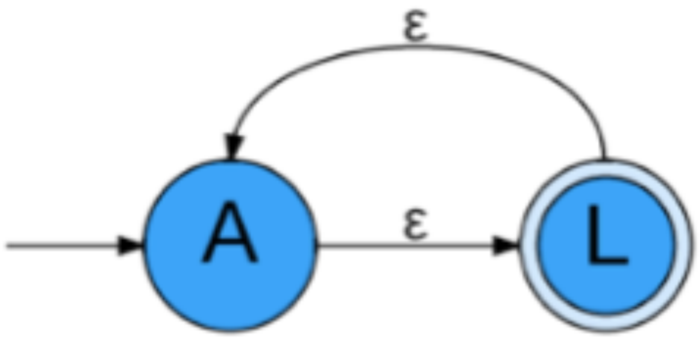
AL



Automata Language

Yujin Ariza
System Integrator

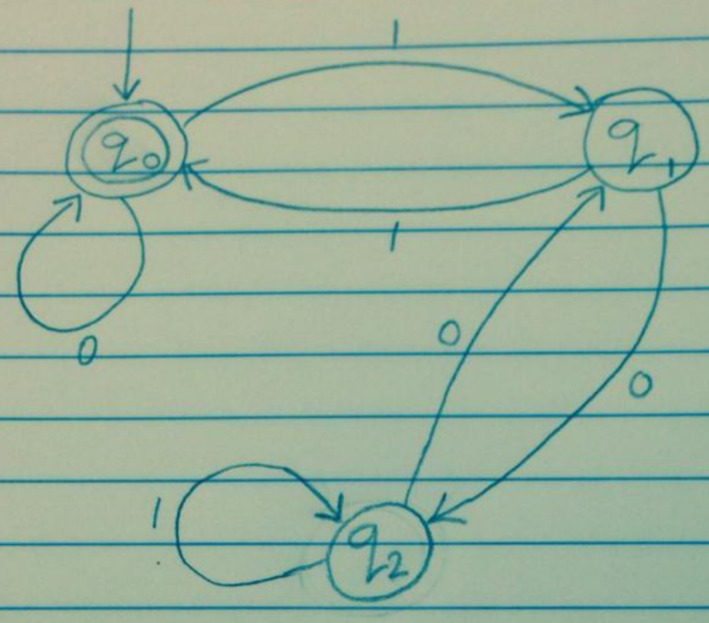
Dianna Hohensee
Language Guru

Chris Erlendson
System Architect

Kristie Howard
Project Manager

How did we decide on AL?

- CS Theory Homework



$Q = \{q_0, q_1, q_2\}$

$q_0 = q_0$

$F = \{q_0\}$

$\Sigma = \{0, 1\}$

	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

Why AL?

- CS Theory Homework
- Automata implementation is complicated...

```java
public void printResult(String input) {
    this.printAutomaton();
    System.out.println("\nRunning Automaton " + this.name + " on input " + input + "\n");
    currentStates = this.epsilonClosure(start);
    nextStates = new StateSet();
    System.out.println("Starting in state: " + start.getName());
    int length = input.length();
    for (int i = 0; i < length; i++) {
        System.out.println();
        String c = input.substring(i,i+1);
        System.out.println("Input char:" + c );
        if (!this.alphabet.contains(c)) {
            System.out.println("\nThe input character " + c + " is not in the alphabet.");
            System.out.println("Automaton " + this.name + " REJECTS the input " + input + ".\n");
            return;
        }
        //check transitions and e transitions
        for (State curr : currentStates)
            updateNextStates(curr, c);
        System.out.println("Current states after input " + c + ": "+ nextStates.toString());
        if (nextStates.isEmpty()) {
            System.out.println("\nThere are no outgoing transitions on input character " +
                    c + " from any of the current states.");
            System.out.println("Automaton " + this.name + " REJECTS the input " + input + ".\n");
            return;
        }
        //end of computation, no transitions
        currentStates.clearEntries();
        currentStates.addStateSet(nextStates);
        nextStates.clearEntries();
    }
    System.out.println("\nThe input is finished.");
    //reached end of input: check currentStates for accepting
    boolean accepted = false;
    StateSet acceptedFinals = new StateSet();
    for (State finalState : currentStates) {
```

Why AL?

