# AN APPROACH TO A UNIFIED THEORY OF AUTOMATA

J. E. Hopcroft†
Princeton University
Princeton, New Jersey

J. D. Ullman
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

## Summary

An automaton called the balloon automaton is defined. The balloon automaton comes in four varieties, depending on whether the device is deterministic or nondeterministic, and whether the input head can move in one or two directions. Subsets of the balloon automata of each variety, called closed classes are defined. Almost all the known types of automata are equivalent to some closed class of balloon automata.

Properties of closed classes are given. For example, whatever the variety, the languages accepted by a closed class are closed under intersection with a regular set.

For a given organization of storage, closed classes of the four varieties can be defined. These four classes are said to form a family. A class may be recursive or not, and the emptiness problem may be solvable or unsolvable. Some surprising relationships exist between the recursiveness and solvability of emptiness for the classes in a family.

## I. Introduction

In the past, and especially recently, people have been examining various species of automata, perhaps as models of the compiling and translating processes, or for the insights they lend to computation.

Many of the properties of each of the automaton classes mentioned are the same. For example, one would expect the set of languages accepted by each class to be closed under intersection with a regular set. Our plan is to propose a model of an automaton abstracting the common features of most of the models mentioned. We will define a closed class of automata to be a subset of the set of all such automata if it satisfies certain simple and physically meaningful closure properties. Then from these closure properties, we will show many of the common closure theorems which have been proven for the specific types of automata mentioned, and which, presumably, would be proven for future types.

Also, we will consider relations between the recursive solvability of certain questions for the classes of automata we define. An example of such a question is: "Is the set accepted by a given automaton of the class empty?"

The basic model is shown in Fig. 1. It consists of an input tape, with end markers, a finite control, and an infinite storage of unspecified structure, called the balloon. There are four varieties, depending on whether the input head can move in two directions or right only, and whether the finite control is deterministic or nondeterministic. We assume that the states of the balloon are represented by the positive integers. A move of the automaton is a three-stage process. First, a recursive function is used to get a finite amount of information from the balloon. Typically, this information is analogous to the symbol scanned by the storage head of an automaton with a tape memory. Second, based on the information from the balloon, the state of the finite control, and the symbol scanned by the input head, a new state of finite control and a direction of input head motion is determined. Third, based on the new state of finite control, and the current state of the balloon, a recursive function determines the next state of the balloon. Certain states of the finite control are final states. If the input causes the automaton to enter a final state, the input is accepted.

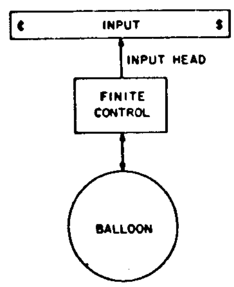


Fig. 1. Balloon Automaton

A subset of the balloon automata of one of the four varieties is called a closed class, or simply a class, if:

† Consultant, Bell Telephone Laboratories, Incorporated, Murray Hill, New Jersey. Currently at Cornell University, Ithaca, New York.

(1) Certain trivial functions that get information from the balloon or operate on the balloon state are allowed.

(2) If two automata are in the class, a third in the class can be found by associating, in any way, the functions getting information from the balloon and determining the next state of the balloon.

The latter condition is vague, but will be made formal.

Most, but not all, of the types of automata mentioned can be interpreted as classes under our definition. It seems that a type of automaton is a class if its definition involves only how the infinite storage may be locally manipulated. The linear bounded automata unfortunately do not form a class in our formulation. Some of the automata which do form classes are:

(1) Pushdown automaton[1,2,3,4]

(2) Stack automaton[5,6]

(3) Nonerasing stack automaton[5,7]

(4) Reading pushdown automaton[8]

(5) Nested stack automaton[8]

(6) Single counter machine[9,10]

(7) Finite automaton[11]

(8) Turing machine[12]

## II. Definitions

The four varieties of balloon automata will be denoted 2NBA, 2DBA, 1NBA and 1DBA. The 1 or 2 indicates the number of directions in which the input head can move. The D or N stands for deterministic or nondeterministic. We will give formal definitions of the four varieties together, pointing out the differences.

A balloon automaton consists of:

(1) A finite, nonempty set of states, S.

