Semántica estática avanzada

Ej. 3.16 : Recursión mutua en un bloque de procedimientos

La regla original de declaración de procedimientos "memorizaba secuencialmente" las realizadas en un mismo bloque, de manera que si consideramos $D_p = \text{proc } p_i \text{ is } S_i$ con $i \in 1..k$, cada "llamada" a p_i desde S_j, con $i<j$, tiene en cuenta en efecto la "reciente" declaración de p_i, pero evidentemente una llamada a p_i con $i>j$ no ve la declaración de p_i, que "está aún por llegar". Ello sucedería del mismo modo con la regla original call_{ns}, cuando $i=j$, pero en cambio igual que cuando $i<j$, cuando en este caso utilizásemos la "regla recursiva" call_{ns}^{rec}. El efecto de la misma es que "ignoramos" el "antiguo" valor de p, $env'_p\ p$, y en su lugar "utilizamos directamente" el que será "nuevo valor" de p tras su declaración (S, env'_p). Como esto se hace "cada vez" que se llama (recursivamente) a p, el efecto es el mismo que si antes de la declaración "compilada" de p la acabásemos de volver a hacer "exactamente igual".

Se nos indica ahora que la declaración D_p "memorizará" en el significado de p un nuevo "campo" adicional, en el que se "guarda" el propio "texto" de dicha D_p, de manera que en el entorno $env'_p = upd'_p(D_p, env, env_p, D_p)$ cada p_i "recuerda" la totalidad de las declaraciones D_p. Ahora, cuando se nos pida ejecutar una llamada "antes de hacerlo" tendremos que "calcular de nuevo" el efecto del bloque de declaraciones en el que se declaró el procedimiento llamado. Obtenemos una única regla $[\text{call}_{ns}^{rm}]$ que "simultáneamente" captura las llamadas no recursivas, las recursivas y las obtenidas por recursión mutua, entendiéndose que éste es el efecto perseguido cuando aparecen llamadas entre procedimientos declarados en el mismo bloque.

$$[\ \text{call}^{rm}_{ns}\ ] \quad \frac{env'_v,\ env''_p \vdash \langle S_p, sto\rangle \longrightarrow sto'}{env_v,\ env_p \vdash \langle \text{call } p, sto\rangle \longrightarrow sto'}$$

donde $env_p\ p = (S_p, env'_v, env'_p, D_{Bp})$,

siendo $env''_p = upd'_p\ (D_{Bp}, env'_v, env'_p, D_{Bp})$

Observación: Al "ejecutar" upd'_p se van "generando" los nuevos valores asociados a cada p_i declarado en D_{Bp}. Se obtienen cuaternas cuya cuarta componente permanece "constante" (igual a D_{Bp}) mientras que en la tercera se van modificando los valores de los procedimientos "según se van" leyendo", en realidad ¡pero nada!, pues cuando se llame a cualquiera de estos procedimientos se va a "volver a calcular" su significado. Por contra, los valores asociados a todos los demás procedimientos no declarados en D_{Bp} permanecen iguales a los que tenían en el correspondiente entorno "inicial" env'_p. Es en efecto cierto que en relación a los procedimientos estamos "recalculando" reiteradamente una misma información, lo que eventualmente podría "simplificarse". Esto en realidad es lo mismo que sucedía en el caso de las reglas previas para el manejo de la recursión en ambientes estáticos y corresponde a lo que en compilación se llama "generación de código con doble pasada".

Finalmente, en relación a la pregunta final que se plantea en el enunciado del texto, como quiera que

$upd_p\ (D_{Bp}, env'_v, env'_p) = upd'_p\ (D_{Bp}, env'_v, env'_p, D_{Bp})$,

podríamos en efecto usar directamente upd_p en la regla, en vez de upd'_p.

Un último hecho a observar es que el entorno de variables en el que se ejecuta cada procedimiento del bloque mutuamente recursivo, es el mismo, lo que también sucedía sin recursión mutua.