- CS Theory Homework
- Automata implementation is complicated…
- You can see why an input is not accepted

```
Automaton name: myAhoMachine
Alphabet: {a, h, o}
States: {begin, s1, s2, s3}
Starting state: begin
Accepting states: {s3}
Transition Table:
+--------------------------+----------+--------------------------------------+
|   State name             | Input    | Next state(s)                        |
+--------------------------+----------+--------------------------------------+
|-->begin                  | a   | {s1}                                 |
|   s1                     | h        | {s2}                                 |
|   s2                     | o        | {s3}                                 |
+--------------------------+----------+--------------------------------------+
NOTE:' * ' designates accepting state, '-->' designates start state

Running Automaton myAhoMachine on input aho

Starting in state: begin

Input char:a
Current states after input a: {s1}

Input char:h
Current states after input h: {s2}

Input char:o
Current states after input o: {s3}

The input is finished.
The automaton is in the following accepted states: {s3}
Automaton myAhoMachine ACCEPTS the input aho.
```

```
printResult(myAhoMachine, “ah”);

# output to screen with automaton
information omitted
-----------------------------------
Running Automaton myAhoMachine on
input ah

Starting in state: begin

Input char:a
Current states after input a: {s1}

Input char:h
Current states after input h: {s2}

The input is finished.
None of the final states were
accepting states.
Automaton myAhoMachine REJECTS the
input ah.
```

```
printResult(myAhoMachine, “hah”);

# output to screen with automaton
information omitted
----------------------------------------
Running Automaton myAhoMachine on input
hah

Starting in state: begin

Input char:h
Current states after input h: {}

There are no outgoing transitions on
input character h from any of the current
states.
Automaton myAhoMachine REJECTS the input
hah.
```

Buzzwords

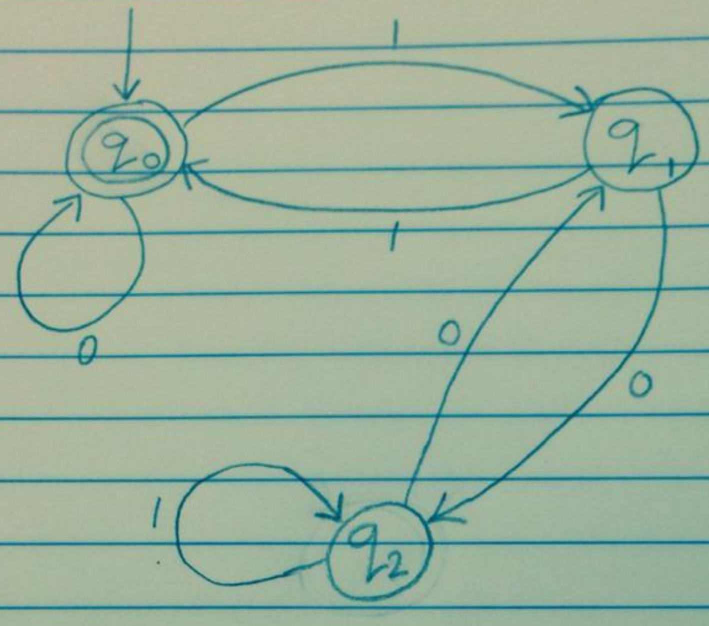
- Domain specific
- Object-oriented
- Easy and Concise
- Powerful

Syntactic Constructs

- State
- Automaton
- Set
- Alphabet
- StateSet
- *Boolean*
- *String*
- *Char*

Divide Example Program

```
Automaton auto {
      State q0, q1, q2;
      StateSet states = {q0, q1, q2};
      State start = q0;
      StateSet accept = {q0};
      Alphabet alphabet = {'0', '1'};
      q0.trans('0') = {q0};
      q0.trans('1') = {q1};
      q1.trans('0') = {q2};
      q1.trans('1') = {q0};
      q2.trans('0') = {q1};
      q2.trans('1') = {q2};
};
```