(2) A finite set of input symbols, I, which always includes ¢ and \$, the left and right endmarkers, respectively.

(3) A set of balloon states, which is always the positive integers, denoted Z.

(4) A finite, nonempty set of integers, M, known as the balloon information.

(5) A total recursive function, h, from Z to M, known as the balloon information function.

(6) A partial recursive function, f, from $S \times Z$ to Z, known as the balloon control function.

(7) A subset, F, of S, known as the final states.

(8) A state, $q_0$, in S, known as the start state. We do not allow $q_0$ to be in F.

(9) A function, g, with domain $S \times I \times M$, known as the finite control function.

(a) For the 1DBA, the range of g is $S \times \{0,+1\} \cup \{\phi\}$.

(b) For the 1NBA, the range of g is the subsets of $S \times \{0,+1\}$.[(1)]

(c) For the 2DBA, the range of g is $S \times \{-1,0,+1\} \cup \{\phi\}$.

(d) For the 2NBA, the range of g is the subsets of $S \times \{-1,0,+1\}$.

The balloon automaton of any variety is denoted $(S,I,M,f,g,h,q_0,F)$.

We denote a configuration of the balloon automaton $A = (S,I,M,f,g,h,q_0,F)$, whatever the variety, by $(q,w,j,i)$, where:

(1) q is a state of the finite control, in S.

(2) $w = ¢a_1a_2\ldots a_n\$$, $n \geq 0$, where for $1 \leq k \leq n$, $a_k$ is in $I - \{¢,\$\}$. We call n the length of w, denoted $|w|$. Note that the endmarkers are ignored when computing the input's length.

(3) j is an integer between 0 and n + 1, denoting the position of the input head of A. That is, the input head is considered to be at the $j^{th}$ symbol of w. The ¢ sign is symbol 0, \$ is symbol n + 1, and $a_j$ is symbol j for $1 \leq j \leq n$.

(4) i is a positive integer, the state of the balloon.

A move of A is a three-stage process. Let $(q_1,w,j_1,i_1)$ and $(q_2,w,j_2,i_2)$ be configurations of A, with $|w| = n$. For the 2NBA and 1NBA, let $h(i_1) = m$, and let a be the $j_1{}^{th}$ symbol of w. Let $(q_2,d)$ be any element in $g(q_1,a,m)$, where $d = 0$ or $+1$ for the 1NBA, $d = -1, 0,$ or $+1$ for the 2NBA. Let $j_2 = j_1 + d$, but require that $0 \leq j_2 \leq n + 1$. Finally, let $i_2 = f(q_2,i_1)$. Exactly when these conditions are met do we say

(1) The closed class of 1NBA are similar to the Abstract Families of Acceptors discussed in [13]. All the theorems on closure properties of 1NBA presented here are also shown in [13] for an AFA.

$(q_1,w,j_1,i_1) \vdash_A (q_2,w,j_2,i_2)$. We define the relation $\vdash_A^*$ by $(q,w,j,i) \vdash_A^* (q,w,j,i)$ for any configuration. Also, $(q_1,w,j_1,i_1) \vdash_A^* (q_2,w,j_2,i_2)$ whenever there is a third configuration, $(q_3,w,j_3,i_3)$, such that $(q_1,w,j_1,i_1) \vdash_A^* (q_3,w,j_3,i_3)$ and $(q_3,w,j_3,i_3) \vdash_A (q_2,w,j_2,i_2)$.

For the 2DBA and 1DBA, a move is defined as for the 2NBA and 1NBA, respectively, except that in the paragraph above, replace "Let $(q_2,d)$ be any element in $g(q_1,a,m)$" with "Let $(q_2,d)$ be $g(q_1,a,m)$."

If $A = (S,I,M,f,g,h,q_0,F)$ is a 2DBA or 2NBA, define the tapes accepted by A, denoted T(A), to be $\{w|(q_0,w,0,1) \vdash_A^* (q,w,j,i)$, for some q in F, and any integers j and i$\}$. If A is a 1DBA or 1NBA, $T(A) = \{w|(q_0,w,0,1) \vdash_A^* (q,w,n+1,i)$, for some q in F, i any integer, and $|w| = n\}$. (Recall that endmarkers do not count in computing $|w|$.)

