Microwave Journal

Significant Test Time Reduction and Equipment Utilization in 5G RF Production Testing

Sascha Laumann
Rohde & Schwarz
Munich, Germany

5G base station and infrastructure manufacturing is currently one of the hottest areas in communication technology as deployments are accelerating. Success among competitors is generally measured by traditional metrics such as cost of production and integration, productivity and time-to-market. Yet, delivering products in large quantities becomes more important as regulatory bodies start to pave the way to mass installations and delivery bottlenecks may cause significant loss of business. The question is, "How can we manufacture products faster, given the known technical and financial conditions?" There are probably many answers to this question, but in the context of testing of units in production, the answer is likely: "In a given period of time, increase the number of units properly produced and successfully tested, according to the specification."

Improving the 'speed of test' has been an ongoing subject since the dawn of testing endeavors. The ever-increasing frequency of product releases, shorter time-to-market cycles and budgetary constraints call for new ideas on how to improve one of the fundamental metrics particularly important in production environments: test throughput.

Test throughput is essentially a reciprocal figure to the speed-of-test. Besides indicating how fast a single test can be executed, test throughput can additionally address cases where multiple tests are accomplished in parallel. Achieving higher throughput almost always demands some degree of concurrent operation, particularly in cases where speeding up a single test is either technically not possible or too expensive. Forms of parallelization may range from simple duplication to a configuration of independently executed tasks and subtasks that allow fully asynchronous test execution. The speed of test is either difficult or impossible to measure in such a highly parallelized testing environment. In contrast, test throughput remains a valid metric with an increasing significance.

Parallelization of tests must be well thought out; higher throughput rates must not lead to deterioration of quality-related metrics. The more tests that are being executed asynchronously, the more care needs to be put into ensuring that test results remain valid and consistent. Solutions designed for sequential execution are not necessarily suitable for parallel operation. Luckily, technological advancements in the domain of software architecture render such a shift in testing paradigms possible without sacrificing test quality and reproducibility. Solutions supporting parallelized operation are already commercially available.

PROBLEM DESCRIPTION

In the context of emerging 5G infrastructure business, ensuring a high production throughput rate is crucial for success and competitiveness. Clearly, test throughput is one of many factors that influences the production throughput; yet, it is comparatively easy to improve this metric in specific environments if few prerequisites are met.

Traditionally, the execution of 5G production tests is done close to or in conjunction with the product assembly line. Once the product is assembled, a manual or automated set of a multitude of tests, including an RF test, is performed. After test execution and results retrieval, the system decides if the device-under-test (DUT) has fulfilled the testing requirements or not. Lastly, the tested product is sorted accordingly. This is a typical representative of a sequential and synchronous test execution; the following task waits

until the previous task has finished, and its result is available. Although RF tests generally do not have a large impact on the overall performance, the complexity of the analysis of the test data increases with tighter testing conditions and thus may render testing throughput worth improving.

While the sequential execution in its simplicity is acceptable for many applications, it also bears one potentially significant impediment: it generally does not scale well. Here, increasing throughput calls for either the installation of one or multiple duplicates of the production and testing equipment, which is expensive, or reduction of testing complexity. The latter may result in a poor overall yield or higher failure rates of devices in operation.

Coming back to the 5G infrastructure manufacturing and testing use case, the 3GPP TS 38.141 specification requires multiple metrics to be evaluated. Error Vector Magnitude (EVM), Operating Band Unwanted Emission (OBUE) / Spectral Emission Mask (SEM) and Adjacent Channel Leakage Power Ratio (ACLR) are the commonly used measurements to specify if a DUT can pass a test or not. All tests are jointly performed in several frequency and output power ranges; the depth and number of tests depend on the product category and its specifications.

