Original source of publication:

Adam Martin, Vössner Siegfried, Schlange Lutz E.; Evolutionary Creation and Adaptation of Management Rules – A Genetic Approach to Strategic Management; in: Schwaninger Markus (Ed.) Intelligente Organisationen, Duncker & Humbold, Berlin 1999

Evolutionary Creation and Adaptation of Management Rules

A Genetic Approach to Strategic Management

Martin Adam, Siegfried Vössner, Lutz E. Schlange

Abstract

This paper shows a new approach by using evolutionary methods - namely Genetic Programming (GP) to generate and adapt strategies in order to master complex situations in social systems (evolution/induction of rules). To show the principles of the presented technique, we apply it to the well known dilemma in strategic management: "short-run restructuring versus long-term organizational development".

Introduction

„Strategic management" is the generation of strategies to guide human behavior in order to reach selected goals[1]. Strategies are defined as a set of rules. We are aiming at the generation and adaptation of these management rules. The central question is to figure out in advance *which rules will fit best*. Which rules will produce the system-behavior we want?[2]

The characteristic activities of generating strategies have been described as developing the functions of „policy", „intelligence" and „control" and fulfilling them simultaneously.[3] To create strategies we have to be aware of the overall values within the system, called „policy", which are restricting our strategies. Short term operation control influences our strategic possibilities as well. Furthermore we also have to look beyond the enterprise and analyze future opportunities[4].

The management situation may be characterized on one side as very complex with all the different, uncertain and sometimes unpredictable factors influencing the system from outside and inside - and on the other side by a lack of requisite variety available.

---

[1] *Malik.*
[2] *Hayek, Vanberg, Brennan/ Buchanan.*
[3] *Espejo/ Schuman/ Schwaninger/ Bilello.*
[4] *Beer.*

Evolutionary Creation and Adaptation of Management Rules

Existing Approaches

Because of the complexity of managerial situations, step by step procedures are considered to cope best with these situations. Evolutionary problem solving is one of those procedures. Computer supported evolutionary mechanisms applied in managerial science are often mentioned[5]. There is also a common agreement that an instrument is needed to prove strategies before implementing them in real situations[6]. Most of the current applications of computer supported evolutionary mechanisms in management applications are in optimizing production plans, logistics, layouts, construction or finance. Very little has been done in strategic management and rule generation[7].

Our approach

Our solution to cope with the problems described above is to increase the variety of strategic management. This is mainly done by improving the „intelligence function“ of rule generation, known as the “power of appropriate selection”[8]. We apply an instrument that *automatically* generates and adapts viable management strategies. We create an artificial competition between different rules guiding the systems behavior. Through step by step adaptation the mechanism selects appropriate rules and thereby improves their fitness. For this we use an evolutionary procedure called Genetic Programming (GP). We *induce strategies* given a social system within a changing environment rather than performing a conventional parameter optimization: The systemic tool we present here improves the system’s intelligence by giving individuals (managers) the ability to choose appropriate systemic decisions from a pool of generated strategies. It is not designed as an automatism to generate, select and implement strategies.

Management may be modeled as the interaction of multi-control loops. For reasons of simplicity we only show one single control loop (figure 1): A heuristic model of the social system is put into a cybernetic control loop where a GP kernel acts as a support of the control unit. One input (decision variable) can be influenced directly - the other one represents disturbances induced by changes in the internal and external environment. The outputs are transmitted via a „feedback”-loop to the „control unit”, where the „fitness function“ assigns a „fitness“ value to them. This value indicates how well the system output

---

[5] *Malik, Gomez.*
[6] *Malik, Vanberg, Popper.*
[7] *Nissen.*
[8] *Ashby.*

agrees with the given „goals" and is sent to the "strategy generator", where new rules will be derived from the best rules (fitness value).

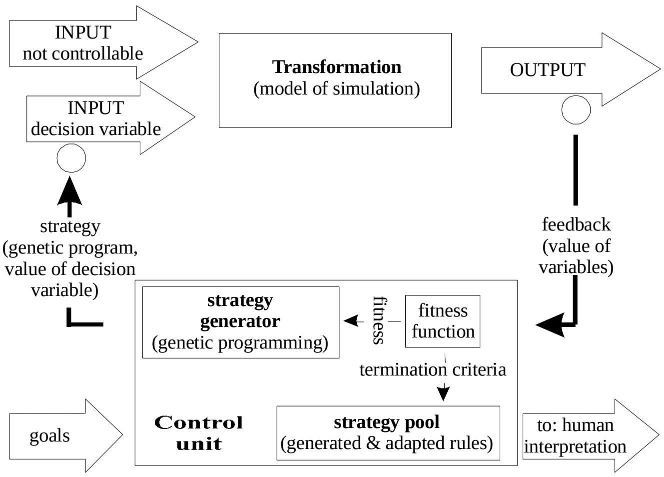

Figure 1: Cybernetic Control Loop

These new rules change the values of the decision variables and influence the system's output. Again the output is sent to control unit and a new iteration step begins. When the automatic adaptation fulfills the termination criterion the rule generation will stop and the new rule will be send to the „strategy deposit". There it is available for use and interpretation by the human strategist.

