Proceedings of the 2007 American Control Conference
Marriott Marquis Hotel at Times Square
New York City, USA, July 11-13, 2007

A Combined Seed-Identification and Generation Analysis Algorithm for Self-Reproducing Systems

Amor Menezes and Pierre Kabamba

Abstract— **This paper is motivated by the need to minimize the payload mass required to establish an extraterrestrial robotic colony. The basic premise is that the colony will consist of individual robots that have the capability to self-reproduce. In this paper, self-reproduction is achieved by the actions of a robot on available resources. Hence, a seed for the colony consists of a set of robots and a set of resources. The technical problem addressed is the identification of a seed for a class of generation systems. An algorithm is provided for the solution of this problem, and is illustrated on a self-replicating system that has been documented in the literature.**

I. INTRODUCTION

SCIENTIFIC research conducted to explore the field of self-reproduction has demonstrated much promise, with the potential of significant impact on such diverse areas as space colonization, bioengineering, evolutionary software and autonomous manufacturing. This field owes much to the efforts of John von Neumann [1], whose work on the theory of automata in the 1940s and 1950s inspired extensive research into the simulation and implementation of such self-reproducing systems as: cellular automata, computer programs, kinematic machines, molecular machines, and even robotic colonies. A detailed overview of the research activities in the field is presented in [2] and [3].

Von Neumann postulated the existence of a threshold of complexity below which any attempt at self-reproduction was doomed to degeneracy. However, he did not define either complexity or degeneracy, nor did he go on to compute the threshold's value. An extensive literature survey in [4] indicates that no one had published an evaluation of this threshold in the following 60 years. Recently, [5] developed a novel theory of generation that is able to compute this von Neumann threshold. The results in [5] included a necessary and sufficient condition for non-degenerate offspring, i.e., offspring with the same reproductive capability as the progenitor. Reference [6] presented a generalized version of these results, and also demonstrated parallels with information theory. The present paper extends these results by providing an algorithm that identifies elements necessary for the initiation of a given self-reproducing system.

The remainder of this section presents a rationale for seed-identification, and surveys background material on Probabilistic Generation Theory [6]. Section II discusses the necessary assumptions and definitions for seed-identification, before proceeding to outline a combined seed-identification and generation analysis algorithm. Section III illustrates the application of the algorithm to a self-replicating system documented in [7] and [8].

A. Menezes and P. Kabamba are with the Department of Aerospace Engineering at the University of Michigan, 1320 Beal Avenue, Ann Arbor, MI 48109, USA amenezes@umich.edu; kabamba@umich.edu

A. Motivation

Within the context of extra-terrestrial colonization, current phased approaches to Martian exploration see the development of an enduring robotic presence on the Moon in the next five years. Several space agency roadmaps, of which [9] is typical, suggest that individual countries will deploy advanced robots on an as-needed basis to expand the size of an established colony. It is well known in the aerospace community that for every unit mass of payload to be launched into space, eighty additional units of mass are required to be launched as well [10]. Instead, it would be much more efficient to have robots endowed with the capacity for self-reproduction. These machines would be able to utilize available resources on-site to enlarge their numbers when deemed necessary for a given task. Such technology is not dependent on either the launch capabilities or the fiscal constraints surrounding the multiple launches of robots required for the colony, and therefore may provide a highly cost-effective solution to the problem of establishing extra-terrestrial colonies.

In order to minimize mass, it would be even more efficient to recognize the required elements for the initiation of a self-reproducing system, and send the smallest quantity of these elements into space. The identification of this "seed" is the goal of this paper.

B. Highlights of Probabilistic Generation Theory

We first state what is meant by the following terms that will be used throughout the paper: reproduction, replication, self-reproduction, and self-replication. For a historical perspective of the first two terms, the reader is referred to Freitas' excellent discussion on the subject in [2]. We consider reproduction in biological systems to imply the capacity for genetic mutations and the potential for evolution. Thus from an information standpoint, reproduction involves a change to the DNA code during the generation of progeny. Likewise, we will take *reproduction* in an artificial generation system to imply a change in the information specifications of an offspring. We reserve the term *replication* for progeny that have identical information content to that of the progenitor. *Self-reproducing* and *self-replicating* will be used to refer to those entities that perform the information equivalent of asexual reproduction or mitosis, i.e., the entities can

1-4244-0989-6/07/$25.00 ©2007 IEEE.

reproduce or replicate based on the information specifications of only one progenitor.

