,906	426,129
delta	2,975.67	3,533.28	890,252	103,025	993,277
image-downloader	430.50	458.40	24,655	9,339	33,994
node-dirty	1,527.47	209.67	246,248	34,892	281,140
node-geo-point	1,411.06	994.07	316,333	30,715	347,048
node-jsonfile	690.64	421.39	57,516	15,398	72,914
plural	1,521.06	112.22	265,602	34,926	300,528
pull-stream	2,493.00	1,182.21	208,130	77,307	285,437
q	5,177.80	11,862.47	2,127,655	231,214	2,358,869
spacl-core	1,350.98	606.49	162,705	30,694	193,399
zip-a-folder	500.52	486.21	82,457	11,494	93,951
Total	23,834.54	21,563.71	5,841,112	747,127	6,588,239

Table 62: Results from LLMorpheus experiment (run #358). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,065.97	514.43	967,508	104,246	1,071,754
countries-and-timezones	1,070.82	328.45	105,828	23,951	129,779
crawler-url-parser	1,638.27	905.63	386,223	39,906	426,129
delta	2,954.39	3,471.08	890,252	103,085	993,337
image-downloader	430.50	458.13	24,655	9,339	33,994
node-dirty	1,526.96	207.80	246,248	34,892	281,140
node-geo-point	1,411.06	1,003.76	316,333	30,715	347,048
node-jsonfile	690.67	421.74	57,516	15,398	72,914
plural	1,521.03	111.45	265,602	34,926	300,528
pull-stream	2,483.02	1,186.04	208,130	77,307	285,437
q	5,276.52	11,824.34	2,127,655	231,254	2,358,909
spacl-core	1,350.99	617.51	162,705	30,694	193,399
zip-a-folder	500.53	490.95	82,457	11,494	93,951
Total	23,920.75	21,541.32	5,841,112	747,207	6,588,319

Table 63: Results from LLMorpheus experiment (run #359). Model: *codellama-13b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

application	#min	#max	#distinct	#common
Complex.js	955	955	955	955 (100.00%)
countries-and-timezones	207	207	207	207 (100.00%)
crawler-url-parser	257	257	257	257 (100.00%)
delta	712	712	713	711 (99.72%)
image-downloader	77	77	77	77 (100.00%)
node-dirty	245	245	245	245 (100.00%)
node-geo-point	305	305	305	305 (100.00%)
node-jsonfile	137	138	138	137 (99.28%)
plural	200	208	208	200 (96.15%)
pull-stream	663	664	666	661 (99.25%)
q	1,713	1,715	1,725	1,702 (98.67%)
spacl-core	195	195	197	193 (97.97%)
zip-a-folder	87	87	87	87 (100.00%)

Table 64: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *codellama-13b-instruct* LLM at temperature 0.0 (run #354,#355,#356,#358,#359). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

1.7 Results for *mixtral-8x7b-instruct*

Tables 65–74 show the results for 5 experiments with the *mixtral-8x7b-instruct* model at temperature 0.0 using the default prompt and system prompt shown in Figure 7. Table 75 shows the variability of the mutants observed in 5 runs using *mixtral-8x7b-instruct* at temperature 0.0.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,287	310	0	15	962	589	373	0	61.23
countries-and-timezones	106	277	65	2	5	205	166	39	0	80.98
crawler-url-parser	176	414	165	0	3	234	130	104	0	55.56
delta	462	1,156	452	0	24	680	516	128	36	81.18
image-downloader	42	108	38	0	1	69	46	23	0	66.67
node-dirty	154	310	109	0	10	191	111	72	8	62.30
node-geo-point	140	352	88	0	11	247	166	81	0	67.21
node-jsonfile	68	159	23	0	4	132	54	32	46	75.76
plural	153	307	73	0	8	226	166	60	0	73.45
pull-stream	351	940	255	1	6	678	386	248	44	63.42
q	1,051	2,468	772	3	50	1,643	112	1,460	71	11.14
spacl-core	134	333	152	0	3	157	134	22	1	85.99
zip-a-folder	49	117	38	0	0	78	24	44	10	43.59
Total	3,376	8,228	2,540	6	140	5,502	2,600	2,686	216	—

Table 65: Results from LLMorpheus experiment (run #360). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,313	303	0	21	989	608	381	0	61.48
countries-and-timezones	106	270	67	2	5	196	157	39	0	80.10
crawler-url-parser	176	403	177	0	1	211	114	97	0	54.03
delta	462	1,136	446	0	24	666	509	123	34	81.53
image-downloader	42	103	37	0	1	65	42	23	0	64.62
node-dirty	154	342	124	0	9	209	125	76	8	63.64
node-geo-point	140	357	79	0	16	257	180	77	0	70.04
node-jsonfile	68	158	24	0	5	129	56	27	46	79.07
plural	153	314	70	1	11	232	174	58	0	75.00
pull-stream	351	936	235	1	6	694	402	243	49	64.99
q	1,051	2,404	748	4	43	1,609	114	1,423	72	11.56
spacl-core	134	335	152	0	4	158	135	22	1	86.08
zip-a-folder	49	119	33	0	0	86	32	46	8	46.51
Total	3,376	8,190	2,495	8	146	5,501	2,648	2,635	218	—

Table 66: Results from LLMorpheus experiment (run #361). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,302	306	0	16	980	594	386	0	60.61
countries-and-timezones	106	272	70	2	3	197	154	43	0	78.17
crawler-url-parser	176	404	168	0	1	223	121	102	0	54.26
delta	462	1,136	455	0	21	660	497	129	34	80.45
image-downloader	42	103	35	0	1	67	41	26	0	61.19
node-dirty	154	349	126	0	10	213	123	81	9	61.97
node-geo-point	140	356	86	0	11	254	177	77	0	69.69
node-jsonfile	68	155	24	0	5	126	52	24	50	80.95
plural	153	311	71	0	11	229	169	60	0	73.80
pull-stream	351	920	244	1	4	671	389	236	46	64.83
q	1,051	2,470	756	3	53	1,658	115	1,477	66	10.92
spacl-core	134	332	149	0	2	159	134	24	1	84.91
zip-a-folder	49	122	41	0	0	81	23	51	7	37.04
Total	3,376	8,232	2,531	6	138	5,518	2,589	2,716	213	—

Table 67: Results from LLMorpheus experiment (run #362). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,299	305	0	25	969	587	382	0	60.58
countries-and-timezones	106	270	61	2	4	203	160	43	0	78.82
crawler-url-parser	176	401	167	0	1	220	118	102	0	53.64
delta	462	1,122	440	0	22	660	500	126	34	80.91
image-downloader	42	103	39	0	0	64	41	23	0	64.06
node-dirty	154	338	125	0	10	203	120	77	6	62.07
node-geo-point	140	358	81	0	15	258	178	80	0	68.99
node-jsonfile	68	166	25	1	5	135	56	32	47	76.30
plural	153	310	71	0	7	232	171	61	0	73.71
pull-stream	351	946	251	1	7	687	398	245	44	64.34
q	1,051	2,504	790	3	51	1,660	116	1,477	67	11.02
spacl-core	134	329	148	0	3	157	134	22	1	85.99
zip-a-folder	49	116	38	0	0	78	27	42	9	46.15
Total	3,376	8,262	2,541	7	150	5,526	2,606	2,712	208	—

Table 68: Results from LLMorpheus experiment (run #363). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,313	328	0	16	969	580	389	0	59.86
countries-and-timezones	106	269	64	2	4	199	156	43	0	78.39
crawler-url-parser	176	404	178	0	1	212	117	95	0	55.19
delta	462	1,148	458	0	16	674	521	116	37	82.79
image-downloader	42	107	40	0	1	66	42	24	0	63.64
node-dirty	154	339	121	0	10	208	124	73	11	64.90
node-geo-point	140	362	84	0	15	258	176	82	0	68.22
node-jsonfile	68	144	20	0	3	121	50	25	46	79.34
plural	153	319	75	0	12	232	177	55	0	76.29
pull-stream	351	926	241	1	6	678	390	242	46	64.31
q	1,051	2,446	751	3	48	1,644	117	1,455	72	11.50
spacl-core	134	337	156	0	3	156	134	21	1	86.54
zip-a-folder	49	114	39	0	0	75	25	42	8	44.00
Total	3,376	8,228	2,555	6	135	5,492	2,609	2,662	221	—

Table 69: Results from LLMorpheus experiment (run #364). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,756.38	507.41	960,545	96,072	1,056,617
countries-and-timezones	1,080.09	296.25	104,291	22,693	126,984
crawler-url-parser	1,697.04	762.88	384,404	33,495	417,899
delta	3,636.95	3,476.06	882,477	90,094	972,571
image-downloader	430.46	387.36	24,140	8,238	32,378
node-dirty	1,665.69	159.16	234,503	24,705	259,208
node-geo-point	1,497.29	833.16	315,891	27,864	343,755
node-jsonfile	691.90	425.95	56,273	12,371	68,644
plural	1,583.06	115.03	259,916	25,067	284,983
pull-stream	2,802.21	1,201.97	204,431	70,423	274,854
q	7,003.28	11,371.39	2,103,232	192,284	2,295,516
spacl-core	1,369.93	534.67	162,695	26,484	189,179
zip-a-folder	506.29	470.28	81,279	9,124	90,403
Total	27,720.59	20,541.58	5,774,077	638,914	6,412,991

Table 70: Results from LLMorpheus experiment (run #360). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,372.45	520.01	960,545	98,260	1,058,805
countries-and-timezones	1,071.57	285.14	104,291	22,228	126,519
crawler-url-parser	1,669.78	698.95	384,404	32,771	417,175
delta	3,199.74	3,398.17	882,477	89,050	971,527
image-downloader	432.08	378.08	24,140	8,005	32,145
node-dirty	1,554.61	185.18	242,671	26,837	269,508
node-geo-point	1,416.99	850.15	318,251	28,316	346,567
node-jsonfile	740.68	431.76	56,273	12,101	68,374
plural	1,546.99	120.70	261,626	25,293	286,919
pull-stream	2,660.01	1,259.31	204,431	70,142	274,573
q	6,149.24	11,173.93	2,103,232	188,223	2,291,455
spacl-core	1,416.02	534.70	162,695	26,751	189,446
zip-a-folder	500.53	489.08	81,279	9,372	90,651
Total	25,730.68	20,325.15	5,786,315	637,349	6,423,664

Table 71: Results from LLMorpheus experiment (run #361). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,359.29	519.50	960,545	96,727	1,057,272
countries-and-timezones	1,074.42	277.16	104,291	22,353	126,644
crawler-url-parser	1,661.26	731.82	384,404	32,772	417,176
delta	3,170.76	3,384.81	882,477	89,334	971,811
image-downloader	430.53	386.39	24,140	7,934	32,074
node-dirty	1,530.96	182.12	244,297	27,524	271,821
node-geo-point	1,413.38	829.76	318,251	27,995	346,246
node-jsonfile	690.64	444.51	56,273	11,970	68,243
plural	1,524.56	117.41	261,626	25,277	286,903
pull-stream	2,644.38	1,205.37	204,431	69,081	273,512
q	6,079.84	11,465.34	2,103,232	192,672	2,295,904
spacl-core	1,354.60	540.52	162,695	26,151	188,846
zip-a-folder	500.60	464.06	81,279	9,340	90,619
Total	25,435.21	20,548.77	5,787,941	639,130	6,427,071

Table 72: Results from LLMorpheus experiment (run #362). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,511.86	522.92	960,545	96,846	1,057,391
countries-and-timezones	1,073.78	299.00	104,291	22,090	126,381
crawler-url-parser	1,664.52	748.35	384,404	32,721	417,125
delta	3,343.71	3,332.02	882,477	88,421	970,898
image-downloader	440.52	362.20	24,140	7,972	32,112
node-dirty	1,532.40	164.86	244,297	26,801	271,098
node-geo-point	1,436.26	842.23	318,251	28,074	346,325
node-jsonfile	692.85	440.06	56,273	12,731	69,004
plural	1,525.78	117.87	261,626	25,198	286,824
pull-stream	2,725.42	1,227.21	204,431	70,751	275,182
q	6,622.34	11,507.75	2,103,232	194,705	2,297,937
spacl-core	1,359.32	532.62	162,695	26,100	188,795
zip-a-folder	500.56	463.12	81,279	9,118	90,397
Total	26,429.31	20,560.22	5,787,941	641,528	6,429,469

Table 73: Results from LLMorpheus experiment (run #363). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,649.24	528.79	960,545	98,038	1,058,583
countries-and-timezones	1,087.49	295.42	104,291	22,259	126,550
crawler-url-parser	1,672.78	705.16	384,404	32,640	417,044
delta	3,494.16	3,421.43	882,477	90,244	972,721
image-downloader	430.50	378.73	24,140	8,213	32,353
node-dirty	1,530.34	197.97	244,297	26,982	271,279
node-geo-point	1,432.72	865.00	318,251	28,015	346,266
node-jsonfile	702.02	419.40	56,273	11,348	67,621
plural	1,533.26	118.22	261,626	25,664	287,290
pull-stream	2,763.15	1,206.16	204,431	69,471	273,902
q	6,879.90	11,339.61	2,103,232	191,046	2,294,278
spacl-core	1,409.48	527.44	162,695	26,807	189,502
zip-a-folder	500.54	444.67	81,279	9,009	90,288
Total	27,085.58	20,447.99	5,787,941	639,736	6,427,677

Table 74: Results from LLMorpheus experiment (run #364). Model: *mixtral-8x7b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

application	#min	#max	#distinct	#common
Complex.js	962	989	1,425	604 (42.39%)
countries-and-timezones	196	205	287	132 (45.99%)
crawler-url-parser	225	246	349	144 (41.26%)
delta	660	680	958	431 (44.99%)
image-downloader	64	69	95	42 (44.21%)
node-dirty	191	213	307	120 (39.09%)
node-geo-point	253	263	368	169 (45.92%)
node-jsonfile	121	135	187	81 (43.32%)
plural	226	232	374	128 (34.22%)
pull-stream	669	692	981	451 (45.97%)
q	1,609	1,660	2,438	999 (40.98%)
spacl-core	178	181	261	113 (43.30%)
zip-a-folder	75	86	112	56 (50.00%)

Table 75: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *mixtral-8x7b-instruct* LLM at temperature 0.0 (run #360,#361,#362,#363,#364). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

1.8 Results for *gpt-4o-mini*

Tables 76–85 show the results for 5 experiments with the *gpt-4o-mini* model at temperature 0.0 using the default prompt and system prompt shown in Figure 7. Table 86 shows the variability of the mutants observed in 5 runs using *gpt-4o-mini* at temperature 0.0.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,470	446	0	38	986	596	390	0	60.45
countries-and-timezones	106	317	99	0	9	209	171	38	0	81.82
crawler-url-parser	176	527	227	0	11	275	181	94	0	65.82
delta	462	1,385	636	2	40	707	564	108	35	84.72
image-downloader	42	126	58	0	3	65	45	20	0	69.23
node-dirty	154	462	188	0	9	265	154	103	8	61.13
node-geo-point	140	419	86	0	20	311	225	86	0	72.35
node-jsonfile	68	204	44	0	6	154	64	26	64	83.12
plural	153	454	110	5	26	313	257	55	1	82.43
pull-stream	351	1,053	302	0	16	735	420	247	68	66.39
q	1,051	3,153	1,287	2	69	1,795	137	1,597	61	11.03
spacl-core	134	402	158	0	10	215	195	20	0	90.70
zip-a-folder	49	147	62	0	4	81	14	7	60	91.36
Total	3,376	10,119	3,703	9	261	6,111	3,023	2,791	297	—

Table 76: Results from LLMorpheus experiment (run #58). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,470	453	0	35	982	597	385	0	60.79
countries-and-timezones	106	318	105	0	9	204	166	38	0	81.37
crawler-url-parser	176	527	238	0	9	266	175	91	0	65.79
delta	462	1,385	651	1	35	698	566	97	35	86.10
image-downloader	42	126	58	0	2	66	41	25	0	62.12
node-dirty	154	462	195	0	8	259	153	99	7	61.78
node-geo-point	140	420	88	0	25	305	217	88	0	71.15
node-jsonfile	68	204	43	0	6	155	67	24	64	84.52
plural	153	453	114	6	22	311	255	55	1	82.32
pull-stream	351	1,053	300	0	14	739	436	240	63	67.52
q	1,051	3,153	1,289	7	71	1,786	132	1,588	66	11.09
spacl-core	134	402	157	0	10	216	198	18	0	91.67
zip-a-folder	49	147	63	0	2	82	22	4	56	95.12
Total	3,376	10,120	3,754	14	248	6,069	3,025	2,752	292	—

