Applying Interaction-based Problem Solving for Locating Deep Water Oil Wells and Offshore Platforms

Leandro Prade Nadaletti, Flávio Miguel Varejão

Departamento de Informática – Universidade Federal do Espírito Santo (UFES)
Av. Fernando Ferrari, s/n - Campus de Goiabeiras
CEP 29060-900 - Vitória - ES – Brazil
lpradena@ig.com.br, fvarejao@inf.ufes.br

Abstract. Several computational approaches have been applied to support problem solving on oil exploration activities in deep waters. While the focus of these works consists of finding effective methods that bring solutions to the problem, this article describes a new research perspective. It is based on the idea that the adoption of principles that improve the interaction between the user and the computational system can lead to a more efficient final result than the ones obtained by any of those entities (user or computational system) alone.

1 Introduction

One of the most important decisions for planning the installations of oil exploration in deep waters is the submarine layout, i.e., the choice of wells and offshore platforms location. It involves locating and associating the wells and platforms through pipelines for conducting the drained liquid mass. Good choices reduce the overall costs by reducing pipelines length and the need of additional equipments. Such problem is characterized by a great complexity that makes the process of problem solving difficult for human specialists as well as stand-alone computational systems. The objective of this work is to investigate if the deep interaction between a human expert and a computational system may obtain better results than the ones obtained individually by each one of these agents when performing submarine layout problem solving.

The idea of interaction-based problem solving follows from the perception of the limitations of human beings and computational systems when solving large scale and complex problems. In the case of human beings, the limitation is due to our limited capacity of efficiently process large quantities of data and information. In the case of computational systems, the limitation is caused by the combinatorial nature of some problems. These problems, in large scale, are computationally intractable.

The view that the interaction may contribute to problem solving follows from the combination of capabilities among the agents responsible for solving the problem. This idea has had wide acceptance in the multiagent research area [3]. Moreover, there are conjectures that the simple interaction of computational systems can bypass the computational intratactability of problems of combinatorial nature [8]. We believe that, in the case of human-computer interaction, the combination of the creative and learning capacities of the human being and the wide capacity of processing data of computer systems can optimize problem solving, improving the solutions quality

and also extending the space of problems that may be solved. In this direction, Hearst [6] makes an interesting proposal that conciliates the fields of Artificial Intelligence (AI) and Human Computer Interaction (HCI). This area of research was called "mixed-initiative interaction" (MII). Despite the apparent duality of the AI and IHC areas, MII tries to work on the limit of these fields. From AI, it uses the idea of creating a computational system that can carry intelligent actions. From HCI, it uses the concepts of interfaces design to facilitate users in the execution of intelligent actions.

Initially this paper presents the features of the submarine layout problem. Then, it shows some types of interactions incorporated to a under-development computational tool, called SALAS, to support the generation of the submarine layout. Next, it describes a small scale experiment using the current version of SALAS that compares the results obtained by the three forms of problem solving (human, computer and human-computer) in a submarine layout problem. Finally, we have an analysis of the results, along with a discussion about the contributions of this work and the perspectives of future work.

2 Submarine Layout

Exploration of oil fields in deep and ultra-deep water involves the draining and pumping of oil from the wells to the offshore platforms to be processed and exported to land. One of the largest costs involved in offshore exploration is related to the pipelines used for sending the oil from the wells to the platform. These pipelines are flexible and reinforced for supporting the deep water requirements of pressure and movement. They may cost over seven thousand dollars by linear meter.

One offshore platform may be responsible for processing the oil drained from many wells. Moreover, many platforms may be used for draining the wells in a field. Since wells may be kilometers farther from the platform, the decisions about the number of platforms to be used and about the actual position of the platforms and wells are extremely important. Better decisions may save hundreds of thousands, or even millions, of dollars. Beyond platforms and wells, there are other important elements to be considered when solving this problem. There are manifolds used for merging the oil of pipelines coming from different wells. There may also be obstacles restricting the plataform, manifold and pipeline positions.