Genetic Programming

Unlike classical Operations Research techniques, Evolutionary Algorithms like Evolutionary Strategies[9] [10] (ES), Genetic Algorithms[11] [12] (GAs) and Genetic Programming[13] (GP) use the mechanics of natural selection ("survival of the fittest")[14] and genetics that were originally inspired by biological structures and their evolution.

---

[9] *Rechenberg*.
[10] *Schwefel*.
[11] *Holland*.
[12] *Goldberg*.
[13] *Koza*.
[14] *Darwin*.

Evolutionary Algorithms process many candidate solutions (a population) at one time, in parallel. The best solutions survive a competition based on their performance in a simulation run, and go on in the selection process. Out of these solutions, new ones are created by recombination. It is basically a competition between different rule systems - favoring the better performing ones. The artificial evolutionary process itself is driven by an objective function which is provided externally. There are two principal ways to decide when to terminate the iteration loop (figure 2), either if a certain criterion is fulfilled or if an optimal value is not known (which is true in our case), the loop is terminated if there is no improvement for a certain number of iterations.

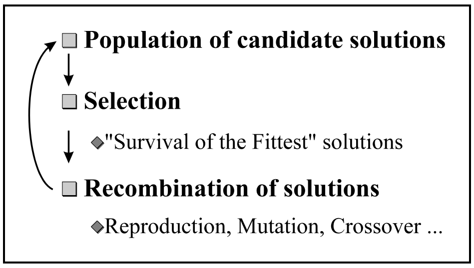

Figure 2: Scheme

In Genetic Programming, a candidate solution would be a single computer program - in our context one possible set of management rules, called "policy" from here on. Programs are hierarchically structured and can change dynamically during the evolutionary process. The set of possible structures is a combination of functions that can be composed recursively from the set of N functions $F = \{f_1, f_2, \ldots, f_N\}$ and the set of M terminals $T = \{t_1, t_2, \ldots, t_M\}$. Each particular function f_i out of F can take a certain number of arguments.

The function set F could include arithmetic operators, mathematical functions, conditional operators (such as *if then else*), iterative operators (such as *do while*), recursive operators or any other user defined function. Typical terminals are state variables of system, numerical constants or commands which trigger an action. In designing a proper function and terminal set it is important to achieve closure of both sets, which means that each function f_i should be able to accept any value or data type returned by a function or any terminal t_i. One common way of representing programs in GP is as a parse tree[13] as shown in figure 3.

The recombination of programs takes place at the level of their tree representation. Operators like Crossover, Mutation, Permutation, Editing and

Encapsulation chose branches of trees from "good performing" programs and combine them into a new program tree.

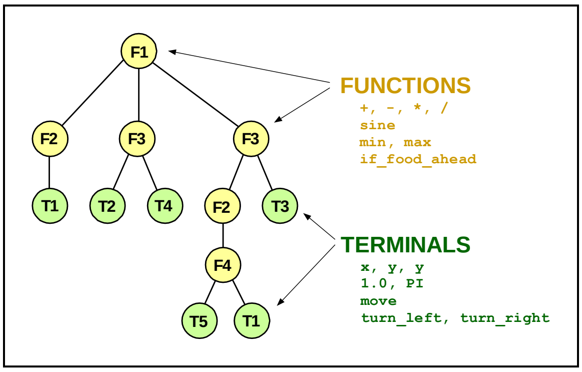

Figure 3: Structure of a Genetic Program

Genetic Programming has some major advantages over other techniques. First GP produces rules, which can be read, interpreted and changed by human beings. Furthermore the ingredients can be either simple operators and functions or complex building blocks like rules derived from human experience. The algorithm itself is extremely robust in searching for solutions and can handle high dimensional and complex search spaces. Since GP does not assume any particular structure of the search space a priori it can handle stochastic or discontinuous systems very well. The GP implementation is independent from the problem. Therefore it is possible to separate the GP from the objective function (problem) loss of generality, which means it does not have to be changed for different problems.

Example

The conceptual stage of our work has led us to opt for an example which in its very basic features is characteristic of managerial situations. Therefore we will describe a well-known managerial dilemma, which has been discussed as the strategic issue of "short-run restructuring versus long-term organizational development". The situation is represented as an "opaque box"[15], the exact details of which we only know to a certain extent. In fact, our knowledge is restricted to the very basic structures consisting of certain variables and relations between them (figure 4). Therefore, our modeling approach is

[15] *Beer*.

grounded in the philosophy of "soft systems"[16]. Its stochastic character permits us to make uncertainty explicit with respect to the system's components as well as to its environment.

