Compiler Writing

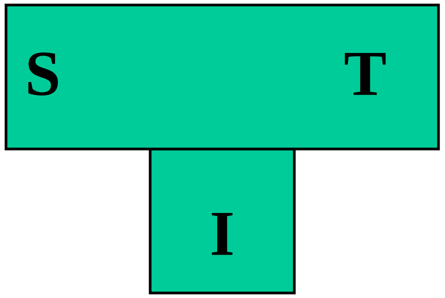
- Source Language Issues
  - Size of the source language (bigger = harder)
  - Extent of change during compiler construction (more changes = harder)
- Performance Criteria
  - Compiler Speed
  - Code Quality
  - Error Diagnostics
  - Portability
  - Maintainability

Performance Criteria

- Portability
  - Retargetability
  - Rehostability
- A Retargetable compiler is one that can be modified easily to generate code for a new target language
- A rehostable compiler is one that can be moved easily to run on a new machine
- A portable compiler may not be as efficient as a compiler designed for a specific machine, because we cannot make any specific assumption about the target machine

How was the first compiler compiled?

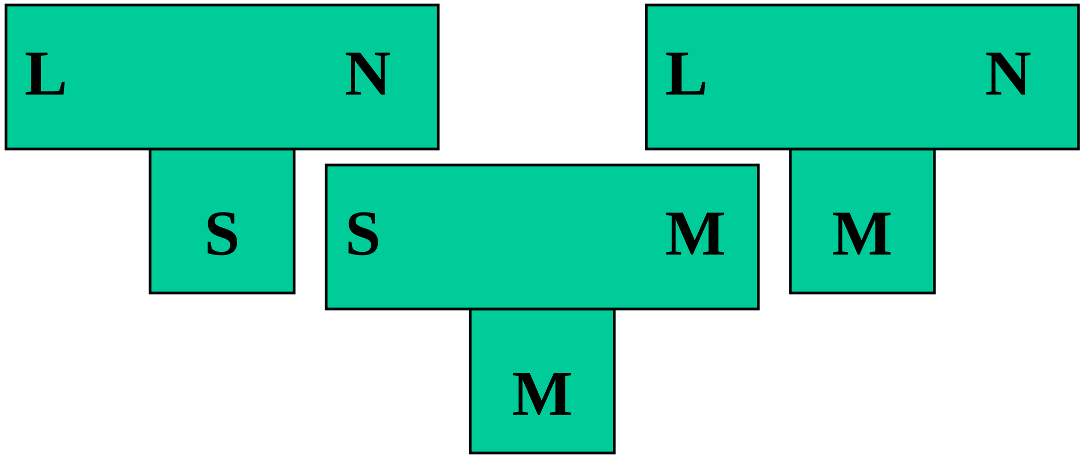
- Bootstrapping: using the facilities offered by a language to compile itself is essence of bootstrapping
- There are three languages involved in writing a compiler
  - Source Language (S)
  - Target Language (T)
  - Implementation Language (I)
- T-Diagram



How was the first compiler compiled?

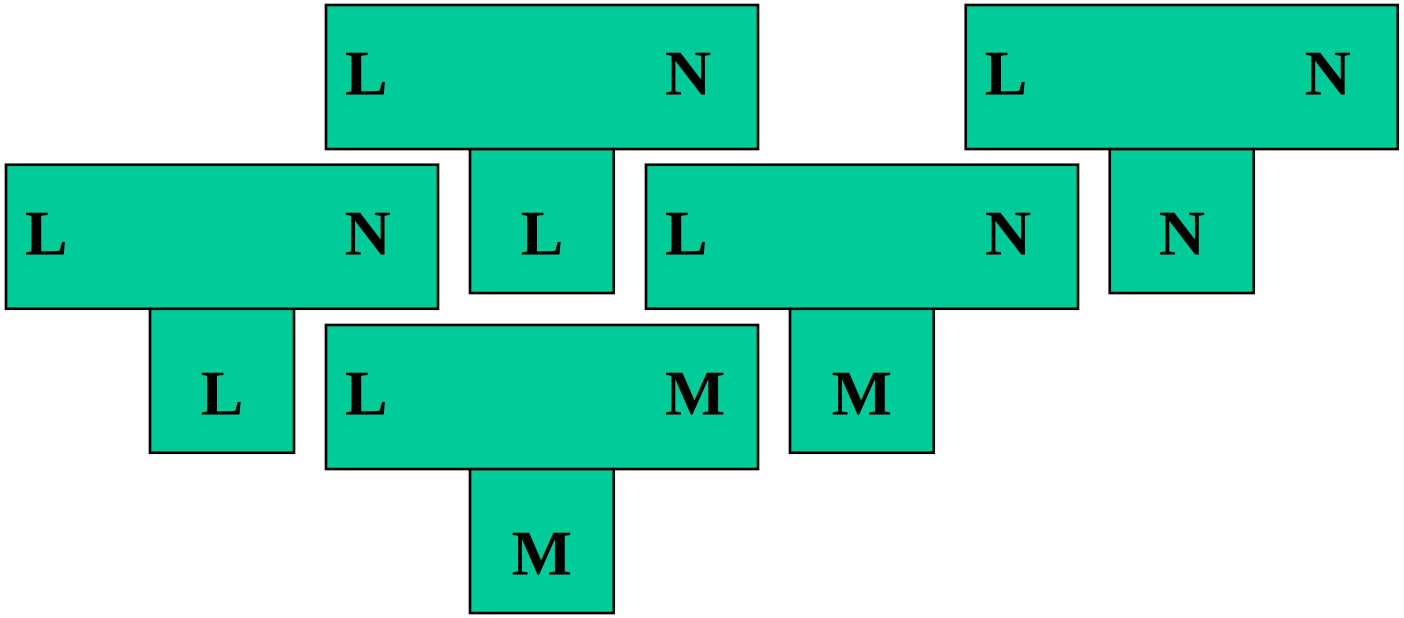
- Cross Compiler: If S, T, and I are all different Languages, the compiler is a cross-Compiler

