Feedback Can Reduce the Specification Complexity of Motor Programs*

Magnus Egerstedt and Roger Brockett

magnus@ece.gatech.edu
Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332

brockett@hrl.harvard.edu
Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138

Abstract

In this paper we show that when it is possible to use feedback in the specification of "motor programs", the length of the descriptions of the instruction sequences for carrying out a given task can be reduced by a factor that reflects the richness of the available feedback signals. The model on which this work is based is that of a finite automaton, modified in such a way that instruction processing is akin to the way in which difference or differential equations "process" piecewise constant inputs.

1 Introduction

The many visible and successful applications of feedback mechanisms at work testify to its effectiveness and over the years a variety of arguments have been advanced showing why, in particular settings, it is useful. The models commonly used bring to the fore considerations of sensitivity, uncertainty, etc, and a common element is the explicit or implied subdivision of the system into two parts, a forward path whose performance can only be characterized loosely and a feedback path whose behavior is known with greater certainty. Specific formalizations which start from this point include:

1. The Black argument for reducing the effect of drift in a high-gain amplifier by the use of a relatively constant, but low gain, feedback term [2].

2. The stochastic disturbance argument for using measurements to reduce the effect of probabilistic uncertainty. (See for example [6].)

3. The game theoretic argument in which a saddle point condition is enforced by feedback. (H_∞ control can be thought of this way.)

To this list we now add a fourth item which we cast in terms of the effect feedback has on reducing the complexity of the implementation.

4. The complexity argument, proposed in this paper, showing that feedback can shorten motor programs if reliable sensory information is available.

We consider the problem of describing a procedure that will assure that a system (robot, machine tool, etc), will reach a certain goal state. We will give a quantitative analysis showing that the availability of feedback can reduce the length of the shortest description of such a procedure. In particular, Theorem 4.1 shows that the length of the description can be reduced by a factor that depends on the ratio of the size of the entire state space to the size of the set of states for which feedback is locally effective. In some natural settings this argument can be used recursively, leading to further reductions, as seen in Theorem 4.2.

We have found it necessary to use a few terms that are not (as yet) standard in the field of automatic control. Two in particular require clarification. The term "motor program" is taken from the field of biological motor control where it is a standard idea referring to the mechanisms that support the definition and execution of commands that synchronize and coordinate the movement of muscle groups. (See for example [7].) From our point of view, a motor program is a symbolic string of instructions that specifies a particular movement, as suggested in [3]. Secondly, rather than using differential equations as the starting point for our analysis we use a type of finite state model that we refer to as *free-running, feedback automata* (or FRF-automata for short). Briefly stated, a FRF-automaton is a fi-

*This work was supported by the US Army Research Office, Grant number DAAG 5597-1-0114.

nite state machine that reads an input and then makes transitions repeatedly, using the same input value, until a particular output condition is realized.

2 Free-Running Feedback Automata

The input symbols for our automaton will be drawn from the finite set S, called the *input alphabet*, and the input strings are called *words*. We use $S^\star$ to denote the set of all such words, including the empty one. We let $s \in S$ denote an element in S, and use boldface $\mathbf{s} \in S^\star$ to denote elements in $S^\star$. If we define the associative operation of concatenation on $S^\star$, the empty word serves as an identity under this operation. Thus $S^\star$ is the *free monoid* generated by S.

If we let X, V be finite sets, and let $\delta \in X^{X \times V}$, then we can identify (X, V, δ) with a *finite automaton* (see for example [1]), whose operation is given by $x_{k+1} = \delta(x_k, v_k)$. If we add another finite set Y and a mapping $\gamma \in Y^X$ to the definition, we get an *output* automaton $(X, Y, V, \delta, \gamma)$, where $x_{k+1} = \delta(x_k, v_k)$ and $y_k = \gamma(x_k)$.

Given a word $\mathbf{v} \in V^\star$, where $\mathbf{v} = v_1 \cdots v_p$, we use $\delta(x, \mathbf{v})$ as shorthand for $\delta(\delta(\cdots(\delta(x, v_1), v_2) \cdots, v_{p-1}), v_p)$, and we let v^p denote the word obtained by concatenating v with itself $p-1$ times, i.e. $v^p = v \cdots v$.

We now introduce a dynamical system called a *free-running automaton*. The idea is to let such an automaton read an input from a given alphabet, and then advance the state of the automaton repeatedly without reading any new inputs until an interrupt is triggered.

