DDS para generar código de tres direcciones para la sentencia *switch*

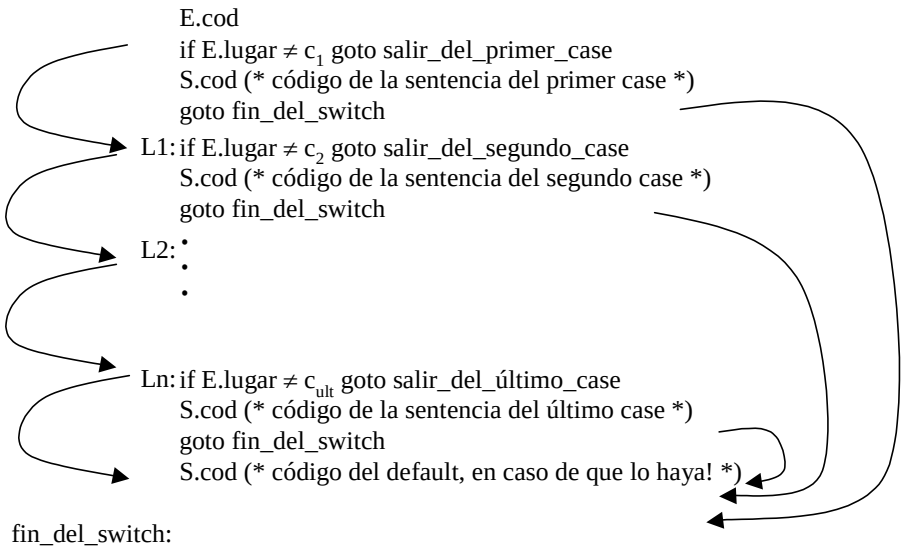
Se plantean a continuación tres soluciones diferentes para obtener la DDS pedida. La gramática para los tres casos es:

1. S → switch E begin L M end
2. L → B L_1
3. L → B
4. B → case C : S
5. M → default : S
6. M → λ
7. C → cte_entera
8. C → cte_real
9. C → cte_char

En la primera SOLUCIÓN:

- Se trabaja con el atributo S.fin (para el *switch*) al cual se da valor dentro de la propia regla (la 1), generándose la etiqueta después de que se termina de obtener todo el código del *switch* (no se trata del atributo heredado S.sig)

- La traducción que se obtiene para el *switch* evalúa la expresión E y, si ésta es distinta del valor de la primera línea *case*, salta a compararla con la de la segunda línea y así sucesivamente (se termina de una forma u otra dependiendo de que haya *default* o no). En cuanto se consigue equiparar uno de los valores *case* con el resultado de la expresión se ejecuta el código de la sentencia de ese *case* (venía secuencialmente después de la comparación) y se salta al final del *switch*.



Se muestran a continuación las reglas en un orden que trata de favorecer su comprensión.

4. B → case C : S

B.salir := nuevaetiqueta
B.cod := gen (‘if’ B.lugar ≠ C.val ‘goto’ B.salir) ||
S.cod || gen (‘goto’ B.fin) || gen (B.salir ‘:’)

(hay que conseguir llevar el resultado de la expresión hasta B.lugar, así como el del fin del *switch* a B.fin. En la regla 1 se dispone de estos valores, que se meten en L para que L los pase a B en las reglas 2 y 3; lógicamente, L también lo pasa a “la otra L” en la regla 2)