Executing a typical production line test on a 5G macrocell base station can take up to several minutes to conduct. Speeding up this process can significantly improve the throughput rate in production. Repetitive characterization test may require multiple hours of testing scenarios. It is safe to say that by analyzing the complex process of production and testing of devices, one may find a few tasks or subtasks that are executed sub-optimally. Depending on the product and the testing specification the device needs to be tested against, this could be either test data acquisition time or analysis time. Clearly, upcoming higher requirements in 5G FR2 testing (and beyond) will likely result in more complex analyses and thus longer processing times.

BACKGROUND

The concept of parallelization in the context of technology or productivity optimization is nothing new; in the last few decades, the development has been boosted by an increasing number of available parallel processing units (multicore CPUs) and the simplified access to those resources from both programming and operational standpoints. Asynchrony or asynchronous operation has gained more attention in recent years due to the rise of distributed systems. It forms a fundamental construct allowing non-deterministic operation mainly found in communication technology.

The rise of parallelization as a technology aspect paved the way to new problem-solving techniques: instead of focusing on improving individual execution performance, parallelization renders new levels of scalability. By looking at cloud-based technologies available nowadays, it should be clear that speed performance is simpler to implement within a framework allowing parallel processing.

The demands from an economic perspective arise as well; more than ever, it has become important to optimize the cost of usage of assets, such as test equipment, especially in cases where substantial investment is required; metrics such as Return-on-Investment (ROI) or Total Cost of Ownership (TCO) have gained importance. Lastly, new business and cost models such as Operational Expenditures (OPEX) asks for lower upfront spending and thus call for different commercial solutions. This, in turn, increases the importance of a qualified equipment usage metric which allows usage-based business models (such as pay-per-use) to be attractive for both the solution providers and their customers.

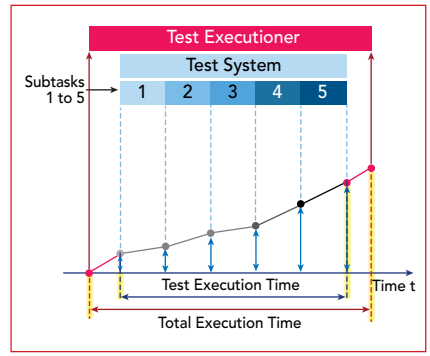

Fig. 1 **Sequential test scenario.**

APPROACHES

Sequential Operation

Consider the following scenario: a DUT requires a specific set of tests to be performed upon it to ensure its functionality. Let us imagine each individual test sequence consists of five subtasks: test preparation and equipment setup, stimulus generation and data capture, test data retrieval (transmission), data processing/analysis, and test result delivery as shown in ***Figure 1***. The test sequence is repeated either with a different set of testing parameters or through provisioning of another DUT.

In a sequential approach, subtasks are executed one after the other since the degree of dependency between two adjacent subtasks is high; it makes no sense to start data acquisition before test equipment has been instructed what exactly to capture. Equivalently, analysis of test data cannot happen before the data has been successfully transmitted.

Each subtask has its own execution time. In many cases it is a largely consistent number across test sequences, but it may exhibit some degree of uncertainty and jitter (e.g. in cases of non-deterministic data transmission). In sequential testing, the overall test execution is the sum of individual task execution times. Thus, improving the overall test duration is only possible by accelerating any respective subtask within that test sequence. Some 'expensive' subtasks may be subject to optimization but in general there will always be technical (e.g. CPU clock speed) or procedural (e.g. handling or settling time) boundaries which do not allow further execution speedup.

By looking at ***Figure 2***, one can easily identify a further implication: test equipment used in subtasks 1 and 2 returns to idle mode when the respective subtasks have finished and becomes operational only when next test sequence is executed. The period between the two subsequent test sequences determines the utilization ratio; the shorter the period is, the higher the ratio. This is generally favorable; yet, improving this number in pure sequential operation is equally challenging.

