ANNALS OF THE FACULTY OF ENGINEERING HUNEDOARA
2004, Tome II, Fascicole 2

NOGOOD ELIMINATION IN THE ALGORITHMS DISTRIBUTED WITHIN THE DCSP MODELING (DISTRIBUTED CONSTRAINT SATISFACTION PROBLEM)

MUSCALAGIU Ionel, ABRUDEAN Cristian

THE UNIVERSITY „POLITEHNICA” DIN TIMISOARA,
THE FACULTY OF ENGINEERING OF HUNEDOARA, ROMANIA

ABSTRACT

One of the characteristics of the algorithms of asynchronous search is that of the appearance of the "nogood" values while searching for the solution. The number of the recordings of nogood messages determines the exponential complexity of the algorithm in the unfavorable case in the first place. As a consequence, we have the fact that the calculus time and the askings for the hardware resources grow much faster than the dimension of the problem. Doubtlessly in the practical applications, in which we use slow means communications such as Internet, the algorithms become inefficient because of the large number of the developed nogood values and their necessity for communications during the searching. In this article we will study the effect of their elimination or diminution. We will demonstrate that is a connection between the quantity of nogood values and the efficiency of the asynchronous searching algorithms.

Keywords
Artificial intelligence, distributed programming, constraints, agents.

1. INTRODUCTION

According to the IT literature the backtracking algorithm distributed in an asynchronous way- ABT, existing for the DCSP model, is considered the first complete algorithm for the asynchronous search. It is the first complete algorithm asynchronous, distributed and competitor, in which the agents the agents can roll up in a competitive and asynchronous way, published Yokoo and al. in [8]. This algorithm is based on sending nogood messages among agents for doing an intelligent backtracking and to assure the complexity of the algorithm. The nogood messages are lists of joined values at distinct variables in which there are inconsistent among some variables.

The appearance of the nogood values has as an effect the introduction of some new constraints. Although the nogood list indicates the cause of the failure and its incorporation as a new constraint will teach the agents not to repeat the same mistake, it is expected that during the course of the algorithm the nogood values to be as few as possible, because they have as an effect the increasing of the execution time (because of the fact that new messages are sent, etc.)

The number of the recordings of nogood messages determines the exponential complexity of the algorithm in the first place. As a consequence, in the practical applications, in which slow means of communications are used, such as Internet, the algorithm becomes totally inefficient because of the large number of the nogood values that must be communicated during the search.

As the nogoods are in fact a way of constraints dynamic generated while searching, which is some situations rise in an explosive way, we have to study the effect of their elimination or diminution.

In this context, during the last years a lot of studies were done and they ended in results based on the fact that this algorithm of reference can be modified in an algorithm with a polynomial space of searching. As a fact there are studies from [3], [4], [9], [6], which created some algorithms efficiently distributed. There is about the algorithms DIBT-Distributed Backtracking, AAS-Asynchronous Search with Aggregations, DisDB-Distributed Dynamic Backtracking. We will demonstrate further how these algorithms are in fact (in some sort) variants ABT with the reductions or elimination of nogood messages. At the other extreme there is AWCS- Asynchronous weak- Commitment Search [9], which records all the nogood values, algorithm which is however much more efficient than ABT because of the dynamical order. The idea of permitting the agents to be able to modify a wrong decision by a dynamical change of the agent priority order, proved beneficial in case of AWCS.

We will demonstrate that there is a connection between the quantity of nogood values and the efficiently of the asynchronous searching algorithms. For example, the asynchronous search with (AAS) units or the backtracking distributed (DIBT) is efficient algorithms because they eliminate a part of the nogood values.

2. THE FRAMEWORK

In order to do this analysis of the impact of the nogood values, in this paragraph we will present some notions known from the IT literature relative to the DCSP modeling and ABT algorithm.

Definition 1.-CSP model. The model based on constraints CSP-Constraint Satisfaction Problem, existing for centralized architectures, consists in:

-n variables X_1, X_2. X_n, whose values are taken from finite, discrete domains D_1, D_2, …., D_n, respectively .

-a set of constraints on their values.