$Q = \{q_0, q_1, q_2\}$

$q_0 = q_0$

$F = \{q_0\}$

$\Sigma = \{0, 1\}$

	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

Auto Example Program

```
Automaton auto {
      State q0, q1, q2;
      StateSet states = {q0, q1, q2};
      State start = q0;
      StateSet accept = {q0};
      Alphabet alphabet = {'0', '1'};
      q0.trans('0') = {q0};
      q0.trans('1') = {q1};
      q1.trans('0') = {q2};
      q1.trans('1') = {q0};
      q2.trans('0') = {q1};
      q2.trans('1') = {q2};
};
```

```
# Indicate success or failure

if (auto.run(“01”)) {
      print(“Success!”);
} else {
      print(“Failure…”);
}



# Or, give a detailed report

printResult(auto, “1001”);
```

Object-Oriented Syntax

```
Automaton rabbit { … };

StateSet bunnies = rabbit.states;           # Access the StateSet of Automaton bomb
State fred = bomb.states.jackrabbit;        # Access a state in Automaton bomb

rabbit.begin.trans('h') = {fred};
-----------------------------
```

If / Else Statements

```
Automaton bomb { … };

if (bomb.run("atomic")) {
        print("Success!");
 } else {
        print("Failure...");
 }
```

For Loops

```
Automaton patch { … };

# for each char in patch’s Alphabet
for each Char ch in patch.alphabet {
        patch.states.vegetable.trans(ch) = {patch.states.pumpkin};
}

# for each State in patch’s StateSet
for each State s in patch.states {
        ...
}
```

Type Conversion

```
# for each char in patch's Alphabet
for each Char ch in patch.alphabet {
        patch.states.vegetable.trans(ch) = {patch.states.pumpkin};     # conversion to StateSet
}
```

Functions

```
Def StateSet epsilonClosure(State state) {

        # initialize the StateSet
        StateSet set = {state};

        # call the recurse function
        for each State st in state.trans('epsilon')
{
                if (!set.contains(st)) {
                        set = set & {st};
                        set = recurse(st, set);
                }
        }

        return set;
}
```

```
Def StateSet recurse(State s, StateSet set) {

        # iterate through all e-transitions
        for each State state in s.trans('epsilon') {
                if (!set.contains(state)) {
                        set = set & {state};
                        set = recurse(state, set);
                }
        }

        return set;
}
```

Email Operator Example

```
Automaton emailStart {
      State name, at, domainPrefix, valid;
      StateSet states = {name, at, domainPrefix, valid};
      State start = name;
      StateSet accept = {valid};
      Alphabet alphabet = {LETTERS, NUMBERS, '@'};
      # Transitions
      name.trans(LETTERS,NUMBERS) = {name};
      name.trans('@') = {at};
      at.trans(LETTERS,NUMBERS) = {valid};
      valid.trans(LETTERS, NUMBERS) = {valid};
};
```