Let C be a subset of the balloon automata of some type (2DBA, 2NBA, 1DBA or 1NBA). We say C is a closed class for that type, often shortened to class, if the following two conditions are satisfied:

I. $(S,I,M,f,g,h,q_0,F)$ is in C for arbitrary finite sets S, $F \subseteq S$, $I \supseteq \{¢,\$\}$. M is the range of h; $q_0$ is in S. We insist that $h(i) = j$ for all i and some constant, j. Also, for each q in S, $f(q,i)$ is either i, for all i, or some constant, j, for all i. Lastly, g is any appropriate mapping with domain $S \times I \times M$. The range of g is, of course, $(S\times\{0,+1\}) \cup \{\phi\}$ for the 1DBA, $(S\times\{-1,0,+1\}) \cup \{\phi\}$ for the 2DBA, and the range is the subsets of $S\times\{0,+1\}$ or $S\times\{-1,0,+1\}$, respectively, for the 1NBA and 2NBA.

II. Let $(S_1,I_1,M_1,f_1,g_1,h_1,q_1,F_1)$ and $(S_2,I_2,M_2,f_2,g_2,h_2,q_2,F_2)$ be in C. Then $(S_3,I_3,M_3,f_3,g_3,h_3,q_3,F_3)$ is in C if:

(1) $S_3$, $F_3$ and $I_3$ are arbitrary finite sets, with $F_3 \subseteq S_3$ and $I_3 \supseteq \{¢,\$\}$.

(2) $M_3$ is the range of $h_3$.

(3) $q_3$ is in $S_3$.

(4) For each q in $S_3$, $f_3(q,i) = f_1(p,i)$ for all i or $f_3(q,i) = f_2(p,i)$ for all i, for some particular p in $S_1$ or $S_2$, respectively.

(5) $h_3$ is a total recursive function with finite range such that for any integers $i_1$ and $i_2$, $h_3(i_1) \neq h_3(i_2)$ implies that either $h_1(i_1) \neq h_1(i_2)$ or $h_2(i_1) \neq h_2(i_2)$. I.e., $h_3$ can distinguish between two balloon states if either $h_1$ or $h_2$ distinguishes between them.

(6) $g_3$ is an arbitrary mapping with domain $S_3\times I_3\times M_3$ and range $(S_3\times\{0,+1\}) \cup \{\phi\}$, $(S_3\times\{-1,0,+1\}) \cup \{\phi\}$, the subsets of $S_3\times\{0,+1\}$, or the subsets of $S_3\times\{-1,0,+1\}$, according as C is a class of 1DBA, 2DBA, 1NBA, or 2NBA, respectively.

Intuitively, rule I insures that the balloon control functions which either leave the balloon fixed or set it to a given state are available to the automata of any class. Note that any known type of automaton has the power to leave its infinite storage fixed. Also, they can set their storage to any specified state, although not usually in one move. Even a nonerasing stack automaton can print a dummy "end of stack" marker to simulate an erasure of the stack, then print any finite length stack configuration. The ability of a balloon automaton to reset its balloon is vital to several of the closure properties of the classes. Rule II insures that if two automata are in a class, a new automaton in the class can be constructed, using all the information from the balloon that either of the original automata use and manipulating the balloon state in any way that either of the original automata would.

Thus, a closed class of balloon automata can be thought of as defined by a set of allowable balloon information functions and a set of allowable balloon control functions. These sets each have a closure property, defined by II-4 and II-5 above.

Example: Let us indicate how to interpret a pushdown automaton as a closed class of balloon automata. Most readers should be familiar with a device with pushdown storage, usually taken to be one-way nondeterministic. The one-way deterministic variety is defined in [3] and the two-way varieties in [4].

Informally, the infinite storage is a pushdown tape, from which the automaton can at any time read only the top symbol. The pushdown tape can be altered by erasing the top symbol or printing a new symbol at the top of the tape.[2]

To describe the pushdown tape as a balloon state, we use a standard Gödel numbering of tapes. That is, let a pushdown automaton, P, have tape symbols $Z_1,Z_2,\ldots,Z_k$. Define $\pi(i)$ to be the $i^{th}$ prime. Then the pushdown tape $Z_{j_1}Z_{j_2}\ldots Z_{j_r}$ is represented by the integer $i = [\pi(1)]^{j_1}[\pi(2)]^{j_2}\ldots[\pi(r)]^{j_r}$.

