Heuristic Networks for Aerospace Systems Control: Totally Concurrent Motions

D R A F T

Boris Stilman
Department of Computer Science & Engineering
University of Colorado at Denver, Campus Box 109, Denver, CO 80217-3364.
Email: bstilman@cse.cudenver.edu

Abstract This paper* reports new results in the development of Linguistic Geometry for multiagent systems. The Linguistic Geometry allows for the decomposition of a complex system into a dynamic hierarchy of subsystems, solving intractable search problems by reducing the search dramatically. In this paper we consider briefly the Linguistic Geometry tools and their application to the new multiagent system represented 2D optimization problem for autonomous robotic vehicles in aerospace environment. This example represents a totally concurrent system where all the vehicles move simultaneously, which is different from our earlier examples where only vehicles of each of the opposing sides could move simultaneously.

Keywords: Artificial Intelligence, Multiagent Systems, Linguistic Geometry, Heuristic Search, Network Languages, Concurrent Motions.

1 INTRODUCTION

Problems of long and short-range mission planning, especially for autonomous navigation, scheduling, aerospace robot control, long-range satellite service, aerospace combat operations control, etc. can be formally represented as reasoning about complex large-scale control systems. The field of efficient aerospace control systems needs new technology from the science of artificial intelligence (Rodin, 1988; Lirov, Rodin et al., 1988).

The classic approach based on the theory of Differential Games alone is insufficient, especially in case of dynamic, multiagent models (Garcia-Ortiz et al., 1993). Following (Rodin, 1988; Shinar, 1990) discrete-event modeling of complex control systems can be implemented as a purely interrogative simulation. These techniques can be based on generating geometrically meaningful states rather than time increments with due respect to the timeliness of actions. By discretizing time, a finite game tree can be obtained. The nodes of the tree represent the states of the game, where the players can select their controls for a given period of time. It is also possible that players do not make their decisions simultaneously and in this case, the respective moves of the two sides can be easily distinguished. Thus, the branches of the tree are the moves in the game space. The pruning of such tree is the basic task of heuristic search techniques. Interrogative approach to control problems offers much faster execution and clearer simulator definition (Lirov et al., 1988). For this kind of approach a series of hierarchical dynamic multiagent goal-oriented systems should be developed and investigated.

There are many such problems where human expert skills in reasoning about complex goal-oriented systems are incomparably higher than the level of modern computing systems. Unfortunately, problems of tactics planning and automatic control of autonomous agents such as aerospace vehicles, space stations and robots with cooperative and opposing interests are of the type where human problem-solving skills can not be directly applied. Moreover, there are no highly-skilled human experts in these fields ready to substitute for robots (on a virtual model) or transfer their knowledge to them. There is no grand-master in robot control, although, of course, the knowledge of existing experts in this field should not be neglected – it is even more valuable.

In this respect it is very important to study human expert reasoning about similar complex systems in the areas where the results are successful, in order to discover the keys to success, and then apply and adopt these keys to the new, as yet, unsolved problems, and first and foremost to the aerospace critical complex systems.

2 BACKGROUND

In the beginning of 80's Botvinnik, Stilman, and others developed one of the most interesting and powerful heuristic hierarchical models. It was successfully applied to scheduling, planning, control, and computer chess. The hierarchical networks were introduced in (Botvinnik, 1984; Stilman, 1977) in the form of ideas, plausible discussions, and program implementations. We consider this model as an ideal case for transferring the developed search heuristics to other domains employing formal mathematical tools.

An application of the developed model to a chess domain was implemented in full as program PIONEER (Botvinnik, 1984). Similar heuristic model was implemented for power equipment maintenance in a number of computer programs being used for maintenance scheduling all over the USSR (Botvinnik et al., 1983; Stilman, 1985, 1993a).

In the 1960's, a formal syntactic approach to the investigation of properties of natural language resulted in the fast development of a theory of formal languages by Chomsky (1963), Ginsburg (1966), and others. This

* Supported in part by the U.S. Air Force Office of Scientific Research through Phillips Lab, Kirtland AFB, NM, USA.

development provided an interesting opportunity for dissemination of this approach to different areas. In particular, there came an idea of analogous linguistic representation of images. This idea was successfully developed into syntactic methods of pattern recognition by Fu (1982), Narasimhan (1966), and Pavlidis (1977), and picture description languages by Shaw (1969), Feder (1971), and Rosenfeld (1979).

Searching for adequate mathematical tools formalizing human heuristics of dynamic hierarchies, we have transformed the idea of linguistic representation of complex real-world and artificial images into the idea of similar representation of complex hierarchical systems (Stilman, 1985). However, the appropriate languages possess more sophisticated attributes than languages usually used for pattern description. The origin of such languages can be traced back to the research on programmed attribute grammars by Knuth (1968), Rozenkrantz (1969).

A mathematical environment (a "glue") for the formal implementation of this approach was developed following the theories of formal problem solving and planning by Nilsson (1980), Fikes and Nilsson (1971), Sacerdoti (1975), McCarthy (1980), McCarthy and Hayes (1969), and others based on first order predicate calculus.

3 INTRODUCTION TO LINGUISTIC GEOMETRY

A formal theory, the so-called *Linguistic Geometry* (Stilman, 1992-95), includes the syntactic tools for *knowledge representation* and *reasoning* about large-scale hierarchical complex systems. It relies on the formalization of *search heuristics*, which allow one to decompose complex system into a hierarchy of images (subsystems), and thus solve intractable problems by reducing the search. These *hierarchical images* in the form of networks of paths were extracted from the expert vision of the problem.