Definition 2.1 *(Free-Running Automaton) Let X, Y be finite sets, let $U = V \times \mathcal{T}$, where $\mathcal{T} \subset \{0,1\}^Y$, and V is a finite set. Let $\delta : X \times V \to X$ and $\gamma : X \to Y$ be given functions. We say that $(X, U, Y, \delta, \gamma)$ is a free-running automaton with transitions generated according to $x_{k+1} = \delta(x_k, v_{l_k})$, where*

$$l_{k+1} = \begin{cases} l_k \text{ if } \tau_{l_k}(\gamma(x_{k+1})) = 0 \\ l_k + 1 \text{ otherwise,} \end{cases}$$

given the input string $u_1 \cdots u_p = (v_1, \tau_1) \cdots (v_p, \tau_p) \in U^\star$.

Sometimes the input set has additional structure, e.g. $A \times B$ or $\{0,1\}^P$. We are interested in the special case where the alphabet is a product $V \times K$, where V is a finite set and $K \subset V^{Y \times V}$. We denote an element in $V \times K$ by $(v, \kappa(\cdot))$, and an input to a finite automaton generates transitions according to the rule $\delta(x, (v, \kappa)) = \delta(x, \kappa(\gamma(x), v))$. If the input alphabet W admits this structure, and δ takes this form, then we will say that a finite automaton (X, W, δ) admits the structure of a *feedback automaton*. Of course, most finite automata will not be decomposable in this way.

We now combine the ideas of free-running and feedback automata to get our primary object of study: the *free-running, feedback automaton* (FRF-automaton). It is a free-running automaton whose input alphabet admits the structure $\Sigma = V \times K \times \mathcal{T}$, where V is a finite set, $K \subset V^{Y \times V}$, and $\mathcal{T} \subset \{0,1\}^Y$. Thus the input to a FRF-automaton is a triple (v, κ, τ), where $v \in V, \kappa : Y \times V \to V$, and $\tau : Y \to \{0,1\}$.

Definition 2.2 *(Free-Running, Feedback Automaton) Let X, Y, δ, γ be as in Definition 2.1. Let $\Sigma = V \times K \times \mathcal{T}$, where V is a finite set, $K \subset V^{Y \times V}$, and $\mathcal{T} \subset \{0,1\}^Y$. We say that $(X, \Sigma, Y, \delta, \gamma)$ is a free-running, feedback automaton whose evolution equation is $x_{k+1} = \delta(x_k, \kappa_{l_k}(\gamma(x_k), v_{l_k}))$, where*

$$l_{k+1} = \begin{cases} l_k \text{ if } \tau_{l_k}(\gamma(x_{k+1})) = 0 \\ l_k + 1 \text{ otherwise,} \end{cases}$$

given the input string $(v_1, \kappa_1, \tau_1) \cdots (v_p, \kappa_p, \tau_p) \in \Sigma^\star$.

It can be noted that the free-running property of the FRF-automata implies that they can, in general, be guided along a path using fewer instructions than the classical finite automata. However, since the input set to a finite automaton is a finite set V, while the input set to the corresponding FRF-automaton is of the form $V \times K \times \mathcal{T}$, where $K \subset V^{Y \times V}, \mathcal{T} \subset \{0,1\}^Y$, the input set has a higher cardinality in the latter of these cases. Any reasonable measure of the complexity of a control procedure must take the size of the input space into account since the number of bits required to code a word over a given alphabet typically depends logarithmically on the size of the alphabet. (See for example [5].) This dependency is captured in a natural way if we define the complexity of a control procedure as the description length of the input sequence.

Consider a finite set S. We say that a word $\mathbf{s} \in S^\star$ has description length $\mathcal{L}(\mathbf{s}, S) = |\mathbf{s}| \log_2(card(S))$.

Definition 2.3 *(Complexity) Consider a FRF-automaton, A, with state space X and input set Σ. Let $\boldsymbol{\sigma}$ be the word of minimal description length over Σ that drives the automaton between two given states $x_0, x_f \in X$. We then say that the task of driving A between x_0 and x_f has complexity $\mathcal{C}(A, x_0, x_f) = \mathcal{L}(\boldsymbol{\sigma}, \Sigma)$.*

3 Observer Automata

Consider the finite automaton $(X, Y, V, \delta, \gamma)$. We define the *output sequence map* $\mathcal{O} : \mathbb{Z}^+ \times X \times V^Y \to Y^\star$

as $\mathcal{O}(p,x,w) = \gamma(x_1)\cdot\gamma(x_2)\cdots\gamma(x_p)$, where $w: Y \to V$, and $x_1 = x$, $x_2 = \delta(x_1, w(\gamma(x_1))), \ldots, x_p = \delta(x_{p-1}, w(\gamma(x_{p-1})))$. Here $y_1 \cdot y_2$ denotes the concatenation of the letters y_1 and y_2 from the finite alphabet Y, and $\mathcal{O}(p,x,w) \in Y^p \subset Y^\star$, where Y^p is the set of words of length p over Y.