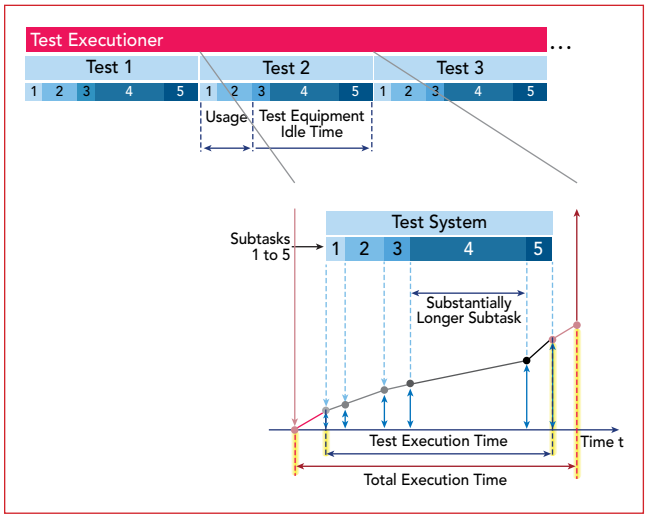

Fig. 2 **General overview of a test sequence.**

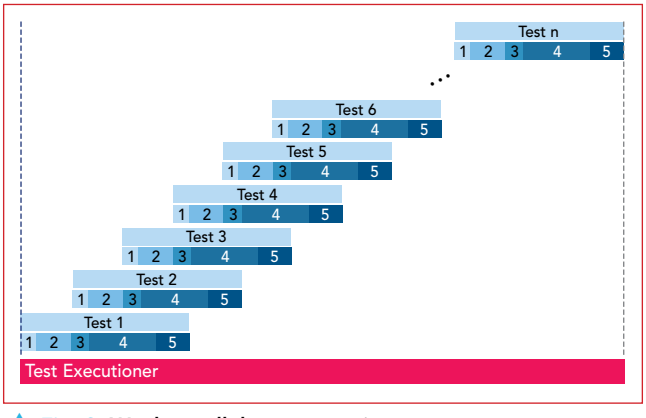

Fig. 3 **Weak parallel test execution.**

Weakly Parallelized Operation

An alternative to a sequential implementation is to consider parallel execution of subtasks, or a set thereof, wherever possible. This may be an interesting approach in cases where individual subtasks require significant time for completion with a certain degree of decoupling rendered possible; any such 'expensive' subtask that bears a lower degree of dependency on adjacent tasks is potentially suited for migration to parallel operation.

Consider again the same test sequence of five subtasks needed to complete a test of a single DUT. In subtask 4, data processing and analysis, execution is likely to be computationally expensive, requiring a substantial amount of time to accomplish the task. This is generally found in 5G NR applications given the complex yet tight 3GPP specification requirements the production tests must fulfill. Clearly, parallelization of the execution of this and the following subtasks results in higher throughput.

In most cases, isolating a processing task is possible; subtasks of test preparation, stimulus generation and test data capture do not need to wait for the analysis to be finished and thus can be restarted. Depending on the parallelization capabilities of the processing unit, this process could be set up such that it allows complete decoupling of the expensive subtask, allowing other tasks to be performed much quicker.

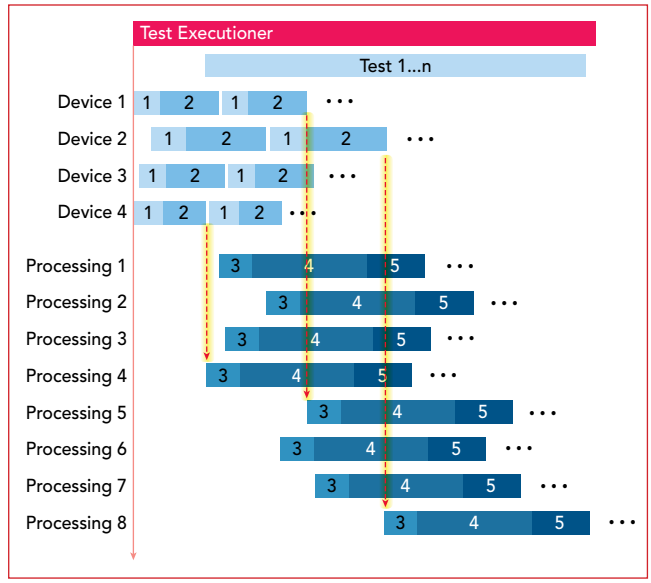

Fig. 4 **Strongly parallelized operation with decoupled subtasks. Arrows depict the arbitrary selection of processing units for the handling of incoming capture data.**