Table 77: Results from LLMorpheus experiment (run #59). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,470	455	0	33	982	594	388	0	60.49
countries-and-timezones	106	318	97	0	13	208	174	34	0	83.65
crawler-url-parser	176	527	236	0	9	268	178	90	0	66.42
delta	462	1,385	644	2	39	700	561	105	34	85.00
image-downloader	42	126	57	0	3	66	45	21	0	68.18
node-dirty	154	462	196	0	9	257	146	103	8	59.92
node-geo-point	140	419	83	0	23	310	221	89	0	71.29
node-jsonfile	68	204	45	0	7	152	67	25	60	83.55
plural	153	455	110	5	27	313	254	58	1	81.47
pull-stream	351	1,053	299	0	14	740	431	244	65	67.03
q	1,051	3,153	1,296	3	81	1,773	128	1,580	65	10.89
spacl-core	134	401	160	0	12	211	195	16	0	92.42
zip-a-folder	49	147	62	0	2	82	17	4	61	95.12
Total	3,376	10,120	3,740	10	272	6,062	3,011	2,757	294	—

Table 78: Results from LLMorpheus experiment (run #60). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,470	443	0	36	991	603	388	0	60.85
countries-and-timezones	106	318	99	0	11	208	173	35	0	83.17
crawler-url-parser	176	528	235	0	12	268	176	92	0	65.67
delta	462	1,385	641	1	40	703	563	103	37	85.35
image-downloader	42	126	55	0	3	68	44	24	0	64.71
node-dirty	154	462	190	0	9	263	157	99	7	62.36
node-geo-point	140	419	85	0	27	305	218	87	0	71.48
node-jsonfile	68	204	46	0	7	151	66	24	61	84.11
plural	153	455	114	4	22	315	262	53	0	83.17
pull-stream	351	1,053	299	0	11	743	432	244	67	67.16
q	1,051	3,153	1,298	7	70	1,778	123	1,595	60	10.29
spacl-core	134	401	153	0	12	220	200	20	0	90.91
zip-a-folder	49	147	59	0	4	83	18	4	61	95.18
Total	3,376	10,121	3,717	12	264	6,096	3,035	2,768	293	—

Table 79: Results from LLMorpheus experiment (run #61). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,468	441	0	39	988	603	385	0	61.03
countries-and-timezones	106	318	103	0	7	208	173	35	0	83.17
crawler-url-parser	176	528	235	0	9	271	181	90	0	66.79
delta	462	1,385	644	1	42	698	558	104	36	85.10
image-downloader	42	126	56	0	2	68	49	19	0	72.06
node-dirty	154	462	187	0	10	265	152	106	7	60.00
node-geo-point	140	419	86	0	24	307	216	91	0	70.36
node-jsonfile	68	204	46	0	7	151	60	30	61	80.13
plural	153	455	112	5	26	312	261	51	0	83.65
pull-stream	351	1,053	300	0	14	739	432	246	61	66.71
q	1,051	3,153	1,288	4	74	1,787	132	1,594	61	10.80
spacl-core	134	402	158	0	12	216	200	16	0	92.59
zip-a-folder	49	147	63	0	3	81	18	4	59	95.06
Total	3,376	10,120	3,719	10	269	6,091	3,035	2,771	285	—

Table 80: Results from LLMorpheus experiment (run #63). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,126.98	305.91	788,189	96,972	885,161
countries-and-timezones	1,070.51	223.26	82,293	21,679	103,972
crawler-url-parser	1,663.34	529.03	288,452	36,368	324,820
delta	2,954.87	2,245.12	705,441	91,727	797,168
image-downloader	430.37	363.94	19,697	8,185	27,882
node-dirty	1,526.46	149.87	198,716	31,214	229,930
node-geo-point	1,534.64	731.35	254,246	26,052	280,298
node-jsonfile	692.89	470.97	46,918	13,532	60,450
plural	1,522.37	93.31	218,037	32,037	250,074
pull-stream	2,570.39	1,189.58	171,725	70,357	242,082
q	5,661.54	11,513.31	1,694,668	207,545	1,902,213
spacl-core	1,350.59	881.31	137,862	26,728	164,590
zip-a-folder	500.39	819.97	65,869	10,140	76,009
Total	24,605.34	19,516.91	4,672,113	672,536	5,344,649

Table 81: Results from LLMorpheus experiment (run #58). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,231.94	306.23	788,189	97,504	885,693
countries-and-timezones	1,071.30	212.00	82,293	21,875	104,168
crawler-url-parser	1,667.41	521.15	288,452	36,277	324,729
delta	3,101.18	2,248.14	705,441	91,878	797,319
image-downloader	430.36	374.81	19,697	8,207	27,904
node-dirty	1,536.82	141.77	198,716	31,186	229,902
node-geo-point	1,411.39	711.52	254,246	25,773	280,019
node-jsonfile	690.45	478.21	46,918	13,386	60,304
plural	1,589.51	93.88	218,037	32,039	250,076
pull-stream	2,599.70	1,147.49	171,725	70,365	242,090
q	5,669.28	11,541.55	1,694,668	207,750	1,902,418
spacl-core	1,353.11	900.14	137,862	26,648	164,510
zip-a-folder	502.44	765.68	65,869	10,124	75,993
Total	24,854.90	19,442.55	4,672,113	673,012	5,345,125

Table 82: Results from LLMorpheus experiment (run #59). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,839.62	303.29	788,189	97,203	885,392
countries-and-timezones	1,070.52	210.16	82,293	21,852	104,145
crawler-url-parser	1,646.04	509.26	288,452	36,258	324,710
delta	2,774.38	2,206.72	705,441	91,952	797,393
image-downloader	430.38	370.79	19,697	8,198	27,895
node-dirty	1,527.61	142.70	198,716	31,200	229,916
node-geo-point	1,410.64	716.58	254,246	25,833	280,079
node-jsonfile	690.45	453.09	46,918	13,455	60,373
plural	1,520.69	94.22	218,037	31,898	249,935
pull-stream	2,438.84	1,171.72	171,725	69,914	241,639
q	4,712.10	11,452.86	1,694,668	207,508	1,902,176
spacl-core	1,353.02	848.87	137,862	26,698	164,560
zip-a-folder	500.38	816.75	65,869	10,016	75,885
Total	22,914.67	19,297.01	4,672,113	671,985	5,344,098

Table 83: Results from LLMorpheus experiment (run #60). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,952.22	307.21	788,189	96,710	884,899
countries-and-timezones	1,070.49	220.94	82,293	21,677	103,970
crawler-url-parser	1,647.14	494.72	288,452	36,287	324,739
delta	2,877.05	2,256.62	705,441	91,533	796,974
image-downloader	430.38	371.91	19,697	8,175	27,872
node-dirty	1,527.58	142.81	198,716	31,147	229,863
node-geo-point	1,410.65	713.98	254,246	25,849	280,095
node-jsonfile	690.43	464.79	46,918	13,608	60,526
plural	1,527.23	88.79	218,037	31,950	249,987
pull-stream	2,481.90	1,186.52	171,725	70,548	242,273
q	4,829.57	11,485.59	1,694,668	207,651	1,902,319
spacl-core	1,350.62	891.52	137,862	26,626	164,488
zip-a-folder	500.41	820.40	65,869	10,073	75,942
Total	23,295.67	19,445.81	4,672,113	671,834	5,343,947

Table 84: Results from LLMorpheus experiment (run #61). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,993.67	305.57	788,189	96,526	884,715
countries-and-timezones	1,070.53	212.32	82,293	21,735	104,028
crawler-url-parser	1,650.74	510.52	288,452	36,399	324,851
delta	3,461.17	2,263.13	705,441	92,146	797,587
image-downloader	430.40	376.81	19,697	8,150	27,847
node-dirty	1,531.95	145.81	198,716	31,238	229,954
node-geo-point	1,619.27	717.56	254,246	25,924	280,170
node-jsonfile	691.27	455.30	46,918	13,572	60,490
plural	1,527.63	89.76	218,037	31,989	250,026
pull-stream	2,527.67	1,150.99	171,725	70,511	242,236
q	5,162.96	11,468.80	1,694,668	207,007	1,901,675
spacl-core	1,350.59	885.50	137,862	26,516	164,378
zip-a-folder	501.32	792.67	65,869	10,026	75,895
Total	24,519.16	19,374.74	4,672,113	671,739	5,343,852

Table 85: Results from LLMorpheus experiment (run #63). Model: *gpt-4o-mini*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

application	#min	#max	#distinct	#common
Complex.js	982	991	1,484	613 (41.31%)
countries-and-timezones	204	209	362	95 (26.24%)
crawler-url-parser	280	289	472	153 (32.42%)
delta	698	707	1,094	413 (37.75%)
image-downloader	65	68	127	31 (24.41%)
node-dirty	257	265	417	147 (35.25%)
node-geo-point	307	313	492	189 (38.41%)
node-jsonfile	151	155	250	80 (32.00%)
plural	311	315	585	145 (24.79%)
pull-stream	728	734	1,176	411 (34.95%)
q	1,773	1,795	2,740	1,093 (39.89%)
spacl-core	229	236	408	109 (26.72%)
zip-a-folder	81	84	131	50 (38.17%)

Table 86: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *gpt-4o-mini* LLM at temperature 0.0 (run #58,#59,#60,#61,#63). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

1.9 Results for *llama-3.3-70b-instruct*

Tables 87–96 show the results for 5 experiments with the *llama-3.3-70b-instruct* model at temperature 0.0 using the default prompt and system prompt shown in Figure 7. Table 97 shows the variability of the mutants observed in 5 runs using *gpt-4o-mini* at temperature 0.0.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,470	279	0	53	1,138	690	447	1	60.72
countries-and-timezones	106	318	64	0	14	240	207	33	0	86.25
crawler-url-parser	176	528	175	0	22	319	208	111	0	65.20
delta	462	1,387	539	0	54	794	626	130	38	83.63
image-downloader	42	127	43	0	5	79	54	25	0	68.35
node-dirty	154	463	121	1	10	331	168	142	21	57.10
node-geo-point	140	421	39	0	22	358	255	103	0	71.23
node-jsonfile	68	204	25	0	6	173	64	37	72	78.61
plural	153	456	96	0	29	331	244	87	0	73.72
pull-stream	351	1,054	262	0	10	782	465	265	52	66.11
q	1,051	3,154	855	0	80	2,219	127	2,006	86	9.60
spacl-core	134	401	134	0	18	236	203	32	1	86.44
zip-a-folder	49	147	24	0	2	121	87	5	29	95.87
Total	3,376	10,130	2,656	1	325	7,121	3,398	3,423	300	—

Table 87: Results from LLMorpheus experiment (run #23). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,471	292	0	56	1,123	677	445	1	60.37
countries-and-timezones	106	318	65	0	13	240	205	35	0	85.42
crawler-url-parser	176	529	182	0	21	315	201	114	0	63.81
delta	462	1,388	537	0	48	803	640	126	37	84.31
image-downloader	42	126	44	0	5	77	45	32	0	58.44
node-dirty	154	463	128	1	14	320	160	141	19	55.94
node-geo-point	140	420	40	0	26	351	256	95	0	72.93
node-jsonfile	68	204	28	0	5	171	63	39	69	77.19
plural	153	455	91	0	25	339	239	100	0	70.50
pull-stream	351	1,053	265	0	12	776	462	262	52	66.24
q	1,051	3,155	863	0	87	2,205	124	1,995	86	9.52
spacl-core	134	401	131	0	16	244	210	33	1	86.48
zip-a-folder	49	147	25	0	2	120	18	5	97	95.83
Total	3,376	10,130	2,691	1	330	7,084	3,300	3,422	362	—

Table 88: Results from LLMorpheus experiment (run #24). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,470	285	0	59	1,126	678	447	1	60.30
countries-and-timezones	106	318	67	0	11	240	199	41	0	82.92
crawler-url-parser	176	528	179	0	24	315	203	112	0	64.44
delta	462	1,387	542	0	47	798	625	134	39	83.21
image-downloader	42	126	46	0	4	76	45	31	0	59.21
node-dirty	154	465	126	2	11	326	160	146	20	55.21
node-geo-point	140	421	39	0	19	360	257	103	0	71.39
node-jsonfile	68	204	28	0	5	171	62	41	68	76.02
plural	153	455	93	0	26	336	245	91	0	72.92
pull-stream	351	1,053	270	0	8	775	467	256	52	66.97
q	1,051	3,155	837	1	80	2,237	129	2,027	81	9.39
spacl-core	134	401	131	0	15	241	205	35	1	85.48
zip-a-folder	49	147	27	0	2	118	17	5	96	95.76
Total	3,376	10,130	2,670	3	311	7,119	3,292	3,469	358	—

Table 89: Results from LLMorpheus experiment (run #25). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,473	273	0	64	1,136	683	452	1	60.21
countries-and-timezones	106	318	62	0	13	243	208	35	0	85.60
crawler-url-parser	176	527	178	0	21	316	201	115	0	63.61
delta	462	1,388	545	0	50	793	634	128	31	83.86
image-downloader	42	128	43	0	4	81	57	24	0	70.37
node-dirty	154	464	121	1	10	332	174	140	18	57.83
node-geo-point	140	421	40	0	17	361	256	105	0	70.91
node-jsonfile	68	204	27	0	6	171	63	38	70	77.78
plural	153	460	92	0	29	339	248	91	0	73.16
pull-stream	351	1,053	266	0	9	778	464	266	48	65.81
q	1,051	3,155	853	1	83	2,218	131	2,007	80	9.51
spacl-core	134	401	124	0	14	250	215	34	1	86.40
zip-a-folder	49	147	28	0	2	117	85	5	27	95.73
Total	3,376	10,139	2,652	2	322	7,135	3,419	3,440	276	—

Table 90: Results from LLMorpheus experiment (run #26). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,477	283	1	65	1,128	680	447	1	60.37
countries-and-timezones	106	318	67	0	12	239	201	38	0	84.10
crawler-url-parser	176	528	179	0	21	318	203	115	0	63.84
delta	462	1,387	539	0	47	801	643	120	38	85.02
image-downloader	42	127	42	0	5	80	57	23	0	71.25
node-dirty	154	463	124	1	10	328	172	140	16	57.32
node-geo-point	140	421	36	0	22	360	264	96	0	73.33
node-jsonfile	68	204	25	0	4	175	67	36	72	79.43
plural	153	453	89	0	26	338	249	89	0	73.67
pull-stream	351	1,053	269	0	8	776	467	259	50	66.62
q	1,051	3,155	850	0	80	2,225	124	2,022	79	9.12
spacl-core	134	401	128	0	16	244	206	37	1	84.84
zip-a-folder	49	148	26	0	2	120	18	5	97	95.83
Total	3,376	10,135	2,657	2	318	7,132	3,351	3,427	354	—

Table 91: Results from LLMorpheus experiment (run #27). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	2,035.43	359.18	785,856	99,437	885,293
countries-and-timezones	503.27	250.43	82,111	20,809	102,920
crawler-url-parser	810.25	606.98	283,056	35,138	318,194
delta	1,881.70	2,576.74	700,458	90,055	790,513
image-downloader	199.98	362.58	19,515	7,900	27,415
node-dirty	739.35	257.29	197,447	30,494	227,941
node-geo-point	688.13	840.31	252,643	27,340	279,983
node-jsonfile	334.39	527.30	46,029	13,579	59,608
plural	748.74	93.78	218,344	32,058	250,402
pull-stream	1,483.02	1,141.94	170,717	68,203	238,920
q	3,563.55	14,535.02	1,690,971	205,818	1,896,789
spacl-core	631.83	898.43	136,724	26,473	163,197
zip-a-folder	260.98	451.69	64,841	10,049	74,890
Total	13,880.61	22,901.67	4,648,712	667,353	5,316,065

Table 92: Results from LLMorpheus experiment (run #23). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	1,935.55	354.56	785,902	98,758	884,660
countries-and-timezones	517.60	248.81	82,111	21,228	103,339
crawler-url-parser	862.45	606.56	283,056	35,246	318,302
delta	1,894.76	2,611.62	700,458	90,310	790,768
image-downloader	197.05	453.31	19,515	7,908	27,423
node-dirty	714.96	253.07	197,447	30,659	228,106
node-geo-point	700.66	831.74	252,643	27,406	280,049
node-jsonfile	325.93	518.65	46,029	13,248	59,277
plural	791.02	95.56	218,344	32,292	250,636
pull-stream	1,417.60	1,132.73	170,717	68,708	239,425
q	3,506.09	14,435.66	1,690,971	205,518	1,896,489
spacl-core	665.00	997.34	136,724	26,424	163,148
zip-a-folder	262.84	1,296.03	64,841	10,226	75,067
Total	13,791.51	23,835.63	4,648,758	667,931	5,316,689