The solution of a CSP suppose to find an association of values to all the variables so that all the constraints to be fulfilled.

Definition 1.-The DCSP model. A problem of satisfying the distributed constraints (DCSP) is a CSP, in which the variables and constraints are distributed among autonomous agents that communicate by transmitting, messages.

In this algorithm, each agent instantiates its variables in a competitive way and sends the value to the agents with which is directly connected, using direct communication channels which function according to the FIFO principle. It is also considered a global statically order among the agents, in which the A_i agent has a smaller priority than A_j, if i>j. In this way A_j can impose the first the favorite values.

Definition 2.- the assignment. It is called assignment for the variable X_i a pair (X_i,v), where v is a value from the X_i domain.

Definition 3. - the list agent_view. The list agent_view of an agent A_i is a set with the newest assignments received by the A_i agent for distinctive variables.

Definition 4.- the nogood list. The Nogood list is a set of assignments for distinctive variables for which looseness was found.

The ABT algorithm uses 3 types of messages:

- the OK message, which contains an assignment variable –value, is sent by an agent to the estimate in order to see if the value is good.
- the nogood message which contains a list (called nogood) with the assignments for the looseness, it is being sent in case in which the estimator agent found an unfulfilled constraint.
- the add- link message, sent to announce the necessity to create a new direct link, owing to a nogood appearance.

Each agent receives a lot of values from the agents it is being connected to through links, these values forming agent_view. If the OK?, message is received the estimator agent adds the variable the value in the list of values and checks if the new pair is consistent with the others. If an assignment that is not consistent is found, the agent tries to change this value so that to be consistent with the values from the agent_view list. It is possible that the agent not to find any good combination for some pairs in the list (such a set is nogood), than the values assigned for other agents can be changed. In this situation, it is said that the agent caused a backtrack, it having to send a nogood message to one of the agents.

3. THE REDUCTION OF *NOGOOD* IN AWCS CASE.

The AWCS algorithm is a hybrid algorithm obtained by the combination of ABT algorithm with WCS algorithm, which exists for CSP. It can be considered as being an improved ABT variant, but not necessarily by reducing the nogood values, but by changing the priority order. It deliberately follows to record all the nogood values (which are fewer) to ensure the completeness of the algorithm, but also the avoidance of some unstable situations.

The authors show in [9] that this new algorithm can be built by the dynamical change of the priority order. The AWCS algorithm uses, like ABT, the two types of *ok* and *nogood* messages, with the same significance. There is a major difference in the way you treat the ok message. In case of receiving the ok message, if the agent can't find a value to its variable that should be consistent with the values of the variables that have a greater priority, the agent not only creates and sends the nogood message, but also increases the priority in order to be maximum among the neighbors.

We have to emphasize that the AWCS algorithm also requires the recording of each nogood list to assure the completeness of the algorithm, giving the impression of an exponential space inefficient in the real environment. However the experiments show a bigger efficiency towards the ABT algorithm. According to the experimental results, the AWCS algorithm, especially for problems of a larger dimension, proved to be very efficient. More than that the algorithm could give an answer to certain instance in an acceptable period of time, matter that the basically ABT algorithm hadn't succeeded.

In conclusion, AWCS is efficient and complete because of the recording of all the nogoods (which are much fewer), but suffer after the explosion of the appearance of nogoods. So, the costs owed to the checking of the constraints can become expensive because an agent in AWCS can create nogoods for all its neighbours. A last idea, linked by the practice, is to limitate the number of nogood recordings at a fixed value, sacrificing the completeness, but getting fast results.

4. IN ELIMINATION OF NOGOOD IN CASE OF ASYNCHRONOUS SEARCH WITH AGGREGATIONS.

Silaghi suggests, in [6], two techniques the ensurance of a polynomial space in case of asynchronous search: the tagging of assignments and retransmission of the messages.

