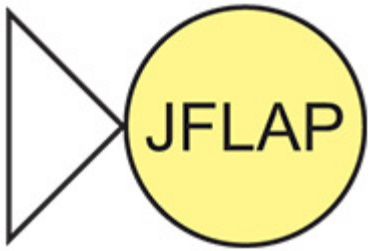
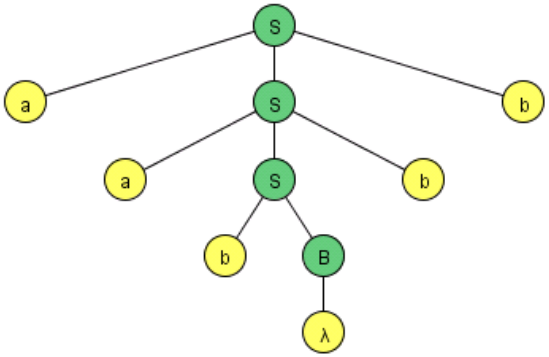
Increasing the Use of JFLAP in Courses



Susan H. Rodger
Duke University
Durham, NC USA
rodger@cs.duke.edu



Program Visualization Workshop
June 30, 2011

NSF Grants CCLI-0442513 and TUES-1044191

Co-Authors

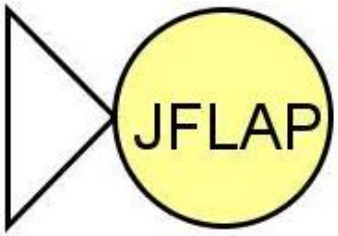
Henry Qin

Jonathan Su

Thanks to Students - Worked on JFLAP and Automata Theory Tools

Over 20 years!

- NPDA - 1990, C++, Dan Caugherty
- FLAP - 1991, C++, Mark LoSacco, Greg Badros
- JFLAP - 1996-1999, Java version
  Eric Gramond, Ted Hung, Magda and Octavian Procopiuc
- Pâté, JeLLRap, Lsys
  Anna Bilska, Jason Salemme, Lenore Ramm, Alex Karweit, Robyn Geer
- JFLAP 4.0 – 2003, Thomas Finley, Ryan Cavalcante
- JFLAP 6.0 – 2005-2008 Stephen Reading, Bart Bressler, Jinghui Lim, Chris Morgan, Jason Lee
- JFLAP 7.0 - 2009 Henry Qin, Jonathan Su

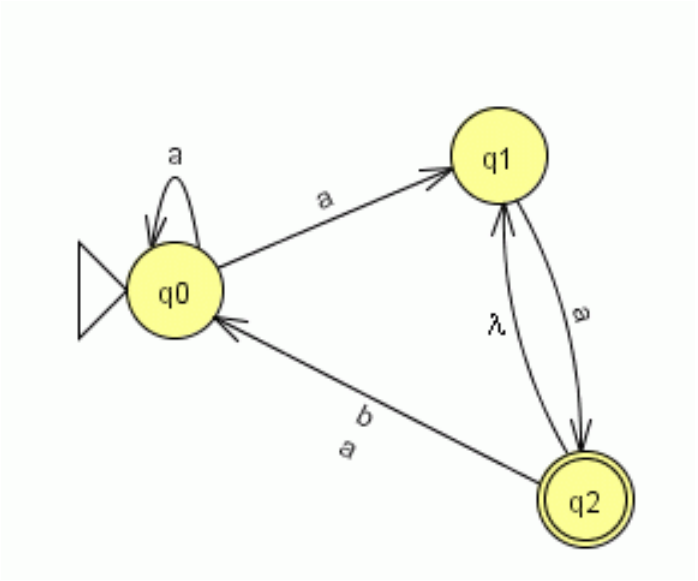


Overview of JFLAP

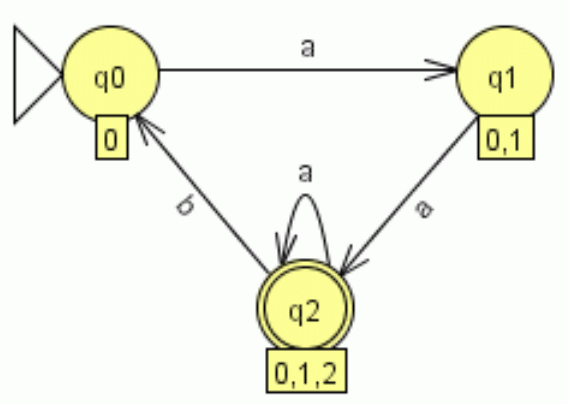
- **J**ava **F**ormal **L**anguages and **A**utomata **P**ackage
- Instructional tool to learn concepts of Formal Languages and Automata Theory
- Topics:
  - Regular Languages
  - Context-Free Languages
  - Recursively Enumerable Languages
  - Lsystems
- With JFLAP your creations come to life!

JFLAP – Regular Languages

- Create
  - DFA and NFA
  - Moore and Mealy
  - regular grammar
  - regular expression