The Submarine Layout (SL) problem consists of locating oil wells, plataforms, manifolds and pipelines in a bi-dimensional plan (representing the sea), taking into account a set of possible obstacles. Thus, the elements that must be considered when solving a submarine layout problem are the following: a) Production Area: the geographic region with oil fields to be explored; b) Well: a point in a circular region of interest for drilling the oil well, defined as its objective; c) Manifold: - Substation for concentrating the oil coming from several wells; d) Platform: element responsible for receiving and processing the oil obtained from the wells or manifolds; e) Obstacles: polyhedric region located inside the production area where one layout element (objective, well, manifold, platform or pipeline) cannot be located. The process of associating wells to a platform must take into account the platform production capacity. So, the overall production of the wells associated to a platform cannot surpass the plataform production capacity.

Some subproblems involved in the location of oil wells and platforms have already been investigated. Devine and Lesso [1] proposed a heuristic solution for the problem of associating wells to platforms, and locating the plataforms in a continuous space, considering predefined wells positions. Dogru [2] and Hansen et al. [5] also consider heuristic solutions for this problem, although restricting the possible locations of the platforms to a finite set. Goldbarg and Luna [4] consider the problem of the location of wells and manifolds associated to one platform only. Silva and Garcia [7] consider the location and association of wells and platforms, as well as the determination of the pipelines paths, taking in account the existence of obstacles in the production area. However, their approach does not use manifolds.

3 Interaction-Based Problem Solving

Algorithms have been used to investigate and support the solution of combinatorial problems (such as the one presented in this work). Due to the high complexity of these problems, it is not easy to find an algorithm that obtains good solutions to all problem cases. On the other hand, approaches where humans must take all decisions to find the problem solutions do not take advantage of the overall computer capability.

This work is based on the idea that the adoption of principles for promoting a better interaction between human users and computational systems may lead to a more efficient final result than the one obtained from any of these entities (program or user) alone. The main advantage of this approach is the complementarity of capabilities between humans and computational systems. While the human user may use his common sense knowledge, creativity and experience to guide the search for a solution, the computational system may explore efficiently a broad space of alternatives, searching for an appropriate answer. Another advantage is that this partnership tends to produce good results even if the available computational system algorithms are simple or widely known. Therefore, this approach does not require a huge effort for finding algorithms that always generate optimal or almost optimal solutions for the problem.

We have identified several types of interaction that may help interaction-based problem solving. In order to illustrate this approach we are developing a computational system for submarine layout interaction-based problem solving. It is known as SALAS and uses most of the identified interaction types. Next, we describe each type of interaction provided by SALAS.

3.1 Solution Editing and Visualizing

The most elementary way of human-computer interaction consists of a direct manipulation interface that allows the user to create and modify problem solutions. Beyond making it easy to edit a solution, this interface provides a realistic visualization of the problem solution. The visualization allows the user to identify errors and helps the generation of new ideas. SALAS offers this type of interface by providing a environment containing graphical representations of each submarine layout element.

Figure 1 shows the SALAS graphical interface. Each element may be selected, included, removed, modified and moved through the interface in a similar way as

known CAD systems. This figure also shows the graphical representation of the submarine layout elements: wells (circles), manifolds (small rectangle), platforms (large rectangle) and obstacles (multiform polygons).