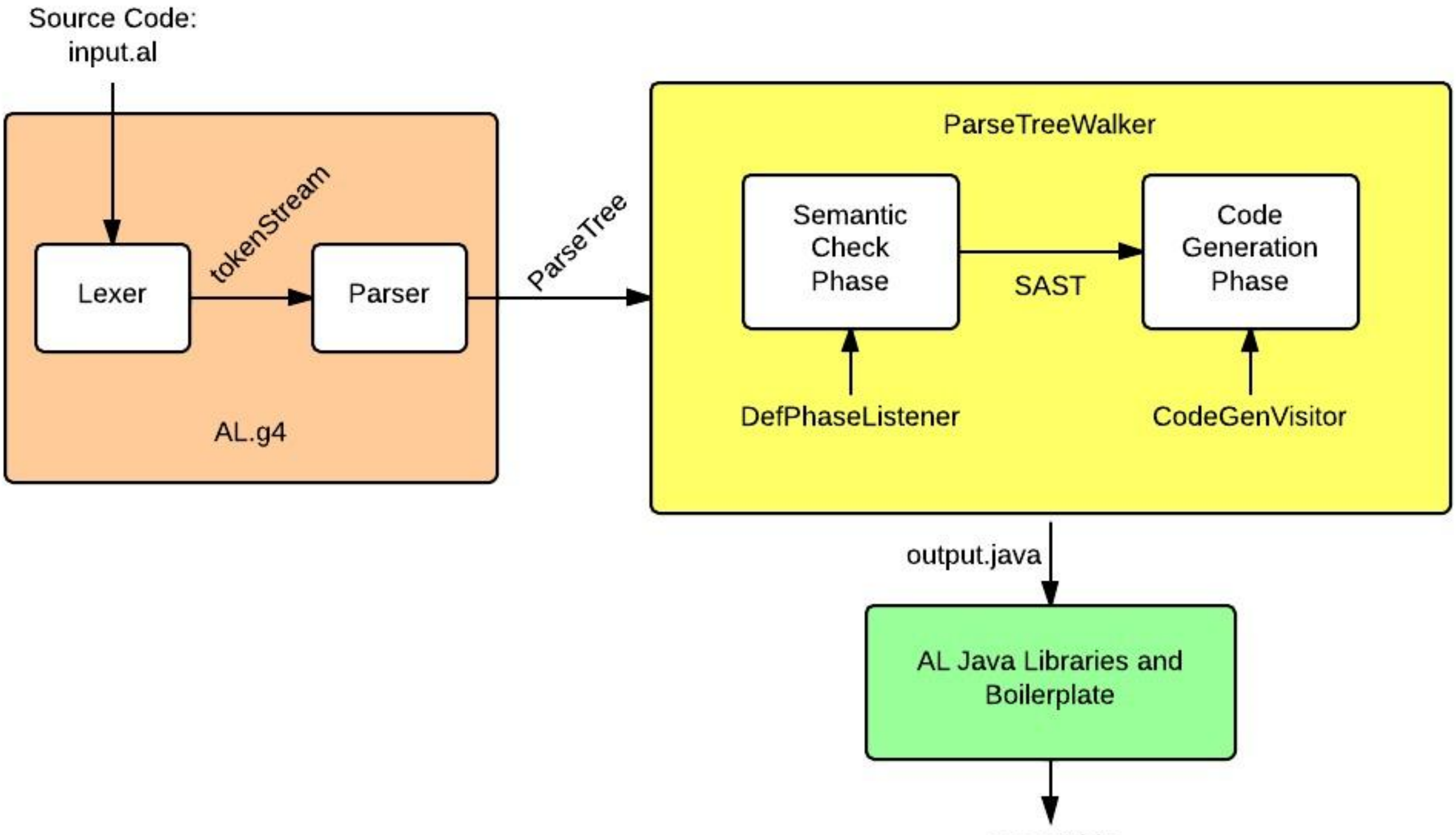
Email Operator Example

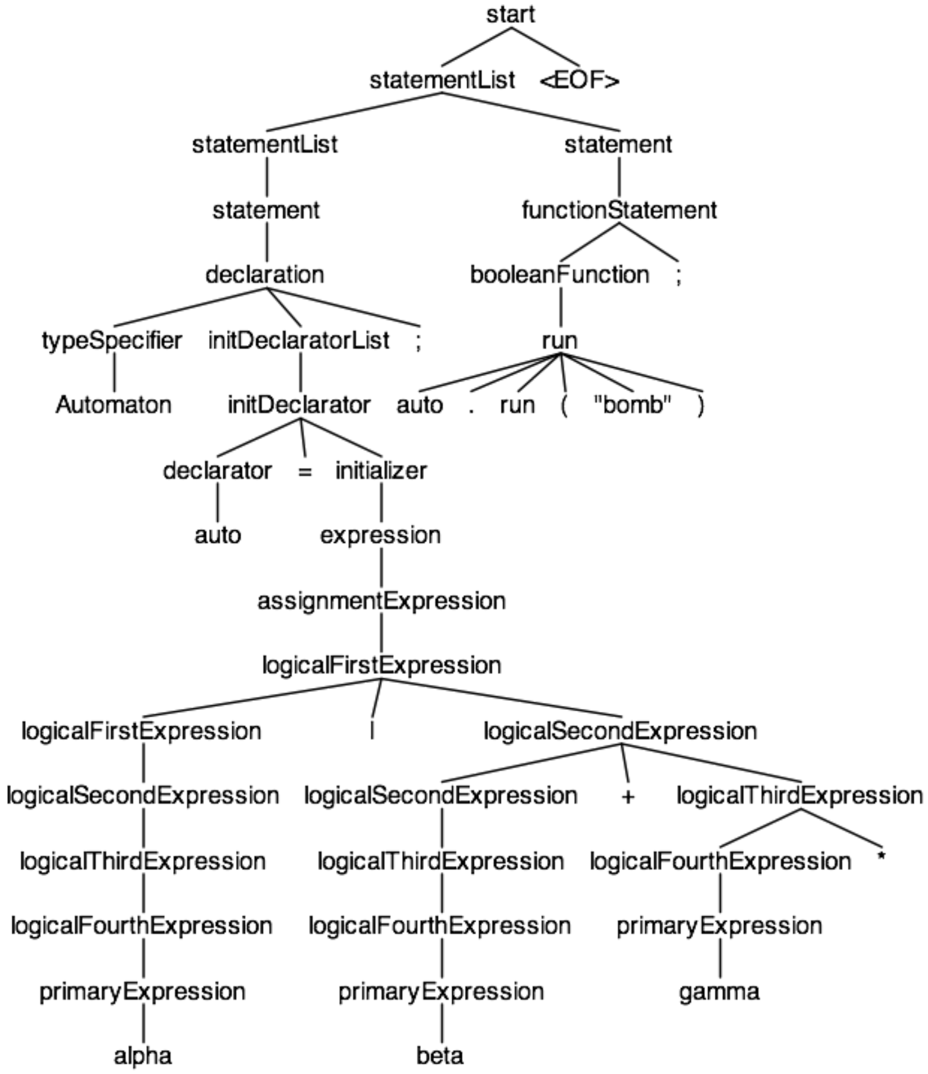
```
Automaton endings = automaton(".edu") | automaton(".com") | automaton(".gov");

Automaton email = emailStart + endings;

if (comEmail.run("automaton@fun.com")) {
      if (schoolEmail.run("project@plt.edu") {
            if (govEmail.run("taxes@IRS.gov")) {
                  print(“Success!”);
            }
      }
} else {
      print("Not an .edu, .gov, or .com email.");
}
```

Tools

```
* dc05973 - Renamed test files for clarity; Codegen does macros (3 days ago) <Yujin Ariza>
* 561357e - updated (3 days ago) <Dianna Hohensee>
*   fc7d775 - Merge branch 'master' of https://bitbucket.org/CDKYZ/fail (3 days ago) <Chris Erlendson>
|\
| * b5c5934 - Updated test script with diff functionality (3 days ago) <Yujin Ariza>
* |   0a170b0 - merge res, overwrote visitAssignmentExpression with my own (3 days ago) <Chris Erlendson>
|\ \
| |/
| * c9bb59b - Fixed copy issue (3 days ago) <Yujin Ariza>
| * a0888f0 - Concatenation fix (3 days ago) <Yujin Ariza>