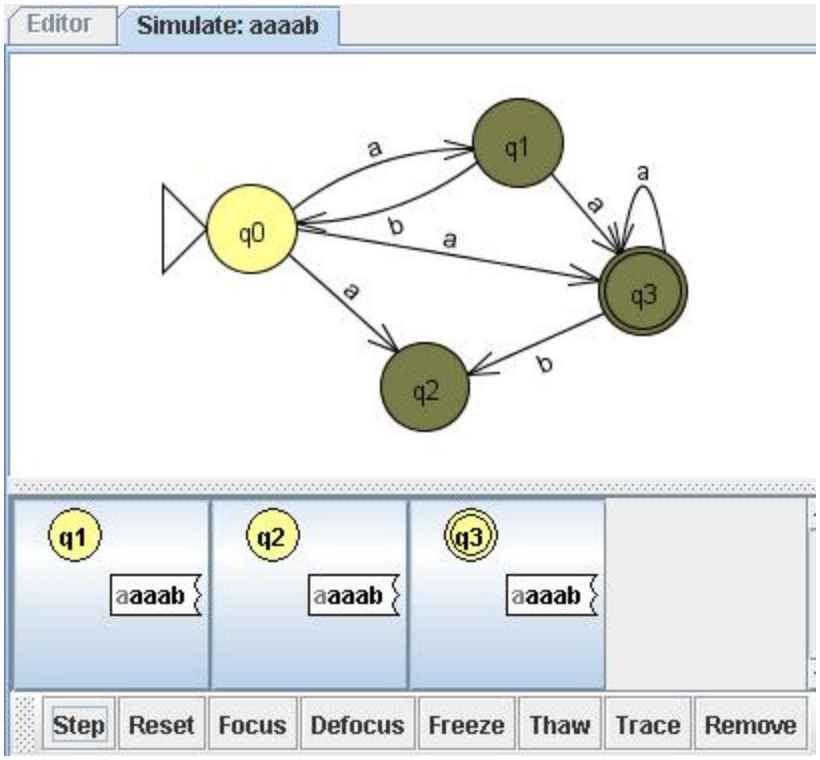


- Conversions
  - NFA to DFA to minimal DFA
  - NFA ←→ regular expression
  - NFA ←→ regular grammar



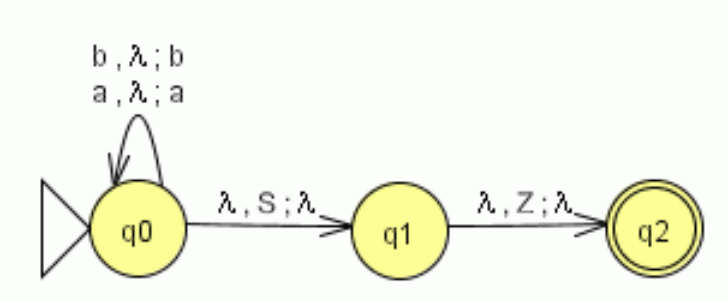
JFLAP – Regular languages (more)

- Simulate DFA and NFA
  - Step with Closure or Step by State
  - Fast Run
  - Multiple Run
- Combine two DFA
- Compare Equivalence
- Brute Force Parser
- Pumping Lemma

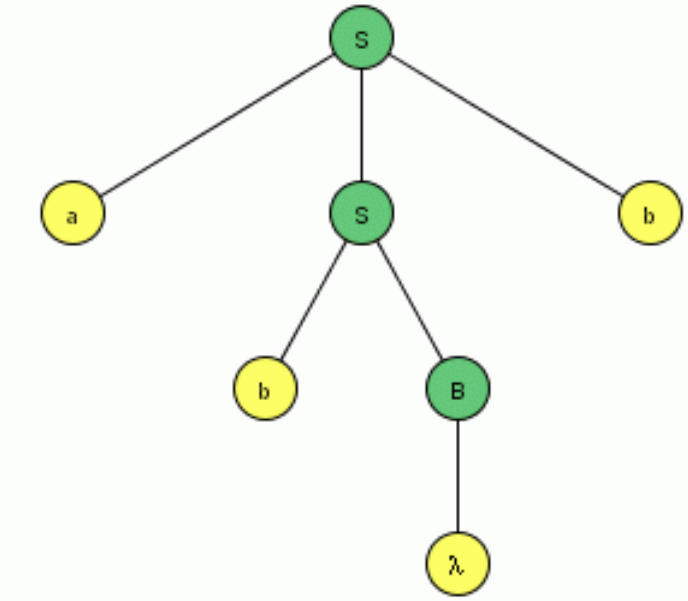


JFLAP – Context-free Languages

- Create
  - Nondeterministic PDA
  - Context-free grammar
  - Pumping Lemma

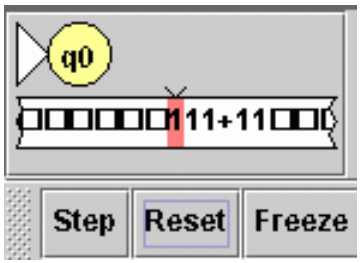


- Transform
  - PDA → CFG
  - CFG → PDA (LL & SLR parser)
  - CFG → CNF
  - CFG → Parse table (LL and SLR)
  - CFG → Brute Force Parser