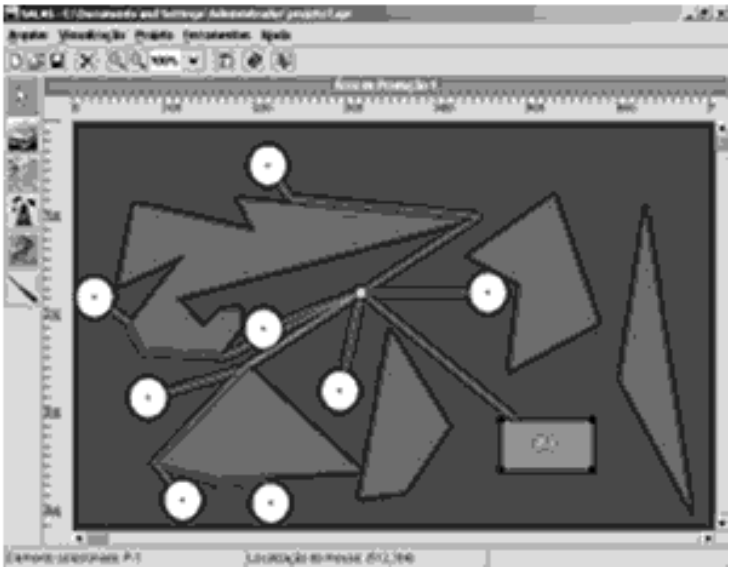

Fig. 1. Visual specification of a SALAS project

3.2 Automatic Calculations

SALAS automatically calculate the costs of partial or full alternatives for a submarine layout solution. The user does not need to request this functionality since it is permanently available in the graphical interface. Direct visualization of the total cost allows the user to immediately perceive the impact of any modification in the layout.

3.3 Automatic Verifications

In this type of interaction the computer verifies the solution consistency. Whenever some inconsistency is found, the computer may show a warning message or undo the action that caused the inconsistency. SALAS implements one example of this type of interaction by avoiding that any element be positioned over an obstacle. After a drag operation, if the dragged element is over an obstacle, the action is not performed.

3.4 Suggestion of Complete Solutions

The human user may specify a submarine layout problem (defining the objectives, wells capacities, obstacles and platforms capacity, size and number) and request the

computational system to produce one or more problem solutions. After SALAS generates a complete solution, the user may adjust it using the graphical interface. Figure 2 shows how the user may request the generation of a solution in SALAS. He must choose the option "Executar Algoritmos" (to execute algorithms in portuguese).

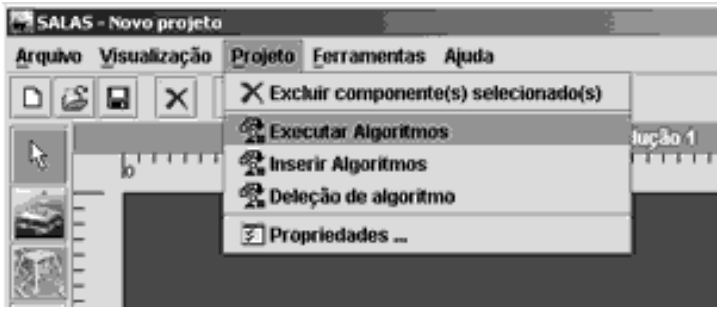

Fig. 2. Requesting algorithm execution

3.5 Suggestion of Partial Solutions

The human user may also specify subproblems of the submarine layout problem and request SALAS to solve them. The idea underneath this type of interaction is to avoid the generation of bad solutions (or no solution at all) when dealing with very complex and large scale combinatorial problems. By allowing the user to subdivide the problem in small scale subproblems, interaction-based problem solving may circumvent combinatorial explosion. The advantage of this type of divide-and-conquer strategy over pure algorithms is the use of the human contextualized and opportunistic knowledge and experience for driving the problem solving search.

SALAS allows the specification of five subproblems primitive types: wells clustering, association of plataforms to well clusters, platforms and wells positioning, manifolds inclusion and pipelines linking. The user may also specify some subproblems combination. For instance, the user may specify a subproblem consisting of wells clustering and plataform association. Furthermore, the user may select any subset of elements, specify a subproblem and request SALAS to generate a solution considering these elements only.

After a partial solution has been generated, the user may also adjust this solution and specify another subproblem to be solved by SALAS algorithms from the modified solution. For example, the user may select a wells subset and request SALAS to make clusters. After SALAS makes the clusters, the user may see that one well does not fit to the cluster chosen by SALAS. Then, the user may change this well to a more appropriate cluster. After making the change, the user may request SALAS to perform association of plataforms to the adjusted well clusters.

