Konrad Miziński

Logika dla Informatyków – projekt, część I

1. Opis wejścia-wyjścia:
   Wejściem programu jest przekazana jako parametr jego wywołania formułą logiczna, dozwolone znaki to +, &, =>, ~, wielkie i małe litery oraz nawiasy.
   Wyjściem programu jest drukowane na konsoli drzewo rozkładu oraz informacja, czy dana formuła jest tautologią. W przypadku, gdy formuła nie jest tautologią drukowany jest informacja o sekwencie, które będąc nieredukowalnym nie jest prosty, na którego podstawie można sformułować kontrprzykład oraz badane jest czy formuła jest spełnialna (na podstawie sprawdzenia czy jej negacja jest tautologią).
2. Środowisko uruchomieniowe:
   Program został napisany w języku Java a jego zbudowanie możliwe jest za pomocą narzędzia Maven. Aplikacja dostarczona jest w postaci archiwum JAR znajdującego się (zgodnie z konwencją Maven'a w katalogu target). Uruchomienie programu możliwe jest poprzez skrypt run.sh np.:
   `./run.sh '(A+B)=>(A&C)'`
   Dodatkowo zostały dostarczone skrypty example1.sh, oraz example2.sh uruchamiające aplikację z gotowymi przykładami. Na potrzeby uruchomiena pod systemem Windows przygotowany został plik run.bat (odpowiednik pliku run.sh, niestety nie byłem w stanie zweryfikować jego działania). Do uruchomienia programu wymagana jest Java w wersji 7.
3. Fragmenty kodu źródłowego:

Sekwent reprezentowany jest w aplikacji jako obiekt będący zbiorem formuł, na którym można stwierdzić czy jest nieredukowalny oraz czy jest pierwszy:

```
public class Sequent extends HashSet<Formula> {
	...
	public Sequent(Formula... forms) {
		super(Arrays.asList(forms));
	}

	public boolean isIrreducible() {
		...
	}

	public boolean isPrimary() {
		...
	}
}
```

Na Sekwentach są wykonywane operacje reprezentowane przez klasy zawierające motodę pozwalającą stwierdzić czy na danym sekwencie może być ona wykonana, oraz metodę która w wyniku wykonanej operacji zwraca listę nowych sekwentów (być może długości 1):

```
public interface Operation {

	List<Sequent> doOperation(Sequent sequent);

	boolean isOperationPossible(Sequent sequent);
}
```

Przykładem jest operacja usuwania koniunkcji:

```
public class RemoveConjunction extends AbstractOperation {
```

```
	public List<Sequent> doOperation(Sequent sequent) {
		for (Formula formula : sequent) {
			if(formula instanceof Conjunction) {
				return branchSequent(sequent, (Conjunction) formula);
			}
		}
		return asList(sequent);
	}

	private List<Sequent> branchSequent(Sequent sequent, Conjunction
			conjunction) {
		Sequent sequent1 = (Sequent) sequent.clone();
		sequent1.remove(conjunction);
		Sequent sequent2 = (Sequent) sequent1.clone();
		sequent1.add(conjunction.getLeftArgument());
		sequent2.add(conjunction.getRightArgument());
		return asList(sequent1, sequent2);
	}

	public boolean isOperationPossible(Sequent sequent) {
		return hasFormula(sequent, Conjunction.class);
	}

}
```

Sprawdzenie czy dana formuła jest tautologią odbywa poprzez wykonywanie wszystkich operacji na sekwentach tak długo jak długo możliwe jest ich wykonywanie (a możliwe jest jeśli dany sekwent jest jeszcze redukowalny). Nie znalezienie sekwentu, który byłby nieredukowalny oraz nie był prosty oznacza, że dana formuła jest tautologią:

```
private void checkTautogyViaRSDistribution(boolean printTree, Sequent sequent,
			String spaces) {
	printSequent(printTree, sequent, spaces);
	if (sequent.isIrreducible() && !sequent.isPrimary()) {
		counterexample = sequent;
		return;
	}
	for (Operation operation : operations) {
		if (operation.isOperationPossible(sequent)){
			for(Sequent sequent2 : operation.doOperation(sequent)){
				checkTautogyViaRSDistribution(printTree, sequent2,
					spaces + "\t");
			}
			break;
		}
	}
}
```

Całość źródeł znajduje się w katalogu src, przy czym właściwy kod aplikacji znajduje się w katalogu src/main/java, zaś testy w katalogu src/test/java.