The theory surveyed here formalizes self-reproduction by "machines," a term describing any entity that is capable of producing an offspring regardless of its physical nature. Thus a robot, a bacterium, or even a piece of software code is considered to be a machine in this theory if they can each produce another robot, bacterium or some lines of code respectively. These machines require resources to self-reproduce, and each resource is chosen with some prior probability. The selected resource is then manipulated by the parent machine via an embedded generation action to produce an outcome, which itself may or may not be a machine. Thus we can state the following:

Definition 1: A *Probabilistic Generation System* is a quintuple $\Gamma = (U, M, R, P, G)$, where

- U is a *universal set* that contains machines, resources and outcomes of attempts at self-reproduction;
- $M \subseteq U$ is a *set of machines* in the context described;
- $R \subseteq U$ is a *set of resources* that can be utilized for self-reproduction;
- P is a probability mass function (pmf) on R, that is, $R \to \mathbb{R}$ with $P[r] \in [0,1]$ and $\sum_i P[r_i] = 1$;
- $G : M \times R \to U$ is a generation function that maps a machine and a resource into an outcome in the universal set, and not necessarily in the set of machines.

Furthermore, it is possible that $M \cap R \neq \oslash$, and also $M \cup R \neq U$, as illustrated in Fig. 1. The former implies that machines can belong to the set of resources, and the latter states that outcomes of attempts at generation may be neither machines nor resources.

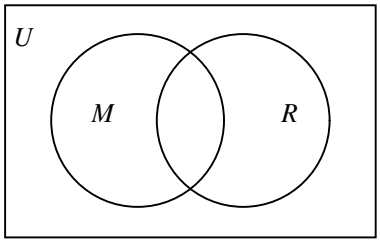


Fig. 1. Pictorial representation of Definition 1.

One can define an indicator function, I, over a predicate, p, such that:

$$I(p) = 1 \ if \ \ p = True$$

$$I(p) = 0 \ if \ \ p = False.$$

Thus, the probability of a machine $x \in M$ processing a resource $r \in R$ to generate an outcome $y \in U$ may be written as:

$$P[y = G(x,r)] = \sum_{r \in R} I(y = G(x,r)) \cdot P[r]. \qquad (1)$$

If, in (1), $P[y = G(x,r)] > \varepsilon$, where $\varepsilon > 0$, then we say that "x is ε-capable of generating y," and we call the process *ε-reproduction*. If we have $P[x = G(x,r)] > \varepsilon$ in (1), where $\varepsilon > 0$, then we say that "x is ε-capable of generating itself," and we call the process *ε-replication*.

Of course, if we set $\varepsilon = 0$, then we allow every machine to ε-reproduce no matter what resource is selected. This is termed *Free Generation*. If $\varepsilon = 1$, then the deterministic theory of generation proposed in [5] is recovered. This is called *Strict Generation* or *Unity Generation*.

Definition 2: The *Generation Sets* in a probabilistic generation system are defined as:

- $M_0 = M$, the set of all machines;
- M^{ε}_{i+1}, the set of all machines that are ε-capable of producing a machine of M^{ε}_i, $\forall i \geq 0$. That is, for $x \in M^{\varepsilon}_{i+1}$, $\exists\ y \in M^{\varepsilon}_i$ such that $P[y = G(x,r)] > \varepsilon$.

These sets are nested with the innermost generation set being important for self-reproduction. This set can be defined as:

$$M^{\varepsilon}_{\infty} = \bigcap_{i=0}^{\infty} M^{\varepsilon}_i. \qquad (2)$$

It is shown in [6] that generation always proceeds outwards. Also, the notion of the rank of a probabilistic generation system, as defined below, is emphasized.