These two techniques are the beginning of the creation of three variants that reduce or eliminate completely the nogood values:

- **AAS-2:** is based on the complete recording of nogood list, similar to the Yokoo's asynchronous backtracking algorithm (ABT).
- **AAS-1:** proceeds similar to the dynamic backtracking variant in [1]. The nogoods that depend on the assignment of the modified variables, being obtained a space of polynomial complexity.
- **AAS-0**: it is a modified variant of AAS-1 with the fewest nogood recordings. AAS-0 is an algorithm, which ties the entire nogood list, kept by every agent of AAS1 in just one nogood using more of relaxation presented in [6].

The tagging of assignments is done by introducing a local counter, which is incremented by every agent every time when a new instance is chosen and the current value labels each generated assignment. This labeling assures an order of sending the messages. Silaghi presents in [7] an algorithm named ABT_p got through modifying the ABT algorithm by adding the previous labeling. This demonstrates much more direct the link between the elimination of nogood values and the polynomial space.

The retransmission of the messages is a second solution of nogood suggested by Silaghi for the elimination of nogood values from ABT. It consists in retransmitting the suggestions after every change of a view. The agents that have a smaller priority and received once again deduce a nogood for these values. Silaghi presents in [7] an algorithm named ABT_r obtained by modifying the ABT algorithm through adding the techniques of retransmitting the messages getting an algorithm with a polynomial space. This second technique requires transmitting much more messages, being much more expensive. It is applied in AAS-0 after the nogood values are eliminated.

The algorithms AAS0, AAS1, AAS2 were estimated using long snapshots of messages and constraints. The AAS2 algorithm supplies better results than the ABT algorithm. It is remarked the fact that, in case that there is no solution, the AAS2 algorithm supplies constantly better solutions than ABT and can reduce the long snapshots of messages and the number of nogoods on an average with 50 %.

As a conclusion, the use of the sets technique offers an improvement of the efficiency for searching for the solution, conversely proportional to the quantity of nogood recorded.

5. THE ELIMINATION OF NOGOOD IN DIBT CASE

Another algorithm of asynchronous search is DIBT published in [4] and [3. this is a variant of algorithm which does not requires for add-link messages eliminating almost entirely the recording of nogood values.

The variant of DIBT, has as a starting base the classical algorithm of backtracking, the centralized case. If the ABT uses schemes of learning, this algorithm eliminates the schemes of learning, such as the recording of the nogood.

The elimination of the nogood messages is done using few techniques of improving the backjumping based on the graph of constraints. In fact, this method is based most of it on relative techniques of backjumping. In the first place, the centralized variant of backjumping used the graph of constraints to determine the origin of the failure. The DIBT algorithm uses this idea also. The second idea is that of preserving the previous work through the fact that when getting a message, first it is checked if the current value satisfies the constraints with the transmitter agent, before trying another value form its domain. This idea reduces the effort of calculation because if there isn't any change, neither its children are announced. The third idea consists in using a so named repair technique. At the beginning the entire system initializes the variables simultaneously. While rolling the algorithm, the agents revise their values according to their environment. So, the system starts with global instances of the variables and then executes local reparation of different parts of the instance, but simultaneously.

The assessment of this method, comparatively with the ABT method was done solving aleator DCSP problems getting better results. It must be mentioned the fact that the elimination of add-link messages is favorable for problems that have the graph of constraints very dense, but has as an effect the engendering of some useless connections which the ok messages produces and throttle the network.

We also have to specify the fact that the first DIBT variant, published in [4], had some difficulties as far as the finding of the solution is concerned. To eliminate these difficulties and assure the completeness of the algorithm, in [3] the author extends the algorithm, adding in advance some ABT connections. Unfortunately, in [10], it is shown that this algorithm is incomplete (with all these extensions). As a conclusion, we notice the complete elimination of the additional ABT connections and of nogood values is not possible 100 %.

6. EXPERIMENTAL RESULTS

In the previous paragraph we saw diverse estimations of the algorithm distributed for search, comparatively with the ABT algorithm, results obtained from IT literature and from our own estimations. We will present our own experimental results for the estimation of the quantity of nogood recorded and used by the presented techniques. In order to make such estimation, we implemented these techniques in NetLogo 1.3, a distributed environment, done in Java, using a special language named NetLogo (see [10]).