Table 93: Results from LLMorpheus experiment (run #24). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	1,951.37	356.92	785,856	99,172	885,028
countries-and-timezones	522.56	246.91	82,111	21,238	103,349
crawler-url-parser	804.63	604.74	283,056	35,252	318,308
delta	1,862.31	2,566.04	700,458	90,059	790,517
image-downloader	195.17	449.66	19,515	7,845	27,360
node-dirty	724.87	254.24	197,447	31,135	228,582
node-geo-point	664.03	860.50	252,643	27,500	280,143
node-jsonfile	353.83	501.73	46,029	13,505	59,534
plural	742.45	96.06	218,344	32,077	250,421
pull-stream	1,454.40	1,137.23	170,717	68,279	238,996
q	3,808.92	14,578.83	1,690,971	204,821	1,895,792
spacl-core	632.50	974.79	136,724	26,240	162,964
zip-a-folder	261.77	1,272.87	64,841	10,055	74,896
Total	13,978.80	23,900.50	4,648,712	667,178	5,315,890

Table 94: Results from LLMorpheus experiment (run #25). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	1,965.89	356.55	785,856	99,239	885,095
countries-and-timezones	520.83	252.88	82,111	21,050	103,161
crawler-url-parser	845.23	415.43	283,056	35,204	318,260
delta	1,820.88	2,526.25	700,458	90,336	790,794
image-downloader	199.63	376.57	19,515	8,115	27,630
node-dirty	732.92	248.37	197,447	30,614	228,061
node-geo-point	648.30	852.65	252,643	27,289	279,932
node-jsonfile	337.73	504.46	46,029	13,500	59,529
plural	771.86	96.38	218,344	32,276	250,620
pull-stream	1,482.72	1,117.82	170,717	68,160	238,877
q	3,818.36	14,500.22	1,690,971	205,244	1,896,215
spacl-core	661.68	1,006.28	136,724	26,127	162,851
zip-a-folder	261.75	423.42	64,841	10,119	74,960
Total	14,067.79	22,677.29	4,648,712	667,273	5,315,985

Table 95: Results from LLMorpheus experiment (run #26). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	1,949.16	359.45	785,856	98,758	884,614
countries-and-timezones	518.42	249.03	82,111	21,290	103,401
crawler-url-parser	843.83	610.56	283,056	35,543	318,599
delta	1,930.29	2,578.87	700,458	90,277	790,735
image-downloader	199.97	357.91	19,515	7,985	27,500
node-dirty	711.46	241.03	197,447	30,092	227,539
node-geo-point	664.09	878.84	252,643	27,234	279,877
node-jsonfile	334.95	526.44	46,029	13,288	59,317
plural	756.46	95.20	218,344	31,877	250,221
pull-stream	1,444.48	1,125.62	170,717	68,975	239,692
q	3,636.06	14,547.79	1,690,971	205,379	1,896,350
spacl-core	641.07	1,006.59	136,724	26,304	163,028
zip-a-folder	255.10	1,275.79	64,841	9,904	74,745
Total	13,885.34	23,853.13	4,648,712	666,906	5,315,618

Table 96: Results from LLMorpheus experiment (run #27). Model: *meta-llama/llama-3.3-70b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

application	#min	#max	#distinct	#common
Complex.js	1,123	1,138	1,712	698 (40.77%)
countries-and-timezones	239	243	374	139 (37.17%)
crawler-url-parser	325	331	559	186 (33.27%)
delta	793	803	1,222	492 (40.26%)
image-downloader	76	81	131	40 (30.53%)
node-dirty	320	332	511	189 (36.99%)
node-geo-point	354	364	533	232 (43.53%)
node-jsonfile	171	175	273	97 (35.53%)
plural	331	339	598	169 (28.26%)
pull-stream	768	775	1,158	500 (43.18%)
q	2,205	2,237	2,898	1,708 (58.94%)
spacl-core	249	263	445	131 (29.44%)
zip-a-folder	117	121	172	78 (45.35%)

Table 97: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *meta-llama/llama-3.3-70b-instruct* LLM at temperature 0.0 (run #23,#24,#25,#26,#27). The columns of the table show, from left to right: (i) the minimum number of mutants observed in any of the runs, (ii) the maximum number of mutants observed in any of the runs, (iii) the total number of distinct mutants observed in all runs, and (iv) the number (percentage) of mutants are observed in all runs.

1.10 Results for `template-onemutation-0.0`

Tables 98–107 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 15 and using the system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	67	6	11	406	245	161	0	60.34
countries-and-timezones	106	106	24	1	2	79	65	14	0	82.28
crawler-url-parser	176	175	70	8	5	86	50	36	0	58.14
delta	462	461	182	4	9	266	221	37	8	86.09
image-downloader	42	42	8	0	0	34	26	8	0	76.47
node-dirty	154	155	50	3	3	99	55	41	3	58.59
node-geo-point	140	140	32	0	3	104	74	30	0	71.15
node-jsonfile	68	68	11	0	0	57	18	18	21	68.42
plural	153	153	39	8	6	100	70	30	0	70.00
pull-stream	351	351	67	3	1	280	165	95	20	66.07
q	1,051	1,052	306	26	17	703	46	630	27	10.38
spacl-core	134	134	41	3	2	80	63	17	0	78.75
zip-a-folder	49	49	9	0	1	39	19	17	3	56.41
Total	3,376	3,376	906	62	60	2,333	1,117	1,134	82	—

Table 98: Results from LLMorpheus experiment (run #365). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	67	6	11	406	244	162	0	60.10
countries-and-timezones	106	106	24	1	2	79	66	13	0	83.54
crawler-url-parser	176	175	70	8	5	86	50	36	0	58.14
delta	462	461	181	4	9	267	222	37	8	86.14
image-downloader	42	42	8	0	0	34	26	8	0	76.47
node-dirty	154	155	50	3	3	99	55	41	3	58.59
node-geo-point	140	140	32	0	3	104	74	30	0	71.15
node-jsonfile	68	68	11	0	0	57	18	18	21	68.42
plural	153	153	39	8	6	100	70	30	0	70.00
pull-stream	351	351	67	3	1	280	164	96	20	65.71
q	1,051	1,052	306	26	17	703	46	630	27	10.38
spacl-core	134	134	41	3	2	80	63	17	0	78.75
zip-a-folder	49	49	9	0	1	39	19	17	3	56.41
Total	3,376	3,376	905	62	60	2,334	1,117	1,135	82	—

Table 99: Results from LLMorpheus experiment (run #366). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	485	67	6	11	401	241	160	0	60.10
countries-and-timezones	106	106	24	1	2	79	66	13	0	83.54
crawler-url-parser	176	175	71	8	5	85	50	35	0	58.82
delta	462	461	181	4	9	267	222	37	8	86.14
image-downloader	42	42	8	0	0	34	26	8	0	76.47
node-dirty	154	155	50	3	3	99	55	41	3	58.59
node-geo-point	140	140	32	0	3	104	74	30	0	71.15
node-jsonfile	68	68	11	0	0	57	18	18	21	68.42
plural	153	153	39	7	6	101	71	30	0	70.30
pull-stream	351	351	67	3	1	280	165	95	20	66.07
q	1,051	1,052	306	26	16	704	45	632	27	10.23
spacl-core	134	134	41	3	2	80	63	17	0	78.75
zip-a-folder	49	49	9	0	1	39	19	17	3	56.41
Total	3,376	3,371	906	61	59	2,330	1,115	1,133	82	—

Table 100: Results from LLMorpheus experiment (run #369). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	67	6	11	406	245	161	0	60.34
countries-and-timezones	106	106	24	1	2	79	66	13	0	83.54
crawler-url-parser	176	175	69	8	5	87	51	36	0	58.62
delta	462	461	182	4	9	266	221	37	8	86.09
image-downloader	42	42	8	0	0	34	26	8	0	76.47
node-dirty	154	155	50	4	3	98	55	40	3	59.18
node-geo-point	140	140	32	0	3	104	74	30	0	71.15
node-jsonfile	68	68	11	0	0	57	18	18	21	68.42
plural	153	153	39	7	6	101	71	30	0	70.30
pull-stream	351	351	67	3	1	280	165	95	20	66.07
q	1,051	1,052	306	26	17	703	46	630	27	10.38
spacl-core	134	134	41	3	2	80	63	17	0	78.75
zip-a-folder	49	49	9	0	1	39	19	17	3	56.41
Total	3,376	3,376	905	62	60	2,334	1,120	1,132	82	—

Table 101: Results from LLMorpheus experiment (run #370). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	67	6	11	406	245	161	0	60.34
countries-and-timezones	106	106	24	1	2	79	66	13	0	83.54
crawler-url-parser	176	175	69	8	5	87	51	36	0	58.62
delta	462	461	182	4	9	266	221	37	8	86.09
image-downloader	42	42	8	0	0	34	26	8	0	76.47
node-dirty	154	155	50	4	3	98	55	40	3	59.18
node-geo-point	140	140	32	0	3	104	74	30	0	71.15
node-jsonfile	68	68	11	0	0	57	18	18	21	68.42
plural	153	153	39	7	6	101	71	30	0	70.30
pull-stream	351	351	67	3	1	280	165	95	20	66.07
q	1,051	1,052	306	26	17	703	46	630	27	10.38
spacl-core	134	134	41	3	2	80	63	17	0	78.75
zip-a-folder	49	49	9	0	1	39	19	17	3	56.41
Total	3,376	3,376	905	62	60	2,334	1,120	1,132	82	—

Table 102: Results from LLMorpheus experiment (run #371). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	2,784.11	210.86	927,818	39,567	967,385
countries-and-timezones	1,071.07	117.59	97,242	8,518	105,760
crawler-url-parser	1,636.44	292.18	371,967	15,504	387,471
delta	2,676.03	1,251.08	852,830	37,401	890,231
image-downloader	430.61	139.23	21,253	3,459	24,712
node-dirty	1,526.39	77.95	233,774	12,906	246,680
node-geo-point	1,411.29	330.28	304,993	11,192	316,185
node-jsonfile	690.81	183.43	52,008	5,846	57,854
plural	1,521.37	54.03	253,209	13,450	266,659
pull-stream	2,400.61	499.06	179,699	30,228	209,927
q	4,195.04	4,866.38	2,042,524	82,318	2,124,842
spacl-core	1,351.30	271.81	151,851	10,803	162,654
zip-a-folder	500.63	219.06	78,488	4,405	82,893
Total	22,195.69	8,512.94	5,567,656	275,597	5,843,253

Table 103: Results from LLMorpheus experiment (run #365). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	2,757.00	215.00	927,818	39,486	967,304
countries-and-timezones	1,091.07	116.98	97,242	8,527	105,769
crawler-url-parser	1,636.38	300.85	371,967	15,532	387,499
delta	2,681.00	1,235.39	852,830	37,383	890,213
image-downloader	460.67	139.42	21,253	3,476	24,729
node-dirty	1,526.36	75.52	233,774	12,907	246,681
node-geo-point	1,411.28	327.69	304,993	11,211	316,204
node-jsonfile	730.85	184.62	52,008	5,779	57,787
plural	1,521.30	53.73	253,209	13,418	266,627
pull-stream	2,397.86	497.98	179,699	30,310	210,009
q	4,204.99	4,839.22	2,042,524	82,262	2,124,786
spacl-core	1,351.25	273.58	151,851	10,809	162,660
zip-a-folder	500.62	219.57	78,488	4,403	82,891
Total	22,270.63	8,479.55	5,567,656	275,503	5,843,159

Table 104: Results from LLMorpheus experiment (run #366). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	2,820.16	211.48	916,945	39,061	956,006
countries-and-timezones	1,071.10	117.43	97,242	8,548	105,790
crawler-url-parser	1,636.46	288.17	371,967	15,519	387,486
delta	2,686.73	1,239.88	852,830	37,432	890,262
image-downloader	430.69	142.46	21,253	3,475	24,728
node-dirty	1,536.37	75.72	233,774	12,869	246,643
node-geo-point	1,411.33	338.80	304,993	11,209	316,202
node-jsonfile	690.78	183.73	52,008	5,845	57,853
plural	1,521.35	54.18	253,209	13,392	266,601
pull-stream	2,398.19	499.73	179,699	30,182	209,881
q	4,188.82	4,807.59	2,042,524	82,120	2,124,644
spacl-core	1,351.23	272.94	151,851	10,813	162,664
zip-a-folder	500.64	222.35	78,488	4,405	82,893
Total	22,243.85	8,454.47	5,556,783	274,870	5,831,653

Table 105: Results from LLMorpheus experiment (run #369). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	2,754.08	218.66	927,818	39,566	967,384
countries-and-timezones	1,071.03	119.64	97,242	8,579	105,821
crawler-url-parser	1,636.41	281.15	371,967	15,525	387,492
delta	2,676.11	1,249.77	852,830	37,449	890,279
image-downloader	430.60	138.45	21,253	3,475	24,728
node-dirty	1,526.42	73.92	233,774	12,859	246,633
node-geo-point	1,411.28	329.40	304,993	11,210	316,203
node-jsonfile	690.77	183.71	52,008	5,787	57,795
plural	1,521.36	53.81	253,209	13,434	266,643
pull-stream	2,397.58	499.25	179,699	30,160	209,859
q	4,211.46	4,819.77	2,042,524	82,203	2,124,727
spacl-core	1,361.19	269.34	151,851	10,818	162,669
zip-a-folder	500.61	217.85	78,488	4,400	82,888
Total	22,188.89	8,454.73	5,567,656	275,465	5,843,121

Table 106: Results from LLMorpheus experiment (run #370). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,763.09	214.80	927,818	39,505	967,323
countries-and-timezones	1,071.04	115.36	97,242	8,565	105,807
crawler-url-parser	1,636.32	285.19	371,967	15,616	387,583
delta	2,676.35	1,249.33	852,830	37,349	890,179
image-downloader	430.63	137.29	21,253	3,461	24,714
node-dirty	1,526.34	74.50	233,774	12,868	246,642
node-geo-point	1,411.31	337.02	304,993	11,183	316,176
node-jsonfile	690.79	183.54	52,008	5,774	57,782
plural	1,521.35	52.33	253,209	13,401	266,610
pull-stream	2,403.01	497.08	179,699	30,238	209,937
q	4,196.27	4,832.34	2,042,524	82,120	2,124,644
spacl-core	1,351.29	269.13	151,851	10,793	162,644
zip-a-folder	500.65	220.66	78,488	4,403	82,891
Total	22,178.44	8,468.57	5,567,656	275,276	5,842,932

Table 107: Results from LLMorpheus experiment (run #371). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-onemutation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

````
Your task is to apply mutation testing to the following code:
```
{{{code}}}
```

by replacing the PLACEHOLDER with a buggy code fragment that has different
behavior than the original code fragment, which was:
```
{{{orig}}}
```
Please consider changes such as using different operators, changing constants,
referring to different variables, object properties, functions, or methods.

Provide your answer as a fenced code block containing a single line of code,
using the following template:

The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because <brief explanation>.

Please conclude your response with "DONE."
````

Figure 15: Variation on the template of Figure 7 that requests only one mutation.