Definition 3: The *rank of a probabilistic generation system*, $\rho^{\varepsilon}(\Gamma)$, where $\Gamma = (U, M, R, P, G)$ with generation sets M^{ε}_i, $i \geq 0$, is the smallest integer ρ such that $M^{\varepsilon}_{\rho} = M^{\varepsilon}_{\rho+1}$. If $\forall i, M^{\varepsilon}_i \neq M^{\varepsilon}_{i+1}$, then the generation system has infinite rank.

For a probabilistic generation system of finite rank ρ, the nesting of the generation sets stop at the integer ρ. All generation sets of order greater than ρ (up to and including M^{ε}_{∞}) are equal. A probabilistic generation system that has a finite number of machines always has finite rank.

Definition 4: The *rank of a machine*, $\rho^{\varepsilon}(x)$, in a probabilistic generation system $\Gamma = (U, M, R, P, G)$ with generation sets M^{ε}_i, $i \geq 0$, and $\rho^{\varepsilon}(\Gamma) = \rho$, is equal to i if $x \in M^{\varepsilon}_i \setminus M^{\varepsilon}_{i+1}$ ("deficient generation rank"), or is equal to ρ if $x \in \bigcap_{i=0}^{\infty} M^{\varepsilon}_i$ ("full generation rank").

Definition 5: An *ε-generation cycle* is a sequence of ε-generations resulting in the production of a machine identical to itself after n generations.

Machines capable of ε-replication (an ε-generation cycle of order one) in a probabilistic generation system must belong to M^{ε}_{∞}, and any exit from M^{ε}_{∞} is irreversible. It is possible for offspring machines to belong to M^{ε}_{∞} as long as their progenitors do as well. Thus the requirements for non-degenerate ε-reproduction and ε-replication are quantified. It is proved in [6] that there is a minimum threshold of rank above which a machine is able to ε-generate an offspring without a decrease in generation rank. We call this the *von Neumann Rank Threshold*, τ^{ε}_r, and define

$$\tau^{\varepsilon}_r = \rho^{\varepsilon}(\Gamma). \qquad (3)$$

The reader is referred to the material in [6] for proofs of the above statements, as well as many other insights into the information requirements of self-reproducing sytems.

II. SEED IDENTIFICATION

To formulate the seed requirements of a self-reproducing system in a mathematically precise way, we first make some assumptions about the nature of the probabilistic generation system. These assumptions help structure the seeding problem but, in some cases, unfortunately make non-optimal seeds possible as described.

A. Assumptions

We first assume that every resource in the set R of a probabilistic generation system is utilized by a progenitor machine so that another machine can be produced.

Assumption 1: Given a probabilistic generation system $\Gamma = (U, M, R, P, G)$, we assume that $\forall r \in R, \exists x \in M$ such that $G(x,r) \in M$.

As we will see later, this assumption will simplify the selection procedure of resources since it points to the condition that all resources are necessary to ε-produce an offspring. Hence, a seeding algorithm can simply identify all possible resources as constituents of a seed. If we accept that all resources are necessary however, then we allow ourselves the possibility of selecting redundant resources. For instance, if there exist two resources such that a progenitor machine will produce the same offspring with each of those two resources, then by taking both resources to belong to the seed, a redundant selection has been made and the resulting seed is non-optimal. We ignore this possibility and consider it an avenue for future refinement.

We allow for complexity within the resource set, and enable each resource to itself contain an ordered list of physical elements that may include machines. We therefore define a containment relation as follows.

Definition 6: If machine x_i belongs to an ordered list of the elements of resource r_j, then we say that x_i is *contained* in r_j, and we write $x_i \prec r_j$, where "$\prec$" is the *Containment Operator*.

Of course, if machine x_i is a resource itself, then this relation still holds true.

Definition 7: If machines $x_1, x_2, \ldots, x_v$ are contained in resource r, then we use the notation $r \backslash (x_1, x_2, \ldots, x_v)$ to refer to an ordered list of the elements of r that does not contain the machines $x_1, x_2, \ldots, x_v$.