Definition 3.1 *(Observability) A finite automaton (X,Y,V,δ,γ) is observable if there exist a positive integer p_{obs} and an output-to-input mapping $w_{obs}: Y \to V$ that satisfies $\mathcal{O}(p_{obs}, x_1, w_{obs}) \neq \mathcal{O}(p_{obs}, x_2, w_{obs})$ for all $x_1, x_2 \in X$, $x_1 \neq x_2$.*

Definition 3.2 *(Observer Automaton) Consider the observable finite automaton $A = (X,Y,V,\delta,\gamma)$. Let Z, O be finite sets, $\Omega = V \times V^{O\times V}$, $g: Z\times Y\times\Omega \to Z$, and $h: Z\times Y \to O$. Then (Z,O,Ω,g,h) is said to be an observer automaton to A if there exists a $\omega = (v,w) \in \Omega$ such that*

$$x_{k+1} = \delta(x_k, w(o_k, v)),\ y_k = \gamma(x_k)$$
$$z_{k+1} = g(z_k, y_k, w(o_k, v)),\ o_k = h(z_k, y_k)$$

implies that the current state of X can be uniquely determined from the current state of Z, provided that sufficiently many iterations have been made. We say that the number of iterations necessary for achieving this mapping is the settling time of the observer.

Theorem 3.3 *(Observers Exist) Given an observable automaton, there exists an observer automaton.*

Proof: The proof is constructive. Let p_{obs} be the positive integer in Definition 3.1, and let $Z = \{e\} \cup Y \cup Y^2 \cup \cdots \cup Y^{p_{obs}-1} \cup X$, where e is any symbol distinguishable from the rest of the states in Z.

Consider $w_{obs}: Y \to V$ from Definition 3.1, and define the mapping $\Xi: Y^{p_{obs}} \times V \to X$ as follows: For any given $x_1 \in X$ and $v \in V$, we let $\Xi(\mathcal{O}(p_{obs}, x_1, w_{obs}), v) = \delta(x_{p_{obs}}, v)$, where $x_{p_{obs}} = \delta(x_{p_{obs}-1}, w_{obs}(\gamma(x_{p_{obs}-1}))), \ldots, x_2 = \delta(x_1, w_{obs}(\gamma(x_1)))$. For every $\mathbf{y} \in Y^{p_{obs}}$ that does not satisfy $\mathbf{y} = \mathcal{O}(p_{obs}, x, w_{obs})$ for some $x \in X$, we furthermore let $\Xi(\mathbf{y}, v)$ be assigned any arbitrary value.

Now, let $h: Z\times Y \to O$ be given by

$$h(z,y) = \begin{cases} y & \text{if } z \notin X \\ z & \text{otherwise,} \end{cases}$$

and we thus have $O = Y \cup X$.

Let $z_0 = e$, and let z_{k+1} be given by

$$\begin{cases} y_k \text{ if } z_k = e \\ z_k \cdot y_k \text{ if } z_k \in Y^q,\ q < p_{obs} - 1 \\ \Xi(z_k \cdot y_k, w_k(h(z_k, y_k), v_k)) \text{ if } z_k \in Y^{p_{obs}-1} \\ \delta(z_k, v_k) \text{ if } z_k \in X. \end{cases}$$

Given $w_{obs}: Y \to V$ from Definition 3.1 we define $\bar{w}_{obs}: O\times V \to V$ as $\bar{w}_{obs}(o,v) = w_{obs}(o)$ if $o \in Y$, and $w_{obs}(\gamma(o))$ if $o \in X$. If we apply this input, together with any $v \in V$, to both the finite and the observer automata we get that the observer automaton thus has settling time p_{obs}, and the theorem follows. ∎

Definition 3.4 *(Global Attractor) We say that $x_f \in X$ is a global attractor for the difference equation $x_{k+1} = \lambda(x_k)$ if, for all $x_0 \in X$, it holds that $\lim_{k\to\infty} x_k = x_f$.*