1.11 Results for `template-noexplanation-0.0`

Tables 108–113 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 16 and using the system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,465	276	27	37	1,125	676	448	1	60.18
countries-and-timezones	106	307	84	2	10	211	183	28	0	86.73
crawler-url-parser	176	514	216	23	19	239	140	99	0	58.58
delta	462	1,375	592	29	20	734	598	110	26	85.01
image-downloader	42	126	41	4	2	77	62	15	0	80.52
node-dirty	154	458	160	30	10	258	146	99	13	61.63
node-geo-point	140	414	103	3	9	297	216	81	0	72.73
node-jsonfile	68	204	48	4	0	152	54	45	53	70.39
plural	153	449	110	50	16	273	198	74	1	72.89
pull-stream	351	1,037	236	16	11	774	440	278	56	64.08
q	1,051	3,143	1,106	122	59	1,856	138	1,635	83	11.91
spacl-core	134	395	143	17	5	211	175	35	1	83.41
zip-a-folder	49	143	41	3	1	98	27	3	68	96.94
Total	3,376	10,030	3,156	330	199	6,305	3,053	2,950	302	—

Table 108: Results from LLMorpheus experiment (run #372). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,465	276	27	37	1,125	678	446	1	60.36
countries-and-timezones	106	307	84	2	10	211	183	28	0	86.73
crawler-url-parser	176	514	216	23	19	239	140	99	0	58.58
delta	462	1,375	595	28	19	733	598	108	27	85.27
image-downloader	42	126	41	4	2	77	62	15	0	80.52
node-dirty	154	458	160	30	10	258	146	99	13	61.63
node-geo-point	140	414	103	2	9	298	216	82	0	72.48
node-jsonfile	68	204	48	4	0	152	54	45	53	70.39
plural	153	449	110	50	16	273	198	74	1	72.89
pull-stream	351	1,037	236	16	11	774	439	279	56	63.95
q	1,051	3,143	1,107	122	60	1,854	138	1,633	83	11.92
spacl-core	134	395	143	18	6	209	175	33	1	84.21
zip-a-folder	49	143	41	3	1	98	26	3	69	96.94
Total	3,376	10,030	3,160	329	200	6,301	3,053	2,944	304	—

Table 109: Results from LLMorpheus experiment (run #374). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,465	280	27	37	1,121	674	446	1	60.21
countries-and-timezones	106	307	84	2	10	211	183	28	0	86.73
crawler-url-parser	176	514	216	23	19	239	140	99	0	58.58
delta	462	1,375	595	28	19	733	597	110	26	84.99
image-downloader	42	126	41	4	2	77	62	15	0	80.52
node-dirty	154	458	160	30	10	258	146	99	13	61.63
node-geo-point	140	414	103	2	9	298	216	82	0	72.48
node-jsonfile	68	204	48	4	0	152	54	45	53	70.39
plural	153	449	110	50	16	273	198	74	1	72.89
pull-stream	351	1,037	236	16	11	774	439	279	56	63.95
q	1,051	3,143	1,107	122	59	1,855	138	1,634	83	11.91
spacl-core	134	396	144	18	6	209	175	33	1	84.21
zip-a-folder	49	143	41	3	1	98	26	3	69	96.94
Total	3,376	10,031	3,165	329	199	6,298	3,048	2,947	303	—

Table 110: Results from LLMorpheus experiment (run #375). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,465	275	27	37	1,126	679	446	1	60.39
countries-and-timezones	106	307	84	2	10	211	183	28	0	86.73
crawler-url-parser	176	514	216	23	19	239	140	99	0	58.58
delta	462	1,375	598	25	19	733	597	109	27	85.13
image-downloader	42	126	41	4	2	77	62	15	0	80.52
node-dirty	154	458	160	30	10	258	146	99	13	61.63
node-geo-point	140	414	103	2	9	298	216	82	0	72.48
node-jsonfile	68	204	48	4	0	152	54	45	53	70.39
plural	153	449	110	50	16	273	198	74	1	72.89
pull-stream	351	1,037	239	16	9	773	440	277	56	64.17
q	1,051	3,143	1,108	121	60	1,854	136	1,634	84	11.87
spacl-core	134	396	144	18	6	209	177	31	1	85.17
zip-a-folder	49	143	41	3	1	98	27	3	68	96.94
Total	3,376	10,031	3,167	325	198	6,301	3,055	2,942	304	—

Table 111: Results from LLMorpheus experiment (run #376). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,465	276	27	37	1,125	678	446	1	60.36
countries-and-timezones	106	307	84	2	10	211	183	28	0	86.73
crawler-url-parser	176	514	213	26	19	239	140	99	0	58.58
delta	462	1,375	595	28	19	733	597	110	26	84.99
image-downloader	42	126	41	4	2	77	62	15	0	80.52
node-dirty	154	458	160	30	10	258	146	99	13	61.63
node-geo-point	140	414	106	2	9	295	213	82	0	72.20
node-jsonfile	68	204	48	4	0	152	54	45	53	70.39
plural	153	449	110	50	16	273	198	74	1	72.89
pull-stream	351	1,037	236	16	11	774	438	280	56	63.82
q	1,051	3,143	1,107	121	59	1,856	137	1,635	84	11.91
spacl-core	134	396	144	18	6	209	175	33	1	84.21
zip-a-folder	49	143	41	3	1	98	27	3	68	96.94
Total	3,376	10,031	3,161	331	199	6,300	3,048	2,949	303	—

Table 112: Results from LLMorpheus experiment (run #377). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,056.33	600.05	948,398	75,551	1,023,949
countries-and-timezones	1,070.68	309.71	101,694	23,742	125,436
crawler-url-parser	1,656.21	778.34	379,359	31,115	410,474
delta	2,870.28	3,625.25	872,234	64,880	937,114
image-downloader	430.47	329.74	23,017	9,110	32,127
node-dirty	1,526.57	243.81	240,242	24,279	264,521
node-geo-point	1,411.01	1,009.62	310,873	26,100	336,973
node-jsonfile	690.59	482.67	54,864	15,154	70,018
plural	1,522.48	146.41	259,635	26,465	286,100
pull-stream	2,632.74	1,388.06	194,441	73,821	268,262
q	4,694.78	12,908.57	2,086,666	127,647	2,214,313
spacl-core	1,350.87	711.26	157,479	28,201	185,680
zip-a-folder	500.54	1,097.23	80,546	10,243	90,789
Total	23,413.56	23,630.72	5,709,448	536,308	6,245,756

Table 113: Results from LLMorpheus experiment (run #372). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,088.22	620.07	948,398	75,464	1,023,862
countries-and-timezones	1,070.66	302.71	101,694	23,766	125,460
crawler-url-parser	1,653.80	799.42	379,359	31,089	410,448
delta	2,871.99	3,718.30	872,234	65,148	937,382
image-downloader	430.48	328.19	23,017	9,096	32,113
node-dirty	1,526.65	242.49	240,242	24,129	264,371
node-geo-point	1,410.97	985.29	310,873	26,143	337,016
node-jsonfile	690.54	481.86	54,864	15,125	69,989
plural	1,522.07	145.74	259,635	26,527	286,162
pull-stream	2,643.33	1,393.56	194,441	73,922	268,363
q	4,623.16	12,860.56	2,086,666	127,954	2,214,620
spacl-core	1,360.90	649.12	157,479	28,174	185,653
zip-a-folder	500.51	1,117.51	80,546	10,267	90,813
Total	23,393.28	23,644.84	5,709,448	536,804	6,246,252

Table 114: Results from LLMorpheus experiment (run #374). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,054.59	596.41	948,398	75,593	1,023,991
countries-and-timezones	1,070.75	306.48	101,694	23,740	125,434
crawler-url-parser	1,653.25	791.38	379,359	31,096	410,455
delta	2,869.41	3,656.14	872,234	64,872	937,106
image-downloader	430.47	328.44	23,017	9,109	32,126
node-dirty	1,526.59	236.98	240,242	24,096	264,338
node-geo-point	1,410.99	992.09	310,873	26,143	337,016
node-jsonfile	690.55	485.04	54,864	15,130	69,994
plural	1,521.62	145.62	259,635	26,482	286,117
pull-stream	2,631.99	1,391.93	194,441	73,754	268,195
q	4,695.78	12,866.72	2,086,666	127,918	2,214,584
spacl-core	1,350.86	706.03	157,479	28,159	185,638
zip-a-folder	500.51	1,109.02	80,546	10,227	90,773
Total	23,407.38	23,612.27	5,709,448	536,319	6,245,767

Table 115: Results from LLMorpheus experiment (run #375). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,053.84	617.90	948,398	75,377	1,023,775
countries-and-timezones	1,070.72	306.03	101,694	23,805	125,499
crawler-url-parser	1,656.36	838.63	379,359	31,102	410,461
delta	2,886.73	3,644.21	872,234	64,947	937,181
image-downloader	430.48	330.08	23,017	9,107	32,124
node-dirty	1,526.82	243.25	240,242	24,153	264,395
node-geo-point	1,410.93	999.44	310,873	26,143	337,016
node-jsonfile	690.54	480.47	54,864	15,130	69,994
plural	1,522.37	144.72	259,635	26,473	286,108
pull-stream	2,649.32	1,395.27	194,441	73,826	268,267
q	4,627.96	12,851.25	2,086,666	127,807	2,214,473
spacl-core	1,350.90	680.52	157,479	28,203	185,682
zip-a-folder	500.52	1,098.96	80,546	10,244	90,790
Total	23,377.51	23,630.74	5,709,448	536,317	6,245,765

Table 116: Results from LLMorpheus experiment (run #376). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,048.61	622.31	948,398	75,411	1,023,809
countries-and-timezones	1,070.74	302.32	101,694	23,740	125,434
crawler-url-parser	1,653.97	835.85	379,359	30,947	410,306
delta	2,880.16	3,648.84	872,234	65,086	937,320
image-downloader	430.49	326.72	23,017	9,110	32,127
node-dirty	1,526.71	230.04	240,242	24,142	264,384
node-geo-point	1,410.98	963.40	310,873	26,313	337,186
node-jsonfile	690.67	530.41	54,864	15,130	69,994
plural	1,522.04	148.93	259,635	26,465	286,100
pull-stream	2,630.34	1,397.17	194,441	73,763	268,204
q	4,627.86	12,869.01	2,086,666	127,790	2,214,456
spacl-core	1,350.91	714.31	157,479	28,174	185,653
zip-a-folder	500.51	1,073.84	80,546	10,244	90,790
Total	23,344.00	23,663.14	5,709,448	536,315	6,245,763

Table 117: Results from LLMorpheus experiment (run #377). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noexplanation.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

````
Your task is to apply mutation testing to the following code:
```
{{{code}}}
```

by replacing the PLACEHOLDER with a buggy code fragment that has different
behavior than the original code fragment, which was:
```
{{{orig}}}
```
Please consider changes such as using different operators, changing constants,
referring to different variables, object properties, functions, or methods.

Provide three answers as fenced code blocks containing a single line of code,
using the following template:

Option 1: The PLACEHOLDER can be replaced with:
```
<code fragment>
```

Option 2: The PLACEHOLDER can be replaced with:
```
<code fragment>
```

Option 3: The PLACEHOLDER can be replaced with:
```
<code fragment>
```

Please conclude your response with "DONE."
````

Figure 16: Variation on the template of Figure 7 that does not request explanations for the suggested mutations.

1.12 Results for `template-noinstructions-0.0`

Tables 118–127 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 17 and using the system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,453	235	36	45	1,137	696	440	1	61.30
countries-and-timezones	106	315	85	8	4	218	174	44	0	79.82
crawler-url-parser	176	518	211	23	20	246	134	112	0	54.47
delta	462	1,366	563	26	18	759	612	115	32	84.85
image-downloader	42	126	39	2	0	84	69	15	0	82.14
node-dirty	154	457	162	26	9	260	146	103	11	60.38
node-geo-point	140	413	85	7	13	306	230	76	0	75.16
node-jsonfile	68	200	42	9	1	148	45	51	52	65.54
plural	153	444	109	57	17	261	189	71	1	72.80
pull-stream	351	1,040	224	19	16	781	467	248	66	68.25
q	1,051	3,130	1,019	94	59	1,958	138	1,726	94	11.85
spacl-core	134	395	158	30	7	187	155	31	1	83.42
zip-a-folder	49	144	41	5	1	97	26	4	67	95.88
Total	3,376	10,001	2,973	342	210	6,442	3,081	3,036	325	—

Table 118: Results from LLMorpheus experiment (run #378). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,453	235	36	45	1,137	696	440	1	61.30
countries-and-timezones	106	315	85	8	4	218	174	44	0	79.82
crawler-url-parser	176	518	211	23	20	246	134	112	0	54.47
delta	462	1,366	563	26	18	759	610	117	32	84.58
image-downloader	42	126	39	2	0	84	70	14	0	83.33
node-dirty	154	457	162	26	9	260	146	103	11	60.38
node-geo-point	140	413	85	7	13	306	230	76	0	75.16
node-jsonfile	68	200	42	9	1	148	45	51	52	65.54
plural	153	444	109	57	17	261	189	71	1	72.80
pull-stream	351	1,040	224	19	16	781	467	248	66	68.25
q	1,051	3,130	1,017	93	60	1,960	137	1,728	95	11.84
spacl-core	134	395	157	30	7	188	156	31	1	83.51
zip-a-folder	49	144	41	5	1	97	26	4	67	95.88
Total	3,376	10,001	2,970	341	211	6,445	3,080	3,039	326	—

Table 119: Results from LLMorpheus experiment (run #379). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,453	235	36	45	1,137	696	440	1	61.30
countries-and-timezones	106	315	85	8	4	218	174	44	0	79.82
crawler-url-parser	176	518	211	23	20	246	134	112	0	54.47
delta	462	1,365	562	26	18	759	611	116	32	84.72
image-downloader	42	126	39	2	0	84	69	15	0	82.14
node-dirty	154	457	162	26	9	260	146	103	11	60.38
node-geo-point	140	412	84	7	13	306	230	76	0	75.16
node-jsonfile	68	200	42	9	1	148	45	51	52	65.54
plural	153	444	109	57	17	261	189	71	1	72.80
pull-stream	351	1,040	224	19	16	781	466	249	66	68.12
q	1,051	3,132	1,021	94	60	1,957	137	1,727	93	11.75
spacl-core	134	395	158	30	7	187	155	31	1	83.42
zip-a-folder	49	144	41	5	1	97	26	4	67	95.88
Total	3,376	10,001	2,973	342	211	6,441	3,078	3,039	324	—

Table 120: Results from LLMorpheus experiment (run #380). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,452	234	36	46	1,136	695	440	1	61.27
countries-and-timezones	106	315	85	8	4	218	174	44	0	79.82
crawler-url-parser	176	518	211	23	20	246	134	112	0	54.47
delta	462	1,366	563	26	18	759	612	115	32	84.85
image-downloader	42	126	39	2	0	84	70	14	0	83.33
node-dirty	154	457	162	26	9	260	147	102	11	60.77
node-geo-point	140	413	85	7	13	306	230	76	0	75.16
node-jsonfile	68	200	42	9	1	148	45	51	52	65.54
plural	153	445	109	57	17	262	189	72	1	72.52
pull-stream	351	1,040	226	19	16	779	464	249	66	68.04
q	1,051	3,131	1,019	96	60	1,956	138	1,725	93	11.81
spacl-core	134	395	158	30	7	187	155	31	1	83.42
zip-a-folder	49	144	41	5	1	97	26	4	67	95.88
Total	3,376	10,002	2,974	344	212	6,438	3,079	3,035	324	—

Table 121: Results from LLMorpheus experiment (run #381). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,453	235	36	45	1,137	696	440	1	61.30
countries-and-timezones	106	315	85	8	4	218	174	44	0	79.82
crawler-url-parser	176	518	211	23	20	246	134	112	0	54.47
delta	462	1,366	563	26	18	759	612	115	32	84.85
image-downloader	42	126	39	2	0	84	69	15	0	82.14
node-dirty	154	457	159	26	10	262	149	102	11	61.07
node-geo-point	140	413	85	7	13	306	230	76	0	75.16
node-jsonfile	68	200	42	9	1	148	45	51	52	65.54
plural	153	443	108	57	17	261	189	71	1	72.80
pull-stream	351	1,040	226	19	16	779	465	248	66	68.16
q	1,051	3,130	1,019	93	60	1,958	137	1,728	93	11.75
spacl-core	134	395	158	30	7	187	155	31	1	83.42
zip-a-folder	49	144	41	5	1	97	26	4	67	95.88
Total	3,376	10,000	2,971	341	212	6,442	3,081	3,037	324	—

Table 122: Results from LLMorpheus experiment (run #382). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,363.13	610.09	953,788	104,944	1,058,732
countries-and-timezones	1,070.70	333.60	102,860	23,506	126,366
crawler-url-parser	1,668.12	865.68	381,295	38,817	420,112
delta	3,242.91	4,060.50	877,316	99,522	976,838
image-downloader	430.47	361.71	23,479	8,905	32,384
node-dirty	1,530.58	228.43	241,936	33,033	274,969
node-geo-point	1,410.89	1,019.95	312,413	28,975	341,388
node-jsonfile	690.55	469.01	55,612	14,598	70,210
plural	1,523.05	141.16	261,318	34,491	295,809
pull-stream	2,608.43	1,433.25	198,302	74,144	272,446
q	5,802.92	13,526.38	2,098,227	218,277	2,316,504
spacl-core	1,350.79	626.46	158,953	29,519	188,472
zip-a-folder	500.51	1,087.03	81,085	10,694	91,779
Total	25,193.04	24,763.25	5,746,584	719,425	6,466,009