Assumption 2: Given a probabilistic generation system $\Gamma = (U, M, R, P, G)$, we assume that if machine x is contained in resource r, $x \prec r$, then the ordered list of the elements of r that does not contain the machine x also belongs to the set of resources, i.e., $r \backslash x \in R$.

Next, we assume that every machine in the probabilistic generation system has a progenitor machine.

Definition 8: A *Surjective Generation System* is a probabilistic generation system $\Gamma = (U, M, R, P, G)$ where $\forall y \in M, \exists x \in M$, and $\exists r \in R$ such that $y = G(x,r)$.

We further assume that there exists a machine in the probabilistic generation system that is capable of producing any machine in the system after m generations. This is a special case of a surjective generation system.

Assumption 3: We assume that in the probabilistic generation system $\Gamma = (U, M, R, P, G)$, $\exists x_1 \in M$ such that $\forall x \in M, \exists \mu_0 \leq \mu, \exists r_1, r_2, \ldots, r_{\mu_0}$ selected from R such that

$$G(\ldots G(G(G(x_1, r_1), r_2), r_3) \ldots, r_{\mu_0}) = x.$$

We are now in a position to define the seed.

B. Problem Definition

Using the assumptions in Section IIA, we formalize the definition of a seed as follows.

Definition 9: A *seed of order k* is a set

$$\begin{aligned} S &= \{x_1\} \cup R_0, \text{ where} \\ R_0 &= \{r_1, r_2, \ldots, r_k\}, \text{ and} \\ R_0 &\subseteq R, \end{aligned}$$

such that $\forall x \in M, \exists \mu_0 \leq k, \exists r_1, r_2, \ldots, r_{\mu_0}$ selected from R_0 such that

$$G(\ldots G(G(G(x_1, r_1), r_2), r_3) \ldots, r_{\mu_0}) = x.$$

We design an algorithm to produce a seed as per the above definition, and do not impose a restriction on the order of k.

C. Seed Identification Algorithm Methodology

Based on the assumptions of the probabilistic generation system and the resulting seed definition, one possible seed is the set containing all resources and a machine of highest rank. However, a more sophisticated approach is possible, one that takes into account the possibility that machines of deficient rank (see Definition 4) can be used as resources or even constitute them, and also the fact that machines belonging to generation cycles or loops need to be isolated.

The approach to developing the Seed Identification Algorithm is similar to the Generation Analysis Algorithm (GAA) stated in [5], and in fact utilizes the GAA in its operation. The GAA employs the concept of an *outer layer*, first introduced in [5] and defined as follows.

Definition 10: In a generation system $\Gamma = (U, M, R, P, G)$, the *outer layer* is the set $M_0 \backslash M_1^{\varepsilon}$. This is the set of machines such that, no matter what resource they use, they produce an offspring that is no longer a machine, i.e.,

$$\{x \in M : \forall r \in R, G(x,r) \notin M\}.$$

After an outer layer is removed, a reduced order generation system remains. The GAA works by peeling away the outer layers of each of the generation systems Γ_i, $0 \leq i \leq \rho$. We apply a similar notion to the develoment of a seed identification algorithm.

By Assumption 1, any resource that does not contain a machine is assigned to be a part of the seed. Next, any machines that belong to the outer layer of the probabilistic generation system, the set $M_0 \backslash M_1^{\varepsilon}$, are the machines of lowest rank that will not help to perpetuate the system, and hence belong to the set that is not the seed, $\bar{S}$. Thus the outer layer of the machine set needs to be identified.

Let $M = \{x_1, x_2, \ldots, x_n\}$ and consider the *Descendancy Matrix*, defined as the $n \times n$ matrix of integers, D, such that

$$\begin{aligned} D_{ij} &= 1 \text{ if } \exists r \in R : P[x_j = G(x_i, r)] > \varepsilon, \quad (4) \\ &= 0 \text{ otherwise}, \quad (5) \end{aligned}$$

that is, $D_{ij} = 1$ if machine x_i is ε-capable of generating machine x_j, and $D_{ij} = 0$ otherwise.