Theorem 3.5 *(Creating Global Attractors) Consider the finite automaton (X,V,δ). If $x_f \in X$ can be reached from every initial state, and there is a $v_f \in V$ such that $\delta(x_f, v_f) = x_f$, then there is a mapping $w_{attr}: X \to V$ such that $x_{k+1} = \delta(x_k, w_{attr}(x_k))$ has x_f as a global attractor.*

Proof: Choose an arbitrary $x_1 \in X$. Let $\mathbf{v}_1$ denote a (not necessarily unique) shortest input sequence that drives the automaton from x_1 to x_f. Decompose $\mathbf{v}_1$ as $\mathbf{v}_1 = v_1 \cdot \tilde{\mathbf{v}}_1$, where $v_1 \in V$ and $\tilde{\mathbf{v}}_1 \in V^\star$. Let the candidate for the controller that makes x_f a global attractor satisfy $w_{attr}(x_1) = v_1$.

Now, let x_2 denote the state $\delta(x_1, v_1)$ and repeat the argument until the automaton reaches x_f. By letting the initial state vary over all of X, a control that drives the automaton between an arbitrary initial state and x_f is obtained. Furthermore, let $w_{attr}(x_f) = v_f$, which implies that x_f is a global attractor. The theorem thus follows. ∎

Corollary 3.6 *Given the FRF-automaton $(X,Y,\Sigma,\delta,\gamma)$, where $\Sigma = V\times V^{Y\times V}\times\{0,1\}^Y$, $Y = X$, and $\gamma(x) = x, \forall x \in X$. This automaton can reach any given state $x_f \in X$ using only one instruction if x_f can be reached from any initial state when using the finite automaton (X,V,δ).*

Proof: Choose the input (v,κ,τ) as

$$\begin{cases} v \text{ arbitrary} \\ \kappa(x,w) = w_{attr}(x), \forall x \in X, w \in V \\ \tau(x) = 1 \text{ if and only if } x = x_f, \end{cases}$$

where $w_{attr}(x)$ is defined in Theorem 3.5. This input drives the automaton to x_f, and the corollary follows. ∎

Lemma 3.7 *(Observers Make Single Instruction Goal Achievement Possible) Let the observable finite automaton (X,Y,V,δ,γ) be such that x_f can be*

reached from any initial state. Then, by using the observer automaton from Theorem 3.3, it is possible to drive the state of the FRF-automaton $(X, Y, \Sigma_O, \delta, \gamma)$, where $\Sigma_O = V \times V^{O\times V} \times \{0,1\}^O$, between any initial state and x_f using only one instruction.

Proof: Construct the observer automaton as in Theorem 3.3. Pick $w_{obs} : Y \to V$ as in Definition 3.1, and choose the input sequence to the FRF-automaton as

$$\begin{cases} v = \text{ arbitrary} \\ \kappa(o, v) = w_{obs}(o), \quad \text{if } o \in Y \\ \kappa(o, v) = w_{attr}(o), \quad \text{if } o \in X \\ \tau(o) = 1 \text{ if and only if } o = x_f, \end{cases}$$

where $w_{attr}(x)$ is defined in Theorem 3.5.

By using this input, the FRF-automaton traverses its states until the observer has converged, i.e. k advances p_{obs} steps. Then it drives its state to x_f as in Corollary 3.6, which concludes the proof. ∎

Definition 3.8 *(Observable Subset) Consider the finite automaton $(X, Y, V, \delta, \gamma)$. A subset $X_g \subset X$ such that $\gamma(X_g) \cap \gamma(X \backslash X_g) = \emptyset$ is said to be observable if there exist a positive integer p_{obs} and a $w_{obs} : Y \to V$ that satisfies the following conditions:*

- $\mathcal{O}(p_{obs}, x_1, w_{obs}) \neq \mathcal{O}(p_{obs}, x_2, w_{obs}), \ \forall x_1, x_2 \in X_g, x_1 \neq x_2$;
- *For all $x_1 \in X_g$ it follows that $x_q \in X_g$, $q = 1, \ldots, p_{obs}$, where $x_2 = \delta(x_1, w_{obs}(\gamma(x_1)))$, $x_3 = \delta(x_2, w_{obs}(\gamma(x_2)))$, ...*