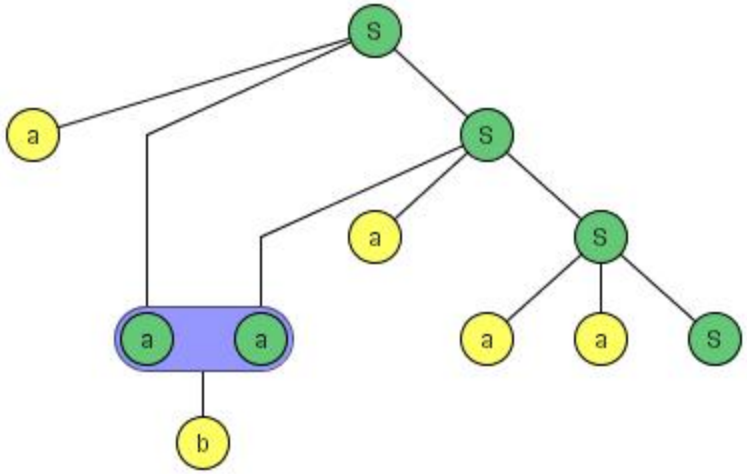


JFLAP – Recursively Enumerable Languages

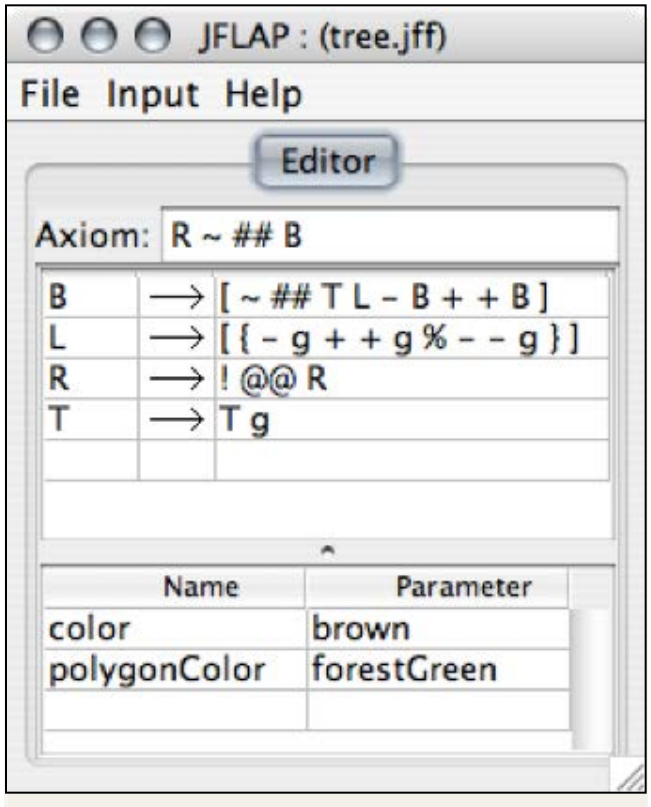
- Create
  - Turing Machine (1-Tape)
  - Turing Machine (multi-tape)
  - Building Blocks
  - Unrestricted grammar



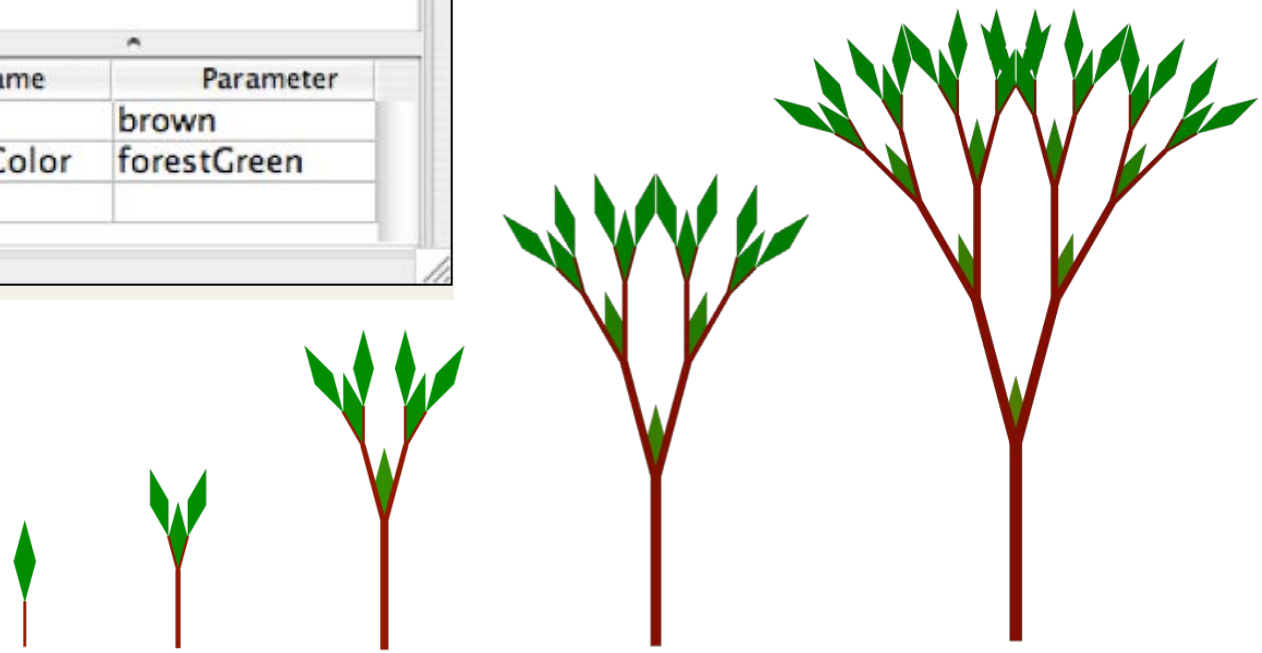
- Parsing
  - Unrestricted grammar with brute force parser



JFLAP - L-Systems



- This L-System renders as a tree that grows larger with each successive derivation step.



Goal – JFLAP defined generally

- Want JFLAP to fit with all textbooks
- Generalities
  - Empty string: preference - lambda or epsilon
  - FA
    - Single or multiple symbols
    - [1-9] on label to mean characters 1-9
  - PDA
    - Single or multiple symbols (restricted)
    - Accept by empty stack or final state

Goal – JFLAP defined generally (cont)

- Generalities (cont)
  - Turing machine preferences
    - Allow transitions from Final states
    - Accept by final state
    - Accept by halting
    - Allow Stay option (R, L, S) for moving tape head
  - Turing machine other
    - Single or multi tape
    - Building blocks – generalized transitions

