1 Considerazioni

Queste sono piccole note che cercano di ampliare le slide sulla parte di generazione codice per le operazioni sugli array. Come dalle slide si considerreranno array salvati in memoria per **Row-Major order**. Ricordo anche che la notazione per il tipo di un array di interi k-dimensionale viene denotato:

$$array(size_1, array(size_2, array(...(size_k, int))))$$

2 Syntax Directed Translation per gli Array

2.1 Indirizzamento

Il problema maggiore nel generare codice per referenziare array sta nel calcolo degli indirizzi dei suoi elementi, prendiamo l'indirizzo di $A[i]$:

$$base + i \cdot w$$

in cui *base* è l'indirizzo relativo al primo elemento di A e w è la grandezza di ogni elemento dell'array, se volessimo indirizzare l'elemento $A[i_1][i_2]$ di una matrice assumendo che w_1 è la dimensione di una riga della matrice e w_2 è la dimensione di ogni elemento avremo:

$$base + i_1 \cdot w_1 + i_2 \cdot w_2$$

In k-dimensioni otteniamo:

$$base + i_1 \cdot w_1 + i_2 \cdot w_2 + \cdots + i_k \cdot w_k \tag{1}$$

2.2 Grammatica

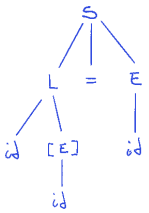
Consideriamo la seguente grammatica in cui E rappresenta delle espressioni mentre L rappresenta il nome di un array (quindi la sua locazione in memoria):

$$S \to id = E \mid L = E$$

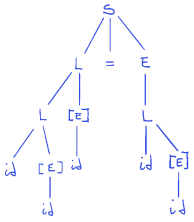
$$E \to id \mid L \mid E + E$$

$$L \to id[E] \mid L[E]$$

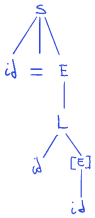
Da questa grammatica possiamo generare ad esempio:



$a[b] = c$



$a[b][c] = d[e]$



$a = b[c]$

2.3 Attributi

- $L.addr$ denota un temporaneo che viene usato per calcolare l'indirizzo dell'elemento dell'array sommando i termini $i_j \cdot w_j$ in (1)
- $L.array$ è un puntatore ad una entry della tabella dei simboli per il nome dell'array, in cui $L.array.base$ rappresenta l'indirizzo base dell'array
- $L.type$ è il tipo del sotto array generato da L, per ricavare il tipo utilizzeremo $arg2$ che va a recuperare il secondo argomento (per esempio $arg2(2, array(3, int)) = array(3, int)$)
- $L.width$ é la grandezza del subarray generato da L, ricavata attraverso la funzione ausiliaria $width()$ (per esempio $width(array(3, int)) = 3 \cdot int_{size}$)

2.4 Regole semantiche

La produzione $\mathbf{S} \to \mathbf{id} = \mathbf{E}$ deve generare il codice di assegnamento a una variabile con il valore di un'espressione:

$$\{gen(table.get(id)\ '='\ E.addr);\}$$

Le azioni semantiche per $\mathbf{S} \to \mathbf{L} = \mathbf{E}$ devono generare il codice che assegna il valore dell'espressione **E** alla locazione di memoria referenziata da **L**.
Ricordando che:

- $L.array$ è il puntatore all'array nella tabella dei simboli
- $L.array.base$ è l'indirizzo del primo elemento
- $L.addr$ denota il temporaneo che contiene l'offset dell'elemento

possiamo concludere che l'elemento sarà in $L.array.base[L.addr]$ e l'istruzione generata copierà r-value da $E.addr$ alla locazione di L:

$$\{gen(L.array.base\ '['\ L.addr\ ']'\ '='\ E.addr);\}$$

Le produzioni $\mathbf{E} \to \mathbf{E_1} + \mathbf{E_2}$ e per $\mathbf{E} \to \mathbf{id}$ sono analoghe alla prima:

$$\{gen(E.addr\ '='\ E_1.addr\ '+'\ E_2.addr);\}$$

$$\{gen(E.addr\ '='\ table.get(id));\}$$

La produzione $\mathbf{E} \to \mathbf{L}$ deve generare il codice per copiare l'elemento dell'array denotato da **L** in un nuovo temporaneo:

$$\{E.addr = newTemp();\quad gen(E.addr'='L.array.base\ '['\ L.addr\ ']');\}$$

La produzione $\mathbf{L} \to \mathbf{id}[\mathbf{E}]$ deve trasferire le informazioni sull'array denotato da **id** in **L**.

$\{L.array = table.get(id);$	Recupera la voce della tabella dei simboli dell'array utilizzando l'identificatore id (ad es., "a")
$L.type = arg2(table.getType(id));$	Estrae il tipo del sottoarray. Ad esempio, se a è $array(2, array(3, integer))$, $L.type$ diventa $array(3, integer)$
$L.width = width(L.type)$	Determina la dimensione (in byte) del sottoarray rappresentato da $L.type$
$L.addr =$ newTemp()	Crea un nuovo temporaneo
gen($L.addr\ '='\ E.addr\ '*'\ L.width\ )\}$	Calcola il primo offset, cioè in $a[i_1][i_2][...] \to i_1 \cdot w_1$

Infine la produzione $\mathbf{L} \to \mathbf{L_1}[\mathbf{E}]$ dovrà trasferire i dati processati da id, ed aggiornare l'offset per il corretto indirizzamento dell'elemento:

$\{L.array = L_1.array$	L'array a cui si fa riferimento rimane lo stesso.
$L.type = arg2(L_1.type)$	Estrae il tipo del sottoarray a questo livello. Se L1.type era array(3, integer), L.type diventa integer.
$L.width = width(L.type)$	Calcola la dimensione del sottoarray corrente (ad esempio, la dimensione di un integer).
$t =$ newTemp()	Crea un nuovo temporaneo per l'elaborazione intermedia
gen($t\ '='\ E.addr\ '*'\ L.width$)	Calcola l'offset per la dimensione corrente (utilizzando E.addr per l'indice e L.width) e lo memorizza in una variabile temporanea t.
$L.addr =$ newTemp()	Crea un altro temporaneo per l'indirizzo risultante
gen($L.addr\ '='\ L_1.addr\ '+'\ t)\}$	Combina l'offset del livello precedente (L1.addr) con l'offset appena calcolato (t) per ottenere l'offset totale per questo elemento. Questo offset totale viene quindi memorizzato in L.addr.