The hierarchy of subsystems is represented as a *hierarchy of formal attribute languages* where each "sentence" (a group of "words" or symbols) of the lower level language corresponds to the "word" of the higher level one. Following a linguistic approach each subsystem could be represented as a string of symbols with parameters: $a(x_1)a(x_2)...a(x_n)$, where the values of the parameters incorporate the semantics of the problem domain or lower-level subsystems. The lowest-level language of the hierarchy of languages, the Language of Trajectories (Stilman, 1993a, 1993c), serves as a building block to create the upper-level languages, the Languages of Networks (Stilman, 1993b, 1993c, 1994a, 1994b).

The Language of Trajectories describes the set of various paths between different points of the complex control system. An element might follow a path to achieve the goal "connected with the ending point of this path." The Language of Networks is a formalization of a set of networks of certain paths unified by the mutual goal. For example, in the chess model such network represents planning for a local fight, in the robot control model an analogous network of planning paths represents a draft short-range plan for approaching local goal in hazardous environment, i.e., getting over mobile and immobile obstacles. In the scheduling problem it corresponds to the maintenance schedule of a certain power unit including the schedule for the provision of resources required.

Network languages describe the "statics", i.e., the states of the System. To describe the "dynamics" of the System, i.e., the motions from one state to another, we have to regenerate the entire hierarchy of languages. Different variations of these motions, a search tree, are represented as a string of the highest level formal language, the Language of Translations (Stilman, 1994b, 1994c).

4 CLASS OF PROBLEMS

A **Complex System** is the following eight-tuple:

$< X, P, R_p, \{ON\}, v, S_i, S_t, TR>$, where

$X=\{x_i\}$ is a finite set of *points*;

$P=\{p_i\}$ is a finite set of *elements*; P is a union of two non-intersecting subsets P_1 and P_2;

$R_p(x, y)$ is a set of binary relations of *reachability* in X (x and y are from X, p from P);

ON(p)=x, where ON is a partial function of *placement* from P into X;

v is a function on P with positive integer values describing the *values* of elements.

The Complex System searches the state space, which should have initial and target states;

S_i and S_t are the descriptions of the *initial* and *target* states in the language of the first order predicate calculus, which matches with each relation a certain Well-Formed Formula (WFF). Thus, each state from S_i or S_t is described by a certain set of WFF of the form $\{ON(p_j) = x_k\}$;

TR is a set of operators, TRANSITION(p, x, y), of transitions of the System from one state to another one. These operators describe the transition in terms of two lists of WFF (to be removed from and added to the description of the state), and of WFF of applicability of the transition. Here,

Remove list: ON(p)=x, ON(q)=y;

Add list: ON(p)=y;

Applicability list: $(ON(p)=x)\wedge R_p(x,y)$,

where p belongs to P_1 and q belongs to P_2 or vice versa. The transitions are carried out with participation of one *or many elements* p from P_1 and P_2.

According to the definition of the set P, the elements of the System are divided into two subsets P_1 and P_2. They might be considered as units moving along the reachable points. Element p can move from point x to point y if these points are reachable, i.e., $R_p(x, y)$ holds. The current location of each element is described by the equation ON(p)=x. Thus, the description of each state of the System $\{ON(p_j)=x_k\}$ is the set of descriptions of the locations of elements. The operator TRANSITION(p, x, y) describes the change of the state of the System caused by the move of the element p from point x to point y. The element q from point y must be withdrawn (eliminated) if p and q do not belong to the same subset (P_1 or P_2).

The problem of the optimal operation of the System is considered as a search for the optimal sequence of transitions leading from the initial state of S_i to a target state of S_t.

It is easy to show formally that a robotic system can be considered as a Complex System (see below). Many different technical and human society systems (including military

battlefield systems, systems of economic competition, positional games) that can be represented as twin sets of movable units (representing two or more opposing sides) and their locations can be considered as Complex Systems.

To solve this class of problems, we could use formal methods like those in the problem-solving system STRIPS (Fikes and Nilsson, 1971), nonlinear planner NOAH (Sacerdoti, 1975), or in subsequent planning systems. However, the search would have to be made in a space of a huge dimension (for nontrivial examples). Thus, in practice, no solution would be obtained.

We devote ourselves to finding an approximate solution of a reformulated problem.

5 SET OF PATHS: LANGUAGE OF TRAJECTORIES

This language is a formal description of the set of lowest-level subsystems, the set of all paths between points of the Complex System. An element might follow a path to achieve the goal "connected with the ending point" of this path.

A ***trajectory*** for an element p of P with the beginning at x of X and the end at the y of X ($x \neq y$) with a length l is following formal string of symbols $\boldsymbol{a}(x)$ with points of X as parameters:

$$t_0=\boldsymbol{a}(x)\boldsymbol{a}(x_1)\ldots\boldsymbol{a}(x_l),$$

where $x_l = y$, each successive point x_{i+1} is reachable from the previous point x_i, i.e., $R_p(x_i, x_{i+1})$ holds for i = 0, 1,…, l–1; element p stands at the point x: ON(p)=x. We denote by $t_p(x, y, l)$ the set of all trajectories for element p, beginning at x, end at y, and with length l.

A ***shortest trajectory*** t of $t_p(x, y, l)$ is the trajectory of minimum length for the given beginning x, end y, and element p.

Properties of the Complex System permit us to define (in general form) and study formal grammars for generating the shortest trajectories. Different examples are considered in (Stilman, 1994c).

A ***Language of Trajectories*** $L_t^H(S)$ for the Complex System in a state S is the set of all the trajectories of length less than H. Different properties of this language and generating grammars were investigated in (Stilman, 1993a).