Now let $R = \{r_1, r_2, \ldots, r_m\}$ and consider the *Containment Matrix*, defined as the $n \times m$ matrix of integers, C, such that

$$C_{ij} = 1 \text{ if } x_i \prec r_j, \quad (6)$$
$$= 0 \text{ otherwise}, \quad (7)$$

that is, $C_{ij} = 1$ if machine x_i is contained in resource r_j, or is indeed a resource itself, and $C_{ij} = 0$ otherwise.

Let the *Seed Matrix*, be defined as the $n \times (n+m)$ matrix of integers, Σ, such that

$$\Sigma = \begin{bmatrix} D & C \end{bmatrix}. \quad (8)$$

Then the set of resources that do not contain any machine consists of those resources such that the corresponding columns of matrix C are zero. These columns may be removed from Σ, and the respective resources added to S. The outer layer of M consists of those machines in the matrix D that have corresponding rows of zeroes. These rows, and the corresponding machine columns (even if not all zero), may be removed from Σ, and the respective machines added to $\bar{S}$. If any of the removed machines exactly equal one of the resources, then that corresponding column may be removed from C as well.

We are now left with a reduced order probabilistic generation system, that can be seeded in a similar fashion. The process of removing resources in the containment matrix, followed by removing lower-rank machines in the descendancy matrix and possibly in the containment matrix too, can be repeated in order to deflate the seed matrix until there are no more resources left to remove. For each iteration, the columns of zeroes in the matrix C now denote those resources that are devoid of lower-rank machines. Of course, if a particular resource is nothing but a lower-rank machine, then this algorithm removes it from consideration as a seed resource.

Once the iterations are over, one of two conditions may occur. It could be that all columns of the containment matrix have been removed, leaving nothing but the descendancy matrix. Thus all k resource elements of S have been found, $k \leq m$. If D can be further deflated, then this should be continued in order to obtain x_1, the machine of highest rank. When deflations of D are no longer possible, the machine of highest rank can be added as the x_1 required by S. If several machines are of equally high rank, then any one of these machines may be selected as x_1.

If, on the other hand, there are still resource columns left, but they cannot be removed due to the presence of a 1, then each resource can now be added to S as long as the corresponding machine that the resource requires (the row with the 1) is included. Assumption 3 guarantees that there should be only one machine that the resources point to, and it will be the required x_1 for S. These conditions are obviously satisfied if a remaining resource is a machine itself.

We summarize this algorithm in the next subsection.

D. Combined Seed Identification and Generation Analysis (SIGA) Algorithm

Inputs: a generation system $\Gamma = (U, M, R, P, G)$, where $M = \{x_1, x_2, \ldots, x_n\}$ and $R = \{r_1, r_2, \ldots, r_m\}$, satisfying Assumptions 1 through 3.

Outputs: the sets $(M_0 \backslash M_1^\varepsilon)$, $(M_1^\varepsilon \backslash M_2^\varepsilon)$, …, $(M_{\rho-1}^\varepsilon \backslash M_\rho^\varepsilon)$, $M_\rho^\varepsilon = M_\infty^\varepsilon$, the von Neumann rank threshold $\tau_r^\varepsilon = \rho^\varepsilon(\Gamma)$, the seed set S, and its order k.

