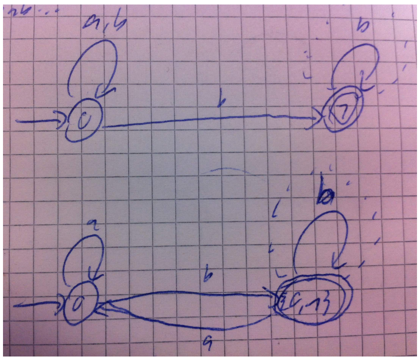
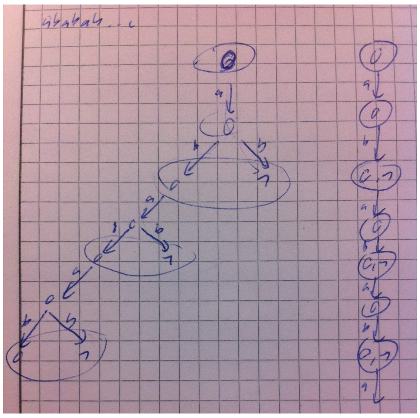
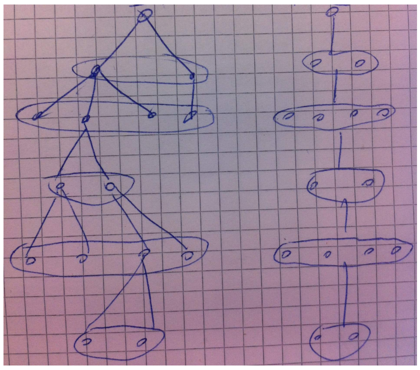
Master's Thesis Notes

Daniel Weibel, 10.04.2014

1 Theoretical Part

1.1 Basics

Subset construction on Büchi automaton	Run trees of word ababab... on nondet. and det. automaton	Impact of subset construction on run tree
		

The subset construction merges several nondeterministic runs to a single run. All the information about specific runs get lost: their state sequences, if and where they die, the number of specific runs until a given point, etc.

For automata on finite words this is not a problem, because all we are interested in are the states the automaton is in at a certain point in time.

With ω-automata, however, we are interested in the behaviour of *specific runs*. In the case of Büchi automata we need to know whether a specific run visited an accepting state infinitely often.

As mentioned, this information gets lost with the subset construction. All that we can say after the subset construction is that an *accepting state q* has been visited *infinitely often*. This can have three reasons:

1. A finite number of runs visited q *infinitely* often
2. An infinite number of runs visited q *infinitely* often
3. An infinite number of runs visited q *finitely* often

In Cases 1 and 2 we have the case that there exists a specific run that visited an accepting state infinitely often. This satisfies the Büchi acceptance condition. In Case 3 however, there is no specific run that visited the accepting state q infinitely often. This is the case in the above example.

After the subset construction we cannot distinguish Case 3 from Cases 1 and 2 anymore. Hence, whereas the nondeterministic Büchi automaton correctly accepts only Cases 1 and 2, the determinised automaton additionally accepts Case 3.

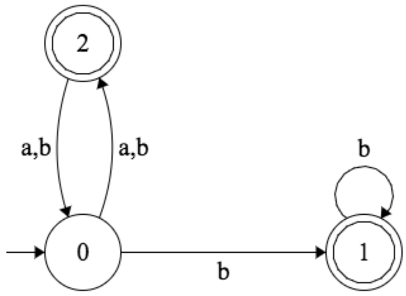
What if in the subset construction accepting and non-accepting states are never mixed?

Lemma: if accepting and non-accepting states are not mixed in subsets, then the above problem with the subset construction does not occur.
Proof: subset state q = {q1,q2,..,qn}, with q1,..,qn accepting, visited infinitely often → there exists a specific run that visited at least one of q1,...,qn infinitely often.
There exists an infinite path (run) in the run tree (König's Lemma). This run necessarily visits subset

state q infinitely often (one component of q at a time). Since the number of component states of q (q1,...,qn) is finite, the run must visit at least one of them infinitely often.

1.2 Questions

1.2.1 In which cases accepting and non-accepting states can be mixed, or what happens if they are mixed?



Example: mix states 0 and 2 into state q = {0,2}. In the end it must hold that if q is visited infinitely often (implying that both 0 and 1 are visited infinitely often), then there is a single run that visited 2 infinitely often. Which approach to take?