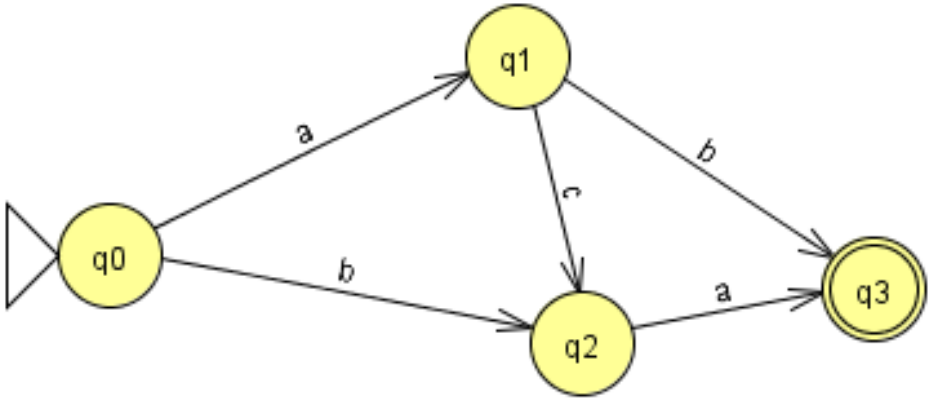
      a – read a, write a, do not move tape head

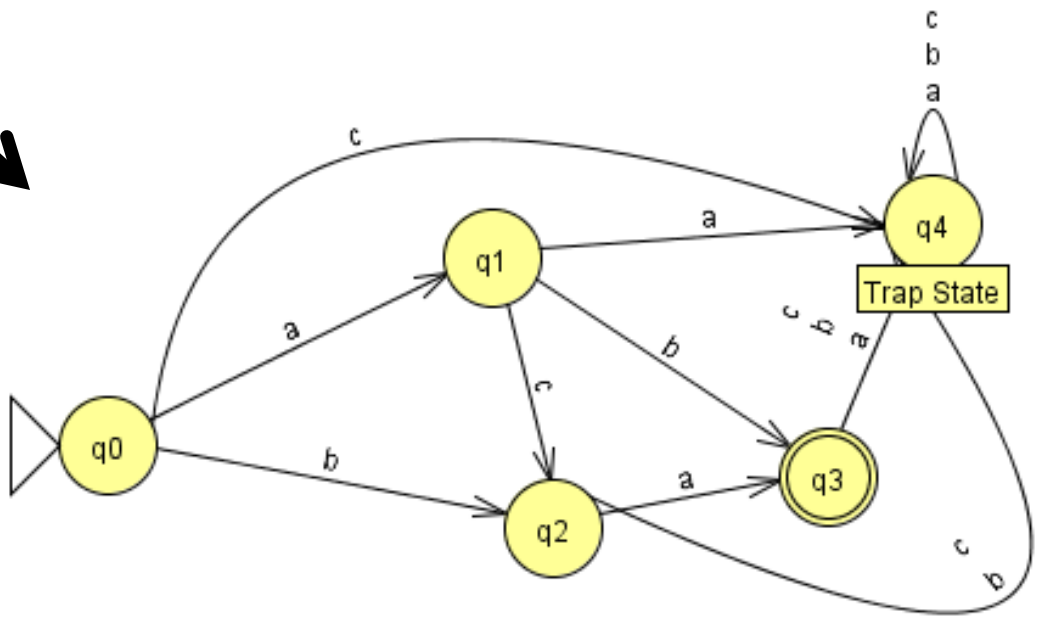
Other New changes to JFLAP

- Curve transitions
- Save images in several formats (png, jpg, gif, bmp, svg)
- Undo/redo capability
- Can zoom many of the panes

