

Paper1: Bug characteristics in blockchain systems: A large-scale empirical study

Link to paper: <https://ieeexplore.ieee.org/abstract/document/7962390>

Category	Description of category
Memory	Bugs caused by improper handling of memory objects.
Concurrency	Synchronization problems among the concurrent tasks in concurrent programs?, including data races and deadlocks
Performance	Bugs that make a system perform abnormally in terms of responsiveness and stability under a normal workload
Security	Vulnerabilities that cause damage to the software or the information on them, as well as the services they provide
Environment and Configuration	Errors in dependent libraries, underlying operating systems, or non-code that affects functionality
GUI	Graphical user interface errors, including incorrect font, alignment, and button size
Compatibility	A software cannot normally run on a particular CPU architecture, operating system, or Web browser, etc.
Hard Fork	Errors due to radical changes to the protocol that makes previous valid blocks/transactions invalid (or vice-versa).
Semantic	Inconsistencies with the requirements or the programmers' intention that do not belong to the categories above.

Category	Subcategory	Description of subcategory
Semantic bug	Missing Features	A feature is supposed to be but is not implemented.
	Missing Cases	A case in a functionality is not implemented
	Corner Cases	Some boundary cases are considered incorrectly or ignored.
	Wrong Control Flow	The control flow is incorrectly implemented.
	Exception Handling	Does not have proper exception handling
	Input	Input handling or validation is incorrect or ignored.
	Output	Output displays incorrectly.

category	Description of category	Sub category	Subsubcategory
Build prepare	This category includes all problems occurring before starting the build process, e.g., due to (i) fetching data on the repository and (ii) overcoming rate limits on GitHub. While the former are relatively trivial problems rarely discussed, the latter are brought to the attention of developers because depend on the way in which the CI pipeline is configured.	GitHub related error	Rate limit for GitHub APIs
		Problem fetching data	
Build hangs	These kinds of problems occur when the CI server is not able to run the build because of issues such as connection errors, not enough space on the device, or out of memory errors. Noteworthy such failures are often due to (not reproducible) transient-errors.	Connection error	
		No space left on device: Cash issues	
Configuration issues	The configuration of CI might not be trivial in many circumstances, causing itself different kinds of build failures. Such kinds of problems are discussed quite often in the analyzed sample of PRs. Also, as reported by our survey respondents, fixing such problems require complex and effort-prone activities for reconfiguring the CI pipeline.	Travis configuration	Credentials
			CI image configuration
			Emulator related errors
			Travis does not pull external PR
			Build timeouts
		SSH configuration	
		Build configuration	Environment settings
			Deployment errors
			Platform dependent commands
Process related issues	One noticeable case of process-related issue is when a PR fails to build because of unrelated problems currently being handled in a completely different (open) PR.	Maintenance issues	Readability of Travis log
			Excessive configuration file size
		Build failure depends on other PR	
Compilation errors	might be generic issues occurring in production or test code, but also problems due to broken dependencies or to the use of a wrong compiler version	Code compilation	Syntax error
			Dependenc. unavailable/broken
			Compilation fails on a given platform
		Test compilation	Syntax error
			Dependenc. unavailable/broken
Test case failures	may be due to different kinds of problems, including (i) the presence of bugs in the production code and/or in third-party libraries, (ii) the way in which the developers have configured/executed the test suite, e.g., flaky tests or wrong configuration of the test suite to be executed, (iii) a misalignment between production and test code, and (iv)	bugs	Bug in external libraries
			Application logic errors
			Integration bugs
			Bug manifests in specific environment
		Test/code alignment	Uncommit. code
			Test case not updated to reflect code changes
			Functional. tested but not implement

Static Analysis Checks	missing/wrong dependencies only identified at runtime (i.e., not discovered by the compiler, if any).	Test case issues	Flaky tests
			Test case config
			Errors in test cases
	These include (i) styling and formatting issues in the documentation, (ii) licensing issues, (iii) malformed commit messages, and (iv) code checks violations e.g., code smells, potential vulnerabilities or bugs, and code style issues.	Depend. issues	Depend. not found at runtime
		Process checks	Validation of commit messages
		Document. Style/formatting	
		Licensing	Missing contributor license agreement
			Licensing header warnings
		Code checks	Syntax errors
			Candidate bugs (e.g., miss null check)
			Style / formatting / indentation
			Code smells
			Vulnerabil. issues
			Coding standards violated

Paper3: Taxonomy of Real Faults in Deep Learning Systems

Link to paper: <https://dl.acm.org/doi/abs/10.1145/3377811.3380395>

Category	Description of category	Subcategory
API	This part of the taxonomy represents a broad category of problems arising from framework's API usage. The most frequent is usage, which means that a developer is using an API wrong API in a way that does not conform to the logic set out by developers of the framework. Another illustrating example could be a missing or wrongly positioned API call.	Deprecated API
		wrong usage of image decoding API
		wrong usage of placeholder restoration API
		Missing argument scoping
		Wrong position of data shuffle operation
		missing global variables initialisation
		wrong API usage
		missing API call
		Wrong reference to operational graph

