13^{th} Annual Johns Hopkins Math Tournament
Saturday, February 18, 2012
Explorations Unlimited Round – Automata Theory

1. Introduction

Automata theory is an investigation of the intersection of mathematics, formal logic, and computer science which studies how abstract machines work. But what is an abstract machine?

Clearly, a machine has to perform some task in a predictable way. For example, a toaster is a machine that makes toast, and you expect that it will not instead act from time to time as a dishwasher. Much like the toaster or the dishwasher, there needs to be some sort of input: bread, dishes... And an output: toast, clean dishes... In addition, we want to require "a user manual" that tells you how your machine takes the input and produces an output. Mathematically, if we wished to define a machine, we ask that it has an **input**, a **formal set of rules** for handling this input, and an **output**. This is what we mean by an abstract machine.

Why do we wish to study such things? Because this turns out to be a very general and a very powerful idea which has various modeling applications. For example, we can use automata to model the operation of a vending machine, a system of tollbooths or even entire economies. Here, we will look at the modeling applications of one type of automata, **Vector Addition System with States (VASS) automata**.

2. Definitions and Notation

We begin with some technical definitions and then present examples.[1]

Definition 2.1 (4-tuples)**.** We call (a, b, c, d) a **4-tuple**, where a, b, c, d are all integers.

Definition 2.2 (non-negative 4-tuples)**.** Define a **non-negative 4-tuple** as a 4-tuple where a, b, c, d are all *non-negative* integers.

Definition 2.3 (VASS Automata Diagrams)**.** An automaton diagram consists of the following components:

(1) Vertices:
- There are **vertices** (drawn as circles).
- There is an **initial vertex**. (drawn as a doubled circle).

(2) Directed Edges:
- There are **directed edges**, which are like one-way streets from a vertex to a different vertex, in a specified direction (on the drawing, it is indicated by the direction of the arrow). Each directed edge has an associated 4-tuple which is the **cost of transition**.
- There are **loops**, which is a vertex with a directed edge going from itself to itself. This is treated just like going from one vertex to another (see below).

(3) Starting State:
- There is a **starting state**, which is a non-negative 4-tuple that is given. (placed inside a rectangle next to the initial vertex)

Definition 2.4 (Automata Dynamics)**.** Given a VASS automata diagram, a move to another vertex on the diagram is permissible from the initial vertex if and only if:

(1) There is a directed edge between the initial and final vertices.
(2) The move is in the direction of the directed edge.
(3) Your first move must start at the initial vertex with the initial 4-tuple.

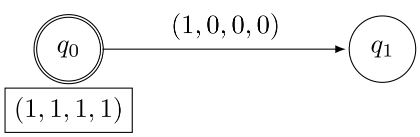
[1] It may benefit the reader to skip ahead and then come back to these definitions.

(4) If your starting 4-tuple is (a, b, c, d) and the directed edge you are traveling through has 4-tuple (w, x, y, z), then your 4-tuple after the transition will be $(a + w, b + x, c + y, d + z)$. This new 4-tuple is what you will use for further transitions. **You can move if and only if all entries are non-negative.**

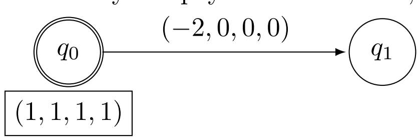
Another way to view this is to imagine you are given a certain strictly positive amount of 4 different kinds of currencies at the beginning of a trip. You use these currencies to purchase travels between countries (your vertices). We can represent this currency supply as a 4-tuple (w, x, y, z). Each path you take will charge you or give you a specific amount of each of these currencies, represented as another 4-tuple (a, b, c, d). So each time you take an edge, you move to the next vertex and update your initial 4-tuple by $(w+a, x+b, y+c, z+d)$.

Consider the following examples.

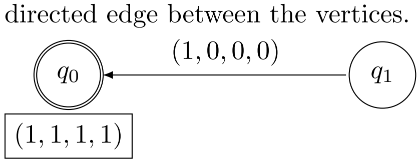
On the diagram below, it is permissible to go from q_0 to q_1, because the direction agrees with the directed edge and the final 4-tuple is $(2, 1, 1, 1)$, which satisfies the payment restriction.



Here, it is *NOT* permissible to go from q_0 to q_1, because the final 4-tuple would be $(-1, 1, 1, 1)$, which would not satisfy the payment restriction, as $-1 < 0$.