3.6 Selection of Library Algorithms

For each primitive subproblem type, SALAS provides some algorithms that may be used for proposing partial solutions for specific subproblems. There are also algorithms and predefined combinations that may be used for proposing complete solutions. The user is responsible for choosing the algorithm to be applied.

The reason for including this type of interaction is for allowing the user to utilize contextualized and opportunistic knowledge regarding the specific problem to be solved for selecting the most appropriate algorithm to be applied to the problem. For instance, if the subproblem alternative space is not large, the user may select a blind search algorithm for finding a guaranteed optimal solution for the subproblem. If this is not the case, the user may select an heuristic algorithm that seems to be more appropriate.

Another feature to be emphasized is the possibility of interrupting an algorithm execution. If an algorithm is running for enough time, the user may use this feature for stopping it. SALAS will return the best solution found until the interruption. Figure 3 (left) shows the SALAS interface for selecting an algorithm from the library.

3.7 Algorithm Parameter Configuration

Some algorithms requires that the user set some parameter values for executing them. If a selected algorithm requires parameter configuration, it will request the user to set the parameter values. Figure 3 (right) illustrates the interface for configuring the parameters of a genetic algorithm used in the SALAS for clustering the wells. Note that the interface requires the user to set the maximum number of iterations, the population size and the crossover and mutation rates.

3.8 Algorithm Configuration

This type of interaction is a little step further than selection of algorithms. It allows the user to dynamically configure algorithms using the subproblems algorithms available in the library. Instead of making a long list of all possible algorithms configurations available in the library, SALAS let the user free for configuring the desired algorithm combination. The same interface used for selecting algorithms (shown in figure 3) may be used for algorithm configuration.

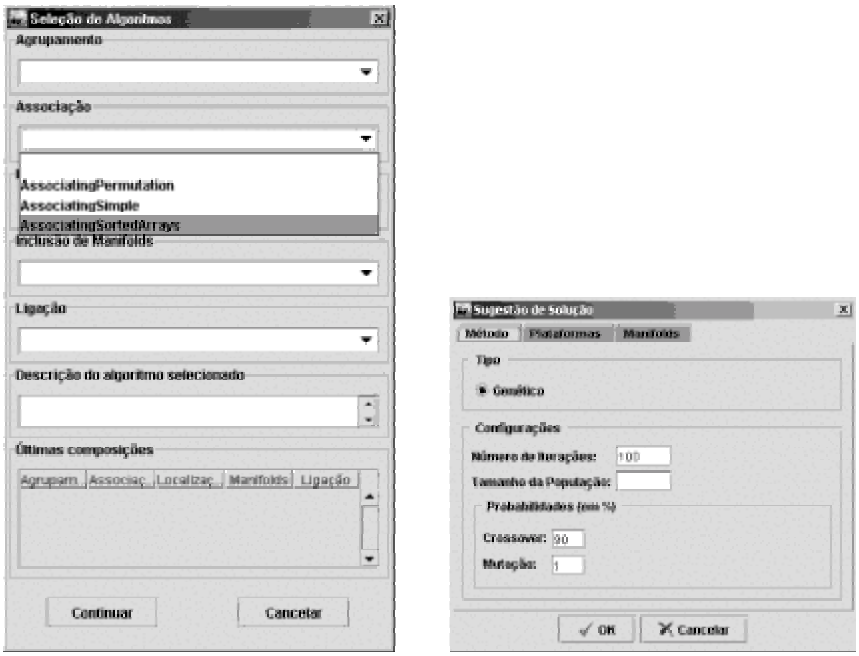

Fig. 3. SALAS Interfaces. Algorithm selection and parameters configuration

3.9 Algorithm Modification, Deletion and Insertion

Users may include, remove and update library algorithms. Whenever the user identifies some algorithm inadequacy to a specific problem or a new algorithm appliable to a subproblem, the user may remove, change or include a library algorithm.