JFLAP Demos

DFA - Add Trap state and curve the arcs

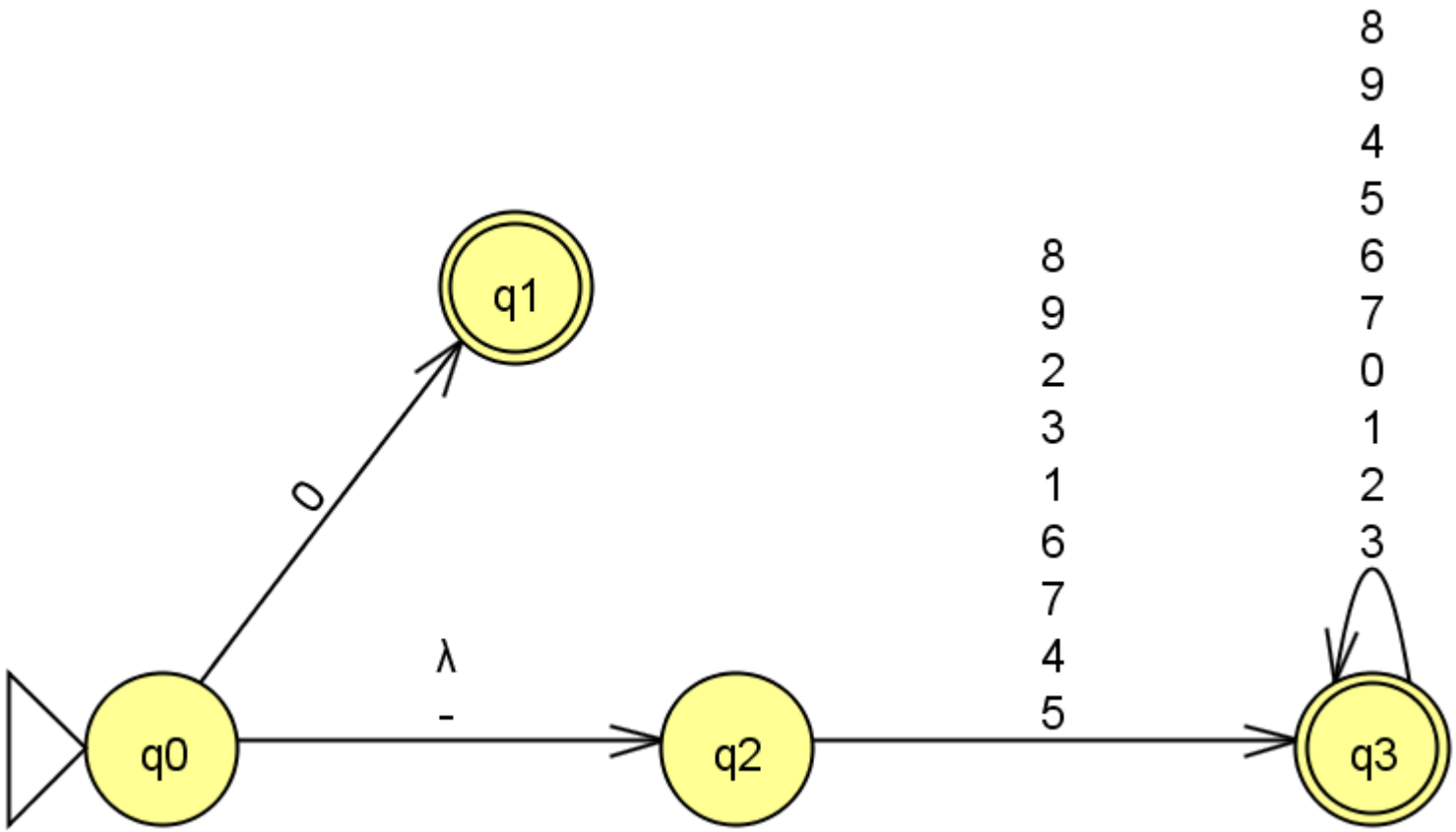




Example: Build an NFA for valid integers

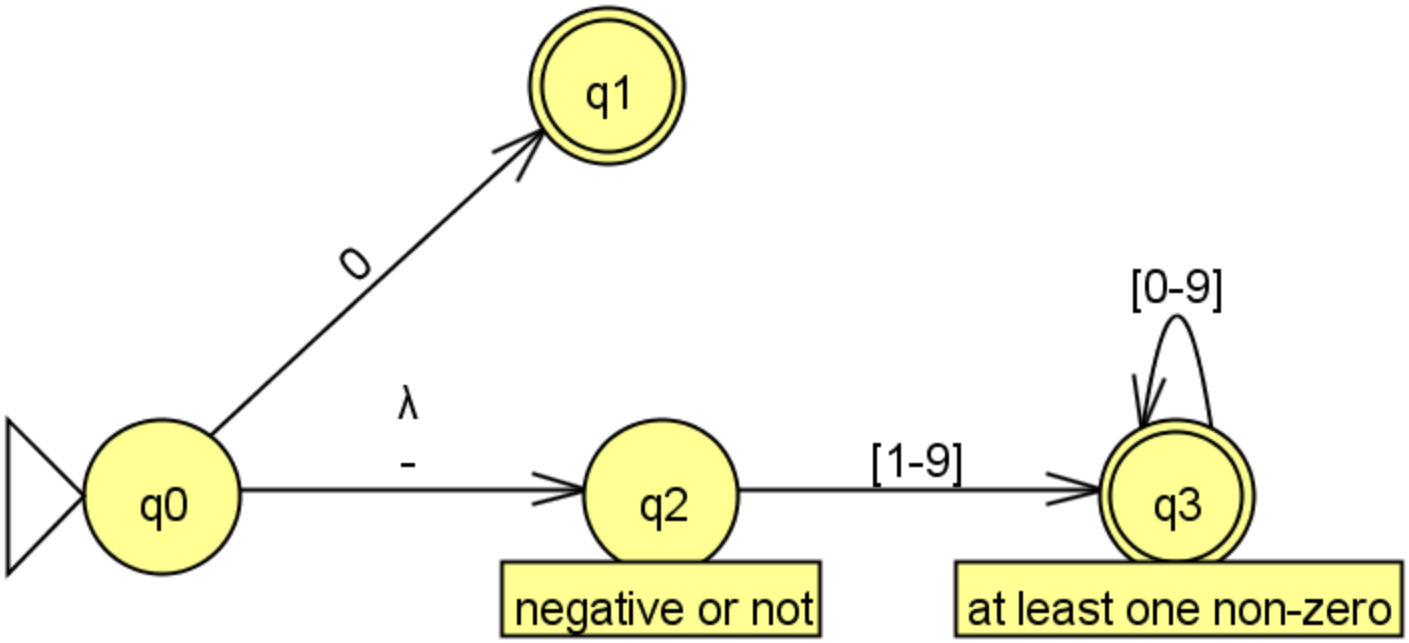
- Example:
  - Valid integers {-3, 8, 0, 456, 13, 500, …}
  - Not valid: {006, 3-6, 4.5, …}

