Assembler

Leonardo Bizzoni

April 2, 2023

Il compito dell'assembler è quello di tradurre un file assembly in un file contenente istruzioni macchina e dati macchina.

Il primo passo della traduzione è quello di identificare i **label** presenti e creare una **tabella di associazioni** tra il label e l'istruzione/dato a cui è collegato. Il secondo passo è quello di tradurre ogni istruzione assembly con i registri su cui opera e gli eventuali valori immediate in codice macchina.

L'output dell'assembler è un object file tipicamente **non** eseguibile in quanto può contenere reference a procedure esterne o label contenuti in un altro modulo.

Un label è detto **esterno** o **globale** se un altro modulo può accedervi. Un label di base è **locale**.

Per rendere un label globale si utilizza la direttiva *.globl*

Dato che ogni modulo viene assemblato individualmente c'è bisogno di un tool esterno, il linker, per risolvere i label o chiamate a procedure non definite all'interno del modulo.

Anche i label definiti all'interno dello stesso modulo possono causare problemi all'assembler in quanto è possibile utilizzare label non ancora definiti all'interno di un'istruzione. Questo tipo di reference a label non ancora definiti viene chiamata **forward reference**.

Ci sono 2 strategie per risolvere le forward reference:

- Effettuare una prima lettura dove ogni istruzione viene suddivisa in **lessemi** (*ble $t0,100,loop contiene 6 lessemi*). Se una riga inizia con un label, l'assembler memorizza il nome e l'indirizzo di memoria nella

symbol table. Quando l'assembler arriva alla fine del modulo, nella symbol table ci sono tutte le associazioni label-indirizzo definite. L'assembler effettua una seconda lettura sta volta traducendo ogni istruzione in istruzione macchina grazie alle informazioni presenti nella symbol table.

- **Backpatching**: in una singola lettura l'assembler costruisce una rappresentazione binaria **incompleta** di ogni istruzione. Se un'istruzione contiene una reference ad un label non ancora definito allora il label e l'istruzione vengono salvate in una table. Quando un label viene definito l'assembler cerca nella table e completa la rappresentazione binaria dell'istruzione corrispondente al label definito. Backpatching richiede però che l'assembler tenga in memoria la rappresentazione binaria dell'intero programma.