Case Study (Human Robot Chess Play)

Syntax Analysis

Divya Kundra and Ashish Sureka, *An Experience Report on Teaching Compiler Design Concepts using Case-Based and Project-Based Learning Approaches*, International Conference on Technology for Education (T4E 2016)

GOLEMS is Humanoid Robotics Lab at Georgia Institute of Technology. The lab works towards developing robots having human and even super human capabilities. The current task that the lab is working on is building a physical human-robot chess. One side of the chess would have a movable robot arm with sensors providing suitable force to locate, pick, drop and rotate the chess piece's while on other side would be the human playing against the robot. The requirement is to design the control mechanism of the robot in such a way that it imitates a human playing the game.

As robots come into increasing contact with humans, it is absolutely vital to prove that these potentially dangerous machines are safe and reliable. Furthermore, applying robots to increasingly complicated and varied tasks requires a tractable way to generate desired behaviors. Typical strategies do not provide a way to prove how the system will respond during complicated tasks with uncertain outcomes. Existing planners deliberately simplify the control system or have prohibitive computational cost. Thus developers at GOLEMS have found a new way to represent and verify the controlling mechanisms of robots- using Context Free Grammars (CFG). The production rules of the grammar represent a task decomposition of robot behavior. The name that they have given to the grammars controlling robots is Motion Grammar (MG). They plan to use this grammar in uncertain environments with guarantees on correctness and completeness. The developers feel that there are several reasons to use Context Free Grammar than regular languages. While prior language methods have focused on finite-state Regular languages, a broader class of system behavior can be described using the Context-Free language class. The Context-Free set provides more descriptive power while maintaining the efficiency and verifiability of regular languages. In addition, Context-Free Grammars provide a natural representation for hierarchies in the system. An example CFG and parse tree are given in Fig. 1 for a loading and unloading task. In production (1), the system will repeatedly perform [load] operations until receiving a [full] token from production (2). Then the system will perform [unload] operations of the same number as the prior [load] operations. This simple use of memory is possible with Context-Free systems. Regular systems are not powerful enough.

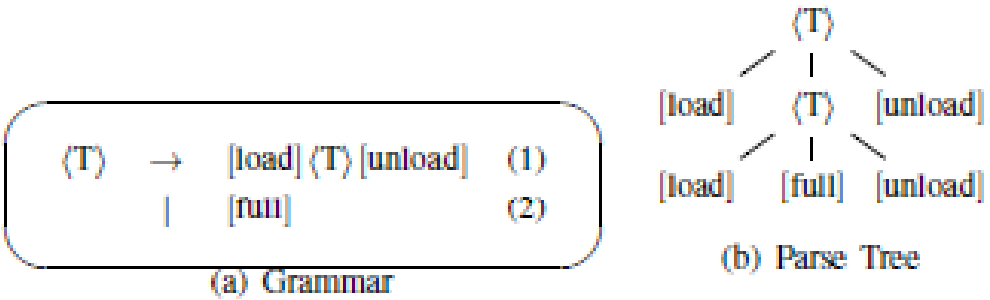

Fig. 1 : CFG and parse tree for loading and unloading task

Also the safety of the robot is essential for the developers as poor safety would impose physical costs resulting in complaints from the clients. Imagine a robot searching for earthquake survivors. This robot must carefully remove pieces of rubble while ensuring the larger structures does not collapse. Using software to handle robot's dynamics presents challenges due to the general case inability to guarantee software performance. So what the developers thought the solution to be is to address this difficulty using formal language models to syntactically define the system. They thought of describing a broader class of system behavior using Context Free language class. Developers are trying to find analogy of Formal Language with the control mechanism of robots. Each path, or language string, is nothing but a sequence of abstract symbols representing relevant events, predicates, states, or actions. When this system language is parsed online, it defines a control policy enabling response to unpredictable events. Thus Motion Grammar represents a policy for the task which is parsed in real-time based on perceptual input. Developers also need to device the strategy to implement Motion Grammar. With the Motion Grammar, they have the same freedom to designate components between what is plant and what is the controller in whatever way is most convenient to the design of the overall system.

The main task of the developers is to identify various challenges that will come in design of system of robot human chess play and address those challenges by building the suitable grammar. Developers need to decide what actually will be the tokens in the system under consideration. Tokens are the terminal symbols of the language, which we use to model discrete elements of the system. For example, there can a token to indicate a winning position on the chessboard. These may be regions in which the underlying dynamics of the system change, for example a position where contact is made with another object. Number of tokens needed are equal to the number of events that cause a discrete change in the system. The Motion Parser reads in tokens and chooses the appropriate production from the grammar to expand and execute. The functionalities that are required in the human robot chess game is to have a guarded move i.e. use the Motion Grammar to allow human and robot to safely operate in same workspace. The grammar should make the provision that the robot must respond immediately to the dangerous

situation of impact with the human and it should re attempt a move only after the human removes his hands from the pieces. Then the grammar is also to be built to set fallen pieces upright. The robot should pick a fallen piece at location x, moves its arm to location x, grasp it, lift the piece sufficiently high above the ground and rotate it so that it can be replaced upright. Finally it should release the grasp on the piece setting it upright. The Motion Grammar build would consists of alternating sequences of robot moving, followed by human moving, waiting until the game has ended via checkmate, resignation or draw. When it is robot's turn it will correct any fallen pieces, make it move and then correct any pieces that may have fallen while it was making the move. Making a move, can be either a simple move between squares, a castle, a draw or resignation etc. A simple piece move, requires first grasping the piece, then placing it on the correct square. To grasp the piece, the robot will move its hand around the piece then tighten its grip, until there is sufficient pressure registered on the touch sensors. To capture a piece, the robot will remove the captured piece from the board, and then move the capturing piece onto that square. To resign – indicating a failure in chess strategy, not motion planning – the robot moves it through the square occupied by the king, knocking it over. Thus for the challenges discussed here along with identify other challenges that will rise in designing the system. Propose the suitable Motion Grammar for the required system.