12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference
10 - 12 September 2008, Victoria, British Columbia Canada

AIAA 2008-6058

A Modeling Framework for Applying Discrete Optimization to System Architecture Selection and Application to In-Situ Resource Utilization

Ariane B. Chepko[1] and Olivier de Weck[2]
Massachusetts Institute of Technology, Cambridge, MA, 02139

William A. Crossley[3]
Purdue University, West Lafayette, IN, 47906

Edgardo Santiago-Maldonado[4]
NASA Kennedy Space Center

and

Diane Linne[5]
NASA Glenn Research Center

This paper presents an approach to apply optimization to the selection of a system architecture. Early-stage design decisions that define the subsystem and component technologies that comprise a system architecture are often made with information generated from a limited number of trade studies and a historical background. These decisions can lock in a design too early, resulting in a sub-optimal or even infeasible design. Decisions for systems that have little or no historical background, such as lunar in-situ resource utilization (ISRU) systems, have only analyses and tests to be based on. To enable a wider search of the design space, a modeling framework is discussed that decomposes a system in a hierarchical fashion to describe primary subsystem functions and their technology implementation alternatives. This framework allows a system model to be built that captures all of the subsystem technology alternatives in one reconfigurable model for which the categorical, discrete technology decisions can be treated as design variables in an optimization routine. The NASA ISRU System Model follows this framework and is used as an application of system architecture optimization. A genetic algorithm is used to explore both the discrete and continuous design space of the model. Because the current ISRU System Model has a small set of discrete variables, the performance and computational cost of the genetic algorithm search are compared to a previously developed ISRU optimization scheme that involves full enumeration of the discrete variables followed by gradient-based optimization with the continuous variables. The initial tests of the genetic algorithm approach provide comparable results to the previously used approach, requiring 5100 function evaluations compared to a range of 4800-10,000 function evaluations with the enumeration methods.

Nomenclature

f	=	objective function
g_i	=	inequality constraint
r_p	=	penalty multiplier
x_i	=	design variable

[1] Graduate Research Assistant, Aeronautics and Astronautics, MIT/37-344a, AIAA Student Member.
[2] Associate Professor, Aeronautics and Astronautics, MIT/33-410, AIAA Associate Fellow.
[3] Associate Professor, Aeronautics and Astronautics, 701 West Stadium Ave., West Lafayette, IN 47907, AIAA Associate Fellow.
[4] Applied Science Division, NASA/KSC KT-D-3
[5] Senior Research Engineer, Propulsion and Propellants Branch, NASA/GRC MS 301-3, AIAA Senior Member

Copyright © 2008 by Ariane Chepko. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

x_{LB} = design variable lower bounds
x_{UB} = design variable upper bounds
GA = Genetic Algorithm
ISRU = In-Situ Resource Utilization
PEM = Proton Exchange Membrane (electrolysis system)
SA = Simulated Annealing
SQP = Sequential Quadratic Programming

I. Introduction

The design of large-scale engineering systems is a complex process that involves discrete choices between subsystem options. For example, in the design of a satellite, choices in the type of propulsion system (e.g. electric, bipropellant), power supply system (solar, nuclear), and thermal control system (active, passive) all must be made early in the design process. In addition, each subsystem choice carries with it different technology options (e.g. hypergolic or cryogenic propellants), compounding the number of potential system architectures. Such problems are combinatorial in nature, having categorical, discrete variables rather than only continuous decision variables as in classical design problems where an architecture has already been chosen. In current approaches to these design problems, decisions are based on a set of requirements, engineering judgment, historical background, and a few initial trade studies. Subsystems are designed concurrently and separately, and any optimization usually occurs at the subsystem level or below. More-refined exploration of the vast number of alternatives is costly and time consuming to perform, so while the design may meet the objectives, it is not necessarily the optimal solution. Optimality may involve a number of objective functions such as performance, throughput, or lifecycle cost among other criteria. When the technologies involved have little or no historical background, the information upon which to base decisions becomes even more limited.

As NASA prepares for a return to the Moon, a new set of technologies is being developed that may help create a more sustainable approach to space exploration. Termed in-situ resource utilization (ISRU), this concept involves using any resources available in the lunar or space environment that would help reduce the quantity of supplies that must be launched from Earth[1]. Oxygen is a consumable resource used to supply crew air and oxidizer in rocket propellant that can be extracted from the lunar regolith. Systems to perform lunar oxygen extraction are being studied and built by NASA, including the development of a system modeling tool that captures the many architecture alternatives in the system selection and design[2,3]. Several chemical processes can be used to produce oxygen from the metal oxides and glasses present in lunar soil, but because there is no historical data to draw from in the design of these systems, a detailed set of engineering models has been constructed to help assess the system-level trades of some of these processes.

