LLMorpheus: Mutation Testing using Large Language Models

Anonymous Author(s)

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 where a Large Language Model (LLM) is prompted to suggest mutations by asking it what placeholders that have been inserted in source code 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:
Anonymous Author(s). 2024. LLMorpheus: Mutation Testing using Large Language Models. In *Proceedings of ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2024).* ACM, New York, NY, USA, 39 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 [33–35]. 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 [13], 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 [21, 24].

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 [14, 19], state-of-the-practice tools like Pitest [9, 11], Major [20] and Stryker [38] typically do not implement them because of the implementation effort required and, especially, the increased cost of mutation analysis. Since the cost of performing mutation testing is roughly (number of tests) × (number of mutation operators) × (lines of code), each additional mutation operator can dramatically increase the time needed for the 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" [10]. 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 [32, 39, 41]. However, the need for developers to train a model for their project is an impediment to adoption of such techniques.

Our approach, *LLMorpheus*, repeatedly prompts an LLM to inject faults at designated locations into a code fragment, allowing it to suggest a diversity of mutations. By default, the prompts used by *LLMorpheus* 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.

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 three "open" LLMs for which the training process is documented: *codellama-34b-instruct*, *codellama-13b-instruct*, and *mixtral-8x7b-instruct*. We manually examine a subset of the surviving mutants to determine whether they are (near-) 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.

We find *LLMorpheus* to be capable of generating a diverse set of mutants, some of which resemble real-world bugs that cannot be created by *StrykerJS*' standard mutation operators. We find that

ISSTA 2024, 16-20 September, 2024, Vienna, Austria

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

the majority (63.2%) of surviving mutants reflect behavioral differences, 8.5% are equivalent to the original code, and a further 9.7% are near-equivalent (e.g., replacing the "===" operator with the "==" operator). At temperature 0.0, results are generally stable when experiments are repeated, but the use of higher temperatures yields more variable results. 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. *codellama-34b-instruct* LLM generally produces the largest number of mutants and surviving mutants, but *LLMorpheus* is still effective when *codellama-13b-instruct* and *mixtral-8x7b-instruct* are used.

In summary, the contributions of this paper are:

(1) An LLM-based technique in which an LLM is prompted to suggest mutations by asking what placeholders have been inserted in source code 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.

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 that resemble 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.

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 [4] 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* 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 performance [10, 22, 23]. More significantly, *StrykerJS* does not introduce

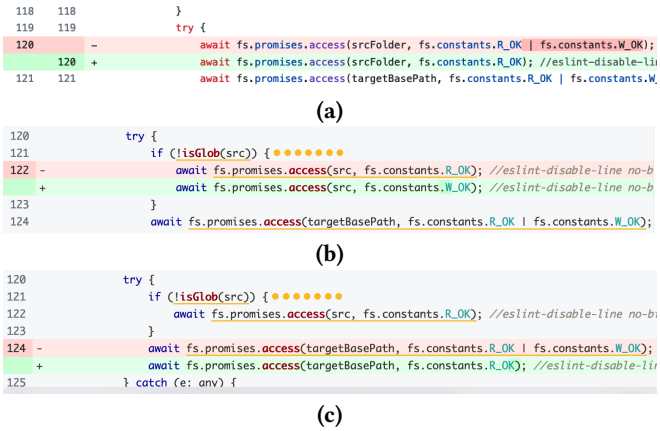

Figure 1: (a) Fix for a bug reported in issue #36 in `zip-a-folder`. (b) A mutation suggested by *LLMorpheus* at the same line that involves replacing read-access with write-access. (c) A mutation suggested by *LLMorpheus* elsewhere in the same file that mirrors the change made by the developer.

or modify property access expressions and has very limited support for replacing an expression with a different expression[1].

Example 2. Countries-and-timezones [2] 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

[1]In particular, *StrykerJS* can only replace control-flow predicates in `if`-statements and loops with boolean constants, string literals with the value `"Stryker was here"`, and it can replace object literals with an empty object literal.
[2]See https://stryker-mutator.io/docs/mutation-testing-elements/supported-mutators/.

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.

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 result in the generation of 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 involves replacing 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 the use of 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." and provided an example illustrating the problem. The developers soon responded with a fix, shown in Figure 3(a), which involved 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

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

```
...  ...  @@ -19,9 +19,13 @@ module.exports.image = ({ extractFilename = tru
19   19       const basename = path.basename(pathname);
20   20       const decodedBasename = decodeURIComponent(basename);
21   21
22        -   options.dest = path.resolve(options.dest, decodedBasename);
     22   +   options.dest = path.join(options.dest, decodedBasename);
23   23     }
24   24   }
```

(a)

```
16       const pathname = url.pathname;
17       const basename = path.basename(pathname);
18       const decodedBasename = decodeURIComponent(basename);
19
20   -   options.dest = path.join(options.dest, decodedBasename);
     +   options.dest = path.basename(options.dest, decodedBasename);
21     }
22   }
23
```

(b)

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.

```
137        return new Promise((resolve, reject) => {
138   -        output.on('close', resolve);
      +        output.on('end', resolve);
139        output.on('error', reject);
```

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

values to identify events precludes static checking, and previous research has focused on static analysis [29] and statistical methods [5] 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, modifying function calls by either adding or removing arguments, and modifying object literals by adding or removing property-value pairs.

In practice, the number of such mutations that could be applied is effectively infinite, so an approach based on exhaustively applying a fixed set of mutation operators is unlikely to be practical. *LLMorpheus*'s 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 suggested code fragments 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 [32, 39, 41]. 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, 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*[4] We now discuss each of these components.

Prompt generator. This component takes as input a package and generates a set of prompts. The prompt generator parses the source files and identifies 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)

[4]In particular, we rely on *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) to generate an interactive web page for inspecting the mutants.

Figure 5: Overview of approach.

`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.

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.