Example: NFA for all valid integers



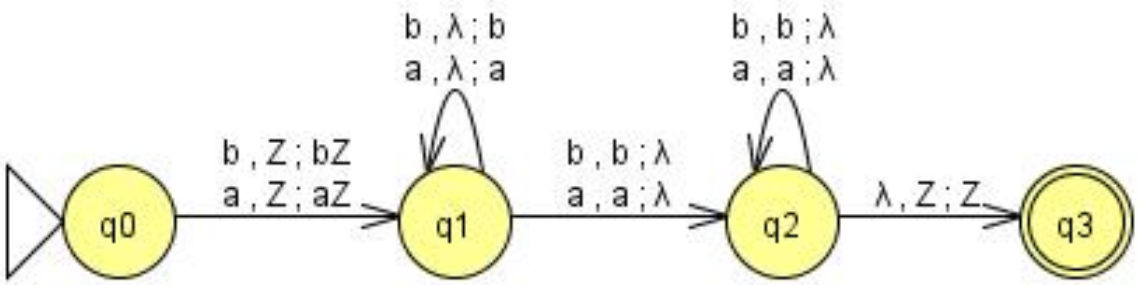
NFA annotated and shortcut

- New feature: [1-9] on labels



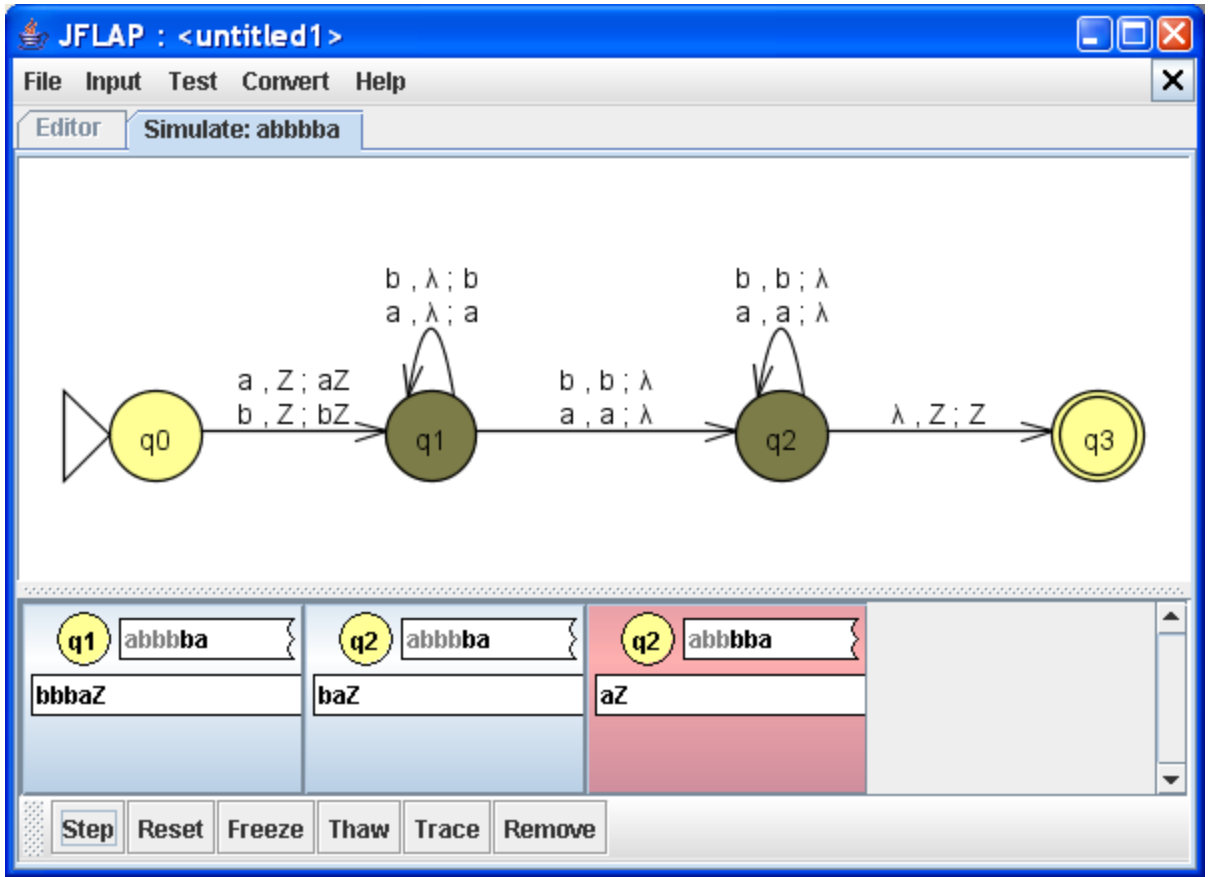
Example : NPDA

- NPDA for palindromes of even length
- New feature: asks for multi-char or single char for labels



Example (cont)

- Run input strings on the NPDA
  - Shows the nondeterminism



Example : JFLAP during Lecture

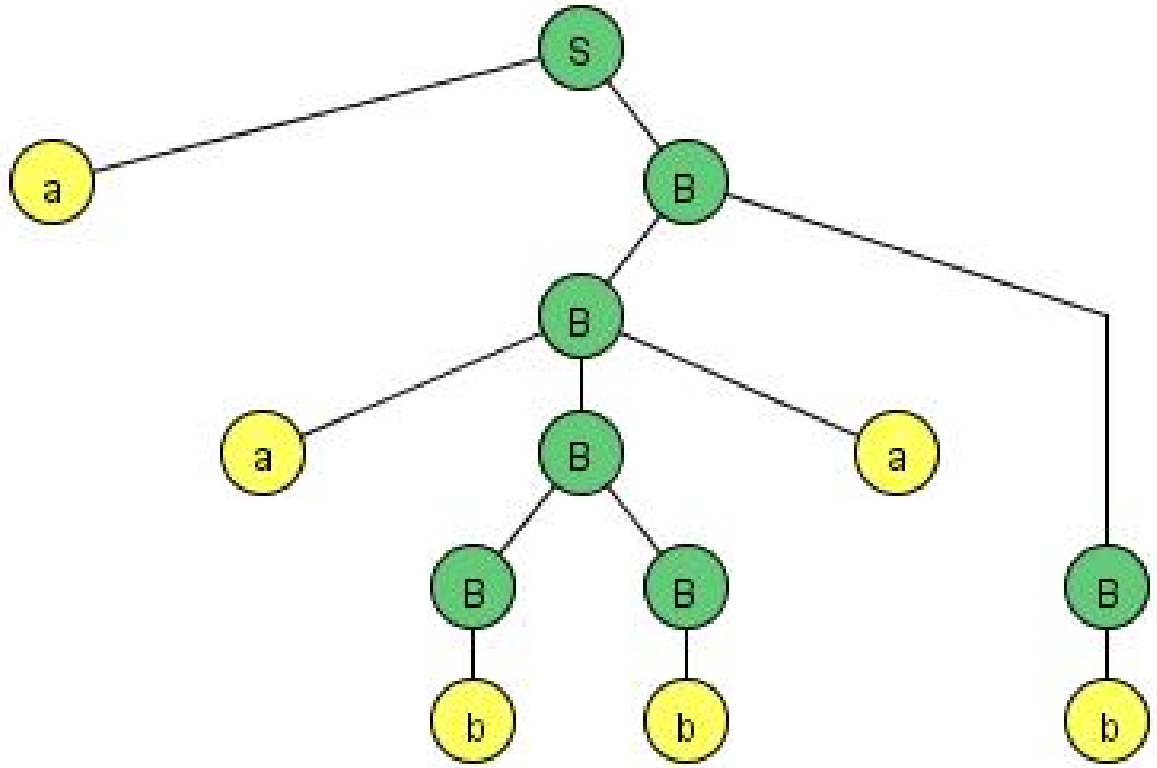
- Brute Force Parser
  - Give a grammar with a lambda-production and unit production
  - Run it in JFLAP, see how long it takes (LONG)
    - Is aabbab in L?
  - Transform the grammar to remove the lambda and unit-productions
  - Run new grammar in JFLAP, runs much faster!