6 NETWORKS OF PATHS: LANGUAGES OF TRAJECTORY NETWORKS

After defining the Language of Trajectories, we have new tools for the breakdown of our System into subsystems. According to the ideas presented in (Botvinnik, 1984), these subsystems should be various types of trajectory networks, i.e., the sets of interconnected trajectories with one singled out and called the *main trajectory*. An example of such network is shown in Fig. 1. The basic idea behind these networks is as follows. Element p_o should move along the main trajectory $\boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5)$ to reach the ending point 5 and remove the target q_4 (an opposing element). Naturally, the opposing elements should try to disturb those motions by controlling the intermediate points of the main trajectory. They should come closer to these points (to the point 4 in Fig. 1) and remove element p_o after its arrival (at point 4). For this purpose, elements q_3 or q_2 should move along the trajectories $\boldsymbol{a}(6)\boldsymbol{a}(7)\boldsymbol{a}(4)$ and $\boldsymbol{a}(8)\boldsymbol{a}(9)\boldsymbol{a}(4)$, respectively, and wait (if necessary) on the next to last point (7 or 9) for the arrival of element p_o at point 4. Similarly, element p_1 of the same side as p_o might try to disturb the motion of q_2 by controlling point 9 along the trajectory $\boldsymbol{a}(13)\boldsymbol{a}(9)$. It makes sense for the opposing side to include the trajectory $\boldsymbol{a}(11)\boldsymbol{a}(12)\boldsymbol{a}(9)$ of element q_1 to prevent this control.

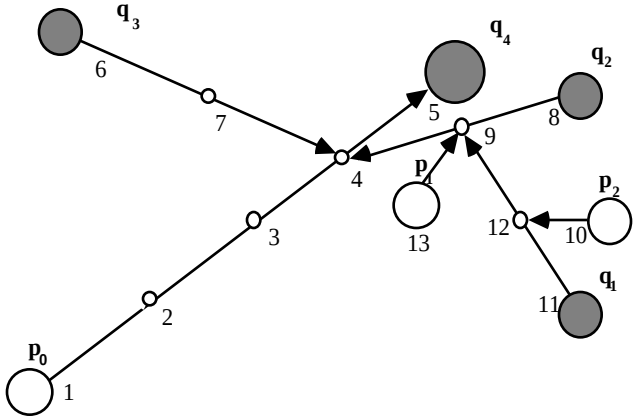

Fig. 1. Network language interpretation.

Similar networks are used for the breakdown of complex systems in different areas. Let us consider a linguistic formalization of such networks. The Language of Trajectories describes "one-dimensional" objects by joining symbols into a string employing a reachability relation $R_p(x, y)$. To describe networks, i.e., "multi-dimensional" objects made up of trajectories, we use the relation of *trajectory connection.*

A ***trajectory connection*** of the trajectories t_1 and t_2 is the relation $C(t_1,t_2)$. It holds if the ending link of the trajectory t_1 coincides with an intermediate link of the trajectory t_2; more precisely, t_1 is connected with t_2 if among the parameter values $\boldsymbol{P}(t_2)=\{y,y_1,\ldots,y_l\}$ of trajectory t_2 there is a value $y_i = x_k$, where $t_1=\boldsymbol{a}(x_o)\boldsymbol{a}(x_1)\ldots\boldsymbol{a}(x_k)$. If t_1 belongs to a set of trajectories with the common end-point, then the entire set is said to be connected with the trajectory t_2.

For example, in Fig. 1 the trajectories $\boldsymbol{a}(6)\boldsymbol{a}(7)\boldsymbol{a}(4)$ and $\boldsymbol{a}(8)\boldsymbol{a}(9)\boldsymbol{a}(4)$ are connected with the main trajectory $\boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5)$ through point 4. Trajectories $\boldsymbol{a}(13)\boldsymbol{a}(9)$ and $\boldsymbol{a}(11)\boldsymbol{a}(12)\boldsymbol{a}(9)$ are connected with $\boldsymbol{a}(8)\boldsymbol{a}(9)\boldsymbol{a}(4)$.

To formalize the trajectory networks, we define and use routine operations on the set of trajectories: $C_A^k(t_1,t_2)$, a ***k-th degree of connection***, and $C_A^+(t_1,t_2)$, a ***transitive closure***.

Trajectory $\boldsymbol{a}(11)\boldsymbol{a}(12)\boldsymbol{a}(9)$ in Fig. 1 is connected degree 2 with trajectory $\boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5)$, i.e., $C^2(\boldsymbol{a}(11)\boldsymbol{a}(12)\boldsymbol{a}(9), \boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5))$ holds. Trajectory $\boldsymbol{a}(10)\boldsymbol{a}(12)$ in Fig. 1 is in transitive closure to the trajectory $\boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5)$ because $C^3(\boldsymbol{a}(10)\boldsymbol{a}(12), \boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5))$ holds by means of the chain of trajectories $\boldsymbol{a}(11)\boldsymbol{a}(12)\boldsymbol{a}(9)$ and $\boldsymbol{a}(8)\boldsymbol{a}(9)\boldsymbol{a}(4)$.

A ***trajectory network*** **W** relative to trajectory t_o is a finite set of trajectories $t_o,t_1,\ldots,t_k$ from the language $L_t^H(S)$ that possesses the following property: for every trajectory t_i from W (i = 1, 2,…,k) the relation $C_W^+(t_i,t_o)$ holds, i.e.,

each trajectory of the network W is connected with the trajectory t_o that was singled out by a subset of interconnected trajectories of this network. If the relation $C_W^{\mathbf{m}}(t_i, t_o)$ holds, i.e., this is the **m**-th degree of connection, trajectory t_i is called the **m** ***negation trajectory***.