In this paper, we present an approach for system modeling and optimization focused on selecting subsystems and subsystem technologies that result in an optimal system. We apply this to the NASA ISRU System Model. This consists of a system modeling framework that is structured hierarchically, capturing the breakdown of a system into major subsystems, minor subsystems, and leading down to the component model level. The framework is flexible to allow for simple incorporation of new or updated subsystem alternatives, and is automatically reconfigured between system options. The set of choices that comprise a system configuration are treated as discrete design variables, and the trade space is explored using a genetic algorithm. Lunar oxygen production is a good application for this architecture selection approach, because the oxygen production system can be functionally broken down into subsystems and components, a plant design involves a combination of existing and new technologies in a new environment (no historical background), and a set of models for the various subsystems already exist to populate the system model framework[2,4,5].

Previous work in applying optimization to system architecture selection and design includes a state-vector approach to model the effect of different component technologies in diesel exhaust after-treatment systems[6]. However, in this approach, only three architectures were considered. Each architecture was individually optimized for performance according to system sizing characteristics, and reconfiguration and comparison between different architectures was done manually. Most studies that have assessed technology choices in a system-level optimization use trade studies to enumerate the choices. Wilcox and Wakayama demonstrated simultaneous optimization of a family of aircraft wherein any of four structural components of the aircraft could be specified to be common among all aircraft in the family[7]. In this case, discrete levels of commonality (which parallel architecture technology selections), were explored via trade study. In another problem, technology alternatives were considered as part of the optimization problem formulation[8]. This study developed a satellite sizing tool that included choices for types of

communications hardware, solar cells, batteries, and launch vehicles as design variables in a genetic algorithm search. The analysis models were primarily based on first-order approximations and correlation with historical data. We build upon this approach in three ways:

1. consideration of incompatible technology combinations,
2. incorporating the hierarchical nature of technology choices: selection of an electrical propulsion system for a satellite would then involve technologies specific only to that selection, and
3. including in the problem formulation the continuous variables that contribute to the sizing and performance of the system.

The problem we seek to address is providing a method to explore the architecture trade space of systems that do not have a historical background upon which to base early-stage design decisions.

A genetic algorithm was chosen for this large-scale discrete optimization problem because it provides a global search method that does not require gradients for convergence and can handle discrete variables. Genetic algorithms are evolutionary search techniques that mimic Darwin's Theory of Natural Selection[9]. The design variables are coded as a string of binary numbers, creating a chromosome that is used to obtain one function evaluation. A population of chromosomes is generated and designs are competed against one another using their function evaluations as criteria. Concepts like inheritance, selection, crossover, and mutation are incorporated into the execution of the genetic algorithm, with mutation providing a probabilistic trigger that keeps the search from settling on local minima and helps the search cover the entire design space.

Genetic algorithms are computationally costly, so they are usually only used when faster, gradient-based methods cannot be used., either because of a highly nonlinear or discontinuous design space or the presence of discrete and categorical variables. The ISRU System Model is an evolving tool that is updated to include new technology alternatives as they are developed. As it grows, the number of system alternatives will become too large to fully enumerate and a method such as a genetic algorithm will become less costly than full enumeration. As present, the available ISRU system alternatives are few enough to enumerate, providing a good way to test and compare the performance of the genetic algorithm search tool to the currently-employed ISRU optimization methods. The future large design space in combination with having a mix of continuous and categorical, discrete design variables makes a genetic algorithm well suited for this problem.

In the following sections, we will discuss the modeling framework employed in the ISRU System Model design, the problem formulation for the genetic algorithm, and comparison of preliminary results from the GA to the current optimization scheme of the ISRU System Model.

II. System Model Framework

A. ISRU System Model

1. System Model Structure

The key to the system modeling framework is to follow a functional decomposition of the system. Breaking a system down in this manner can be somewhat intuitive, but it allows the modeler to define the stable interfaces and basic functions that must always be present to comprise the particular type of system being analyzed, which is in this case a lunar oxygen production plant. A lunar oxygen production plant can be broken down into three major processes: oxygen production, power supply, and liquefaction with storage[2]. These, in turn, can be accomplished in several different ways: oxygen production has four main technologies that are under development, two of which have been modeled in detail thus far with the ISRU System Model. Decomposing the system into its constituent elements reveals a pattern in the levels of the hierarchy (Fig. 1). Each alternating level consists of either subsystems whose functions must be *combined together* to create their parent function ("AND" levels), or a set of choices from which *only one* is needed to achieve its parent function ("OR" levels- which act as the Boolean exclusive, xOR).