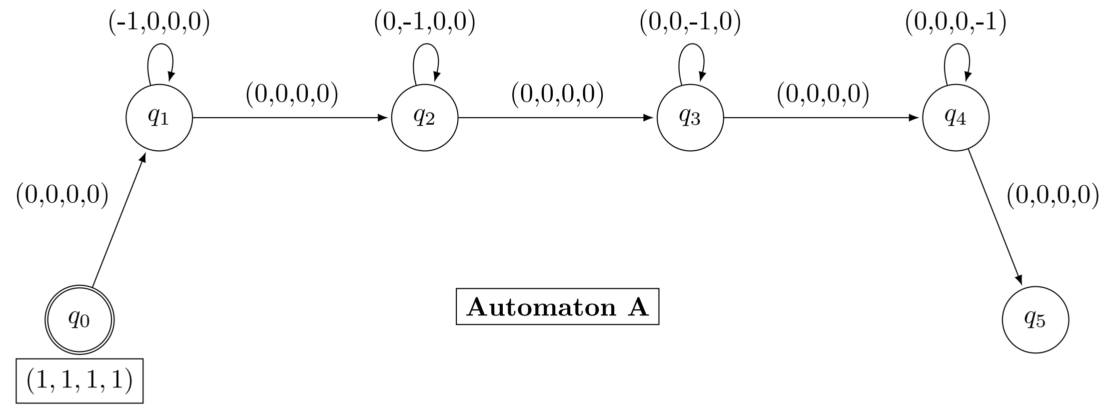
And here, it is *NOT* permissible to go from q_0 to q_1, because the transition is not in the direction of the directed edge between the vertices.



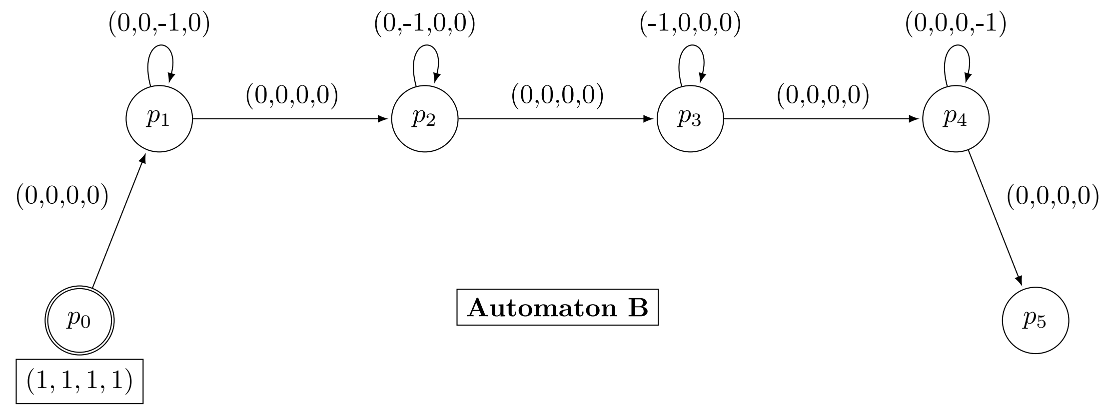
3. Modeling Trading

VASS Automata can be used to model certain economic trading patterns. Suppose you are planning a commercial trip. You move between countries, selling and buying goods. You will start with 4 different types of goods, a, b, c, d, and we have some non-negative quantity of each. We can write what we have as a 4-tuple (a, b, c, d). For example, suppose we have $(1, 1, 1, 1)$. Now suppose that you never want to be in debt- that is, you never want to owe someone goods- meaning that your 4-tuple should always be non-negative. You can use a VASS automaton to represent this. At q_1, you sell type a goods. At q_2, you sell type b goods. At q_3, you sell type c goods. At q_4, you sell type d goods. Now, when you move between countries, there could be a cost of transition. Perhaps you collect 'free' goods on your way to a particular country (fishing), or perhaps you have to give up some goods to enter (taxes).

We can assume we have a starting state: $(1, 1, 1, 1)$. In this example, this could mean 1 ton of plywood, 1 ton of foodstuffs, 1 ton of chemicals, and 1 ton of scrap metal. In this example the transitions between countries do not alter our 4-tuple; we neither gain nor lose goods on the way.



Now look at this diagram:



Intuitively, from a trading perspective, we know that the two automata are modeling the same type of route: there is no incentive to pick one over the other given the information presented here. We can formalize this intuition with the notion of **isomorphic automata**.