We will define a closed class, C, of balloon automata of some variety, equivalent to the pushdown automata of that variety. C is defined by

(2) A pushdown automaton is often allowed to print more than one symbol at a time, but clearly, no extra power results.

describing sets $H_C$ and $F_C$ of allowable balloon information and balloon control functions. These sets will each contain the functions required by rule I of the definition of closed class and have the properties required by rule II.

Define $\kappa(i)$ to be the number of times the largest prime dividing i divides it. Then $H_C$ contains each function h such that there is a constant, d, for which $h(i) \neq h(j)$ implies that $\kappa(i) \neq \kappa(j)$ and at least one of these is $\leq d$. Thus, h distinguishes between i and j only if the top symbols of the lists represented by i and j are different. For the pushdown automaton P, above, we choose a function h which distinguishes all i and j if $\kappa(i) \neq \kappa(j)$ and $\kappa(i)$ and $\kappa(j)$ are both $\leq k$, since P has k tape symbols.

In $F_C$ we place functions $f(q,i) = j$, where j is $i/p^{\kappa(i)}$, and p is the largest prime dividing i. Functions of this type erase the top symbol of the tape. Also in $F_C$ are functions $f(q,i) = j$, where $j = ip^d$, and p is the smallest prime larger than any dividing i. These functions serve to add a new symbol to the top of the tape.

Armed with these functions, it should not be hard for the reader to simulate any pushdown automaton, P, with a balloon automaton, A, in class C. In addition, to show that C is equivalent to the class of pushdown automata, one must simulate any A in C with a pushdown automaton, P. The tape of P can represent the balloon state of A in the obvious manner. In addition, P needs a "bottom of tape" marker, in case A resets its balloon state, in which case, P must erase its tape down to the marker and then write a finite string. Also, P needs a symbol which represents a prime, p, to the $0^{th}$ power, since A could reset its balloon to a state like 9, which although divisible by 3, is not divisible by 2. This symbol is erased whenever it appears on the top of the tape.

## III. Closure Properties

We shall state the theorems on closure properties of closed classes of balloon automata, with no proof or only an intuitive proof. Formal proofs will soon appear in [14].

Theorem 1: Every set accepted by a 2NBA, 2DBA, 1NBA or 1DBA is recursively enumerable.

Proof: Simulation of a 2NBA, hence any of the other types, by a nondeterministic multitape Turing machine is straightforward.

Theorem 2: (reversal) If L is accepted by an automaton in class C of 2NBA or 2DBA, then $L^R = \{w | w$, reversed is in $L\}$ is accepted by some automaton in C.

Proof: This theorem, like many others in this section is completely elementary. If $A_1$ is a balloon automaton in C with $T(A_1) = L$, then we can construct $A_2$ in C which first moves its input head to the right endmarker, keeping its balloon state at 1, then simulates $A_1$ with the direction of the input head reversed.

Theorem 3: (union) If $L_1$ and $L_2$ are accepted by automata of class C of 2NBA or 1NBA, then $L_1 \cup L_2$ is accepted by an automaton in class C.

Theorem 4: (intersection) If $L_1$ and $L_2$ are accepted by automata of class C of 2NBA or 2DBA, then $L_1 \cap L_2$ is accepted by an automaton in class C.

Definition: A generalized sequential machine [15] (gsm) is a finite automaton with output of arbitrary length, including zero, for each input. Formally, it is denoted $G = (K,\Sigma,\Delta,\delta,\lambda,q_o)$, where:

(1) K is a finite set of states, including $q_o$, the start state.

(2) $\Sigma$ is a finite set of input symbols.

(3) $\Delta$ is a finite set of output symbols. We assume neither $\Sigma$ nor $\Delta$ contain ¢ or \$.

(4) $\delta$ is the next state function, a mapping from $K \times \Sigma$ to $\Sigma$.

(5) $\lambda$ is the output function, mapping $K \times \Sigma$ to $\Delta^*$.