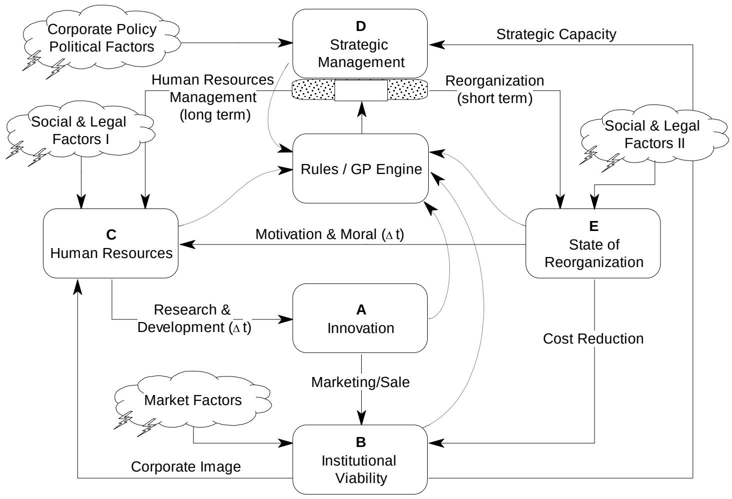

Figure 4: Example - System Model

Faced with environmental pressures for change management has to make choices to lay the ground for the company's long term survival. Restructuring, cutting down costs or, on the other, of continuing to invest in human resources. Strategic choice occurs in a variable called "Strategic Management" (D). It is confronted with the need for change, which may be due to internal, corporate policy decisions or to external, political factors (shown as clouds in figure 4). Regarding its overall goal of this company's future Institutional Viability (B), the strategic management team, on one hand, can decide to reorganize Corporate Structure (E). However, this has an undesired side-effect on Human Resources (C) because under certain circumstances motivation and moral in the company may decrease. This systemic structure resembles the "quick-fix" systems archetype[17]: a dangerous time delay (Δ t) indicates that the situation may tend to deteriorate unnoticeable. On the other hand, the team may choose to intensify its management of Human Resources (C) thus building competencies for the long term. Social and legal factors such as qualification of potential employees or labor market regulation may externally influence the

[16] *Checkland*.
[17] *Senge*.

availability of human resources. Investing in human resources is done because it will lay the ground for Innovation (A), a prerequisite for differentiation in competitive markets (market factors). However, for innovation to actually take place, research and development must be successfully carried out over a rather long period of time (Δ t). Moreover, to become a long term support for corporate viability, results of innovation have to be successfully marketed. Any of both choices will have either positive or negative effects on the corporate image, which may potentially attract, or as well detract, highly qualified employees from joining the company. A second feedback loop delineates that any improvement of institutional viability will create more maneuvering space for Management, thus enhancing its strategic capacity.

GP-Kernel and Simulation Package

We used the a standard GP kernel (GPQUICK library[18]) that calls an external simulation package to evaluate the performance of the different policies as an objective function (SysSim[19]). SysSim can model arbitrary complex networks. In some nodes there is a variable supply which can be provided externally. The arc flow can have a delay of up to five simulation cycles and is an arbitrary function of the value of its tail. SysSim can treat node values in two different ways. One is that all arcs with share the resources of the node they start from (shared resources) at a given percentage. The other is that all arcs have access to the entire resources available at the node (common resources).

The simulation starts with initial values for the amount of D which should be saved for future rounds (Savings) of 20% and 50% for the amount of D that should be invested in Human Resources. The following tables (table 1 and 2) show the functions and terminals we allowed the GP-kernel to choose from:

Table 1: Set of functions.

FUNCTION	Arguments	Description
IfXisHigh	**(y, n)**	executes branch **y** if value of node X ≥ 20 **n** otherwise.
IfXisMedium	**(y, n)**	executes branch **y** if value of node X > 10 and X < 20

[18] *Singleton.*
[19] *Vössner*.

		n otherwise.
IfXisLow	**(y, n)**	executes branch **y** if value of node X < 10 **n** otherwise. Node X={A, B, C, D, E}
Prog2	**(t1, t2)**	executes branch **t1** first, then **t2**.
Prog3	**(t1, t2, t3)**	executes branch **t1** first, then **t2** and at least **t2**.

Table 2: Set of terminals.

TERMINAL	Description
IncSavings	increments the amount of node D which should be saved for future rounds by 10%.
DecSavings	decrements the amount of node D which should be saved for future rounds by 10%.
IncHumanRes	increments the remaining amount of node D (without savings) which should be sent to Node C by 10%.
DecHumanRes	decrements the remaining amount of node D (without savings) which should be sent to Node C by 10%.
ContinueSim	continues the simulation run for the next period with the current values for the decision variables (Savings and HumanRes).