Table 123: Results from LLMorpheus experiment (run #378). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,324.82	606.13	953,788	104,909	1,058,697
countries-and-timezones	1,070.79	314.67	102,860	23,502	126,362
crawler-url-parser	1,673.87	789.35	381,295	38,816	420,111
delta	3,186.56	3,835.89	877,316	99,463	976,779
image-downloader	430.46	362.97	23,479	8,961	32,440
node-dirty	1,530.82	229.51	241,936	33,036	274,972
node-geo-point	1,410.92	1,035.06	312,413	28,975	341,388
node-jsonfile	691.16	471.51	55,612	14,598	70,210
plural	1,523.31	142.93	261,318	34,484	295,802
pull-stream	2,610.16	1,428.74	198,302	74,195	272,497
q	5,806.23	13,570.79	2,098,227	218,057	2,316,284
spacl-core	1,350.78	598.84	158,953	29,457	188,410
zip-a-folder	500.50	1,065.03	81,085	10,694	91,779
Total	25,110.38	24,451.42	5,746,584	719,147	6,465,731

Table 124: Results from LLMorpheus experiment (run #379). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,340.83	597.34	953,788	105,056	1,058,844
countries-and-timezones	1,070.75	323.54	102,860	23,502	126,362
crawler-url-parser	1,659.88	803.01	381,295	38,801	420,096
delta	3,211.07	3,830.91	877,316	99,521	976,837
image-downloader	430.47	362.10	23,479	8,905	32,384
node-dirty	1,530.67	228.98	241,936	33,033	274,969
node-geo-point	1,410.92	997.90	312,413	29,053	341,466
node-jsonfile	690.56	466.32	55,612	14,598	70,210
plural	1,522.82	141.80	261,318	34,484	295,802
pull-stream	2,609.79	1,444.02	198,302	74,220	272,522
q	6,945.19	13,543.90	2,098,227	218,309	2,316,536
spacl-core	1,350.87	605.57	158,953	29,527	188,480
zip-a-folder	500.50	1,075.42	81,085	10,694	91,779
Total	26,274.31	24,420.80	5,746,584	719,703	6,466,287

Table 125: Results from LLMorpheus experiment (run #380). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,492.66	629.00	953,788	104,886	1,058,674
countries-and-timezones	1,177.99	323.34	102,860	23,502	126,362
crawler-url-parser	1,751.37	786.06	381,295	38,800	420,095
delta	3,365.20	3,872.79	877,316	99,562	976,878
image-downloader	441.26	358.96	23,479	8,961	32,440
node-dirty	1,608.97	228.53	241,936	33,053	274,989
node-geo-point	1,492.54	1,035.93	312,413	28,984	341,397
node-jsonfile	717.49	465.56	55,612	14,548	70,160
plural	1,630.97	140.39	261,318	34,444	295,762
pull-stream	2,738.37	1,433.35	198,302	74,222	272,524
q	6,155.10	13,521.99	2,098,227	218,163	2,316,390
spacl-core	1,440.32	585.59	158,953	29,512	188,465
zip-a-folder	509.75	1,063.09	81,085	10,694	91,779
Total	26,522.01	24,444.59	5,746,584	719,331	6,465,915

Table 126: Results from LLMorpheus experiment (run #381). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,378.96	601.89	953,788	104,940	1,058,728
countries-and-timezones	1,080.75	325.94	102,860	23,502	126,362
crawler-url-parser	1,660.11	741.99	381,295	38,801	420,096
delta	3,233.82	3,810.62	877,316	99,525	976,841
image-downloader	430.46	361.68	23,479	8,905	32,384
node-dirty	1,531.35	234.89	241,936	33,044	274,980
node-geo-point	1,410.90	1,014.28	312,413	28,969	341,382
node-jsonfile	690.54	466.32	55,612	14,598	70,210
plural	1,523.06	136.58	261,318	34,492	295,810
pull-stream	2,606.22	1,436.07	198,302	74,135	272,437
q	5,921.20	13,597.41	2,098,227	218,214	2,316,441
spacl-core	1,350.81	585.65	158,953	29,520	188,473
zip-a-folder	500.48	1,078.08	81,085	10,745	91,830
Total	25,318.67	24,391.40	5,746,584	719,390	6,465,974

Table 127: Results from LLMorpheus experiment (run #382). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-noinstructions.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

````
Your task is to apply mutation testing to the following code:
```
{{{code}}}
```

by replacing the PLACEHOLDER with a buggy code fragment that has different
behavior than the original code fragment, which was:
```
{{{orig}}}
```

Provide three answers as fenced code blocks containing a single line of code,
using the following template:

Option 1: The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because <brief explanation>.

Option 2: The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because <brief explanation>.

Option 3: The PLACEHOLDER can be replaced with:
```
<code fragment>
```
This would result in different behavior because <brief explanation>.

Please conclude your response with "DONE."
````

Figure 17: Variation on the template of Figure 7 that does not provide instructions on how to create mutants.

1.13 Results for `template-full-genericsystemprompt-0.0`

Tables 128–137 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 7 and the generic system prompt of Figure 18.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,459	201	9	50	1,199	740	458	1	61.80
countries-and-timezones	106	316	90	0	9	217	191	26	0	88.02
crawler-url-parser	176	522	226	16	19	246	143	103	0	58.13
delta	462	1,373	553	7	23	790	659	99	32	87.47
image-downloader	42	123	34	1	0	88	72	16	0	81.82
node-dirty	154	450	153	12	8	277	162	104	11	62.45
node-geo-point	140	414	95	1	11	305	229	76	0	75.08
node-jsonfile	68	199	47	2	0	150	49	49	52	67.33
plural	153	442	104	45	21	272	209	62	1	77.21
pull-stream	351	1,039	252	15	9	763	442	266	55	65.14
q	1,051	3,128	1,037	30	54	2,007	145	1,770	92	11.81
spacl-core	134	397	146	12	8	214	181	32	1	85.05
zip-a-folder	49	145	40	1	2	101	27	3	71	97.03
Total	3,376	10,007	2,978	151	214	6,629	3,249	3,064	316	—

Table 128: Results from LLMorpheus experiment (run #384). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,459	201	9	50	1,199	740	458	1	61.80
countries-and-timezones	106	316	90	0	9	217	191	26	0	88.02
crawler-url-parser	176	522	226	16	19	246	143	103	0	58.13
delta	462	1,372	555	6	23	788	657	99	32	87.44
image-downloader	42	123	34	1	0	88	72	16	0	81.82
node-dirty	154	450	153	12	8	277	162	104	11	62.45
node-geo-point	140	414	95	1	11	305	229	76	0	75.08
node-jsonfile	68	200	47	2	0	151	49	49	53	67.55
plural	153	443	105	45	21	272	209	62	1	77.21
pull-stream	351	1,040	252	15	9	764	443	266	55	65.18
q	1,051	3,127	1,035	30	54	2,008	145	1,771	92	11.80
spacl-core	134	397	146	12	8	214	181	32	1	85.05
zip-a-folder	49	145	40	1	2	101	27	3	71	97.03
Total	3,376	10,008	2,979	150	214	6,630	3,248	3,065	317	—

Table 129: Results from LLMorpheus experiment (run #385). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,459	201	9	50	1,199	740	458	1	61.80
countries-and-timezones	106	316	90	0	9	217	191	26	0	88.02
crawler-url-parser	176	522	226	16	19	246	143	103	0	58.13
delta	462	1,373	551	6	24	792	660	100	32	87.37
image-downloader	42	123	34	1	0	88	72	16	0	81.82
node-dirty	154	450	153	12	8	277	162	104	11	62.45
node-geo-point	140	415	96	1	11	305	229	76	0	75.08
node-jsonfile	68	200	47	2	0	151	49	49	53	67.55
plural	153	442	106	45	21	270	207	62	1	77.04
pull-stream	351	1,039	252	15	9	763	440	268	55	64.88
q	1,051	3,127	1,035	30	55	2,007	144	1,771	92	11.76
spacl-core	134	396	146	12	8	216	183	32	1	85.19
zip-a-folder	49	145	40	1	2	101	27	3	71	97.03
Total	3,376	10,007	2,977	150	216	6,632	3,247	3,068	317	—

Table 130: Results from LLMorpheus experiment (run #386). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,459	201	9	49	1,200	741	458	1	61.83
countries-and-timezones	106	316	90	0	9	217	191	26	0	88.02
crawler-url-parser	176	522	226	16	19	246	144	102	0	58.54
delta	462	1,372	551	6	24	791	659	100	32	87.36
image-downloader	42	123	34	1	0	88	72	16	0	81.82
node-dirty	154	450	153	12	8	277	163	103	11	62.82
node-geo-point	140	414	95	1	11	305	229	76	0	75.08
node-jsonfile	68	200	47	2	0	151	49	49	53	67.55
plural	153	442	106	45	21	270	207	62	1	77.04
pull-stream	351	1,039	252	15	9	763	442	266	55	65.14
q	1,051	3,125	1,036	29	54	2,006	144	1,770	92	11.76
spacl-core	134	397	146	12	8	214	181	32	1	85.05
zip-a-folder	49	145	40	1	2	101	27	3	71	97.03
Total	3,376	10,004	2,977	149	214	6,629	3,249	3,063	317	—

Table 131: Results from LLMorpheus experiment (run #387). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	1,459	201	9	50	1,199	740	458	1	61.80
countries-and-timezones	106	292	81	0	9	202	179	23	0	88.61
crawler-url-parser	176	522	226	16	19	246	143	103	0	58.13
delta	462	1,373	551	6	24	792	660	100	32	87.37
image-downloader	42	123	34	1	0	88	72	16	0	81.82
node-dirty	154	450	153	12	8	277	162	104	11	62.45
node-geo-point	140	414	95	1	11	305	229	76	0	75.08
node-jsonfile	68	200	47	2	0	151	49	49	53	67.55
plural	153	442	104	45	21	272	209	62	1	77.21
pull-stream	351	1,039	252	15	9	763	442	266	55	65.14
q	1,051	3,127	1,039	30	53	2,005	146	1,768	91	11.82
spacl-core	134	397	145	12	8	215	182	32	1	85.12
zip-a-folder	49	145	40	1	2	101	27	3	71	97.03
Total	3,376	9,983	2,968	150	214	6,616	3,240	3,060	316	—

Table 132: Results from LLMorpheus experiment (run #388). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	3,198.91	631.03	943,498	97,397	1,040,895
countries-and-timezones	1,070.69	306.50	100,634	22,822	123,456
crawler-url-parser	1,666.72	773.61	377,599	38,968	416,567
delta	3,188.39	3,973.51	867,614	96,702	964,316
image-downloader	430.51	376.36	22,597	8,748	31,345
node-dirty	1,531.02	243.07	238,702	32,642	271,344
node-geo-point	1,410.94	993.80	309,473	28,703	338,176
node-jsonfile	690.59	470.19	54,184	13,966	68,150
plural	1,523.29	146.19	258,105	33,232	291,337
pull-stream	2,621.42	1,374.37	190,931	73,130	264,061
q	5,911.30	13,892.03	2,076,156	216,002	2,292,158
spacl-core	1,350.90	688.46	156,139	28,052	184,191
zip-a-folder	500.50	1,139.30	80,056	10,370	90,426
Total	25,095.20	25,008.45	5,675,688	700,734	6,376,422

Table 133: Results from LLMorpheus experiment (run #384). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*

```
You are a programming assistant. You are expected to be concise and precise and
avoid any unnecessary examples, tests, and verbosity.
```

Figure 18: Generic system prompt.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,198.28	628.16	943,498	97,360	1,040,858
countries-and-timezones	1,070.75	313.88	100,634	22,817	123,451
crawler-url-parser	1,667.00	785.35	377,599	38,968	416,567
delta	3,193.53	4,061.32	867,614	96,672	964,286
image-downloader	430.50	373.49	22,597	8,748	31,345
node-dirty	1,530.73	243.51	238,702	32,641	271,343
node-geo-point	1,411.00	1,029.33	309,473	28,703	338,176
node-jsonfile	690.57	473.38	54,184	13,976	68,160
plural	1,523.28	142.88	258,105	33,183	291,288
pull-stream	2,622.56	1,368.80	190,931	73,002	263,933
q	5,761.33	13,943.75	2,076,156	216,075	2,292,231
spacl-core	1,350.88	668.94	156,139	28,048	184,187
zip-a-folder	500.52	1,170.72	80,056	10,370	90,426
Total	24,950.92	25,203.51	5,675,688	700,563	6,376,251

Table 134: Results from LLMorpheus experiment (run #385). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,246.58	637.01	943,498	97,351	1,040,849
countries-and-timezones	1,070.74	317.35	100,634	22,822	123,456
crawler-url-parser	1,666.54	828.48	377,599	38,968	416,567
delta	3,141.81	3,972.35	867,614	96,648	964,262
image-downloader	430.53	375.91	22,597	8,740	31,337
node-dirty	1,531.64	237.14	238,702	32,632	271,334
node-geo-point	1,410.80	1,007.52	309,473	28,670	338,143
node-jsonfile	690.60	479.57	54,184	13,982	68,166
plural	1,523.32	141.62	258,105	33,221	291,326
pull-stream	2,590.59	1,371.20	190,931	73,097	264,028
q	5,912.17	13,970.49	2,076,156	216,015	2,292,171
spacl-core	1,461.05	690.80	156,139	28,074	184,213
zip-a-folder	500.51	1,128.17	80,056	10,370	90,426
Total	25,176.88	25,157.62	5,675,688	700,590	6,376,278

Table 135: Results from LLMorpheus experiment (run #386). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,229.79	627.53	943,498	97,331	1,040,829
countries-and-timezones	1,070.66	315.55	100,634	22,813	123,447
crawler-url-parser	1,677.85	830.94	377,599	39,015	416,614
delta	3,135.61	3,928.08	867,614	96,647	964,261
image-downloader	430.47	374.42	22,597	8,748	31,345
node-dirty	1,530.49	236.58	238,702	32,642	271,344
node-geo-point	1,410.83	1,012.70	309,473	28,703	338,176
node-jsonfile	690.54	473.16	54,184	13,999	68,183
plural	1,523.30	142.61	258,105	33,221	291,326
pull-stream	2,590.76	1,370.63	190,931	73,109	264,040
q	5,913.75	13,931.58	2,076,156	216,172	2,292,328
spacl-core	1,350.92	677.55	156,139	28,048	184,187
zip-a-folder	500.55	1,132.09	80,056	10,370	90,426
Total	25,055.51	25,053.41	5,675,688	700,818	6,376,506

Table 136: Results from LLMorpheus experiment (run #387). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	3,200.91	614.84	943,498	97,375	1,040,873
countries-and-timezones	1,501.26	297.33	95,530	21,102	116,632
crawler-url-parser	1,664.69	819.62	377,599	38,970	416,569
delta	3,159.72	4,099.01	867,614	96,624	964,238
image-downloader	430.49	373.69	22,597	8,748	31,345
node-dirty	1,555.91	239.75	238,702	32,632	271,334
node-geo-point	1,410.92	994.48	309,473	28,709	338,182
node-jsonfile	690.59	467.78	54,184	13,996	68,180
plural	1,523.33	146.45	258,105	33,232	291,337
pull-stream	2,596.08	1,354.67	190,931	73,098	264,029
q	5,912.23	13,931.75	2,076,156	216,129	2,292,285
spacl-core	1,350.85	689.90	156,139	28,037	184,176
zip-a-folder	530.56	1,147.59	80,056	10,370	90,426
Total	25,527.53	25,176.85	5,670,584	699,022	6,369,606

Table 137: Results from LLMorpheus experiment (run #388). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-full.hb*, systemPrompt: *SystemPrompt-Generic.txt*

1.14 Results for `template-basic-0.0`

Tables 138–147 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 19 and using the system prompt shown in Figure 7.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	212	92	1	185	120	65	0	64.86
countries-and-timezones	106	106	36	22	0	48	44	4	0	91.67
crawler-url-parser	176	176	75	27	1	67	49	18	0	73.13
delta	462	462	201	54	6	201	167	28	6	86.07
image-downloader	42	42	21	11	0	10	7	3	0	70.00
node-dirty	154	154	70	38	2	44	24	18	2	59.09
node-geo-point	140	140	39	33	6	62	54	8	0	87.10
node-jsonfile	68	68	17	28	1	22	11	3	8	86.36
plural	153	152	35	21	4	92	78	14	0	84.78
pull-stream	351	351	115	87	0	149	88	54	7	63.76
q	1,051	1,051	405	232	13	401	38	350	13	12.72
spacl-core	134	134	65	37	1	25	23	2	0	92.00
zip-a-folder	49	49	18	11	0	20	5	1	14	95.00
Total	3,376	3,375	1,309	693	35	1,326	708	568	50	—