We can extend $\delta$ and $\lambda$ to domain $K \times \Sigma^*$ by, for all q in K, a in $\Sigma$ and w in $\Sigma^*$, $\delta(q,\varepsilon) = q$,(3) $\lambda(q,\varepsilon) = \varepsilon$, $\delta(q,wa) = \delta(\delta(q,w),a)$ and $\lambda(q,wa) = \lambda(q,w)\lambda(\delta(q,w),a)$.

Define G(w) to be $\lambda(q_o,w)$ and $G^{-1}(w) = \{y | G(y) = w\}$. If L is a language whose words are of the form ¢w\$, where w does not involve ¢ or \$, then define G(L) = {¢y\$ | y = G(w) for some ¢w\$ in L}. Also, $G^{-1}(L)$ = {¢w\$ | ¢G(w)\$ is in L}.

Theorem 4: (gsm) If L is accepted by an automaton in class C of 1NBA, and G is a gsm, then G(L) is accepted by some automaton in C.

Proof: Let $L = T(A_1)$. An automaton, $A_2$ in C, with $T(A_2) = G(L)$ can be constructed according to the diagram of Fig. 2.

A nondeterministic generator of symbols in $\Sigma$ generates a symbol which is passed to the gsm, G. If the output of G is not $\varepsilon$, the output is compared with an equal number of remaining input symbols. If they compare properly, the symbol generated is used as though it were input to $A_1$. If the input head of $A_1$ would move right, another symbol is nondeterministically generated.

$A_2$ will accept the input if there is some word, w, with ¢G(w)\$ equal to the input, and ¢w\$ accepted by $A_1$. Thus $A_2$ accepts G(L).

---

(3) $\varepsilon$ is the word of zero length.

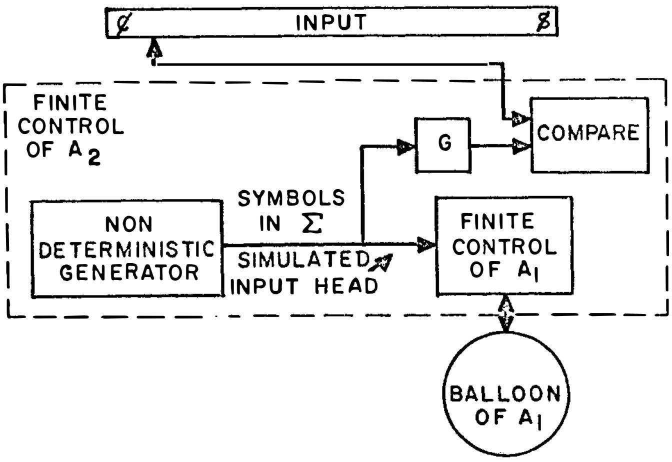


Fig. 2. "Picture Proof" of Gsm Theorem

Theorem 5: (gsm inverse) If L is accepted by an automaton in class C of 2NBA, 2DBA, 1NBA or 1DBA, and G is a gsm, then $G^{-1}(L)$ is accepted by some automaton in class C.[(4)]

Proof: Let L be accepted by $A_1$. We will construct $A_2$, accepting $G^{-1}(L)$ according to the plan of Fig. 3. Suppose C were a class of 1NBA or

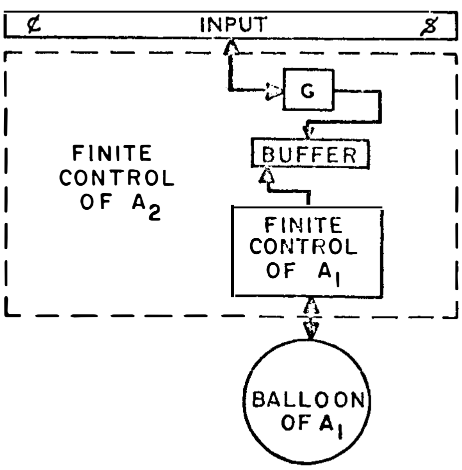


Fig. 3. "Picture Proof" of Gsm Inverse Theorem

1DBA. Then whenever $A_1$ needed a simulated input, $A_2$ could move its input head right, processing the input through G. When an input symbol with non-$\varepsilon$ output was found, the output would be stored in the buffer and used as input to $A_1$ until exhausted. Thus, $A_2$ accepts its input, ¢w\$ if ¢G(w)\$ is accepted by $A_1$. I.e., $T(A_2) = G^{-1}(T(A_1))$.