- $L_SN + S_MM = L_MN$



Using BootStrapping to Port a Compiler

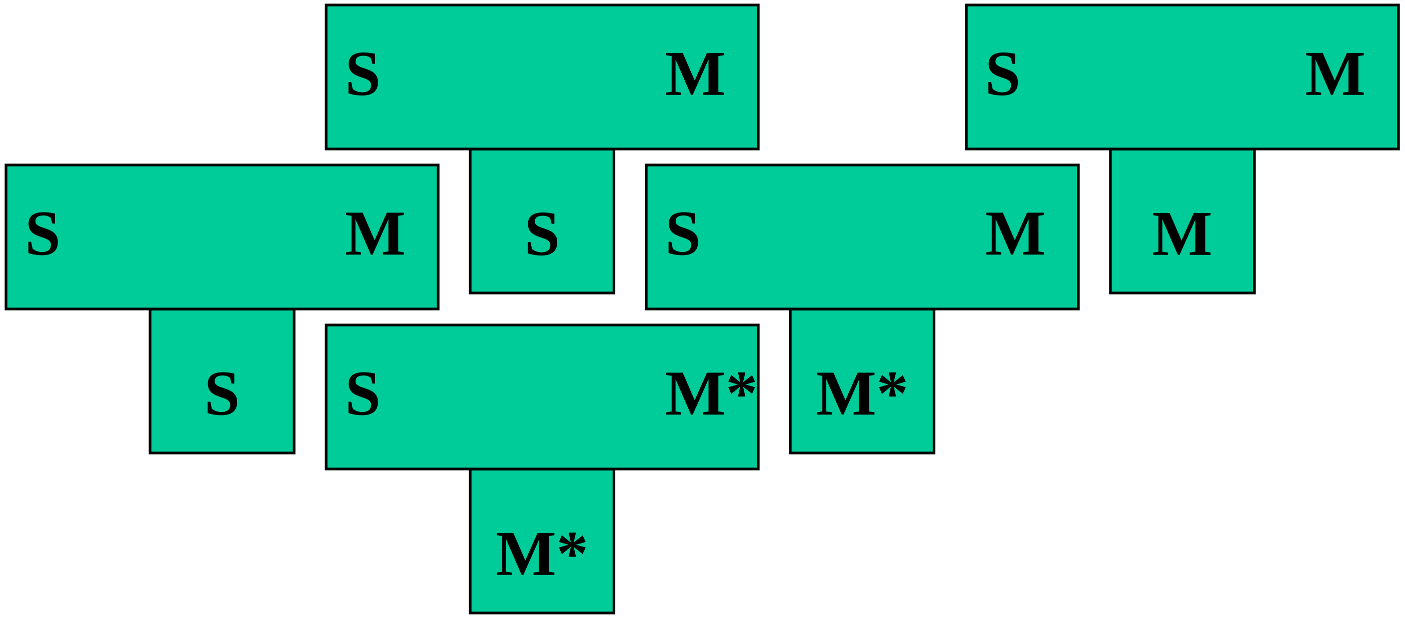
1. $L_L N + L_M M = L_M N$
2. $L_L N + L_M N = L_N N$



Using Bootstrapping to Optimize a Compiler

- Using bootstrapping, an optimizing compiler can optimize itself (e.g., M is an optimizing Compiler)

1. $S_SM + L_{M*}M^* = S_{M*}M$
2. $S_SM + S_{M*}M = S_MM$


S
M
S
M
S
M
S
S
M
M
S
S
M*
M*
M*