Definition 3.1. Let X and Y be sets. We say that a function $f : X \to Y$ is:

injective: if for all x, z in X, $f(x) = f(z)$ implies $x = z$. (also called one-to-one)

surjective: if for all y in Y, there exists an x in X such that $f(x) = y$. (also called onto)

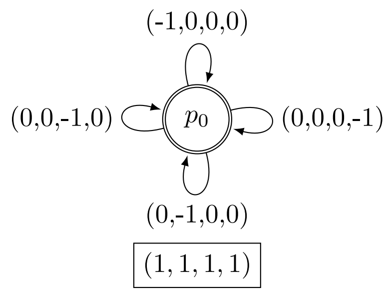
a bijection: if it is both injective and surjective.

Definition 3.2 (isomorphic automata)**.** Two automata are **isomorphic** if there is a map that is both a bijection between their vertices, and that preserves the transitions between any two vertices.

Problem 1. Prove that any surjective function between finite sets of the same cardinality is a bijection. (5 points)

Problem 2. We can now return to automaton B. Is B isomorphic to A? Prove your assertion. (5 points)

Problem 3. Consider automaton C, represented below. Prove that it is *not* isomorphic to A. (5 points)



Automaton C

4. Equivalence between automata

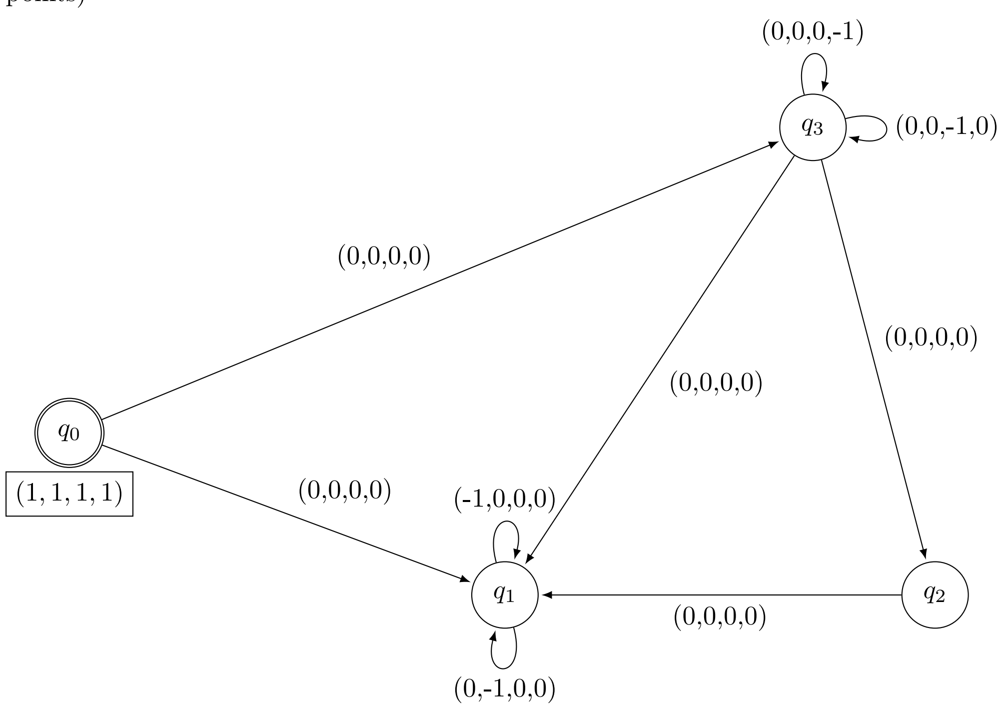
Even though automaton C is not strictly isomorphic to automaton A, we nevertheless observe that it behaves in a very similar way. This leads us to work with a new concept, that of equivalent VASS automata.

Definition 4.1 (equivalent automata)**.** We say two automata are **equivalent** to each other if, when we apply the following rules, they are isomorphic.

(1) Any $(0, 0, 0, 0)$ loops are removed.
(2) Any $(0, 0, 0, 0)$ transitions are removed; the two vertices affected by the removal of a particular transition are merged together into one new vertex, which maintains all of the other loops and transitions that BOTH vertices used to have, apart from the removed one.

