Zadanie 2 PW

Marta Nowakowska 385914

22 stycznia 2018

1 PODSTAWOWE ZASADY KOMUNIKACJI:

Validator funkcjonuje jako serwer otrzymujący od testerów słowa do sprawdzenia przez automat, który jest zapisany w formie struktury w pamięci Validatora. Validator uruchamia po otrzymaniu słowa walidację poprzez przekazanie automatu i słowa do procesu run, który w pojedynczym wykonaniu sprawdza jedno słowo i następnie się zakańcza.
Tester dzieli obowiązki pomiędzy parenta i childa po wykonaniu fork. Child ma za zadanie sczytywanie z stdin wszystkich wiadomości, które ma wysłać i wysyłanie ich do validatora, znając PID swojego rodzica. Parent z kolei odbiera wiadomości od Validatora na "spersonalizowanej" kolejce wiadomości, której nazwa zawiera w sobie PID (link_v_t_pid) testera. Validator odbiera wiadomości na swojej "masowej" kolejce wiadomości, gdzie przez masowość rozumiemy otrzymywanie tam wiadomości zarówno od procesów run jak i tester. Validator wysyła wyniki procesów run do testerów. Poinformuje ich także przy kończeniu swojej pracy ile mają wiadomości do odebrania (tj. niezależnie od tego ile już ich odebrali).
Na każdej gałęzi sprawdzania zgodności słowa z automatem uruchamiany jest fork dla danego runa. Walidacja jest więc zaimplementowana współbieżnie.

2 FORMAT KOMUNIKACJI:

v $->$ r
jedna kolejka v i r do pisania komunikatów do r-a
"[a-z] PIDt"
r $->$ v
"PIDt czyOK [a-z]" czyOK to 0 albo 1

v $->$ t
"ok [a-z]" ok: T/N
"! rcd" koniec
t $->$ v
"! PID"

"[a-z] PID" słowo+pid

3 KWESTIE WARTE PORUSZENIA:

Validator nie musi czekać na runa (wykonywać funkcji wait) dzięki zastosowaniu prctl.