Streszczenie

Systemy typów dla języków programowania są zarówno pierwszą linią obrony przed błędami programistycznymi, jak i pomocą przy strukturowaniu programów wokół struktur danych oraz funkcji które nimi manipulują. Systemy typów to dobry formalizm do wyrażania tych aspektów specyfikacji programów, które mogą być automatycznie, statycznie sprawdzone. Inferencja typów zwiększa wydajność programisty i adaptowalność kodu dostarczając programiście typy wyrażeń, zamiast wymagać podawania typów. Badania nad automatyczną analizą i weryfikacją programów od dawna obejmowały między innymi generowanie specyfikacji spełnianych przez dane programy, na przykład generowanie niezmienników pętli. Ta praca rozszerza znany system typów dla języków funkcyjnych i podaje algorytm inferencji typów, generujący niezmienniki i warunki końcowe funkcji rekurencyjnych.

Generalized Algebraic Data Types (GADTs) rozszerzają systemy typów polimorficznych o wnioskowanie przez przypadki podczas sprawdzania typu definicji przez przypadki. Prezentuję system typów MMG(X) z GADTs, oparty o system typów HMG(X) François Pottiera i Vincenta Simoneta ale bez annotacji typami. Rozszerzam go do języka z typami egzystencjalnymi reprezentowanymi jako domyślnie definiowane i używane struktury GADTs. Pokazuję redukcję problemu inferencji typów do spełnialności więzów drugiego rzędu.

Więzy drugiego rzędu rozwiązuję iterując dwa algorytmy: (1) generalizację więzów, znajdującą najbardziej specyficzną wspólną konsekwencję więzów; (2) łączną abdukcję więzów, znajdująca najogólniejsze wspólne objaśnienie, przesłankę implikującą więzy. Abdukcji używamy głównie do generowania niezmienników, inferencji i sprawdzania typów. Generalizacji używamy do generowania typów egzystencjalnych, służących jako warunki końcowe. Więzy obejmują arytmetykę liniową (równania i nierówności).

System implementujący te techniki nazwałem InvarGenT. Rozwiązuje on zdecydowaną większość zadań inferencji które opracowałem lub zaadaptowałem, bez annotacji typami. InvarGenT rozwiązuje zdecydowanie więcej zadań inferencji typów dla GADTs niż dotychczasowe systemy. Rozwiązuje też więcej problemów inferencji niezmienników i warunków końcowych niż podejście *Liquid Types*, jeśli ograniczymy się do zadań nie potrzebujących inferencji niezmienników dla argumentów funkcji wyższego rzędu. Dodatkowo, prezentuję kilka programów operujących na listach z długością, liczbach binarnych i drzewach AVL o niezbalansowaniu nie przekraczającym 2. Te programy mogą służyć jako część testów dla przyszłych prac nad wszechstronnymi systemami inferencji niezmienników i warunków końcowych.