Obviously, the trajectories in Fig. 1 form a trajectory network relative to the main trajectory ***a***(1)***a***(2)***a***(3)***a***(4)***a***(5). We are now ready to define network languages.

A ***family of trajectory network languages*** $\mathbf{L_C(S)}$ in a state S of the Complex System is the family of languages that contains strings of the form

$$t(t_1, param)t(t_2, param)\ldots t(t_m, param),$$

where *param* in parentheses substitute for the other parameters of a particular language. All the symbols of the string t_1, t_2,…, t_m correspond to trajectories that form a trajectory network W relative to t_1.

Different members of this family correspond to different types of trajectory network languages, which describe particular subsystems for solving search problems. One such language is the language that describes specific networks called Zones. They play the main role in the model considered here (Botvinnik, 1984; Stilman, 1977, 1993b, 1993c, 1994a). A formal definition of this language is essentially constructive and requires showing explicitly a method for generating this language, i.e., a certain formal grammar, which is presented in (Stilman, 1993b, 1993c,1994a). In order to make our points transparent here, we define the Language of Zones informally.

A ***Language of Zones*** is a trajectory network language with strings of the form

$$Z=t(p_o,t_o,t_o)\ t(p_1,t_1,t_1)\ldots t(p_k,t_k,t_k),$$

where $t_o,t_1,\ldots,t_k$ are the trajectories of elements $p_o,p_2,\ldots,p_k$ respectively; $t_o,t_1,\ldots,t_k$ are nonnegative integers that "denote the time allotted for the motion along the trajectories" in a correspondence to the mutual goal of this Zone: to remove the target element – for one side, and to protect it – for the opposing side. Trajectory $t(p_o,t_o,t_o)$ is called the *main trajectory* of the Zone. The element q standing on the ending point of the main trajectory is called the *target.* The elements p_o and q belong to the opposing sides.

Consider the ***concurrent*** **Zone** corresponding to the trajectory network in Fig. 1.

$$Z=\boldsymbol{t}(p_o,\boldsymbol{a}(1)\boldsymbol{a}(2)\boldsymbol{a}(3)\boldsymbol{a}(4)\boldsymbol{a}(5),4)\ \boldsymbol{t}(q_3,\boldsymbol{a}(6)\boldsymbol{a}(7)\boldsymbol{a}(4),3)$$
$$\boldsymbol{t}(q_2,\ \boldsymbol{a}(8)\boldsymbol{a}(9)\boldsymbol{a}(4),\ 3)\boldsymbol{t}(p_1,\ \boldsymbol{a}(13)\boldsymbol{a}(9),\ 1)$$
$$\boldsymbol{t}(q_1,\ \boldsymbol{a}(11)\boldsymbol{a}(12)\boldsymbol{a}(9),\ 3)\ \boldsymbol{t}(p_2,\ \boldsymbol{a}(10)\boldsymbol{a}(12),\ 1)$$

Assume that the goal of the white side is to remove target q_4, while the goal of the black side is to protect it. According to these goals, element p_o starts the motion to the target, while black starts in its turn to move elements q_2 or q_3 to intercept element p_o. Actually, only those black trajectories are to be included into the Zone where the motion of the element makes sense, i. e., the *length of the trajectory is less than the amount of time (third parameter* t) *allocated to it.* For example, the motion along the trajectories ***a***(6)***a***(7)***a***(4) and ***a***(8)***a***(9)***a***(4) makes sense, because they are of length 2 and time allocated equals 3: each of the elements has 3 time increments to reach point 4 to intercept element p_o assuming one would go along the main trajectory without move omission and all the intercepting elements will move ***simultaneously*** (if necessary). According to definition of Zone, the trajectories of white elements (except p_o) could only be of the length 1, e.g., ***a***(13)***a***(9) or ***a***(10)***a***(12). As element p_1 can intercept the motion of the element q_2 at the point 9, black includes into the Zone the trajectory ***a***(11)***a***(12)***a***(9) of the element q_1, which has enough time for motion to prevent this interception. The total amount of time allocated to the whole bundle of black trajectories connected (directly or indirectly) with the given point of the main trajectory is determined by the number of that point. For example, for the point 4, it equals 3 time increments.

A language $L_Z^H(S)$ generated by the certain grammar $\mathbf{G_Z}$ (Stilman, 1993b, 1993c,1994a) in a state S of a Complex System is called the ***Language of Zones***.

7 ROBOTIC MODEL AS COMPLEX SYSTEM

For this model the set X (Section 4) represents the operational district, which could be the area of combat operation, broken into smaller square or cubic areas, "points", e.g., in the form of the big square or cubic grid. It could be a space operation, where X represents the set of different orbits, or an air force battlefield, etc. P is the set of robots or autonomous vehicles. It is broken into two subsets P_1 and P_2 with opposing interests; $R_p(x,y)$ represent moving capabilities of different robots for different problem domains: robot p can move from point x to point y if $R_p(x, y)$ holds. Some of the robots can crawl, others can jump or ride, sail and fly, or even move from one orbit to another. Some of them move fast and can reach point y (from x) in "one step", i.e., $R_p(x, y)$ holds, others can do that in *k* steps only, and many of them can not reach this point at all. ON(p)=x, if robot p is at the point x; v(p) is the value of robot p. This value might be determined by the technical parameters of the robot. It might include the immediate value of this robot for the given combat operation. S_i is an arbitrary initial state of operation for analysis, or the starting state; S_t is the set of target states. These might be the states where robots of each side reached specified points. On the other hand, S_t can specify states where opposing robots of the highest value are destroyed. The set of WFF $\{ON(p_j) = x_k\}$ corresponds to the list of robots with their coordinates in each state. TRANSITION(p, x, y) represents the move of the robot p from the location x to location y; if a robot of the opposing side stands on y, a removal occurs, i.e., robot on y is destroyed and removed.

