CS6124D: Topics in Programming Languages

Sums
20.03.2020

A *Sum* type denoted $T_1 + T_2$, describes a set of values, where each value is either of type T_1 or of type T_2. Such a type is useful in situations where a collection of values of different types is required, but the collection allows only values of same type (as in the case of a *list*). A value of the *Sum* type $Nat + Bool$ can be either of type Nat or of type $Bool$. But now a question arises: is the value *true* to be typed as $Bool$ or $Nat + Bool$? To resolve this, tags are attached to terms of type Sum. We use two tags - *inl* attached to terms of type T_1 and *inr* to terms of type T_2. The term *inl true* is of sum typ $Bool + T_2$, for some type T_2.

New syntactic forms

t::=	...	*terms*
	inl t	
	inr t	
	case t of inl x ⇒ t | inr x ⇒ t	

v::=	...	*values*
	inl v	
	inr v	

T::=	...	*types*
	T+T	

Evaluation Rules

$$\frac{t_1 \to t_1'}{inl\ t_1 \to inl\ t_1'} \quad \text{E-INL}$$

$$\frac{t_1 \to t_1'}{inr\ t_1 \to inr\ t_1'} \quad \text{E-INR}$$

$$\frac{t_0 \to t_0'}{case\ t_0\ of\ inl\ x_1 \Rightarrow t_1 \mid inr\ x_2 \Rightarrow t_2 \to case\ t_0'\ of\ inl\ x_1 \Rightarrow t_1 \mid inr\ x_2 \Rightarrow t_2} \quad \text{E-CASE}$$

$$\text{case (inl } v_0\text{) of inl } x_1 \Rightarrow t_1 \text{| inr } x_2 \Rightarrow t_2 \to [x_1 \mapsto v_0]t_1 \quad \text{E-CASEINL}$$

$$\text{case (inr } v_0\text{) of inl } x_1 \Rightarrow t_1 \text{| inr } x_2 \Rightarrow t_2 \to [x_2 \mapsto v_0]t_2 \quad \text{E-CASEINR}$$

Typing Rules

$$\frac{\Gamma \vdash t_1 :\ T_1}{\Gamma \vdash inl\ t_1 : T_1 + T_2} \quad \text{T-INL}$$

$$\frac{\Gamma \vdash t_1 :\ T_2}{\Gamma \vdash inr\ t_1 : T_1 + T_2} \quad \text{T-INR}$$

$$\frac{\Gamma \vdash t_0 : T_1 + T_2 \qquad \Gamma,\ x_1 : T_1 \vdash t_1 : T \qquad \Gamma,\ x_2 : T_2 \vdash t_2 : T}{\Gamma \vdash case\ t_0\ of\ inl\ x_1 \Rightarrow t_1 \mid inr\ x_2 \Rightarrow t_2 : T} \quad \text{T-CASE}$$

Exercise: A list is a collection of items of the *same* type. Suppose we need to keep a list of courses where each element is a record type with certain set of attributes. But suppose we

defined different types of records for *Core* and *Elective* as *Elective*=$\{$*CourseName:string, Credits: Nat, Cot:Bool*$\}$ and *Core* = $\{$*CourseName:string, Credits: Nat*$\}$. Now inorder to maintain a list of courses, where each course can be either a *Core* or *Elective*, the Sum type *Core+Elective* will be helpful. Write concrete instances of values of type *Core+Elective*. Define an abstraction *isCotNeeded* that when applied on to a *Course* type term, returns *true* if *cot*(consent of teacher) is needed and *false* otherwise.