Table 138: Results from LLMorpheus experiment (run #390). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	213	92	1	184	119	65	0	64.67
countries-and-timezones	106	106	36	22	0	48	43	5	0	89.58
crawler-url-parser	176	176	75	27	1	67	49	18	0	73.13
delta	462	462	200	54	6	202	168	28	6	86.14
image-downloader	42	42	21	11	0	10	7	3	0	70.00
node-dirty	154	154	70	38	2	44	24	18	2	59.09
node-geo-point	140	140	39	33	6	62	54	8	0	87.10
node-jsonfile	68	68	17	28	1	22	11	3	8	86.36
plural	153	152	35	21	4	92	78	14	0	84.78
pull-stream	351	351	115	87	0	149	88	54	7	63.76
q	1,051	1,051	405	232	13	401	38	350	13	12.72
spacl-core	134	134	65	37	1	25	23	2	0	92.00
zip-a-folder	49	49	18	11	0	20	5	1	14	95.00
Total	3,376	3,375	1,309	693	35	1,326	707	569	50	—

Table 139: Results from LLMorpheus experiment (run #391). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	213	92	1	184	119	65	0	64.67
countries-and-timezones	106	106	36	22	0	48	43	5	0	89.58
crawler-url-parser	176	176	75	27	1	67	49	18	0	73.13
delta	462	462	201	54	6	201	167	28	6	86.07
image-downloader	42	42	21	11	0	10	7	3	0	70.00
node-dirty	154	154	71	38	2	43	24	17	2	60.47
node-geo-point	140	140	39	33	6	62	54	8	0	87.10
node-jsonfile	68	68	17	28	1	22	11	3	8	86.36
plural	153	152	35	21	4	92	78	14	0	84.78
pull-stream	351	351	115	87	0	149	88	54	7	63.76
q	1,051	1,051	405	231	13	402	38	351	13	12.69
spacl-core	134	134	65	37	1	25	23	2	0	92.00
zip-a-folder	49	49	18	11	0	20	5	1	14	95.00
Total	3,376	3,375	1,311	692	35	1,325	706	569	50	—

Table 140: Results from LLMorpheus experiment (run #392). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	213	92	1	184	119	65	0	64.67
countries-and-timezones	106	106	36	22	0	48	43	5	0	89.58
crawler-url-parser	176	176	75	27	1	67	49	18	0	73.13
delta	462	462	201	54	6	201	167	28	6	86.07
image-downloader	42	42	21	11	0	10	7	3	0	70.00
node-dirty	154	154	70	38	2	44	24	18	2	59.09
node-geo-point	140	140	39	33	6	62	54	8	0	87.10
node-jsonfile	68	68	17	28	1	22	11	3	8	86.36
plural	153	150	35	20	4	91	78	13	0	85.71
pull-stream	351	351	115	87	0	149	88	54	7	63.76
q	1,051	1,051	405	231	13	402	38	351	13	12.69
spacl-core	134	134	65	37	1	25	23	2	0	92.00
zip-a-folder	49	49	18	11	0	20	5	1	14	95.00
Total	3,376	3,373	1,310	691	35	1,325	706	569	50	—

Table 141: Results from LLMorpheus experiment (run #393). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

application	#prompts	mutant candidates				#mutants	#killed	#survived	#timeout	mut. score
		total	invalid	identical	duplicate					
Complex.js	490	490	212	92	1	185	120	65	0	64.86
countries-and-timezones	106	106	36	22	0	48	44	4	0	91.67
crawler-url-parser	176	176	75	27	1	67	49	18	0	73.13
delta	462	462	202	54	6	200	166	28	6	86.00
image-downloader	42	42	21	11	0	10	7	3	0	70.00
node-dirty	154	154	71	38	2	43	24	17	2	60.47
node-geo-point	140	140	39	33	6	62	54	8	0	87.10
node-jsonfile	68	68	17	28	1	22	11	3	8	86.36
plural	153	152	35	21	4	92	78	14	0	84.78
pull-stream	351	351	115	87	0	149	88	54	7	63.76
q	1,051	1,051	404	232	13	402	38	351	13	12.69
spacl-core	134	134	65	37	1	25	23	2	0	92.00
zip-a-folder	49	49	18	11	0	20	5	1	14	95.00
Total	3,376	3,375	1,310	693	35	1,325	707	568	50	—

Table 142: Results from LLMorpheus experiment (run #394). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*.

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	prompt	compl.	total
Complex.js	2,731.54	97.72	893,966	14,460	908,426
countries-and-timezones	1,071.22	73.08	89,939	3,087	93,026
crawler-url-parser	1,636.67	215.52	359,498	5,557	365,055
delta	2,659.93	910.49	820,541	13,472	834,013
image-downloader	430.67	65.98	18,348	1,448	19,796
node-dirty	1,526.59	39.53	223,071	4,425	227,496
node-geo-point	1,411.43	204.76	295,321	4,217	299,538
node-jsonfile	690.87	77.82	47,346	1,831	49,177
plural	1,521.52	48.57	241,953	5,075	247,028
pull-stream	2,382.28	245.31	156,016	9,287	165,303
q	4,158.17	2,697.98	1,970,359	30,059	2,000,418
spacl-core	1,351.47	85.42	142,466	4,013	146,479
zip-a-folder	500.75	235.31	75,033	1,594	76,627
Total	22,073.11	4,997.50	5,333,857	98,525	5,432,382

Table 143: Results from LLMorpheus experiment (run #390). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,732.79	102.55	893,966	14,472	908,438
countries-and-timezones	1,071.27	72.88	89,939	3,113	93,052
crawler-url-parser	1,636.61	211.77	359,498	5,557	365,055
delta	2,667.43	897.69	820,541	13,458	833,999
image-downloader	430.64	65.54	18,348	1,448	19,796
node-dirty	1,526.57	38.93	223,071	4,422	227,493
node-geo-point	1,411.51	203.69	295,321	4,218	299,539
node-jsonfile	690.88	77.42	47,346	1,831	49,177
plural	1,521.47	47.92	241,953	5,075	247,028
pull-stream	2,382.21	245.17	156,016	9,288	165,304
q	4,159.06	2,700.88	1,970,359	30,059	2,000,418
spacl-core	1,351.31	85.36	142,466	4,007	146,473
zip-a-folder	500.73	227.68	75,033	1,594	76,627
Total	22,082.48	4,977.48	5,333,857	98,542	5,432,399

Table 144: Results from LLMorpheus experiment (run #391). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,730.84	97.59	893,966	14,461	908,427
countries-and-timezones	1,071.16	72.83	89,939	3,113	93,052
crawler-url-parser	1,636.68	220.83	359,498	5,576	365,074
delta	2,659.86	887.50	820,541	13,473	834,014
image-downloader	430.65	65.72	18,348	1,449	19,797
node-dirty	1,526.53	37.30	223,071	4,496	227,567
node-geo-point	1,411.50	195.42	295,321	4,230	299,551
node-jsonfile	690.89	77.90	47,346	1,831	49,177
plural	1,521.54	49.08	241,953	5,075	247,028
pull-stream	2,382.22	248.75	156,016	9,288	165,304
q	4,158.01	2,694.11	1,970,359	30,070	2,000,429
spacl-core	1,351.43	85.92	142,466	4,013	146,479
zip-a-folder	500.71	229.75	75,033	1,594	76,627
Total	22,072.01	4,962.71	5,333,857	98,669	5,432,526

Table 145: Results from LLMorpheus experiment (run #392). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,730.22	96.64	893,966	14,459	908,425
countries-and-timezones	1,071.13	74.16	89,939	3,112	93,051
crawler-url-parser	1,636.64	206.03	359,498	5,556	365,054
delta	2,659.88	897.06	820,541	13,471	834,012
image-downloader	430.66	65.63	18,348	1,449	19,797
node-dirty	1,526.57	38.82	223,071	4,425	227,496
node-geo-point	1,411.45	200.23	295,321	4,217	299,538
node-jsonfile	690.84	77.61	47,346	1,831	49,177
plural	1,556.64	47.70	238,779	5,029	243,808
pull-stream	2,382.19	247.28	156,016	9,288	165,304
q	4,156.62	2,695.05	1,970,359	30,071	2,000,430
spacl-core	1,351.45	84.84	142,466	4,008	146,474
zip-a-folder	500.75	235.51	75,033	1,594	76,627
Total	22,105.05	4,966.57	5,330,683	98,510	5,429,193

Table 146: Results from LLMorpheus experiment (run #393). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

project	time (sec)		#tokens		
	LLMorpheus	*StrykerJS*	**prompt**	**compl.**	**total**
Complex.js	2,730.17	96.67	893,966	14,460	908,426
countries-and-timezones	1,071.19	73.83	89,939	3,086	93,025
crawler-url-parser	1,636.65	205.98	359,498	5,577	365,075
delta	2,659.91	887.25	820,541	13,475	834,016
image-downloader	430.71	65.63	18,348	1,449	19,797
node-dirty	1,526.60	38.47	223,071	4,500	227,571
node-geo-point	1,411.44	197.24	295,321	4,217	299,538
node-jsonfile	690.88	78.16	47,346	1,831	49,177
plural	1,522.37	47.69	241,953	5,075	247,028
pull-stream	2,382.20	245.09	156,016	9,290	165,306
q	4,154.13	2,691.66	1,970,359	30,055	2,000,414
spacl-core	1,351.41	83.75	142,466	4,013	146,479
zip-a-folder	500.72	232.15	75,033	1,594	76,627
Total	22,068.39	4,943.55	5,333,857	98,622	5,432,479

Table 147: Results from LLMorpheus experiment (run #394). Model: *codellama-34b-instruct*, temperature: 0.0, maxTokens: 250, template: *template-basic.hb*, systemPrompt: *SystemPrompt-MutationTestingExpert.txt*

````
Consider the following code fragment:
```
{{{code}}}
```

Please provide a code fragment that PLACEHOLDER can be replaced with.

Provide your answer as a fenced code block containing a single line of code,
using the following template:

The PLACEHOLDER can be replaced with:
```
<code fragment>
```

Please conclude your response with "DONE."
````

Figure 19: A minimal template for requesting a replacement for the PLACEHOLDER.

1.15 Summary of variability of results for all models examined in the paper

For each configuration of *LLMorpheus*, this section provides a summary of the mutation results, showing the average of each value across all five trials, along with the standard deviation for that value.

- Model codellama-13b-instruct with template full at temperature 0.0: Table 148
- Model codellama-34b-instruct with template basic at temperature 0.0: Table 149
- Model codellama-34b-instruct with template full at temperature 0.0: Table 150
- Model codellama-34b-instruct with template full at temperature 0.25: Table 151
- Model codellama-34b-instruct with template full at temperature 0.5: Table 152
- Model codellama-34b-instruct with template full at temperature 1.0: Table 153
- Model codellama-34b-instruct with template full at temperature genericsystemprompt-0.0: Table 154
- Model codellama-34b-instruct with template noexplanation at temperature 0.0: Table 155
- Model codellama-34b-instruct with template noinstructions at temperature 0.0: Table 156
- Model codellama-34b-instruct with template onemutation at temperature 0.0: Table 157
- Model gpt-4o-mini with template full at temperature 0.0: Table 158
- Model llama-3.3-70b-instruct with template full at temperature 0.0: Table 159
- Model mixtral-8x7b-instruct with template full at temperature 0.0: Table 160

Table 148: Summary of mutants for codellama-13b-instruct-full-0.0 (runs 354, 355, 356, 358, 359). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,411 ± 0	340 ± 0	116 ± 0	28 ± 0	955 ± 0	553 ± 0	401 ± 0	1 ± 0	58.01 ± 0
countries-and-timezones	106 ± 0	305 ± 0	83 ± 0	15 ± 0	1 ± 0	207 ± 0	177 ± 0	30 ± 0	0 ± 0	85.51 ± 0
crawler-url-parser	176 ± 0	494 ± 0	186 ± 0	51 ± 0	12 ± 0	257 ± 0	128 ± 2	118 ± 0	1 ± 2	52.23 ± 0
delta	462 ± 0	1,334 ± 0	530 ± 0	92 ± 0	16 ± 0	712 ± 0	583 ± 0	107 ± 0	22 ± 0	84.97 ± 0
image-downloader	42 ± 0	122 ± 0	40 ± 0	5 ± 0	2 ± 0	77 ± 0	48 ± 0	29 ± 0	0 ± 0	62.34 ± 0
node-dirty	154 ± 0	439 ± 0	161 ± 0	33 ± 0	11 ± 0	245 ± 0	142 ± 0	92 ± 0	11 ± 0	62.45 ± 0
node-geo-point	140 ± 0	390 ± 0	64 ± 0	21 ± 0	16 ± 0	305 ± 0	237 ± 0	67 ± 0	0 ± 0	77.96 ± 0
node-jsonfile	68 ± 0	192 ± 0	45 ± 1	10 ± 0	6 ± 0	137 ± 0	43 ± 0	45 ± 0	49 ± 0	67.2 ± 0.11
plural	153 ± 0	404 ± 6	100 ± 1	98 ± 1	17 ± 0	206 ± 4	153 ± 3	53 ± 1	1 ± 0	74.52 ± 0.01
pull-stream	351 ± 0	1,002 ± 0	279 ± 0	54 ± 0	13 ± 0	669 ± 0	386 ± 1	236 ± 1	46 ± 0	64.65 ± 0.08
q	1,051 ± 0	2,992 ± 1	899 ± 1	379 ± 0	55 ± 0	1,714 ± 1	122 ± 0	1,519 ± 1	73 ± 0	11.38 ± 0
spacl-core	134 ± 0	377 ± 0	142 ± 0	40 ± 0	7 ± 0	195 ± 0	160 ± 0	25 ± 0	0 ± 0	86.38 ± 0.24
zip-a-folder	49 ± 0	137 ± 0	43 ± 0	7 ± 0	1 ± 0	87 ± 0	27 ± 0	55 ± 0	5 ± 0	36.78 ± 0
Total	3,376	9,599	2,911	921	185	5,766	2,759	2,777	209	63.41

Table 149: Summary of mutants for codellama-34b-instruct-basic-0.0 (runs 390, 391, 392, 393, 394). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	489 ± 0	213 ± 1	92 ± 0	1 ± 0	184 ± 1	119 ± 1	65 ± 0	0 ± 0	64.75 ± 0.1
countries-and-timezones	106 ± 0	106 ± 0	36 ± 0	22 ± 0	0 ± 0	48 ± 0	43 ± 1	5 ± 1	0 ± 0	90.42 ± 1.14
crawler-url-parser	176 ± 0	175 ± 0	75 ± 0	27 ± 0	1 ± 0	73 ± 0	49 ± 0	18 ± 0	0 ± 0	73.13 ± 0
delta	462 ± 0	456 ± 0	201 ± 1	54 ± 0	6 ± 0	201 ± 1	167 ± 1	28 ± 0	6 ± 0	86.07 ± 0.05
image-downloader	42 ± 0	42 ± 0	21 ± 0	11 ± 0	0 ± 0	10 ± 0	7 ± 0	3 ± 0	0 ± 0	70 ± 0
node-dirty	154 ± 0	152 ± 0	70 ± 1	38 ± 0	2 ± 0	44 ± 1	24 ± 0	18 ± 1	2 ± 0	59.64 ± 0.76
node-geo-point	140 ± 0	134 ± 0	39 ± 0	33 ± 0	6 ± 0	62 ± 0	54 ± 0	8 ± 0	0 ± 0	87.1 ± 0
node-jsonfile	68 ± 0	67 ± 0	17 ± 0	28 ± 0	1 ± 0	22 ± 0	11 ± 0	3 ± 0	8 ± 0	86.36 ± 0
plural	153 ± 0	148 ± 1	35 ± 0	21 ± 0	4 ± 0	92 ± 0	78 ± 0	14 ± 0	0 ± 0	84.97 ± 0.42
pull-stream	351 ± 0	351 ± 0	115 ± 0	87 ± 0	0 ± 0	149 ± 0	88 ± 0	54 ± 0	7 ± 0	63.76 ± 0
q	1,051 ± 0	1,038 ± 0	405 ± 0	232 ± 1	13 ± 0	402 ± 1	38 ± 0	351 ± 1	13 ± 0	12.7 ± 0.02
spacl-core	134 ± 0	133 ± 0	65 ± 0	37 ± 0	1 ± 0	31 ± 0	23 ± 0	2 ± 0	0 ± 0	92 ± 0
zip-a-folder	49 ± 0	49 ± 0	18 ± 0	11 ± 0	0 ± 0	20 ± 0	5 ± 0	1 ± 0	14 ± 0	95 ± 0
Total	3,376	3,340	1,310	692	35	1,337	707	569	50	74.3