8 TOTALLY CONCURRENT ROBOTIC MODEL: PROBLEM STATEMENT

Robots with different moving capabilities are shown in Fig. 2. The operational district X is the table 8 x 8. Robot W-FIGHTER (White Fighter) standing on h8, can move to any next square (shown by arrows). The other robot B-BOMBER (Black Bomber) from h7 can move only straight ahead, one square at a time, e.g., from h7 to h6, from h6 to h5, etc. Robot B-FIGHTER (Black Fighter) standing on a6, can move to any next square similarly to robot W-FIGHTER (shown by arrows). Robot W-BOMBER (White Bomber)

standing on c6 is analogous with the robot B-BOMBER; it can move only straight ahead but in reverse direction. Thus, robot W-FIGHTER on h8 can reach any of the points y {h7, g7, g8} in one step, i.e., $R_{W\text{-}FIGHTER}$(h8, y) holds, while W-BOMBER can reach only c7 in one step.

Assume that robots W-FIGHTER and W-BOMBER belong to one side, while B-FIGHTER and B-BOMBER belong to the opposing side: W-FIGHTER P_1, W-BOMBER P_1, B-FIGHTER P_2, B-BOMBER P_2. Also assume that two more robots, W-TARGET and B-TARGET, (unmoving devices or targeted areas) stand on h2 and c8, respectively. W-TARGET belongs to P_1, while B-TARGET P_2. Each of the BOMBERs can destroy unmoving TARGET ahead of the course. Each of the FIGHTERs is able to destroy an opposing BOMBER approaching its location, but it also able to destroy an opposing BOMBER if this BOMBER itself arrives at the current FIGHTER's location. For example, if the B-FIGHTER is at location c8 and W-BOMBER arrives there (unprotected) then during the same time increment it destroys the TARGET and is destroyed itself by B-FIGHTER. Each BOMBER can be protected by its friendly FIGHTER by approaching BOMBER's prospective location. In the latter case the joint protective power of the combined weapons of the friendly BOMBER and FIGHTER can protect the BOMBER from interception. For example, W-FIGHTER located at d6 can protect W-BOMBER on c6 and c7.

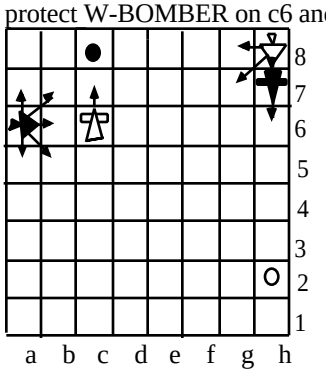

Fig. 2. 2D optimization problem for robotic vehicles with totally concurrent motions.

Each of the BOMBERs is vulnerable not only to a FIGHTER's attack but also to the explosion of another BOMBER. If W-FIGHTER hits B-BOMBER while the latter is fully armed, i.e., it is not at its final destination – square h2, and W-BOMBER is moving during the same time increment, it will be destroyed as a result of the B-BOMBER's explosion. If it is not moving at this moment it is safe. Similar condition holds for B-BOMBER: it can not move at the moment when W-BOMBER is being destroyed (excluding c8).

The combat considered can be broken into two local operations. The first operation is as follows: robot B-BOMBER should reach point h2 to destroy the W-TARGET, while W-FIGHTER will try to intercept this motion. The second operation is similar: robot W-BOMBER should reach point c8 to destroy the B-TARGET, while B-FIGHTER will try to intercept this motion. After destroying the opposing TARGET and keeping the BOMBER safe, the attacking side is considered as a winner of the local operation and the global combat. The only chance for the opposing side to avenge is to hit its TARGET at the same time increment and this way end the battle in a ***draw***. The conditions considered above give us S_t, the description of target states of the Complex System. The description of the initial state S_i is obvious and follows from Fig. 2.

Assume that all the agents of the opposing sides can move ***simultaneously***. There is no alternation of turns. It means, for example, that during the current time increment, all the four vehicles, W-BOMBER, W-FIGHTER, B-BOMBER, and B-FIGHER, three of them, two, one, or none of them can move. This means that this is a model with ***incomplete information*** about the current move (before it is done). When moving each side does not know the opposing side component of the concurrent move, i.e., the immediate opposing side motions, if they are not constrained to one or zero motions and, thus, can be predicted. Moreover, after developing a strategy each side can not follow it because of the uncertainty with the other side current motions. However, it the strategy includes only variations of concurrent moves with single "universal" component (group of motions) for one side good for all possible components of the other side, this strategy can be actually implemented.

It seems that local operations are independent, because they are located far from each other. Moreover, the operation of B-BOMBER from h7 looks like unconditionally winning operation, and, consequently, the global battle can be easily won by the Black side. ***Is there a strategy for the White side to make a draw?***

9 TOTALLY CONCURRENT ROBOTIC MODEL: SEARCH GENERATION

Consider how the hierarchy of languages works for the optimal control of this model. We have to generate the Language of Trajectories and the Language of Zones in each state of the search. The details these generations areconsidered in (Stilman, 1993b, 1993c, 1993d, 1994a). We generate the string of the Language of Translations (Stilman, 1994a) representing it as a search tree (Fig. 3) and comment on its generation. This tree is different from conventional search trees. Every concurrent move is represented by *two consecutive arcs*. The arc outgoing the white node represents the White component of a concurrent move, the concurrent motions of the White side, while the arc outgoing the black node represents the Black component of the same move.