These four techniques were used in order to make estimation to a classical problem (distributed variant) the problem of the n queens. The number of nogood values was countered. The obtained results are visualized in figure 1.

Studying the effect of elimination or reducing the nogood, we notice that there is a connection between the quantity of nogood and efficiency of the asynchronous search algorithm.

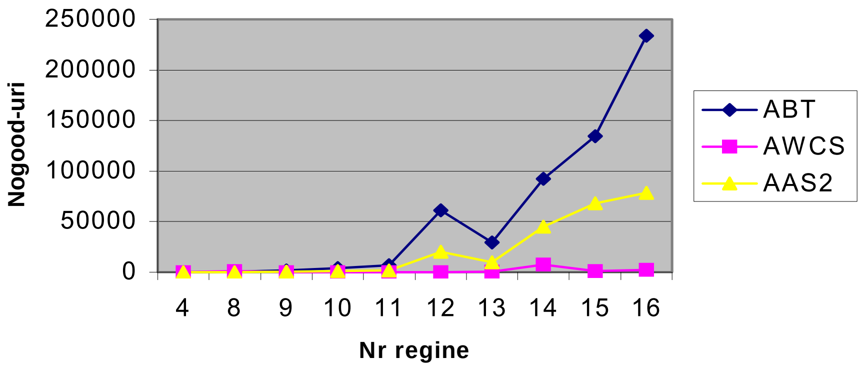

Figure 1. The Evaluation of the efficiency after the number of the nogood

7. CONCLUSIONS. THE ANALYSIS OF THE NOGOOD IMPACT

The main feature of the ABT algorithm is the way in which the backtracking is processed (deadend) to assure the completeness while searching and that is the appearance of the nogood. It is the main problem of the ABT algorithm that leads to the exponential complexity of the algorithm.

The AWCS algorithm uses a dynamic order that can be changed while searching. However, to assure the completeness, the nogood can't be eliminated and so the algorithm has a space of exponential complexity, but very efficient.

The AAS algorithm constantly supplies better solution than ABT and can reduce the number of nogoods recorded on an average with 50 %. As a conclusion, the use of the aggregation technique offers an improvement of the efficiency for finding the solution, conversely proportional with the quantity of nogood stocked.

The DIBT algorithm tries to keep the distributed structure of the network as much as possible. It is built hierarchy of the agents using a method called Distributed Agents Ordering (DisAO), without adding other new connections neither before, nor during other the search). DIBT doesn't have to record the nogood. Unfortunately, the completeness is not guaranteed any more by adding some additional connections.

8. BIBLIOGRAPHY

[1] Christian Bessière, Arnold Maestre, Pedro Meseguer. *Distributed Dynamic Backtracking*. In Proceedings of the CP Workshop on DCSP, Singapore, 2000.

[2] M. Ginsberg. *Dynamic backtracking.* Journal of A.I Research 1:25-46, 1993.

[3] Hamadi Y. – Traitement des problemes de satisfaction de constraintes distributes. PhD theisis, Universitatea Motpellier II, 1999.

[4] Hamadi Y., Bessiere C., Quinqueton J. *Backtracking in Distributed Constraint Networks*. In Proceedings of the 13th, ECAI , Brighton, UK, 1998, pag. 219-223.

[6] Marius Călin Silaghi, D. Sam-Haroud, B. Faltings. *Asynchronous Search with Aggregations*. In Proceedings AAAI'00, pages 917-922, Austin, Texas, 2000.

[7] Marius Călin Silaghi . *Asynchronously Solving Distributed Problems With Privacy Requirements.* PhD theisis, Lausanne , EPFL 2002, Elvetia.

[8] Yokoo M., E.H. Durfee, T. Ishida, K. Kuwabara - *Distributed constraint satisfaction for formalizing distributed problem solving.* In ICDCS, 614{621). 1992.

[9] Yokoo M., E.H. Durfee, T. Ishida , K. Kuwabara - *The distributed constraint satisfaction problem : formalization and algorithms* . IEEE, 1998 .

[10] Yokoo Makoto . Private communication, 2000.

[11] http://ccl.northwestern.edu/netlogo/