* |   935c369 - trans in assignment AND return expr rules is now working (3 days ago) <Chris Erlendson>
|\ \
| |/
| *   420611d - merge (3 days ago) <Yujin Ariza>
| |\
| | *   be3204d - Updated java code (3 days ago) <KRISTIE HOWARD>
| | |\
| | * | 2427d7b - Fix on .getTransitionStates(...), all ref tests (3 days ago) <KRISTIE HOWARD>
| | * |   1655bb6 - trying to get LRM (3 days ago) <KRISTIE HOWARD>
| | |\ \
| | * | | dd22003 - print.al (3 days ago) <KRISTIE HOWARD>
| * | | | b296fe2 - Function calls + declaration works now (3 days ago) <Yujin Ariza>
| | |_|/
| |/| |
| * | |   581fdb7 - merged (3 days ago) <Yujin Ariza>
| |\ \ \
| * | | | 2e6e0b0 - Added AssignmentExpression. trans todo (3 days ago) <Yujin Ariza>
* | | | | 2199c83 - attempt at trans, utter failure so far. keep getting null on visitor, pulling and resolving. (3 days ago) <Chris Erlendso
| |/ / /
|/| | |
* | | |   4969c26 - Merge branch 'master' of https://bitbucket.org/CDKYZ/fail (3 days ago) <Chris Erlendson>
|\ \ \ \
| |/ / /
| * | |   8ad729e - Merge branch 'master' of bitbucket.org:CDKYZ/fail (3 days ago) <Yujin Ariza>
| |\ \ \
```