1) Compute the $n \times n$ matrix D, the $n \times m$ matrix C, and the $n \times (n+m)$ matrix Σ.
2) Initialize $i = 0$, $k = 0$.
3) While R is not empty, and C has at least one column of zeroes, and M is not empty, do:
   - For each column of zeroes in C, add r_j to S and $k = k+1$.
   - Update R by removing the resource elements corresponding to zero columns of C.
   - Update Σ by removing the corresponding zero columns of C.
   - Return $(M_i^\varepsilon \backslash M_{i+1}^\varepsilon)$, the set of machines corresponding to zero rows of D.
   - Update M by removing the machines corresponding to zero rows of D.
   - Update Σ by removing the zero rows and corresponding zero columns of D, and any columns in C for which the resource exactly equals the machine that has a zero row in D.
   - $i = i+1$.
4) If R is empty then:
   - While M is not empty and D has at least one row of zeroes, do:
     - Return $(M_i^\varepsilon \backslash M_{i+1}^\varepsilon)$, the set of machines corresponding to zero rows of D.
     - Update M by removing the machines corresponding to zero rows of D.
     - Update Σ by removing the zero rows and corresponding zero columns of D.
     - $i = i+1$.
   - Return k.
   - Return $M_\infty^\varepsilon = M$
   - If $|M_\infty^\varepsilon| > 1$ then pick an element of M_∞^ε to add to S, else add machine in M_∞^ε to S.
5) If C does not have a column of zeroes then:
   - For each remaining column in C, do:
     - Add r_j to S
     - $k = k+1$
     - Add the machine for which r_j has a one in its corresponding row to S.
     - Update R by removing these resources.
     - Update Σ by removing the last columns of C.
   - While M is not empty and D has at least one row of zeroes, do:
     - Return $(M_i^\varepsilon \backslash M_{i+1}^\varepsilon)$, the set of machines corresponding to zero rows of D.

- Update M by removing the machines corresponding to zero rows of D.
- Update Σ by removing the zero rows and corresponding zero columns of D.
- $i = i+1$.

• Return k.
• Return $M_\infty^\varepsilon = M$.

6) Return $\tau_r^\varepsilon = i$.
7) Stop.

The SIGA algorithm is guaranteed to stop after a finite number of steps. Each *while* loop removes elements from a set with finite cardinality, stopping once a set is depleted.

III. EXAMPLE APPLICATION OF THE SIGA ALGORITHM

We can use Probabilistic Generation Theory and the SIGA algorithm to analyze the *Semi-Autonomous Replicating System* designed by Chirikjian et al. [7], [8]. For a more accurate analysis, let us consider this generation system under the auspices of Strict Generation with $\varepsilon = 1$, since the designers of the system require the prototype robot to construct a replica of itself in a series of deterministic steps. To be explicit, the sytem is designed in such a way that at any given stage in the replication process, only one resource has a probability of being selected, and that probability is 1. The pmf over the resources is dynamically updated at each stage of the process, to reflect which resource now has the probability of being selected. With $I(y = G(x,r))$ always 1, Strict Generation is a simplistic representation that also allows us drop the ε in our notation.

Take M to be the set of all entities that are each made up of two or more LEGO Mindstorm kit components fixed together in some way. Let

$$\begin{aligned} M &= \{x_1, x_2, x_3, x_4, x_5, x_6\}, \text{ and} \\ R &= \{r_1, r_2, r_3, r_4, r_5, r_6\}, \end{aligned}$$

where we define each of the constituent machines and resources in the manner that follows. The sequence of generation steps is also outlined. The replication process is illustrated in Fig. 2.

$x_1 \triangleq$ Prototype Robot
$r_1 \triangleq$ (conveyor-belt/sensor unit, docking unit, electrical connector, central controller unit (CCU), electrical cable)
$x_2 \triangleq$ Chassis Assembly Station
$x_2 = G(x_1, r_1)$
$r_2 \triangleq (x_1$, chassis, robot control system (RCX))
$x_3 \triangleq$ RCX-Chassis Assembly
$x_3 = G(x_2, r_2)$
$r_3 \triangleq$ Gripper Assembly Station $\triangleq$ (CCU, electrical connector, ramp and lift system, gripper)
$x_4 \triangleq$ Prototype Robot with Gripper
$x_4 = G(x_1, r_3)$
$x_1 = G(x_4, r_3)$
$r_4 \triangleq$ (left LEGO hook, right LEGO hook, CCU, electrical connector, stationary docking sensor, motorized pulley unit)
$x_5 \triangleq$ Motor and Track Assembly Station
$x_5 = G(x_4, r_4)$
$r_5 \triangleq$ (motor/sensor unit, x_3)
$x_6 \triangleq$ RCX-Chassis-Motor Assembly
$x_6 = G(x_1, r_5)$
$r_6 \triangleq (x_1$, left LEGO track, right LEGO track, $x_6)$
$x_1 = G(x_5, r_6)$
$r_7 \triangleq r_2 \backslash x_1 \triangleq$ (chassis, robot control system (RCX))
$r_8 \triangleq r_5 \backslash x_3 \triangleq$ (motor/sensor unit)
$r_9 \triangleq r_6 \backslash (x_1, x_6) \triangleq$ (left LEGO track, right LEGO track)