Ideally, the user should be able to include and update algorithms without disrupting the problem solving process. It means that the user should not be forced to logout, make the changes and start the process from where he has stopped. However, this is possible only if the computational environment provides some type of end-user programming language.

SALAS adopts an intermediary approach. The user must logout, but it only needs to change or create the algorithms and compile them. The user does not need to change anything else in SALAS. Next time SALAS is loaded, the algorithms are updated.

4 Experimentation

Aiming to get indications about the hypothesis raised here, i.e., the human-computer partnership can achieve better results than the results obtained exclusively by the hu-

man user or the computational system when solving submarine layout problems, a small scale experiment has been accomplished using SALAS.

A submarine layout problem was specified with an initial distribution of 25 wells, 8 platforms and 5 obstacles in a production area. In order to make it simple, we considered that all the wells and platforms have the same features: the wells have the same production capacity and the same objective size; the platforms have the same dimensions and the same production capacity. Then, the problem was solved according to the three approaches:

1. The user solves the problem by himself; the machine is used in a passive form;
2. The computational system presents complete solutions for the problem;
3. The problem is solved by an active human-computer partnership using interaction-based problem solving.

In the first stage of the experiment, a computer science student solved the problem without being able to use any suggestion feature provided by the SALAS. After that, some combinations of the algorithms available in SALAS have been used to generate complete solutions to the problem. Finally, the same computer science student solved the problem again, at this time using all interaction features made available by SALAS.

4.1 Results

Figure 4 shows the submarine layout problem used by this experiment. The initial layout cost reaches US$ 800.000.000,00, essentially in function of the platform values. Although this problem may be considered a small one, we may notice the difficulty of visually identifying the element clusters.

Table 1 summarizes the results obtained by the experiment. As we may observe, the total solution cost obtained by the user in an isolated form, without any kind of active support from the tool, was U$ 833,977,177.83.

The three best results of different combinations of algorithms cost U$ 827,318,203.06, U$827,728,344.82 and U$ 828,852,646.98, respectively. It is important to emphasize that the interactive solution reduced the total cost of the user solution by US$ 14.305.103,10, which corresponds to a reduction of approximately 42 % in the total cost (since the initial layout cost is fixed for all solutions). In relation to the best algorithm, the reduction obtained was USS 7.646.128,33, corresponding to an approximate reduction of 28 % in the solution cost.

5 Conclusions

This work considers the submarine layout problem over a new perspective, promoting high level types of interaction between the user and the computational system. While computer-aided design (CAD) systems offer well-known and valuable human-computer interaction features allowing the user to find a solution by himself (such as, the edition and realistic visualization of the solution alternatives), these systems do not provide an active support for problem solving (for example, suggestion of solutions and consistency checking).

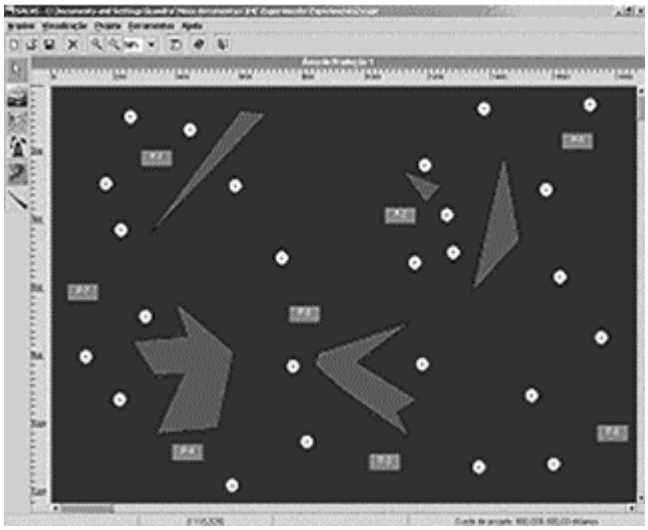

Fig. 4. Visual problem specification

Table 1. Results

Method		Total Cost Design
User		833,977,177.83
Complete Solution	Method 1	827,318,203.06
	Method 2	827,728,344.82
	Method 3	828,852,646.98
Interactive		819,672,074.73