System Architecture


Source Code:
input.al
tokenStream
Lexer
Parser
AL.g4
ParseTree
ParseTreeWalker
Semantic
Check
Phase
SAST
Code
Generation
Phase
DefPhaseListener
CodeGenVisitor
output.java
AL Java Libraries and
Boilerplate


start
statementList
<EOF>
statementList
statement
statement
functionStatement
declaration
booleanFunction
;
typeSpecifier
initDeclaratorList
;
run
Automaton
initDeclarator
auto
.
run
(
"bomb"
)
declarator
=
initializer
auto
expression
assignmentExpression
logicalFirstExpression
logicalFirstExpression
|
logicalSecondExpression
logicalSecondExpression
logicalSecondExpression
+
logicalThirdExpression
logicalThirdExpression
logicalThirdExpression
logicalFourthExpression
*
logicalFourthExpression
logicalFourthExpression
primaryExpression
primaryExpression
primaryExpression
gamma
alpha
beta

Testing

```
RunTests() {
    basename=$(echo $1 | sed 's/.*\///' | sed 's/.al//')
    reffile=$(echo $1 | sed 's/.al$/.ref/')
    javafile=$(echo $1 | sed 's/.al$/.java/')
    directory=$(echo $1 | sed 's/\/[^/]+\.al/\//')
    outfile=$(echo $1 | sed 's/.al$/.out/')
    difffile=$(echo $1 | sed 's/.al$/.diff/')

    echo "testing $javafile..."

    ./al.sh $1 >> $globallog &&
    javac $javafile -classpath "runtimelib/boilerplate.jar" -d "runtimebuild" 2>> $globallog &&
    java -cp "runtimebuild:runtimelib/boilerplate.jar" $basename | tee $outfile &&
    diff -B $outfile $reffile | tee $difffile

    cat $outfile >> $globallog

    if [ $? -ne 0 ]
        then
        echo $1 FAILED
        error=1
    elif [ -s $difffile ]
        then
        echo $1 FAILED
        error=1
    else
        echo $1 PASSED
        ((passes++))
    fi
    ((tries++))

    printf "\n"
}

mkdir runtimebuild

```

```
chris@chris-HP-EliteBook-8460w ~/workspace/fai
./runtests.sh
mkdir: cannot create directory `runtimebuild':
testing tests/test01.java...
Success
Success
tests/test01.al PASSED

testing tests/test02.java...
Success
Success
tests/test02.al PASSED

testing tests/test03.java...
Success
Success
tests/test03.al PASSED

testing tests/test04.java...
Success
Success
Success
tests/test04.al PASSED

testing tests/test05.java...
Success
Success
Success
tests/test05.al PASSED
```

alltests.log

```
Buildfile: /home/chris/workspace/fail/build.xml

generate:
     [echo] generating parser source files...

compile:
    [javac] Compiling 6 source files to /home/chris/workspace

run-args:
     [java] Outfile: tests/test01.java
     [java] globals:[myAhoMachine]

run-noargs:

run:

BUILD SUCCESSFUL
Total time: 4 seconds
Success
Success
Buildfile: /home/chris/workspace/fail/build.xml

generate:
     [echo] generating parser source files...

compile:
    [javac] Compiling 6 source files to /home/chris/workspace

run-args:
     [java] Outfile: tests/test02.java
     [java] globals:[helloWorld]

run-noargs:
```