Definition 3.9 *(Subset-Observer Automaton) Consider the finite automaton $A = (X, Y, V, \delta, \gamma)$, where $X_g \subset X$ is an observable subset. (Z, O, Ω, g, h), where Z, O are finite sets, $\Omega = V \times V^{O\times V}$, $g : Z \times Y \times \Omega \to Z$, and $h : Z \times Y \to O$ is a subset-observer automaton to A if there exists a $\omega = (v, w) \in \Omega$ such that the following conditions hold:*

$$x_{k+1} = \delta(x_k, w(o_k, v)), \ y_k = \gamma(x_k)$$
$$z_{k+1} = g(z_k, y_k, w(o_k, v)), \ o_k = h(z_k, y_k)$$

gives that the current state in Z can be mapped uniquely to the current state in X after sufficiently many iterations. Also, for all $x_1 \in X_g$ it holds that $x_q \in X_g$, $q = 1, \ldots, p_{obs}$, where $x_2 = \delta(x_1, w(o_1, v))$, $x_3 = \delta(x_2, w(o_2, v))$, and so on.

Lemma 3.10 *(Subset-Observers Exist) Let $X_g \subset X$ be an observable subset to the finite automaton $A = (X, Y, V, \delta, \gamma)$. Then a subset-observer automaton (Z, O, Ω, g, h) to A can always be constructed.*

Proof: Let Z, Ω be as in Theorem 3.3. Let $O = \{e\} \cup \gamma(X_g) \cup X_g$, and let

$$h(z, y) = \begin{cases} y & \text{if } z \notin X \cup \{e\} \\ z & \text{otherwise.} \end{cases}$$

Furthermore, let z_{k+1} be given by

$$\begin{cases} e \text{ if } y_k \notin \gamma(X_g) \\ y_k \text{ if } z_k = e \text{ and } y_k \in \gamma(X_g) \\ z_k \cdot y_k \text{ if } z_k \in \gamma(X_g)^q, \ q < p_{obs} - 1 \\ \Xi(z_k \cdot y_k, w_k(h(z_k, y_k), v_k)) \text{ if } z_k \in \gamma(X_g)^{p_{obs}-1} \\ \delta(z_k, v_k) \text{ if } z_k \in X_g, \end{cases}$$

where the mapping $\Xi(\cdot)$ is defined in Theorem 3.3.

If we now use $\bar{w}_{obs}(o, v) = v$ if $o = e$, $\bar{w}_{obs}(o, v) = w_{obs}(o)$ if $o \in \gamma(X_g)$, and $\bar{w}_{obs}(o, v) = w_{obs}(\gamma(o))$ if $o \in X_g$, then a repetition of the argument in Theorem 3.3 gives that it is in fact a subset-observer automaton, with settling time p_{obs}. ∎

It should be noted that the subset-observer's output set has a lower cardinality than the full observer automaton's output set in Theorem 3.3 as long as $card(X_g) < card(X) - 1$. Furthermore, the value of the subset-observer's output is e whenever $x \notin X_g$. Thus, by using open-loop control on $X \backslash X_g$, and using observer-based feedback on X_g, we can expect a decrease in complexity. In order to make this observation rigorous, we need to introduce the notions of ballistic reachability and control-invariant reachability: A set $X_s \subset X$ is *ballistically reachable from x* if there exists a $v \in V$ such that $\delta(x, v^q) \in X_s$ for some $q \in \mathbb{Z}^+$. Furthermore, X_s is ballistically reachable from $X_t \subset X$ if there exists a $v \in V$ such that for all $x \in X_t$ it holds that $\delta(x, v^{q(x)}) \in X_s$ for some $q(x) \in \mathbb{Z}^+$. An element $x \in X_s \subset X$ is said to be *control-invariantly reachable in X_s* if it can be reached from all states in X_s without the trajectory leaving X_s.

Lemma 3.11 *(One Instruction Suffices When Using Subset-Observers) Let X_f be an observable subset to the finite automaton $(X, Y, V, \delta, \gamma)$, and let $x_0 \notin X_f$, $x_f \in X_f$. If X_f is ballistically reachable from x_0, and x_f is control-invariantly reachable in X_f, then there exists a FRF-automaton $(X, Y, \Sigma', \delta, \gamma)$ that can reach x_f from x_0 using only one instruction.*

Proof: Construct the subset-observer from Lemma 3.10 and let $v_{ol} \in V$ be an open loop control that drives the automaton from x_0 to X_f. (The existence of such a control follows since X_f is ballistically reachable from x_0.) Let the input to the FRF-automaton,