A genetic program built out of elements from table 2 and table 3 could for example look like the following: **`IfCisLow(Prog2(IncHumanRes, IncHumanRes), IncSavings)`**. This program would be read as: "*If the value of node C is less than 10 then increment the percentage of D that is sent to Node C (HumanRes) by 20% (2x10%) - otherwise increment the percentage of Node D that is saved for future expenses by 10% !* ".

Results, Conclusion and Future Research

We tested the rule generator on the given example modeled in SysSim[20] (2000 individuals, default settings for all other parameters[21]). The issue was to „survive" 100 periods and thereby to increase the viability (B) of the company as much as possible. The simulation ends sooner if the viability becomes zero.

The best individual of the first generation of the GP run gives: **`(IfDisHigh IncSavings IncSavings)`** This rule causes the simulation to terminate at period 12 because of zero viability. After 12239 generations, the GP finds the following strategy which yields a viability of 451 after 100 periods:

[20] *Vössner*.
[21] *Singleton*.

`(Prog2 (IfBisHigh IncSavings DecHumanRes) (IfDisHigh (IfAisMedium IncHumanRes (IfAisMedium (Prog2 DecHumanRes DecHumanRes) DecSavings)) DecSavings)).`With this experiment we verified that the algorithm is capable of doing what it was designed for.

To answer the question how good this strategy is we elaborated an experiment where we confronting a couple of people with a stand-alone version of the problem modeled in SysSim[22] - without the GP-Kernel. Under the standards of the experiment it turned out that many people had difficulties to „survive“ the 100 periods. The details of the experiment will be published soon.

A pure GP approach can produce highly efficient management rules of moderate size. These rules have in general to be reviewed, edited, extended or simplified by a human expert.

In future research we will perform more detailed experiments comparing human expertise and computer generated rules as well as a combination of both, where human experts can use the computer generated rules. We will also provide a more powerful set of functions *F* and allow the algorithm to choose therefrom.

References

Asbhy R.W.: An Introduction to Cybernetics: London, Beccles, Colchester 1975.

Beer St.: The Heart of Enterprise: Chichester, New York, Brisbane, Toronto, Singapore 1994.

Biethahn J./ *Nissen* V.: Evolutionary algorithms in management applications: Berlin, Heidelberg, New York, Barcelona, Budapest, Hong Kong, London, Milan, Paris, Santa Clara, Singapore, Tokyo 1995.

Brennan G./ *Buchanan* J.M.: The reason of rules, Constitutional political economy: Cambridge, London, New York, New Rochelle, Melbourne, Sydney 1985.

Checkland P.: Systems Theory, Systems Practice, John Wiley, 1981.

Darwin C.: On the Origin of Species by Means of Natural Selection, J. Murray, London 1859.

Espejo R./ *Schuhmann* W./ *Schwaninger* M./ *Bilello* U.:Organizational Transformation and Learning, A Cybernetic Approach to Management: Chichester, New York, Brisbane, Toronto, Singapore 1996.

Goldberg D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, Massachusetts, Addison-Wesley, 1989.

---

[22] *Vössner*.

Gomez P.: Modelle und Methoden des systemorientierten Managements, Eine Einführung: Bern, Stuttgart 1981.

Hayek F. A. v.: Recht, Gesetzgebung und Freiheit, Bd.1. Regeln und Ordnung: Eine neue Darstellung der liberalen Prinzipien der Gerechtigkeit und der politischen Ökonomie: München 1980.

Holland J.: Adaptation in Natural and Artificial Systems. Ann Arbor, Michigan, University of Michigan Press, 1975.

Koza, J. R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, Massachusetts, MIT Press, 1992.

Malik F.: Strategie des Managements komplexer Systeme, Ein Beitrag zur Management-Kybernetik evolutionärer Systeme: Bern, Stuttgart, Wien 1977, 4. Auflage 1992.

Nissen V.: An Overview of Evolutionary Algorithms in Management Applications: in: Biethahn, Nissen 44 - 100.

Popper K.: Alles Leben ist Problemlösen: München 1994, 4. Auflage 1995.

Rechenberg I.: Evolutionsstrategie. Stuttgart, Friedrich Fromm Verlag, 1972.

Schwefel H. P.: Evolutionsstrategie und numerische Optimierung. Berlin, Technische Universität Berlin, 1975.

Senge, P.: The Fifth Discipline - The Art and Practice of the Learning Organisation, Doubleday, New York, 1990.

Singleton A.: GPQUICK. Peterborough, NH, USA, 1994.

Vanberg V.J.: Rules and Choice in Economics: London, New York 1994.

Vössner S.: SysSim (System Simulation Tool). Stanford, CA, USA, Genetic Optimization Lab, 1997.