Project Management

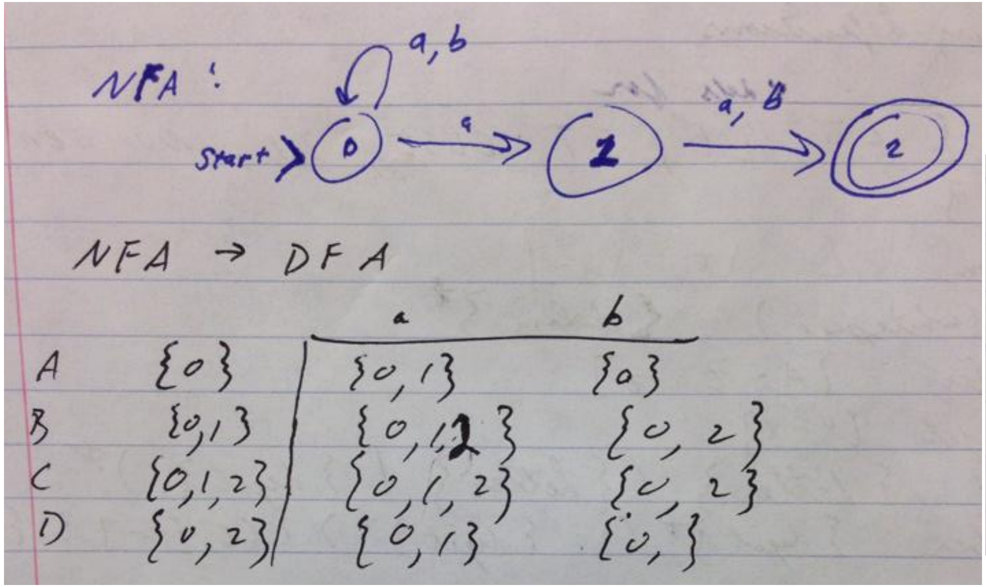
Project Timeline

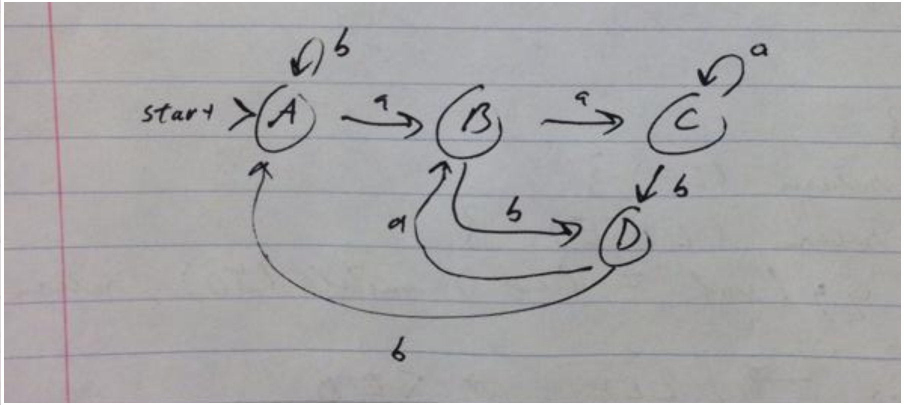
Timeline	Milestone
February 5	Brainstorm ideas for language
February 14	Decide on language and begin writing out rules, implementation, and use cases.
February 24	Finalize white paper
March 12	Write basic "Hello, World" program in AL and begin language tutorial and reference manual
March 19	Resolve any remaining confusing or controversial design decisions with TA and Language Guru (Dianna).
March 24	Finalize language tutorial and reference manual
April 20	Implement a minimal end-to-end compiler for basic 3 AL programs
May 1	Implement more sophisticated semantic checks, finalize translation rules
May 7	Regression testing
May 10-12	Debugging and final report

```
Need to make sure that variables dont get named things that would conflict in JAVA

  ---------------------------------
| SOURCE TO TARGET CONVERSION RULES |
  ---------------------------------

SOURCE: Given Automaton auto {}; we will need to make a block of code, define all the 'compo
initialize an automaton.
i.e. set up states, start, accept, and alphabet, the transitions, and then define the new ma

**MUST PUT THESE LINES AT THE TOP:
StateSet states;
StateSet accept;
Alphabet alphabet;
State start;

SOURCE: StateSet states = {name1, name2, name3, ...};
TARGET:
 - For each State (i.e. it is being defined for the first time) in the Set, a line must be c
    State name1 = new State("name1");
    State name2 = new State("name2");
    ...
 - Create and initialize StateSet states with every state defined so far.
    states = new StateSet(name1, name2, ...);

SOURCE: State name1;
TARGET: If this is encountered inside the automaton declaration, it is up to the user to hav
to be in their machine in the StateSet states = {...} line. So, we just have to initialize i
    State name1 = new State("name1");
```

Live Demo

(a|b)* a(a|b)





What We Learned

- Start early
- Modularize
- Split into even smaller groups
- Document
- Start early