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 specific 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 that were used (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

````
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 for querying the LLM.

all mutants and determines for each mutant whether it causes 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 [1] for parsing source code to identify locations where placeholders should be inserted, and to check syntactic validity of candidate mutants. Handlebars [3] 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, *LLMorpheus* enables users to specify a number of LLM-specific

application	description	weekly downloads	#LOC	#tests	coverage		*StrykerJS*					
					stmt	branch	#total	#killed	#survived	#timeout	mut. score	time (sec)
Complex.js	complex numbers	671K	1,425	216	71.82%	67.54%	1302	763	539	0	58.60	687.01
countries-and-timezones	accessing countries and timezones data	152K	165	58	100%	92.55%	140	134	6	0	95.71	205.88
crawler-url-parser	URL parser for crawling	495	209	185	96.39%	92.5%	226	143	83	0	63.27	739.58
delta	Format for representing rich text documents and changes	1.76M	806	180	98.99%	95.89%	834	686	88	60	89.45	4,200.75
image-downloader	downloading image to disk from a given URL	17.75K	64	11	100%	93.75%	43	28	11	4	74.42	299.72
node-dirty	key value store with append-only disk log	5,604	207	37	83.01%	71.15%	160	78	56	26	65.00	267.33
node-geo-point	calculations involving geographical coordinates	5,618	406	10	85.36%	70.58%	158	98	60	0	62.03	502.91
node-jsonfile	reading/writing JSON files	57.7M	102	43	97.87%	94.11%	61	31	5	25	91.80	220.17
plural	plural forms of nouns	2,271	103	14	95.38%	72.72%	180	143	37	0	79.44	92.90
pull-stream	pipeable pull-stream	57.8K	602	364	90.96%	80.84%	474	318	116	40	75.53	834.18
q	promises	10.1M	2111	243	89.5%	70.92%	1058	68	927	63	12.38	7,516.19
spacl-core	path-based access control	3	377	38	100%	100%	259	239	20	0	92.28	813.55
zip-a-folder	zip/tar utility	60.1K	156	22	100%	96.87&	74	38	8	28	89.19	604.29

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

parameters, such as the maximum length of completions that should be generated, the sampling temperature[5], 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 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.

An open-source release of *LLMorpheus* and of our customized version of *StrykerJS* will take place around the time that this paper is submitted to ISSTA. We plan to submit *LLMorpheus* as an artifact to the Artifact Evaluation Track if the paper is accepted.

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*?

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 [38] could be applied successfully. As a starting point for benchmark selection, we considered the 25 subject applications that were used to evaluate TestPilot [36], a recent LLM-based unit test generation tool. These applications are written in JavaScript or TypeScript, cover a range of different domains, and have associated test suites that can be executed successfully.

Of these 25 subject applications, ten could not be used because StrykerJS does not work on them, either because its dependences conflict with those of the subject application itself[6], 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*[7] reported by *StrykerJS*, and the time required to run *StrykerJS*, respectively.

LLM selection. RQ6 explores how the effectiveness of the proposed technique depends on the LLM being used. After exploring available options, we decided to use Meta's *codellama-34b-instruct* model for our main experiments. In addition, we evaluated the technique with the *codellama-13b-instruct* and *mixtral-8x7b-instruct* models. The *codellama* models are specifically trained for tasks involving code; the *mixtral-8x7b-instruct* is a state-of-the-art general-purpose "mixture-of-experts" LLM developed by Mistral. All three of these LLMs are "open" in the sense that their training process is documented. We relied on a commercial service provided by octo.ai for the experiments reported in this paper.

[5]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.

[6]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.

[7]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/.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	725	473	1	60.55	3,050.00	637.85	967,508	102,517	1,070,025
countries-and-timezones	106	217	188	29	0	86.64	1,070.89	313.86	105,828	23,441	129,269
crawler-url-parser	176	285	157	128	0	55.09	1,642.70	929.43	386,223	39,175	425,398
delta	462	767	634	101	32	86.83	2,961.66	3,839.60	890,252	98,974	989,226
image-downloader	42	89	72	17	0	80.90	430.53	379.25	24,655	9,134	33,789
node-dirty	154	275	163	100	12	63.64	1,526.20	241.81	246,248	33,070	279,318
node-geo-point	140	302	223	79	0	73.84	1,411.11	987.17	316,333	30,013	346,346
node-jsonfile	68	154	49	48	57	68.83	690.61	474.78	57,516	14,797	72,313
plural	153	281	205	75	1	73.31	1,521.32	155.24	265,602	34,174	299,776
pull-stream	351	769	441	271	57	64.76	2,492.50	1,608.97	208,130	76,513	284,643
q	1051	2035	158	1792	85	11.94	5,241.46	14,034.67	2,127,655	220,215	2,347,870
spacl-core	134	239	199	39	1	83.68	1,351.08	798.96	162,705	29,236	191,941
zip-a-folder	49	100	23	3	74	97.00	500.57	1,156.11	82,457	10,725	93,182
Total	3376	6712	3237	3155	320	-	23,890.63	25,557.70	5,841,112	721,984	6,563,096

Table 2: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

	temp. 0.0				temp. 0.25				temp. 0.50				temp. 1.0			
	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout
Complex.js	1199	725	473	1	1197	730	466	1	1191	739	452	0	1028	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	2035	158	1792	85	2050	153	1813	84	2073	163	1823	87	1899	147	1671	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

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

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.

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 1051 prompts for these projects, resulting in between 89 and 2035 mutants. Of these mutants, between 23 and 725 are killed, between 3 and 1792 survive, and between 0 and 85 time out.

LLMs are nondeterministic, even at temperature 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 4 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. [8].

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

Project	*LLMorpheus*				Stryker.js			
	equiv	near	not equiv	unknown	equiv	near	not equiv	unknown
Complex.js	5	0	42	3	0	0	46	4
countries-and-timezones	3	18	8	0	0	0	3	3
crawler-url-parser	1	1	34	14	1	0	32	17
delta	1	1	37	11	5	0	29	16
image-downloader	1	1	6	9	0	0	4	4
node-dirty	5	8	27	10	1	0	35	14
node-geo-point	1	2	40	7	2	0	40	8
node-jsonfile	1	6	23	2	0	0	1	2
plural	10	1	33	6	0	0	28	9
pull-stream	0	10	29	11	0	1	41	8
q	1	2	42	5	0	1	43	6
spacl-core	15	0	6	18	0	1	7	12
zip-a-folder	0	0	0	0	0	0	5	1
Total	44	50	327	96	9	3	314	104

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

Using *codellama-34b-instruct* at temperature 0, *LLMorpheus* generates between 89 and 2035 mutants, of which between 3 and 1792 survive. These results are stable across experiments, with between 89.29% and 98.89% of all mutants being observed in all five experiments.

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 [13]. Mutants produced by *LLMorpheus* may involve arbitrary code changes, so it is entirely possible for the LLM to suggest code that is effectively a refactored version of the code that was originally present. Moreover, it is also possible for the LLM to suggest code that is "near-equivalent". As an example, consider a situation where an LLM suggests replacing a condition `if (x !== false)` with `if (!x)`, which checks if a value is not "falsy" as opposed to comparing it for strict equality against the value `false`.

To determine to what extent surviving mutants produced by *LLMorpheus* are (near-)equivalent, we conducted a study in which we manually examined 50 surviving mutants[9] in each project and classified each mutant as "equivalent", "near-equivalent", "not equivalent", or "unknown". Here, we label mutants as "near-equivalent" if the only observable impact of the mutant could be a change between comparing on a general truthy/falsy value (e.g. if (! x)) and a specific truthy/falsy value (e.g. if (x === undefined)). If evaluating the equivalence of a mutant required examining more than the 15 lines above and below the mutation location, we marked the mutant as "unknown." The results are shown in Table 4. As can be seen from the table, of the 517 mutants under consideration, the majority 327% (63.2%) are "not equivalent", 44 (8.5%) are "equivalent", 50 (9.7%) are "near-equivalent", and remaining 96 (18.6%) are classified as "unknown". We further examined the 44 equivalent mutants, and noticed that most of them involved swapping a utility method call for another with slightly different semantics. For example, *LLMorpheus* might replace a call to `String.indexOf` with a call to `String.lastIndexOf` — but if the surrounding code only checks to see if a substring occurs in a string (and doesn't care about the position), this mutant will be equivalent. As another example, JavaScript's `String` class provides three similar methods: `substr`, `substring` and `slice`. Each returns a portion of a string and accepts two numeric parameters but differ in their exact semantics (e.g., whether the second parameter represents the length to extract or the last index of the string to extract, whether the search wraps around). The behavior of these three methods will be equivalent for some parameter combinations (particularly when only one argument is used).

To place these findings in perspective, we applied the same manual classification to up to 50 surviving mutants produced by *StrykerJS* and found that of 430 surviving mutants, 314 (73%) are "not equivalent", 9 (2%) are "equivalent", 3 (0.7%) are "near-equivalent", and 104 (24.2%) are classified as "unknown".

The majority (63.2%) of the surviving mutants produced by *LLMorpheus* are not equivalent to the original code fragments they replace. *LLMorpheus* produces significantly more "equivalent" and "near equivalent" mutants than *StrykerJS*. However, the number of "not-equivalent" mutants exceeds the number of equivalent and near-equivalent mutants by more than a factor of three.

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

4.5 RQ3: What is the effect of using 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 fairly 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 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.

At temperatures ≤ 0.5, *LLMorpheus* generally produces similar numbers of mutants, of which a similar number survives. At temperature 1.0, the number of generated (and surviving) mutants declines noticeably because the LLM produces more candidate mutants that are syntactically invalid. The variability of results is inversely dependent on the temperature, with mostly the same mutants being produced in different runs at temperature 0, and mostly different mutants being produced in each run at temperature 1.

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[10]:

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>." from the form.

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."

[10] All prompt templates are included in the supplemental materials associated with this paper.

	full				onemutation				noexplanation				noinstructions				genericsystemprompt				basic			
	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout	#total	#killed	#survived	#timeout
Complex.js	1199	725	473	1	406	245	161	0	1125	676	448	1	1137	696	440	1	1199	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	2035	158	1792	85	703	46	630	27	1856	138	1635	83	1958	138	1726	94	2007	145	1770	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	6712	3237	3155	320	2333	1117	1134	82	6305	3053	2950	302	6442	3081	3036	325	6629	3249	3064	316	1326	708	568	50

Table 5: Comparison of the number of mutants generated using the *codellama-34b-instruct* LLM at temperature 0.0 using templates *full, onemutation, noexplanation, noinstructions, gen.system prompt, basic.*

	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.

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 that the placeholder can be replaced with, without any additional context.

Table 5 shows, for each template, the total number of mutants, and the number of mutants that were killed, that survived, and that 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 does not make much difference,
- *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 6712 to 2333 in the aggregate, demonstrating that it is helpful to request multiple suggestions, and
- using *basic* reduces the number of mutants further to 1326, suggesting that additional context in prompts is helpful.

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 of the different configurations would be infeasible, we instead rely on an automated measure. We calculate the Levenshtein string edit distance for each mutant between the mutated code and the 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 that the *basic* template tends 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 the project *crawler-url-parser*, the mutant with the largest string edit distance (297) consists of 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 and surviving mutants. 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?

Thus far, the results 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* and *mixtral-8x7b-instruct* at temperature 0.0.

	codellama-34b-instruct				*codellama-13b-instruct*				*mixtral-8x7b-instruct*			
	#total	**#killed**	**#survived**	**#timeout**	**#total**	**#killed**	**#survived**	**#timeout**	**#total**	**#killed**	**#survived**	**#timeout**
Complex.js	1199	725	473	1	955	553	401	1	962	589	373	0
countries-and-timezones	217	188	29	0	207	177	30	0	205	166	39	0
crawler-url-parser	285	157	128	0	247	129	118	0	234	130	104	0
delta	767	634	101	32	712	583	107	22	680	516	128	36
image-downloader	89	72	17	0	77	48	29	0	69	46	23	0
node-dirty	275	163	100	12	245	142	92	11	191	111	72	8
node-geo-point	302	223	79	0	304	237	67	0	247	166	81	0
node-jsonfile	154	49	48	57	138	43	45	50	132	54	32	46
plural	281	205	75	1	208	154	53	1	226	166	60	0
pull-stream	769	441	271	57	669	386	237	46	678	386	248	44
q	2035	158	1792	85	1713	122	1518	73	1643	112	1460	71
spacl-core	239	199	39	1	185	160	25	0	157	134	22	1
zip-a-folder	100	23	3	74	87	27	55	5	78	24	44	10

Table 7: Comparison of mutants generated using the *codellama-34b-instruct*, *codellama-13b-instruct*, and *mixtral-8x7b-instruct* LLMs at temperature 0.0

Table 7 shows the number of mutants each model produces, classified as killed, surviving, or timed out. From these results, it can be seen that *codellama-34b-instruct* produces more mutants than *codellama-13b-instruct* and *mixtral-8x7b-instruct* in all but one case. However, in terms of surviving mutants, the situation is more ambiguous, with *codellama-13b-instruct* producing the most surviving mutants in 2 cases and *mixtral-8x7b-instruct* in 3 cases.

We also explored the variability of results produced using *codellama-13b-instruct* and *mixtral-8x7b-instruct*, by conducting each experiment 5 times and determining how many distinct mutants are produced and how many mutants occur in all five runs. This revealed that, at temperature 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*, we encountered more variability, with only between 34.22%–50% mutants occurring in 5 runs.

We also examined how similar the mutants produced by the three LLMs are to the original code by calculating the string similarity of the mutants to the original code fragments. We 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 the project *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 the project *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 of the projects, with mutants that tended to include large code declarations. Examining the mutants with the greatest string edit distance for the two codellama variants, we did not find significant trends that held across all targets. For further details, the reader is referred to supplemental materials.

All three LLMs under consideration can be used successfully to generate large numbers of (surviving) mutants. *codellama-34b-instruct* and *codellama-13b-instruct* tend to produce results that are stable across experiments when temperature 0 is used, but *mixtral-8x7b-instruct* produces highly variable results, even at temperature 0.

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

The **time** column in Table 2 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 (about 87 minutes) and *StrykerJS* between 155.24 seconds (about 2.5 minutes) and 14,034.67 seconds (about 234 minutes).

The cost of commercial use of LLMs is typically calculated as a function of the number of tokens that constitute prompts and completions. The last three columns of Table 1 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 writing this paper, 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.