This process can be named weak (or incomplete) parallelization since not all subtasks are eligible for parallel execution. As depicted in ***Figure 3***, subtask 2 (capture) must wait for subtask 1 (preparation) to complete. However, once subtask 2 has completed, the subtask 3 (data transmission) can be executed while subtask 1 of the next test iteration is started immediately. In contrast to the sequential execution, the interval where testing equipment idles is also minimized or even eliminated given the immediate reuse for next test.

Strongly Parallelized Operation

In the previous case, performance is limited by two factors: the number of parallel processing units available and the speed of execution of subtasks 1 and 2. While enabling additional parallel processing capabilities nowadays is a relatively simple technical modification or upgrade, accelerating capture time is rather difficult—unless parallelization of the capturing process becomes viable. This is generally done by adding more capture devices to the setup. In that case, the test execution may be considered a strongly (or complete) parallelized operation; this is the case when a high degree of decoupling of subtasks is given.

Consider the ideal case depicted in ***Figure 4***. There are 4 devices available independently delivering capture data, whereas the analysis of the test data can be done by 8 parallel processing units.

In such a case, any arbitrary processing unit that is 'free' at a given time can fetch and handle the available data coming from any of the devices. The next available unit does the same, resulting in an asynchronous operation. In addition, each device may have a specific capture duration, resulting in slower or later provisioning of the capture data. This has no significant impact on the overall performance since the processing units are rather agnostic towards 'who delivers data when'; they simply process any capture data available independently of its source.

While asynchrony is difficult to depict, ***Figure 5*** shows

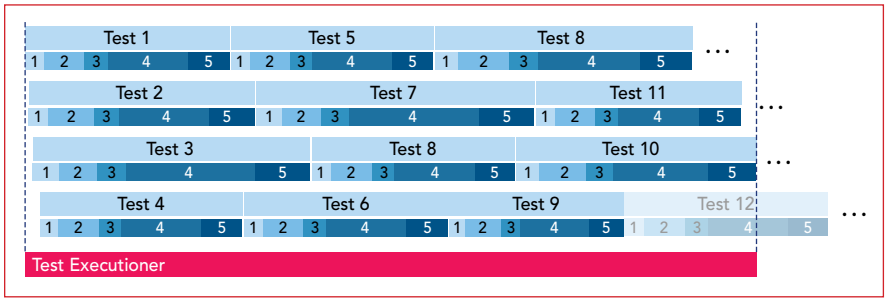

Fig. 5 **Strongly parallelized operation with 'sorted' operators.**

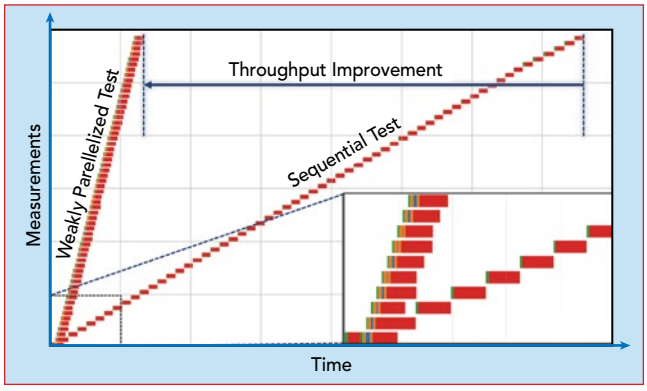

Fig. 6 **Comparison of sequential (right) and weakly parallelized (left) test execution. Lower right shows the first few steps.**