$\sigma = (v, \kappa, \tau),\ v \in V,\ \kappa : O \times V \to V,\ \tau : O \to \{0,1\}$ be given by

$$\begin{cases} v = v_{ol} \\ \kappa(o,v) = v \ \text{ if } \ o = e \\ \kappa(o,v) = w_{obs}(o) \ \text{ if } \ o \in \gamma(X_f) \\ \kappa(o,v) = w_{attr}(o) \ \text{ if } \ o \in X_f \\ \tau(o) = 1 \ \text{ if and only if } \ o = x_f. \end{cases}$$

In other words, $\Sigma' = \{v_{ol}\} \times V^O \times \{0,1\}^O$, and the previous input drives the state of the automaton from x_0 to X_f using the open-loop input v_{ol}. It then executes the observer based motion from Lemma 3.7 on the subset X_f. ∎

4 Instructions Which Lead to the Goal

The reason for studying the situation in Lemma 3.11 is that it captures the idea that it is possible to successfully combine uncertain feed-forward control and high-precision feedback control on different parts of the state space.

But, in order to compare purely open-loop control, i.e. when no observations are made, with a situation where sensory information is available we must be able to generate open-loop motions on the FRF-automata. It is clear that the input sequence $\boldsymbol{\sigma}_{ol} = (v_1, \kappa_{ol}, \tau_{ol}) \cdots (v_q, \kappa_{ol}, \tau_{ol}) \in \Sigma^\star$, where $\kappa_{ol}(v,y) = v\ \forall v \in V,\ y \in Y,\ \tau_{ol}(y) = 1\ \forall y \in Y$ achieves this. However, this word has length q, and it is drawn from the input alphabet $\Sigma = V \times V^{Y \times V} \times \{0,1\}^Y$, and thus the description length is $\mathcal{L}(\boldsymbol{\sigma}_{ol}, \Sigma) = q \log_2(card(\Sigma))$. But, this is clearly not the result we would expect. Instead we can restrict the input alphabet to be $\Sigma_{ol} = V \times \{\kappa_{ol}\} \times \{\tau_{ol}\}$, which has cardinality $card(V)$. The description length of $\boldsymbol{\sigma}_{ol}$ is now $\mathcal{L}(\boldsymbol{\sigma}_{ol}, \Sigma_{ol}) = q \log_2(card(V))$, relative to the smaller input set Σ_{ol}.

Now, consider the connected, classical, finite automaton $A = (X, V, \delta)$. We recall that the *backwards eccentricity* of a state, $ecc(A,x)$, denotes the minimum number of instructions necessary for driving the automaton from any other state to x. (See for example [4].) We furthermore let the *radius* of A be given by $radius(A) = \min_{x \in X} ecc(A,x)$.

Consider the FRF-automaton $\tilde{A}$. If we let $\mathcal{C}(\tilde{A}, x) = \max_{x_0 \in X} \mathcal{C}(\tilde{A}, x_0, x)$, then we directly get that $\mathcal{C}(A_{ol}, x) = ecc(A,x) \log_2(card(V)) \geq radius(A) \log_2(card(V))$, where A_{ol} is the FRF-automaton $(X, Y, \Sigma_{ol}, \delta, \gamma)$.

Theorem 4.1 *(Main Theorem) Assume that* $card(V) \geq 2$. *Suppose that* $x_f \in X_f$, *where* X_f *is an observable subset for the finite automaton* A. *Assume that* $card(\gamma(X_f)) < card(X_f)$ *and* $\gamma(X_f) \cap \gamma(X \backslash X_f) = \emptyset$. *If* X_f *is ballistically reachable from* $X \backslash X_f$, *and* x_f *is control-invariantly reachable in* X_f, *then there exists a FRF-automaton* $A_{FRF} = (X, Y, \Sigma', \delta, \gamma)$ *such that*

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4 card(X_f)}{radius(A)}.$$

Proof: The proof is found by investigating the size of the input alphabet necessary for generating the input in Lemma 3.11. We can let $\Sigma' = \{v_{ol}\} \times V^O \times \{0,1\}^O$, and let the input, $\sigma = (v, \kappa, \tau)$, be given by

$$\begin{cases} v = \text{arbitrary} \\ \kappa(o,v) = v_{ol} \ \text{ if } \ o = e \\ \kappa(o,v) = w_{obs}(o) \ \text{ if } \ o \in \gamma(X_f) \\ \kappa(o,v) = w_{attr}(o) \ \text{ if } \ o \in X_f \\ \tau(o) = 1 \ \text{ if and only if } \ o = x_f. \end{cases}$$