Each block in figure 1 can represent a separate model or stand-alone analysis. The system model is structured with constant interfaces defined for each block such that if two different designs of a hydrogen reduction reactor exist, they can both be included at the "OR" level for hydrogen reduction reactors. The internal analysis of the reactor designs may be very different as long as they each provide the same basic required inputs and outputs that define the reactor function to the system model. The ISRU System Model models all of the subsystems and technology alternatives in this fashion.

In-situ oxygen production is a batch process, and each of the architectures requires a reactor into which regolith is transferred, reacted, and then dumped. Some architectures have reactors that create water as the reaction product.

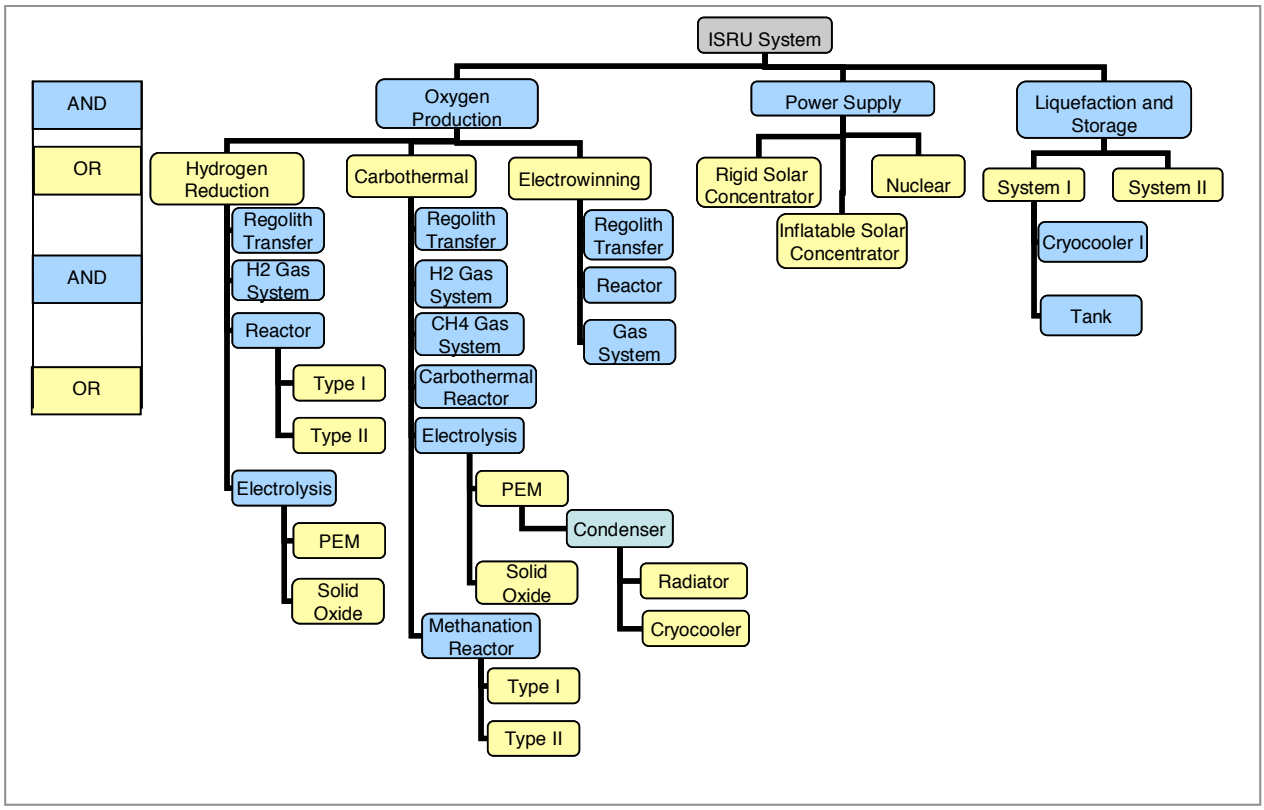

Figure 1. Example functional breakdown of ISRU system- not all discrete choices are represented.

These then include a water electrolysis subsystem to produce oxygen. Some architectures require a gas recycling system that re-circulates the reactants of the oxygen extraction processes.

