History of Artificial Intelligience Research Paper

Fast planning through planning graph analysis

Graphplan was introduced in 1997 as a new approach to planning. Graphplan will always return the shortest plan or report if no such plan exists. Also, Graphplan can be built in polynomial time. However, Graphplan can consume a lot of memory. Graphplan creates Planning Graph and uses it to search for a plan. The search is the combination of total-order and partial-order planners. Another strong point of Graphplan is that the order of the goals does not matter much. Planning Graph's operators have preconditions, add-effects and delete-effects and parameters. No objects will be created or deleted by the operators, and time can be represented individually.

For a plan to be considered valid, or Valid Plan, it must have a list of actions and a time that limits those actions. Many actions can be executed at the same time as long as they are not interfering with each other. Actions of an operator that count as interfering are those who change the preconditions and effect of another operator. If the actions are not interfering with each other, they can be ordered in any fashion without changing the outcome.

Planning Graph is similar to Valid Plan. However, it does not have to avoid any interfering actions. Planning Graph is a direct, leved-graph and it has two types of nodes and three types of edges. There are action levels which contain action nodes and proposition levels which contain proposition nodes. At the root of the graph is a proposition level with only one proposition node. As the level increases, it switches between propositions true and proposition actions.

Edge in Planning Graph is used to show the relationship between actions and propositions. One end of action nodes is connected to preconditions in proposition level i, while the others are connected to their Add-Effects in proposition level i+1 and their Delete-Effects in proposition level i+1.

Planning Graph has less condition than Valid Plan. There is no requirement for independence. Therefore propositions can appear at level i and level i+1 if there are actions that are Add-Effect or Delete-Effect. Two actions in Planning Graph are mutually exclusive if no valid plan to contain both of them or to make them true. The same applies to two proposition. Graphplan stores exclusive relationships by applying a few rules to the Planning Graph. One of them is Interference that was mentioned above, and the other one is Competing Needs. Competing Needs is the precondition of one action is exclusive to another action.

Graphplan algorithm is starting at a single proposition contains an Initial Conditions. In stage i, Graphplan uses the Planning Graph from stage i-1 to created an extended version of the graph. This helps Graphplan to either find a solution or report if no solution exists. Graphplan also keeps a record of plans that are unsolvable.

CIRCA: The Cooperative Intelligent Real-time Control Architecture

Real-time computing and Artificial Intelligent are two opposing field because of their differences in using resources and predictability. In order to compensate that, CIRCA performs high level performing tasks while create low-control plans for the real-time subsystems. While other systems can only guaranty that they will try their best and may not ahive task goals, CIRCA aims to achieve AI's task goals within the limited resources like memory and time.

To achive these goals, CIRCA created a new interative improvement algorithm, graph-based world model, and a reactive plan to isolate AI subsystem from domain deadline to keep control agents safe. The result is CIRCA was able to trade between performance and resources when resources are available and not available while doing best-effort algorithms.

Mechanical Generation of Admissible Heuristics

The goal of heuristic search is to find the most efficient path from start to goal. While some problems can be solved using algorithms like Dijkstra's algorithm, AI problems are often too large and complicated to be solved that way. However, if the search algorithm is given extra information related the problem, the problem can be solved quickly. A heuristic function is a lower bound of the true cost of reaching the goals.

Distance of a given state space can be estimated in many ways. One way to do so is relaxation. This focus on the idea of simplifying the problem by removing constrains. It ends up making the path between start state and the goal state a line, which is good for estimating.

Citation

Blum, Avrim L., and Merrick L. Furst. "Fast planning through planning graph analysis." Artificial Intelligence, vol. 90, no. 1-2, 1997, pp. 281–300., doi:10.1016/s0004-3702(96)00047-1.

Musliner, D., Durfee, E., & Shin, K. (1993). CIRCA: a cooperative intelligent real-time control architecture. IEEE Transactions on Systems, Man, and Cybernetics, 23(6), 1561-1574. doi:10.1109/21.257754

Holte, R., Schaeffer, J., & Felner, A. (n.d.). Mechanical Generation of Admissible Heuristics.