Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Autonóm robotok működésének futásidőbeli ellenőrzése

Vas Ádám (QTVG7Y), IV. évf, (MSc) mérnök inf. szakos hallgató
Konzulens: dr. Majzik István docens, MIT
Szolgáltatásbiztos rendszertervezés szakirány
Önálló laboratórium 1 összefoglaló
2013/14. II. félév

A mai világban rendkívüli szerephez jut a biztonság kérdésével foglalkozó tudományok munkássága. A működés közbeni hibák detektálásának egyik módszere a futásidőbeli verifikáció. Ennek lényege, hogy formalizált követelmények alapján forráskód szintézissel olyan monitorokat generálunk, amelyek folyamatosan követik a robot vezérlőjének belső működését, és a specifikált követelményektől eltérő működés esetén megindítják a beavatkozást (pl. a biztonságos leállást). A feladat az, hogy egy inputként kapott LTL kifejezéshez C++ környezetben készüljön egy olyan eszköz, ami képes C nyelvű monitor forráskódjának generálására, ami később felhasználható ROS (Robot Operation System) modulokkal való kommunikációra.

Az alapötlet, hogy olyan kiértékelő blokkokat hozzon létre a program, ami olyan függvényeket reprezentál, aminek a változóit bemenetként kapja meg és értékeli ki. A blokknak három interfésze van, amik rendre a következőket jelentik:

- Felső interfész: adott kiértékelő blokk eredményei
- Bal interfész: lokális input
- Alsó interfész: következő kiértékelő blokk eredményei

Látható, hogy az egyes blokkok összekapcsolódnak. A feladat megoldása a következő lépéseken keresztül valósul meg, ezek dőlten betűvel vannak szedve:

A kifejezés *validációjának* lényege, hogy meghatározott szabályok szerint vizsgálja a függvény, hogy a következő karakter szerepelhet a helyén vagy sem.

A *kifejezésfa felépítése* során a kapott kifejezést fastruktúrába rendezi, ezzel könnyebben alakíthatóvá válik a bemenet. Ezalatt már végez a program operátormanipulációkat, hogy minél kevesebb operátorral kelljen a feldolgozást elvégezni.

A *connection normal form-ra való hozáshoz* szükségesek egyéb operátormanipulációk. Lényege, hogy az U operátor csak az X után jelenjen meg, mivel az X operátor jelöli a következő kiértékelő blokk felső kimenetét. A normál formára hozás célja, hogy lokálisan és a jövőben kiértékelendő kifejezéseket egyértelműen szétválassza.

Izomorfia vizsgálat azt vizsgálja, hogy egy X operátortól kezdődő részfában található elemek sorrendje megegyezik e bármely más ugyanilyen operátorral kezdődő részfa elemeivel. Ez alapján megmondható hány típusú kiértékelő blokkra lesz szükség a program futása során.

A kiértékelő blokk típusok azonosítása során kialakul, hogy egy kiértékelő blokk lokális bemeneteit milyen atomi kifejezések, illetve milyen felső kimenetek és alsó függőségek alkotják. Ekkor kerülnek azonosításra a kiértékelő blokkok típusai is.

A C nyelvű kód generálása alatt egy szabványos C szintaktikával létrejövő, az előző lépésekből által felépülő monitort kapunk. Ez alatt generálásra kerül többek között a kiértékelő blokkok struktúrája és az egyes kiértékelő blokkokban szereplő részfák kiértékelő függvénye. Emellett a main függvény is legenerálódik, ami a program alapját képezi.

A fenti lépéseket megvalósító kódgenerátor elkészül és jelenleg a kapott monitor tesztelése zajlik, valamint a fejlesztési tervek között az időbeliség kezelése és más temporális logikák támogatása is szerepel.