It should be pointed out that the cost of the LLMs we used is significantly lower than that of state-of-the-art proprietary LLMs such as OpenAI's GPT-4 Turbo, for which https://openai.com/pricing quotes a cost of $10 per million input tokens and $30 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. The cost of running *LLMorpheus* on all 13 applications using a commercial LLM service is approximately $3.62, suggesting that cost is not a prohibitive limiting factor.

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. We implemented *LLMorpheus* to target programs written in JavaScript and TypeScript only, and although the approach should apply to other languages, the quality of results is likely to vary between languages. 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 in fact 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 that it was not pre-trained on. We mitigate this risk by including several

projects taken from GitLab (a data source not used in training these models) and did not find a significant variation in *LLMorpheus*'s performance between the GitLab and GitHub projects.

Truly determining if a mutant is equivalent requires significant effort; hence, we use a lightweight manual analysis to determine the number of equivalent mutants generated by *LLMorpheus*. Future research might directly evaluate the coupling of *LLMorpheus*'s faults against real faults. We attempted to utilize a dataset of real faults in JavaScript programs for this purpose [16], but unfortunately found that none of the projects in the dataset were compatible with modern versions of Node.js that *StrykerJS* requires and, hence, not compatible with *LLMorpheus*).

Evaluating tools that rely on LLMs face significant reproducibility challenges. We mitigate these risks by (i) evaluating *LLMorpheus* using open LLMs that are version-controlled and permanently archived, (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.

6 RELATED WORK

Mutation testing, first introduced in the 1970's [13], has a long history. The era of "big code" and software repository mining has enabled the large-scale evaluation of the core hypothesis behind mutation testing, namely, that 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 [21]. This finding has since been replicated on newer, larger datasets consisting of faults from even more Java programs [24]. While this empirical research 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 that are coupled to more real bugs.

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 [14, 18]. More recently, Brown *et al.* improve mutation by mining patches for new idioms to use as mutation operators [8]. Beller *et al.* design a similar tool and evaluate it at Facebook, with the goal of increasing adoption of mutation testing [7] Taking this idea further, Tufano *et al.* create *DeepMutation*, an approach that learns models for performing mutation from real bugs [41]. This idea was refined by Tian *et al.*'s *LEAM*, which improves the search process by leveraging program grammars [39]. Patra and Pradel's *SemSeed* learns to generate mutants from fixes of real-world identifier and literal semantic bugs [32]. Unlike these approaches, *LLMorpheus* uses a *pre-trained LLM*, requiring no training to apply it to a new project.

Much of the research advancing the state of mutation testing tooling has targeted Java, such as MuJava [27], Javalanche [37], Jumble [17], Judy [28] and Major [20]. Gopinath *et al.* empirically compared two of these research-oriented tools tools (Judy [28], Major [20]) with an industry-oriented tool (Pit [11]), finding that despite the stated similarities between the tools, each produced a somewhat different set of mutants [15]. Pit is actively maintained, and the open-source tool is also available packaged with professional plugins under the name 'ArcMutate' [9]. Also aimed at practitioners, the *Stryker* mutation tool is a framework that supports code written in JavaScript, TypeScript, C#, and Scala [38]. We build *LLMorpheus* atop Stryker. Deb *et al.* examine a new, language-agnostic approach to generating mutants using regular expressions [12]. Future work may examine the efficacy of implementing *LLMorpheus* using this approach, which could ease its application to other languages.

Beyond mutation testing, LLMs have also been used for test generation. Bareiß et al. [6] present an approach for test generation that follows a few-shot learning paradigm, outperforming traditional feedback-directed test generation [30]. Tufano et al. [40] present an approach for test generation using a BART transformer model [26] that is fine-tuned on a training set of functions and corresponding tests. Lemieux et al. [25] present an approach where tests generated by Codex are used to assist search-based testing techniques [31] in situations where such techniques get "stuck" because the generated test cases diverge too far from expected uses of the code under test. TestPilot [36] 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, an LLM is given a prompt that includes: general background on mutation testing source code in which a single code fragment is replaced with the word "PLACEHOLDER", the original code fragment that was replaced by the placeholder, 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.

The results of an empirical evaluation on 13 subject applications demonstrate that *LLMorpheus* is capable of producing mutants that resemble real bugs that cannot be produced using standard mutation operators. In a detailed study, we found that the majority (63.2% of surviving mutants produced by *LLMorpheus* are behavioral changes, that the number of (near-)equivalent mutants is less than 20%. 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 three "open" LLMs, we found that *codellama-34b-instruct* generally produced the largest number of mutants and surviving mutants.

In future work, we plan to explore techniques for reducing the number of (near-)equivalent mutants, techniques for generating tests that kill surviving mutants, and methods for evaluating the coupling of *LLMorpheus*'s mutants to real faults.

REFERENCES

[1] 2024. Babel. https://babeljs.io/. Accessed 4/12/2024.

[2] 2024. countries-and-timezones. https://github.com/manuelmhtr/countries-and-timezones. Accessed 4/12/2024.

[3] 2024. Handlebars. https://handlebarsjs.com/. Accessed 4/12/2024.

[4] 2024. zip-a-folder. https://github.com/maugenst/zip-a-folder. Accessed 4/12/2024.

[5] Ellen Arteca, Max Schäfer, and Frank Tip. 2023. Learning How to Listen: Automatically Finding Bug Patterns in Event-Driven JavaScript APIs. *IEEE Trans. Software Eng.* 49, 1 (2023), 166–184. https://doi.org/10.1109/TSE.2022.3147975

[6] Patrick Bareiß, Beatriz Souza, Marcelo d'Amorim, and Michael Pradel. 2022. Code Generation Tools (Almost) for Free? A Study of Few-Shot, Pre-Trained Language Models on Code. *CoRR* abs/2206.01335 (2022). https://doi.org/10.48550/ARXIV.2206.01335 arXiv:2206.01335

[7] Moritz Beller, Chu-Pan Wong, Johannes Bader, Andrew Scott, Mateusz Machalica, Satish Chandra, and Erik Meijer. 2021. 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* (Virtual Event, Spain) *(ICSE-SEIP '21)*. IEEE Press, 268–277. https://doi.org/10.1109/ICSE-SEIP52600.2021.00036

[8] David Bingham Brown, Michael Vaughn, Ben Liblit, and Thomas Reps. 2017. The care and feeding of wild-caught mutants. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering* (Paderborn, Germany) *(ESEC/FSE 2017)*. Association for Computing Machinery, New York, NY, USA, 511–522. https://doi.org/10.1145/3106237.3106280

[9] Henry Coles. 2024. ArcMutate: Advanced mutation testing for Java and Kotlin. https://www.arcmutate.com.

[10] Henry Coles. 2024. ArcMutate: Extended Mutation Operators. https://docs.arcmutate.com/docs/extended-operators.html.

[11] Henry Coles, Thomas Laurent, Christopher Henard, Mike Papadakis, and Anthony Ventresque. 2016. PIT: a practical mutation testing tool for Java (demo). In *Proceedings of the 25th International Symposium on Software Testing and Analysis* (Saarbrücken, Germany) *(ISSTA 2016)*. Association for Computing Machinery, New York, NY, USA, 449–452. https://doi.org/10.1145/2931037.2948707

[12] Sourav Deb, Kush Jain, Rijnard Van Tonder, Claire Le Goues, and Alex Groce. 2024. Syntax Is All You Need: A Universal-Language Approach to Mutant Generation. In *Proceedings of the ACM on Software Engineering (FSE 2024)*.

[13] R.A. DeMillo, R.J. Lipton, and F.G. Sayward. 1978. Hints on Test Data Selection: Help for the Practicing Programmer. *Computer* 11, 4 (1978), 34–41. https://doi.org/10.1109/C-M.1978.218136

[14] Ahmed S. Ghiduk, Moheb R. Girgis, and Marwa H. Shehata. 2017. Higher order mutation testing: A Systematic Literature Review. *Computer Science Review* 25 (2017), 29–48. https://doi.org/10.1016/j.cosrev.2017.06.001

[15] Rahul Gopinath, Iftekhar Ahmed, Mohammad Amin Alipour, Carlos Jensen, and Alex Groce. 2017. Does choice of mutation tool matter? *Software Quality Journal* 25, 3 (2017), 871–920. https://doi.org/10.1007/s11219-016-9317-7

[16] Péter Gyimesi, Béla Vancsics, Andrea Stocco, Davood Mazinanian, Árpád Beszédes, Rudolf Ferenc, and Ali Mesbah. 2019. BugJS: A Benchmark of JavaScript Bugs. In *Proceedings of 12th IEEE International Conference on Software Testing, Verification and Validation (ICST)*.

[17] Sean A. Irvine, Tin Pavlinic, Leonard Trigg, John G. Cleary, Stuart Inglis, and Mark Utting. 2007. 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)*. IEEE Computer Society, USA, 169–175.

[18] Yue Jia and Mark Harman. 2009. Higher Order Mutation Testing. *Inf. Softw. Technol.* 51, 10 (oct 2009), 1379–1393. https://doi.org/10.1016/j.infsof.2009.04.016

[19] Yue Jia and Mark Harman. 2011. An Analysis and Survey of the Development of Mutation Testing. *IEEE Transactions on Software Engineering* 37, 5 (2011), 649–678. https://doi.org/10.1109/TSE.2010.62

[20] René Just. 2014. The Major mutation framework: efficient and scalable mutation analysis for Java. In *Proceedings of the 2014 International Symposium on Software Testing and Analysis* (San Jose, CA, USA) *(ISSTA 2014)*. Association for Computing Machinery, New York, NY, USA, 433–436. https://doi.org/10.1145/2610384.2628053

[21] René Just, Darioush Jalali, Laura Inozemtseva, Michael D. Ernst, Reid Holmes, and Gordon Fraser. 2014. 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* (Hong Kong, China) *(FSE 2014)*. Association for Computing Machinery, New York, NY, USA, 654–665. https://doi.org/10.1145/2635868.2635929

[22] René Just, Bob Kurtz, and Paul Ammann. 2017. Inferring Mutant Utility from Program Context. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*. Santa Barbara, CA, USA, 284–294.

[23] René Just and Franz Schweiggert. 2015. Higher accuracy and lower run time: Efficient mutation analysis using non-redundant mutation operators. *Software Testing, Verification and Reliability (JSTVR)* 25, 5-7 (Jan. 2015), 490–507.

[24] Thomas Laurent, Stephen Gaffney, and Anthony Ventresque. 2022. 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)*. 189–198. https://doi.org/10.1109/ICSTW55395.2022.00042

[25] Caroline Lemieux, Jeevana Priya Inala, Shuvendu K. Lahiri, and Siddhartha Sen. 2023. CodaMOSA: Escaping Coverage Plateaus in Test Generation with Pre-trained Large Language Models. In *45th International Conference on Software Engineering (ICSE)*.

[26] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[27] Yu-Seung Ma, Jeff Offutt, and Yong-Rae Kwon. 2006. MuJava: a mutation system for Java. In *Proceedings of the 28th International Conference on Software Engineering* (Shanghai, China) *(ICSE '06)*. Association for Computing Machinery, New York, NY, USA, 827–830. https://doi.org/10.1145/1134285.1134425

[28] L. Madeyski. 2010. Judy – a mutation testing tool for Java. *IET Software* 4 (February 2010), 32–42(10). Issue 1. https://digital-library.theiet.org/content/journals/10.1049/iet-sen.2008.0038

[29] Magnus Madsen, Frank Tip, and Ondrej Lhoták. 2015. 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, part of SPLASH 2015, Pittsburgh, PA, USA, October 25-30, 2015*, Jonathan Aldrich and Patrick Eugster (Eds.). ACM, 505–519. https://doi.org/10.1145/2814270.2814272

[30] Carlos Pacheco and Michael D. Ernst. 2007. Randoop: Feedback-directed Random Testing for Java. In *Companion to the 22Nd ACM SIGPLAN Conference on Object-oriented Programming Systems and Applications Companion* (Montreal, Quebec, Canada) *(OOPSLA '07)*. ACM, New York, NY, USA, 815–816. https://doi.org/10.1145/1297846.1297902

[31] Annibale Panichella, Fitsum Meshesha Kifetew, and Paolo Tonella. 2018. Automated Test Case Generation as a Many-Objective Optimisation Problem with Dynamic Selection of the Targets. *IEEE Transactions on Software Engineering* 44, 2 (2018), 122–158. https://doi.org/10.1109/TSE.2017.2663435

[32] Jibesh Patra and Michael Pradel. 2021. 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* (Athens, Greece) *(ESEC/FSE 2021)*. Association for Computing Machinery, New York, NY, USA, 906–918. https://doi.org/10.1145/3468264.3468623

[33] Goran Petrović, Marko Ivanković, Gordon Fraser, and René Just. 2021. Does Mutation Testing Improve Testing Practices?. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. 910–921. https://doi.org/10.1109/ICSE43902.2021.00087

[34] Goran Petrović, Marko Ivanković, Gordon Fraser, and René Just. 2022. Practical Mutation Testing at Scale: A view from Google. *IEEE Transactions on Software Engineering* 48, 10 (2022), 3900–3912. https://doi.org/10.1109/TSE.2021.3107634

[35] Goran Petrović, Marko Ivanković, Gordon Fraser, and René Just. 2023. 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)*. 150–161. https://doi.org/10.1109/ICSE-SEIP58684.2023.00019

[36] Max Schäfer, Sarah Nadi, Aryaz Eghbali, and Frank Tip. 2024. An Empirical Evaluation of Using Large Language Models for Automated Unit Test Generation. *IEEE Trans. Software Eng.* 50, 1 (2024), 85–105. https://doi.org/10.1109/TSE.2023.3334955

[37] David Schuler and Andreas Zeller. 2009. 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* (Amsterdam, The Netherlands) *(ESEC/FSE '09)*. Association for Computing Machinery, New York, NY, USA, 297–298. https://doi.org/10.1145/1595696.1595750

[38] Stryker Team. 2024. Stryker Mutator. https://stryker-mutator.io. Accessed 4/12/2024.

[39] Zhao Tian, Junjie Chen, Qihao Zhu, Junjie Yang, and Lingming Zhang. 2023. Learning to Construct Better Mutation Faults. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering (ASE '22)*. Association for Computing Machinery, New York, NY, USA, Article 64, 13 pages. https://doi.org/10.1145/3551349.3556949

[40] Michele Tufano, Dawn Drain, Alexey Svyatkovskiy, Shao Kun Deng, and Neel Sundaresan. 2021. Unit Test Case Generation with Transformers and Focal Context. arXiv. https://www.microsoft.com/en-us/research/publication/unit-test-case-generation-with-transformers-and-focal-context/

[41] Michele Tufano, Cody Watson, Gabriele Bavota, Massimiliano Di Penta, Martin White, and Denys Poshyvanyk. 2019. Learning How to Mutate Source Code from Bug-Fixes. In *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. 301–312. https://doi.org/10.1109/ICSME.2019.00046

A APPENDIX: EXPERIMENTAL DATA

This appendix contains the following experimental results:

- Section A.1 includes the results for 5 experiments using the *codellama-34b-instruct* LLM at temperature 0.0.
- Section A.2 includes the results for 5 experiments using the *codellama-34b-instruct* LLM at temperature 0.25.
- Section A.3 includes the results for 5 experiments using the *codellama-34b-instruct* LLM at temperature 0.5.
- Section A.3 includes the results for 5 experiments using the *codellama-34b-instruct* LLM at temperature 1.0.
- Section A.5 includes the results of variability analysis of the mutants observed in 5 runs using *codellama-34b-instruct*, for each of the temperature settings under consideration.
- Section A.6 includes results for 5 experiments using the *codellama-13b-instruct* LLM at temperature 0.0.
- Section A.7 includes results for 5 experiments using the *mixtral-8x7b-instruct* LLM at temperature 0.0.
- Section A.8 includes results for 5 experiments using the `onemutation` template.
- Section A.9 includes results for 5 experiments using the `noexplanation` template.
- Section A.10 includes results for 5 experiments using the `noinstructions` template.
- Section A.11 includes results for 5 experiments using a generic system prompt.
- Section A.12 includes results for 5 experiments using the `basic` template.
- Section A.13 includes the results of running the standard mutation operators of *StrykerJS* on the subject applications.
- Section A.14 reports measurements of the average string edit distance observed when using different LLMs.

A.1 Results for *codellama-34b-instruct*, `full` template, temperature 0

Tables 8–12 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	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	725	473	1	60.55	3,050.00	637.85	967,508	102,517	1,070,025
countries-and-timezones	106	217	188	29	0	86.64	1,070.89	313.86	105,828	23,441	129,269
crawler-url-parser	176	285	157	128	0	55.09	1,642.70	929.43	386,223	39,175	425,398
