Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Modell-alapú tesztelési algoritmusok implementálása

Verbőczy Kristóf I. évf, (MSc) mérnökinformatikus szakos hallgató
Konzulens: dr. Micskei Zoltán egyetemi docens, MIT
Kritikus rendszerek specializáció
Önálló laboratórium 1. összefoglaló
2016/17. II. félév

A szoftverek egyre nagyobb szerepet kapnak az életünkben, ezért nyilvánvalóan elvárjuk tőlük, hogy megfelelően működjenek. Ennek ellenőrzésére szolgáló technika a tesztelés, ahol tulajdonképpen a szoftver elvárt viselkedést hasonlítjuk össze a tényleges működésével.

A modell-alapú tesztelés esetén a rendszer valamely modellje alapján származtatjuk a teszteseteket. Ennek az előnyei a hagyományos teszteléssel szemben, hogy már a fejlesztés korai fázisában képesek vagyunk ellenőrizni a működést. Továbbá a modellekből könnyedén származtathatók különféle teszteseteket. A féléves munkám során ilyen tesztgeneráló algoritmusok implementációjával foglalkoztam.

A modellek az esetek döntő többségében felfoghatók gráfokként, így ezeket a gráfokat bejárva származtathatunk belőlük teszteseteket. A munkám során a modelleket, mint véges állapotgépeket értelmeztem, az algoritmusokat Java nyelven implementáltam.

Az algoritmusokat két csoportba osztanám. Az első csoportba az egyszerűbb bejárásokat tenném, ahol a cél lehet, hogy meglátogassuk a gráf összes csúcsát vagy élét. Ilyen például az Euler-körséta keresése a gráfban, de egy saját és egy véletlenszerű algoritmust is implementáltam. A másik csoportba két bonyolultabb algoritmus sorolnék. Ezek a „homing sequence” és a „synchronizing sequence”. Mindkettő esetében az alapfeltevés az, hogy nem tudjuk, hogy hol vagyunk a gráfban, de egy megfelelő bemeneti szekvencia végrehajtása esetén ismert lesz a végső állapot.

A féléves munka további részét képezte az algoritmusok implementációjának ellenőrzése. Ehhez a JUnit keretrendszer paraméteres tesztjét használtam fel, és különböző segédosztályokat kellett létrehoznom, amelyek képesek leellenőrizni az algoritmus futásának az eredményét. A tesztelésnél olyan esetekre is vizsgáltam az implementációt, amelyekre az algoritmus nem működőképes, például nem összefüggő gráfban Euler-körsétát találni. Ennek a célja az volt, hogy megvizsgáljam, hogy az algoritmus vajon örökké fog-e futni, vagy kifut-e a memóriából, vagy képes-e jelezni, hogy nem tud megoldást találni ez esetben.

Kiemelném még a munkám során használt SonarQube nevezteű statikus analízis eszközt, amely a programkód minőségét hivatott ellenőrizni. Véleményem szerint sokat fejlődtem a használata által. Továbbá a Jacoco eszközzel együttműködve segített közel teljes kódfedettséget elérni.