AFP - 3-SAT server - Report

Koen Timmermans & Maximo Garcia Martinez

1. Structure of the project

The project has 4 files in total:

1. `sat_server.erl`: It starts the OTP server using the supervisor behavior[1]. As it says in the exercise, the server starts with the function `start_link()`. For each new connection, it will create a listener using `listener.erl` which will handle the different requests of the client. Another of the requirements was not to accept more than 8 children; This module will keep track on how many children it has and it won't accept more clients if the maximum number of children is reached.
2. `listener.erl`: After it is decided whether or not a new client can connect, this module will handle all the requests and send back the response. For each request, it will parse the raw data and convert it to an Erlang data structure. Then, it will spawn a worker using the `sat` module. Its method of communication with its children is using the message passing system. It will continue listening for other clients until it receives a message from a worker, which in that case, it will send message given by the worker to the client who made the request. After that, the worker will be killed.

   This module will handle the connection with the clients with different actions:

   - If a client is waiting for a response, every 7.5 seconds, this module will send a message to the client saying "trying".
   - If a client closes the connection, the worker associated with that client will be stopped.
   - If a client ignores a request, it will send "ignored" to the client.
   - If a client request abort its request, then the worker will be killed and the connection will be closed.
3. `sat.erl`: It provides the algorithm to solve the 3-SAT problem. We are using a brute force algorithm, which it takes all of the possible options and check against the expression given by the client. It will return the first option that is valid. If no solution is found, it will return `unsat`.
4. `instances.txt`: It provides some cases to test the 3-SAT algorithm.

2. How did we divide the work?

We have done the projects separately, but we checked each others work. This project has been made by Maximo.

3. Extra requirements fulfilled.

We think all the normal and extra requirements are fulfilled.

4. Tests.

We have added some unit tests cases for testing the 3-SAT algorithm. The unit tests can be found at `instances.txt`. The simpler tests have been written by us, whereas, the complex tests are from this webpage[2].

5. Material that we have used.

~~We have used the official documentation~~ as well as some tutorials[3] for the creation of the OTP server.

[1] `https://erlang.org/doc/man/supervisor.html`

[2] `https://www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/RND3SAT/uf20-91.tar.gz`

[3] `https://learnyousomeerlang.com/`