delta	462	767	634	101	32	86.83	2,961.66	3,839.60	890,252	98,974	989,226
image-downloader	42	89	72	17	0	80.90	430.53	379.25	24,655	9,134	33,789
node-dirty	154	275	163	100	12	63.64	1,526.20	241.81	246,248	33,070	279,318
node-geo-point	140	302	223	79	0	73.84	1,411.11	987.17	316,333	30,013	346,346
node-jsonfile	68	154	49	48	57	68.83	690.61	474.78	57,516	14,797	72,313
plural	153	281	205	75	1	73.31	1,521.32	155.24	265,602	34,174	299,776
pull-stream	351	769	441	271	57	64.76	2,492.50	1,608.97	208,130	76,513	284,643
q	1051	2035	158	1792	85	11.94	5,241.46	14,034.67	2,127,655	220,215	2,347,870
spacl-core	134	239	199	39	1	83.68	1,351.08	798.96	162,705	29,236	191,941
zip-a-folder	49	100	23	3	74	97.00	500.57	1,156.11	82,457	10,725	93,182
Total	3376	6712	3237	3155	320	-	23,890.63	25,557.70	5,841,112	721,984	6,563,096

Table 8: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1198	726	471	1	60.68	3,087.58	637.10	967,508	102,428	1,069,936
countries-and-timezones	106	217	188	29	0	86.64	1,070.89	313.12	105,828	23,427	129,255
crawler-url-parser	176	282	156	126	0	55.32	1,645.11	679.89	386,223	39,210	425,433
delta	462	768	636	100	32	86.98	2,941.55	3,838.23	890,252	98,951	989,203
image-downloader	42	90	73	17	0	81.11	430.54	377.12	24,655	9,175	33,830
node-dirty	154	274	161	101	12	63.14	1,526.11	248.57	246,248	33,038	279,286
node-geo-point	140	302	223	79	0	73.84	1,411.06	999.59	316,333	29,959	346,292
node-jsonfile	68	153	49	47	57	69.28	690.69	478.32	57,516	14,829	72,345
plural	153	280	204	75	1	73.21	1,521.04	147.85	265,602	34,164	299,766
pull-stream	351	771	441	273	57	64.59	2,477.99	1,400.76	208,130	76,398	284,528
q	1051	2031	159	1788	84	11.96	5,231.88	14,004.86	2,127,655	220,252	2,347,907
spacl-core	134	239	197	41	1	82.85	1,351.05	805.30	162,705	29,283	191,988
zip-a-folder	49	100	23	3	74	97.00	500.57	1,152.62	82,457	10,705	93,162
Total	3376	6705	3236	3150	319	-	23,886.06	25,083.33	5,841,112	721,819	6,562,931

Table 9: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	724	474	1	60.47	3,000.46	630.00	967,508	102,314	1,069,822
countries-and-timezones	106	217	188	29	0	86.64	1,070.90	314.35	105,828	23,438	129,266
crawler-url-parser	176	283	157	126	0	55.48	1,644.58	1,051.08	386,223	39,105	425,328
delta	462	765	633	100	32	86.93	3,006.51	3,795.29	890,252	98,978	989,230
image-downloader	42	89	72	17	0	80.90	430.54	376.08	24,655	9,186	33,841
node-dirty	154	275	161	102	12	62.91	1,526.69	247.49	246,248	33,089	279,337
node-geo-point	140	302	223	79	0	73.84	1,411.05	1,003.93	316,333	30,010	346,343
node-jsonfile	68	154	49	48	57	68.83	690.67	478.93	57,516	14,803	72,319
plural	153	281	205	75	1	73.31	1,521.23	148.33	265,602	34,082	299,684
pull-stream	351	770	441	272	57	64.68	2,492.02	1,392.68	208,130	76,599	284,729
q	1051	2035	159	1793	83	11.89	5,296.20	14,072.49	2,127,655	220,395	2,348,050
spacl-core	134	239	197	41	1	82.85	1,351.09	802.30	162,705	29,334	192,039
zip-a-folder	49	100	23	3	74	97.00	500.56	1,155.04	82,457	10,764	93,221
Total	3376	6709	3232	3159	318	-	23,942.50	25,467.99	5,841,112	722,097	6,563,209

Table 10: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	725	473	1	60.55	3,034.16	631.23	967,508	102,497	1,070,005
countries-and-timezones	106	217	188	29	0	86.64	1,070.91	309.40	105,828	23,444	129,272
crawler-url-parser	176	283	157	125	1	55.83	1,644.09	1,022.89	386,223	39,174	425,397
delta	462	766	634	100	32	86.95	2,983.03	3,969.11	890,252	99,003	989,255
image-downloader	42	89	72	17	0	80.90	430.51	374.45	24,655	9,148	33,803
node-dirty	154	274	160	102	12	62.77	1,563.30	251.17	246,248	33,068	279,316
node-geo-point	140	302	222	80	0	73.51	1,411.00	1,001.75	316,333	30,041	346,374
node-jsonfile	68	154	49	48	57	68.83	690.69	474.83	57,516	14,750	72,266
plural	153	279	203	75	1	73.12	1,521.09	151.19	265,602	34,132	299,734
pull-stream	351	769	444	268	57	65.15	2,541.67	1,398.89	208,130	76,567	284,697
q	1051	2032	158	1790	84	11.91	5,399.09	13,959.40	2,127,655	220,191	2,347,846
spacl-core	134	238	197	40	1	83.19	1,351.08	959.37	162,705	29,287	191,992
zip-a-folder	49	100	23	3	74	97.00	510.58	1,154.14	82,457	10,725	93,182
Total	3376	6702	3232	3150	320	-	24,151.20	25,657.82	5,841,112	722,027	6,563,139

Table 11: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	726	472	1	60.63	3,019.69	669.52	967,508	102,524	1,070,032
countries-and-timezones	106	216	187	29	0	86.57	1,070.82	313.99	105,828	23,425	129,253
crawler-url-parser	176	285	157	128	0	55.09	1,641.20	958.26	386,223	39,160	425,383
delta	462	767	636	100	31	86.96	3,013.74	3,867.52	890,252	99,031	989,283
image-downloader	42	90	73	17	0	81.11	430.56	378.47	24,655	9,117	33,772
node-dirty	154	275	161	102	12	62.91	1,527.49	251.95	246,248	33,113	279,361
node-geo-point	140	302	223	79	0	73.84	1,451.19	1,042.75	316,333	29,894	346,227
node-jsonfile	68	154	49	48	57	68.83	690.66	479.95	57,516	14,803	72,319
plural	153	280	204	75	1	73.21	1,521.18	150.87	265,602	34,163	299,765
pull-stream	351	771	441	273	57	64.59	2,486.78	1,384.03	208,130	76,520	284,650
q	1051	2031	159	1788	84	11.96	5,250.40	14,006.76	2,127,655	220,193	2,347,848
spacl-core	134	239	198	40	1	83.26	1,350.99	808.62	162,705	29,297	192,002
zip-a-folder	49	100	23	3	74	97.00	500.63	1,171.49	82,457	10,705	93,162
Total	3376	6709	3237	3154	318	-	23,955.33	25,484.18	5,841,112	721,945	6,563,057

Table 12: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

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

Tables 13–17 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	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1197	730	466	1	61.07	3,044.19	631.79	967,508	101,588	1,069,096
countries-and-timezones	106	219	181	38	0	82.65	1,070.90	327.25	105,828	23,471	129,299
crawler-url-parser	176	260	167	93	0	64.23	1,646.30	818.05	386,223	39,000	425,223
delta	462	781	642	111	28	85.79	2,954.37	3,844.00	890,252	99,341	989,593
image-downloader	42	86	71	15	0	82.56	430.54	348.86	24,655	9,217	33,872
node-dirty	154	279	175	93	11	66.67	1,532.06	232.35	246,248	32,400	278,648
node-geo-point	140	293	225	68	0	76.79	1,708.63	961.35	291,061	26,301	317,362
node-jsonfile	68	151	52	41	58	72.85	740.73	502.52	57,516	14,400	71,916
plural	153	273	208	63	2	76.92	1,537.89	149.38	265,602	33,182	298,784
pull-stream	351	779	452	270	57	65.34	2,522.38	1,395.14	208,130	76,091	284,221
q	1051	2050	153	1813	84	11.56	5,294.80	14,085.10	2,127,655	218,620	2,346,275
spacl-core	134	223	187	36	0	83.86	1,351.05	728.32	162,705	29,167	191,872
zip-a-folder	49	97	24	4	69	95.88	500.56	1,086.06	82,457	10,557	93,014
Total	3376	6688	3267	3111	310	-	24,334.40	25,110.17	5,815,840	713,335	6,529,175

Table 13: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.25, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1200	729	471	0	60.75	3,029.62	629.66	967,508	100,540	1,068,048
countries-and-timezones	106	223	198	25	0	88.79	1,070.85	314.98	105,828	23,186	129,014
crawler-url-parser	176	269	163	106	0	60.59	1,777.23	867.78	386,223	38,916	425,139
delta	462	752	608	110	34	85.37	2,978.88	3,758.68	890,252	99,176	989,428
image-downloader	42	85	67	18	0	78.82	430.55	366.05	24,655	9,223	33,878
node-dirty	154	271	163	96	12	64.58	1,528.73	237.16	246,248	32,776	279,024
node-geo-point	140	311	243	68	0	78.14	1,411.08	1,021.61	316,333	29,301	345,634
node-jsonfile	68	158	50	54	54	65.82	690.69	485.24	57,516	14,071	71,587
plural	153	283	213	68	2	75.97	1,521.09	154.20	265,602	33,560	299,162
pull-stream	351	772	450	265	57	65.67	2,503.60	1,383.12	208,130	76,551	284,681
q	1051	1980	144	1756	80	11.31	5,379.31	13,584.78	2,127,655	217,699	2,345,354
spacl-core	134	225	194	30	1	86.67	1,350.98	739.03	162,705	29,184	191,889
zip-a-folder	49	95	36	53	6	44.21	500.57	531.10	82,457	10,753	93,210
Total	3376	6624	3258	3120	246	-	24,173.18	24,073.39	5,841,112	714,936	6,556,048

Table 14: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.25, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1195	737	457	1	61.76	3,026.34	650.01	967,508	101,118	1,068,626
countries-and-timezones	106	215	189	26	0	87.91	1,070.91	309.33	105,828	23,331	129,159
crawler-url-parser	176	247	144	103	0	58.30	1,644.86	811.88	386,223	39,215	425,438
delta	462	776	647	100	29	87.11	2,962.02	3,896.15	890,252	99,274	989,526
image-downloader	42	87	71	16	0	81.61	430.54	373.38	24,655	9,163	33,818
node-dirty	154	274	162	100	12	63.50	1,526.45	247.44	246,248	32,894	279,142
node-geo-point	140	294	218	76	0	74.15	1,421.09	935.39	316,333	29,830	346,163
node-jsonfile	68	154	55	43	56	72.08	690.68	504.57	57,516	14,702	72,218
plural	153	286	204	81	1	71.68	1,521.32	151.42	265,602	33,298	298,900
pull-stream	351	773	442	280	51	63.78	2,497.56	1,351.06	208,130	76,100	284,230
q	1051	2002	146	1780	76	11.09	5,341.32	13,704.14	2,127,655	218,805	2,346,460
spacl-core	134	228	193	35	0	84.65	1,351.03	756.66	162,705	28,939	191,644
zip-a-folder	49	98	24	3	71	96.94	500.57	1,119.70	82,457	10,786	93,243
Total	3376	6629	3232	3100	297	-	23,984.69	24,811.13	5,841,112	717,455	6,558,567

Table 15: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.25, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1192	721	471	0	60.49	3,031.72	631.91	967,508	102,075	1,069,583
countries-and-timezones	106	227	200	27	0	88.11	1,070.83	324.27	105,828	23,502	129,330
crawler-url-parser	176	260	151	109	0	58.08	1,642.15	835.28	386,223	39,240	425,463
delta	462	765	619	109	37	85.75	2,969.01	3,912.89	890,252	99,383	989,635
image-downloader	42	82	62	20	0	75.61	430.57	486.28	24,655	9,228	33,883
node-dirty	154	267	158	97	12	63.67	1,527.16	237.65	246,248	32,850	279,098
node-geo-point	140	308	226	82	0	73.38	1,411.10	1,015.42	316,333	28,895	345,228
node-jsonfile	68	151	48	47	56	68.87	690.64	496.41	57,516	14,557	72,073
plural	153	279	212	66	1	76.34	1,521.03	148.10	265,602	33,162	298,764
pull-stream	351	773	443	277	53	64.17	2,509.05	1,370.63	208,130	75,917	284,047
q	1051	2044	141	1822	81	10.86	5,300.14	14,131.12	2,127,655	218,921	2,346,576
spacl-core	134	214	183	30	1	85.98	1,351.01	712.33	162,705	28,809	191,514
zip-a-folder	49	105	27	2	76	98.10	500.63	1,266.61	82,457	10,707	93,164
Total	3376	6667	3191	3159	317	-	23,955.04	25,568.90	5,841,112	717,246	6,558,358

Table 16: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.25, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1168	717	451	0	61.39	3,082.97	613.65	967,508	101,316	1,068,824
countries-and-timezones	106	225	199	26	0	88.44	1,070.86	326.66	105,828	22,979	128,807
crawler-url-parser	176	268	161	107	0	60.07	1,641.17	853.01	386,223	38,790	425,013
delta	462	760	620	106	34	86.05	2,952.50	3,773.99	890,252	99,524	989,776
image-downloader	42	85	71	14	0	83.53	470.58	359.76	24,655	8,898	33,553
node-dirty	154	265	167	86	12	67.55	1,526.98	239.71	246,248	32,476	278,724
node-geo-point	140	296	220	76	0	74.32	1,411.01	1,001.71	316,333	29,427	345,760
node-jsonfile	68	158	55	49	54	68.99	690.74	500.76	57,516	14,495	72,011
plural	153	291	215	75	1	74.23	1,521.11	158.00	265,602	33,838	299,440
pull-stream	351	775	448	276	51	64.39	2,510.21	1,355.46	208,130	75,432	283,562
q	1051	2041	151	1803	87	11.66	5,390.06	14,133.05	2,127,655	217,855	2,345,510
spacl-core	134	226	189	36	1	84.07	1,350.98	752.72	162,705	29,399	192,104
zip-a-folder	49	104	48	50	6	51.92	520.57	564.11	82,457	10,749	93,206
Total	3376	6662	3261	3155	246	-	24,139.74	24,632.59	5,841,112	715,178	6,556,290

Table 17: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.25, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

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

Tables 18–22 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	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1191	739	452	0	62.05	3,016.85	592.07	967,508	100,627	1,068,135
countries-and-timezones	106	224	194	30	0	86.61	1,070.93	319.44	105,828	22,932	128,760
crawler-url-parser	176	298	166	108	24	63.76	1,668.66	1,244.64	386,223	38,920	425,143
delta	462	769	642	93	34	87.91	2,980.99	3,869.42	890,252	98,347	988,599
image-downloader	42	89	68	21	0	76.40	430.55	365.90	24,655	8,925	33,580
node-dirty	154	277	158	107	12	61.37	1,532.57	247.29	246,248	32,995	279,243
node-geo-point	140	302	230	72	0	76.16	1,411.06	1,006.90	316,333	29,454	345,787
node-jsonfile	68	153	51	43	59	71.90	720.71	494.61	57,516	15,051	72,567
plural	153	289	219	69	1	76.12	1,522.56	152.03	265,602	33,550	299,152
pull-stream	351	796	465	278	53	65.08	2,506.81	1,378.63	208,130	75,239	283,369
q	1051	2073	163	1823	87	12.06	5,332.46	14,311.02	2,127,655	217,886	2,345,541
spacl-core	134	250	210	39	1	84.40	1,351.01	850.39	162,705	28,919	191,624
zip-a-folder	49	87	48	33	6	62.07	900.98	471.62	72,362	9,438	81,800
Total	3376	6798	3353	3168	277	-	24,446.14	25,303.96	5,831,017	712,283	6,543,300

Table 18: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.5, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1166	714	452	0	61.23	3,065.41	611.71	967,508	100,190	1,067,698
countries-and-timezones	106	221	192	29	0	86.88	1,070.87	313.43	105,828	22,897	128,725
crawler-url-parser	176	310	173	137	0	55.81	1,643.05	1,045.88	386,223	39,148	425,371
delta	462	792	668	89	35	88.76	3,028.61	4,054.65	890,252	99,121	989,373
image-downloader	42	82	54	28	0	65.85	430.56	496.08	24,655	8,793	33,448
node-dirty	154	288	173	104	11	63.89	1,527.72	252.26	246,248	33,054	279,302
node-geo-point	140	309	230	79	0	74.43	1,411.09	1,060.50	316,333	28,836	345,169
node-jsonfile	68	152	53	38	61	75.00	690.68	520.06	57,516	14,997	72,513
plural	153	285	215	70	0	75.44	1,522.17	146.72	265,602	33,944	299,546
pull-stream	351	784	463	270	51	65.56	2,517.82	1,365.08	208,130	75,400	283,530
q	1051	2020	166	1774	80	12.18	5,232.53	13,865.94	2,127,655	217,305	2,344,960
spacl-core	134	257	221	35	1	86.38	1,351.05	848.71	162,705	28,593	191,298
zip-a-folder	49	103	29	5	69	95.15	500.59	1,105.81	82,457	10,544	93,001
Total	3376	6769	3351	3110	308	-	23,992.15	25,686.83	5,841,112	712,822	6,553,934

Table 19: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.5, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1182	730	452	0	61.76	3,080.04	618.08	967,508	102,242	1,069,750
countries-and-timezones	106	222	199	23	0	89.64	1,070.88	311.67	105,828	22,556	128,384
crawler-url-parser	176	284	166	117	1	58.80	1,645.90	965.22	386,223	38,423	424,646
delta	462	760	621	104	35	86.32	2,992.70	3,820.99	890,252	99,184	989,436
image-downloader	42	85	60	25	0	70.59	430.53	500.27	24,655	9,240	33,895
node-dirty	154	279	166	101	12	63.80	1,527.92	252.60	246,248	32,911	279,159
node-geo-point	140	269	211	58	0	78.44	1,740.11	890.54	289,389	26,285	315,674
node-jsonfile	68	150	54	39	57	74.00	690.70	487.19	57,516	14,355	71,871
plural	153	293	208	84	1	71.33	1,677.14	158.69	249,979	30,944	280,923
pull-stream	351	813	476	278	59	65.81	2,510.30	1,455.39	208,130	75,369	283,499
q	1051	2057	143	1838	76	10.65	5,453.21	14,010.80	2,127,655	217,999	2,345,654
spacl-core	134	260	217	42	1	83.85	1,351.08	861.29	162,705	28,654	191,359
zip-a-folder	49	114	31	4	79	96.49	500.56	1,278.26	82,457	10,677	93,134
Total	3376	6768	3282	3165	321	-	24,671.07	25,610.99	5,798,545	708,839	6,507,384

Table 20: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.5, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1190	716	474	0	60.17	3,025.70	617.91	967,508	101,251	1,068,759
countries-and-timezones	106	221	191	29	1	86.88	1,070.86	333.07	105,828	23,224	129,052
crawler-url-parser	176	279	158	121	0	56.63	1,686.57	939.88	386,223	39,014	425,237
delta	462	775	634	105	36	86.45	3,020.99	3,992.69	890,252	99,683	989,935
image-downloader	42	85	68	17	0	80.00	430.56	347.98	24,655	9,059	33,714
node-dirty	154	269	152	105	12	60.97	1,526.16	227.18	246,248	32,693	278,941
node-geo-point	140	312	229	83	0	73.40	1,411.04	1,057.16	316,333	29,723	346,056
node-jsonfile	68	160	58	36	66	77.50	690.68	565.63	57,516	14,528	72,044
plural	153	296	232	63	1	78.72	1,521.10	156.75	265,602	34,049	299,651
pull-stream	351	789	466	268	55	66.03	2,516.94	1,385.89	208,130	75,071	283,201
q	1051	2033	152	1784	97	12.25	5,280.14	14,131.09	2,127,655	217,911	2,345,566
spacl-core	134	262	223	38	1	85.50	1,350.99	879.83	162,705	28,767	191,472
zip-a-folder	49	100	45	50	5	50.00	510.57	542.42	82,457	10,709	93,166
Total	3376	6771	3324	3173	274	-	24,042.30	25,177.48	5,841,112	715,682	6,556,794

Table 21: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.5, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1186	725	461	0	61.13	3,058.33	648.14	967,508	100,962	1,068,470
countries-and-timezones	106	224	197	27	0	87.95	1,070.88	328.85	105,828	23,165	128,993
crawler-url-parser	176	297	173	124	0	58.25	1,645.45	966.36	386,223	38,649	424,872
delta	462	791	660	92	39	88.37	2,954.29	4,010.15	890,252	97,995	988,247
image-downloader	42	80	68	12	0	85.00	430.57	330.02	24,655	9,182	33,837
node-dirty	154	283	161	110	12	61.13	1,528.13	239.77	246,248	32,568	278,816
node-geo-point	140	305	239	66	0	78.36	1,411.08	1,024.88	316,333	29,330	345,663
node-jsonfile	68	160	51	42	67	73.75	690.67	543.04	57,516	14,833	72,349
plural	153	289	225	64	0	77.85	1,522.09	148.56	265,602	33,906	299,508
pull-stream	351	806	490	255	61	68.36	2,509.37	1,433.54	208,130	75,574	283,704
q	1051	2028	141	1801	86	11.19	5,254.98	13,947.69	2,127,655	216,579	2,344,234
spacl-core	134	252	212	39	1	84.52	1,351.03	834.35	162,705	29,103	191,808
zip-a-folder	49	103	27	5	71	95.15	500.60	1,144.50	82,457	10,347	92,804
Total	3376	6804	3369	3098	337	-	23,927.47	25,599.85	5,841,112	712,193	6,553,305

