Content

- Prologue
- Functional Programming
- ATS – Applied Typed System
- Practice of ATS
Prologue

- Types, nothing but types.
- Mathematics and Logic
  - Set theory and recursion
  - Curry-Howard Isomorphism
Functional Programming

- Haskell, Meta Language (ML), Erlang, Applied Type System (ATS)
- Functional Program = no assignment (no update)
- Example: append two lists (imperative programming)
Functional Programming

- Example: append two lists (functional programming)
Functional Programming

- No loop (no while or for)
- Using recursive function call to replace loop
- Example (Factorial Function)

C / C++, Java
```c
int fact(int n) {
    if (n <= 1) return 1;
    int accu = 1;
    for (; n > 1; --n) {
        accu = accu * n;
    }
    return accu;
}
```

ML, ATS
```ml
fun fact (n: int): int = if n <= 1 then 1 else n * fact (n - 1)
```

```ml
fun fact2 (n: int, accu: int): int = if n <= 1 then accu else fact2 (n - 1, accu * n)
```
ATS – Advanced Typed System

- Specialty of ATS (In types, we trust.)
- Specification of program

<table>
<thead>
<tr>
<th>Python</th>
<th>C / C++</th>
<th>ATS</th>
</tr>
</thead>
<tbody>
<tr>
<td>def foo(*args):</td>
<td>int foo (int x, int y)</td>
<td>fun add (x: int, y: int): int</td>
</tr>
</tbody>
</table>

**Weak**

def foo(*args):
    int add (int x, int y)

fun add {m, n: int}.<>.
    (x: int m, y: int n):<> int (m + n)
    = x + y  // cannot be x - y

**Strong**
ATS – Advanced Typed System

- Statics and Dynamics (Two separate worlds bridged by Dependent Type)
- Sorts V.S. Types
- Dependent Type

```plaintext
fun add {m, n: int} .<>.
  (x: int m, y: int n):<> int (m + n)

fun add {m, n: int} .<>.
  (x: int m, y: int n):<> [sum: int | sum == m + n] int sum
```

- `{m: int} (x: int m) ➔ What is the type of x?`
- `[sum: int | sum == m + n] int sum ➔ What’s the type of return value?`
Practice of ATS

- How to install ATS
  - [http://sites.google.com/site/alex2ren/cs320-summer-i-2010-tf-1/howtoinstallats](http://sites.google.com/site/alex2ren/cs320-summer-i-2010-tf-1/howtoinstallats)

- How to use ATS installed on csa2.bu.edu
  - [http://sites.google.com/site/alex2ren/cs320-summer-i-2010-tf-1/howtouseatsinstalledoncsa2buedu](http://sites.google.com/site/alex2ren/cs320-summer-i-2010-tf-1/howtouseatsinstalledoncsa2buedu)

- How to use ATS
  - Emacs, Vim
  - Compile and run (Makefile)
  - atsc c xxx.dats
Practice of ATS

- Dynamics: purely functional programming language
- Types and Terms
  - 1 is of type int, true is of type bool
  - user defined type \( \rightarrow \) datatype

- Example: operations on list
Practice of ATS

- Statics: Simply typed language
- Sorts and Terms
  - 1 is of sort int, true is of sort bool
  - user defined sort \(\rightarrow\) datasort
- prop: a special sort
  - specify a relation between multiple sorts
Practice of ATS

- Where to find other resources?