To construct an architecture scenario, one option is selected at each "OR" level, and the inputs and outputs of the models are linked together to form an end-to-end system model. A constant set of system-level, global variables is specified that apply to all system architecture alternatives. These global variables define the lunar location of the plant, the number of reactors that can operate concurrently, and the continuous system sizing parameters that describe reactor size and throughput. With these definitions and a set of discrete variables that capture each "OR" level decision that can be made in the ISRU model, a complete system architecture is characterized. These choices include oxygen production methods, reactor types, electrolysis types, and condenser types.

The system model is integrated using Phoenix Integration's modeling software called "ModelCenter," with each technology choice represented by Excel-based models provided by the NASA ISRU team. A linking tool enables automatic reconfiguration of the system model between architecture scenarios based on the input set of discrete design variables.

2. *Current Optimization Strategies*

The current ISRU System Model employs a split-level approach to system optimization. On an "inner level" for a particular architecture selection a sequential quadratic programming (SQP) algorithm that uses finite difference approximations for gradient calculations is applied using the continuous variables that determine the sizing and performance of a system as design variables[3]. The objective function is to minimize the ratio of system mass to produced oxygen, a measure of mass efficiency. The function evaluation entails linking together and executing numerous engineering models resulting in a highly non-linear design space. Due to this, the SQP must be run from multiple starting points for each architecture scenario to avoid local minima, and even then a feasible point sometimes is not found. To address this, the current tool switches to a simulated annealing (SA) algorithm after three unsuccessful attempts with the SQP program. Simulated annealing is another heuristic search technique

modeled after the process of annealing metal[10]. Because SA algorithms are also computationally expensive, a termination criteria is set to end the search after more than 100 function evaluations have not found a new, best design point. Generally, this tool takes between 200 and 700 evaluations to terminate on ISRU System Model scenarios.

For the "outer level" technology selection exploration the discrete space of technology options is narrowed down to only the choices that have large effects on the system performance. This results in nine primary architectures. Each of these nine architectures was executed at varying oxygen production capacities with optimization performed only on the continuous variables. A detailed description of these optimization tools can be found in Ref. 3.

B. Genetic Algorithm Applied to ISRU System Model

1. Problem formulation

While the current ISRU System Model architecture has only nine primary architectures, as the models are developed further more technology options will be added to the tree shown in figure 1. The projected design space of the ISRU System Model with even modest upgrades to include lower level discrete variables, such as material types and operational modes, expands to over 10,000 possible combinations. Many of these types of choices are already represented in the component models of the system, but do not exhibit trade-offs in the objective of system mass (e.g. when the selection between two material types always yields the same choice to minimize mass). As the models are developed further and multiple objectives are assessed, these choices may come into play, compounding the discrete design space.

With this future in mind, a method was developed that will help search a large discrete design space at the point when full enumeration becomes infeasible. Preliminary tests have been performed on the ISRU System Model and compared to results from the optimization approaches mentioned in Section A. In this approach, a genetic algorithm is employed to handle both the discrete and continuous variables in the system model, combining the "inner" and "outer" levels discussed previously into one optimization search. The optimization problem is formulated as follows:

$$\text{Minimize fitness:} \quad f(\vec{x}) = \frac{Mass(\vec{x})}{O_{2\_produced}} + r_p * \sum_i \max\left[0, g_i(\vec{x})\right]^2 \tag{1}$$

Where:

Discrete variables:

x_1= Number of Reactors (3 choices)
x_2 = Power System Location (2 choices)
x_3 = Oxygen Production Process (2 choices, but only 1 used in preliminary tests)
x_4 = Electrolysis System (2 choices)
x_5 = Condenser Type (2 choices)
x_6 = Reactor Type (2 choices)

Continuous variables:

x_7= batches per day
x_8 = mass of regolith per batch ratio
x_9 = warm-up time ratio
x_{10} = reactor diameter ratio
x_{11} = H_2 flow rate ratio
x_{12} = CH_4 flow rate ratio

Mass = System Mass [kg]
$O_{2\_produced}$ = oxygen produced per year [kg/yr]
r_p = penalty multiplier
g_i = constraints