The same argument works for the two-way cases, but one problem arises. If the input head of $A_1$ moves left, $A_2$ must determine what state G was in before reading the symbol to which $A_2$ moves. Only then can $A_2$ determine what the output of G is.

The fact is that $A_2$ can determine this state. Let $G = (K,\Sigma,\Delta,\delta,\lambda,q_o)$, and define $\gamma$ to be a function from $K\times\Sigma^*$ to the subsets of K such that $\gamma(q,w) = \{p \mid \delta(p,w) = q\}$. We observe the following:

(1) For a in $\Sigma$, $\gamma(q,aw)$ is uniquely determined by $\delta$ and $\gamma(q,w)$.

(2) If $q_1$ and $q_2$ are states, $q_1 \neq q_2$, then $\gamma(q_1,w)$ and $\gamma(q_2,w)$ are disjoint. (Else, there would be a p with $\delta(p,w) = q_1 = q_2$.)

(3) Let a be in $\Sigma$ and $\delta(q_1,a) = \delta(q_2,a) = q$. For any y in $\Sigma^*$, let $p_1$ be in $\gamma(q_1,y)$ and $p_2$ be in $\gamma(q_2,y)$. Then $\delta(p_1,ya) = \delta(p_2,ya) = q$. However, if x is any prefix of y (including y itself), then $\delta(p_1,x) \neq \delta(p_2,x)$. (Else, let y = xz and $\delta(p_1,x) = \delta(p_2,x) = p$. But it is impossible to have $\delta(p,z) = q_1 = q_2$.)

Suppose $¢a_1a_2\ldots a_n\$$ is input to $A_2$, and the input head of $A_2$ is at $a_i$. Also, assume $\delta(q_o,a_1a_2\ldots a_{i-1}) = q$, and this state is recorded as the state of G by $A_2$. If $A_2$ must move left, it must find $\delta(q_o,a_1a_2\ldots a_{i-2})$. It considers $\gamma(q,a_{i-1})$. If this set contains a single element, that state is $\delta(q_o,a_1a_2\ldots a_{i-2})$. Suppose $\gamma(q,a_{i-1})$ contains $q_1,q_2,\ldots,q_r$, $r \geq 2$. $A_2$ moves its input head left, successively computing $\gamma(q_j,a_ka_{k+1}\ldots a_{i-2})$ for $1 \leq j \leq r$ and $k = i-2, i-3, \ldots$ . Two cases may arise.

(1) $\gamma(q_j,a_ka_{k+1}\ldots a_{i-2})$ is empty unless j = m, but for at least two values j, $\gamma(q_j,a_{k+1}a_{k+2}\ldots a_{i-2})$ is nonempty. Then $q_m$ is $\delta(q_o,a_1a_2\ldots a_{i-2})$, and $A_2$ must find its way back to $a_{i-1}$. For some distinct $j_1$ and $j_2$, $A_2$ can find $p_1$ and $p_2$ in $\gamma(q_{j_1},a_{k+1}\ldots a_{i-2})$ and $\gamma(q_{j_2},a_{k+1}\ldots a_{i-2})$, respectively. $A_2$ then moves right, computing $\delta(p_1,a_{k+1}\ldots a_s)$ and $\delta(p_2,a_{k+1}\ldots a_s)$ for successive values of s, starting with s = k + 1. By comment (3) above, these

(4) This theorem was not previously known for the two-way deterministic and nondeterministic stack automata.

quantities will first be equal when $s = i - 1$.

(2) $A_2$ reaches the left endmarker. The $q_m$ for which $q_o$ is in $\gamma(q_m, a_1a_2 \ldots a_{i-2})$ is $\delta(q_o, a_1a_2 \ldots a_{i-2})$. Presumably, for distinct $j_1$ and $j_2$, $A_2$ can find $p_1$ and $p_2$ in $\gamma(q_{j_1}, a_1a_2 \ldots a_{i-2})$ and $\gamma(q_{j_2}, a_1a_2 \ldots a_{i-2})$, respectively. $A_2$ then uses $p_1$ and $p_2$ to find its way back to $a_{i-2}$ as in case (1).

