Szimbolikus modellellenőrzés és végrehajtási szekvencia generálás vezérelt kereséssel

Absztrakt

A modellellenőrzés egy formális verifikációs technika, melynek feladata a rendszerrel szemben támasztott követelmények matematikailag precíz ellenőrzése a rendszer viselkedésmodelljén. A modellellenőrzés előnye, hogy a követelmények megsértése esetén képes végrehajtási szekvenciával (ellenpéldával) demonstrálni a rendszer helytelen működését. Állapotelérhetőség vizsgálatakor kiemelten fontos az elérhető állapotokba vezető szekvenciák generálása, mivel sok alkalmazási területen (például modellalapú tesztelés) a fő kimenet maga az ellenpélda.

Az első modellellenőrzők a problémát a rendszer lehetséges állapotainak szisztematikus és gyakran kimerítő átvizsgálásával, vagyis az állapottér gráfjának előállításával oldották meg. Ezeket az algoritmusokat explicit modellellenőrzőknek nevezzük, mivel az állapotokat és állapotátmeneteket explicit módon tartják nyilván. Ez a megoldás azonban nagyon hamar korlátokba ütközik, mivel az állapottér már viszonylag kis modellek esetén is kezelhetetlenül nagyméretű lehet: ezt a jelenséget a szakirodalom állapottér-robbanásnak nevezi.

Az állapottér-robbanás kezelésének egyik, a dolgozatban is használt módja a szimbolikus modellellenőrzés, amely az állapotok belső szerkezetét kihasználva döntési diagramokként kódolja el az állapotteret. Ezzel jelentősen növeli a kezelhető állapottér méretét, ugyanakkor megnehezíti az explicit modellellenőrzőkben alkalmazott technikák használatát és az ellenpéldák generálását is.

Konkurens rendszerekben az állapottér-robbanásnak gyakori oka az egymástól független aszinkron műveletek teljes sorrendezése, amely rengeteg, a követelmény szempontjából érdektelen köztes állapothoz vezet. A részleges rendezéses redukció az ekvivalens sorrendezéseket egyetlen reprezentatív szekvenciával helyettesítve csökkenti a bejárt állapotok számát. A módszer az explicit modellellenőrzésre épít, így az ott használt technikák többnyire alkalmazhatóak maradnak.

Ebben a dolgozatban két, egymást kiegészítő algoritmust adunk az elérhetőségi probléma és az ellenpélda-generálás megoldására. Első lépésként egy szimbolikus algoritmus megállapítja, hogy a célállapotok elérhetőek-e, majd az elérhető állapotokról szerzett információt átadja egy explicit, vezérelt modellellenőrző algoritmusnak, amely így képes az elérhető állapotokba vezető végrehajtási szekvenciák hatékony előállítására. Az eredményként kapott hibrid modellellenőrző algoritmus 1) a szimbolikus kódoláshoz az ún. hierarchikus döntési diagramokat használja, 2) az explicit módon felderítendő állapotteret a részleges rendezéses redukció használatával hatékonyan csökkenti, illetve 3) az explicit keresést vezérelt módon, a modell és a követelmények struktúráját figyelembe véve hajtja végre. A két algoritmus között átadott adatok lehetővé teszik, hogy a szimbolikus és explicit módszerek gyengeségeit a másik módszer erősségeivel kompenzáljuk.

Symbolic model checking and trace generation by guided search

Abstract

Model checking is a formal verification technique for verifying requirements on behavioral models of systems in a mathematically precise way. An advantage of model checking is that in case the requirements are violated, it can produce an execution trace (counterexample) to demonstrate the incorrect behavior of the system. When analyzing state reachability, it is especially important to generate traces to reachable states, because a lot of applications (e.g., model-based testing) use them as the primary output of the model checker.

Traditional model checking algorithms systematically and often exhaustively explore the state space of the system, i.e., they build the graph representation of the state space. These algorithms are called explicit model checkers, because they explicitly enumerate the states and state transitions of the system. While explicit approaches are mature, they are inherently limited by the size of the state space that can fit into memory. Unfortunately, even relatively small systems can have huge state spaces, a phenomenon that is commonly referred to as state-space explosion.

One way of handling the state space explosion problem is symbolic model checking, which exploits the inner structure of states to encode the state space as a decision diagram. Symbolic encoding significantly extends the size of manageable state spaces, but also considerably hinders the use of common techniques of explicit approaches and counterexample generation.

One reason of state space explosion in concurrent systems is the interleaving semantics of model checking, i.e., total ordering of independent, asynchronous operations. Examination of each interleaving leads to a large amount of intermediate states that are irrelevant with regard to the requirements. The main idea of partial order reduction is to substitute the equivalent interleavings with a single representative trace to reduce the number of states to discover. The technique builds on the explicit approach, so most of the methods used there remain applicable.

In this work, we present two algorithms that complement each other in checking reachability and generating counterexamples. First a symbolic algorithm determines if target states are reachable, then it collects information about reachable states to pass it to an explicit model checking algorithm. With the provided information, the explicit algorithm is able to efficiently produce an execution trace to the target states. The resulting hybrid model checking approach 1) uses so-called hierarchical decision diagrams for the symbolic encoding of the state space, 2) employs partial order reduction to reduce the state space for the explicit algorithm, and 3) executes the explicit search in a guided way, considering the structure of the model and the requirements. The data exchanged between the two algorithms enables the symbolic and explicit methods to compensate each other's weaknesses with their strengths.