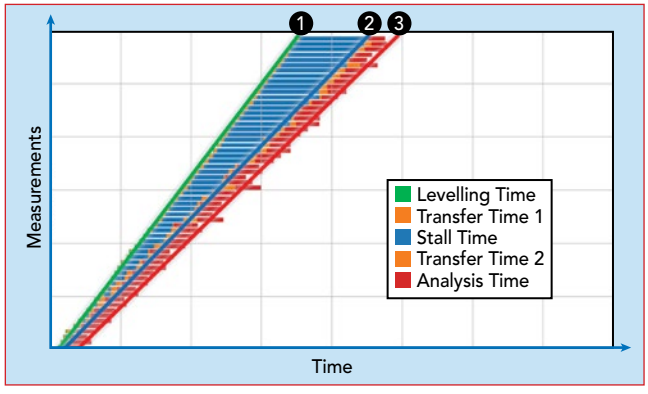

Fig. 7 **Throughput subtask segmentation and the respective effect on the overall systems' performance.**

a qualitatively similar graph where the aforementioned processing units are assigned to each devices' 'capture stream'; by looking at the number of executed test iterations, one sees that the throughput has significantly improved when compared against both previous cases. However, this does not necessarily mean that the processing of the capture data coming from a specific device is also handled by a specific processing unit.

It should be mentioned that parallel data delivery is not always possible: in cases where single DUTs are tested, it may not be possible to add multiple capture devices (such as spectrum analyzers) due to complex RF connectivity or instrumentation.

TEST RESULTS

To verify the benefits of parallel test execution, a realistic scenario often found in 5G applications was set up and executed in multiple test configurations. The setup allowed two modes, a sequential and a weakly parallelized operation. The specific test system consisted of the signal generator (R&S SMBV100B), spectrum analyzer (R&S FSVA3000) and R&S server-based testing (SBT) unit, a processing platform capable of processing 16 data sets in parallel. The SBT was configured so that EVM, ACLR and SEM metrics were calculated in each test iteration. The generator was set up in such a way that the stimulus output power was stepped across 59 different levels. The analyzer was set up so that data was captured and transported as I/Q data files. The connectivity between devices was given through a local gigabit Ethernet with no special hardware used to interface the participating devices. All calculations were compared to ensure validity of the results.

Figure 6 depicts the qualitative results of the two scenarios. It confirms that the performance in a sequential operation (shown in Figure 6 on the right) is limited by individual test step execution time. The lower right diagram is a zoom-in into the first few test steps executed, depicting the difference between sequential (flat) and parallel (steep) approaches. A breakdown into individual subtasks within each step is irrelevant since all test steps have a largely consistent duration. Total test time for all 59 steps is simply the sum of individual test execution periods.

The left part of the diagram depicts the weakly parallelized operation with a single analyzer being operated as the sole capture data source. The zoomed-in view on the lower right shows the first 10 iterations of the parallel and 8 iterations of the sequential test. It illustrates how each individual test step, divided in subtasks, is being executed. The parallel execution of analysis task is reflected by the overlapping red bars. One can also notice that the single-device operation results in the test preparation and data capture subtasks effectively determining the systems performance. This is a result of having both fast data transfer capabilities and enough parallel processing units available.

By quickly inspecting the difference between the two scenarios in total test execution time in Figure 6 (dashed vertical lines), one clearly sees that the performance gain in the second scenario is substantial; the weakly parallelized setup performs almost six times faster than the sequential test. This can be considered a significant improvement for any scenarios requiring acceleration of repetitive test execution. In contrast, the ratio of used equipment is almost at 100% for the duration of the test. The ratio in sequential execution is lower than 10% on average for the complete test sequence.