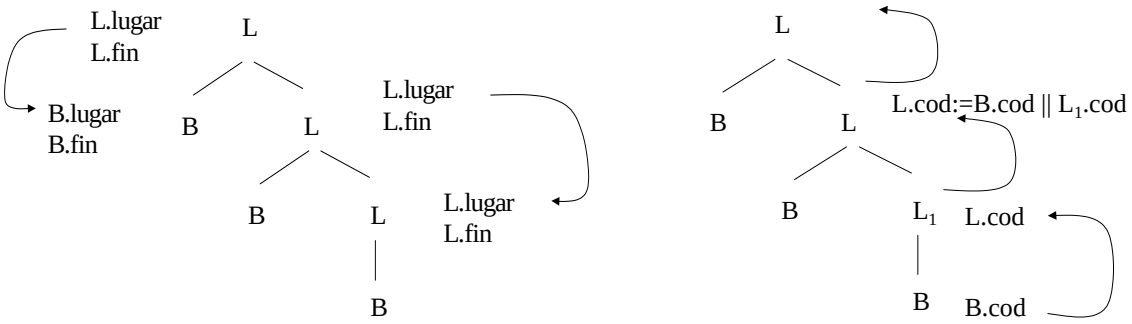
3. L → B

B.lugar := L.lugar
B.fin := L.fin
L.cod := B.cod

(el código va llevándose hacia arriba en el atributo sintetizado .cod, mientras que el resultado de la expresión y la etiqueta de fin del *switch* van bajando por el árbol llevados en atributos heredados)

2. L → B L_1

B.lugar := L.lugar
B.fin := L.fin
L.cod := B.cod || L_1.cod
L_1.lugar := L.lugar
L_1.fin := L.fin



1. S → switch E begin L M end

S.fin := nuevaetiqueta
S.cod := E.cod || L.cod || M.cod || gen (S.fin ‘:’)
L.lugar := E.lugar
L.fin := S.fin

5. M → default : S

M.cod := S.cod

6. M → λ

M.cod := ‘ ‘

7. C → cte_entera

C.val := cte_entera.val (* lo da el A.L. *)

8. C → cte_real

C.val := cte_real.val (* lo da el A.L. *)

9. C → cte_char

C.val := cte_char.val (* lo da el A.L. *)

La segunda SOLUCIÓN presenta una pequeña diferencia: se va a trabajar con el atributo heredado .sig, cuyo valor es la etiqueta de la siguiente instrucción. La inicialización de este atributo no se ve en este ejercicio porque no corresponde a ninguna de las reglas que aquí se manejan.

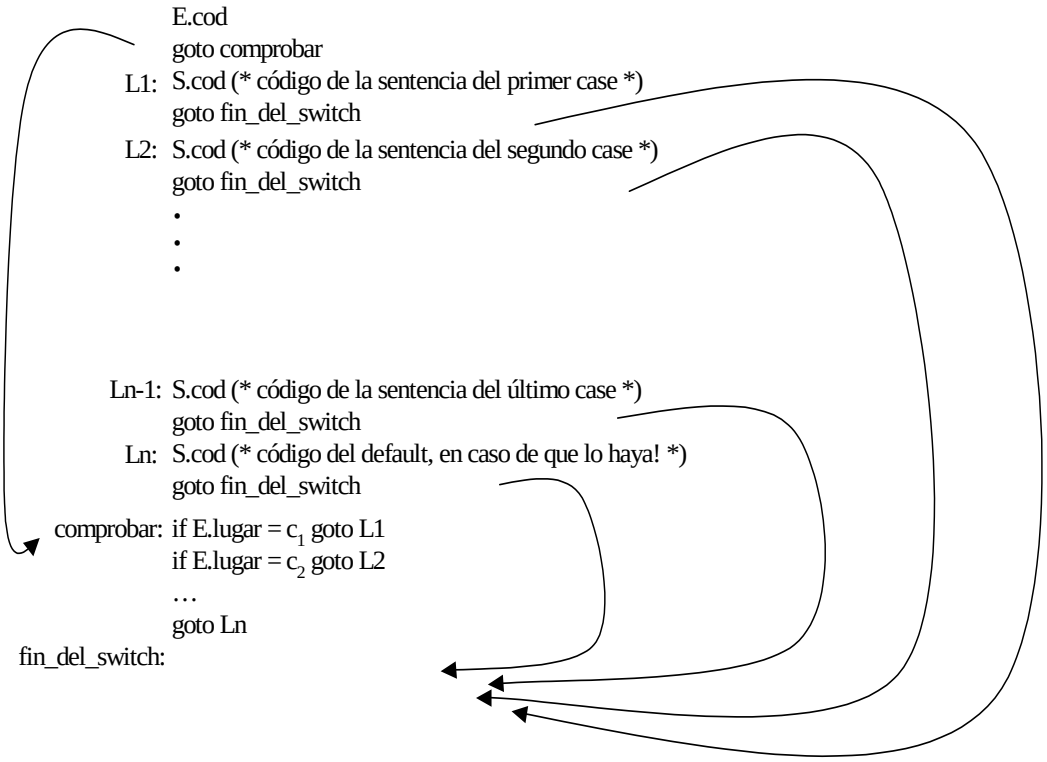
1. S → switch E begin L M end	S.cod := E.cod || L.cod || M.cod L.lugar := E.lugar L.sig := S.sig M.sig := S.sig
2. L → B L_1	B.lugar := L.lugar B.sig := L.sig L.cod := B.cod || L_1.cod L_1.lugar := L.lugar L_1.sig := L.sig
3. L → B	B.lugar := L.lugar B.sig := L.sig L.cod := B.cod
4. B → case C : S	B.salir := nuevaetiqueta S.sig := B.sig B.cod := gen (‘if’ B.lugar ≠ C.val ‘goto’ B.salir) || S.cod || gen (‘goto’ S.sig) || gen (B.salir ‘:’)
5. M → default : S	M.cod := S.cod S.sig := M.sig
6. M → λ	M.cod := ‘ ‘
7. C → cte_entera	C.val := cte_entera.val (* lo da el A.L. *)
8. C → cte_real	C.val := cte_real.val (* lo da el A.L. *)
9. C → cte_char	C.val := cte_char.val (* lo da el A.L. *)