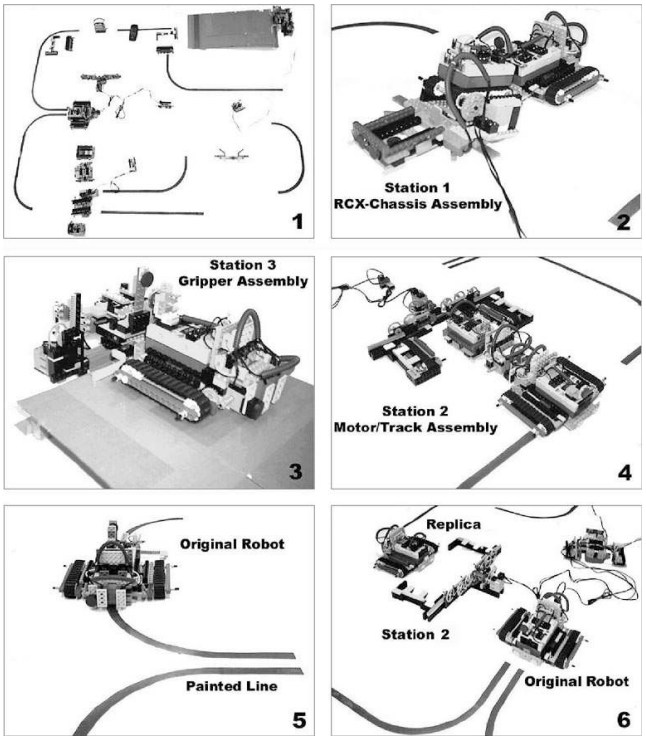


Fig. 2. The semi-autonomous replication process of the Suthakorn-Kwon-Chirikjian robot [7].

It follows that we have the generation representation indicated in Fig. 3.

With the SIGA algorithm,

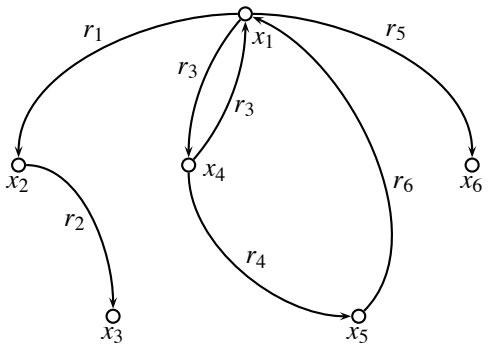
$$D_0 = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$C_0 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

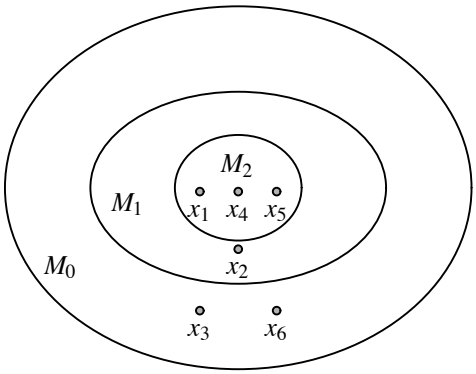
$$\Sigma_0 = \begin{bmatrix} D_0 & C_0 \end{bmatrix};$$

so that r_1, r_3 and r_4 can be immediately identified as part of the seed. D_0 can also be deflated, yielding

$$M_0 \backslash M_1 = \{x_3, x_6\}.$$



(a) Generation diagram.



(b) Generation set structure.