Paper 4: Repairing Deep Neural Networks: Fix Patterns and Challenges

Link to paper: <https://ieeexplore.ieee.org/abstract/document/9284050>

Category	Description of category
Versioning	Adapt the code to the new version of the library
API contract	fix API compositions so that the output of an API meets the preconditions of another API

Category	Description of category
Dimension mismatch	We put a bug into this category if it is caused by dimension mismatch in tensor computations and transformations.
Type confusion	Type confusions are caused by the mismatches of types.
Processing	We put a bug into this category, if it is caused by wrong assignment or initialization of variables, wrong formats of variables, or other wrong usages that are related to data processing.
Inconsistency	We put a bug into this category, if it is caused by incompatibility due to API change or version update.
Algorithm	We put a bug into the algorithm category, if it is caused by wrong logic in algorithms.
Corner case	We put a bug into the this category, if it is caused by erroneous handling of corner cases.
Logic error	We put a bug into this category, if mistakes happen in the logic of a program. A logic error can be an incorrect program flow or a wrong order of actions.
Configuration error	We put a bug into this category, if it is caused by wrong configuration
Referenced types error	We put a bug into this category, if it is caused by missing or adding unnecessary include or import statements.
Memory	We put a bug into the memory category, if it is caused by incorrect memory usages.
Concurrency	We put a bug into this category, if it is caused by synchronization problems

Paper6: An Exploratory Characterization of Bugs in COVID-19 Software Projects

Link to paper: <https://arxiv.org/abs/2006.00586>

Category	Description of category
Algorithm	This category corresponds to bugs when implementation of an algorithm does not follow expected behavior
Data	This category corresponds to bugs that occur during mining and storage of data.
Dependency	This category corresponds to bugs that occur when execution of the software is dependent on a software artifact that is either missing or incorrectly specified.
Documentation	This category corresponds to bugs that occur when incorrect and/or incomplete information is specified in release notes, maintenance notes, and documentation les, such as README les.
Performance	This category corresponds to bugs that cause performance discrepancies for the software. Performance bugs are manifested in slow response of the web or mobile app.
Security	This category corresponds to bugs that violate confidentiality, integrity, or availability for the software.
Syntax	This category corresponds to bugs related with the syntax of the programming languages used to develop the software.
UI	UI bugs include navigation-related bugs on web pages, bugs related to accessibility, displaying incorrect images, links, and color, and responsiveness.

Category	Subcategory	Description of subcategory
Algorithm	bugs related to statistical modeling algorithms	This bug happens where statistical modeling results are incorrect due to incorrect assumptions and/or implementations,
	bugs related to incorrect logic implemented in the software	-

Category	Subcategory	Description of subcategory
Data	storage	bugs that occur while storing data in a database
	mining	bugs that occur while retrieving data from data APIs
	location	bugs where location information in stored data is incorrect
	time series	bugs that correspond to missing data for a certain time period

Paper7: Bug Characteristics in Open Source Software

Link to paper: <https://link.springer.com/article/10.1007/s10664-013-9258-8>

Dimension	Category	Description of category
Root Cause	Memory	Bugs caused by improper handling Mem of memory objects.
	Concurrency	Synchronization problems among the concurrent tasks in concurrent programs” [55], including data races and deadlocks.
	Semantic	Inconsistencies with the requirements or the programmers’ intention that do not belong to the categories above.
Impact	Hang	Program keeps running but does not Hang respond.
	crash	Program halts abnormally.
	Data Corruption	Mistakenly change user data.
	Performance Degradation	Functions correctly but runs/responds slowly.
	Incorrect Functionality	Not behaving as expected.
	Other	Bugs that cause other impacts.
Software Component. (Mozilla & Apache)	Core	Bugs related to core functionality Core implementations.
	GUI	Bugs related to graphical user inter- GUI faces.
	Network	Bugs related to network environment and network communication.
	I/O	Bugs related to I/O handling.
OS Component (Linux)	Drivers	Bugs related to device drivers.
	Core	Bugs in the kernel directory ‘mm’, ‘kernel’, and ‘include’.
	Network	Bugs related to network environment and network communication.
	File system	Bugs related to file system.
	Architecture	Bugs related to hardware architecture.

Category	Subcategory	Description of subcategory
Memory	Memory Leak	Failures to release unused memory.
	Uninitialized Memory Read	Read memory data before it is initialized.
	Dangling Pointer	Pointers still keep freed memory ad- dresses.
	NULL Pointer Dereference	Dereference of a null pointer.
	Overflow	Illegal access beyond a buffer boundary.
	Double Free	One memory location is freed twice.
	Other	Other memory bugs.
Semantic	Missing Features	A feature is supposed to be but is not implemented.
	Missing Cases	A case in a functionality is not implemented.
	Corner Cases	Some boundary cases are considered incorrectly or ignored.
	Wrong Control Flow	The control flow is incorrectly implemented.
	Exception Handling	Does not have proper exception handling.
	Processing	Processing such as evaluation of ex- pressions and equations is incorrect.
	Typo	Typographical mistakes.
	Other Wrong Functionality Implementation	Any other semantic bug that does not meet the design requirement.