1.2.2 Extending the information from 1 towards complementation

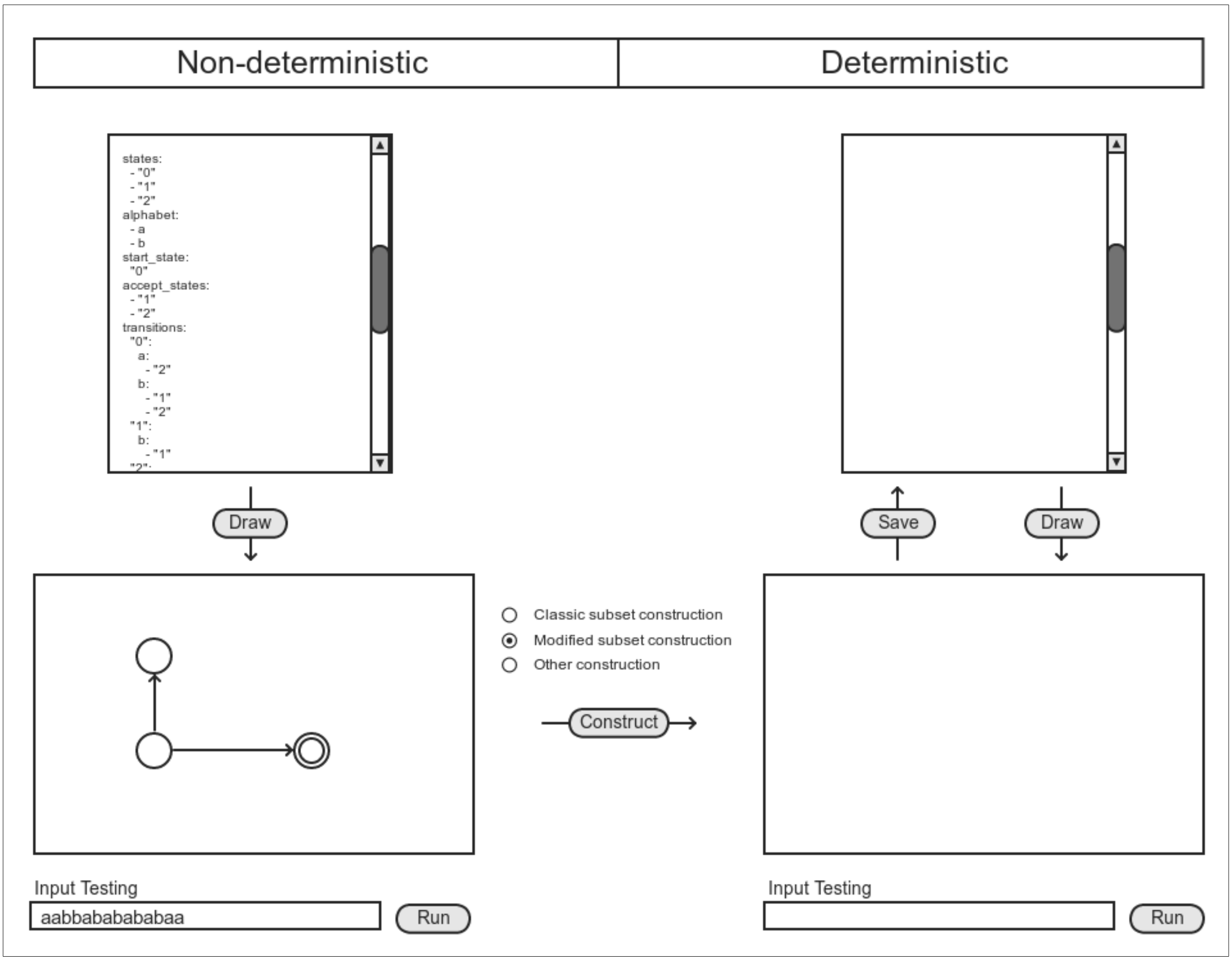
Standard subset construction on Büchi automaton A would give the picture:

	Nondeterministic		Deterministic
Language	L(A)	⊆	L(D)
	||		||
Complement language	L(A')	⊇	L(D')

Part 1 is about the first row. What's the idea to arrive finally at A'?

2 Practical Part

- Existing Ruby program
- Web-based application
- More functionality
    - For NFA, drawing run tree of a given input word


Non-deterministic
Deterministic
states:
- "0"
- "1"
- "2"
alphabet:
- a
- b
start_state:
"0"
accept_states:
- "1"
- "2"
transitions:
"0":
a:
- "2"
b:
- "1"
- "2"
"1":
b:
- "1"
Draw
Save
Draw
Classic subset construction
Modified subset construction
Other construction
Construct
Input Testing
aabbababababaa
Run
Input Testing
Run