A more detailed proof appears in [14,16].

Theorem 6: (intersection with regular set) If L is accepted by an automaton of class C of any variety, and R is a regular set, then $L \cap R$ is accepted by an automaton in C.

Theorem 7: (concatenation) Let $L_1$ and $L_2$ be accepted by automata in class C of 1NBA. Then $\{\cent wy\$ | \cent w\$ \text{ in } L_1 \text{ and } \cent y\$ \text{ in } L_2\}$ is accepted by some automaton in C.

Theorem 8: (Kleene closure) Let L be accepted by an automaton in class C of 1NBA. Then $\{\cent w_1w_2 \ldots w_k\$ | k \geq 0 \text{ and } \cent w_i\$ \text{ is in L for all } i, 1 \leq i \leq k\}$ is accepted by some automaton in C.

Theorem 9: (quotient with regular set) Let L be accepted by an automaton in class C of 1NBA, and let R be a regular set. Then $\{\cent w\$ |$ for some y in R, $\cent wy\$$ is in L$\}$ is accepted by an automaton in C.

These results are summarized in Fig. 4.

| | 2DBA | 2NBA | 1DBA | 1NBA |
|---|---|---|---|---|
| reversal | ✓ | ✓ | | |
| intersection | ✓ | ✓ | | |
| gsm inverse | ✓ | ✓ | ✓ | ✓ |
| union | | ✓ | | ✓ |
| intersection with regular set | ✓ | ✓ | ✓ | ✓ |
| concatenation (·) | | | | ✓ |
| Kleene closure (*) | | | | ✓ |
| gsm forward | | | | ✓ |
| quotient with regular set (/) | | | | ✓ |

Fig. 4

## IV. Decidability Results

For a class C, of one of the four varieties, one can define classes of the other three types which are naturally related to C. These classes have the same balloon information and balloon control functions as C. The four classes, one of each type (2NBA, 2DBA, 1NBA and 1DBA) whose allowable functions correspond are called a family. For example, four types of stack automata are known - the one-way and two-way, deterministic and nondeterministic varieties. These are together equivalent to a family of balloon automata.

We are interested in the decidability of two questions concerning closed classes of automata. First, given a balloon automaton, A, in class C, and an input w to A, is w in T(A)? This question is called the membership question, and if it is solvable for any A in C, then C is said to be recursive. The second question is: Given a balloon automaton, A, in class C, does A accept any words at all? This question is called the emptiness question.

Given a family of closed classes, there are eight questions. These are the membership and emptiness questions for each of the four varieties. We will denote the questions by M for membership or E for emptiness, followed by the designation of the variety of balloon automata involved. (M-2NBA, for example.)

If $Q_1$ and $Q_2$ are two questions, we say $Q_1$ implies $Q_2$ if $Q_2$ is solvable for any family for which $Q_1$ is solvable. We say $Q_1$ is equivalent to $Q_2$ if $Q_1$ is solvable for a family if and only if $Q_2$ is.

We can prove several relationships among the questions. Formal proofs of the relations are found in [17].

Some relations follow immediately from the definition of the four varieties of balloon automata. Among them are:

(a) M-2DBA implies M-1DBA

(b) M-2NBA implies M-1NBA

(c) M-2NBA implies M-2DBA

(d) E-2NBA implies E-2DBA

(e) E-2DBA implies E-1DBA

(f) E-1NBA implies E-1DBA

Some less trivial results can be proven.

Theorem 10: M-1NBA implies E-1NBA.

Proof: Let L be accepted by a 1NBA in class C. Let G be the gsm which maps every word to ε. Then by Theorem 4, G(L) is accepted by a 1NBA in class C. L is empty if and only if ¢\$ is not in G(L).

Theorem 11: M-1DBA implies M-2DBA; M-1NBA implies M-2NBA.

Proof: For a given 2DBA, with given input, w, we can combine the input and finite control to form the finite control of a 1DBA which accepts ¢\$ if and only if the 2DBA accepts w.

Likewise for the nondeterministic case.

Corollary 1: M-1DBA is equivalent to M-2DBA.

Corollary 2: M-1NBA is equivalent to M-2NBA.