Fig. 3. Representations of the Suthakorn-Kwon-Chirikjian semi-autonomous replicating system, $\tau_r = 2$.

We are left with

$$D_1 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix};$$

$$C_1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix};$$

$$\Sigma_1 = \begin{bmatrix} D_1 & C_1 \end{bmatrix};$$

so that $r_5 \backslash x_3 = r_8$ now belongs to the seed, and also

$$M_1 \backslash M_2 = \{x_2\}.$$

Since

$$C_2 = \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

cannot be further deflated, we add $r_2 \backslash x_1 = r_7$, $r_6 \backslash (x_1, x_6) = r_9$, and x_1 to the seed. Also,

$$D_2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

cannot be further reduced, giving us

$$M_2 = M_\infty = \{x_1, x_4, x_5\}$$

and $\tau_r = 2$. The seed set of order 6 for this system is

$$S = \{x_1\} \cup \{r_1, r_3, r_4, r_7, r_8, r_9\}.$$

We have thus arrived at a very logical, yet informative result - the original robot (with and without the gripper) and the final assembly station are the most important elements of the semi-autonomous replicating system, and the original robot is (of course!) needed to initiate the system, assuming the existence of plentiful resources.

IV. CONCLUSIONS AND FUTURE WORK

A novel algorithm to identify the seed of a generation system has been proposed. It utilizes the earlier Generation Analysis Algorithm of [5], but expands the scope to consider resources and their composition. It is capable of dealing with machines of deficient rank that are used as resources, as well as isolating a seed machine from a generation cycle or loop.

The avenues for current and immediate future research include investigating the relationship between the rank of a probabilistic generation system and the size of the seed, and developing the necessary and sufficient conditions to produce an optimal seed. This also gives rise to the issue of control; specifically, how does one control the rank of a generation system to produce an optimal seed? With the theory in place to analyze generation systems, the next step is to develop theory to synthesize generation systems.

The SIGA algorithm needs to be extended to 1) allow for the determination of a seed of order k, with k pre-specified; 2) incorporate some notion of the quantity of a seed resource needed to perpetuate a system; 3) recognize and compensate for time constraints that may impose a larger-size seed upon the system; and 4) eliminate any redundant resources. These four apparent limitations will be overcome in future work.

REFERENCES

[1] J. von Neumann, *Theory of Self-Reproducing Automata*, A. Burks, Ed. University of Illinois Press, 1966.

[2] R. A. Freitas Jr. and R. C. Merkle, *Kinematic Self-Replicating Machines*. Landes Bioscience, 2004. [Online]. Available: http://www.molecularassembler.com/KSRM.htm

[3] M. Sipper, "Fifty years of research on self-replication: An overview," *Artifical Life*, vol. 4, no. 3, pp. 237–257, 1998.

[4] P. Owens and A. G. Ulsoy, "Self-replicating machines: Preventing degeneracy," The University of Michigan, Tech. Rep. CGR-06-02, 2006.

[5] P. Kabamba, "The von neumann threshold of self-reproducing systems: Theory and computation," The University of Michigan, Tech. Rep. CGR-06-11, 2006.

[6] A. Menezes and P. Kabamba, "Information requirements for self-reproducing systems in lunar robotic colonies," in *Proceedings of the 57th International Astronautical Congress*, no. IAC-06-A5.P.04, 2-6 October 2006.

[7] G. S. Chirikjian, Y. Zhou, and J. Suthakorn, "Self-replicating robots for lunar development," *IEEE/ASME Transactions on Mechatronics*, vol. 7, no. 4, Dec. 2002.

[8] J. Suthakorn, Y. T. Kwon, and G. S. Chirikjian, "A semi-autonomous replicating robotic system," in *Proceedings of the 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, July 2003.

[9] B. Foing, "Roadmap for robotic and human exploration of the moon and beyond," in *Proceedings of the 56th International Astronautical Congress*, no. IAC-05-A5.1.01, 17-21 October 2005.

[10] J. R. Wertz and W. J. Larson, Eds., *Space Mission Analysis and Design*, 3rd ed. Microcosm Press, 1999.