Table 150: Summary of mutants for codellama-34b-instruct-full-0.0 (runs 312, 314, 315, 316, 317). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,406 ± 1	194 ± 1	13 ± 0	45 ± 0	1,199 ± 0	725 ± 1	473 ± 1	1 ± 0	60.58 ± 0.08
countries-and-timezones	106 ± 0	306 ± 0	89 ± 0	0 ± 0	12 ± 0	217 ± 0	188 ± 0	29 ± 0	0 ± 0	86.63 ± 0.03
crawler-url-parser	176 ± 0	504 ± 0	207 ± 2	14 ± 0	17 ± 0	284 ± 1	157 ± 0	127 ± 1	0 ± 0	55.36 ± 0.31
delta	462 ± 0	1,342 ± 1	565 ± 0	10 ± 0	25 ± 1	767 ± 1	635 ± 1	100 ± 0	32 ± 0	86.93 ± 0.06
image-downloader	42 ± 0	124 ± 0	34 ± 1	1 ± 1	0 ± 0	89 ± 1	72 ± 1	17 ± 0	0 ± 0	80.98 ± 0.12
node-dirty	154 ± 0	443 ± 0	153 ± 1	15 ± 0	7 ± 0	275 ± 1	161 ± 1	101 ± 1	12 ± 0	63.07 ± 0.34
node-geo-point	140 ± 0	396 ± 1	94 ± 1	0 ± 0	13 ± 0	302 ± 0	223 ± 0	79 ± 0	0 ± 0	73.77 ± 0.15
node-jsonfile	68 ± 0	199 ± 0	42 ± 1	3 ± 0	0 ± 0	154 ± 0	49 ± 0	48 ± 0	57 ± 0	68.92 ± 0.2
plural	153 ± 0	424 ± 1	101 ± 0	42 ± 1	18 ± 0	280 ± 1	204 ± 1	75 ± 0	1 ± 0	73.23 ± 0.08
pull-stream	351 ± 0	1,019 ± 0	237 ± 1	12 ± 0	9 ± 0	770 ± 1	442 ± 1	271 ± 2	57 ± 0	64.75 ± 0.23
q	1,051 ± 0	3,068 ± 1	1,002 ± 2	34 ± 0	53 ± 1	2,033 ± 2	159 ± 1	1,790 ± 2	84 ± 1	11.93 ± 0.03
spacl-core	134 ± 0	387 ± 1	138 ± 1	10 ± 0	7 ± 0	239 ± 0	198 ± 1	40 ± 1	1 ± 0	83.17 ± 0.34
zip-a-folder	49 ± 0	142 ± 0	41 ± 0	1 ± 0	1 ± 0	100 ± 0	23 ± 0	3 ± 0	74 ± 0	97 ± 0
Total	3,376	9,760	2,898	155	207	6,707	3,235	3,154	319	69.72

Table 151: Summary of mutants for codellama-34b-instruct-full-0.25 (runs 348, 350, 351, 352, 353). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,413 ± 7	212 ± 12	10 ± 2	43 ± 4	1,190 ± 13	727 ± 8	463 ± 9	0 ± 1	61.09 ± 0.5
countries-and-timezones	106 ± 0	308 ± 5	85 ± 3	1 ± 1	7 ± 4	222 ± 5	193 ± 8	28 ± 5	0 ± 0	87.18 ± 2.55
crawler-url-parser	176 ± 0	501 ± 4	210 ± 5	13 ± 4	15 ± 4	278 ± 8	157 ± 9	104 ± 6	0 ± 0	60.25 ± 2.47
delta	462 ± 0	1,341 ± 6	566 ± 13	8 ± 2	25 ± 3	767 ± 12	627 ± 17	107 ± 4	32 ± 4	86.01 ± 0.66
image-downloader	42 ± 0	123 ± 2	36 ± 2	1 ± 2	1 ± 1	86 ± 2	68 ± 4	17 ± 2	0 ± 0	80.43 ± 3.22
node-dirty	154 ± 0	445 ± 5	160 ± 4	14 ± 3	10 ± 4	271 ± 6	165 ± 6	94 ± 5	12 ± 0	65.19 ± 1.82
node-geo-point	140 ± 0	394 ± 12	90 ± 10	1 ± 1	12 ± 4	303 ± 8	226 ± 10	74 ± 6	0 ± 0	75.36 ± 2.01
node-jsonfile	68 ± 0	201 ± 2	44 ± 3	3 ± 1	1 ± 1	154 ± 4	52 ± 3	47 ± 5	56 ± 2	69.72 ± 2.82
plural	153 ± 0	419 ± 2	101 ± 2	36 ± 5	20 ± 2	282 ± 7	210 ± 4	71 ± 7	1 ± 1	75.03 ± 2.12
pull-stream	351 ± 0	1,023 ± 1	239 ± 3	9 ± 2	7 ± 2	774 ± 3	447 ± 4	274 ± 6	54 ± 3	64.67 ± 0.8
q	1,051 ± 0	3,073 ± 8	1,023 ± 14	27 ± 11	51 ± 8	2,023 ± 31	147 ± 5	1,795 ± 27	82 ± 4	11.3 ± 0.33
spacl-core	134 ± 0	389 ± 3	139 ± 5	11 ± 2	6 ± 3	240 ± 6	189 ± 4	33 ± 3	1 ± 1	85.05 ± 1.23
zip-a-folder	49 ± 0	140 ± 2	39 ± 4	1 ± 1	1 ± 0	101 ± 4	32 ± 10	22 ± 27	46 ± 36	77.41 ± 26.94
Total	3,376	9,771	2,943	136	200	6,692	3,242	3,129	283	69.13

Table 152: Summary of mutants for codellama-34b-instruct-full-0.5 (runs 318, 319, 320, 321, 322). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,417 ± 7	226 ± 16	8 ± 1	37 ± 3	1,183 ± 10	725 ± 10	458 ± 10	0 ± 0	61.27 ± 0.72
countries-and-timezones	106 ± 0	305 ± 6	79 ± 6	4 ± 2	7 ± 3	222 ± 2	195 ± 3	28 ± 3	0 ± 0	87.59 ± 1.26
crawler-url-parser	176 ± 0	500 ± 3	193 ± 12	13 ± 4	15 ± 2	294 ± 12	167 ± 6	121 ± 11	5 ± 11	58.65 ± 3.1
delta	462 ± 0	1,340 ± 5	553 ± 12	10 ± 2	22 ± 5	777 ± 14	645 ± 19	97 ± 7	36 ± 2	87.56 ± 1.12
image-downloader	42 ± 0	122 ± 2	36 ± 3	2 ± 1	3 ± 2	84 ± 3	64 ± 6	21 ± 6	0 ± 0	75.57 ± 7.56
node-dirty	154 ± 0	446 ± 3	158 ± 7	9 ± 2	9 ± 4	279 ± 7	162 ± 8	105 ± 3	12 ± 0	62.23 ± 1.48
node-geo-point	140 ± 0	390 ± 18	89 ± 6	2 ± 1	13 ± 3	299 ± 17	228 ± 10	72 ± 10	0 ± 0	76.16 ± 2.27
node-jsonfile	68 ± 0	200 ± 2	42 ± 3	3 ± 1	1 ± 2	155 ± 5	53 ± 3	40 ± 3	62 ± 4	74.43 ± 2.05
plural	153 ± 0	415 ± 7	98 ± 7	27 ± 4	19 ± 5	290 ± 4	220 ± 9	70 ± 8	1 ± 1	75.89 ± 2.87
pull-stream	351 ± 0	1,023 ± 7	219 ± 10	6 ± 3	10 ± 4	798 ± 12	472 ± 11	270 ± 9	56 ± 4	66.17 ± 1.28
q	1,051 ± 0	3,071 ± 6	1,000 ± 17	29 ± 5	52 ± 8	2,042 ± 22	153 ± 11	1,804 ± 27	85 ± 8	11.67 ± 0.71
spacl-core	134 ± 0	388 ± 4	125 ± 7	7 ± 3	5 ± 2	256 ± 5	217 ± 6	39 ± 3	1 ± 0	84.93 ± 1.01
zip-a-folder	49 ± 0	138 ± 9	35 ± 5	1 ± 1	1 ± 1	101 ± 10	36 ± 10	19 ± 21	46 ± 37	79.77 ± 22.09
Total	3,376	9,755	2,853	121	194	6,782	3,336	3,143	303	69.38

Table 153: Summary of mutants for codellama-34b-instruct-full-1.0 (runs 341, 342, 343, 345, 347). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,359 ± 13	338 ± 20	6 ± 2	24 ± 2	1,015 ± 22	641 ± 20	374 ± 6	0 ± 1	63.12 ± 0.83
countries-and-timezones	106 ± 0	284 ± 20	91 ± 11	2 ± 3	3 ± 1	191 ± 17	168 ± 16	24 ± 6	0 ± 0	87.55 ± 2.79
crawler-url-parser	176 ± 0	489 ± 7	218 ± 13	5 ± 1	6 ± 2	266 ± 12	186 ± 14	80 ± 9	0 ± 0	69.79 ± 3.43
delta	462 ± 0	1,274 ± 28	556 ± 41	8 ± 3	19 ± 4	710 ± 25	599 ± 28	79 ± 6	32 ± 1	88.79 ± 1.09
image-downloader	42 ± 0	116 ± 2	42 ± 6	1 ± 1	2 ± 3	72 ± 8	55 ± 9	17 ± 7	0 ± 1	76.21 ± 9.17
node-dirty	154 ± 0	427 ± 10	164 ± 15	5 ± 3	5 ± 3	257 ± 11	150 ± 11	96 ± 7	11 ± 3	62.68 ± 2.25
node-geo-point	140 ± 0	392 ± 3	125 ± 5	3 ± 1	9 ± 2	265 ± 6	204 ± 8	61 ± 5	0 ± 0	76.95 ± 1.81
node-jsonfile	68 ± 0	190 ± 4	45 ± 4	2 ± 1	3 ± 2	142 ± 7	60 ± 8	20 ± 5	62 ± 4	85.71 ± 3.84
plural	153 ± 0	420 ± 12	110 ± 9	9 ± 2	13 ± 2	302 ± 5	239 ± 7	61 ± 5	1 ± 1	79.78 ± 1.62
pull-stream	351 ± 0	999 ± 6	245 ± 7	10 ± 2	6 ± 3	743 ± 5	455 ± 7	239 ± 5	49 ± 6	67.82 ± 0.77
q	1,051 ± 0	2,993 ± 11	1,078 ± 30	18 ± 3	35 ± 8	1,896 ± 28	145 ± 16	1,668 ± 23	83 ± 7	12.02 ± 0.47
spacl-core	134 ± 0	372 ± 3	146 ± 12	4 ± 3	5 ± 1	222 ± 13	194 ± 19	28 ± 8	0 ± 0	87.36 ± 4.02
zip-a-folder	49 ± 0	138 ± 5	43 ± 2	1 ± 1	1 ± 1	94 ± 7	34 ± 12	10 ± 16	50 ± 27	89.46 ± 16.3
Total	3,376	9,454	3,201	73	131	6,179	3,130	2,759	290	72.86

Table 154: Summary of mutants for codellama-34b-instruct-full-genericsystemprompt-0.0 (runs 384, 385, 386, 387, 388). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,409 ± 0	201 ± 0	9 ± 0	50 ± 0	1,199 ± 0	740 ± 0	458 ± 0	1 ± 0	61.81 ± 0.01
countries-and-timezones	106 ± 0	302 ± 11	88 ± 4	0 ± 0	9 ± 0	214 ± 7	189 ± 5	25 ± 1	0 ± 0	88.14 ± 0.26
crawler-url-parser	176 ± 0	503 ± 0	226 ± 0	16 ± 0	19 ± 0	261 ± 0	143 ± 0	103 ± 0	0 ± 0	58.21 ± 0.18
delta	462 ± 0	1,349 ± 1	552 ± 2	6 ± 0	24 ± 1	791 ± 2	659 ± 1	100 ± 1	32 ± 0	87.4 ± 0.05
image-downloader	42 ± 0	123 ± 0	34 ± 0	1 ± 0	0 ± 0	88 ± 0	72 ± 0	16 ± 0	0 ± 0	81.82 ± 0
node-dirty	154 ± 0	442 ± 0	153 ± 0	12 ± 0	8 ± 0	277 ± 0	162 ± 0	104 ± 0	11 ± 0	62.52 ± 0.17
node-geo-point	140 ± 0	403 ± 0	95 ± 0	1 ± 0	11 ± 0	307 ± 0	229 ± 0	76 ± 0	0 ± 0	75.08 ± 0
node-jsonfile	68 ± 0	200 ± 0	47 ± 0	2 ± 0	0 ± 0	151 ± 0	49 ± 0	49 ± 0	53 ± 0	67.51 ± 0.1
plural	153 ± 0	421 ± 0	105 ± 1	45 ± 0	21 ± 0	271 ± 1	208 ± 1	62 ± 0	1 ± 0	77.14 ± 0.09
pull-stream	351 ± 0	1,030 ± 0	252 ± 0	15 ± 0	9 ± 0	763 ± 0	442 ± 1	266 ± 1	55 ± 0	65.1 ± 0.12
q	1,051 ± 0	3,073 ± 1	1,036 ± 2	30 ± 0	54 ± 1	2,007 ± 1	145 ± 1	1,770 ± 1	92 ± 0	11.79 ± 0.03
spacl-core	134 ± 0	389 ± 0	146 ± 0	12 ± 0	8 ± 0	231 ± 1	182 ± 1	32 ± 0	1 ± 0	85.09 ± 0.06
zip-a-folder	49 ± 0	143 ± 0	40 ± 0	1 ± 0	2 ± 0	102 ± 0	27 ± 0	3 ± 0	71 ± 0	97.03 ± 0
Total	3,376	9,787	2,976	150	214	6,662	3,247	3,064	317	70.66

Table 155: Summary of mutants for codellama-34b-instruct-noexplanation-0.0 (runs 372, 374, 375, 376, 377). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,428 ± 0	277 ± 2	27 ± 0	37 ± 0	1,124 ± 2	677 ± 2	446 ± 1	1 ± 0	60.3 ± 0.1
countries-and-timezones	106 ± 0	297 ± 0	84 ± 0	2 ± 0	10 ± 0	211 ± 0	183 ± 0	28 ± 0	0 ± 0	86.73 ± 0
crawler-url-parser	176 ± 0	495 ± 0	215 ± 1	24 ± 1	19 ± 0	256 ± 0	140 ± 0	99 ± 0	0 ± 0	58.58 ± 0
delta	462 ± 0	1,356 ± 0	595 ± 2	28 ± 2	19 ± 0	733 ± 0	597 ± 1	109 ± 1	26 ± 1	85.08 ± 0.12
image-downloader	42 ± 0	124 ± 0	41 ± 0	4 ± 0	2 ± 0	79 ± 0	62 ± 0	15 ± 0	0 ± 0	80.52 ± 0
node-dirty	154 ± 0	448 ± 0	160 ± 0	30 ± 0	10 ± 0	258 ± 0	146 ± 0	99 ± 0	13 ± 0	61.63 ± 0
node-geo-point	140 ± 0	405 ± 0	104 ± 1	2 ± 0	9 ± 0	299 ± 1	215 ± 1	82 ± 0	0 ± 0	72.47 ± 0.19
node-jsonfile	68 ± 0	204 ± 0	48 ± 0	4 ± 0	0 ± 0	152 ± 0	54 ± 0	45 ± 0	53 ± 0	70.39 ± 0
plural	153 ± 0	433 ± 0	110 ± 0	50 ± 0	16 ± 0	273 ± 0	198 ± 0	74 ± 0	1 ± 0	72.89 ± 0
pull-stream	351 ± 0	1,026 ± 1	237 ± 1	16 ± 0	11 ± 1	774 ± 0	439 ± 1	279 ± 1	56 ± 0	63.99 ± 0.13
q	1,051 ± 0	3,084 ± 1	1,107 ± 1	122 ± 1	59 ± 1	1,855 ± 1	137 ± 1	1,634 ± 1	83 ± 1	11.9 ± 0.02
spacl-core	134 ± 0	390 ± 0	144 ± 1	18 ± 0	6 ± 0	228 ± 1	175 ± 1	33 ± 1	1 ± 0	84.24 ± 0.62
zip-a-folder	49 ± 0	142 ± 0	41 ± 0	3 ± 0	1 ± 0	98 ± 0	27 ± 1	3 ± 0	68 ± 1	96.94 ± 0
Total	3,376	9,832	3,162	329	199	6,341	3,051	2,946	303	69.67