First, the Language of Zones in the start state is generated. Every agent tries to attack every opposing side agent. The targets for attack are determined within the limit of five steps. It means that horizon H of the language $L_Z(S)$ is equal to 5, i.e., the length of main trajectories of all Zones must not exceed 5 steps. The algorithm for choosing the right value of the horizon is considered in (Stilman, 1994c). All the Zones generated in the start state are shown in Fig. 4. Zones for FIGHTERs as attacking elements are shown in the left diagram, while Zones for BOMBERs – in the right one.

Generation begins with the concurrent move 1. c6-c7 a6-b7 in the White Zone with the vulnerable Black target of the highest value and the shortest main trajectory. The order of consideration of Zones and particular trajectories is determined by the grammar of translations.

The Black component of this move, 1. ... a6-b7, is in the same Zone along the first negation trajectory. The

interception continues: 2. c7-c8 b7-c8/h7-h6 (Fig. 5, left). This is a triple move. During the second time increment W-BOMBER hit the TARGET at c8 and was destroyed by the B-FIGHTER at c8. Also, immediately, the attack Zone of the B-BOMBER from h7 to h2 was activated: h7-h6 is the motion during the same time increment. Here the grammar terminates this branch with the value -1 (as a win of the Black side). This value is given by the special branch termination procedure built into the grammar. This procedure determined that W-FIGHTER is out of the Zone of B-BOMBER, thus, it can not intercept B-BOMBER which means that the latter will successfully hit the TARGET on h2.

Fig. 3. Search tree for the Totally Concurrent Model.

Then, the grammar initiates the backtracking climb. Each backtracking move is followed by the inspection procedure, the analysis of the subtree generated so far. After climb up to the move 1. c6-c7 a6-b7, the subtree to be analyzed consists of one branch (of one move): 2. c7-c8 b7-c8/h7-h6. The inspection procedure determined that the current minimax value (-1) can be "improved" by the improvement of the exchange on c8 (in favor of the White side). This can be achieved by participation of W-FIGHTER from h8, i.e., by generation and inclusion of the new so-called "control" Zones with the main trajectory from h8 to c8. These Zones were detected (within the horizon 5) in the terminal state after the move 2. c7-c8 b7-c8/h7-h6, Fig. 5 (left). Obviously they could not be detected in the initial state of this problem (Fig. 4) because the main element, W-BOMBER, could not "see" the target, B-FIGHTER, within given horizon. Also, at the moment of detection it was too late to include them into the search. These Zones have been stored and kept idle for possible activation at the higher levels of the search tree. The set of different Zones from h8 to c8 (the bundle of Zones) is shown in Fig. 5 (right). The move-ordering procedure picks the subset of Zones with main trajectories passing g7. These trajectories partly coincide with the main trajectory of another Zone attacking the opposing W-BOMBER on its future location h6. The motion along such trajectories allows to "gain time", i.e., to approach two goals simultaneously.

The generation continues with the simultaneous motion of all four agents, the four-move, W-BOMBER, W-FIGHTER

and B-FIGHTER, B-BOMBER, in their respective Zones: 2. c7-c8/h8-g7 b7-c8/h7-h6. The B-FIGHTER intercepted W-BOMBER at c8 while W-FIGHTER is unable to intercept the B-BOMBER during its attack from h6 to h2. The branch termination procedure determined that W-FIGHTER is outside the B-BOMBER's attack Zone, terminated this branch, evaluated it as a win for the Black (-1), and initiated the backtracking climb. Move 2. ... was changed for the triple move 2. h8-g7 b7-c8/h7-h6 in attempt to find a better combination of White motions.

Black side, after finding b7-c8/h7-h6 to be a "good" component of the concurrent move 2. in the previous branches, continues to include this component in the following branches. Obviously, this component is very important. As it was noted above, a totally concurrent model is a model with incomplete information. Each side knows all the previous moves, the history of operation, and, theoretically, all possible future outcomes of the current move, the look-ahead tree. The only thing it does not know is the concurrent action of the opposing side as a component of the current move. Thus, for each side it is important to find not just a "good" own component of a concurrent move but a component to be "good" for all components of the opposing side. Such component would allow to avoid uncertainty in constructing an optimal variation, a branch, which can be implemented. A component b7-c8/h7-h6 is a candidate to be a good one for the Black while h8-g7 is a candidate for White.

After 2. h8-g7 b7-c8/h7-h6 termination procedure did not terminate the branch, and continued 3. c7-c8 h6-h5 in the same Black and White Zones. Then it terminated the branch and evaluated it as a win (-1) for the Black side (Fig. 6, left). Indeed, W-BOMBER hit B-TARGET on c8 but it is being destroyed itself by B-FIGHTER which was waiting for it at c8. Also, W-FIGHTER again is out of the attack Zone of B-BOMBER from h5 to h2. In this state a set of new control Zones of W-FIGHTER from g7 to c8 were detected and stored as idle to be activated later if necessary.

New climb up to the move 2. h8-g7 b7-c8/h7-h6 and execution of the inspection procedure result in the inclusion of the groups of new control Zones from g7 to c7 and c8 in order to improve the exchanges at these locations. Both groups of Zones (to c7 and c8) have been detected earlier in the search tree. The set of Zones with different main trajectories from g7 to c7 and from g7 to c8 is shown in Fig. 6 (right). Besides that, the trajectories from g7 to h4, h3, and h2, are shown in the same Fig. 6. These are "potential" first negation trajectories. It means that beginning with the second symbol ***a***(f6), ***a***(g6) or ***a***(h6) these trajectories become first negation trajectories in the Zone of B-BOMBER on h6. Speaking informally, from the squares f6, g6, and h6, Zone gateways, W-FIGHTER can intercept B-BOMBER. The move-ordering procedure picks the subset of Zones with the main trajectories passing f6. These trajectories partly coincide with the potential first negation trajectories. The motion along such trajectories allows to "gain time", i.e., to approach two goals simultaneously.