Theorem 12: E-1DBA implies M-1NBA.

Proof: Let A be a 1NBA in class C, which has at most r choices in any configuration. Let ¢$a_1 a_2 \ldots a_n$\$ be an input to A. We can consider a language, L, whose words are of the form ¢$w_0 a_1 w_1 a_2 w_2 \ldots a_n w_n \# w_{n+1}$\$.[5] The w's are strings of integers from 1 to r, $w_i$ indicates the choices of moves made by A while scanning $a_i$. ($a_0$ = ¢ and $a_{n+1}$ = \$.)

There is a 1DBA, of a class in the same family as C, accepting those words in L for which A would accept, making the choices of moves indicated by the $w_i$'s. The language accepted by this automaton is nonempty if and only if ¢$a_1 a_2 \ldots a_n$\$ is accepted by A.

Corollary: The following four questions are equivalent: E-1DBA, E-1NBA, M-1NBA, M-2NBA.

Proof: From Corollary 2 to Theorem 11, and Theorems 10 and 12.

These results are summarized in Fig. 5. The decidability of questions low on the hierarchy of Fig. 5 imply the decidability of questions above.

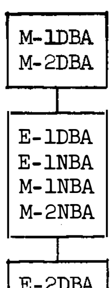


Fig. 5. Hierarchy of Questions

(5) The symbol # represents the right endmarker of A.

## V. Conclusions

We have attempted to present a framework in which certain theorems that are proven for each familiar automaton could be proven once and for all. At the same time, we hope we have gained insight into the properties that make these theorems true.

We feel further work should lead to restrictions on balloon automata that will enable other theorems to be proven. For example, under what conditions would a class of 1DBA or 2DBA be closed under complement?

## Bibliography


1. N. Chomsky, "Context Free Languages and Pushdown Storage," Quarterly Progress Report No. 65, Research Laboratory of Electronics, M.I.T., Cambridge, Massachusetts, 1962.
2. R. J. Evey, "The Theory and Application of Pushdown Store Machines," Mathematical Linguistics and Automatic Translation, Report No. NSF-10, Computation Laboratory of Harvard University, May 1963.
3. S. Ginsburg, S. Greibach, "Deterministic Context Free Languages," Information and Control, Vol. 9, No. 6, pp. 620-648, December 1966.
4. J. Gray, M. Harrison, O. Ibarra, "Two-Way Pushdown Automata," University of California, Berkeley, Dept. of Elect. Eng. Report, 1966.
5. S. Ginsburg, S. Greibach, M. Harrison, "Stack Automata and Compiling," JACM, Vol. 14, No. 1, pp. 172-201, January 1967.
6. S. Ginsburg, S. Greibach, M. Harrison, "One-Way Stack Automata," JACM, Vol. 14, No. 2, pp. 389-418, April 1967.
7. J. Hopcroft, J. Ullman, "Nonerasing Stack Automata," JCSS, to appear.
8. A. Aho, "Indexed Grammars - An Extension of Context Free Grammars," these proceedings.
9. M. Schutzenberger, "Finite Counting Automata," Information and Control, Vol. 5, No. 2, pp. 91-107, June 1962.
10. M. Minsky, "Recursive Unsolvability of Post's Problem of 'Tag' and Other Topics in the Theory of Turing Machines," Annals of Math., Vol. 74, No. 3, pp. 437-455, November 1961.
11. M. Rabin, D. Scott, "Finite Automata and Their Decision Problems," IBMJ, Vol. 3, pp. 114-125, 1959.
12. A. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem," Proc. London Math. Soc., Vol. 42, pp. 230-265, 1936.

13. S. Ginsburg, S. Greibach, "Abstract Families of Languages," these proceedings.

14. J. Hopcroft, J. Ullman, "An Approach to a Unified Theory of Automata," BSTJ, to appear.

15. S. Ginsburg, "Examples of Abstract Machines," IRE TEC, Vol. EC-11, pp. 132-135, 1962.

16. J. Hopcroft, J. Ullman, "Deterministic Stack Automata and the Quotient Operator," submitted to JCSS.

17. J. Hopcroft, J. Ullman, "Decidable and Undecidable Questions About Automata," submitted to JACM.