S	→	aB
B	→	BB
B	→	aBa
B	→	b
B	→	λ

S	→	aB
B	→	BB
B	→	aBa
B	→	b
S	→	a
B	→	B
B	→	aa

Example 2 (cont)
Parse Tree Results

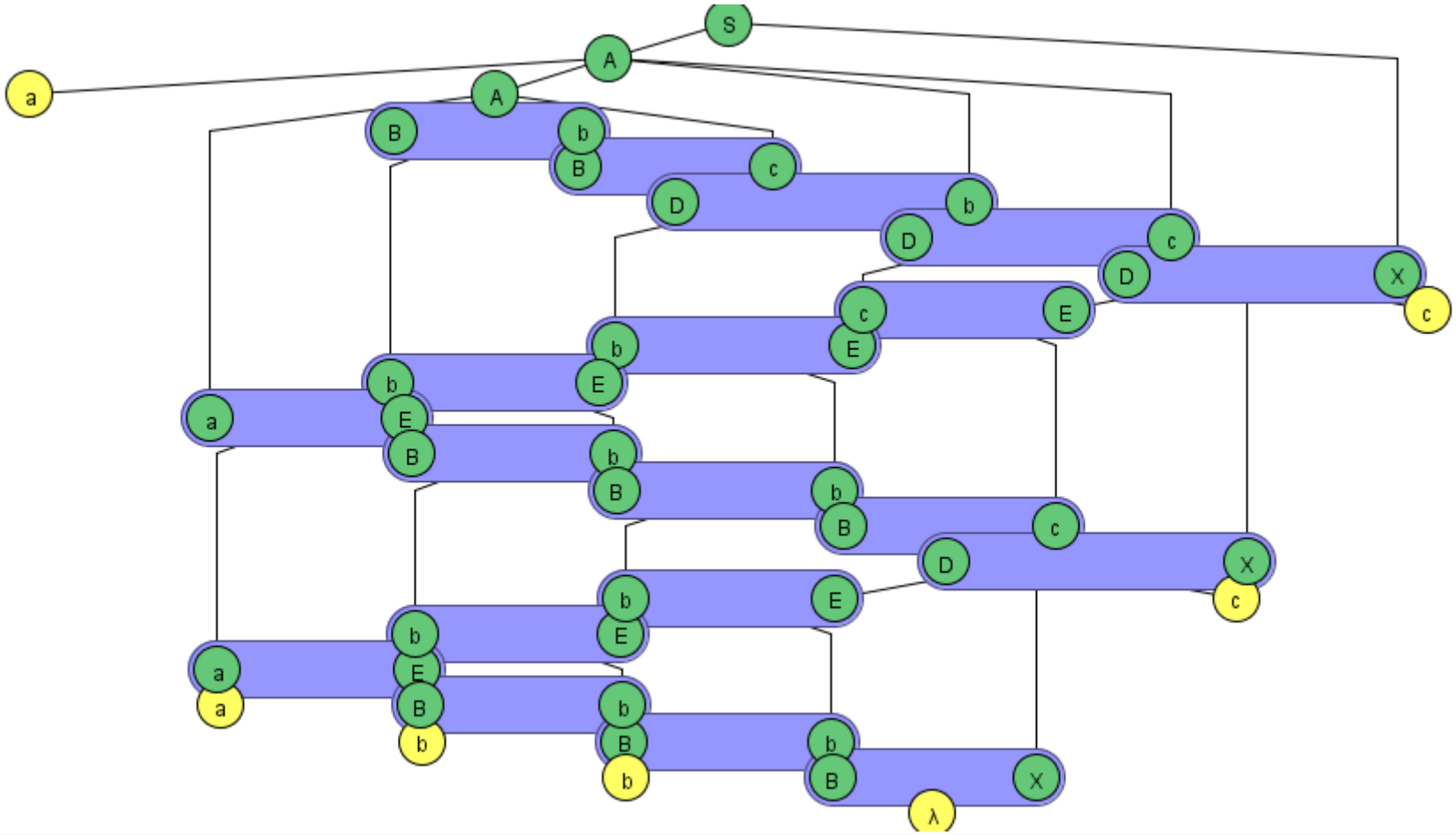
- First Grammar – 1863 nodes generated
- Second Grammar – 40 nodes generated
- Parse tree is the same.