Table 22: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0.5, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

A.4 Results for *codellama-34b-instruct*, `full` template, temperature 1.0

Tables 23–27 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	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1028	648	379	1	63.13	3,090.47	545.31	967,508	101,516	1,069,024
countries-and-timezones	106	186	156	30	0	83.87	1,070.88	268.04	105,828	23,041	128,869
crawler-url-parser	176	278	202	76	0	72.66	1,786.52	902.86	377,644	36,825	414,469
delta	462	698	583	83	32	88.11	3,134.68	3,531.25	877,266	93,814	971,080
image-downloader	42	75	53	22	0	70.67	430.55	457.54	24,655	9,002	33,657
node-dirty	154	246	150	84	12	65.85	1,526.84	215.86	246,248	32,721	278,969
node-geo-point	140	273	213	60	0	78.02	1,411.05	897.39	316,333	29,494	345,827
node-jsonfile	68	132	50	22	60	83.33	710.73	477.34	57,516	14,447	71,963
plural	153	299	229	69	1	76.92	1,533.27	157.43	265,602	31,993	297,595
pull-stream	351	743	461	235	47	68.37	2,504.20	1,268.62	208,130	73,625	281,755
q	1051	1899	147	1671	81	12.01	5,355.10	13,120.09	2,127,655	213,824	2,341,479
spacl-core	134	218	180	38	0	82.57	1,351.04	679.82	162,705	28,574	191,279
zip-a-folder	49	96	54	38	4	60.42	500.58	493.11	82,457	10,747	93,204
Total	3376	6171	3126	2807	238	-	24,405.91	23,014.66	5,819,547	699,623	6,519,170

Table 23: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 1, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	987	611	376	0	61.90	3,099.95	515.00	967,508	99,831	1,067,339
countries-and-timezones	106	164	145	19	0	88.41	1,537.50	234.01	96,352	19,743	116,095
crawler-url-parser	176	257	177	80	0	68.87	1,643.69	843.29	386,223	36,746	422,969
delta	462	701	596	75	30	89.30	3,119.28	3,448.69	890,252	95,930	986,182
image-downloader	42	59	46	13	0	77.97	430.53	351.02	24,655	8,785	33,440
node-dirty	154	268	164	96	8	64.18	1,526.21	216.70	246,248	31,856	278,104
node-geo-point	140	270	209	61	0	77.41	1,422.53	877.31	316,333	30,037	346,370
node-jsonfile	68	150	67	16	67	89.33	690.70	537.75	57,516	14,738	72,254
plural	153	304	245	58	1	80.92	1,521.12	159.03	265,602	32,288	297,890
pull-stream	351	741	460	238	43	67.88	2,659.93	1,222.06	208,130	73,902	282,032
q	1051	1894	169	1652	73	12.78	5,264.59	12,818.11	2,127,655	212,677	2,340,332
spacl-core	134	205	173	32	0	84.39	1,361.03	682.26	162,705	27,970	190,675
zip-a-folder	49	84	23	4	57	95.24	500.58	908.74	82,457	9,890	92,347
Total	3376	6084	3085	2720	279	-	24,777.64	22,813.97	5,831,636	694,393	6,526,029

Table 24: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 1, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1002	637	365	0	63.57	3,024.06	525.74	967,508	98,687	1,066,195
countries-and-timezones	106	198	177	21	0	89.39	1,070.86	293.27	105,828	23,318	129,146
crawler-url-parser	176	269	173	96	0	64.31	1,644.85	845.43	386,223	36,638	422,861
delta	462	727	611	85	31	88.31	2,948.43	3,663.87	890,252	96,381	986,633
image-downloader	42	78	70	8	0	89.74	430.52	274.68	24,655	9,228	33,883
node-dirty	154	245	136	97	12	60.41	1,526.11	216.05	246,248	31,882	278,130
node-geo-point	140	259	194	65	0	74.90	1,411.07	852.26	316,333	29,152	345,485
node-jsonfile	68	146	68	14	64	90.41	690.73	549.73	57,516	13,670	71,186
plural	153	295	236	58	1	80.34	1,521.17	153.33	265,602	31,946	297,548
pull-stream	351	740	443	248	49	66.49	2,506.18	1,294.61	208,130	72,667	280,797
q	1051	1941	147	1707	87	12.06	5,178.82	13,300.53	2,127,655	213,258	2,340,913
spacl-core	134	225	196	29	0	87.11	1,351.08	743.18	162,705	27,865	190,570
zip-a-folder	49	94	28	4	62	95.74	500.55	1,038.21	82,457	10,518	92,975
Total	3376	6219	3116	2797	306	-	23,804.43	23,750.89	5,841,112	695,210	6,536,322

Table 25: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 1, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1042	667	374	1	64.11	3,070.12	559.72	967,508	100,998	1,068,506
countries-and-timezones	106	202	183	19	0	90.59	1,070.86	285.32	105,828	22,109	127,937
crawler-url-parser	176	276	200	76	0	72.46	1,644.31	872.80	386,223	38,267	424,490
delta	462	744	639	71	34	90.46	2,974.80	3,703.16	890,252	96,594	986,846
image-downloader	42	75	49	26	0	65.33	430.54	438.80	24,655	8,660	33,315
node-dirty	154	260	145	100	15	61.54	1,527.36	248.76	246,248	31,479	277,727
node-geo-point	140	259	205	54	0	79.15	1,411.07	843.77	316,333	29,660	345,993
node-jsonfile	68	143	56	24	63	83.22	690.69	512.58	57,516	14,276	71,792
plural	153	303	242	60	1	80.20	1,521.47	156.91	265,602	32,805	298,407
pull-stream	351	752	456	238	58	68.35	2,495.10	1,354.27	208,130	73,802	281,932
q	1051	1879	130	1659	90	11.71	5,350.94	13,107.50	2,127,655	213,504	2,341,159
spacl-core	134	240	221	19	0	92.08	1,351.11	811.62	162,705	28,193	190,898
zip-a-folder	49	96	38	3	55	96.88	500.61	968.74	82,457	10,354	92,811
Total	3376	6271	3231	2723	317	-	24,038.98	23,863.95	5,841,112	700,701	6,541,813

Table 26: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 1, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1018	640	378	0	62.87	3,021.09	531.51	967,508	100,183	1,067,691
countries-and-timezones	106	207	177	30	0	85.51	1,070.89	297.61	105,828	23,229	129,057
crawler-url-parser	176	252	178	74	0	70.63	1,642.94	776.31	386,223	36,920	423,143
delta	462	679	564	83	32	87.78	2,984.22	3,374.88	890,252	96,698	986,950
image-downloader	42	75	56	17	2	77.33	430.54	435.23	24,655	9,024	33,679
node-dirty	154	267	155	103	9	61.42	1,528.77	227.23	246,248	32,731	278,979
node-geo-point	140	263	198	65	0	75.29	1,411.05	858.36	316,333	29,774	346,107
node-jsonfile	68	141	60	25	56	82.27	690.66	494.30	57,516	14,127	71,643
plural	153	308	245	60	3	80.52	1,521.30	172.76	265,602	32,844	298,446
pull-stream	351	741	457	237	47	68.02	2,481.70	1,251.69	208,130	73,022	281,152
q	1051	1868	130	1652	86	11.56	5,205.03	13,013.51	2,127,655	214,495	2,342,150
spacl-core	134	224	202	21	1	90.63	1,351.07	731.40	162,705	27,723	190,428
zip-a-folder	49	102	28	1	73	99.02	500.58	1,168.69	82,457	10,656	93,113
Total	3376	6145	3090	2746	309	-	23,839.84	23,333.48	5,841,112	701,426	6,542,538

Table 27: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 1, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

A.5 Results for variability analysis for *codellama-34b-instruct*

Table 28 shows the variability of the mutants observed in 5 runs using *codellama-34b-instruct*, for each of the temperature settings under consideration.

application	#min	#max	#distinct	#common
Complex.js	1198	1199	1217	1181 (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	2031	2035	2053	2014 (98.10%)
spacl-core	238	239	252	225 (89.29%)
zip-a-folder	100	100	102	98 (96.08%)

temperature 0.0

application	#min	#max	#distinct	#common
Complex.js	1168	1200	2354	447 (18.99%)
countries-and-timezones	215	227	551	53 (9.62%)
crawler-url-parser	266	285	700	67 (9.57%)
delta	752	781	1706	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	1664	280 (16.83%)
q	1980	2050	4491	714 (15.90%)
spacl-core	230	244	613	58 (9.46%)
zip-a-folder	98	105	233	36 (15.45%)

temperature 0.25

application	#min	#max	#distinct	#common
Complex.js	1166	1191	3196	205 (6.41%)
countries-and-timezones	221	224	735	13 (1.77%)
crawler-url-parser	279	310	966	31 (3.21%)
delta	760	792	2322	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	2183	161 (7.38%)
q	2020	2073	5964	327 (5.48%)
spacl-core	250	262	782	23 (2.94%)
zip-a-folder	87	114	313	17 (5.43%)

temperature 0.5

application	#min	#max	#distinct	#common
Complex.js	987	1042	4200	17 (0.40%)
countries-and-timezones	164	207	835	2 (0.24%)
crawler-url-parser	252	278	1171	1 (0.09%)
delta	681	744	3013	2 (0.07%)
image-downloader	59	78	298	1 (0.34%)
node-dirty	245	268	1075	2 (0.19%)
node-geo-point	259	273	1111	2 (0.18%)
node-jsonfile	132	150	606	1 (0.17%)
plural	295	308	1316	5 (0.38%)
pull-stream	736	749	2814	30 (1.07%)
q	1868	1942	7695	43 (0.56%)
spacl-core	205	240	958	2 (0.21%)
zip-a-folder	84	102	366	6 (1.64%)

temperature 1.0

Table 28: Variability of the mutants generated in 5 runs of *LLMorpheus* at different temperatures. The columns of the tables 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.

A.6 Results for *codellama-13b-instruct*

Tables 29–33 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 34 shows the variability of the mutants observed in 5 runs using *codellama-13b-instruct* at temperature 0.0.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	955	553	401	1	58.01	3,041.69	525.25	967,508	104,246	1,071,754
countries-and-timezones	106	207	177	30	0	85.51	1,070.79	310.80	105,828	23,971	129,799
crawler-url-parser	176	247	129	118	0	52.23	1,637.98	803.17	386,223	39,906	426,129
delta	462	712	583	107	22	84.97	2,993.70	3,529.05	890,252	103,085	993,337
image-downloader	42	77	48	29	0	62.34	430.56	460.64	24,655	9,339	33,994
node-dirty	154	245	142	92	11	62.45	1,526.96	211.68	246,248	34,892	281,140
node-geo-point	140	304	237	67	0	77.96	1,411.04	1,000.74	316,333	30,715	347,048
node-jsonfile	68	138	43	45	50	67.39	690.65	425.39	57,516	15,398	72,914
plural	153	208	154	53	1	74.52	1,521.00	111.98	265,602	34,926	300,528
pull-stream	351	669	386	237	46	64.57	2,489.93	1,188.59	208,130	77,308	285,438
q	1051	1713	122	1518	73	11.38	5,187.67	11,850.22	2,127,655	231,175	2,358,830
spacl-core	134	185	160	25	0	86.49	1,350.97	616.43	162,705	30,694	193,399
zip-a-folder	49	87	27	55	5	36.78	500.51	496.32	82,457	11,494	93,951
Total	3376	5747	2761	2777	209	-	23,853.45	21,530.26	5,841,112	747,149	6,588,261

Table 29: Results obtained with LLMorpheus using the following parameters: model: *codellama-13b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	955	553	401	1	58.01	3,065.29	504.22	967,508	104,246	1,071,754
countries-and-timezones	106	207	177	30	0	85.51	1,070.80	296.35	105,828	23,971	129,799
crawler-url-parser	176	247	129	118	0	52.23	1,646.62	799.72	386,223	39,938	426,161
delta	462	712	583	107	22	84.97	2,972.99	3,511.68	890,252	103,085	993,337
image-downloader	42	77	48	29	0	62.34	430.51	461.29	24,655	9,339	33,994
node-dirty	154	245	142	92	11	62.45	1,527.57	208.98	246,248	34,892	281,140
node-geo-point	140	304	237	67	0	77.96	1,411.07	1,010.61	316,333	30,715	347,048
node-jsonfile	68	137	43	45	49	67.15	690.63	424.92	57,516	15,398	72,914
plural	153	200	148	51	1	74.50	1,696.97	108.86	255,187	33,552	288,739
pull-stream	351	669	386	237	46	64.57	2,488.59	1,182.03	208,130	77,307	285,437
q	1051	1714	122	1519	73	11.38	5,141.59	11,793.14	2,127,655	231,269	2,358,924
spacl-core	134	185	160	25	0	86.49	1,350.98	624.29	162,705	30,694	193,399
zip-a-folder	49	87	27	55	5	36.78	500.55	488.31	82,457	11,494	93,951
Total	3376	5739	2755	2776	208	-	23,994.16	21,414.40	5,830,697	745,900	6,576,597

Table 30: Results obtained with LLMorpheus using the following parameters: model: *codellama-13b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	955	553	401	1	58.01	3,041.58	506.22	967,508	104,246	1,071,754
countries-and-timezones	106	207	177	30	0	85.51	1,070.84	308.67	105,828	23,971	129,799
crawler-url-parser	176	247	129	118	0	52.23	1,639.49	858.45	386,223	39,915	426,138
delta	462	712	583	107	22	84.97	2,978.12	3,447.64	890,252	103,085	993,337
image-downloader	42	77	48	29	0	62.34	430.52	459.07	24,655	9,339	33,994
node-dirty	154	245	142	92	11	62.45	1,527.40	208.98	246,248	34,892	281,140
node-geo-point	140	304	237	67	0	77.96	1,411.02	1,011.58	316,333	30,715	347,048
node-jsonfile	68	137	43	45	49	67.15	690.68	420.13	57,516	15,398	72,914
plural	153	208	154	53	1	74.52	1,521.00	112.81	265,602	34,926	300,528
pull-stream	351	668	386	236	46	64.67	2,481.30	1,179.99	208,130	77,302	285,432
q	1051	1713	122	1518	73	11.38	5,249.72	11,806.54	2,127,655	231,355	2,359,010
spacl-core	134	185	160	25	0	86.49	1,351.04	617.86	162,705	30,694	193,399
zip-a-folder	49	87	27	55	5	36.78	500.58	495.97	82,457	11,494	93,951
Total	3376	5745	2761	2776	208	-	23,893.29	21,433.91	5,841,112	747,332	6,588,444

Table 31: Results obtained with LLMorpheus using the following parameters: model: *codellama-13b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	955	553	401	1	58.01	3,045.49	511.92	967,508	104,246	1,071,754
countries-and-timezones	106	207	177	30	0	85.51	1,070.84	308.11	105,828	23,971	129,799
crawler-url-parser	176	247	129	118	0	52.23	1,639.50	877.27	386,223	39,906	426,129
delta	462	712	583	107	22	84.97	2,975.67	3,533.28	890,252	103,025	993,277
image-downloader	42	77	48	29	0	62.34	430.50	458.40	24,655	9,339	33,994
node-dirty	154	245	142	92	11	62.45	1,527.47	209.67	246,248	34,892	281,140
node-geo-point	140	304	237	67	0	77.96	1,411.06	994.07	316,333	30,715	347,048
node-jsonfile	68	137	43	45	49	67.15	690.64	421.39	57,516	15,398	72,914
plural	153	208	154	53	1	74.52	1,521.06	112.22	265,602	34,926	300,528
pull-stream	351	669	387	236	46	64.72	2,493.00	1,182.21	208,130	77,307	285,437
q	1051	1715	122	1520	73	11.37	5,177.80	11,862.47	2,127,655	231,214	2,358,869
spacl-core	134	185	160	25	0	86.49	1,350.98	606.49	162,705	30,694	193,399
zip-a-folder	49	87	27	55	5	36.78	500.52	486.21	82,457	11,494	93,951
Total	3376	5748	2762	2778	208	-	23,834.53	21,563.71	5,841,112	747,127	6,588,239

Table 32: Results obtained with LLMorpheus using the following parameters: model: *codellama-13b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	955	553	401	1	58.01	3,065.97	514.43	967,508	104,246	1,071,754
countries-and-timezones	106	207	177	30	0	85.51	1,070.82	328.45	105,828	23,951	129,779
crawler-url-parser	176	247	124	118	5	52.23	1,638.27	905.63	386,223	39,906	426,129
delta	462	712	583	107	22	84.97	2,954.39	3,471.08	890,252	103,085	993,337
image-downloader	42	77	48	29	0	62.34	430.50	458.13	24,655	9,339	33,994
node-dirty	154	245	142	92	11	62.45	1,526.96	207.80	246,248	34,892	281,140
node-geo-point	140	304	237	67	0	77.96	1,411.06	1,003.76	316,333	30,715	347,048
node-jsonfile	68	137	43	45	49	67.15	690.67	421.74	57,516	15,398	72,914
plural	153	208	154	53	1	74.52	1,521.03	111.45	265,602	34,926	300,528
pull-stream	351	669	387	236	46	64.72	2,483.02	1,186.04	208,130	77,307	285,437
q	1051	1713	123	1518	72	11.38	5,276.52	11,824.34	2,127,655	231,254	2,358,909
spacl-core	134	185	159	26	0	85.95	1,350.99	617.51	162,705	30,694	193,399
zip-a-folder	49	87	27	55	5	36.78	500.53	490.95	82,457	11,494	93,951
Total	3376	5746	2757	2777	212	-	23,920.73	21,541.31	5,841,112	747,207	6,588,319

Table 33: Results obtained with LLMorpheus using the following parameters: model: *codellama-13b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

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	1713	1715	1725	1702 (98.67%)
spacl-core	195	195	197	193 (97.97%)
zip-a-folder	87	87	87	87 (100.00%)

Table 34: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *codellama-13b-instruct* LLM at temperature 0.0. 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.

A.7 Results for *mixtral-8x7b-instruct*

Tables 35–39 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 40 shows the variability of the mutants observed in 5 runs using *mixtral-8x7b-instruct* at temperature 0.0.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	962	589	373	0	61.23	3,756.38	507.41	960,545	96,072	1,056,617
countries-and-timezones	106	205	166	39	0	80.98	1,080.09	296.25	104,291	22,693	126,984
crawler-url-parser	176	234	130	104	0	55.56	1,697.04	762.88	384,404	33,495	417,899
delta	462	680	516	128	36	81.18	3,636.95	3,476.06	882,477	90,094	972,571
image-downloader	42	69	46	23	0	66.67	430.46	387.36	24,140	8,238	32,378
node-dirty	154	191	111	72	8	62.30	1,665.69	159.16	234,503	24,705	259,208
node-geo-point	140	247	166	81	0	67.21	1,497.29	833.16	315,891	27,864	343,755
node-jsonfile	68	132	54	32	46	75.76	691.90	425.95	56,273	12,371	68,644
plural	153	226	166	60	0	73.45	1,583.06	115.03	259,916	25,067	284,983
pull-stream	351	678	386	248	44	63.42	2,802.21	1,201.97	204,431	70,423	274,854
q	1051	1643	112	1460	71	11.14	7,003.28	11,371.39	2,103,232	192,284	2,295,516
spacl-core	134	157	134	22	1	85.99	1,369.93	534.67	162,695	26,484	189,179
zip-a-folder	49	78	24	44	10	43.59	506.29	470.28	81,279	9,124	90,403
Total	3376	5502	2600	2686	216	-	27,720.57	20,541.57	5,774,077	638,914	6,412,991

Table 35: Results obtained with LLMorpheus using the following parameters: model: *mixtral-8x7b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	989	608	381	0	61.48	3,372.45	520.01	960,545	98,260	1,058,805
countries-and-timezones	106	196	157	39	0	80.10	1,071.57	285.14	104,291	22,228	126,519
crawler-url-parser	176	211	114	97	0	54.03	1,669.78	698.95	384,404	32,771	417,175
delta	462	666	509	123	34	81.53	3,199.74	3,398.17	882,477	89,050	971,527
image-downloader	42	65	42	23	0	64.62	432.08	378.08	24,140	8,005	32,145
node-dirty	154	209	125	76	8	63.64	1,554.61	185.18	242,671	26,837	269,508
node-geo-point	140	257	180	77	0	70.04	1,416.99	850.15	318,251	28,316	346,567
node-jsonfile	68	129	56	27	46	79.07	740.68	431.76	56,273	12,101	68,374
plural	153	232	174	58	0	75.00	1,546.99	120.70	261,626	25,293	286,919
pull-stream	351	694	402	243	49	64.99	2,660.01	1,259.31	204,431	70,142	274,573
q	1051	1609	114	1423	72	11.56	6,149.24	11,173.93	2,103,232	188,223	2,291,455
spacl-core	134	158	135	22	1	86.08	1,416.02	534.70	162,695	26,751	189,446
zip-a-folder	49	86	32	46	8	46.51	500.53	489.08	81,279	9,372	90,651
Total	3376	5501	2648	2635	218	-	25,730.69	20,325.16	5,786,315	637,349	6,423,664

Table 36: Results obtained with LLMorpheus using the following parameters: model: *mixtral-8x7b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	980	594	386	0	60.61	3,359.29	519.50	960,545	96,727	1,057,272