(la etiqueta de la primera instrucción que irá detrás del *switch* va bajando por el árbol mediante el atributo heredado .sig)

Se plantea ahora una tercera SOLUCIÓN. Esta vez:

- Se va a utilizar el atributo heredado .sig

- La traducción que se obtiene para el *switch* evalúa la expresión E y salta hacia delante a una etiqueta 'comprobar' que marca la zona donde se comprobará con qué valor de *case* coincide la E; en función de esto se saltará hacia atrás a la etiqueta correspondiente al código de la sentencia de ese *case* concreto, cuya última instrucción será un salto incondicional al final de la sentencia *switch*. Nótese que, tanto si hay *default* como si no, siempre se tienen la etiqueta "Ln" y el "goto fin_del_switch" correspondiente.



1. S → switch E begin L M end

S.comprob := nuevetiqueta
L.sig := S.sig
M.sig := S.sig
L.lugar := E.lugar
S.cod := E.cod || gen ('goto' S.comprob) || L.cod ||
M.cod || gen (S.comprob ':')

for i=1 to L.long do
S.cod := S.cod ||
gen ('if' E.lugar '=' L.casos[i] 'goto' L.etiqs[i])

S.cod := S.cod || gen ('goto' M.inicio)

Se tienen dos listas (L.casos y L.etiqs) en las que se van guardando, por cada sentencia *case*, el valor que equipara a ese *case* y la etiqueta a la que hay que saltar si es ése el que se ejecuta. Estas listas se van rellenando cada vez que se aplica la regla 2 y cuando se aplica la 3. Se rellenan con los valores de los atributo B.val y B.inicio. Con el for que se ha escrito en la acción semántica de la regla 1 se consigue que en el código 3d aparezca una instrucción de salto condicional por cada *case* que hubiera en el fuente. Al terminar este bucle, se añade al código la última instrucción que falta: un salto incondicional a la etiqueta correspondiente al código del *default*

2. L → B L_1	B.lugar:= L.lugar B.sig := L.sig L.cod := B.cod || L_1.cod L_1.sig := L.sig L.casos := B.val + L_1.casos L.etiqs := B.inicio + L_1.etiqs L.long := L.long + 1
3. L → B	B.lugar:= L.lugar B.sig := L.sig L.cod := B.cod L.casos := B.val L.etiqs := B.inicio L.long := 1
4. B → case C : S	B.inicio:= nuevaetiqueta B.cod := gen (B.inicio ':') || S.cod || gen ('goto' B.sig) S.sig := B.sig B.val := C.val
5. M → default : S	M.inicio:= nuevaetiqueta M.cod := gen (M.inicio ':') || S.cod || gen ('goto' M.sig) S.sig := M.sig
6. M → λ	M.inicio:= nuevaetiqueta M.cod := gen (M.inicio ':') || gen ('goto' M.sig)
7. C → cte_entera	C.val := cte_entera.val (* lo da el A.L. *)
8. C → cte_real	C.val := cte_real.val (* lo da el A.L. *)
9. C → cte_char	C.val := cte_char.val (* lo da el A.L. *)