Problem 4. Apply rules (1) and (2) of definition 4.1 to Automaton A to simplify the drawing. (Sketch it) (5 points)

Problem 5. Is automaton D, represented below, equivalent to Automaton A? Prove your assertion. (5 points)



Automaton D

Problem 6. Consider the set of all VASS automata equivalent to A. Call this set Ξ. Now consider the set of all isomorphic VASS automata to A. Call this set Υ. Is Ξ a subset of Υ? Is Υ a subset of Ξ? (Prove or disprove) (6 points total, 3 per each part)

5. Equivalence relations

In mathematics, it is often the case for objects to be superficially different but still deeply related, and it is useful to "link" them together. One way to do this is through the use of equivalence relations.

Definition 5.1. An **equivalence relation** is a relation $\sim$ on a set A that satisfies the following rules for all a,b,c, in A:

(1) Reflexive: $a \sim a$
(2) Symmetric: if $a \sim b$ then $b \sim a$
(3) Transitive: if $a \sim b$ and $b \sim c$ then $a \sim c$.

You are already familiar with several such relations: $=$ on the set of integers and $\equiv \mod n$ also on the set of integers are two such examples.

Let us state the first as an example. For $a, b \in \mathbb{Z}$,

(1) $a = a$
(2) $a = b$ implies $b = a$
(3) if $a = b$ and $b = c$, then $a = c$. Therefore "=" is an equivalence relation on $\mathbb{Z}$

Problem 7. Prove that "is-isomorphic-to" is an equivalence relation on the set of all VASS automata. (9 points total, 3 points per property)

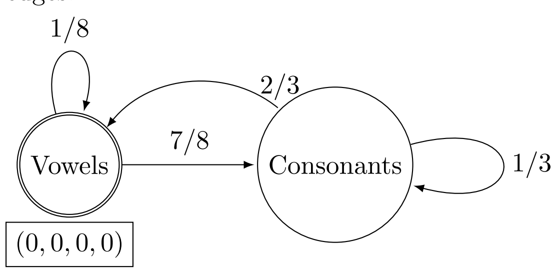
Problem 8. Prove that "is-equivalent-to" is an equivalence relation on the set of all VASS automata. (9 points total, 3 points per property)

Problem 9. Given injective functions $f : A \to B$ and $g : B \to A$, show that there exists a bijective function $h : A \to B$. (20 points).

6. Probabilities and VASS models

Now that some of the theory has been established, we can start using automata to represent probability problems. In the same way that we modeled the machine using a VASS automaton, we can condense all of the information of a particular probability problem into something that is easy to work with using a special kind of automaton.

Problem 10. Here is one such example:
In 1903, A. A. Markov, reading Pouchkine's poem "Eugene Oneguinde", remarked that the consonants and vowels succeeded one another according to the following rule, represented here by an automaton. This particular automaton works exactly like a VASS automaton, with the sole difference that you do not get to choose which vertex to go to; it is determined with the probability indicated on the diagram next to the edges.



The first letter of the poem is a vowel.

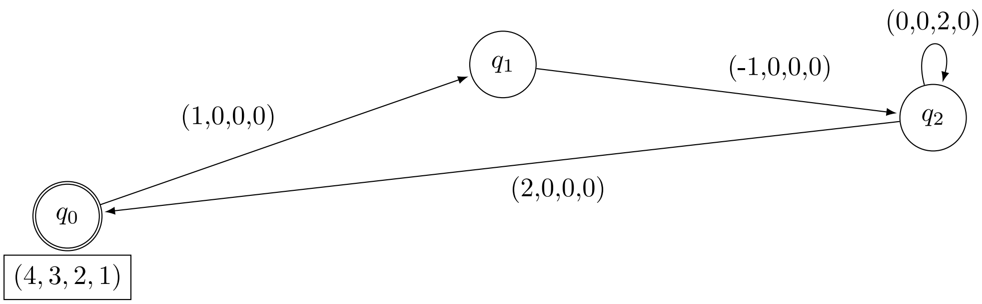
(1) What is the probability that the 2^{nd} letter is a consonant?
(2) What is the probability that the 3^{rd} letter is a consonant?

(6 points, 3 points for each part)

7. The reachability problem

Continuing with the basic theory, there is an important and general question in automata theory called the reachability problem, and the following is the VASS automata version: given a starting 4-tuple (a, b, c, d) and some other 4-tuple (e, f, g, h), can a given automata ever reach this other 4-tuple (e, f, g, h)? We can phrase this in terms of what it might mean for a particular model: given what we know about a system *right now*, we can ask if it is ever possible for the system to evolve into some other state that we might have in mind, e.g., given that we are feeding dollar bills into a vending machine, will the machine ever switch from a neutral state into a state where it gives us nutritious and delicious snacks?