Table 156: Summary of mutants for codellama-34b-instruct-noinstructions-0.0 (runs 378, 379, 380, 381, 382). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,408 ± 1	235 ± 0	36 ± 0	45 ± 0	1,137 ± 0	696 ± 0	440 ± 0	1 ± 0	61.29 ± 0.01
countries-and-timezones	106 ± 0	311 ± 0	85 ± 0	8 ± 0	4 ± 0	218 ± 0	174 ± 0	44 ± 0	0 ± 0	79.82 ± 0
crawler-url-parser	176 ± 0	498 ± 0	211 ± 0	23 ± 0	20 ± 0	264 ± 0	134 ± 0	112 ± 0	0 ± 0	54.47 ± 0
delta	462 ± 0	1,348 ± 0	563 ± 0	26 ± 0	18 ± 0	759 ± 0	611 ± 1	116 ± 1	32 ± 0	84.77 ± 0.12
image-downloader	42 ± 0	126 ± 0	39 ± 0	2 ± 0	0 ± 0	85 ± 0	69 ± 1	15 ± 1	0 ± 0	82.62 ± 0.65
node-dirty	154 ± 0	448 ± 0	161 ± 1	26 ± 0	9 ± 0	260 ± 1	147 ± 1	103 ± 1	11 ± 0	60.6 ± 0.31
node-geo-point	140 ± 0	400 ± 0	85 ± 0	7 ± 0	13 ± 0	308 ± 0	230 ± 0	76 ± 0	0 ± 0	75.16 ± 0
node-jsonfile	68 ± 0	199 ± 0	42 ± 0	9 ± 0	1 ± 0	148 ± 0	45 ± 0	51 ± 0	52 ± 0	65.54 ± 0
plural	153 ± 0	427 ± 1	109 ± 0	57 ± 0	17 ± 0	261 ± 0	189 ± 0	71 ± 0	1 ± 0	72.74 ± 0.13
pull-stream	351 ± 0	1,024 ± 0	225 ± 1	19 ± 0	16 ± 0	780 ± 1	466 ± 1	248 ± 1	66 ± 0	68.16 ± 0.09
q	1,051 ± 0	3,071 ± 1	1,019 ± 1	94 ± 1	60 ± 0	1,958 ± 1	137 ± 1	1,727 ± 1	94 ± 1	11.8 ± 0.05
spacl-core	134 ± 0	388 ± 0	158 ± 0	30 ± 0	7 ± 0	200 ± 0	155 ± 0	31 ± 0	1 ± 0	83.44 ± 0.04
zip-a-folder	49 ± 0	143 ± 0	41 ± 0	5 ± 0	1 ± 0	97 ± 0	26 ± 0	4 ± 0	67 ± 0	95.88 ± 0
Total	3,376	9,790	2,972	342	211	6,476	3,080	3,037	325	68.95

Table 157: Summary of mutants for codellama-34b-instruct-onemutation-0.0 (runs 365, 366, 369, 370, 371). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	478 ± 2	67 ± 0	6 ± 0	11 ± 0	405 ± 2	244 ± 2	161 ± 1	0 ± 0	60.24 ± 0.13
countries-and-timezones	106 ± 0	104 ± 0	24 ± 0	1 ± 0	2 ± 0	79 ± 0	66 ± 0	13 ± 0	0 ± 0	83.29 ± 0.56
crawler-url-parser	176 ± 0	170 ± 0	70 ± 1	8 ± 0	5 ± 0	92 ± 1	50 ± 1	36 ± 0	0 ± 0	58.47 ± 0.31
delta	462 ± 0	452 ± 0	182 ± 1	4 ± 0	9 ± 0	266 ± 1	221 ± 1	37 ± 0	8 ± 0	86.11 ± 0.03
image-downloader	42 ± 0	42 ± 0	8 ± 0	0 ± 0	0 ± 0	34 ± 0	26 ± 0	8 ± 0	0 ± 0	76.47 ± 0
node-dirty	154 ± 0	152 ± 0	50 ± 0	3 ± 1	3 ± 0	99 ± 1	55 ± 0	41 ± 1	3 ± 0	58.83 ± 0.32
node-geo-point	140 ± 0	137 ± 0	32 ± 0	0 ± 0	3 ± 0	105 ± 0	74 ± 0	30 ± 0	0 ± 0	71.15 ± 0
node-jsonfile	68 ± 0	68 ± 0	11 ± 0	0 ± 0	0 ± 0	57 ± 0	18 ± 0	18 ± 0	21 ± 0	68.42 ± 0
plural	153 ± 0	147 ± 0	39 ± 0	7 ± 1	6 ± 0	101 ± 1	71 ± 1	30 ± 0	0 ± 0	70.18 ± 0.16
pull-stream	351 ± 0	350 ± 0	67 ± 0	3 ± 0	1 ± 0	280 ± 0	165 ± 0	95 ± 0	20 ± 0	66 ± 0.16
q	1,051 ± 0	1,035 ± 0	306 ± 0	26 ± 0	17 ± 0	703 ± 0	46 ± 0	630 ± 1	27 ± 0	10.35 ± 0.07
spacl-core	134 ± 0	132 ± 0	41 ± 0	3 ± 0	2 ± 0	88 ± 0	63 ± 0	17 ± 0	0 ± 0	78.75 ± 0
zip-a-folder	49 ± 0	48 ± 0	9 ± 0	0 ± 0	1 ± 0	39 ± 0	19 ± 0	17 ± 0	3 ± 0	56.41 ± 0
Total	3,376	3,315	905	62	60	2,348	1,118	1,133	82	64.97

Table 158: Summary of mutants for gpt-4o-mini-full-0.0 (runs 58, 59, 60, 61, 63). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,433 ± 3	448 ± 6	0 ± 0	36 ± 2	986 ± 4	599 ± 4	387 ± 2	0 ± 0	60.72 ± 0.25
countries-and-timezones	106 ± 0	308 ± 2	101 ± 3	0 ± 0	10 ± 2	207 ± 2	171 ± 3	36 ± 2	0 ± 0	82.64 ± 0.98
crawler-url-parser	176 ± 0	517 ± 1	234 ± 4	0 ± 0	10 ± 1	283 ± 4	178 ± 3	91 ± 2	0 ± 0	66.1 ± 0.48
delta	462 ± 0	1,346 ± 3	643 ± 5	1 ± 1	39 ± 3	701 ± 4	562 ± 3	103 ± 4	35 ± 1	85.25 ± 0.52
image-downloader	42 ± 0	123 ± 1	57 ± 1	0 ± 0	3 ± 1	67 ± 1	45 ± 3	22 ± 3	0 ± 0	67.26 ± 3.9
node-dirty	154 ± 0	453 ± 1	191 ± 4	0 ± 0	9 ± 1	262 ± 4	152 ± 4	102 ± 3	7 ± 1	61.04 ± 1.08
node-geo-point	140 ± 0	395 ± 3	86 ± 2	0 ± 0	24 ± 3	310 ± 3	219 ± 4	88 ± 2	0 ± 0	71.33 ± 0.71
node-jsonfile	68 ± 0	197 ± 1	45 ± 1	0 ± 0	7 ± 1	153 ± 2	65 ± 3	26 ± 2	62 ± 2	83.09 ± 1.74
plural	153 ± 0	430 ± 2	112 ± 2	5 ± 1	25 ± 2	313 ± 1	258 ± 4	54 ± 3	1 ± 1	82.61 ± 0.84
pull-stream	351 ± 0	1,039 ± 2	300 ± 1	0 ± 0	14 ± 2	739 ± 3	430 ± 6	244 ± 3	65 ± 3	66.96 ± 0.43
q	1,051 ± 0	3,080 ± 5	1,292 ± 5	5 ± 2	73 ± 5	1,784 ± 9	130 ± 5	1,591 ± 7	63 ± 3	10.82 ± 0.32
spacl-core	134 ± 0	390 ± 2	157 ± 3	0 ± 0	11 ± 1	233 ± 3	198 ± 3	18 ± 2	0 ± 0	91.66 ± 0.86
zip-a-folder	49 ± 0	144 ± 1	62 ± 2	0 ± 0	3 ± 1	82 ± 1	18 ± 3	5 ± 1	59 ± 2	94.37 ± 1.68
Total	3,376	9,857	3,727	11	263	6,120	3,026	2,768	292	71.06

Table 159: Summary of mutants for llama-3.3-70b-instruct-full-0.0 (runs 23, 24, 25, 26, 27). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,413 ± 3	282 ± 7	0 ± 0	59 ± 5	1,130 ± 6	682 ± 5	448 ± 3	1 ± 0	60.39 ± 0.19
countries-and-timezones	106 ± 0	305 ± 1	65 ± 2	0 ± 0	13 ± 1	240 ± 2	204 ± 4	36 ± 3	0 ± 0	84.86 ± 1.34
crawler-url-parser	176 ± 0	506 ± 1	179 ± 3	0 ± 0	22 ± 1	328 ± 2	203 ± 3	113 ± 2	0 ± 0	64.18 ± 0.65
delta	462 ± 0	1,338 ± 3	540 ± 3	0 ± 0	49 ± 3	798 ± 4	634 ± 8	128 ± 5	37 ± 3	84.01 ± 0.69
image-downloader	42 ± 0	122 ± 1	44 ± 2	0 ± 0	5 ± 1	79 ± 2	52 ± 6	27 ± 4	0 ± 0	65.52 ± 6.21
node-dirty	154 ± 0	453 ± 2	124 ± 3	1 ± 0	11 ± 2	327 ± 5	167 ± 7	142 ± 2	19 ± 2	56.68 ± 1.07
node-geo-point	140 ± 0	400 ± 4	39 ± 2	0 ± 0	21 ± 3	361 ± 4	258 ± 4	100 ± 5	0 ± 0	71.96 ± 1.09
node-jsonfile	68 ± 0	199 ± 1	27 ± 2	0 ± 0	5 ± 1	172 ± 2	64 ± 2	38 ± 2	70 ± 2	77.81 ± 1.31
plural	153 ± 0	429 ± 2	92 ± 3	0 ± 0	27 ± 2	337 ± 3	245 ± 4	92 ± 5	0 ± 0	72.79 ± 1.33
pull-stream	351 ± 0	1,044 ± 2	266 ± 3	0 ± 0	9 ± 2	777 ± 3	465 ± 2	262 ± 4	51 ± 2	66.35 ± 0.45
q	1,051 ± 0	3,073 ± 3	852 ± 9	0 ± 1	82 ± 3	2,221 ± 12	127 ± 3	2,011 ± 13	82 ± 3	9.43 ± 0.19
spacl-core	134 ± 0	385 ± 1	130 ± 4	0 ± 0	16 ± 1	256 ± 5	208 ± 5	34 ± 2	1 ± 0	85.93 ± 0.74
zip-a-folder	49 ± 0	145 ± 0	26 ± 2	0 ± 0	2 ± 0	119 ± 2	45 ± 37	5 ± 0	69 ± 38	95.8 ± 0.06
Total	3,376	9,812	2,665	2	321	7,145	3,352	3,436	330	68.9

Table 160: Summary of mutants for mixtral-8x7b-instruct-full-0.0 (runs 360, 361, 362, 363, 364). Each column shows the average number of mutants from all runs, ± the standard deviation.

application	#prompts	Candidates	Invalid	Identical	Duplicate	#mutants	#killed	#survived	#timeout	mut. score
Complex.js	490 ± 0	1,284 ± 11	310 ± 10	0 ± 0	19 ± 4	974 ± 11	592 ± 10	382 ± 6	0 ± 0	60.75 ± 0.63
countries-and-timezones	106 ± 0	267 ± 3	65 ± 3	2 ± 0	4 ± 1	200 ± 4	159 ± 5	41 ± 2	0 ± 0	79.29 ± 1.2
crawler-url-parser	176 ± 0	404 ± 4	171 ± 6	0 ± 0	1 ± 1	233 ± 9	120 ± 6	100 ± 4	0 ± 0	54.54 ± 0.81
delta	462 ± 0	1,118 ± 14	450 ± 7	0 ± 0	21 ± 3	668 ± 9	509 ± 10	124 ± 5	35 ± 1	81.37 ± 0.89
image-downloader	42 ± 0	104 ± 2	38 ± 2	0 ± 0	1 ± 0	66 ± 2	42 ± 2	24 ± 1	0 ± 0	64.04 ± 1.97
node-dirty	154 ± 0	326 ± 15	121 ± 7	0 ± 0	10 ± 0	205 ± 8	121 ± 6	76 ± 4	8 ± 2	62.98 ± 1.27
node-geo-point	140 ± 0	343 ± 3	84 ± 4	0 ± 0	14 ± 2	260 ± 4	175 ± 5	79 ± 2	0 ± 0	68.83 ± 1.14
node-jsonfile	68 ± 0	152 ± 7	23 ± 2	0 ± 0	4 ± 1	129 ± 5	54 ± 3	28 ± 4	47 ± 2	78.28 ± 2.19
plural	153 ± 0	302 ± 3	72 ± 2	0 ± 0	10 ± 2	230 ± 3	171 ± 4	59 ± 2	0 ± 0	74.45 ± 1.19
pull-stream	351 ± 0	928 ± 10	245 ± 8	1 ± 0	6 ± 1	682 ± 9	393 ± 7	243 ± 4	46 ± 2	64.38 ± 0.61
q	1,051 ± 0	2,409 ± 34	763 ± 18	3 ± 0	49 ± 4	1,643 ± 20	115 ± 2	1,458 ± 22	70 ± 3	11.23 ± 0.29
spacl-core	134 ± 0	330 ± 3	151 ± 3	0 ± 0	3 ± 1	179 ± 1	134 ± 0	22 ± 1	1 ± 0	85.9 ± 0.6
zip-a-folder	49 ± 0	118 ± 3	38 ± 3	0 ± 0	0 ± 0	80 ± 4	26 ± 4	45 ± 4	8 ± 1	43.46 ± 3.81
Total	3,376	8,086	2,532	7	142	5,547	2,610	2,682	215	63.81

1.16 StrykerJS standard mutation operators

Table 161 shows the results of running the standard mutation operators of *StrykerJS* on the subject applications.

application	#mutants	#killed	#survived	#timeout	mut. score	time
Complex.js	1,302	763	539	0	58.60	405.08
countries-and-timezones	140	134	6	0	95.71	142.37
crawler-url-parser	226	143	83	0	63.27	433.53
delta	834	686	88	60	89.45	2,747.04
image-downloader	43	28	11	4	74.42	284.20
node-dirty	160	78	56	26	65.00	215.93
node-geo-point	158	98	60	0	62.03	357.70
node-jsonfile	61	31	5	25	91.80	188.91
plural	180	143	37	0	79.44	53.66
pull-stream	474	318	116	40	75.53	694.33
q	1,058	68	927	63	12.38	7,075.02
spacl-core	259	239	20	0	92.28	1,053.16
zip-a-folder	74	38	8	28	89.19	513.27
Total	4,969	2,767	1,956	246	—	14,164.20

Table 161: Results of applying the standard mutation operators of StrykerJS.

1.17 String edit distance measurements for different LLMs

Table 162 shows average string similarity for each project, for each of the five LLMs under consideration.

Project	*codellama-34b-instruct*	*codellama-13b-instruct*	*llama-3.3-70b-instruct*	*gpt-4o-mini*	*mixtral-8x7b-instruct*
Complex.js	4.27	4.63	6.90	6.35	8.95
countries-and-timezones	11.13	9.78	11.71	11.33	13.65
crawler-url-parser	9.50	8.82	10.71	10.05	12.65
delta	9.55	8.88	13.01	11.44	14.15
image-downloader	12.67	10.74	12.66	13.88	14.21
node-dirty	7.53	7.42	12.08	10.20	12.08
node-geo-point	8.86	8.15	8.97	9.53	15.27
node-jsonfile	9.73	10.07	11.10	11.00	10.81
plural	8.14	6.29	9.27	8.31	10.37
pull-stream	6.72	8.71	7.82	8.06	9.67
q	8.61	9.71	11.17	9.92	13.23
spacl-core	9.30	10.84	8.95	8.70	12.61
zip-a-folder	9.85	12.38	11.87	11.23	11.61

Table 162: Average string similarity to the original code fragments that they replace, for mutants generated using five LLMs at temperature 0.0.