Thus, the new White component 3. c7-c8/g7-f6 is included with the same Black component 3. ... h6-h5, the branch was terminated with the value -1. The following climb and branching with inclusion of g7-f6 as a single motion component resulted in 3. g7-f6 h6-h5, and the branch is not terminated. It continues with the move 4. c7-c8 h5-h4. This state is shown in Fig. 7, left. Then this branch is terminated with the value -1. As usual, this value was assigned by the termination procedure which detected that W-FIGHTER is outside the Zone of B-BOMBER and thus does not have enough time for interception.

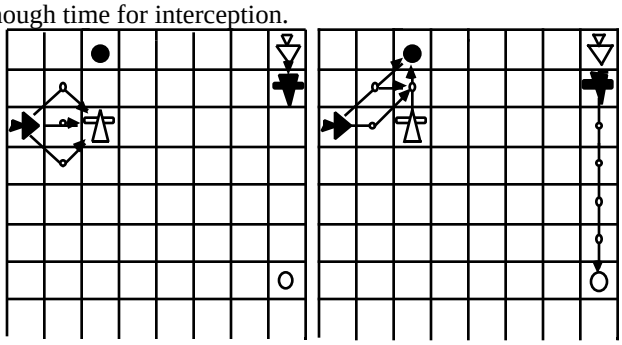

Fig. 4. Zones in the initial state

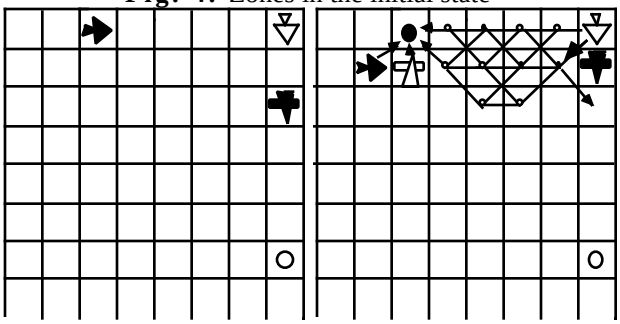

Fig. 5. States where the control Zone from h8 to c8 was detected (left) and where it was included into the search (right)

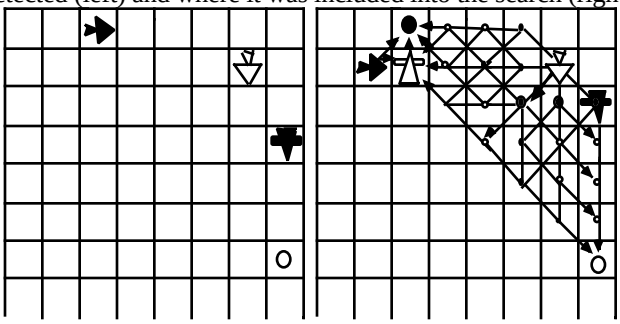

Fig. 6. States where the control Zones from g7 to c7, c8 were detected (left) and where they were included into the search (right)

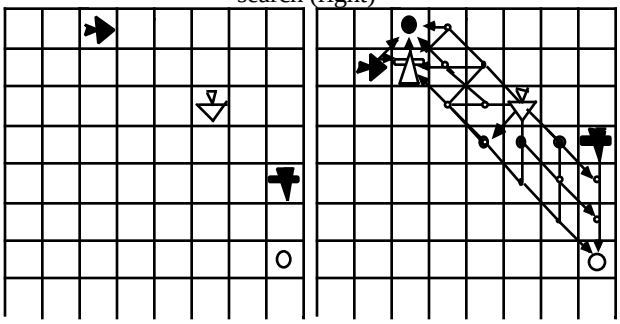

Fig. 7. States where the control Zones from f6 to c7, c8 were detected (left) and where they were included into the search (right).

After the climb, the grammar continued branching 4. c7-c8/f6-e5 h5-h4. The component f6-e5 is selected by the move ordering procedure as the time-gaining move approaching two goals simultaneously, c7 as a goal of the control Zone of W-

FIGHTER and one of the gateways (e5, f5, g5) of the Zone of B-BOMBER (Fig. 7, right). But it was also terminated with the value -1. After 4. f5-e5 h4-h3, 5. e5-d6 h4-h3, and 6. c7-c8/d6-d7 h3-h2, the branch is terminated with the value of 0.

It seems that the sought draw is found. The following climb with activation of the inspection procedure in every node ended at the top level. All the attempts of the Black to change the components 4. ... h5-h4, 3. ... h6-h5, 2. ... h7-h6 for a different motion failed. If B-BOMBER's motion is not included in these concurrent moves the W-FIGHTER appears in the B-BOMBER's attack Zone and these branches should be terminated with the value 0 which does not improve the current minimax value for Black.

The Black component of 1. c6-c7 a6-b7 was changed for the double motions 1. c6-c7 a6-b7/h7-h6. It seems that this move almost depreciated previous search. The minimax value brought to the top of the subtree outgoing this move is -1. However, the tree generation followed after the change of 1. c6-c7 a6-b7/h7-h6 for the double move 1. c6-c7/h8-g7 a6-b7/h7-h6 showed that previous search was very important. As a result of this search the grammar *learned key networks*, Zones of W-FIGHTER with main trajectories from g8 to c8, from g7, f6 to c7 and c8. The optimal branch is shown in Fig. 3 with bold lines.