countries-and-timezones	106	197	154	43	0	78.17	1,074.42	277.16	104,291	22,353	126,644
crawler-url-parser	176	223	121	102	0	54.26	1,661.26	731.82	384,404	32,772	417,176
delta	462	660	497	129	34	80.45	3,170.76	3,384.81	882,477	89,334	971,811
image-downloader	42	67	41	26	0	61.19	430.53	386.39	24,140	7,934	32,074
node-dirty	154	213	123	81	9	61.97	1,530.96	182.12	244,297	27,524	271,821
node-geo-point	140	254	177	77	0	69.69	1,413.38	829.76	318,251	27,995	346,246
node-jsonfile	68	126	52	24	50	80.95	690.64	444.51	56,273	11,970	68,243
plural	153	229	169	60	0	73.80	1,524.56	117.41	261,626	25,277	286,903
pull-stream	351	671	389	236	46	64.83	2,644.38	1,205.37	204,431	69,081	273,512
q	1051	1658	115	1477	66	10.92	6,079.84	11,465.34	2,103,232	192,672	2,295,904
spacl-core	134	159	134	24	1	84.91	1,354.60	540.52	162,695	26,151	188,846
zip-a-folder	49	81	23	51	7	37.04	500.60	464.06	81,279	9,340	90,619
Total	3376	5518	2589	2716	213	-	25,435.22	20,548.77	5,787,941	639,130	6,427,071

Table 37: Results obtained with LLMorpheus using the following parameters: model: *mixtral-8x7b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	969	587	382	0	60.58	3,511.86	522.92	960,545	96,846	1,057,391
countries-and-timezones	106	203	160	43	0	78.82	1,073.78	299.00	104,291	22,090	126,381
crawler-url-parser	176	220	118	102	0	53.64	1,664.52	748.35	384,404	32,721	417,125
delta	462	660	500	126	34	80.91	3,343.71	3,332.02	882,477	88,421	970,898
image-downloader	42	64	41	23	0	64.06	440.52	362.20	24,140	7,972	32,112
node-dirty	154	203	120	77	6	62.07	1,532.40	164.86	244,297	26,801	271,098
node-geo-point	140	258	178	80	0	68.99	1,436.26	842.23	318,251	28,074	346,325
node-jsonfile	68	135	56	32	47	76.30	692.85	440.06	56,273	12,731	69,004
plural	153	232	171	61	0	73.71	1,525.78	117.87	261,626	25,198	286,824
pull-stream	351	687	398	245	44	64.34	2,725.42	1,227.21	204,431	70,751	275,182
q	1051	1660	116	1477	67	11.02	6,622.34	11,507.75	2,103,232	194,705	2,297,937
spacl-core	134	157	134	22	1	85.99	1,359.32	532.62	162,695	26,100	188,795
zip-a-folder	49	78	27	42	9	46.15	500.56	463.12	81,279	9,118	90,397
Total	3376	5526	2606	2712	208	-	26,429.32	20,560.21	5,787,941	641,528	6,429,469

Table 38: Results obtained with LLMorpheus using the following parameters: model: *mixtral-8x7b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	969	580	389	0	59.86	3,649.24	528.79	960,545	98,038	1,058,583
countries-and-timezones	106	199	156	43	0	78.39	1,087.49	295.42	104,291	22,259	126,550
crawler-url-parser	176	212	117	95	0	55.19	1,672.78	705.16	384,404	32,640	417,044
delta	462	674	521	116	37	82.79	3,494.16	3,421.43	882,477	90,244	972,721
image-downloader	42	66	42	24	0	63.64	430.50	378.73	24,140	8,213	32,353
node-dirty	154	208	124	73	11	64.90	1,530.34	197.97	244,297	26,982	271,279
node-geo-point	140	258	176	82	0	68.22	1,432.72	865.00	318,251	28,015	346,266
node-jsonfile	68	121	50	25	46	79.34	702.02	419.40	56,273	11,348	67,621
plural	153	232	177	55	0	76.29	1,533.26	118.22	261,626	25,664	287,290
pull-stream	351	678	390	242	46	64.31	2,763.15	1,206.16	204,431	69,471	273,902
q	1051	1644	117	1455	72	11.50	6,879.90	11,339.61	2,103,232	191,046	2,294,278
spacl-core	134	156	134	21	1	86.54	1,409.48	527.44	162,695	26,807	189,502
zip-a-folder	49	75	25	42	8	44.00	500.54	444.67	81,279	9,009	90,288
Total	3376	5492	2609	2662	221	-	27,085.58	20,448.00	5,787,941	639,736	6,427,677

Table 39: Results obtained with LLMorpheus using the following parameters: model: *mixtral-8x7b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#min	#max	#distinct	#common
Complex.js	962	989	1425	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	1609	1660	2438	999 (40.98%)
spacl-core	178	181	261	113 (43.30%)
zip-a-folder	75	86	112	56 (50.00%)

Table 40: Variability of the mutants generated in 5 runs of *LLMorpheus* using the *mixtral-8x7b-instruct* LLM at temperature 0.0. 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.

A.8 Results for `template-onemutation-0.0`

Tables 41–45 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 8 and using the system prompt shown in Figure 7.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	406	245	161	0	60.34	2,784.11	210.86	927,818	39,567	967,385
countries-and-timezones	106	79	65	14	0	82.28	1,071.07	117.59	97,242	8,518	105,760
crawler-url-parser	176	86	50	36	0	58.14	1,636.44	292.18	371,967	15,504	387,471
delta	462	266	221	37	8	86.09	2,676.03	1,251.08	852,830	37,401	890,231
image-downloader	42	34	26	8	0	76.47	430.61	139.23	21,253	3,459	24,712
node-dirty	154	99	55	41	3	58.59	1,526.39	77.95	233,774	12,906	246,680
node-geo-point	140	104	74	30	0	71.15	1,411.29	330.28	304,993	11,192	316,185
node-jsonfile	68	57	18	18	21	68.42	690.81	183.43	52,008	5,846	57,854
plural	153	100	70	30	0	70.00	1,521.37	54.03	253,209	13,450	266,659
pull-stream	351	280	165	95	20	66.07	2,400.61	499.06	179,699	30,228	209,927
q	1051	703	46	630	27	10.38	4,195.04	4,866.38	2,042,524	82,318	2,124,842
spacl-core	134	80	63	17	0	78.75	1,351.30	271.81	151,851	10,803	162,654
zip-a-folder	49	39	19	17	3	56.41	500.63	219.06	78,488	4,405	82,893
Total	3376	2333	1117	1134	82	-	22,195.70	8,512.94	5,567,656	275,597	5,843,253

Table 41: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-onemutation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	406	244	162	0	60.10	2,757.00	215.00	927,818	39,486	967,304
countries-and-timezones	106	79	66	13	0	83.54	1,091.07	116.98	97,242	8,527	105,769
crawler-url-parser	176	86	50	36	0	58.14	1,636.38	300.85	371,967	15,532	387,499
delta	462	267	222	37	8	86.14	2,681.00	1,235.39	852,830	37,383	890,213
image-downloader	42	34	26	8	0	76.47	460.67	139.42	21,253	3,476	24,729
node-dirty	154	99	55	41	3	58.59	1,526.36	75.52	233,774	12,907	246,681
node-geo-point	140	104	74	30	0	71.15	1,411.28	327.69	304,993	11,211	316,204
node-jsonfile	68	57	18	18	21	68.42	730.85	184.62	52,008	5,779	57,787
plural	153	100	70	30	0	70.00	1,521.30	53.73	253,209	13,418	266,627
pull-stream	351	280	164	96	20	65.71	2,397.86	497.98	179,699	30,310	210,009
q	1051	703	46	630	27	10.38	4,204.99	4,839.22	2,042,524	82,262	2,124,786
spacl-core	134	80	63	17	0	78.75	1,351.25	273.58	151,851	10,809	162,660
zip-a-folder	49	39	19	17	3	56.41	500.62	219.57	78,488	4,403	82,891
Total	3376	2334	1117	1135	82	-	22,270.63	8,479.55	5,567,656	275,503	5,843,159

Table 42: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-onemutation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	401	241	160	0	60.10	2,820.16	211.48	916,945	39,061	956,006
countries-and-timezones	106	79	66	13	0	83.54	1,071.10	117.43	97,242	8,548	105,790
crawler-url-parser	176	85	50	35	0	58.82	1,636.46	288.17	371,967	15,519	387,486
delta	462	267	222	37	8	86.14	2,686.73	1,239.88	852,830	37,432	890,262
image-downloader	42	34	26	8	0	76.47	430.69	142.46	21,253	3,475	24,728
node-dirty	154	99	55	41	3	58.59	1,536.37	75.72	233,774	12,869	246,643
node-geo-point	140	104	74	30	0	71.15	1,411.33	338.80	304,993	11,209	316,202
node-jsonfile	68	57	18	18	21	68.42	690.78	183.73	52,008	5,845	57,853
plural	153	101	71	30	0	70.30	1,521.35	54.18	253,209	13,392	266,601
pull-stream	351	280	165	95	20	66.07	2,398.19	499.73	179,699	30,182	209,881
q	1051	704	45	632	27	10.23	4,188.82	4,807.59	2,042,524	82,120	2,124,644
spacl-core	134	80	63	17	0	78.75	1,351.23	272.94	151,851	10,813	162,664
zip-a-folder	49	39	19	17	3	56.41	500.64	222.35	78,488	4,405	82,893
Total	3376	2330	1115	1133	82	-	22,243.85	8,454.46	5,556,783	274,870	5,831,653

Table 43: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-onemutation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	406	245	161	0	60.34	2,754.08	218.66	927,818	39,566	967,384
countries-and-timezones	106	79	66	13	0	83.54	1,071.03	119.64	97,242	8,579	105,821
crawler-url-parser	176	87	51	36	0	58.62	1,636.41	281.15	371,967	15,525	387,492
delta	462	266	221	37	8	86.09	2,676.11	1,249.77	852,830	37,449	890,279
image-downloader	42	34	26	8	0	76.47	430.60	138.45	21,253	3,475	24,728
node-dirty	154	98	55	40	3	59.18	1,526.42	73.92	233,774	12,859	246,633
node-geo-point	140	104	74	30	0	71.15	1,411.28	329.40	304,993	11,210	316,203
node-jsonfile	68	57	18	18	21	68.42	690.77	183.71	52,008	5,787	57,795
plural	153	101	71	30	0	70.30	1,521.36	53.81	253,209	13,434	266,643
pull-stream	351	280	165	95	20	66.07	2,397.58	499.25	179,699	30,160	209,859
q	1051	703	46	630	27	10.38	4,211.46	4,819.77	2,042,524	82,203	2,124,727
spacl-core	134	80	63	17	0	78.75	1,361.19	269.34	151,851	10,818	162,669
zip-a-folder	49	39	19	17	3	56.41	500.61	217.85	78,488	4,400	82,888
Total	3376	2334	1120	1132	82	-	22,188.90	8,454.72	5,567,656	275,465	5,843,121

Table 44: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-onemutation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	406	245	161	0	60.34	2,763.09	214.80	927,818	39,505	967,323
countries-and-timezones	106	79	66	13	0	83.54	1,071.04	115.36	97,242	8,565	105,807
crawler-url-parser	176	87	51	36	0	58.62	1,636.32	285.19	371,967	15,616	387,583
delta	462	266	221	37	8	86.09	2,676.35	1,249.33	852,830	37,349	890,179
image-downloader	42	34	26	8	0	76.47	430.63	137.29	21,253	3,461	24,714
node-dirty	154	98	55	40	3	59.18	1,526.34	74.50	233,774	12,868	246,642
node-geo-point	140	104	74	30	0	71.15	1,411.31	337.02	304,993	11,183	316,176
node-jsonfile	68	57	18	18	21	68.42	690.79	183.54	52,008	5,774	57,782
plural	153	101	71	30	0	70.30	1,521.35	52.33	253,209	13,401	266,610
pull-stream	351	280	165	95	20	66.07	2,403.01	497.08	179,699	30,238	209,937
q	1051	703	46	630	27	10.38	4,196.27	4,832.34	2,042,524	82,120	2,124,644
spacl-core	134	80	63	17	0	78.75	1,351.29	269.13	151,851	10,793	162,644
zip-a-folder	49	39	19	17	3	56.41	500.65	220.66	78,488	4,403	82,891
Total	3376	2334	1120	1132	82	-	22,178.44	8,468.57	5,567,656	275,276	5,842,932

Table 45: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-onemutation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

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

A.9 Results for `template-noexplanation-0.0`

Tables 46–50 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 9 and using the system prompt shown in Figure 7.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1125	676	448	1	60.18	3,056.33	600.05	948,398	75,551	1,023,949
countries-and-timezones	106	211	183	28	0	86.73	1,070.68	309.71	101,694	23,742	125,436
crawler-url-parser	176	239	140	99	0	58.58	1,656.21	778.34	379,359	31,115	410,474
delta	462	734	598	110	26	85.01	2,870.28	3,625.25	872,234	64,880	937,114
image-downloader	42	77	62	15	0	80.52	430.47	329.74	23,017	9,110	32,127
node-dirty	154	258	146	99	13	61.63	1,526.57	243.81	240,242	24,279	264,521
node-geo-point	140	297	216	81	0	72.73	1,411.01	1,009.62	310,873	26,100	336,973
node-jsonfile	68	152	54	45	53	70.39	690.59	482.67	54,864	15,154	70,018
plural	153	273	198	74	1	72.89	1,522.48	146.41	259,635	26,465	286,100
pull-stream	351	774	440	278	56	64.08	2,632.74	1,388.06	194,441	73,821	268,262
q	1051	1856	138	1635	83	11.91	4,694.78	12,908.57	2,086,666	127,647	2,214,313
spacl-core	134	211	175	35	1	83.41	1,350.87	711.26	157,479	28,201	185,680
zip-a-folder	49	98	27	3	68	96.94	500.54	1,097.23	80,546	10,243	90,789
Total	3376	6305	3053	2950	302	-	23,413.55	23,630.72	5,709,448	536,308	6,245,756

Table 46: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noexplanation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1125	678	446	1	60.36	3,088.22	620.07	948,398	75,464	1,023,862
countries-and-timezones	106	211	183	28	0	86.73	1,070.66	302.71	101,694	23,766	125,460
crawler-url-parser	176	239	140	99	0	58.58	1,653.80	799.42	379,359	31,089	410,448
delta	462	733	598	108	27	85.27	2,871.99	3,718.30	872,234	65,148	937,382
image-downloader	42	77	62	15	0	80.52	430.48	328.19	23,017	9,096	32,113
node-dirty	154	258	146	99	13	61.63	1,526.65	242.49	240,242	24,129	264,371
node-geo-point	140	298	216	82	0	72.48	1,410.97	985.29	310,873	26,143	337,016
node-jsonfile	68	152	54	45	53	70.39	690.54	481.86	54,864	15,125	69,989
plural	153	273	198	74	1	72.89	1,522.07	145.74	259,635	26,527	286,162
pull-stream	351	774	439	279	56	63.95	2,643.33	1,393.56	194,441	73,922	268,363
q	1051	1854	138	1633	83	11.92	4,623.16	12,860.56	2,086,666	127,954	2,214,620
spacl-core	134	209	175	33	1	84.21	1,360.90	649.12	157,479	28,174	185,653
zip-a-folder	49	98	26	3	69	96.94	500.51	1,117.51	80,546	10,267	90,813
Total	3376	6301	3053	2944	304	-	23,393.28	23,644.82	5,709,448	536,804	6,246,252

Table 47: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noexplanation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1121	674	446	1	60.21	3,054.59	596.41	948,398	75,593	1,023,991
countries-and-timezones	106	211	183	28	0	86.73	1,070.75	306.48	101,694	23,740	125,434
crawler-url-parser	176	239	140	99	0	58.58	1,653.25	791.38	379,359	31,096	410,455
delta	462	733	597	110	26	84.99	2,869.41	3,656.14	872,234	64,872	937,106
image-downloader	42	77	62	15	0	80.52	430.47	328.44	23,017	9,109	32,126
node-dirty	154	258	146	99	13	61.63	1,526.59	236.98	240,242	24,096	264,338
node-geo-point	140	298	216	82	0	72.48	1,410.99	992.09	310,873	26,143	337,016
node-jsonfile	68	152	54	45	53	70.39	690.55	485.04	54,864	15,130	69,994
plural	153	273	198	74	1	72.89	1,521.62	145.62	259,635	26,482	286,117
pull-stream	351	774	439	279	56	63.95	2,631.99	1,391.93	194,441	73,754	268,195
q	1051	1855	138	1634	83	11.91	4,695.78	12,866.72	2,086,666	127,918	2,214,584
spacl-core	134	209	175	33	1	84.21	1,350.86	706.03	157,479	28,159	185,638
zip-a-folder	49	98	26	3	69	96.94	500.51	1,109.02	80,546	10,227	90,773
Total	3376	6298	3048	2947	303	-	23,407.36	23,612.28	5,709,448	536,319	6,245,767

Table 48: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noexplanation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1126	679	446	1	60.39	3,053.84	617.90	948,398	75,377	1,023,775
countries-and-timezones	106	211	183	28	0	86.73	1,070.72	306.03	101,694	23,805	125,499
crawler-url-parser	176	239	140	99	0	58.58	1,656.36	838.63	379,359	31,102	410,461
delta	462	733	597	109	27	85.13	2,886.73	3,644.21	872,234	64,947	937,181
image-downloader	42	77	62	15	0	80.52	430.48	330.08	23,017	9,107	32,124
node-dirty	154	258	146	99	13	61.63	1,526.82	243.25	240,242	24,153	264,395
node-geo-point	140	298	216	82	0	72.48	1,410.93	999.44	310,873	26,143	337,016
node-jsonfile	68	152	54	45	53	70.39	690.54	480.47	54,864	15,130	69,994
plural	153	273	198	74	1	72.89	1,522.37	144.72	259,635	26,473	286,108
pull-stream	351	773	440	277	56	64.17	2,649.32	1,395.27	194,441	73,826	268,267
q	1051	1854	136	1634	84	11.87	4,627.96	12,851.25	2,086,666	127,807	2,214,473
spacl-core	134	209	177	31	1	85.17	1,350.90	680.52	157,479	28,203	185,682
zip-a-folder	49	98	27	3	68	96.94	500.52	1,098.96	80,546	10,244	90,790
Total	3376	6301	3055	2942	304	-	23,377.49	23,630.73	5,709,448	536,317	6,245,765

Table 49: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noexplanation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1125	678	446	1	60.36	3,048.61	622.31	948,398	75,411	1,023,809
countries-and-timezones	106	211	183	28	0	86.73	1,070.74	302.32	101,694	23,740	125,434
crawler-url-parser	176	239	140	99	0	58.58	1,653.97	835.85	379,359	30,947	410,306
delta	462	733	597	110	26	84.99	2,880.16	3,648.84	872,234	65,086	937,320
image-downloader	42	77	62	15	0	80.52	430.49	326.72	23,017	9,110	32,127
node-dirty	154	258	146	99	13	61.63	1,526.71	230.04	240,242	24,142	264,384
node-geo-point	140	295	213	82	0	72.20	1,410.98	963.40	310,873	26,313	337,186
node-jsonfile	68	152	54	45	53	70.39	690.67	530.41	54,864	15,130	69,994
plural	153	273	198	74	1	72.89	1,522.04	148.93	259,635	26,465	286,100
pull-stream	351	774	438	280	56	63.82	2,630.34	1,397.17	194,441	73,763	268,204
q	1051	1856	137	1635	84	11.91	4,627.86	12,869.01	2,086,666	127,790	2,214,456
spacl-core	134	209	175	33	1	84.21	1,350.91	714.31	157,479	28,174	185,653
zip-a-folder	49	98	27	3	68	96.94	500.51	1,073.84	80,546	10,244	90,790
Total	3376	6300	3048	2949	303	-	23,343.99	23,663.15	5,709,448	536,315	6,245,763

Table 50: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noexplanation.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

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

A.10 Results for `template-noinstructions-0.0`

Tables 51–55 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 10 and using the system prompt shown in Figure 7.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1137	696	440	1	61.30	3,363.13	610.09	953,788	104,944	1,058,732
countries-and-timezones	106	218	174	44	0	79.82	1,070.70	333.60	102,860	23,506	126,366
crawler-url-parser	176	246	134	112	0	54.47	1,668.12	865.68	381,295	38,817	420,112
delta	462	759	612	115	32	84.85	3,242.91	4,060.50	877,316	99,522	976,838
image-downloader	42	84	69	15	0	82.14	430.47	361.71	23,479	8,905	32,384
node-dirty	154	260	146	103	11	60.38	1,530.58	228.43	241,936	33,033	274,969
node-geo-point	140	306	230	76	0	75.16	1,410.89	1,019.95	312,413	28,975	341,388
node-jsonfile	68	148	45	51	52	65.54	690.55	469.01	55,612	14,598	70,210
plural	153	261	189	71	1	72.80	1,523.05	141.16	261,318	34,491	295,809
pull-stream	351	781	467	248	66	68.25	2,608.43	1,433.25	198,302	74,144	272,446
q	1051	1958	138	1726	94	11.85	5,802.92	13,526.38	2,098,227	218,277	2,316,504
spacl-core	134	187	155	31	1	83.42	1,350.79	626.46	158,953	29,519	188,472
zip-a-folder	49	97	26	4	67	95.88	500.51	1,087.03	81,085	10,694	91,779
Total	3376	6442	3081	3036	325	-	25,193.05	24,763.25	5,746,584	719,425	6,466,009