The size of the input space is thus $card(\Sigma') = (2card(V))^{(1+card(\gamma(X_f))+card(X_f))}$, which is less than or equal to $card(V)^{4card(X_f)}$. The description length $\mathcal{L}(\sigma, \Sigma')$ is thus given by $4card(X_f) \log_2(card(V))$. Now, since $\mathcal{C}(A_{ol}, x_f) \geq radius(A) \log_2(card(V))$, the theorem follows. ∎

This idea of using an open-loop control until a part of the state space is encountered, where observer based feedback control is effective, can be used recursively if the automaton admits a *shell-partitioning* of the state space. Formally, we say that a FRF-automaton admits a shell-partitioning if the state space X can be partitioned as $X = X_1 \cup X_2 \cup \cdots \cup X_n \cup \mathcal{X}_0 \cup \cdots \mathcal{X}_{n-1}$, in such a way that:

- $\gamma(X_i) \cap \gamma(X_j) = \emptyset,\ i \neq j,\ \gamma(X_i) \cap \gamma(\mathcal{X}_j) = \emptyset$;
- X_i is an observable subset, $i = 1, \ldots, n$;
- $card(\mathcal{X}_{n-1}) = \theta card(X_n)$, $card(\mathcal{X}_j) = \theta card(\mathcal{X}_{j+1})$, $j = 0, \ldots, n-2$, $card(X_j) = \lambda card(X_{j+1})$, $j = 1, \ldots, n-1$, where $\theta \gg 1,\ \lambda \gg 1,\ \theta \gg \lambda$.

Theorem 4.2 *(Nested Version of the Main Theorem) Assume that* $card(V) \geq 2$. *Let the automaton* A, *with* $radius(A) = \xi card(X)$ *for some* $\xi \in \mathbb{R}$, *admit a shell-partitioning such that* $card(\gamma(X_i)) < card(X_i)$. *Let* X_1 *be ballistically reachable from* $\mathcal{X}_0$, *and let* $x_f \in X_n$ *be control-invariantly reachable in* X_n. *If there exist* $x_i \in X_i$, $i = 1, \ldots, n-1$ *such that* x_i *is control-invariantly reachable in* X_i, *and* X_{i+1} *is ballistically reachable from* x_i, *then there exists a FRF-automaton* $A_{FRF} = (X, Y, \Sigma', \delta, \gamma)$ *such that*

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4}{\xi\lambda} \left(\frac{\lambda}{\theta}\right)^n.$$

Proof: We need to construct an observer that makes it possible to reach x_f using as few instructions as possible. We propose $Z = \{e_1, \ldots, e_n\} \cup Y \cup \ldots \cup Y^{\bar{p}_{obs}-1} \cup X$, where $\bar{p}_{obs}$ is the largest of all the positive integers p_{obs_i} associated with each observable subset $X_i,\ i = 1, \ldots, n$. We furthermore let

$$h(z,y) = \begin{cases} y & \text{if } z \notin X \cup \{e_1, \ldots, e_n\} \\ z & \text{otherwise} \end{cases}$$
$$O = \{e_1, \ldots, e_n\} \cup \gamma(X_1) \cup \ldots \cup \gamma(X_n) \cup X_1 \cup \ldots \cup X_n,$$

and define z_{k+1} as

$$\begin{cases} e_i \text{ if } z_k = e_i \text{ and } y_k \notin \gamma(X_i) \\ y_k \text{ if } z_k = e_i \text{ and } y_k \in \gamma(X_i) \\ z_k \cdot y_k \text{ if } z_k \in Y^q,\ q < p_{obs_i} - 1 \\ \Xi(z_k \cdot y_k, w_k(h(z_k, y_k), v_k)) \text{ if } z_k \in Y^{p_{obs_i}-1} \\ \delta(z_k, v_k) \text{ if } z_k \in X_i. \end{cases}$$

If we furthermore let $\Sigma' = \{\hat{v}\} \times V^O \times \{0,1\}^O$, for some arbitrary $\hat{v} \in V$, we can choose to use the input $\sigma = (v, \kappa, \tau)$, where

$$\begin{cases} v = \hat{v} \\ \kappa(o,v) = v_{ol_i} \ \text{ if } \ o = e_i \\ \kappa(o,v) = w_{obs_i}(o) \ \text{ if } \ o \in \gamma(X_i) \\ \kappa(o,v) = w_{attr_i}(o) \ \text{ if } \ o \in X_i \\ \tau(o) = 1 \ \text{ if and only if } \ o = x_f. \end{cases}$$

Here v_{ol_i} is the open-loop control that drives the automaton from x_i to X_{i+1}, w_{obs_i} is the feedback control that makes the observer converge on X_i, and w_{attr_i} is the feedback control that drives the automaton to x_i without the trajectory leaving X_i. It is straight forward to check that by using this input, the FRF-automaton $(X, Y, \Sigma', \delta, \gamma)$ drives from any initial state to x_f.