Unrestricted Grammar - anbncn

S	→	AX
A	→	aAbc
A	→	aBbc
Bb	→	bB
Bc	→	D
Dc	→	cD
Db	→	bD
DX	→	EXc
BX	→	λ
cE	→	Ec
bE	→	Eb
aE	→	aB

Trace aabbcc



Two-year JFLAP Study
2005-2007

Fourteen
Faculty Adopter
Participants

-small, large
- public, private
- includes minority institutions

- Duke
- UNC-Chapel Hill
- Emory
- Winston-Salem State University
- United States Naval Academy
- Rensselaer Polytechnic Institute
- UC Davis
- Virginia State University
- Norfolk State University
- University of Houston
- Fayetteville State University
- University of Richmond
- San Jose State University
- Rochester Institute of Technology

Key Findings

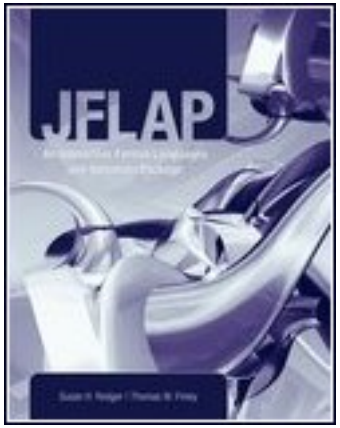
- All the faculty used JFLAP in their courses
  - They used it mostly for homework, some used it for class demonstrations.
- Students had a high opinion of JFLAP
- Four-fifths of the students thought JFLAP was easy to use to draw automata, simulate and interpret the results.
- The majority of students felt that having access to JFLAP made learning course concepts easier, made them feel more engaged in the course and made the course more enjoyable.
- Over half of the students used JFLAP to study for exams, and thought that the time and effort spent using JFLAP helped them get a better grade in the course.
- There was a control group in the second year, but the difference in knowledge between the control group and the JFLAP group was not statistically significant.

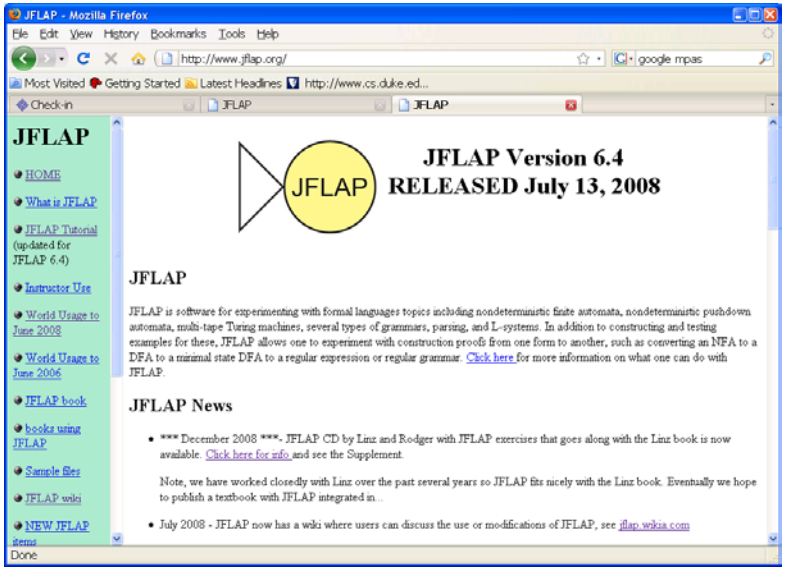
JFLAP Materials

www.jflap.org

JFLAP book

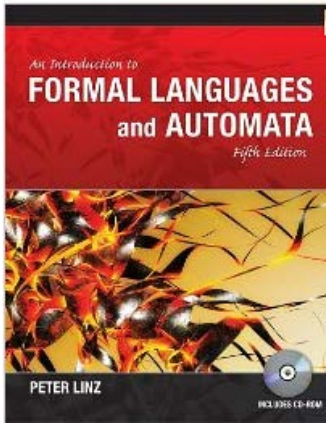
Use with automata theory textbook



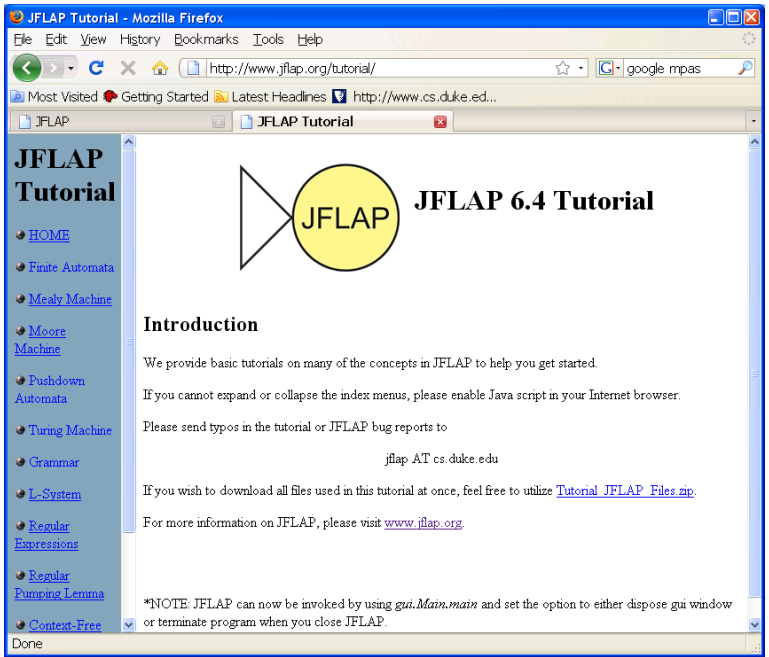


JFLAP online tutorial

JFLAP works well with Linz book



CD supplement with JFLAP exercises to go with this book



JFLAP is FREE

Future Work

Questions?