Subject to:

$$g_1 = 1 - \frac{O_2 produced}{O_2 desired} \leq 0 \tag{2}$$

$$g_2 = 1 - \frac{O_2 produced}{O_2 desired} \geq -.1 \tag{3}$$

$$g_3 = \frac{\left(Time\,\mathrm{Re}\,action + TimeWarmup + Time\,\mathrm{Re}\,golithTransfer\right)}{24/NumberBatchPerDay} - 1 \leq 0 \tag{4}$$

$$\frac{x_{LBi}}{x_{UB_i}} \leq \frac{x_i}{x_{UBi}} \leq 1 \tag{5}$$

The continuous design variables are normalized by their upper bounds to place them on the interval (0,1]. The objective and constraints are also scaled to be close to [-1,1]. In the ISRU System Model, x_7, representing batches of regolith processed per day, is treated as a continuous variable. Constraints g_1 and g_2 limit the produced oxygen quantity to be within a range of the desired production level, and g_3 can consist of up to three equations representing operational time constraints on the batch process: if the system uses one reactor, the time to transfer regolith, preheat, and react must be less than the time allotted per batch, as specified by x_7. If two reactors are selected, one can react while the other transfers regolith in and out and preheats the regolith to reaction temperatures. Three reactors each perform one of the operations at all times. With the discrete and continuous design space combined (the continuous variables being represented with discrete increments between their bounds), the overall number of possible combinations of variables becomes $4.02e^8$. For the initial tests of the GA optimization of the ISRU System Model, only one oxygen production process was explored due to development issues in the model. This reduces the design space to about $2.52e^8$ possibilities. The discrete variables comprise only 48 of these possibilities.

2. Genetic algorithm parameters and variable coding

The MATLAB-based genetic algorithm used for this problem entails a tournament selection, uniform crossover, a mutation probability of .01, and binary grey coding of the design variables. The constraints are handled via a quadratic exterior penalty method, as shown in Eqn. 1. Due to the hierarchical nature of the problem, decoding of the design variables is not a direct process. For example, if in Fig. 1 a hydrogen reduction system is selected for the oxygen production function as the value assigned to x_3, the resulting assignments of x_4 and, x_5 must be of the options for a *hydrogen reduction* reactor and electrolyzer. We refer to this as *conditional decoding*. In addition, because the condenser is a component used only by a PEM electrolysis choice, the value assigned to x_6 is only relevant when PEM Electrolysis is chosen. This approach to variable decoding mimics the concept of sex-limited inheritance in genetics where a particular gene is only expressed if another gene (indicating sex) is present[11]. Male-pattern baldness is an example of a gene that women can carry, but is only expressed in men. This decoding method is useful because it allows a small set of variables to describe all of the choices in the system without representing every branch in the tree with its own variables. It is also employed here to help handle compatibility constraints between technology choices.

One of the issues encountered in formulating a large-scale technology selection problem is that when including all possible technologies that can be used to construct a system, some choices may be entirely incompatible with others. This occurs in the ISRU System Model for a combination of a particular hydrogen reduction reactor type with a PEM electrolyzer- the two do not make physical sense to use together. To ensure that this combination is never encountered in the variable decoding process, the domain of the reactor variable is limited only to feasible choices, based on the decoded value of the electrolyzer choice. Thus the selection of one gene, the electrolyzer, effects what the possible values of the next genes, condenser type and reactor type, can be.

The discrete variables are represented with a binary resolution of 1, and the continuous variables are represented with between 5 and 8 bits, making the total chromosome length 45 bits. To generate preliminary results with minimal computation time, a population size of 100 was selected based on empirical guidelines presented in Ref. 12.

Stopping criteria was set to a maximum of 50 generations or when five consecutive generations have the same best function evaluation, but all runs hit the maximum generation number before reaching this "optimality" criterion. Runs were performed with the ISRU System Model for four different oxygen production levels. The purpose of the runs was to determine how well the GA performed at providing approximate, good design points when searching the combined continuous and discrete design space. Section III presents the results of these runs compared to the results of the optimization methods discussed in Section A.

III. Results

Figure 2 shows the convergence history of the genetic algorithm at three different oxygen production levels. The minimum fitness evaluation for each generation was plotted. The 2000 kg/yr case was run for 70 generations, but in each case the objective had leveled off significantly by the 50th generation. Figures 3-5 provide an example of the generation progression for the 5000 kg/yr case. By generation 50 in Fig. 5 most designs have leveled out at a fitness evaluation of about 0.13.