Table 51: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noinstructions.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1137	696	440	1	61.30	3,324.82	606.13	953,788	104,909	1,058,697
countries-and-timezones	106	218	174	44	0	79.82	1,070.79	314.67	102,860	23,502	126,362
crawler-url-parser	176	246	134	112	0	54.47	1,673.87	789.35	381,295	38,816	420,111
delta	462	759	610	117	32	84.58	3,186.56	3,835.89	877,316	99,463	976,779
image-downloader	42	84	70	14	0	83.33	430.46	362.97	23,479	8,961	32,440
node-dirty	154	260	146	103	11	60.38	1,530.82	229.51	241,936	33,036	274,972
node-geo-point	140	306	230	76	0	75.16	1,410.92	1,035.06	312,413	28,975	341,388
node-jsonfile	68	148	45	51	52	65.54	691.16	471.51	55,612	14,598	70,210
plural	153	261	189	71	1	72.80	1,523.31	142.93	261,318	34,484	295,802
pull-stream	351	781	467	248	66	68.25	2,610.16	1,428.74	198,302	74,195	272,497
q	1051	1960	137	1728	95	11.84	5,806.23	13,570.79	2,098,227	218,057	2,316,284
spacl-core	134	188	156	31	1	83.51	1,350.78	598.84	158,953	29,457	188,410
zip-a-folder	49	97	26	4	67	95.88	500.50	1,065.03	81,085	10,694	91,779
Total	3376	6445	3080	3039	326	-	25,110.38	24,451.42	5,746,584	719,147	6,465,731

Table 52: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noinstructions.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1137	696	440	1	61.30	3,340.83	597.34	953,788	105,056	1,058,844
countries-and-timezones	106	218	174	44	0	79.82	1,070.75	323.54	102,860	23,502	126,362
crawler-url-parser	176	246	134	112	0	54.47	1,659.88	803.01	381,295	38,801	420,096
delta	462	759	611	116	32	84.72	3,211.07	3,830.91	877,316	99,521	976,837
image-downloader	42	84	69	15	0	82.14	430.47	362.10	23,479	8,905	32,384
node-dirty	154	260	146	103	11	60.38	1,530.67	228.98	241,936	33,033	274,969
node-geo-point	140	306	230	76	0	75.16	1,410.92	997.90	312,413	29,053	341,466
node-jsonfile	68	148	45	51	52	65.54	690.56	466.32	55,612	14,598	70,210
plural	153	261	189	71	1	72.80	1,522.82	141.80	261,318	34,484	295,802
pull-stream	351	781	466	249	66	68.12	2,609.79	1,444.02	198,302	74,220	272,522
q	1051	1957	137	1727	93	11.75	6,945.19	13,543.90	2,098,227	218,309	2,316,536
spacl-core	134	187	155	31	1	83.42	1,350.87	605.57	158,953	29,527	188,480
zip-a-folder	49	97	26	4	67	95.88	500.50	1,075.42	81,085	10,694	91,779
Total	3376	6441	3078	3039	324	-	26,274.32	24,420.81	5,746,584	719,703	6,466,287

Table 53: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noinstructions.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1136	695	440	1	61.27	3,492.66	629.00	953,788	104,886	1,058,674
countries-and-timezones	106	218	174	44	0	79.82	1,177.99	323.34	102,860	23,502	126,362
crawler-url-parser	176	246	134	112	0	54.47	1,751.37	786.06	381,295	38,800	420,095
delta	462	759	612	115	32	84.85	3,365.20	3,872.79	877,316	99,562	976,878
image-downloader	42	84	70	14	0	83.33	441.26	358.96	23,479	8,961	32,440
node-dirty	154	260	147	102	11	60.77	1,608.97	228.53	241,936	33,053	274,989
node-geo-point	140	306	230	76	0	75.16	1,492.54	1,035.93	312,413	28,984	341,397
node-jsonfile	68	148	45	51	52	65.54	717.49	465.56	55,612	14,548	70,160
plural	153	262	189	72	1	72.52	1,630.97	140.39	261,318	34,444	295,762
pull-stream	351	779	464	249	66	68.04	2,738.37	1,433.35	198,302	74,222	272,524
q	1051	1956	138	1725	93	11.81	6,155.10	13,521.99	2,098,227	218,163	2,316,390
spacl-core	134	187	155	31	1	83.42	1,440.32	585.59	158,953	29,512	188,465
zip-a-folder	49	97	26	4	67	95.88	509.75	1,063.09	81,085	10,694	91,779
Total	3376	6438	3079	3035	324	-	26,521.99	24,444.58	5,746,584	719,331	6,465,915

Table 54: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noinstructions.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1137	696	440	1	61.30	3,378.96	601.89	953,788	104,940	1,058,728
countries-and-timezones	106	218	174	44	0	79.82	1,080.75	325.94	102,860	23,502	126,362
crawler-url-parser	176	246	134	112	0	54.47	1,660.11	741.99	381,295	38,801	420,096
delta	462	759	612	115	32	84.85	3,233.82	3,810.62	877,316	99,525	976,841
image-downloader	42	84	69	15	0	82.14	430.46	361.68	23,479	8,905	32,384
node-dirty	154	262	149	102	11	61.07	1,531.35	234.89	241,936	33,044	274,980
node-geo-point	140	306	230	76	0	75.16	1,410.90	1,014.28	312,413	28,969	341,382
node-jsonfile	68	148	45	51	52	65.54	690.54	466.32	55,612	14,598	70,210
plural	153	261	189	71	1	72.80	1,523.06	136.58	261,318	34,492	295,810
pull-stream	351	779	465	248	66	68.16	2,606.22	1,436.07	198,302	74,135	272,437
q	1051	1958	137	1728	93	11.75	5,921.20	13,597.41	2,098,227	218,214	2,316,441
spacl-core	134	187	155	31	1	83.42	1,350.81	585.65	158,953	29,520	188,473
zip-a-folder	49	97	26	4	67	95.88	500.48	1,078.08	81,085	10,745	91,830
Total	3376	6442	3081	3037	324	-	25,318.66	24,391.40	5,746,584	719,390	6,465,974

Table 55: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-noinstructions.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

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

A.11 Results for `template-full-genericsystemprompt-0.0`

Tables 56–60 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 11.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	740	458	1	61.80	3,198.91	631.03	943,498	97,397	1,040,895
countries-and-timezones	106	217	191	26	0	88.02	1,070.69	306.50	100,634	22,822	123,456
crawler-url-parser	176	246	143	103	0	58.13	1,666.72	773.61	377,599	38,968	416,567
delta	462	790	659	99	32	87.47	3,188.39	3,973.51	867,614	96,702	964,316
image-downloader	42	88	72	16	0	81.82	430.51	376.36	22,597	8,748	31,345
node-dirty	154	277	162	104	11	62.45	1,531.02	243.07	238,702	32,642	271,344
node-geo-point	140	305	229	76	0	75.08	1,410.94	993.80	309,473	28,703	338,176
node-jsonfile	68	150	49	49	52	67.33	690.59	470.19	54,184	13,966	68,150
plural	153	272	209	62	1	77.21	1,523.29	146.19	258,105	33,232	291,337
pull-stream	351	763	442	266	55	65.14	2,621.42	1,374.37	190,931	73,130	264,061
q	1051	2007	145	1770	92	11.81	5,911.30	13,892.03	2,076,156	216,002	2,292,158
spacl-core	134	214	181	32	1	85.05	1,350.90	688.46	156,139	28,052	184,191
zip-a-folder	49	101	27	3	71	97.03	500.50	1,139.30	80,056	10,370	90,426
Total	3376	6629	3249	3064	316	-	25,095.18	25,008.42	5,675,688	700,734	6,376,422

Table 56: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-Generic.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	740	458	1	61.80	3,198.28	628.16	943,498	97,360	1,040,858
countries-and-timezones	106	217	191	26	0	88.02	1,070.75	313.88	100,634	22,817	123,451
crawler-url-parser	176	246	143	103	0	58.13	1,667.00	785.35	377,599	38,968	416,567
delta	462	788	657	99	32	87.44	3,193.53	4,061.32	867,614	96,672	964,286
image-downloader	42	88	72	16	0	81.82	430.50	373.49	22,597	8,748	31,345
node-dirty	154	277	162	104	11	62.45	1,530.73	243.51	238,702	32,641	271,343
node-geo-point	140	305	229	76	0	75.08	1,411.00	1,029.33	309,473	28,703	338,176
node-jsonfile	68	151	49	49	53	67.55	690.57	473.38	54,184	13,976	68,160
plural	153	272	209	62	1	77.21	1,523.28	142.88	258,105	33,183	291,288
pull-stream	351	764	443	266	55	65.18	2,622.56	1,368.80	190,931	73,002	263,933
q	1051	2008	145	1771	92	11.80	5,761.33	13,943.75	2,076,156	216,075	2,292,231
spacl-core	134	214	181	32	1	85.05	1,350.88	668.94	156,139	28,048	184,187
zip-a-folder	49	101	27	3	71	97.03	500.52	1,170.72	80,056	10,370	90,426
Total	3376	6630	3248	3065	317	-	24,950.93	25,203.51	5,675,688	700,563	6,376,251

Table 57: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-Generic.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	740	458	1	61.80	3,246.58	637.01	943,498	97,351	1,040,849
countries-and-timezones	106	217	191	26	0	88.02	1,070.74	317.35	100,634	22,822	123,456
crawler-url-parser	176	246	143	103	0	58.13	1,666.54	828.48	377,599	38,968	416,567
delta	462	792	660	100	32	87.37	3,141.81	3,972.35	867,614	96,648	964,262
image-downloader	42	88	72	16	0	81.82	430.53	375.91	22,597	8,740	31,337
node-dirty	154	277	162	104	11	62.45	1,531.64	237.14	238,702	32,632	271,334
node-geo-point	140	305	229	76	0	75.08	1,410.80	1,007.52	309,473	28,670	338,143
node-jsonfile	68	151	49	49	53	67.55	690.60	479.57	54,184	13,982	68,166
plural	153	270	207	62	1	77.04	1,523.32	141.62	258,105	33,221	291,326
pull-stream	351	763	440	268	55	64.88	2,590.59	1,371.20	190,931	73,097	264,028
q	1051	2007	144	1771	92	11.76	5,912.17	13,970.49	2,076,156	216,015	2,292,171
spacl-core	134	216	183	32	1	85.19	1,461.05	690.80	156,139	28,074	184,213
zip-a-folder	49	101	27	3	71	97.03	500.51	1,128.17	80,056	10,370	90,426
Total	3376	6632	3247	3068	317	-	25,176.88	25,157.61	5,675,688	700,590	6,376,278

Table 58: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-Generic.txt, rateLimit: benchmark mode, nrAttempts: 3

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

Figure 11: Generic system prompt.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1200	741	458	1	61.83	3,229.79	627.53	943,498	97,331	1,040,829
countries-and-timezones	106	217	191	26	0	88.02	1,070.66	315.55	100,634	22,813	123,447
crawler-url-parser	176	246	144	102	0	58.54	1,677.85	830.94	377,599	39,015	416,614
delta	462	791	659	100	32	87.36	3,135.61	3,928.08	867,614	96,647	964,261
image-downloader	42	88	72	16	0	81.82	430.47	374.42	22,597	8,748	31,345
node-dirty	154	277	163	103	11	62.82	1,530.49	236.58	238,702	32,642	271,344
node-geo-point	140	305	229	76	0	75.08	1,410.83	1,012.70	309,473	28,703	338,176
node-jsonfile	68	151	49	49	53	67.55	690.54	473.16	54,184	13,999	68,183
plural	153	270	207	62	1	77.04	1,523.30	142.61	258,105	33,221	291,326
pull-stream	351	763	442	266	55	65.14	2,590.76	1,370.63	190,931	73,109	264,040
q	1051	2006	144	1770	92	11.76	5,913.75	13,931.58	2,076,156	216,172	2,292,328
spacl-core	134	214	181	32	1	85.05	1,350.92	677.55	156,139	28,048	184,187
zip-a-folder	49	101	27	3	71	97.03	500.55	1,132.09	80,056	10,370	90,426
Total	3376	6629	3249	3063	317	-	25,055.52	25,053.42	5,675,688	700,818	6,376,506

Table 59: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-Generic.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	1199	740	458	1	61.80	3,200.91	614.84	943,498	97,375	1,040,873
countries-and-timezones	106	202	179	23	0	88.61	1,501.26	297.33	95,530	21,102	116,632
crawler-url-parser	176	246	143	103	0	58.13	1,664.69	819.62	377,599	38,970	416,569
delta	462	792	660	100	32	87.37	3,159.72	4,099.01	867,614	96,624	964,238
image-downloader	42	88	72	16	0	81.82	430.49	373.69	22,597	8,748	31,345
node-dirty	154	277	162	104	11	62.45	1,555.91	239.75	238,702	32,632	271,334
node-geo-point	140	305	229	76	0	75.08	1,410.92	994.48	309,473	28,709	338,182
node-jsonfile	68	151	49	49	53	67.55	690.59	467.78	54,184	13,996	68,180
plural	153	272	209	62	1	77.21	1,523.33	146.45	258,105	33,232	291,337
pull-stream	351	763	442	266	55	65.14	2,596.08	1,354.67	190,931	73,098	264,029
q	1051	2005	146	1768	91	11.82	5,912.23	13,931.75	2,076,156	216,129	2,292,285
spacl-core	134	215	182	32	1	85.12	1,350.85	689.90	156,139	28,037	184,176
zip-a-folder	49	101	27	3	71	97.03	530.56	1,147.59	80,056	10,370	90,426
Total	3376	6616	3240	3060	316	-	25,527.54	25,176.86	5,670,584	699,022	6,369,606

Table 60: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-full.hb*, systemPrompt: SystemPrompt-Generic.txt, rateLimit: benchmark mode, nrAttempts: 3

A.12 Results for `template-basic-0.0`

Tables 61–65 show the results for 5 experiments with the *codellama-34b-instruct* model at temperature 0.0 using the prompt template of Figure 12 and using the system prompt shown in Figure 7.

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	185	120	65	0	64.86	2,731.54	97.72	893,966	14,460	908,426
countries-and-timezones	106	48	44	4	0	91.67	1,071.22	73.08	89,939	3,087	93,026
crawler-url-parser	176	67	49	18	0	73.13	1,636.67	215.52	359,498	5,557	365,055
delta	462	201	167	28	6	86.07	2,659.93	910.49	820,541	13,472	834,013
image-downloader	42	10	7	3	0	70.00	430.67	65.98	18,348	1,448	19,796
node-dirty	154	44	24	18	2	59.09	1,526.59	39.53	223,071	4,425	227,496
node-geo-point	140	62	54	8	0	87.10	1,411.43	204.76	295,321	4,217	299,538
node-jsonfile	68	22	11	3	8	86.36	690.87	77.82	47,346	1,831	49,177
plural	153	92	78	14	0	84.78	1,521.52	48.57	241,953	5,075	247,028
pull-stream	351	149	88	54	7	63.76	2,382.28	245.31	156,016	9,287	165,303
q	1051	401	38	350	13	12.72	4,158.17	2,697.98	1,970,359	30,059	2,000,418
spacl-core	134	25	23	2	0	92.00	1,351.47	85.42	142,466	4,013	146,479
zip-a-folder	49	20	5	1	14	95.00	500.75	235.31	75,033	1,594	76,627
Total	3376	1326	708	568	50	-	22,073.11	4,997.49	5,333,857	98,525	5,432,382

Table 61: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-basic.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	184	119	65	0	64.67	2,732.79	102.55	893,966	14,472	908,438
countries-and-timezones	106	48	43	5	0	89.58	1,071.27	72.88	89,939	3,113	93,052
crawler-url-parser	176	67	49	18	0	73.13	1,636.61	211.77	359,498	5,557	365,055
delta	462	202	168	28	6	86.14	2,667.43	897.69	820,541	13,458	833,999
image-downloader	42	10	7	3	0	70.00	430.64	65.54	18,348	1,448	19,796
node-dirty	154	44	24	18	2	59.09	1,526.57	38.93	223,071	4,422	227,493
node-geo-point	140	62	54	8	0	87.10	1,411.51	203.69	295,321	4,218	299,539
node-jsonfile	68	22	11	3	8	86.36	690.88	77.42	47,346	1,831	49,177
plural	153	92	78	14	0	84.78	1,521.47	47.92	241,953	5,075	247,028
pull-stream	351	149	88	54	7	63.76	2,382.21	245.17	156,016	9,288	165,304
q	1051	401	38	350	13	12.72	4,159.06	2,700.88	1,970,359	30,059	2,000,418
spacl-core	134	25	23	2	0	92.00	1,351.31	85.36	142,466	4,007	146,473
zip-a-folder	49	20	5	1	14	95.00	500.73	227.68	75,033	1,594	76,627
Total	3376	1326	707	569	50	-	22,082.48	4,977.48	5,333,857	98,542	5,432,399

Table 62: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-basic.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	184	119	65	0	64.67	2,730.84	97.59	893,966	14,461	908,427
countries-and-timezones	106	48	43	5	0	89.58	1,071.16	72.83	89,939	3,113	93,052
crawler-url-parser	176	67	49	18	0	73.13	1,636.68	220.83	359,498	5,576	365,074
delta	462	201	167	28	6	86.07	2,659.86	887.50	820,541	13,473	834,014
image-downloader	42	10	7	3	0	70.00	430.65	65.72	18,348	1,449	19,797
node-dirty	154	43	24	17	2	60.47	1,526.53	37.30	223,071	4,496	227,567
node-geo-point	140	62	54	8	0	87.10	1,411.50	195.42	295,321	4,230	299,551
node-jsonfile	68	22	11	3	8	86.36	690.89	77.90	47,346	1,831	49,177
plural	153	92	78	14	0	84.78	1,521.54	49.08	241,953	5,075	247,028
pull-stream	351	149	88	54	7	63.76	2,382.22	248.75	156,016	9,288	165,304
q	1051	402	38	351	13	12.69	4,158.01	2,694.11	1,970,359	30,070	2,000,429
spacl-core	134	25	23	2	0	92.00	1,351.43	85.92	142,466	4,013	146,479
zip-a-folder	49	20	5	1	14	95.00	500.71	229.75	75,033	1,594	76,627
Total	3376	1325	706	569	50	-	22,072.02	4,962.70	5,333,857	98,669	5,432,526

Table 63: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-basic.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	184	119	65	0	64.67	2,730.22	96.64	893,966	14,459	908,425
countries-and-timezones	106	48	43	5	0	89.58	1,071.13	74.16	89,939	3,112	93,051
crawler-url-parser	176	67	49	18	0	73.13	1,636.64	206.03	359,498	5,556	365,054
delta	462	201	167	28	6	86.07	2,659.88	897.06	820,541	13,471	834,012
image-downloader	42	10	7	3	0	70.00	430.66	65.63	18,348	1,449	19,797
node-dirty	154	44	24	18	2	59.09	1,526.57	38.82	223,071	4,425	227,496
node-geo-point	140	62	54	8	0	87.10	1,411.45	200.23	295,321	4,217	299,538
node-jsonfile	68	22	11	3	8	86.36	690.84	77.61	47,346	1,831	49,177
plural	153	91	78	13	0	85.71	1,556.64	47.70	238,779	5,029	243,808
pull-stream	351	149	88	54	7	63.76	2,382.19	247.28	156,016	9,288	165,304
q	1051	402	38	351	13	12.69	4,156.62	2,695.05	1,970,359	30,071	2,000,430
spacl-core	134	25	23	2	0	92.00	1,351.45	84.84	142,466	4,008	146,474
zip-a-folder	49	20	5	1	14	95.00	500.75	235.51	75,033	1,594	76,627
Total	3376	1325	706	569	50	-	22,105.04	4,966.56	5,330,683	98,510	5,429,193

Table 64: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-basic.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

application	#prompts	#mutants	#killed	#survived	#timeout	mutation score	time (sec)		#tokens		
							LLMorpheus	*StrykerJS*	prompt	completion	total
Complex.js	490	185	120	65	0	64.86	2,730.17	96.67	893,966	14,460	908,426
countries-and-timezones	106	48	44	4	0	91.67	1,071.19	73.83	89,939	3,086	93,025
crawler-url-parser	176	67	49	18	0	73.13	1,636.65	205.98	359,498	5,577	365,075
delta	462	200	166	28	6	86.00	2,659.91	887.25	820,541	13,475	834,016
image-downloader	42	10	7	3	0	70.00	430.71	65.63	18,348	1,449	19,797
node-dirty	154	43	24	17	2	60.47	1,526.60	38.47	223,071	4,500	227,571
node-geo-point	140	62	54	8	0	87.10	1,411.44	197.24	295,321	4,217	299,538
node-jsonfile	68	22	11	3	8	86.36	690.88	78.16	47,346	1,831	49,177
plural	153	92	78	14	0	84.78	1,522.37	47.69	241,953	5,075	247,028
pull-stream	351	149	88	54	7	63.76	2,382.20	245.09	156,016	9,290	165,306
q	1051	402	38	351	13	12.69	4,154.13	2,691.66	1,970,359	30,055	2,000,414
spacl-core	134	25	23	2	0	92.00	1,351.41	83.75	142,466	4,013	146,479
zip-a-folder	49	20	5	1	14	95.00	500.72	232.15	75,033	1,594	76,627
Total	3376	1325	707	568	50	-	22,068.38	4,943.57	5,333,857	98,622	5,432,479

Table 65: Results obtained with LLMorpheus using the following parameters: model: *codellama-34b-instruct*, temperature: 0, MaxTokens: 250, MaxNrPrompts: 2000, template: *template-basic.hb*, systemPrompt: SystemPrompt-MutationTestingExpert.txt, rateLimit: benchmark mode, nrAttempts: 3

