AI Planning Search

Artificial Intelligence planning problem consists of finding a sequence of applicable actions that allow a planner to move from an initial state – defined by some state variables in a given representation – to a goal state. This can be done using different planning algorithms which may be based on direct brute-force search of in more advanced heuristic search. The main restrictions in AI planning problems are space and time complexities which depend in general from the data representation and the used planning algorithm.

Nowadays Artificial Intelligence planning is used in a many fields, such as robotics, medicine, aviation, industry etc. [1] It consists of choosing a sequence of actions to be done in a specific order to let the planner move from an initial state, to a goal state. A state is defined by a set of variables' values that can be propositional as in some representations and they are called fluents or atoms, or multivalued called state variables. An action is an operation performed that changes the value of one or more variables, and changes consequently the state, it has pre-conditions and post-conditions that help the planner to choose the right action or actions to be applied at a given state. Since AI planning was introduced in many fields that need a very fast, accurate and efficient decision making and doesn't tolerate significant delays and results looseness (e.g. medicine, military, industries and gaming etc...), for these reasons speed and precision are critical in all AI planners, and they require efficient algorithms, with as low as possible space and time complexities, to be applied on data saved in primary memory. Data cannot be saved in the secondary memory, because despite its bigger capacity, the latency that results when reading from static memory is not tolerable in a planning problem. Normally, these algorithms should be domain independent to work in many situations independently from the problem that needs to be solved. For instance the same algorithm should work in blocks world, logistics, or any other problem giving a right and precise answer within the same complexity concerning time and space. As a consequence, variables should all be saved on main memory, as well as possible actions each with its preconditions and effects, and the planner should keep track of the information of every state he evaluated with the corresponding action. This will make a huge amount of continuously increasing data when dealing with planning problems, and will limit planners to solve relatively small problems. There have been many attempts to decrease the time and space complexity by changing the data representation from the use of straight-forward propositional variables to finite domain state-variables that decrease the size of the state and accelerate access to it in search and update operations [2] .

REPRESENTATIONS OVERVIEW
There exist many data representations, the commonly used ones are STRIPS, PDDL and SAS+ . They can all be used for representing planning problems, but they differ by the form of presenting, accessing, analyzing and updating data, and each algorithm is done based on some specific representation, and optimized to be used with data stored following the accordingly designed representation.

STRIPS In STRIPS representation, the problem is represented by a tuple $\Psi = (P, O, I, G)$

PDDL PDDL (Planning Domain Definition Language) is defined as a standard to represent planning problems, it includes internally the STRIPS representation and adds more features to it, like negated preconditions and specifying object types. It also has conditional effects and safety constraint specifications as well as definition for subactions and subgoals. In its recent version it allows numeric values for variables and Manages many problems in different fields by differing language features subsets.

SAS+ SAS+ is a light modification on STRIPS, but there exist some variations between them. There are two main differences between these representations: on the first hand, SAS+ uses multi-valued state variable to represent facts instead of using propositional fluents, so a number of mutually exclusive propositional atoms can be replaced by one multi-valued variable. So we can describe problems in a natural way, we can also reduce the complexity of some problems solved with restrictions using STRIPS and be able to remove these restrictions, and generalizing state variables to other domains will take smaller steps [3]. On the second hand, for the actions representation, each action has a precondition, post-condition and prevail-condition.

BIT ARRAY BASED REPRESENTATION FOR PLANNING PROBLEMS Problem The major deficiencies in AI planning are the huge space and time consumptions that limit the solvable problems to relatively

restricted and small sized problems. As mentioned in the previous chapter, restrictions must be applied on data representation and manipulation, in order to limit the complexity to a determined level. For instance, the restrictions in SAS+ can affect the number of state variables changed by each operation, the number of values in the domain of each multivalued variable, the number of operators that can lead to an effect and obliging all preconditions not changed after the execution to have the same value [3]. All these restrictions are caused by the high complexity, and once the complexity is better in terms of space and time, this increases the capacity of reaching the solution of a bigger and more complex problem with less and less restrictions depending on the complexity amelioration. There have been many attempts to reduce the complexity, but available enhancements on data representations didn't succeed to reduce these complexities enormously, and they still get close results compared to the representations that they intended to improve. For example, using multivalued discrete state variables instead of fluents in SAS and SAS+ has relatively decreased space and time consumptions compared to STRIPS, but computationally when both representations are used for a planning algorithm, they give relatively close results in terms of space and time complexities. As a result, we still can't solve as big and complex problems as we need in the fields where Artificial Intelligence planning is introduced despite the advanced search algorithms mainly based on heuristics, so here comes the need to do domain dependent algorithm in some cases, but researchers are still looking for a better generic and domain independent algorithm that can solve problems from different domains with no restrictions on a chosen domain. Therefore, space consumption can be decreased by using a light and compact data representation model, and time complexity reduction should be based on improving the way the algorithms accesses, compares and updates data in the given representation.

CONCLUSION

Artificial Intelligence planning problems have severe space consumption and time complexities. The space complexity problem is problematic, and should be reduced to the minimum in order to have the capacity to solve bigger problems, and the time complexity problem is critical, and should also be reduced to keep the maximum delay at runtime tolerable. To decrease space consumption, we used the main advantage of SAS+ over other representations to store data, which is the multivalued aspect of variables, so each state variable will be represented in an array of bits, where each bit represents one value of the domain that this variable could have, as a consequence, each array of bit will contain one or many values set to 1 and the others filled with 0. We have also proved that the bit array based data representation helps decreasing time complexity in accessing data in searching, updating and states comparison in all algorithms that work based on this representation and access data in the same way. Our light weight array based representation follows a successful time and space economic strategy. In terms of comparison it is better, faster and easier; since comparing bits is much simpler than comparing Strings. This method, by the use of the array structure, enhances the speed of search and update by the use of indexes pointing to 1 bit. Experimental results have shown the advantage of the new method in finding, comparing and updating results, what brings its advantage over other data access methods used in other planner that have to match strings. String matching complexity is at least linear; data access and comparison complexity through the index is constant. For the new researchers who would like to enter this field, there is a lot of perspective work: first there is need to implement advanced algorithms and used some good heuristics based on this new representation, like FF , fast downward and others.

REFERENCES

[1] S. S. Shukla and V. Jaiswal, 2013, "Applicability of Artificial Intelligence in Different Fields of Life," International Journal of Scientific Engineering and Research (IJSER), pp. 28-35.
[2] M. Helmert, 2009, "Concise finite-domain representations for PDDL planning tasks," Artificial Intelligence 173, p. 503–535.
[3] C. Bäckström and B. Nebel, 1995, "Complexity Results for SAS+ Planning" Computational Intelligence, Volume 11, Issue 4, pp. 625–655.
[4] B. Bonet and H. Geffner, 2001, "Planning as heuristic search," Artificial Intelligence 129, p. 5–33. [5] J. Hoffman and B. Nebel, 2001, "The FF Planning System: Fast Plan Generation Through Heuristic Search," Journal of Artificial Intelligence Research, 14, pp. 253- 302.