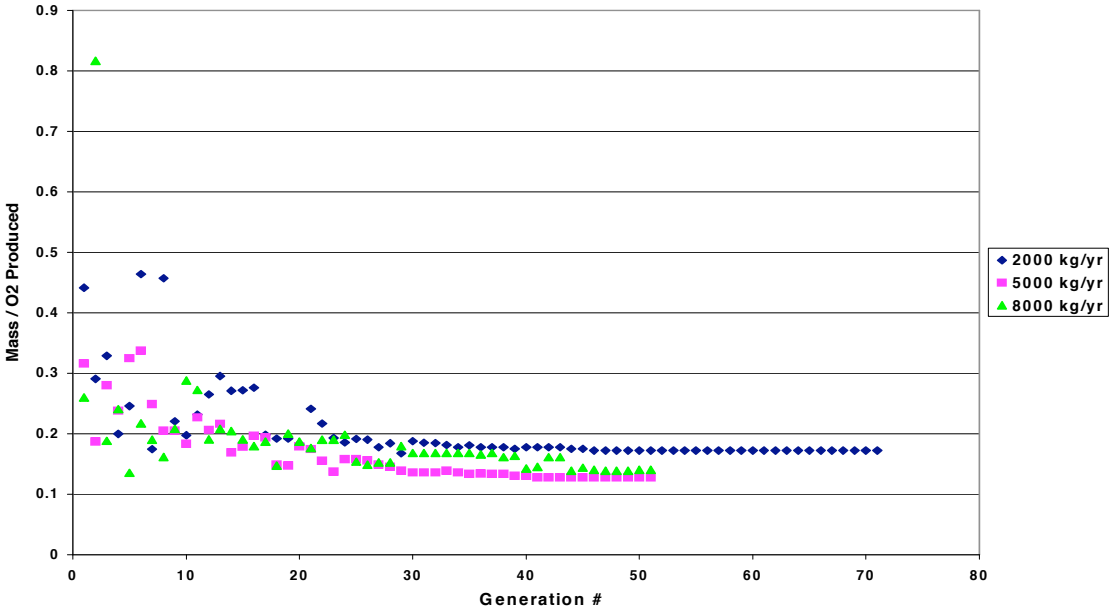

Figure 2. Convergence history for genetic algorithm optimization of ISRU System Model.