Problem 11. We consider a particular case of the *reachability problem*: can one ever reach $(4, 4, 4, 4)$, given the following automaton? The starting 4-tuple is, as indicated, $(4, 3, 2, 1)$.



If so, give an explicit path to take to reach it in terms of the nodes to visit. (in other words something like "go from A to B, to B again, etc..") If not, explain why such a configuration can never be reached. (5 points)

Problem 12. Same question as Problem 11, but with $(1, 4, 0, 4)$ as the goal, instead of $(4, 4, 4, 4)$. (5 points)

Whether or not there is an algorithm for solving the reachability problem for general VASS automata is a famous problem. While the entire proof is very complex, we will consider an important (but simple) piece of the proof.

Problem 13. Prove that if the 4-tuple (a, b, c, d) can be reached by some automata, then a, b, c, d must be non-negative. (5 points)

8. Cycles and simple graphs

Having worked on the previous problem, you will probably have noticed the need for a technical term for "repeat the above moves from vertices to vertices". The definition we need is that of a **cycle** from graph theory. This is a small example of the many links between graph theory and automata theory. We can consider a VASS automaton as a "living graph"; a graph with rules.

Definition 8.1. A **simple graph** G is a pair of sets $G = (V, E)$, where V is a set of vertices and E is the set of edges of the graph. In a simple graph there is at most one edge between two vertices, and there are no edges connecting a vertex to itself. (Note that unlike a VASS automaton the edges in a simple graph are *not* directed.)

We say $u, v \in V$ are **adjacent** vertices in a graph $G = (V, E)$ when they are linked by an edge. We can now return to our thoughts about **cycles**.

Definition 8.2. A **cycle** on a graph G is a path between vertices (and along edges) such that:

(1) The path is closed. In other words, the walk returns to the initial vertex, in a closed loop.
(2) The path can only visit a single vertex once. This implies no self intersections.
(3) Vertices may be added to the path only if they are adjacent to the previous vertex.

Theorem 8.3 (*Konig, 1936*)**.** *Given a simple graph G, you can assign a color to every vertex of G such that no adjacent vertices have the same color, using only 2 colors, if and only if G has no cycles with an odd number of vertices.*

Problem 14. Prove theorem 8.3. (30 points total, 15 for each implication)

Definition 8.4 (graph isomorphism)**.** Simple graphs $G = (V, E)$ and $H = (V', E')$ are **isomorphic** if there is a bijection $f : V \to V'$ such that $f(u)$ and $f(v)$ are adjacent in H if and only if $u, v \in V$ were adjacent in G.
That is, to show that two simple graphs are isomorphic we must find a bijection between their vertices such that adjacency is preserved. So two adjacent vertices in G must map to two adjacent vertices in H.

Problem 15. Give an example (drawings will suffice; use the same notation as for VASS automata, i.e. circles for vertices and lines for edges) of two simple graphs that have the same number of vertices and the same number of edges but are not isomorphic. Prove your assertion. (20 points total, 10 points for drawing, 10 points for proof)

9. The link with self-complementary graphs

Two final definitions:

Definition 9.1. Given a simple graph G, we define the **complement** $\bar{G}$ as the graph with the same vertices as G, but where every pair of adjacent vertices in G are now non-adjacent pairs, and where every pair of non-adjacent vertices in G are now adjacent pairs.

Problem 16. What is the embedding in $\mathbb{R}^3$ of the drawing of the complement of the graph whose vertices and edges are those of a cube? (15 points)

Definition 9.2. (self-complementary graphs). We say G is self-complementary if and only if G is isomorphic to its complement $\bar{G}$.

Problem 17. (self-complementary graphs).
1) Give an example of a self-complementary graph with five vertices. Drawings will suffice; use the same notation as for automata, i.e. circles for vertices and lines for edges.
2) If G is a self complementary graph on n vertices, determine how many edges G has. Prove your assertion. (20 points total, 10 points for each part)

Theorem 9.3 (*The $n \equiv 0$ or $n \equiv 1 \mod 4$ Lemma*)**.** *For a self-complementary graph with n vertices, either n or $n - 1$ is divisible by 4.*

Problem 18. Prove the above theorem (25 points).

Concluding remark for graph theory: We have given a formal definition of cycles, which we wished to have, to help with our troubles with the reachability problem in automata theory. Along the way, we touched on the deep question of graph colorings. Finally, we have also proven a famous lemma regarding self-complementary graphs. This highlights the interplay between automata theory and graph theory, as well as the sheer size of graph theory itself as a field of mathematics.