````
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 12: A minimal template for requesting a replacement for the PLACEHOLDER.

application	#mutants	#killed	#survived	#timeout	mutation score	time (sec)
Complex.js	1302	763	539	0	58.60	687.01
countries-and-timezones	140	134	6	0	95.71	205.88
crawler-url-parser	226	143	83	0	63.27	739.58
delta	834	686	88	60	89.45	4,200.75
image-downloader	43	28	11	4	74.42	299.72
node-dirty	160	78	56	26	65.00	267.33
node-geo-point	158	98	60	0	62.03	502.91
node-jsonfile	61	31	5	25	91.80	220.17
plural	180	143	37	0	79.44	92.90
pull-stream	474	318	116	40	75.53	834.18
q	1058	68	927	63	12.38	7,516.19
spacl-core	259	239	20	0	92.28	813.55
zip-a-folder	74	38	8	28	89.19	604.29
Total	4969	2767	1956	246	-	16,984.46

Table 66: Results of applying the standard mutation operators of *StrykerJS*.

A.13 StrykerJS standard mutation operators

Table 66 shows the results of running the standard mutation operators of *StrykerJS* on the subject applications.

A.14 String edit distance measurements for different LLMs

Table 67 shows average string similarity for each project, for each of the three LLMs under consideration.

Project	***codellama-34b-instruct***	***codellama-13b-instruct***	***mixtral-8x7b-instruct***
Complex.js	4.27	4.63	8.95
countries-and-timezones	11.13	9.78	13.65
crawler-url-parser	9.50	8.82	12.65
delta	9.55	8.88	14.15
image-downloader	12.67	10.74	14.21
node-dirty	7.53	7.42	12.08
node-geo-point	8.86	8.15	15.27
node-jsonfile	9.73	10.07	10.81
plural	8.14	6.29	10.37
pull-stream	6.72	8.71	9.67
q	8.61	9.71	13.23
spacl-core	9.30	10.84	12.61
zip-a-folder	9.85	12.38	11.61

Table 67: Average string similarity to the original code fragments that they replace, for mutants generated using three LLMs at temperature 0.0.