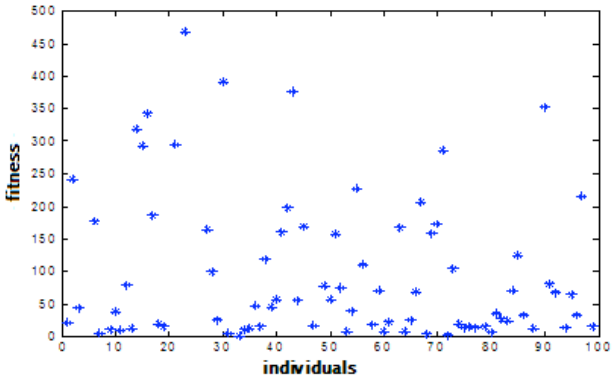

Figure 3. Generation 1 for GA optimization run, 5000 kg/yr

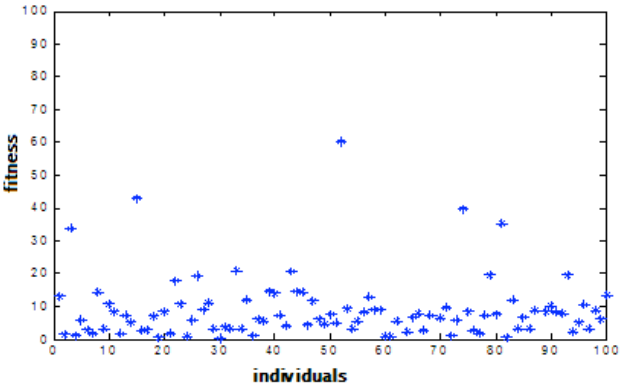

Figure 4. Generation 10 for GA optimization run, 5000 kg/yr

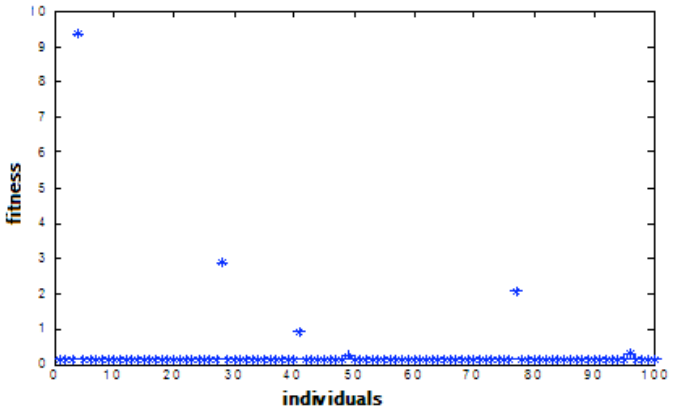

Figure 5. Generation 50 for GA optimization run, 5000 kg/yr

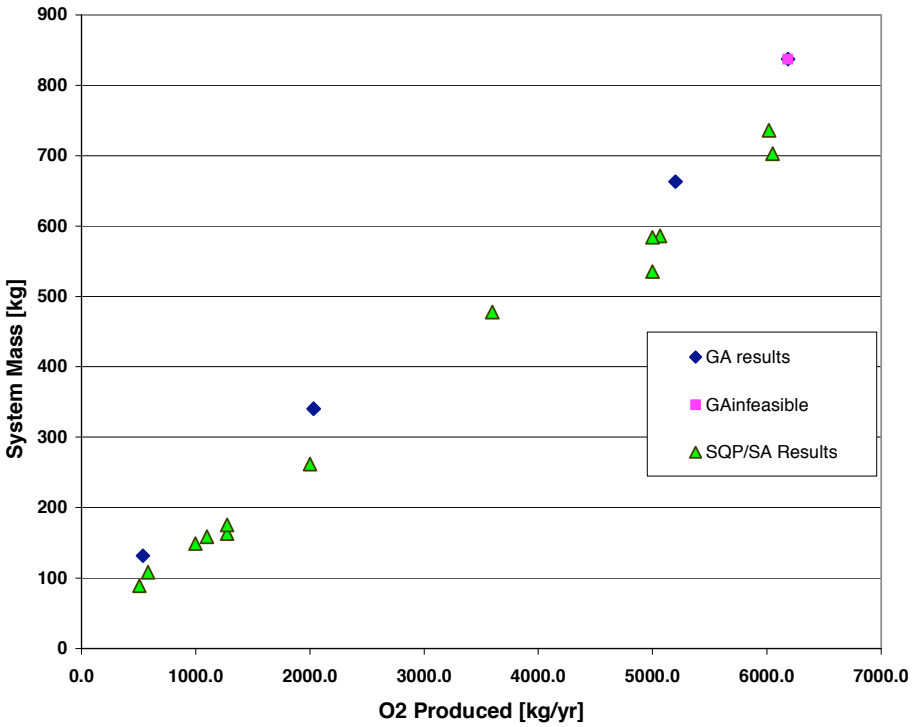

Figure 6. Comparison of GA results and SQP/SA results[3].

While many more tests need to be run for each O_2 production case, the initial results are promising. In 50 generations, each test case produced fitness results comparable to the SQP/SA methods currently used in the ISRU System Model. Figure 6 shows the GA results plotted with a re-produced set of non-dominated (Pareto) points from Ref. 3. In the GA test cases, one point (shown as a pink square) violated one of the time constraints. The 5000 kg/yr case had two active time constraints. The history of the discrete variable choices corresponding to the minimum fitness value of each generation proved to be interesting: the 5000 kg/yr and the 8000 kg/yr cases both settled on the same set of discrete variable choices by about the 25th generation. The 500 kg/yr and the 2000 kg/yr also both settled on a common set of discrete variables that was different from the 5000 kg/yr and 8000 kg/yr set. This would indicate that one particular architecture trades better for lower oxygen production levels, while a different architecture trades better for higher production levels. This result actually differs from preliminary results shown in Ref 3 in which two architectures each dominated the Pareto front across all production levels. Many more tests at different production levels with the GA and further model validation are required to fully affirm this result.

Computation time is one of the biggest concerns with using a genetic algorithm for this problem. However, comparing to the SQP/SA method, the GA in this application exhibits a comparable computation time. Because the SQP/SA method requires full enumeration of the discrete variables, for a single production level the SQP/SA optimizer must be executed 48 times. Each SQP/SA run can take between 100-700 function evaluations to converge, depending on the initial starting point and whether or not the simulated annealing algorithm is necessary. This amounts to between 4800 and 10,000 function evaluations per oxygen production case. Using a population size of 100 with 50 generations, the genetic algorithm requires 5100 function evaluations. Even increasing the population size to 120 and allowing 70 generations to run leads to only 8400 evaluations, which is still comparable to the SQP/SA approach. On the ISRU System Model, 5100 function evaluations takes about 36 hours to complete. Table 1 summarizes this comparison.