Artificial intelligence or optimization systems attempt to generate problem solutions by itself. However, these approaches have only had moderate success. Sometimes, they are ineffective when it is necessary to scale the problem up. Other times, they cannot be generalized and applied to some specific problem instances or situations.

This work raises the hypothesis that a more effective partnership between the human user and the computational system would achieve better results for submarine layout problems than the approaches where these entities try to solve the problem alone. In order to initially investigate this hypotesis, a small scale experiment was carried out that brought indications reinforcing this hypothesis. The interaction-based problem solving approach achieved better results than the approaches where the user and the computational system tried to solve the problem alone.

We emphasize that the results cannot be considered conclusive since the experiment was very small and simplified. First, we have used on the experiment just one small size problem. Besides being small, this problem was also simplified once all

wells and plataforms had the same features. Moreover, just one problem is not enough for validating our hypothesis. Second, the subject set was restricted to only one computer science student. Therefore, the results might be biased since it is strongly dependent on the student skills. Moreover, the fact that the student knew the results obtained by one approach before using the next approach for solving the problem might also introduced bias.

Another important aspect to be discussed is the required human user background knowledge. We assume that the user is a domain expert (or apprentice) on submarine layout problems. Once they are not computer scientists, they probably don't have computational knowledge for choosing the appropriate algorithms, their combinations and parameters. For enabling the users to benefit from these types of interaction, SALAS also provides algorithm detailed descriptions specifying its features and indicating situations where it could or should be used. Other alternatives are also possible. Knowledge engineers may train the users on using SALAS and its algorithms. We also consider the possibility of having a computational agent suggesting which algorithm (and its features) should be applied in a specific situation. Ideally, this agent should learn from experience. In addition, we definitely expect that the user will learn from its own experience of applying the algorithms to submarine layout problems.

Next step on this research is to accomplish more comprehensive experiments in order to obtain experimental validation for the hypothesis raised here. It is important to randomly select a large number of problem specifications to be used in the new experiment. It is also necessary to select a significant number of subjects for performing the experiment, each one dedicated to one approach of problem solving. Other functionalities will also improve SALAS interaction capabilities.

References

1. Devine, M.D. e Lesso, W.G. Models for the Minimum Cost Development of Offshore Oil Fields, Management Science, 18, pp. 378-387, 1972.
2. Dogru, S. Selection of Optimal Platform Locations, Drilling Engineering, 2, pp. 381-386, 1987.
3. Garcia, A.C.B. e Sichman, J.S. Agentes e Multiagentes, Sistemas Inteligentes – Fundamentos e Aplicações, Editora Manole, pp. 270-306, 2002.
4. Goldbarg, M. C. e Luna, H.P.L. Otimização Combinatória e Programação Linear – Modelos e Algoritmos, Editora Campus, 2001.
5. Hansen, P., Filho, E. L. P. e Ribeiro, C. C. Location and Sizing of Offshore Platforms for Oil Exploration, European Journal of Operational Research, Vol. 58, pp. 202-214, 1992.
6. Hearst, M.A. Trends & Controversies: Mixed-initiative interaction. IEEE Intelligent Systems 14(5): 14-23 (1999)
7. Silva, C. L. e Garcia, A. C. B. SpADD: An Active Design Documentation Framework Extension Applied to Spatial Layout Design Problems, VI IberoAmerican Conference on Artificial Intelligence (IBERAMIA'98), 1998, Lisboa.
8. Wegner, P. Why Interaction is More Powerful than Algorithms, Communications of the ACM, vol.40, no.5, pp.81-91, May, 1997.