A general view at the system's performance can be seen in ***Figure 7***. Here, a system was purposely configured with a low number of parallel processing units and used to run the same test. The system's throughput performance data was recorded, with the separation of individual phases in each test step. First phase (green) is the analyzer preparation time where input levelling is performed; second phase (orange) is the data capture and transfer time to a central-

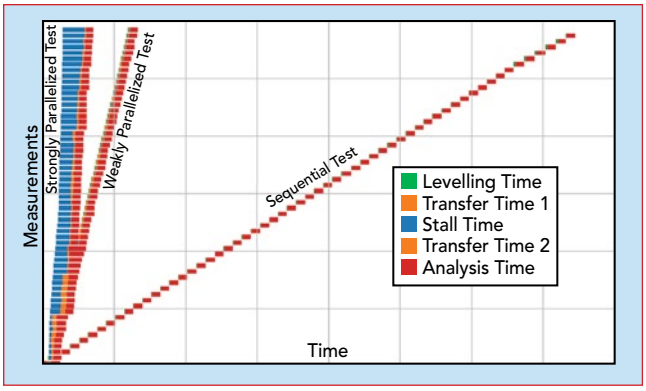

Fig. 8 Performance of strongly and weakly parallelized tests vs sequential test.

ized buffer. Third phase (blue) is a file server that is utilized as a buffer. Fourth phase is the data transfer to the processing unit, whereas phase 5 (red) is the actual processing of the capture.

By observing the diagram, one can identify three segments that have an effect the performance:

Segment 1, identified by the green gradient No 1 in Figure 7, defines how fast data is delivered. Speed improvement is done by both acceleration of data delivery and parallelization of data sources where applicable. This is a large factor to the speed of the system but is often limited by the test setup or physical boundaries.

Segment 2, which is the area between gradients 1 and 2, reflects how much a system must wait (or buffer) due to a lack of parallel processing capabilities. In cases where enough processing units are available at the system's disposal, the blue area is minimized. This segment can be considered as both the biggest contributor to overall performance and the easiest to tune, if the underlying platform supports such scalability. In this test case, SBT is specifically designed to support an arbitrary number of processing units running on single hardware unit (e.g. server) or across multiple units.

Segment 3, the red area between gradients 2 and 3, defines how fast individual processing units can execute the analysis task. Optimizing a single processing time has a comparatively small effect on the overall system's performance and can be generally deemed as an expensive tuning factor.

In additional tests, strong parallelization was approximated by a simulation of an arbitrary number of capture devices, effectively resulting in an almost-immediate availability of capture data. The latter use case functions as a setup allowing an estimation of best possible performance of the processing platform. It shall be considered a borderline, yet real scenario, where enough data capturing devices is available. ***Figure 8*** shows the strongly parallelized test simulation in comparison with the other two scenarios.

CONCLUSION AND OUTLOOK

As expected, the strongly parallelized test performs the best. Due to the system's configuration of 16 parallel processing units, the blue segment shows where buffering takes place since data is almost instantaneously available for processing. Adding further processing units would have minimized the overall throughput time to a minimum, effectively rendering improvement factors of 10 and more versus sequential testing.

Note that data-transfer phases in both Figures 7 and 8 (in orange) have a little impact on the system's performance despite some variance in the duration. This is a typical observation when a non-deterministic transfer protocol (such as TCP over Ethernet) is used. Yet, the asynchronous, non-blocking operation can support such an execution mode.

The asynchrony is enabled through the utilization of system components that are commonly used in scalable IT systems. The recent rise of cloud-based systems, both public and on-premises, render the usage of such building blocks possible for system designs beyond the original use case. Here, the SBT solution tested is built around such scalable components. Yet, it is a purely localized deployment with no external connectivity requirements.

The benefits of such a solution with a focus on throughput improvements are manifold: apart from the discussed faster overall test speed or throughput, parallelization renders better equipment utilization ratios possible and allows a high ROI. Eventually, new usage-based commercial models become more attractive in cases where lower upfront investment in high-value equipment is desired.■