Table 1. Computation Expense Comparison

Algorithm	# total function evaluations (discrete and continuous search)
SQP / SA	4,800-10,000
GA	5,100 – 8,400

IV. Conclusions

A system modeling framework was presented that allows the subsystem technology alternatives inherent in system design to be included in an optimization scheme, enabling designers to search for an optimal system architecture. A genetic algorithm was successfully applied to a previously developed ISRU System Model to handle a hierarchical set of categorical, discrete design variables with compatibility constraints. It simultaneously optimized the continuous sizing parameters of the system as well as the choices that comprised the system architecture. In both performance and computation time, the initial tests of the genetic algorithm approach were comparable to previously tested optimization methods that involved full enumeration of the discrete space outside of a gradient-based or heuristic search of the continuous space.

Further work needs to be done to fully characterize the application of the genetic algorithm to the ISRU System Model, including testing population size, mutation rate, and checking for signs of dominance of one particular design too early. A very limited set of case studies is presented in this paper, and more runs need to be performed to build any conclusions on the model results. Future work will also include testing this method with a larger discrete design space as the ISRU System Model continues its development. At that point it may be necessary to test whether a combination of the discrete and continuous variables in one optimization routine is advantageous to a layered approach, with an optimization of the continuous variables performed for each iteration of the discrete trade space search. A more flexible integration of compatibility constraints also needs to be explored, along with different approaches to this type of constraint. One proposed direction of investigation in this area involves using a constraint satisfaction problem (CSP) solver to filter infeasible designs from the GA's population pool, and either repairing infeasible designs or randomly generating new ones to replenish the population.

Acknowledgements:

This work was financially supported by the National Aeronautics and Space Administration (NASA) through JPL Contract No.1327485 - MIT ISRU SEMI. Dr. Robert Easter served as the technical contract monitor. Technical assistance was provided by Mr. Kristopher Lee and Tom Simon at the NASA Johnson Space Center.

References

[1]Sanders, G., "ISRU: An Overview of NASA's Current Development Activities and Long-Term Goals", *38th Aerospace Sciences Meeting & Exhibit*, AIAA 2000-1062, Jan 2000.

[2] Steffen, C., Freeh, J., Linne, D., Faykus, E., Gallo, C., Green, R., "System Modeling of Lunar Oxygen Production: Mass and Power Requirements", *Proceedings of Space Nuclear Conference 2007*, Paper 2049, Boston, June 2007.

[3]Chepko, A., de Weck, O., Linne, D., Santiago-Maldonado, E., Crossley, W., "Architecture Modeling of In-Situ Oxygen Production and its Impacts on Lunar Campaigns", *AIAA Space 2008 Conference & Exposition,* San Diego, California, 9-11 Sep. 2008.

[4]Hegde, U., Balasubramaniam, R., Gokoglu, S., "Analysis of Thermal and Reaction Times for Hydrogen Reduction of Lunar Regolith", *Space Technology and Applications International Forum—STAIF 2008,* American Institute of Physics, pp 195-202, 2008.

[5]Balasubramaniam, R., Hegde, U., Gokoglu, S., "Carbothermal Processing of Lunar Regolith Using Methane", *Space Technology and Applications International Forum—STAIF 2008,* American Institute of Physics, pp 157-161, 2008

[6]Graff C., de Weck O., "A Modular State-Vector Based Modeling Architecture for Diesel Exhaust System Design, Analysis and Optimization", AIAA-2006-7068, 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, Virginia, 6 - 8 Sep 2006.

[7] Wilcox, K., Wakayama, S., "Simultaneous Optimization of a Multiple-Aircraft Family", Journal of Aircraft, AIAA, Vol 40, No 4, July 2003.

[8]Hassan, R., Crossley, W., "Multi-Objective Optimization of Communication Satellites with Two-Branch Tournament Genetic Algorithm", *Journal of Spacecraft and Rockets*, AIAA, Vol 40 No2, March 2003.

[9]Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs 2nd Ed.*, Springer-Verlag, New York, 1994.

[10] S. Kirkpatrick, C. G. Jr., and M. Vecchi, "Optimization by Simulated Annealing", *Science,* Vol. 220, No 4598, pp. 671–680, 1983.

[11]Crossley, W., "Using Genetic Algorithms as an Automated Methodology for Conceptual Design of Rotorcraft", PhD Dissertation, Arizona State University, Tempe, AZ, 1995, pp. 35-37.

[12] Williams, E. A., and Crossley, W. A., "Empirically-Derived Population Size and Mutation Rate Guidelines for a Genetic Algorithm with Uniform Crossover," *Soft Computing in Engineering Design and Manufacturing*, P. K. Chawdhry, R. Roy and R. K. Pant (editors),Springer-Verlag, 1998, pp. 163-172.