The size of the input space is thus $(2card(V))^{\sum_{i=1}^{n}(1+card(\gamma(X_i))+card(X_i))}$, which is less than or equal to $card(V)^{\sum_{i=1}^{n} 4card(X_i)}$.

We have that $\mathcal{C}(A_{ol}, x_f) \geq radius(A) \log_2(card(V))$, or in other words that $\mathcal{C}(A_{ol}, x_f)$ is greater than or equal to

$$\xi \log_2(card(V)) \left(\sum_{i=0}^{n-1} card(\mathcal{X}_i) + \sum_{i=1}^{n} card(X_i) \right).$$

We also so know that $\mathcal{C}(A_{FRF}, x_f)$ is upper bounded by $4 \log_2(card(V)) \sum_{i=1}^{n} card(X_i)$. Thus

$$\frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} \leq \frac{4}{\xi} \frac{\sum_{i=1}^{n} card(X_i)}{\left(\sum_{i=0}^{n-1} card(\mathcal{X}_i) + \sum_{i=1}^{n} card(X_i) \right)}.$$

Since $card(\mathcal{X}_{n-p}) = \theta^p card(X_n),\ p = 1, \ldots, n$, and $card(X_{n-p}) = \lambda^p card(X_n),\ p = 0, \ldots, n-1$, we get

$$\begin{aligned} \sum_{i=1}^{n} card(X_i) &= \left(\frac{1-\lambda^n}{1-\lambda} \right) card(X_n) \\ \sum_{i=0}^{n-1} card(\mathcal{X}_i) &= \left(\frac{1-\theta^n}{1-\theta} \right) \theta card(X_n). \end{aligned}$$

Thus

$$\begin{aligned} \frac{\mathcal{C}(A_{FRF}, x_f)}{\mathcal{C}(A_{ol}, x_f)} &\leq \frac{4}{\xi} \frac{\frac{1-\lambda^n}{1-\lambda}}{\frac{1-\lambda^n}{1-\lambda} + \theta \frac{1-\theta^n}{1-\theta}} \\ &= \frac{4}{\xi} \frac{(1-\lambda^n)(1-\theta)}{(1-\lambda^n)(1-\theta) + \theta(1-\lambda)(1-\theta^n)} \\ &\approx \frac{4}{\xi} \frac{\theta\lambda^n}{\theta\lambda^n + \theta^{n+1}\lambda} \approx \frac{4}{\xi\lambda} \left(\frac{\lambda}{\theta} \right)^n. \end{aligned}$$

Here we have made use of the fact that $\theta \gg 1,\ \lambda \gg 1,\ \theta \gg \lambda$, and the theorem follows. ∎

It should be noted that this means, first of all, that the fraction between the different complexities goes to zero as n goes to infinity. Furthermore, this rate is determined by how small the observable subsets are, as well as how fast the sizes of these subsets are decreasing. An interpretation of this result is that we let the automaton use shorter and shorter open-loop steps. At the same time it localizes its position between those steps, using a closed-loop control strategy.

References

[1] M. Arbib, Ed. *Algebraic Theory of Machines, Languages, and Semigroups*, Academic Press, New York, 1968.

[2] H.S. Black. Stabilized Feedback Amplifiers. In *The Bell System Technical Journal*, Jan. 1934.

[3] R.W. Brockett. Hybrid Models for Motion Control Systems. In *Perspectives in Control*, H. Trentelman and J.C. Willems, eds., pp. 29-54, Birkhauser, Boston, 1993.

[4] F. Buckley and F. Harary. *Distance in Graphs.* Addison-Wesley Publishing Company, New York, 1990.

[5] T.M. Cover and J.A. Thomas. *Elements of Information Theory*, John Wiley & Sons, Inc., New York, 1991.

[6] W. Fleming and P.L. Lions. *Stochastic Differential Systems, Stochastic Control Theory and Applications.* The IMA Volumes in Mathematics and Its Applications, Vol. 10, Springer Verlag, New York, 1987. 1992.

[7] M. Itô. *The Cerebellum and Neural Control.* Raven Press, New York, 1984.