**THE AUTOMATA SHOPPING FOR ONLINE SHOPPERS**

****

#### **Part 1 - Introduction (design)**

For my final project for EECS 510, I decided to create a non-trivial formal language that relates to online shopping, as in my free time away from school, I like to go shopping for random things, which may include clothes or my regular grocery essentials, usually in an online format because I don’t have a lot of time to go to the store to look for things for a certain amount of time as there may be more options than what the store has. I plan to apply this to the material we have covered in this class and it has to do in part of any online shopping experience where many factors come into play. To keep things simple for the sake of the class, I will only be doing what is common for anyone that does shopping online and ensures that it makes sense for everyone. The transition symbols will include a start, add/delete (items), and ready to checkout and leave states. This will result in multiple states including (q0, q1, …….). This will be based on a regular grammar that is explicit and easily readable to understand.

#### **Part 2- Grammar**

So I am using an NFA (regular language) to create this. I initially tried to do something complex that may involve coupons or a hold, but since it may be beyond an automata practice, I decided to keep things simple and chose scenarios that include multiple self loops and processes. In every instance of a store’s website, there is a cart and while people are searching, they may get something that interests them and will select “add to cart” before checkout. Due to the nature of all distinct items that may be added (ex, food, clothes, housewares, coupons, etc.), I am only going to be using a few symbols (actions) that may be done to any product and consumer. The intended grammar is only going to be a subset since every shopping experience is different where someone could order from a few to a ton of items (which would result in a very long string with the same rules)

**(right grammar rules)**

**q0→ eq1**

**q1 → aq2**

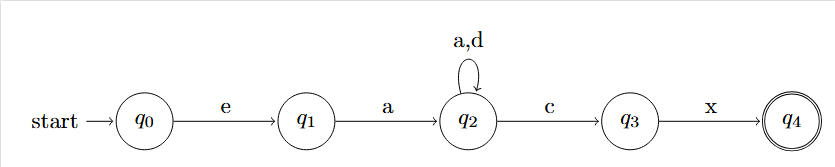
**q2 → aq2 | dq2 | cq3**

**q3 → xq4**

**q4 → λ (accepting)**

#### **Part 3 - Automaton**

The actions that are fair game for the string include: enter, start shopping, adding items, checking out, and leave (exit). There may be more things but to keep a general sense of what online shopping is, again, only a subset is used here, hence there will not be anything too complex. The input symbols will be: {e,a,d,c,x}



A couple sample strings may include “eaadaaacx”, eaacx (valid) or even eaap or cpx (invalid). There are some restrictions that have to be met since you can’t purchase anything if nothing is in your cart (ex: empty strings meaning empty cart). This includes exactly one “E” and at least one “a”. There can only be once “x” and “c” as you can’t exit or checkout twice, and you can’t leave without going through the checkout, hence the single element transition which results in a string that is impossible. [instances: eaadccxx]

The automation shown above (and in the folder) is a simple NFA finite state machine and on every transition, depending where you are, it will either move forward or loop around indicating the string can be as many lengths as it needs to be and obviously will have a start with an “e” (enter and start shopping) and accepting at the end with an “x” to leave the site/complete the transaction. This is based off of the similar NFA’s we’ve done in class. As explained, there may be very long strings, which replicate someone buying a full cart or very short ones–which are only a few items.

#### **Part 4- Data Structure**

**q0 q1 q2 q3 q4 (these are the states that are used in the NFA)**

**e a d c x ( These are the appropriate input symbols for a string generating)**

**q0 (start state)**

**q4 (accepting state)**

**(All of the valid transitions are here!)**

**q0 e q1**

**q1 a q2**

**q2 a q2 (loop)**

**q2 d q2 (loop)**

**q2 c q3**

**q3 x q4**

#### **Part 5 - Testing**

Please visit this [link](https://colab.research.google.com/drive/1UVVAaDXvSmo-iXEf6q3sMJ9-KsBpSMtd#scrollTo=MzFncJNaxG-F) to view the code and try a couple of test strings for this language or view the “.py” file in the folder if CoLab doesn’t work for you! (same code regardless). To run the code in CoLab, press the play button and you will be asked to type in a string. Enter “exit” to stop the program (don’t press the STOP button on the left side of the code because it causes an interrupt). I have made the program to see if the string is a reject, acceptable or invalid (unidentified inputs).

**Here are a couple sample strings with their transitions and expected outputs!!**

**Accept Reject**

**q0 e q1 q0 e q1**

**q1 a q2 q1 p q2**

**q2 a q2 q2 p q3**

**q2 a q2 q3 x q4**

**q2 d q2**

**q2 a q2 \*\*This is impossible due to no “p” symbol”, hence we can’t leave!**

**q2 c q3**

**q3 x q4**