Test Component Assignment and Scheduling in a Load Testing Environment

Ferenc Bozóki and Levente Erős

When the construction of a network component has finished the testing phase begins, which is just as important as the implementation itself. During testing the conformance test has to be executed first. The conformance test examines whether the System Under Test (SUT) corresponds to its specifications, that is whether it generates the appropriate output and transfers to the appropriate state when induced by some kind of an input. Once the SUT has passed the conformance test, the load test begins, which stresses the SUT by the maximal load it has to be able to bear with in its latter real-life environment, while examining its behavior in this extreme situation.

However, for generating this high load we far don't have as much host computers in the test environment as the SUT will have to serve once it gets into its real-life environment. This means each host in the test environment has to emulate more hosts of the real-life environment. To carry this out these real-life hosts are represented by software entities, the so-called test components in the test environment, and each one of these test components is assigned to a host in the test environment. The question now is how these assigments have to be made. Main objectives of most of the task assignment algorithms developed so far are to maximize the number of running tasks (sometimes even overloading the hosts is permitted) and to minimize the communication costs between the tasks [1,2]. In our case the tasks (test components) don't communicate with each other and in order to simulate the real-life environment faithfully, overloading testing hosts is prohibited. This implies that a fully new approach is needed. In this paper we present a heuristic test component assignment algorithm that takes the above mentioned aspects into consideration and that is specialized for handling typical load testing traffic patterns.

The other problem we introduce a possible solution for is scheduling in a load testing environment. In the case of conformance tests where the test environment emulates a single user, scheduling can be derived from real time scheduling. Contrarily, when dealing with load tests the test environment acts as a big amount of users or hosts, and as mentioned above a testing host has to run many test components representing many users parallelly. This leads us to another problem, namely the problem of scheduling these test components inside a testing host. The essence of the problem is that if each one of the many test components assigned to a testing host run on separate threads just like in the classic case, context switching takes a dominant part of the CPU load. The solution we developed for solving the problem of the huge overload caused by context switching introduces the concept of virtual threads. A virtual thread plays the same role as a regular thread in the classical approach, that is, exactly one test component is assigned to a virtual thread. These virtual threads are then branched together into regular threads. By significantly reducing the number of regular threads this way, the CPU load used for context switching can be reduced to a near-zero level.

As our simulations have shown handling these two problems by the solutions we present in this paper results in a significant raise in the efficiency of executing load tests.

References

[1] M. Kafil and I. Ahmad. Optimal Task Assignment in Heterogeneous Distributed Computing Systems, *Heterogeneous Computing Workshop, 1997. (HCW '97) Proceedings, p135-146. 1997.*

[2] Task Assignment and Transaction Clustering Heuristics for Distributed System, *Information Sciences vol. 97. Issue 1-2, p199-219. 1997.*