10 DISCUSSION

The total number of moves included in this tree is 34. The maximum depth reached is 6. This means that the branching factor (Nilsson, 1980) of this tree is 1.53, i.e., the search is highly goal-oriented. The average number of legal motions for each side, i.e., the average number of different legal components of every concurrent move, is 18. Thus, the average number of legal moves in each state, the unreduced branching factor, is $18 \times 18 = 324$(!), taking into account all the combinations of legal components. Obviously, 34 is a dramatic reduction in comparison with a 324^6 move tree that would have to be generated by conventional searchprocedures, or even with the theoretical minimum of the minimax search with alpha-beta cut-offs $(324^6)^{1/2} = 18^6 \approx 34$ million.

Search reduction achieved in the serial case (Stilman, 1994b, 1994c) with one-at-a-time motion of every vehicle multiplied tremendously in the new example with the allowance of *concurrent* moves. The next step of our research will be a formal investigation of the complexity of the hierarchy of languages which represents each state in the search process. It is easy to speculate that the growth from the serial case to the concurrent one is limited by multiplication to a constant factor close to one.

References

Botvinnik, M.M. (1984). *Computers in Chess: Solving Inexact Search Problems.* Springer Series in Symbolic Computation, New York: Springer-Verlag.

Botvinnik, M., Petriyev, E., Reznitskiy, A., et al. (1983). Application of New Method for Solving Search Problems For Power Equipment Maintenance Scheduling. *Economics and Mathematical Methods* (pp. 1030-1041), 6, (in Russian).

Chomsky, N. (1963). Formal Properties of Grammars. in *Handbook of Mathematical Psychology*, eds. R.Luce, R.Bush, E. Galanter., vol. 2 (323-418). New York: Wiley & Sons.

Feder, J. (1971). Plex languages. *Inform. Sciences*, 3,(225–241).

Fikes, R.E. and Nilsson, N.J. (1971). STRIPS: A New Approach to the Application of Theorem Proving in Problem Solving. *Artificial Intelligence* 2: 189–208.

Fu, K.S. (1982). *Syntactic Pattern Recognition and Applications*, Prentice Hall, Englewood Cliffs.

Garcia-Ortiz, A. et al. (1993). Application of Semantic Control to a Class of Pursue-Evader Problems, *Computers and Mathematics with Applications*, 26(5), (97-124).

Ginsburg, S. (1966). *The Mathematical Theory of Context-Free Languages*, McGraw Hill, New York.

Knuth, D.E. (1968). Semantics of Context-Free Languages. *Mathematical Systems Theory*, (pp. 127–146), 2.

Lirov Y., Rodin, E.Y., McElhaney, B.G., and Wilbur, L.W. (1988). Artificial Intelligence Modeling of Control Systems, *Simulation*, (12-24), 50(1).

McCarthy, J. (1980). Circumscription–A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, (27-39), 13.

McCarthy, J. and Hayes, P.J. (1969). Some Philosophical Problems from the Standpoint of Artificial Intelligence. *Machine Intelligence* (463–502), 4.

Narasimhan, R.N. (1966). Syntax–Directed Interpretation of Classes of Pictures. *Comm. of ACM* (166–173), 9.

Nilsson, N.J. (1980). *Principles of Artificial Intelligence*, Palo Alto, CA: Tioga Publ.

Pavlidis, T. (1977). *Structural Pattern Recognition*, New York: Springer-Verlag.

Rodin E. (1988). Semantic Control Theory, *Applied Mathematical Letters*, (pp. 73-78), 1(1).

Rodin E., Garcia-Ortiz et al. (1993). Application of Semantic Control to a Class of Pursue-Evader Problems, *Computers and Mathematics with Applications*, 26(5).

Rosenfeld, A. (1979). *Picture Languages, Formal Models for Picture Recognition*, Academic Press.

Rozenkrantz, D.J. (1969). Programmed Grammars and Classes of Formal Languages, *J. of the ACM* (pp. 107–131), 1.

Sacerdoti, E.D. (1975). The Nonlinear Nature of Plans, *Proc. Int. Joint Conference on Artificial Intelligence*.

Shaw, A.C. (1969). A Formal Picture Description Scheme as a Basis for Picture Processing System, *Information and Control* (9-52), 19.

Shinar, J., (1990). Analysis of Dynamic Conflicts by Techniques of Artificial Intelligence, INRIA Report, Antipolis.

Stilman, B. (1977). The Computer Learns. in Levy, D., *1976 US Computer Chess Championship* (83-90). Computer Science Press, Woodland Hills, CA.

Stilman, B. (1985). Hierarchy of Formal Grammars for Solving Search Problems. In *Artificial Intelligence. Results and Prospects, Proceedings of the International Workshop* (pp. 63-72), Moscow, (in Russian).

Stilman, B. (1993a). A Linguistic Approach to Geometric Reasoning, *Int. J. Computers and Mathematics with Applications* (29-57), 26(7).

Stilman, B. (1993b). Network Languages for Complex Systems, *Int. J. Computers and Mathematics with Applications* (51-79), 26(8).

Stilman, B. (1993c). Syntactic Hierarchy for Robotic Systems, *Integrated Computer-Aided Engineering* (pp. 57-81), 1(1).

Stilman, B. (1993d). A Formal Language for Hierarchical Systems Control, *Languages of Design* (pp. 333-356), 1(4).

Stilman, B. (1994a). Translations of Network Languages. *Int. J. Computers and Mathematics with Applications* (65-98), 27(2).

Stilman, B. (1994b). A Formal Model for Heuristic Search. *Proc. of the 22nd Annual ACM Computer Science Conf.*, (pp. 380-389), March 8-10, Phoenix, AZ.

Stilman, B.(1994c). Heuristic Networks for Space Exploration, *Telematics and Informatics, Int. J. on Telecommunications & Information Technology*, 11(4), (403-428).